Specification Volume 0

SPECIFICATION OF THE *BLUETOOTH* SYSTEM

Experience More



Master Table of Contents & Compliance Requirements

Covered Core Package version: 4.0
Current Master TOC
Publication date: 30 June 2010

Ford Ex. 1052

Revision History

The Revision History is shown in the “Appendix” on page 55[Vol. 0].

Contributors

The persons who contributed to this specification are listed in the Appendix.

Web Site

This specification can also be found on the official Bluetooth web site:
http://www.bluetooth.com

Disclaimer and Copyright Notice

The copyright in these specifications is owned by the Promoter Members of Bluetooth SIG, Inc. (“Bluetooth SIG”). Use of these specifications and any related intellectual property (collectively, the “Specification”), is governed by the Promoters Membership Agreement among the Promoter Members and Bluetooth SIG (the “Promoters Agreement”), certain membership agreements between Bluetooth SIG and its Adopter and Associate Members (the “Membership Agreements”) and the Bluetooth Specification Early Adopters Agreements (“1.2 Early Adopters Agreements”) among Early Adopter members of the unincorporated Bluetooth special interest group and the Promoter Members (the “Early Adopters Agreement”). Certain rights and obligations of the Promoter Members under the Early Adopters Agreements have been assigned to Bluetooth SIG by the Promoter Members.

Use of the Specification by anyone who is not a member of Bluetooth SIG or a party to an Early Adopters Agreement (each such person or party, a “Member”), is prohibited. The legal rights and obligations of each Member are governed by their applicable Membership Agreement, Early Adopters Agreement or Promoters Agreement. No license, express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

Any use of the Specification not in compliance with the terms of the applicable Membership Agreement, Early Adopters Agreement or Promoters Agreement is prohibited and any such prohibited use may result in termination of the applicable Membership Agreement or Early Adopters Agreement and other liability permitted by the applicable agreement or by applicable law to Bluetooth SIG or any of its members for patent, copyright and/or trademark infringement.

THE SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, SATISFACTORY QUALITY, OR REASONABLE SKILL OR CARE, OR ANY WARRANTY ARISING OUT OF ANY COURSE OF DEALING, USAGE, TRADE PRACTICE, PROPOSAL, SPECIFICATION OR SAMPLE.

Each Member hereby acknowledges that products equipped with the Bluetooth® technology (“Bluetooth® Products”) may be subject to various regulatory controls under the laws and regulations of various governments worldwide. Such laws and regulatory controls may govern, among other things, the combination, operation, use, implementation and distribution of Bluetooth® Products. Examples of such laws and regulatory controls include, but are not limited to, airline regulatory controls, telecommunications regulations, technology transfer controls and health and safety regulations. Each Member is solely responsible for the compliance by their Bluetooth® Products with any such laws and regulations and for obtaining any and all required authorizations, permits, or licenses for their Bluetooth® Products related to such regulations within the applicable jurisdictions. Each Member acknowledges that nothing in the Specification provides any information or assistance in connection with securing such compliance, authorizations or licenses. NOTHING IN THE SPECIFICATION CREATES ANY WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING SUCH LAWS OR REGULATIONS.

ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHTS OR FOR NONCOMPLIANCE WITH LAWS, RELATING TO USE OF THE SPECIFICATION IS EXPRESSLY DISCLAIMED. BY USE OF THE SPECIFICATION, EACH MEMBER EXPRESSLY WAIVES ANY CLAIM AGAINST BLUETOOTH SIG AND ITS PROMOTER MEMBERS RELATED TO USE OF THE SPECIFICATION.

Bluetooth SIG reserves the right to adopt any changes or alterations to the Specification as it deems necessary or appropriate.

Copyright © 1999 - 2010

Ericsson AB,
Lenovo,
Intel Corporation,
Microsoft Corporation,
Motorola, Inc.,
Nokia Corporation,
Toshiba Corporation

*Third-party brands and names are the property of their respective owners.

MASTER TABLE OF CONTENTS

This table of contents (TOC) covers the entire Bluetooth Specification.
In addition each volume has a TOC and each part of a volume is preceded by a detailed TOC.

Ford Ex. 1052

Ford Ex. 1052

THE BLUETOOTH SPECIFICATION MASTER TABLE OF CONTENTS

In the following Master Table of Contents:

- The TOC for each Volume starts at the top of a page.
- The Volume number in black is followed by the Volume name in red.

Note: Each Volume is a self contained book which is published and updated separately and is equipped with a TOC of its own. However, this Master TOC is also revised as soon as any of the other Volumes are updated.

- A Volume contains one or more Parts (A, B, etc.); each Part can be viewed independently and has its own TOC.

Red or blue text on the following pages indicates hypertext links that take you directly to the indicated section, on condition that you have access to a complete specification.

Specification Volume 0
Master Table of Contents & Compliance Requirements

Part A
MASTER TABLE OF CONTENTS

Part B
BLUETOOTH COMPLIANCE REQUIREMENTS

1 Introduction 43
2 Scope 44
3 Definitions 45
3.1 Types of Bluetooth Products 45
4 Core Configurations 50
4.1 Basic Rate Core Configuration 50
4.2 Enhanced Data Rate Core Configurations 51
4.3 High Speed Core Configuration 52
4.4 Low Energy Core Configuration 52
4.5 Basic Rate and Low Energy Combined Core Configuration 53
4.6 Host Controller Interface Core Configuration 54

Part C
APPENDIX

1 Revision History 59
1.1 [Vol 0] Master TOC & Compliance Requirements 59
1.2 [Vol 1] Architecture & Terminology Overview 59
1.3 [Vols 2, 3, & 6] Core System Package 61
1.4 [Vol 4] Transport Layers 64
2 Contributors 65
2.1 [Vol 0] Master TOC & Compliance Requirements 65
2.2 [Vol 1] Architecture &Terminology Overview 65
2.3 [Vol 2] Core System Package, Controller 69
2.4 [Vol 3] Core System Package, Host 87
2.5 [Vol 4] Host Controller Interface [Transport Layer] 98
2.6 [Vol 5] Core System Package [AMP Controller volume] 100
2.7 [Vol 6] Low Energy Specification 101

Specification Volume 1
Architecture & Terminology Overview

Part A
ARCHITECTURE

1 General Description 17
1.1 Overview of BR/EDR Operation 18
1.2 Overview Of Bluetooth Low Energy Operation 20
1.3 Overview Of AMP Operation 22
1.4 Nomenclature 24

2 Core System Architecture 30
2.1 Core Architectural Blocks 34

3 Data Transport Architecture 39
3.1 Core Traffic Bearers 40
3.2 Transport Architecture Entities 45
3.3 Physical Channels 49
3.4 Physical Links 58
3.5 Logical Links and Logical Transports 61
3.6 L2CAP Channels 72

4 Communication Topology and Operation 73
4.1 Piconet Topology 73
4.2 Operational Procedures and Modes 76

5 Security Overview 85
5.1 BR/EDR Secure Simple Pairing 85
5.2 LE Security 89
5.3 AMP Security 91

6 Bluetooth Application Architecture 93
6.1 Bluetooth Profiles 93
6.2 Generic Access Profile 93
6.3 Profile Hierarchy 94
6.4 Generic Attribute Profile 95
6.5 GATT-based Profile Hierarchy 96

Part B
ACRONYMS & ABBREVIATIONS

1 List of Acronyms and Abbreviations 101

Part C
CORE SPECIFICATION CHANGE HISTORY

1 Deprecated Features 116

2 Changes from V1.1 to V1.2 117

2.1 New Features 117
2.2 Structure Changes 117
2.3 Deprecated Features list 117
2.4 Changes in Wording 118
2.5 Nomenclature Changes 118

3 Changes from V1.2 to V2.0 + EDR 119

3.1 New Features 119
3.2 Deprecated Features 119

4 Changes from V2.0 + EDR to V2.1 + EDR 120

4.1 New features 120
4.2 Deprecated Features 120

5 Changes From V2.1 + EDR To V3.0 + HS 121

5.1 New Features 121
5.2 Deprecated Features 121

6 Changes From V3.0 + HS To v4.0 122

6.1 New Features 122
6.2 Deprecated Features 122

Part D
MIXING OF SPECIFICATION VERSIONS

1 Mixing of Specification Versions 126

1.1 Features and their Types 127
1.2 Core Specification Addendums 128

Part E
IEEE LANGUAGE

1 Use of IEEE Language 135

1.1 Shall 135
1.2 Must 136
1.3 Will 136
1.4 Should 136
1.5 May 136
1.6 Can 137

Specification Volume 2
Core System Package
[BR/EDR Controller volume]

Part A
RADIO SPECIFICATION

1 Scope 33
2 Frequency Bands and Channel Arrangement 35
3 Transmitter Characteristics 36
3.1 Basic Rate 37
3.2 Enhanced Data Rate 39
4 Receiver Characteristics 46
4.1 Basic Rate 46
4.2 Enhanced Data Rate 48
5 Appendix A 51
5.1 Nominal Test Conditions 51
5.2 Extreme Test Conditions 52
6 Appendix B 53
7 Appendix C 54
7.1 Enhanced Data Rate Modulation Accuracy 54

Part B
BASEBAND SPECIFICATION

1 General Description 65
1.1 Bluetooth Clock 66
1.2 Bluetooth Device Addressing 68
1.3 Access Codes 69
2 Physical Channels 70
2.1 Physical Channel Definition 71
2.2 Basic Piconet Physical Channel 71
2.3 Adapted Piconet Physical Channel 76
2.4 Page Scan Physical Channel 77
2.5 Inquiry Scan Physical Channel 81
2.6 Hop Selection 83
3 Physical Links 96
3.1 Link Supervision 96
4 Logical Transports 97
4.1 General 97
4.2 Logical Transport Address (LT_ADDR) 97

4.3 Synchronous Logical Transports 98
4.4 Asynchronous Logical Transport 98
4.5 Transmit/Receive Routines 99
4.6 Active Slave Broadcast Transport 104
4.7 Parked Slave Broadcast Transport 105

5 Logical Links 106
5.1 Link Control Logical Link (LC) 106
5.2 ACL Control Logical Link (ACL-C) 106
5.3 User Asynchronous/Isochronous Logical Link (ACL-U) 106
5.4 User Synchronous Data Logical Link (SCO-S) 107
5.5 User Extended Synchronous Data Logical Link (eSCO-S) 107
5.6 Logical Link Priorities 107

6 Packets 108
6.1 General Format 108
6.2 Bit Ordering 109
6.3 Access Code 110
6.4 Packet Header 115
6.5 Packet Types 117
6.6 Payload Format 127
6.7 Packet Summary 132

7 Bitstream Processing 134
7.1 Error Checking 135
7.2 Data Whitening 138
7.3 Error Correction 139
7.4 FEC Code: Rate 1/3 139
7.5 FEC Code: Rate 2/3 140
7.6 ARQ Scheme 141
7.7 Erroneous Synchronous Data Reporting 149

8 Link Controller Operation 150
8.1 Overview of States 150
8.2 Standby State 151
8.3 Connection Establishment Substates 151
8.4 Device Discovery Substates 160
8.5 Connection State 165
8.6 Active Mode 166
8.7 Sniff Mode 180
8.8 Hold Mode 183
8.9 Park State 184

9 Audio 192
9.1 LOG PCM CODEC 192
9.2 CVSD CODEC 192

9.3 Error Handling 195
9.4 General Audio Requirements 195
10 List of Figures 196
11 List of Tables 199

Part C
LINK MANAGER PROTOCOL SPECIFICATION

1 Introduction 211
2 General Rules 212
2.1 Message Transport 212
2.2 Synchronization 212
2.3 Packet Format 213
2.4 Transactions 214
2.5 Error Handling 215
2.6 Procedure Rules 216
2.7 General Response Messages 217
2.8 LMP Message Constraints 217
3 Device Features 218
3.1 General Description 218
3.2 Feature Definitions 218
3.3 Feature Mask Definition 225
3.4 Link Manager Interoperability policy 227
4 Procedure Rules 228
4.1 Connection Control 228
4.2 Security 247
4.3 Informational Requests 275
4.4 Role Switch 281
4.5 Modes of Operation 284
4.6 Logical Transports 297
4.7 Test Mode 305
5 Summary 310
5.1 PDU Summary 310
5.2 Parameter Definitions 320
5.3 LMP Encapsulated 331
5.4 Default Values 332
6 List of Figures 333
7 List of Tables 337

Part D
ERROR CODES

1 Overview of Error Codes 343
1.1 Usage Descriptions 343
1.2 HCI Command Errors 343
1.3 List of Error Codes 344
2 Error Code Descriptions 347
2.1 Unknown HCI Command (0X01) 347
2.2 Unknown Connection Identifier (0X02) 347
2.3 Hardware Failure (0X03) 347
2.4 Page Timeout (0X04) 347
2.5 Authentication Failure (0X05) 347
2.6 PIN or key Missing (0X06) 347
2.7 Memory Capacity Exceeded (0X07) 347
2.8 Connection Timeout (0X08) 348
2.9 Connection Limit Exceeded (0X09) 348
2.10 Synchronous Connection Limit to a Device Exceeded (0X0A) 348
2.11 ACL Connection Already Exists (0X0B) 348
2.12 Command Disallowed (0X0C) 348
2.13 Connection Rejected due to Limited Resources (0X0D) 348
2.14 Connection Rejected due to Security Reasons (0X0E) 348
2.15 Connection Rejected due to Unacceptable BD_ADDR (0X0F) 349
2.16 Connection Accept Timeout Exceeded (0X10) 349
2.17 Unsupported Feature or Parameter Value (0X11) 349
2.18 Invalid HCI Command Parameters (0X12) 349
2.19 Remote User Terminated Connection (0X13) 349
2.20 Remote Device Terminated Connection due to Low Resources (0X14) 350
2.21 Remote Device Terminated Connection due to Power Off (0X15) 350
2.22 Connection Terminated by Local Host (0X16) 350
2.23 Repeated Attempts (0X17) 350
2.24 Pairing not Allowed (0X18) 350
2.25 Unknown LMP PDU (0X19) 350
2.26 Unsupported Remote Feature / Unsupported LMP Feature (0X1A) 350
2.27 SCO Offset Rejected (0X1B) 350
2.28 SCO Interval Rejected (0X1C) 351
2.29 SCO Air Mode Rejected (0X1D) 351
2.30 Invalid LMP Parameters (0X1E) 351
2.31 Unspecified Error (0X1F) 351
2.32 Unsupported LMP Parameter Value (0X20) 351
2.33 Role Change Not Allowed (0X21) 351
2.34 LMP Response Timeout / LL Response Timeout (0X22) 351
2.35 LMP Error Transaction Collision (0X23) 352

2.36 LMP PDU Not Allowed (0X24) 352
2.37 Encryption Mode Not Acceptable (0X25) 352
2.38 Link Key cannot be Changed (0X26) 352
2.39 Requested QoS Not Supported (0X27) 352
2.40 Instant Passed (0X28) 352
2.41 Pairing with Unit Key Not Supported (0X29) 352
2.42 Different Transaction Collision (0x2A) 352
2.43 QoS Unacceptable Parameter (0X2C) 352
2.44 QoS Rejected (0X2D) 353
2.45 Channel Assessment Not Supported (0X2E) 353
2.46 Insufficient Security (0X2F) 353
2.47 Parameter out of Mandatory Range (0X30) 353
2.48 Role Switch Pending (0X32) 353
2.49 Reserved Slot Violation (0X34) 353
2.50 Role Switch Failed (0X35) 353
2.51 Extended Inquiry Response Too Large (0x36) 353
2.52 Simple Pairing Not Supported By Host (0X37) 354
2.53 Host Busy–Pairing(0X38) 354
2.54 Connection Rejected Due To No Suitable Channel Found (0X39) 354
2.55 Controller Busy (0X3A) 354
2.56 Unacceptable Connection Interval (0X3B) 354
2.57 Directed Advertising Timeout (0X3C) 354
2.58 Connection Terminated Due To MIC Failure (0X3D) 354
2.59 Connection Failed To Be Established (0X3E) 355
2.60 MAC Connection Failed (0x3F) 355

Part E
HOST CONTROLLER INTERFACE FUNCTIONAL SPECIFICATION

1 Introduction 369
1.1 Lower Layers of the Bluetooth Software Stack 370

2 Overview of Host Controller Transport Layer 372
2.1 Host Controller Transport Layer and AMPS 372

3 Overview of Commands and Events 373
3.1 Generic Events 374
3.2 Device Setup 374
3.3 Controller Flow Control 375
3.4 Controller Information 376
3.5 Controller Configuration 377
3.6 Device Discovery 380
3.7 Connection Setup 382
3.8 Remote Information 387

3.9 Synchronous Connections 388
3.10 Connection State 389
3.11 Piconet Structure 392
3.12 Quality of Service 393
3.13 Physical Links 395
3.14 Host Flow Control 397
3.15 Link Information 398
3.16 Authentication and Encryption 400
3.17 Testing 405
3.18 Alphabetical List of Commands and Events 408
3.19 LE Controller Requirements 415

4 HCI Flow Control 418
4.1 Host to Controller Data Flow Control 418
4.2 Controller to Host Data Flow Control 421
4.3 Disconnection Behavior 421
4.4 Command Flow Control 422
4.5 Command Error Handling 422

5 HCI Data Formats 424
5.1 Introduction 424
5.2 Data and Parameter Formats 424
5.3 Handles 425
5.4 Exchange of HCI-Specific Information 427

6 HCI Configuration Parameters 435
6.1 Scan Enable 435
6.2 Inquiry Scan Interval 435
6.3 Inquiry Scan Window 436
6.4 Inquiry Scan Type 436
6.5 Inquiry Mode 436
6.6 Page Timeout 437
6.7 Connection Accept Timeout 437
6.8 Page Scan Interval 438
6.9 Page Scan Window 438
6.10 Page Scan Period Mode (Deprecated) 438
6.11 Page Scan Type 439
6.12 Voice Setting 439
6.13 PIN Type 440
6.14 Link Key 440
6.15 Failed Contact Counter 440
6.16 Authentication Enable 441
6.17 Hold Mode Activity 441
6.18 Link Policy Settings 443
6.19 Flush Timeout 444

6.20 Num Broadcast Retransmissions 444
6.21 Link Supervision Timeout 445
6.22 Synchronous Flow Control Enable 445
6.23 Local Name 446
6.24 Extended Inquiry Response 446
6.25 Erroneous Data Reporting 446
6.26 Class Of Device 447
6.27 Supported Commands 447
6.28 Logical Link Accept Timeout 454
6.29 Location Domain Aware 455
6.30 Location Domain 455
6.31 Location Domain Options 455
6.32 Location Options 456
6.33 Flow Control Mode 456
6.34 LE Supported Host 457
6.35 Simultaneous LE Host 457

7 HCI Commands and Events 458
7.1 Link Control Commands 458
7.2 Link Policy Commands 535
7.3 Controller & Baseband Commands 560
7.4 Informational Parameters 670
7.5 Status Parameters 680
7.6 Testing Commands 704
7.7 Events 715
7.8 LE Controller Commands 806

8 List of Figures 857

9 List of Tables 858

10 Appendix A: Deprecated Commands, Events and Configuration Parameters 859
10.1 Read Page Scan Mode Command 860
10.2 Write Page Scan Mode Command 861
10.3 Read Page Scan Period Mode Command 862
10.4 Write Page Scan Period Mode Command 863
10.5 Add SCO Connection Command 864
10.6 Page Scan Mode Change Event 866
10.7 Read Country Code Command 867
10.8 Read Encryption Mode Command 868
10.9 Write Encryption Mode Command 868
10.10 Deprecated Parameters 869

Part F
MESSAGE SEQUENCE CHARTS

1 Introduction 877
1.1 Notation 877
1.2 Flow of Control 878
1.3 Example MSC 878
2 Services Without Connection Request 879
2.1 Remote Name Request 879
2.2 One-time Inquiry 881
2.3 Periodic Inquiry 883
3 ACL Connection Establishment and Detachment 885
3.1 Connection Setup 886
4 Optional Activities After ACL Connection Establishment 893
4.1 Authentication Requested 893
4.2 Simple Pairing Message Sequence Charts 894
4.3 Link Supervision Timeout Changed Event 907
4.4 Set Connection Encryption 908
4.5 Change Connection Link Key 909
4.6 Change Connection Link Key with Encryption Pause and Resume 909
4.7 Master Link Key 911
4.8 Read Remote Supported Features 913
4.9 Read Remote Extended Features 913
4.10 Read Clock Offset 914
4.11 Role Switch on an Encrypted Link using Encryption Pause and Resume 914
4.12 Refreshing Encryption Keys 915
4.13 Read Remote Version Information 916
4.14 QOS Setup 916
4.15 Switch Role 917
4.16 AMP Physical Link Creation and Disconnect 918
4.17 AMP Test Mode Sequence Charts 925
5 Synchronous Connection Establishment and Detachment 929
5.1 Synchronous Connection Setup 929
6 Sniff, Hold and Park 934
6.1 Sniff Mode 934
6.2 Hold Mode 935
6.3 Park State 937
7 Buffer Management, Flow Control 940
8 Loopback Mode 942
8.1 Local Loopback Mode 942
8.2 Remote Loopback Mode 944
9 List of Figures 946

Part G
SAMPLE DATA

1 Encryption Sample Data 953
1.1 Generating Kc' from Kc, ... 953
1.2 First Set of Sample Data 956
1.3 Second Set of Sample Data 964
1.4 Third Set of Samples 972
1.5 Fourth Set of Samples 980

2 Frequency Hopping Sample Data 988
2.1 First set 989
2.2 Second set 995
2.3 Third set 1001

3 Access Code Sample Data 1007

4 HEC and Packet Header Sample Data 1010

5 CRC Sample Data 1011

6 Complete Sample Packets 1012
6.1 Example of DH1 Packet 1012
6.2 Example of DM1 Packet 1013

7 Simple Pairing Sample Data 1014
7.1 P-192 Sample Data 1014
7.2 Hash Functions Sample Data 1015

8 Whitening Sequence Sample Data 1026

9 FEC Sample Data 1029

10 Encryption Key Sample Data 1030
10.1 Four Tests of E1 1030
10.2 Four Tests of E21 1035
10.3 Three Tests of E22 1037
10.4 Tests of E22 With Pin Augmenting 1039
10.5 Four Tests of E3 1049

Part H
SECURITY SPECIFICATION

1 Security Overview 1059
1.1 Pausing Encryption and Role Switch 1060
1.2 Change Connection Link Keys 1060
1.3 Periodically Refreshing Encryption Keys 1060

2 Random Number Generation 1061

3 Key Management 1063
3.1 Key Types 1063

3.2 Key Generation and Initialization 1065
4 Encryption 1072
4.1 Encryption Key Size Negotiation 1073
4.2 Encryption of Broadcast Messages 1073
4.3 Encryption Concept 1074
4.4 Encryption Algorithm 1075
4.5 LFSR Initialization 1078
4.6 Key Stream Sequence 1081
5 Authentication 1082
5.1 Repeated Attempts 1084
6 The Authentication And Key-Generating Functions 1085
6.1 The Authentication Function E1 1085
6.2 The Functions Ar and A'r 1087
6.3 E2-Key Generation Function for Authentication 1089
6.4 E3-Key Generation Function for Encryption 1091
7 Secure Simple Pairing 1092
7.1 Phase 1: Public Key Exchange 1093
7.2 Phase 2: Authentication Stage 1 1094
7.3 Phase 3: Authentication Stage 2 1100
7.4 Phase 4: Link Key Calculation 1101
7.5 Phase 5: LMP Authentication and Encryption 1101
7.6 Elliptic Curve Definition 1101
7.7 Cryptographic Function Definitions 1102
8 AMP Security 1108
8.1 Creation of the Initial Generic AMP Link Key 1108
8.2 Creation of Dedicated AMP Link Keys 1108
8.3 Debug Considerations 1109
9 List of Figures 1111

Specification Volume 3
Core System Package
[Host volume]

Part A
LOGICAL LINK CONTROL AND ADAPTATION PROTOCOL SPECIFICATION

1 Introduction 30
1.1 L2CAP Features 30
1.2 Assumptions 33
1.3 Scope 34
1.4 Terminology 34

2 General Operation 38
2.1 Channel Identifiers 38
2.2 Operation Between Devices 38
2.3 Operation Between Layers 40
2.4 Modes of Operation 41
2.5 Mapping Channels to Logical Links 43

3 Data Packet Format 44
3.1 Connection-oriented Channels in Basic L2CAP Mode 44
3.2 Connectionless Data Channel in Basic L2CAP Mode 45
3.3 Connection-oriented Channel in Retransmission/Flow Control/ Streaming Modes 45

4 Signaling Packet Formats 54
4.1 Command Reject (code 0x01) 56
4.2 Connection Request (code 0x02) 58
4.3 Connection Response (code 0x03) 59
4.4 Configuration Request (code 0x04) 60
4.5 Configuration Response (code 0x05) 62
4.6 Disconnection Request (code 0x06) 64
4.7 Disconnection Response (code 0x07) 65
4.8 Echo Request (code 0x08) 66
4.9 Echo Response (code 0x09) 66
4.10 Information Request (code 0x0A) 66
4.11 Information Response (code 0x0B) 67
4.12 Extended Feature Mask 69
4.13 Fixed Channels Supported 70
4.14 Create Channel Request (code 0x0C) 70
4.15 Create Channel Response (code 0x0D) 71
4.16 Move Channel Request (code 0x0E) 73
4.17 Move Channel Response (code 0x0F) 74

4.18 Move Channel Confirmation (code 0x10).....75
4.19 Move Channel Confirmation Response (code 0x11).....76
4.20 Connection Parameter Update Request (code 0X12).....76
4.21 Connection Parameter Update Response (code 0X13).....78
5 Configuration Parameter Options.....79
5.1 Maximum Transmission Unit (MTU).....79
5.2 Flush Timeout Option.....81
5.3 Quality of Service (QoS) Option.....82
5.4 Retransmission and Flow Control Option.....86
5.5 Frame Check Sequence (FCS) Option.....91
5.6 Extended Flow Specification Option.....92
5.7 Extended Window Size Option.....96
6 State Machine.....98
6.1 General rules for the state machine:.....98
6.2 Timers events.....114
7 General Procedures.....120
7.1 Configuration Process.....120
7.2 Fragmentation and Recombination.....126
7.3 Encapsulation of SDUs.....128
7.4 Delivery of Erroneous L2CAP SDUs.....132
7.5 Operation with Flushing On ACL-U Logical Links.....132
7.6 Connectionless Data Channel.....133
7.7 Operation Collision Resolution.....135
7.8 Aggregating Best Effort Extended Flow Specifications.....135
7.9 Prioritizing Data over HCI.....136
7.10 Supporting Extended Flow Specification for BR/EDR And BR/EDR/LE Controllers.....137
8 Procedures for Flow Control and Retransmission.....139
8.1 Information Retrieval.....139
8.2 Function of PDU Types for Flow Control and Retransmission.....139
8.3 Variables and Sequence Numbers.....140
8.4 Retransmission Mode.....144
8.5 Flow Control Mode.....150
8.6 Enhanced Retransmission Mode.....154
8.7 Streaming Mode.....184
9 Procedure for AMP Channel Creation and Handling.....186
9.1 Create Channel.....186
9.2 Move Channel.....189
9.3 Disconnect Channel.....197
10 List of Figures.....198
11 List of Tables.....200

Part B
SERVICE DISCOVERY PROTOCOL (SDP) SPECIFICATION

1 Introduction 209
1.1 General Description 209
1.2 Motivation 209
1.3 Requirements 209
1.4 Non-requirements and Deferred Requirements 210
1.5 Conventions 210

2 Overview 211
2.1 SDP Client-Server Interaction 211
2.2 Service Record 212
2.3 Service Attribute 213
2.4 Service Class 215
2.5 Searching for Services 216
2.6 Browsing for Services 217

3 Data Representation 220
3.1 Data Element 220
3.2 Data Element Type Descriptor 220
3.3 Data Element Size Descriptor 221
3.4 Data Element Examples 222

4 Protocol Description 223
4.1 Transfer Byte Order 223
4.2 Protocol Data Unit Format 223
4.3 Partial Responses and Continuation State 225
4.4 Error Handling 225
4.5 ServiceSearch Transaction 227
4.6 ServiceAttribute Transaction 230
4.7 ServiceSearchAttribute Transaction 233

5 Service Attribute Definitions 237
5.1 Universal Attribute Definitions 237
5.2 ServiceDiscoveryServer Service Class Attribute Definitions 246
5.3 BrowseGroupDescriptor Service Class Attribute Definitions 248

6 Security 250

Part C
GENERIC ACCESS PROFILE

1 Introduction 274
1.1 Scope 274
1.2 Symbols and Conventions 275

2 Profile Overview 277

2.1 Profile Stack 277
2.2 Profile Roles 277
2.3 User Requirements and Scenarios 280
2.4 Profile Fundamentals 281
2.5 Conformance 281
2.6 Other Requirements 281

3 User Interface Aspects 282
3.1 The User Interface Level 282
3.2 Representation of Bluetooth Parameters 282
3.3 Pairing 286

4 Modes 287
4.1 Discoverability Modes 287
4.2 Connectability Modes 291
4.3 Bondable Modes 293

5 Security Aspects 295
5.1 Authentication 295
5.2 Security Modes 296

6 Idle Mode Procedures 314
6.1 General Inquiry 314
6.2 Limited Inquiry 315
6.3 Name Discovery 317
6.4 Device Discovery 318
6.5 Bonding 319

7 Establishment Procedures 323
7.1 Link Establishment 323
7.2 Channel Establishment 326
7.3 Connection Establishment 328
7.4 Establishment of Additional Connection 329

8 Extended Inquiry Response Data Format 330
8.1 EIR Data Type Definitions 331
8.2 Example Extended Inquiry Response 333

9 Operational Modes and Procedures For Use On LE Physical Channels335
9.1 Broadcast Mode and Observation Procedure 335
9.2 Discovery Modes and Procedures 337
9.3 Connection Modes and Procedures 344
9.4 Bonding Modes and Procedures 359

10 LE Security Aspects 362
10.1 Requirements 362
10.2 LE Security Modes 362

10.3 Authentication Procedure 363
10.4 Data Signing 368
10.5 Authorization Procedure 369
10.6 Encryption Procedure 369
10.7 Privacy Feature 369
10.8 Random Device Address 371

11 Advertising and Scan Response Data Format 375
11.1 AD Type Definitions 375
11.2 Example Advertising Data 377

12 GAP Characteristics for Low Energy 379
12.1 Device Name Characteristic 379
12.2 Appearance Characteristic 380
12.3 Peripheral Privacy Flag Characteristic 380
12.4 Reconnection Address Characteristic 380
12.5 Peripheral Preferred Connection Parameters Characteristic 381

13 BR/EDR/LE Operation Modes and Procedure 383
13.1 Modes 383
13.2 Idle Mode Procedures 385
13.3 Establishment Procedures 388
13.4 SDP Interoperability Requirements 388

14 BR/EDR/LE Security Aspects 390

15 Definitions 391
15.1 General Definitions 391
15.2 Connection-related Definitions 391
15.3 Device-related Definitions 392
15.4 Procedure-related Definitions 393
15.5 Security-related Definitions 393

16 Appendix A (Normative): Timers and Constants 395

17 Appendix B (Informative): Information Flows of Related Procedures 398
17.1 LMP – Authentication 398
17.2 LMP – Pairing 399
17.3 Service Discovery 400
17.4 Generating a Resolvable Private Address 400
17.5 Resolving a Resolvable Private Address 400

18 Appendix C (Normative): EIR and AD Formats 401
18.1 Flags 401
18.2 Service 402
18.3 Local Name 402
18.4 TX Power Level 402
18.5 Simple Pairing Optional OOB Tags 403

18.6 Security Manager TK Value 403
18.7 Security Manager OOB Flags 403
18.8 Slave Connection Interval Range....404
18.9 Service Solicitation....404
18.10 Service Data....404
18.11 Manufacturer Specific Data....405

19 References....406

Part D
TEST SUPPORT

1 Test Methodology....411
1.1 BR/EDR Test Scenarios....411
1.2 AMP Test Scenarios....421
1.3 References....429

2 Test Control Interface (TCI)....430
2.1 Introduction....430
2.2 TCI Configurations....431
2.3 TCI Configuration and Usage....434

Part E
AMP MANAGER PROTOCOL SPECIFICATION

1 Introduction....442
1.1 General Description....442

2 General Operation....443
2.1 Basic Capabilities....443
2.2 AMP Manager Channel Over L2CAP....444
2.3 Using the AMP Manager Protocol....445
2.4 Controller IDs....447
2.5 Controller Types....447

3 Protocol Description....448
3.1 Packet Formats....448
3.2 AMP Command Reject (Code 0x01)....449
3.3 AMP Discover Request (Code 0x02)....450
3.4 AMP Discover Response (Code 0x03)....451
3.5 AMP Change Notify (Code 0x04)....455
3.6 AMP Change Response (Code 0x05)....456
3.7 AMP Get Info Request (Code 0x06)....456
3.8 AMP Get Info Response (Code 0x07)....456
3.9 AMP Get AMP Assoc Request (Code 0x08)....458
3.10 AMP Get AMP Assoc Response (Code 0x09)....459
3.11 AMP Create Physical Link Request (Code 0x0A)....460

3.12 AMP Create Physical Link Response (Code 0x0B) 461
3.13 AMP Disconnect Physical Link Request (Code 0x0C) 462
3.14 AMP Disconnect Physical Link Response (Code 0x0D) 463
3.15 Create Physical Link Collision Resolution 464
3.16 Response Timeout 465
3.17 Unexpected BR/EDR Physical Link Disconnect 465

Part F
ATTRIBUTE PROTOCOL (ATT)

1 Introduction 473
1.1 Scope 473
1.2 Conformance 473

2 Protocol Overview 474

3 Protocol Requirements 475
3.1 Introduction 475
3.2 Basic Concepts 475
3.3 Attribute PDU 480
3.4 Attribute Protocol PDUs 483

4 Security Considerations 516

5 Acronyms and Abbreviations 517

Part G
GENERIC ATTRIBUTE PROFILE (GATT)

1 Introduction 524
1.1 Scope 524
1.2 Profile Dependency 524
1.3 Conformance 524
1.4 Bluetooth Specification Release Compatibility 525
1.5 Conventions 525

2 Profile Overview 526
2.1 Protocol Stack 526
2.2 Configurations and Roles 526
2.3 User Requirements and Scenarios 527
2.4 Profile Fundamentals 528
2.5 Attribute Protocol 528
2.6 GATT Profile Hierarchy 531
2.7 Configured Broadcast 533

3 Service Interoperability Requirements 535
3.1 Service Definition 535
3.2 Include Definition 536

3.3 Characteristic Definition 536
3.4 Summary of GATT Profile Attribute Types 547
4 GATT Feature Requirements 548
4.1 Overview 548
4.2 Feature Support and Procedure Mapping 548
4.3 Server Configuration 550
4.4 Primary Service Discovery 551
4.5 Relationship Discovery 554
4.6 Characteristic Discovery 556
4.7 Characteristic Descriptor Discovery 558
4.8 Characteristic Value Read 560
4.9 Characteristic Value Write 563
4.10 Characteristic Value Notification 569
4.11 Characteristic Value Indications 570
4.12 Characteristic Descriptors 571
4.13 GATT Procedure Mapping to ATT Protocol Opcodes 575
4.14 Procedure Timeouts 578
5 L2CAP Interoperability Requirements 579
5.1 BR/EDR L2CAP Interoperability Requirements 579
5.2 LE L2CAP Interoperability Requirements 580
6 GAP Interoperability Requirements 582
6.1 BR/EDR GAP Interoperability Requirements 582
6.2 LE GAP Interoperability Requirements 582
6.3 Disconnected Events 582
7 Defined Generic Attribute Profile Service 584
7.1 Service Changed 584
8 Security Considerations 586
8.1 Authentication Requirements 586
8.2 Authorization Requirements 587
9 SDP Interoperability Requirements 588
10 References 589
11 Appendix: Example Attribute Server Attributes 590

Part H
SECURITY MANAGER SPECIFICATION

1 Introduction 598
1.1 Scope 598
1.2 Conventions 598
2 Security Manager 599
2.1 Introduction 599

2.2 Cryptographic Toolbox 600
2.3 Pairing Methods 603
2.4 Security in Bluetooth low energy 612

3 Security Manager Protocol 620
3.1 Introduction 620
3.2 Security Manager Channel over L2CAP 620
3.3 Command Format 620
3.4 SMP Timeout 621
3.5 Pairing Methods 622
3.6 Security in Bluetooth low energy 628

4 References 635

5 Appendices 638
5.1 Appendix A – EDIV and Rand Generation 638
5.2 Appendix B – Key Management 639
5.3 Message Sequence Charts 643

Specification Volume 4
Host Controller Interface
[Transport Layer]

Part A
UART TRANSPORT LAYER

1 General 13

2 Protocol 14

3 RS232 Settings 15

4 Error Recovery 16

Part B
USB TRANSPORT LAYER

1 Overview 21

2 USB Endpoint Expectations 25

2.1 Descriptor Overview 25
2.2 Control Endpoint Expectations 31
2.3 Bulk Endpoints Expectations 32
2.4 Interrupt Endpoint Expectations 33
2.5 Isochronous Endpoints Expectations 33

3 Class Code 34

3.1 Bluetooth Codes 34

4 Device Firmware Upgrade 36

5 Limitations 37

5.1 Power Specific Limitations 37
5.2 Other Limitations 37

6 Bluetooth Composite Device Implementation 38

6.1 Configurations 38
6.2 Using USB Interface Association Descriptors for a Primary Controller Function 38
6.3 Combined Primary Controller Function and Single AMP Controller Function 39

7 References 43

Part C
SECURE DIGITAL (SD)
TRANSPORT LAYER

1 Introduction 48

2 Goals 49
2.1 Hardware Goals 49
2.2 Software Goals 49
2.3 Configuration Goals 50
2.4 Configuration for Multiple Controllers 50
3 Physical Interface Documents 51
4 Communication 52
4.1 Overview 52
5 Appendix A - Acronyms and Abbreviations 53
6 Appendix B - Related Documents 54
7 Appendix C - Tests 55
7.1 Test Suite Structure 55

Part D
THREE-WIRE UART TRANSPORT LAYER

1 General 61
2 Overview 62
3 Slip Layer 63
3.1 Encoding a Packet 63
3.2 Decoding a Packet 63
4 Packet Header 65
4.1 Sequence Number 65
4.2 Acknowledge Number 66
4.3 Data Integrity Check Present 66
4.4 Reliable Packet 66
4.5 Packet Type 66
4.6 Payload Length 67
4.7 Packet Header Checksum 67
5 Data Integrity Check 68
5.1 16 Bit CCITT-CRC 68
6 Reliable Packets 69
6.1 Header Checksum Error 69
6.2 Slip Payload Length Error 69
6.3 Data Integrity Check Error 69
6.4 Out Of Sequence Packet Error 69
6.5 Acknowledgement 70
6.6 Resending Packets 70
6.7 Example Reliable Packet Flow 70
7 Unreliable Packets 73

7.1 Unreliable Packet Header 73
7.2 Unreliable Packet Error 73
8 Link Establishment 74
8.1 Uninitialized State 75
8.2 Initialized State 75
8.3 Active State 75
8.4 Sync Message 76
8.5 Sync Response Message 76
8.6 Config Message 76
8.7 Config Response Message 77
8.8 Configuration Field 77
9 LOW POWER 80
9.1 Wakeup Message 80
9.2 Woken Message 80
9.3 Sleep Message 81
10 Out of Frame Control 82
10.1 Software Flow Control 82
11 Hardware Configuration 83
11.1 Wires 83
11.2 Hardware Flow 83
12 Recommended Parameters 84
12.1 Timing Parameters 84
13 References 85

Specification Volume 5
Core System Package
[AMP Controller volume]

Part A
802.11 PROTOCOL ADAPTATION LAYER FUNCTIONAL SPECIFICATION

1 Introduction 11
1.1 Organization of the 802.11 PAL 11

2 AMP Host Controller Interface 13
2.1 Read Local Version Information Command 13
2.2 Read Local Amp Info Command 13
2.3 Reset Command 15
2.4 Read Failed Contact Counter Command 16
2.5 Read Link Quality Command 16
2.6 Read RSSI Command 16
2.7 Short Range Mode Command 16
2.8 Write Best Effort Flush Timeout Command 17
2.9 Read Best Effort Flush Timeout Command 17
2.10 Physical Link Loss Early Warning Event 17
2.11 Physical Link Recovery Event 18
2.12 Channel Selected Event 18
2.13 Short Range Mode Change Completed Event 18
2.14 Data Structures 18
2.15 Connection Accept Timeout Configuration Parameter 22

3 Physical Link Manager 23
3.1 Physical Link State Machine 23
3.2 Channel Selection 31
3.3 802.11 Link Creation 34
3.4 Physical Link Maintenance 36
3.5 Physical Link Security 37
3.6 Physical Link Support for QOS 39

4 Logical Link Manager 40
4.1 Logical Link Creation 40
4.2 Logical Link Updates 41
4.3 Logical Link Deletion 42

5 Data Manager 43
5.1 Encapsulation 43
5.2 Coexistence and Local Interference 44
5.3 Explicit Flush 47
5.4 Automatic Flush 47

5.5 Quality Of Service Violations....47
6 Constants....48
7 Message Sequence Charts....49
8 Appendix A: Test Support....50
8.1 AMP Test Command....50
8.2 AMP Start Test Event....53
8.3 AMP Test End Event....54
9 References....55
10 List of Figures....56
11 List of Tables....57

Specification Volume 6
Core System Package [Low Energy Controller volume]

Part A
PHYSICAL LAYER SPECIFICATION

1 Scope 14
2 Frequency Bands and Channel Arrangement 16
3 Transmitter Characteristics 17
3.1 Modulation Characteristics 17
3.2 Spurious Emissions 18
3.3 Radio Frequency Tolerance 19
4 Receiver Characteristics 20
4.1 Actual Sensitivity Level 20
4.2 Interference Performance 20
4.3 Out-of-Band Blocking 21
4.4 Intermodulation Characteristics 21
4.5 Maximum Usable Level 22
4.6 Reference Signal Definition 22
5 Appendix A 23
5.1 Normal Operating Conditions (NOC) 23
5.2 Extreme Operating Conditions (EOC) 23
6 Appendix B - Operating Conditions 24

Part B
LINK LAYER SPECIFICATION

1 General Description 30
1.1 Link Layer States 30
1.2 Bit Ordering 32
1.3 Device Address 33
1.4 Physical Channel 34
2 Air Interface Packets 36
2.1 Packet Format 36
2.2 Reserved for Future Use (RFU) 37
2.3 Advertising Channel PDU 37
2.4 Data Channel PDU 44
3 Bit Stream Processing 52
3.1 Error Checking 52
3.2 Data Whitening 53
4 Air Interface Protocol 55

4.1 Inter Frame Space 55
4.2 Timing Requirements 55
4.3 Link Layer Device Filtering 55
4.4 Non-Connected States 57
4.5 Connection State 67
4.6 Feature Support 77

5 Link Layer Control 78
5.1 Link Layer Control Procedures 78
5.2 Procedure Response Timeout 87

Part C
SAMPLE DATA

1 Encryption sample data 91
1.1 Encrypt Command 93
1.2 Derivation of the MIC and Encrypted Data 93

Part D
MESSAGE SEQUENCE CHARTS

1 Introduction 100
1.1 Notation 100
1.2 Control Flow 101
1.3 Example MSC 101

2 Standby State 102
2.1 Initial Setup 102
2.2 Random Device Address 103
2.3 White Lists 103

3 Advertising State 104
3.1 Undirected Advertising 104
3.2 Directed Advertising 105

4 Scanning State 106
4.1 Passive Scanning 106
4.2 Active Scanning 106

5 Initiating State 108
5.1 Initiating a Connection 108
5.2 Canceling an Initiation 108

6 Connection State 110
6.1 Sending Data 110
6.2 Connection Update 110
6.3 Channel Map Update 111
6.4 Features Exchange 111

6.5 Version Exchange 112
6.6 Start Encryption 113
6.7 Start Encryption without Long Term Key 114
6.8 Start Encryption with Event Masked 115
6.9 Start Encryption Without Slave Supporting Encryption 115
6.10 Restart Encryption 116
6.11 Disconnect 116

Part E
LOW ENERGY LINK LAYER SECURITY

1 Encryption and Authentication Overview 121

2 CCM 122
2.1 CCM Nonce 122
2.2 Counter Mode Blocks 123
2.3 Encryption Blocks 124

Part F
DIRECT TEST MODE

1 Introduction 128

2 Low Energy Test Scenarios 129
2.1 Test Sequences 129
2.2 Message Sequence Charts 130

3 UART Test Interface 132
3.1 UART Interface Characteristics 132
3.2 UART Functional Description 132
3.3 Commands and Events 133
3.4 Events 134
3.5 Timing – Command and Event 136

BLUETOOTH COMPLIANCE REQUIREMENTS

This document specifies the requirements for Bluetooth compliance.

CONTENTS

1 Introduction 43

2 Scope 44

3 Definitions....45

3.1 Types of Bluetooth Products 45

4 Core Configurations....50

4.1 Basic Rate Core Configuration....50

4.2 Enhanced Data Rate Core Configurations....51

4.3 High Speed Core Configuration 52

4.4 Low Energy Core Configuration....52

4.5 Basic Rate and Low Energy Combined Core Configuration 53

4.6 Host Controller Interface Core Configuration....54

1 INTRODUCTION

The Bluetooth Qualification Program Reference Document (PRD) is the primary reference document for the Bluetooth Qualification Program and defines its requirements, functions, and policies. The PRD is available on the Bluetooth Web site.

Passing the Bluetooth Qualification Process demonstrates a certain measure of compliance and interoperability, but because products are not tested for every aspect of this Bluetooth Specification, qualification does not guarantee compliance. Passing the Bluetooth Qualification Process only satisfies one condition of the license grant. The Member has the ultimate responsibility to ensure that the qualified product complies with this Bluetooth Specification and interoperates with other products.

2 SCOPE

This part of the specification defines some fundamental concepts used in the Bluetooth Qualification Program.

3 DEFINITIONS

Bluetooth Qualification Process – The process defined in the Bluetooth Qualification Program Reference Document (PRD) to qualify a design used in implementations of Bluetooth wireless technology.

Bluetooth Qualification Program – The Bluetooth Qualification Process together with other related requirements and processes.

3.1 TYPES OF BLUETOOTH PRODUCTS

Bluetooth Product—Any product containing an implementation of Bluetooth wireless technology. Bluetooth Products as defined herein may require enabling technology external to Bluetooth Scope, as defined by the Patent and Copyright License Agreement, to become functional (e.g. power supply, technology capable of running executable code, etc.). Enabling technology is not part of any of the Bluetooth Product type definitions and is not included in the Bluetooth License grant.

All Bluetooth Products shall be one of the following:

- Bluetooth End Product
- Bluetooth Host Subsystem Product
- Bluetooth Controller Subsystem Product
- Bluetooth Profile Subsystem Product
- Bluetooth Component Product
- Bluetooth Development Tool
- Bluetooth Test Equipment

Table 3.1 defines abbreviations for the different Core Configurations defined in Section 4 on page 50.

Abbreviation	Explanation	Section Reference
BR CC	Bluetooth Basic Rate Core Configuration	Section 4.1
EDR CC	Bluetooth Enhanced Data Rate Core Configuration	Section 4.2
HS CC	High Speed Bluetooth Core Configuration	Section 4.3
LE CC	Bluetooth Low Energy Core Configuration	Section 4.4
BR and LE Combined CC	Bluetooth Basic Rate and Low Energy Combined Core Configuration	Section 4.5
HCI CC	Host Controller Interface Core Configuration	Section 4.6

Table 3.1: Core Configuration abbreviations

Using the abbreviations in Table 3.1 the following tables define Bluetooth product types in terms of Core Configurations. For the respective Core Configuration, the letter “M” indicates that it is mandatory to claim support, “O” indicates that it is optional to claim support, “P” indicates that it is optionally permitted to claim only partial support of the Core Configuration, “I” indicates that the Core Configuration is inherently included in the combined Core Configuration, “X” indicates that support for the Core Configuration shall not be claimed.

Bluetooth End Product—A Bluetooth wireless technology product that claims to implement one or more Core Configurations, in compliance with the required parts of the Specification, and in accordance with the mandatory requirements as defined herein. The following Bluetooth End Product types are defined in Table 3.2:

Bluetooth End Product types:

	BR CC	EDR CC	HS CC	BR and LE Combined CC	LE CC	HCI CC
BR End Product	M	P	P	X	X	O
EDR End Product	M	M	P	X	X	O
HS End Product	M	M	M	X	X	O
LE End Product	X	X	X	X	M	O
BR and LE End Product	I	P	P	M	I	O
EDR and LE End Product	I	M	P	M	I	O
HS and LE End Product	I	M	M	M	I	O

Table 3.2: Required configuration per Bluetooth End Product type

Bluetooth Subsystem Product —A Bluetooth wireless technology product that claims to implement only a portion of the Specification, in compliance with such portion of the Specification, and in accordance with the mandatory requirements as defined herein. Bluetooth Subsystem Products can be qualified solely for distribution; the use of Bluetooth wireless technology in Bluetooth Subsystem Products require such Bluetooth Subsystem Products to be combined with

a complementary Bluetooth End Product or one or more complementary Bluetooth Subsystem Products such that the resulting combination satisfies the requirements of a Bluetooth End Product. There are three types of Bluetooth Subsystem Products defined as Bluetooth Host Subsystems (see Table 3.3), Bluetooth Controller Systems (see Table 3.4), and Bluetooth Profile Subsystems.

Bluetooth Host Subsystem Product types:

	BR CC Host Parts	**HS CC Host Parts**	**BR and LE Combined CC Host Parts**	**LE CC Host Parts**	**HCI CC**
BR/EDR Host Subsystem Product	M	P	X	X	M
HS Host Subsystem Product	M	M	X	X	M
LE Host Subsystem Product	X	X	X	M	M
BR/EDR and LE Host Subsystem Product	I	P	M	I	M
HS and LE Host Subsystem Product	I	M	M	I	M

Table 3.3: Required configuration per Bluetooth Host Subsystem Product type

A Bluetooth Host Subsystem Product may contain, in addition to the required Core Configuration Host parts (as defined in Table 3.3), all the mandatory requirements defined in one or more of the protocols and profiles above HCI. Protocols below HCI required by the Core Configuration Controller parts (as defined in Table 3.4) shall be excluded from the Host Subsystem Product.

Bluetooth Controller Subsystem Product Types

	BR CC Controller Parts	**EDR CC Controller Parts**	**HS CC Controller Parts**	**BR and LE Combined CC Controller Parts**	**LE CC Controller Parts**	**HCI CC**
BR Controller Subsystem Product	M	P	P	X	X	M

Table 3.4: Required configuration per Bluetooth *Controller Subsystem Product type*

	BR CC Controller Parts	EDR CC Controller Parts	HS CC Controller Parts	BR and LE Combined CC Controller Parts	LE CC Controller Parts	HCI CC
EDR Controller Subsystem Product	M	M	P	X	X	M
HS Controller Subsystem Product	M	M	M	X	X	M
HS only Controller Subsystem Product	X	X	M	X	X	M
LE Controller Subsystem Product	X	X	X	X	M	M
BR and LE Controller Subsystem Product	I	P	P	M	I	M
EDR and LE Controller Subsystem Product	I	M	P	M	I	M
HS and LE Controller Subsystem Product	I	M	M	M	I	M

Table 3.4: Required configuration per Bluetooth *Controller Subsystem Product type*

Protocols and Profiles above HCI required by the Core Configuration Host parts (as defined in Table 3.3) shall be excluded from the Controller Subsystem Product.

Bluetooth Profile Subsystem Product—A Bluetooth wireless technology product that claims to implement, at a minimum, all the mandatory requirements defined in one or more of the profile or service specifications.

Bluetooth Component Product—A Bluetooth wireless technology product that claims to implement, at a minimum, all the mandatory requirements, if any, of either one or more of any of the protocol and profile or service parts of the Specification in compliance with such portion of the Specification. Bluetooth Component Products can be qualified solely for distribution and the use of the Bluetooth wireless technology in Bluetooth Component Products require such Bluetooth Component Products to be incorporated in Bluetooth End Products

or Bluetooth Subsystem Products. A product that meets the requirements of a Bluetooth End Product or Bluetooth Subsystem product may be qualified as a Bluetooth Component Product if a manufacturer determines that further integration is necessary prior to qualifying the product as a Bluetooth End Product or Bluetooth Subsystem Product.

Bluetooth Development Tool—A Bluetooth wireless technology product intended to facilitate the development of new Bluetooth designs. Bluetooth Development Tools can be qualified solely for distribution and the use of the Bluetooth wireless technology in development of new Bluetooth Products.

Bluetooth Test Equipment—A Bluetooth wireless technology product intended to facilitate the testing of new Bluetooth Products. Bluetooth Test Equipment can be qualified solely for distribution and the use of the Bluetooth wireless technology in testing of new Bluetooth Products. Where necessary, Bluetooth Test Equipment may deviate from the Specification in order to fulfill the test purposes in the Bluetooth Test Specifications.

4 CORE CONFIGURATIONS

This section defines the set of features that are required for a product to be qualified to a specification name. The Core Specification version number is simply the version number of the specification itself.

Specification names differ from Core Specification version numbers in that products are marked based on meeting requirements for a Core Configuration together with the mixing requirements (see [Vol. 1] Part D, Section 1).

Each Core Configuration is defined by a set of parts and individual features of the Core Specification that shall be supported to allow the configuration name to be used. The configuration requirements imposed on a device may depend on the profiles that it supports.

4.1 BASIC RATE CORE CONFIGURATION

This section specifies compliance requirements for the “Basic Rate” Core Configuration.

To claim support to the “Basic Rate” Core Configuration, an implementation must support a set of Required Features, according to the details in Table 4.1 and Table 4.2.

Host Part:

Layer	Required Features
L2CAP ([Vol. 3] Part A)	L2CAP Signaling Channel (CID 0x0001) and all mandatory features associated with it
SDP ([Vol. 3] Part B)	All mandatory features
ATT ([Vol. 3] Part F)	If ATT is supported, all mandatory features
GATT ([Vol. 3] Part G)	GATT is optional when the ATT is supported. When supported, all mandatory features.
GAP ([Vol. 3] Part C)	All mandatory features in sections 2 through 8

Table 4.1: BR Core Configuration Host requirements

Controller Part:

Layer	Required Features
RF ([Vol. 2] Part A)	All mandatory features
BB ([Vol. 2] Part B)	All mandatory features

Table 4.2: BR Core Configuration Controller requirements

Layer	Required Features
LMP ([Vol. 2] Part C)	All mandatory features

Table 4.2: BR Core Configuration Controller requirements

4.2 ENHANCED DATA RATE CORE CONFIGURATIONS

This section specifies compliance requirements for the “Enhanced Data Rate” Core Configuration.

Table 4.3 defines three categories of Transport Requirements that shall be satisfied subject to the following rules:

- A Bluetooth product shall support category 1 whenever it supports asynchronous transports for the profiles it incorporates.
- A Bluetooth product shall support category 2 whenever it supports asynchronous transports with multislot ACL packets for the profiles it incorporates.
- A Bluetooth product shall support category 3 whenever it supports eSCO synchronous transports for the profiles it incorporates.

A multi-profile product shall support all applicable categories in order to claim support for the Enhanced Data Rate Core Configuration.

		Controller part:	Host part:
Category No.	**Transport Requirements**	**LMP Features Supported**	**L2CAP Feature Bits Required**
1	EDR for asynchronous transports (single slot)	Enhanced Data Rate ACL 2 Mbps mode (25) AND Enhanced Data Rate ACL 3 Mbps mode (26)	None
2	EDR for asynchronous transports (multi-slot)	3-slot Enhanced Data Rate ACL packets (39) AND 5-slot Enhanced Data Rate ACL packets (40)	None
3	EDR for synchronous transports	Enhanced Data Rate eSCO 2 Mbps mode (45)	None

Table 4.3: EDR Core Configuration requirements

Note: No additional requirements are stated on the support of 3-EV3, 2-EV5 and 3-EV5 packets.

4.3 HIGH SPEED CORE CONFIGURATION

This section specifies compliance requirements for the “High Speed” Core Configuration.

To claim support to the “High Speed” Core Configuration, an implementation must support a set of Required Features, according to the details inTable 4.4 and Table 4.5.

Host Part:

Layer	Required Features
L2CAP ([Vol. 3] Part A)	Enhanced Retransmission Mode (L2CAP extended features mask number 3) Fixed Channels (extended features mask number 7) AMP Manager Fixed Channel (CID 0M0003) AMP Channel Creation and Handling [Vol. 5] Part A, Section 9)
A2MP ([Vol. 3] Part E)	All mandatory features

Table 4.4: HS Core Configuration Host requirements

Controller Part:

Layer	Required Features
802.11 PAL ([Vol. 5] Part A)	All mandatory features

Table 4.5: HS Core Configuration Controller requirements

4.4 LOW ENERGY CORE CONFIGURATION

This section specifies compliance requirements for the “Low Energy” Core Configuration.

To claim support to the “Low Energy” Core Configuration, an implementation must support a set of Required Features, according to the details in Table 4.6 and Table 4.7.

Host Part:

Layer	Required Features
L2CAP ([Vol. 3] Part A)	L2CAP LE Signaling Channel (CID 0x0005) and all mandatory features associated with it
GAP ([Vol. 3] Part C)	All mandatory features in sections 9-12
ATT ([Vol. 3] Part F)	All mandatory features
GATT ([Vol. 3] Part G)	All mandatory features
SM ([Vol. 3] Part H)	All mandatory features

Table 4.6: LE Core Configuration Host requirements

Controller Part:

Layer	Required Features
PHY ([Vol. 6] Part A)	All mandatory features
LL ([Vol. 6] Part B)	All mandatory features

Table 4.7: LE Core Configuration Controller requirements

4.5 BASIC RATE AND LOW ENERGY COMBINED CORE CONFIGURATION

This section specifies compliance requirements for the “Basic Rate and Low Energy Combined” Core Configuration.

To claim support for the “Basic Rate and Low Energy” Core Configuration, an implementation must support a set of Required Features, according to the details inTable 4.8 and Table 4.9.

Host Part:

Layer	Required Features
L2CAP ([Vol. 3] Part A)	All L2CAP requirements in the BR CC and LE CC
SDP ([Vol. 3] Part B)	All mandatory features
GAP ([Vol. 3] Part C)	All GAP requirements in the BR CC and LE CC AND All requirements in sections 13 and 14
ATT ([Vol. 3] Part F)	All ATT requirements in the LE CC

Table 4.8: BR and LE Combined Core Configuration Host requirements

Layer	Required Features
GATT ([Vol. 3] Part G)	All GATT requirements in the LE CC
SM ([Vol. 3] Part H)	All SM requirements in the LE CC

Table 4.8: BR and LE Combined Core Configuration Host requirements

Controller Part:

Layer	Required Features
Radio ([Vol. 2] Part A)	All Radio requirements in the BR CC
BB ([Vol. 2] Part B)	All BB requirements in the BR CC
LMP ([Vol. 2] Part C)	All LMP requirements in the BR CC AND LMP feature bits 38 and 65 shall be set
PHY ([Vol. 6] Part A)	All mandatory features in the LE CC
LL ([Vol. 6] Part B)	All mandatory features in the LE CC

Table 4.9: BR and LE Combined Core Configuration Controller requirements

4.6 HOST CONTROLLER INTERFACE CORE CONFIGURATION

This section specifies compliance requirements for the “Host Controller Interface” Core Configuration.

To claim support for the “Host Controller Interface” Core Configuration, an implementation must support a set of Required Features, according to the details in Table 4.10.

Layer	Required Features
HCI ([Vol. 2] Part E)	All the supported features in the implementation shall be compliant to the Host Controller Interface.

Table 4.10: HCI Core Configuration requirements

APPENDIX

Ford Ex. 1052

CONTENTS

1 Revision History 59
1.1 [Vol 0] Master TOC & Compliance Requirements 59
1.1.1 Bluetooth Compliance Requirements 59
1.2 [Vol 1] Architecture & Terminology Overview 59
1.3 [Vols 2, 3, & 6] Core System Package 61
1.4 [Vol 4] Transport Layers 64
2 Contributors 65
2.1 [Vol 0] Master TOC & Compliance Requirements 65
2.1.1 Part B: Bluetooth Compliance Requirements 65
2.1.2 Vol 0 Part C: Appendix (Rev History and Contributors 65
2.2 [Vol 1] Architecture &Terminology Overview 65
2.2.1 Part A: Architectural Overview 65
2.2.2 Part B: Acronyms & Abbreviations 67
2.2.3 Part C: Core Specification Change History 67
2.2.4 Part D: Mixing of Specification Versions 67
2.3 [Vol 2] Core System Package, Controller 69
2.3.1 Part A: Radio Specification 69
2.3.2 Part B: Baseband Specification 70
2.3.3 Part C: Link Manager Protocol 73
2.3.4 Part D: Error Codes 76
2.3.5 Part E: Bluetooth Host Controller Interface Functional Specification 78
2.3.6 Part F: Message Sequence Charts 81
2.3.7 Part G: Sample Data 83
2.3.8 Part H: Security Specification 84
2.4 [Vol 3] Core System Package, Host 86
2.4.1 Part A: Logical Link Control and Adaptation Protocol Specification 86
2.4.2 Part B: Service Discovery Protocol (SDP) 89
2.4.3 Part C: Generic Access Profile 89
2.4.4 Part D: Test Support 91
2.4.5 Part E: AMP Manager Protocol 92
2.4.6 Part F: Attribute Protocol Specification 93
2.4.7 Part G: Generic Attribute Protocol Specification 94
2.4.8 Part H: Security Manager Specification 94
2.5 [Vol 4] Host Controller Interface [Transport Layer] 96
2.6 [Vol 5] Core System Package [AMP Controller volume] 98

2.6.1 Part A: 802.11 PAL 98
2.7 [Vol 6] Low Energy Specification 99
2.7.1 Part A: Physical Layer Specification 99
2.7.2 Part B: Link Layer Specification 99
2.7.3 Part C: Sample Data 100
2.7.4 Part D: Message Sequence Charts 100
2.7.5 Part E: Low Energy Security Specification 100
2.7.6 Part F: Direct Test Mode 101

1 REVISION HISTORY

Beginning with v1.2 of the Core System Package the core Bluetooth specification documents, protocols and profiles were transferred to a new partitioning comprising volumes and individual profile specifications are each contained in an individual document instead of the two volumes (Core and Profiles) used in v1.1.

For more detailed information about changes between versions published before v1.2, please see the Appendix I “Revision History” in v1.1.

1.1 [VOL 0] MASTER TOC & COMPLIANCE REQUIREMENTS

1.1.1 Bluetooth Compliance Requirements

Rev	Date	Comments
4.0	June 30 2010	Updated to support Low Energy, ATT, and GATT support for BR/EDR, and to enable High Speed Controller Subsystems.
3.0 + HS	April 21 2009	Updated to include support for the Alternative MAC/PHY feature and High Speed Core Configuration.
v2.1 + EDR	July 26 2007	No content changes. Updates to the Table of Contents.
v2.0 + EDR	Oct 15 2004	This version of the specification is intended to be a separate Bluetooth Specification that has all the functional characteristics of the v1.2 Bluetooth Specification that adds the Enhanced Data Rate (EDR) feature which required changes to Volume 0, Part A, Master Table of Contents.
v1.2	Nov 05 2003	This Part was moved from the Core volume. No content changes been made to this document since v1.1.

1.2 [VOL 1] ARCHITECTURE & TERMINOLOGY OVERVIEW

Rev	Date	Comments
4.0	June 30 2010	Updated to support Low Energy. and ATT and GATT over BR/.EDR.
3.0 + HS	April 21 2009	Updated to integrate the Alternate MAC/PHY and Unicast Connectionless Data features.
v2.1 + EDR	July 26 2007	Added definitions for new features: Encryption Pause Resume, Erroneous Data reporting, Extended Inquiry Response, Link Supervision Timeout Event, Packet Boundary Flag, Secure Simple Pairing, Sniff Subrating.

Rev	Date	Comments
v2.0 + EDR	Oct 15 2004	This version of the specification is intended to be a separate Bluetooth Specification that has all the functional characteristics of the v1.2 Bluetooth Specification that adds the Enhanced Data Rate (EDR) feature which incorporates changes to Volume 1, Part B, Acronyms and Abbreviations.
v1.2	Nov 05 2003	New volume with informational content. This volume will always be updated in parallel with the Core System volumes.

1.3 [VOLS 2, 3, & 6] CORE SYSTEM PACKAGE

Rev	Date	Comments
4.0	June 30 2010	• New features added in 4.0: -Low Energy • Errata for v2.0 + EDR, v2.1 + EDR, v3.0 + HS
3.0 + HS	April 21 2009	• New features added in 3.0 + HS: -AMP Manager Protocol (A2MP) -Enhancements to L2CAP for AMP -Enhancements to HCI for AMP -Enhancements to Security for AMP -802.11 Protocol Adaptation Layer • Enhanced Power Control -Unicast Connectionless Data -HCI Read Encryption Key Size command -Generic Test Methodology for AMP -Enhanced USB and SDIO HCI Transports • Errata for v 2.0 + EDR and v2.1 + EDR
v2.1 + EDR	July 26 2007	• New features added in 2.1 + EDR: -Encryption Pause and Resume -Erroneous Data Reporting -Extended Inquiry Response -Link Supervision Timeout Changed Event -Non-Flushable Packet Boundary Flag -Secure Simple Pairing -Sniff Subrating -Security Mode 4 • Updates to IEEE language in Volume 2, Part H, Security • Errata for v2.0 + EDR
v2.0 + EDR	Aug 01 2004	This version of the specification is intended to be a separate Bluetooth Specification. This specification was created by adding EDR and the errata.

Rev	Date	Comments
v1.2	Nov 05 2003	New features added in v1.2: - Architectural overview - Faster connection - Adaptive frequency hopping - Extended SCO links - Enhanced error detection and flow control - Enhanced synchronization capability - Enhanced flow specification The Core System Package now comprises two volumes and the text has gone through a radical change both in terms of structure and nomenclature. The language is also more precise and is adapted to meet the IEEE standard. The following parts are moved from the Core System Package to other volumes or were deprecated: RFCOMM [vol 7], Object Exchange (IrDA Interoperability) [vol 8], TCS [vol 9], Interoperability Requirements for Bluetooth as a WAP Bearer [vol 6], HCI USB Transport Layer [vol4], HCI RS232 Transport Layer [vol 4], HCI UART Transport Layer [vol 4], Bluetooth Compliance Requirements [vol 0], Optional Paging Schemes [deprecated]
1.1	Feb 22 2001	The specification was updated with Errata items previously published on the web site. The Bluetooth Assigned Numbers appendix was lifted out from the specification to allow continuous maintenance on the web site.
1.0B	Dec. 1 1999	The specification was updated with Errata items previously published on the web site and was revised from a linguistic point of view. The following parts were added: Interoperability Requirements for Bluetooth as a WAP Bearer, Test Control Interface, Sample Data (appendix), Bluetooth Audio (appendix), Baseband Timers (appendix) and Optional Paging Scheme (appendix)
1.0a	July 26 1999	The first version of the Bluetooth Specification published on the public web site. Added part: Bluetooth Compliance Requirements.
1.0 draft	July 05 1999	The following parts were added: Service Discovery Protocol (SDP), Telephony Control Specification (TCS), Bluetooth Assigned Numbers (appendix) and Message Sequence Charts (appendix)
0.9	April 30 1999	The following parts were added: IrDA Interoperability, HCI RS232 Transport Layer, HCI UART Transport Layer and Test Mode

Rev	Date	Comments
0.8	Jan 21 1999	The following parts were added: Radio Specification, L2CAP, RFCOMM, HCI & HCI USB Transport Layer
0.7	Oct 19 1998	This first version only included Baseband and Link Manager Protocol

1.4 [VOL 4] TRANSPORT LAYERS

Rev	Date	Comments
4.0	June 30 2010	Incorporated errata.
3.0 + HS	April 21 2009	Updated the USB and SDIO HCI Transport protocols for high speed, support composite BR/EDR + AMP devices, and to include errata.
v2.1 + EDR	July 26 2007	Added this volume to the specification

2 CONTRIBUTORS

2.1 [VOL 0] MASTER TOC & COMPLIANCE REQUIREMENTS

2.1.1 Part B: Bluetooth Compliance Requirements

BQRB (Editor)	
Kevin Marquess	Broadcom
Wayne Park	3Com Corporation
Lawrence Jones	ComBit, Inc.
Magnus Sommansson	CSR
Gary Robinson	IBM Corporation
Georges Seuron	IBM Corporation
Rick Jessop	Intel Corporation
John Webb	Intel Corporation
Bruce Littlefield	Lucent Technologies, Inc.
Ellick Sung	Microsoft
Brian A. Redding	Motorola, Inc.
Waldemar Hontscha	Nokia Corporation
Petri Morko	Nokia Corporation
Joel Linksy	Qualcomm
Brian Redding	Qualcomm
Hans Andersson	ST Ericsson
Magnus Hansson	Telefonaktiebolaget LM Ericsson
Magnus Sommansson	Telefonaktiebolaget LM Ericsson
Göran Svennarp	Telefonaktiebolaget LM Ericsson
Warren Allen	Toshiba Corporation
John Shi	Toshiba Corporation

2.1.2 Vol 0 Part C: Appendix (Rev History and Contributors

Steve Davies	Nokia

2.2 [VOL 1] ARCHITECTURE &TERMINOLOGY OVERVIEW

2.2.1 Part A: Architectural Overview

Ayse Findikli	Atheros
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Jennifer Bray	CSR
Joe Decuir	CSR
Robin Heydon	CSR
Henrik Hedlund	CSR
Simon Kingston	CSR
Steven Singer	CSR

Steven Wenham	CSR
Paul Wright	CSR
Martin van der Zee	Ericsson
David Suvak	iAnywhere
Mattias Edlund	Infineon
Penny Chen	Intel
Selim Aissi	Intel
Chris Hansen	Intel
Josh Benaloh	Microsoft
Andy Glass	Microsoft
Peter Hauser	Microsoft
Joby Lafky	Microsoft
Kristin Lauter	Microsoft
Dan Simon	Microsoft
Don Stanwyck	Microsoft
Yacov Yacobi	Microsoft
Gideon Yuval	Microsoft
Greg Muchnik	Motorola
Brian Redding	Motorola
N Asokan	Nokia
Steve Davies	Nokia
Philip Ginzboorg	Nokia
Kanji Kerai	Nokia
Noel Lobo	Nokia
Thomas Müller	Nokia
Kaisa Nyberg	Nokia
Arto Palin	Nokia
Päivi Ruuska	Nokia
Jürgen Schnitzler	Nokia
Dominique Everaere	NXP
Javier del Prado Pavon	NXP
Reinhard Meindl	NXP
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Brian Redding	Qualcomm
Leonard Ott	Socket Mobile
Guido Bertoni	ST
Jorgen van Parijs	ST
Dominique Everaere	ST-NXP Wireless
Tim Howes	Symbian
Michael Hasling	Tality
Amihai Kidron	Texas Instruments
Eran Reuveni	Texas Instruments
Yoshimitsu Shimojo	Toshiba
John Mersh	TTPCom

2.2.2 Part B: Acronyms & Abbreviations

Dan Sönnerstam	Pyramid Communication AB
Steve Koester	Bluetooth SIG, Inc.
Steve Davies	Nokia

2.2.3 Part C: Core Specification Change History

Tom Siep	Bluetooth SIG Inc.
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Robin Heydon	CSR
Henrik Hedlund	CSR
Steven Singer	CSR
Paul Wright	CSR
David Suvak	iAnywhere
Mattias Edlund	Infineon
Rob Davies	Philips
Peter Hauser	Microsoft
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Päivi Ruuska	Nokia
Arto Palin	Nokia
Dominique Everaere	NXP
Javier del Prado Pavon	NXP
Jorgen van Parijs	ST
Tim Howes	Symbian
Amihai Kidron	Texas Instruments
Eran Reuveni	Texas Instruments
Yoshimitsu Shimojo	Toshiba
John Mersh	TTPCom

2.2.4 Part D: Mixing of Specification Versions

David Suvak	iAnywhere
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Kevin Marquess	Broadcom
Joe Decuir	CSR
Robin Heydon	CSR
Magnus Sommansson	CSR
Steven Singer	CSR
Mattias Edlund	Infineon
Peter Hauser	Microsoft
Elick Sung	Microsoft
Waldemar Hontscha	Nokia
Päivi Ruuska	Nokia

Arto Palin	Nokia
Dominique Everaere	NXP
Javier del Prado Pavon	NXP
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Brain Redding	Qualcomm
Leonard Ott	Socket Communications
Jorgen van Parijs	ST
Hans Andersson	ST Ericsson
Tim Howes	Symbian
Amihai Kidron	Texas Instruments
Eran Reuveni	Texas Instruments
Shimojo Yoshimitsu	Toshiba

2.3 [VOL 2] CORE SYSTEM PACKAGE, CONTROLLER

2.3.1 Part A: Radio Specification

Version 3.0 + HS

Phil Hough	Anritsu
Edward Harrison	Anritsu
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Mark Braun	Motorola
Joel Linsky	Qualcomm
Eran Reuveni	TI

Version 2.0 + EDR

Steven Hall	RF Micro Devices
Robert Young	CSR
Robert Kokke	Ericsson
Harald Kafemann	Nokia
Jukka Reunamäki	Nokia
Morton Gade	Digianswer
Mike Fitton	Toshiba
Oren Eliezer	Texas Instruments
Stephane Laurent-Michel	Tality

Version 1.2

Tom Siep	Bluetooth SIG Inc.
Jennifer Bray	CSR
Robin Heydon	CSR
Morten Gade	Digianswer/Motorola
Henrik Hedlund	Ericsson
Stefan Agnani	Ericsson
Robert Kokke	Ericsson
Roland Hellfajer	Infineon
Thomas Müller	Nokia
Antonio Salloum	Philips
Joel Linsky	Silicon Wave
Steven Hall	Silicon Wave
Oren Eliezer	Texas Instruments
Mike Fitton	Toshiba

Previous versions

Steve Williams	3Com Corporation
Todor V. Cooklov	Aware
Poul Hove Kristensen	Digianswer A/S

Kurt B. Fischer	Hyper Corporation
Kevin D. Marquess	Hyper Corporation
Troy Beukema	IBM Corporation
Brian Gaucher	IBM Corporation
Jeff Schiffer	Intel Corporation
James P. Gilb	Mobilian
Rich L. Ditch	Motorola, Inc.
Paul Burgess	Nokia Corporation
Olaf Joeressen	Nokia Corporation
Thomas Müller	Nokia Corporation
Arto T. Palin	Nokia Corporation
Steven J. Shellhammer	Symbol
Sven Mattisson	Telefonaktiebolaget LM Ericsson
Lars Nord (Section Owner)	Telefonaktiebolaget LM Ericsson
Anders Svensson	Telefonaktiebolaget LM Ericsson
Mary A. DuVal	Texas Instruments
Allen Hotari	Toshiba Corporation

2.3.2 Part B: Baseband Specification

Version 2.1 + EDR

Ayse Findikli	Atheros
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Robin Heydon	CSR
Simon Kingston	CSR
Steven Singer	CSR
Steven Wenham	CSR
Paul Wright	CSR
Dave Suvak	iAnywhere
Mattias Edlund	Infineon
Selim Aissi	Intel
Penny Chen	Intel
Oren Haggai	Intel
Josh Benaloh	Microsoft
Andy Glass	Microsoft
Peter Hauser	Microsoft
Joby Lafky	Microsoft
Kristin Lauter	Microsoft
Dan Simon	Microsoft
Don Stanwyck	Microsoft
Yacov Yacobi	Microsoft
Gideon Yuval	Microsoft
Greg Muchnik	Motorola
N Asokan	Nokia

Philip Ginzboorg	Nokia
Kanji Kerai	Nokia
Noel Lobo	Nokia
Kaisa Nyberg	Nokia
Arto Palin	Nokia
Päivi Ruuska	Nokia
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Dominique Everaere	NXP
Javier del Prado Pavon	NXP
Reinhard Meindl	NXP
Jorgen van Parijs	ST
Guido Bertoni	ST
Tim Howes	Symbian
Amihai Kidron	Texas Instruments
Eran Reuveni	Texas Instruments
Yoshimitsu Shimojo	Toshiba

Version 2.0 + EDR

Steven Hall	RF Micro Devices
Robert Young	CSR
Robert Kokke	Ericsson
Harald Kafemann	Nokia
Joel Linsky	RF Micro Devices
Terry Bourk	RF Micro Devices
Arto Palin	Nokia

Version 1.2

P G Madhavan	Agere
Hongbing Gan	Bandspeed, Inc.
Tod Sizer	Bell Labs
Alexander Thoukydides	CSR
Jennifer Bray	CSR
Robin Heydon	CSR
Kim Schneider	Digianswer/Motorola
Knud Dyring-Olsen	Digianswer/Motorola
Niels Nielsen	Digianswer/Motorola
Henrik Andersen	Digianswer/Motorola
Christian Gehrmann	Ericsson
Henrik Hedlund	Ericsson
Jan Åberg	Ericsson
Martin van der Zee	Ericsson
Rakesh Taori	Ericsson
Jaap Haartsen	Ericsson
Stefan Zürbes	Ericsson

Roland Hellfajer	Infineon
YC Maa	Integrated Programmable Communications, Inc.
HungKun Chen	Integrated Programmable Communications, Inc.
Steve McGowan	Intel
Adrian Stephens	Mobilian Corporation
Jim Lansford	Mobilian Corporation
Eric Meihofer	Motorola
Arto Palin	Nokia
Carmen Kühl	Nokia
Hannu Laine	Nokia
Jürgen Schnitzler	Nokia
Päivi Ruuska	Nokia
Thomas Müller	Nokia
Antonio Salloum	Philips
Harmke de Groot	Philips
Marianne van de Casteelee	Philips
Rob Davies	Philips
Roland Matthijssen	Philips
Joel Linsky (section owner)	Silicon Wave
Terry Bourk	Silicon Wave
Gary Schneider	Symbol Technologies, Inc.
Stephen J. Shellhammer	Symbol Technologies, Inc.
Michael Hasling	Tality
Amihai Kidron	Texas Instruments
Dan Michael	Texas Instruments
Eli Dekel	Texas Instruments
Jie Liang	Texas Instruments
Oren Eliezer	Texas Instruments
Tally Shina	Texas Instruments
Yariv Raveh	Texas Instruments
Anuj Batra	Texas Instruments
Katsuhiro Kinoshita	Toshiba
Toshiki Kizu	Toshiba
Yoshimitsu Shimojo	Toshiba
Charles Sturman	TTPCom
John Mersh	TTPCom
Sam Turner	TTPCom
Christoph Scholtz	University of Bonn
Simon Baatz	University of Bonn

Previous versions

Kevin D. Marquess	Hyper Corporation
Chatschik Bisdikian	IBM Corporation
Kris Fleming	Intel Corporation
James P. Gilb	Mobilian

David E. Cypher	NIST
Nada Golmie	NIST
Olaf Joeressen	Nokia Corporation
Thomas Müller	Nokia Corporation
Charlie Mellone	Motorola, Inc.
Harmke de Groot	Philips
Terry Bourk	Silicon Wave
Steven J. Shellhammer	Symbol
Jaap Haartsen	Telefonaktiebolaget LM Ericsson
Henrik Hedlund (Section Owner)	Telefonaktiebolaget LM Ericsson
Tobias Melin	Telefonaktiebolaget LM Ericsson
Joakim Persson	Telefonaktiebolaget LM Ericsson
Mary A. DuVal	Texas Instruments
Onn Haran	Texas Instruments
Thomas M. Siep	Texas Instruments
Ayse Findikli	Zeevo, Inc.

Previous versions [Encryption Sample Data, appendix]

Joel Linksy	RFMD
Thomas Müller	Nokia Corporation
Thomas Sander	Nokia Corporation
Joakim Persson (Section Owner)	Telefonaktiebolaget LM Ericsson

Previous versions [Bluetooth Audio, appendix]

Magnus Hansson	Telefonaktiebolaget LM Ericsson
Fisseha Mekuria	Telefonaktiebolaget LM Ericsson
Mats Omrin	Telefonaktiebolaget LM Ericsson
Joakim Persson (Section Owner)	Telefonaktiebolaget LM Ericsson

Previous versions [Baseband Timers, appendix]

David E. Cyper	NIST
Jaap Haartsen (Section Owner)	Telefonaktiebolaget LM Ericsson
Joakim Persson	Telefonaktiebolaget LM Ericsson
Ayse Findikli	Zeevo, Inc.

2.3.3 Part C: Link Manager Protocol

Version 4.0

Robin Heydon	CSR
Steven Wenham	CSR
Kanji Kerai	Nokia
Steve Davies	Nokia
Tim Howes	Nokia
Brian Redding	Qualcomm
Joel Linsky	Qualcomm
Alon Paycher	Texas Instruments

Version 3.0 + HS

Phil Hough	Anritsu
Edward Harrison	Anritsu
Shawn Ding	Broadcom
Robert Hulvey	Broadcom
Mark Braun	Motorola
Joel Linsky	Qualcomm
Eran Reuveni	TI

Version 2.1 + EDR

P G Madhavan	Agere
Ayse Findikli	Atheros
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Robin Heydon	CSR
Henrik Hedlund	CSR
Simon Kingston	CSR
Steven Singer	CSR
Steven Wenham	CSR
Paul Wright	CSR
Dave Suvak	iAnywhere
Selim Aissi	Intel
Penny Chen	Intel
Mattias Edlund	Infineon
Josh Benaloh	Microsoft
Andy Glass	Microsoft
Peter Hauser	Microsoft
Joby Lafky	Microsoft
Kristin Lauter	Microsoft
Dan Simon	Microsoft
Don Stanwyck	Microsoft
Yacov Yacobi	Microsoft
Gideon Yuval	Microsoft

Peter Hauser	Microsoft
Greg Muchnik	Motorola
N Asokan	Nokia
Philip Ginzboorg	Nokia
Kanji Kerai	Nokia
Noel Lobo	Nokia
Kaisa Nyberg	Nokia
Arto Palin	Nokia
Päivi Ruuska	Nokia
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Dominique Everaere	NXP
Javier del Prado Pavon	NXP
Reinhard Meindl	NXP
Jorgen van Parijs	ST
Guido Bertoni	ST
Tim Howes	Symbian
Amihai Kidron	Texas Instruments
Eran Reuveni	Texas Instruments
Shimojo Yoshimitsu	Toshiba

Version 2.0 + EDR

John Mersh	TTPCom Ltd.
Joel Linsky	RF Micro Devices
Harald Kafemann	Nokia
Simon Morris	CSR

Version 1.2

Jennifer Bray	CSR
Robin Heydon	CSR
Simon Morris	CSR
Alexander Thoukydides	CSR
Kim Schneider	Digianswer/Motorola
Knud Dyring-Olsen	Digianswer/Motorola
Henrik Andersen	Digianswer/Motorola
Jan Åberg	Ericsson
Martin van der Zee	Ericsson
Roland Hellfajer	Infineon
YC Maa	Integrated Programmable Communications, Inc.
Steve McGowan	Intel
Tod Sizer	Lucent Technologies
Adrian Stephens	Mobilian
Jürgen Schnitzler	Nokia
Thomas Müller	Nokia
Carmen Kuhl	Nokia

Arto Palin	Nokia
Thomas Müller	Nokia
Roland Matthijssen	Philips
Rob Davies	Philips
Harmke de Groot	Philips
Antonio Salloum	Philips
Joel Linsky	Silicon Wave
Terry Bourk	Silicon Wave
Yariv Raveh	Texas Instruments
Tally Shina	Texas Instruments
Amihai Kidron	Texas Instruments
Yoshimitsu Shimojo	Toshiba
Toshiki Kizu	Toshiba
John Mersh (section owner)	TTPCom
Sam Turner	TTPCom

Previous versions

Kim Schneider	Digianswer A/S
Toru Aihara	IBM Corporation
Chatschik Bisdikian	IBM Corporation
Kris Fleming	Intel Corporation
David E. Cypher	NIST
Thomas Busse	Nokia Corporation
Julien Corthial	Nokia Corporation
Olaf Joeressen	Nokia Corporation
Thomas Müller	Nokia Corporation
Dong Nguyen	Nokia Corporation
Harmke de Groot	Philips
Terry Bourk	Silicon Wave
Johannes Elg	Telefonaktiebolaget LM Ericsson
Jaap Haartsen	Telefonaktiebolaget LM Ericsson
Tobias Melin (Section Owner)	Telefonaktiebolaget LM Ericsson
Mary A. DuVal	Texas Instruments
Onn Haran	Texas Instruments
John Mersh	TTPCom

2.3.4 Part D: Error Codes

Version 4.0

Robert Hulvey	Broadcom
Steven Wenham	CSR
Kanji Kerai	Nokia
Steve Davies	Nokia
Tim Howes	Nokia
Brian Redding	Qualcomm
Joel Linsky	Qualcomm
Geert Sonck	ST Ericsson

Version 3.0 + HS

Phil Hough	Anritsu
Kevin Hayes	Atheros
Stratos Chatzikyriakos	Artimi
Shawn Ding	Broadcom
Joe Decuir	CSR
David Suvak	iAnywhere
Koen Derom	NXP Semiconductors
Joel Linsky	Qualcomm
Krishnan Rajamani	Qualcomm
John Hillan	Qualcomm
Mayank Sharma	SiRF
Jason Hillyard	Staccato Communications
William Stoye	Staccato Communications

Version 2.1 + EDR

Ayse Findikli	Atheros
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Robin Heydon	CSR
Simon Kingston	CSR
Steven Singer	CSR
Steven Wenham	CSR
Selim Aissi	Intel
Penny Chen	Intel
Mattias Edlund	Infineon
Josh Benaloh	Microsoft
Andy Glass	Microsoft
Peter Hauser	Microsoft
Joby Lafky	Microsoft
Kristin Lauter	Microsoft
Dan Simon	Microsoft
Don Stanwyck	Microsoft

Yacov Yacobi	Microsoft
Gideon Yuval	Microsoft
Greg Muchnik	Motorola
N Asokan	Nokia
Philip Ginzboorg	Nokia
Kanji Kerai	Nokia
Noel Lobo	Nokia
Kaisa Nyberg	Nokia
Arto Palin	Nokia
Päivi Ruuska	Nokia
Dominique Everaere	NXP
Reinhard Meindl	NXP
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Guido Bertoni	ST
Eran Reuveni	Texas Instruments
Shimojo Yoshimitsu	Toshiba

Version 1.2

Robin Heydon (section owner)	CSR
Roland Hellfajer	Infineon
Joel Linsky	Silicon Wave
John Mersh	TTPCom

This part was earlier included in the LMP and HCI functional Specifications.

2.3.5 Part E: Bluetooth Host Controller Interface Functional Specification

Version 4.0

Edward Harrison	Anritsu
Kevin Hayes	Atheros
Joe Decuir	CSR
Robin Heydon	CSR
Steven Wenham	CSR
Magnus Eriksson	Infineon
James Dent	Nokia
James Steele	Nokia
Jonathan Tanner	Nokia
Steve Davies	Nokia
Brian Redding	Qualcomm
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Len Ott	Socket Mobile
Geert Sonck	ST Ericsson

Aaron Atlas	Texas Instruments
Anthony Viscardi	Texas Instruments

Version 3.0 + HS

Phil Hough	Anritsu
Kevin Hayes	Atheros
Stratos Chatzikyriakos	Artimi
Angel Polo	Broadcom
Shawn Ding	Broadcom
Victor Zhodzishsky	Broadcom
Joe Decuir	CSR
David Suvak	iAnywhere
Dominique Everaere	ST-NXP Wireless
Yao Wang	IVT
Koen Derom	NXP Semiconductors
Ana Donezar Ibanez	Parrot
Joel Linsky	Qualcomm
John Hillan	Qualcomm
Krishnan Rajamani	Qualcomm
Mayank Sharma	SiRF
Jason Hillyard	Staccato Communications
William Stoye	Staccato Communications
Doug Clark	Symbian

Version 2.1 + EDR

Ayse Findikli	Atheros
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Henrik Hedlund	CSR
Robin Heydon	CSR
Simon Kingston	CSR
Steven Singer	CSR
Steven Wenham	CSR
Paul Wright	CSR
Mattias Edlund	Infineon
David Suvak	iAnywhere
Selim Aissi	Intel
Penny Chen	Intel
Josh Benaloh	Microsoft
Andy Glass	Microsoft
Peter Hauser	Microsoft
Joby Lafky	Microsoft
Kristin Lauter	Microsoft
Dan Simon	Microsoft
Don Stanwyck	Microsoft

Yacov Yacobi	Microsoft
Gideon Yuval	Microsoft
Greg Muchnik	Motorola
N Asokan	Nokia
Philip Ginzboorg	Nokia
Kanji Kerai	Nokia
Noel Lobo	Nokia
Kaisa Nyberg	Nokia
Arto Palin	Nokia
Päivi Ruuska	Nokia
Dominique Everaere	NXP
Javier del Prado Pavon	NXP
Reinhard Meindl	NXP
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Tim Howes	Symbian
Jorgen van Parijs	ST
Guido Bertoni	ST
Amihai Kidron	Texas Instruments
Eran Reuveni	Texas Instruments
Shimojo Yoshimitsu	Toshiba

Version 2.0 + EDR

Neil Stewart	Tality UK, Ltd.
Joel Linsky	RF Micro Devices
Robin Heydon	CSR

Version 1.2

Robin Heydon (section owner	CSR
Jennifer Bray	CSR
Alexander Thoukydides	CSR
Knud Dyring-Olsen	Digianswer/Motorola
Henrik Andersen	Digianswer/Motorola
Jan Åberg	Ericsson
Martin van der Zee	Ericsson
Don Liechty	Extended Systems
Kevin Marquess	Hyper Corp
Roland Hellfajer	Infineon
YC Maa	Integrated Programmable Communications, Inc.
Steve McGowan	Intel
Tod Sizer	Lucent Technologies
Tsuyoshi Okada	Matsushita Electric Industrial Co. Ltd
Andy Glass	Microsoft
Adrian Stephens	Mobilian
Jürgen Schnitzler	Nokia

Thomas Müller	Nokia
Rene Tischer	Nokia
Rob Davies	Philips
Antonio Salloum	Philips
Joel Linsky	Silicon Wave
Terry Bourk	Silicon Wave
Len Ott	Socket Communications
Randy Erman	Taiyo Yuden
Yoshimitsu Shimojo	Toshiba
Toshiki Kizu	Toshiba
Katsuhiro Kinoshita	Toshiba
Sam Turner	TTPCom
John Mersh	TTPCom

Previous versions

Todor Cooklev	3Com Corporation
Toru Aihara	IBM Corporation
Chatschik Bisdikian	IBM Corporation
Nathan Lee	IBM Corporation
Akihiko Mizutani	IBM Corporation
Les Cline	Intel Corporation
Bailey Cross	Intel Corporation
Kris Fleming	Intel Corporation
Robert Hunter	Intel Corporation
Jon Inouye	Intel Corporation
Srikanth Kambhatla	Intel Corporation
Steve Lo	Intel Corporation
Vijay Suthar	Intel Corporation
Bruce P. Kraemer	Intersil
Greg Muchnik	Motorola, Inc.
David E. Cypher	NIST
Thomas Busse	Nokia Corporation
Julien Courthial	Nokia Corporation
Thomas Müller	Nokia Corporation
Dong Nguyen	Nokia Corporation
Jürgen Schnitzler	Nokia Corporation
Fujio Watanabe	Nokia Corporation
Christian Zechlin	Nokia Corporation
Johannes Elg	Telefonaktiebolaget LM Ericsson
Christian Johansson (Section Owner)	Telefonaktiebolaget LM Ericsson
Patrik Lundin	Telefonaktiebolaget LM Ericsson
Tobias Melin	Telefonaktiebolaget LM Ericsson
Mary A. DuVal	Texas Instruments
Thomas M. Siep	Texas Instruments
Masahiro Tada	Toshiba Corporation
John Mersh	TTPCom

2.3.6 Part F: Message Sequence Charts

Version 3.0 + HS

Phil Hough	Anritsu
Kevin Hayes	Atheros
Stratos Chatzikyriakos	Artimi
Shawn Ding	Broadcom
Joe Decuir	CSR
David Suvak	iAnywhere
Koen Derom	NXP Semiconductors
Joel Linsky	Qualcomm
Krishnan Rajamani	Qualcomm
John Hillan	Qualcomm
Mayank Sharma	SiRF
Jason Hillyard	Staccato Communications
William Stoye	Staccato Communications

Version 2.1 + EDR

Ayse Findikli	Atheros
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Henrik Hedlund	CSR
Robin Heydon	CSR
Simon Kingston	CSR
Steven Singer	CSR
Steven Wenham	CSR
Selim Aissi	Intel
Penny Chen	Intel
Mattias Edlund	Infineon
Josh Benaloh	Microsoft
Andy Glass	Microsoft
Peter Hauser	Microsoft
Joby Lafky	Microsoft
Kristin Lauter	Microsoft
Dan Simon	Microsoft
Don Stanwyck	Microsoft
Yacov Yacobi	Microsoft
Gideon Yuval	Microsoft
Greg Muchnik	Motorola
N Asokan	Nokia
Philip Ginzboorg	Nokia
Kanji Kerai	Nokia
Noel Lobo	Nokia
Kaisa Nyberg	Nokia
Päivi Ruuska	Nokia

Dominique Everaere	NXP
Reinhard Meindl	NXP
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Guido Bertoni	ST
Tim Howes	Symbian
Amihai Kidron	Texas Instruments
Eran Reuveni	Texas Instruments
Shimojo Yoshimitsu	Toshiba

Version 1.2

Tom Siep	Bluetooth SIG Inc.
Robin Heydon (section owner)	CSR
Simon Morris	CSR
Jan Åberg	Ericsson
Christian Gehrmann	Ericsson
Joel Linsky	Silicon Wave
John Mersh	TTPCom

Previous versions

Todor Cooklev	3Com Corporation
Toru Aihara	IBM Corporation
Chatschik Bisdikian	IBM Corporation
Nathan Lee	IBM Corporation
Kris Fleming	Intel Corporation
Greg Muchnik	Motorola, Inc.
David E. Cypher	NIST
Thomas Busse	Nokia Corporation
Dong Nguyen (Section Owner)	Nokia Corporation
Fujio Watanabe	Nokia Corporation
Christian Johansson	Telefonaktiebolaget LM Ericsson
Tobias Melin	Telefonaktiebolaget LM Ericsson
Mary A. DuVal	Texas Instruments

2.3.7 Part G: Sample Data

Version 3.0 + HS

Kevin Hayes	Atheros
Robert Hulvey	Broadcom
Yao Wang	IVT
Joel Linsky	Qualcomm
John Saad	Qualcomm

Version 2.1 + EDR

Ayse Findikli	Atheros
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Robin Heydon	CSR
Simon Kingston	CSR
Steven Singer	CSR
Steven Wenham	CSR
Selim Aissi	Intel
Penny Chen	Intel
Josh Benaloh	Microsoft
Andy Glass	Microsoft
Peter Hauser	Microsoft
Joby Lafky	Microsoft
Kristin Lauter	Microsoft
Dan Simon	Microsoft
Don Stanwyck	Microsoft
Yacov Yacobi	Microsoft
Gideon Yuval	Microsoft
Greg Muchnik	Microsoft
N Asokan	Nokia
Philip Ginzboorg	Nokia
Kanji Kerai	Nokia
Noel Lobo	Nokia
Kaisa Nyberg	Nokia
Arto Palin	Nokia
Päivi Ruuska	Nokia
Dominique Everaere	NXP
Reinhard Meindl	NXP
Guido Bertoni	ST
Eran Reuveni	Texas Instruments
Shimojo Yoshimitsu	Toshiba

Version 1.2

Joel Linsky	Silicon Wave

Previous versions

Thomas Müller	Nokia Corporation
Thomas Sander	Nokia Corporation
Joakim Persson (Section Owner)	Telefonaktiebolaget LM Ericsson

2.3.8 Part H: Security Specification

Version 3.0 + HS

Phil Hough	Anritsu
Edward Harrison	Anritsu
Kevin Hayes	Atheros
Shawn Ding	Broadcom
Robert Hulvey	Broadcom
Yao Wang	IVT
Mark Braun	Motorola
Kaisa Nyberg	Nokia
John Saad	Qualcomm
Joel Linsky	Qualcomm
Eran Reuveni	TI

Version 2.1 + EDR

Ayse Findikli	Atheros
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Henrik Hedlund	CSR
Robin Heydon	CSR
Simon Kingston	CSR
Steven Singer	CSR
Steven Wenham	CSR
Selim Aissi	Intel
Penny Chen	Intel
Josh Benaloh	Microsoft
Andy Glass	Microsoft
Peter Hauser	Microsoft
Joby Lafky	Microsoft
Kristin Lauter	Microsoft
Dan Simon	Microsoft
Don Stanwyck	Microsoft
Yacov Yacobi	Microsoft
Gideon Yuval	Microsoft
Greg Muchnik	Microsoft
Mattias Edlund	Infineon
N Asokan	Nokia
Philip Ginzboorg	Nokia
Kanji Kerai	Nokia
Noel Lobo	Nokia
Kaisa Nyberg	Nokia
Arto Palin	Nokia
Päivi Ruuska	Nokia
Dominique Everaere	NXP

Reinhard Meindl	NXP
Joel Linsky	QUALCOMM
Terry Bourk	QUALCOMM
Guido Bertoni	ST
Amihai Kidron	Texas Instruments
Eran Reuveni	Texas Instruments
Shimojo Yoshimitsu	Toshiba

2.4 [VOL 3] CORE SYSTEM PACKAGE, HOST

2.4.1 Part A: Logical Link Control and Adaptation Protocol Specification

Version 4.0

Xavier Boniface	Alpwise
Alexandre Gimard	Alpwise
Ash Kapur	Broadcom
Robert Hulvey	Broadcom
Joe Decuir	CSR
Laurence Jupp	CSR
Magnus Sommansson	CSR
Robin Heydon	CSR
Dave Suvak	iAnywhere
James Dent	Nokia
James Steele	Nokia
Jonathan Tanner	Nokia
Kanji Kerai	Nokia
Miika Laaksonen	Nokia
Steve Davies	Nokia
Tim Howes	Nokia
David Lopes	Nordic Semiconductor
Brian Redding	Qualcomm
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Andy Estrada	Sony
Karl Torvmark	Texas Instruments

Version 3.0 + HS

Victor Zhodzishsky	Broadcom
Angel Polo	Broadcom
Ash Kapur	Broadcom
Robert Hulvey	Broadcom
Ken Steck	CSR
Dave Suvak	iAnywhere
Yao Wang	IVT
Jacques Chassot	Logitech
Nathan Sherman	Microsoft
Sandy Spinrad	Microsoft
Doug Clarke	Nokia
Kanji Kerai	Nokia
Ana Donezar Ibanez	Parrot
Joel Linksy	Qualcomm

Andy Estrada	Sony
Dominique Everaere	ST-NXP Wireless

Version 2.1 + EDR

Ayse Findikli	Atheros
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Robin Heydon	CSR
Simon Kingston	CSR
Steven Singer	CSR
Steven Wenham	CSR
Selim Aissi	Intel
Penny Chen	Intel
Dave Suvak	iAnywhere
Josh Benaloh	Microsoft
Andy Glass	Microsoft
Peter Hauser	Microsoft
Joby Lafky	Microsoft
Kristin Lauter	Microsoft
Dan Simon	Microsoft
Don Stanwyck	Microsoft
Yacov Yacobi	Microsoft
Gideon Yuval	Microsoft
Greg Muchnik	Microsoft
N Asokan	Nokia
Philip Ginzboorg	Nokia
Kanji Kerai	Nokia
Noel Lobo	Nokia
Kaisa Nyberg	Nokia
Dominique Everaere	NXP
Javier del Prado Pavon	NXP
Reinhard Meindl	NXP
Joel Linsky	QUALCOMM
Terry Bourk	QUALCOMM
Tim Howes	Symbian
Jorgen van Parijs	ST
Amihai Kidron	Texas Instruments
Shimojo Yoshimitsu	Toshiba

Version 1.2

Tom Siep	Bluetooth SIG Inc.
Carsten Andersen (section owner	CSR
Jennifer Bray	CSR
Jan Åberg	Ericsson
Martin van der Zee	Ericsson
Sam Ravnborg	Ericsson
Stefan Agnani	Ericsson
Steve McGowan	Intel Corporation
Joby Lafky	Microsoft
Doron Holan	Microsoft
Andy Glass	Microsoft
Brian Redding	Motorola
Jürgen Schnitzler	Nokia
Thomas Müller	Nokia
Rob Davies	Philips
Terry Bourk	Silicon Wave
Michael Hasling	Tality

Previous versions

Jon Burgess	3Com Corporation
Paul Moran	3Com Corporation
Doug Kogan	Extended Systems
Kevin D. Marquess	Hyper Corporation
Toru Aihara	IBM Corporation
Chatschik Bisdikian	IBM Corporation
Kris Fleming	Intel Corporation
Uma Gadamsetty	Intel Corporation
Robert Hunter	Intel Corporation
Jon Inouye	Intel Corporation
Steve C. Lo	Intel Corporation
Chunrong Zhu	Intel Corporation
Sergey Solyanik	Microsoft Corporation
David E. Cypher	NIST
Nada Golmie	NIST
Thomas Busse	Nokia Corporation
Rauno Makinen	Nokia Corporation
Thomas Müller	Nokia Corporation
Petri Nykänen	Nokia Corporation
Peter Ollikainen	Nokia Corporation
Petri O. Nurminen	Nokia Corporation
Johannes Elg	Telefonaktiebolaget LM Ericsson
Jaap Haartsen	Telefonaktiebolaget LM Ericsson
Elco Nijboer	Telefonaktiebolaget LM Ericsson
Ingemar Nilsson	Telefonaktiebolaget LM Ericsson

Stefan Runesson	Telefonaktiebolaget LM Ericsson
Gerrit Slot	Telefonaktiebolaget LM Ericsson
Johan Sörensen	Telefonaktiebolaget LM Ericsson
Goran Svennarp	Telefonaktiebolaget LM Ericsson
Mary A. DuVal	Texas Instruments
Thomas M. Siep	Texas Instruments
Kinoshita Katsuhiro	Toshiba Corporation

2.4.2 Part B: Service Discovery Protocol (SDP)

Version 2.1 + EDR

Robin Heydon	CSR
Mattias Edlund	Infineon
Päivi Ruuska	Nokia
Arto Palin	Nokia
Joel Linsky	QUALCOMM
Terry Bourk	QUALCOMM
Amihai Kidron	Texas Instruments
Shimojo Yoshimitsu	Toshiba

Previous versions

Ned Plasson	3Com Corporation
John Avery	Convergence
Jason Kronz	Convergence
Chatschik Bisdikian	IBM Corporation
Parviz Kermani	IBM Corporation
Brent Miller	IBM Corporation
Dick Osterman	IBM Corporation
Bob Pascoe	IBM Corporation
Jon Inouye	Intel Corporation
Srikanth Kambhatla	Intel Corporation
Jay Eaglstun	Motorola, Inc.
Dale Farnsworth (Section Owner)	Motorola, Inc.
Jean-Michel Rosso	Motorola, Inc.
Jan Grönholm	Nokia Corporation
Kati Rantala	Nokia Corporation
Thomas Müller	Nokia Corporation
Johannes Elg	Telefonaktiebolaget LM Ericsson
Kazuaki Iwamura	Toshiba Corporation

2.4.3 Part C: Generic Access Profile

Version 4.0

Alexandre Gimard	Alpwise
Xavier Boniface	Alpwise
Mike Tsai	Atheros
John Padgette	Booz Allen Hamilton
Ash Kapur	Broadcom
Norbert Grunert	Broadcom
Robert Hulvey	Broadcom
Shawn Ding	Broadcom
Burch Seymour	Continental Automotive
Joe Decuir	CSR
Magnus Sommansson	CSR
Nick Hunn	CSR
Robin Heydon	CSR
Steven Wenham	CSR
David Suvak	iAnywhere
Chiu-Mien Chi	ISSC
Anindya Bakshi	MindTree
Ashok Kelur	MindTree
John Barr	Motorola
Michael Russell	Motorola
James Dent	Nokia
James Steele	Nokia
Jonathan Tanner	Nokia
Kanji Kerai	Nokia
Miika Laaksonen	Nokia
Steve Davies	Nokia
Tim Howes	Nokia
David Lopes	Nordic Semiconductor
Brian Redding	Qualcomm
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Harsha Master	Sasken
Len Ott	Socket Mobile
Frédéric Virot	ST Ericsson
Geert Sonck	ST Ericsson
Pär-Gunnar Hjälmdahl	ST Ericsson
Aaron Atlas	Texas Instruments

Version 3.0 + HS

Robert Hulvey	Broadcom
Ken Steck	CSR
Jacques Chassot	Logitech
Nathan Sherman	Microsoft
Sandy Spinrad	Microsoft
Kanji Kerai	Nokia
Joel Linksy	Qualcomm
Andy Estrada	Sony

Version 2.1 + EDR

Robin Heydon	CSR
Mattias Edlund	Infineon
Päivi Ruuska	Nokia
Arto Palin	Nokia
Joel Linsky	QUALCOMM
Terry Bourk	QUALCOMM
Shimojo Yoshimitsu	Toshiba

Version 1.2

Jennifer Bray	CSR
Alexander Thoukydides	CSR
Christian Gehrmann	Ericsson
Henrik Hedlund	Ericsson
Jan Åberg	Ericsson
Stefan Agnani	Ericsson
Thomas Müller	Nokia
Joel Linsky	Silicon Wave
Terry Bourk	Silicon Wave
Katsuhiro Kinoshita	Toshiba

Previous versions

Ken Morley	3Com Corporation
Chatschik Bisdikian	IBM Corporation
Jon Inouye	Intel Corporation
Brian Redding	Motorola, Inc.
David E. Cypher	NIST
Stephane Bouet	Nokia Corporation
Thomas Müller	Nokia Corporation
Martin Roter	Nokia Corporation
Johannes Elg	Telefonaktiebolaget LM Ericsson
Patric Lind (Section Owner)	Telefonaktiebolaget LM Ericsson
Erik Slotboom	Telefonaktiebolaget LM Ericsson
Johan Sörensen	Telefonaktiebolaget LM Ericsson

2.4.4 Part D: Test Support

Version 3.0 + HS

Phil Hough	Anritsu
Edward Harrison	Anritsu
Angus Robinson	Anritsu
Kevin Hayes	Atheros
Stratos Chatzikyriakos	Artimi
Shawn Ding	Broadcom
Robert Hulvey	Broadcom
Yao Wang	IVT
Joe Decuir	CSR
Mark Braun	Motorola
Joel Linsky	Qualcomm
Eran Reuveni	TI

Version 1.2

Emilio Mira Escartis	Cetecom
Robin Heydon	CSR
Jennifer Bray	CSR
Stefan Agnani (section owner)	Ericsson
Terry Bourk	Silicon Wave
Joel Linsky	Silicon Wave
Michael Hasling	Tality

Previous versions [Test Mode]

Jeffrey Schiffer	Intel Corporation
David E. Cypher	NIST
Daniel Bencak	Nokia Corporation
Arno Kefenbaum	Nokia Corporation
Thomas Müller (Section Owner)	Nokia Corporation
Roland Schmale	Nokia Corporation
Fujio Watanabe	Nokia Corporation
Stefan Agnani	Telefonaktiebolaget LM Ericsson
Mårten Mattsson	Telefonaktiebolaget LM Ericsson
Tobias Melin	Telefonaktiebolaget LM Ericsson
Lars Nord	Telefonaktiebolaget LM Ericsson
Fredrik Töörn	Telefonaktiebolaget LM Ericsson
John Mersh	TTPCom
Ayse Findikli	Zeevo, Inc.

Previous versions [Test Control Interface]

Mike Feldman	Motorola, Inc.
Thomas Müller	Nokia Corporation
Stefan Agnani (Section Owner)	Telefonaktiebolaget LM Ericsson

Mårten Mattsson	Telefonaktiebolaget LM Ericsson
Dan Sönnerstam	Telefonaktiebolaget LM Ericsson

2.4.5 Part E: AMP Manager Protocol

Version 3.0 + HS

Kevin Hayes	Atheros
Robert Hulvey	Broadcom
Dave Suvak	iAnywhere
Yao Wang	IVT
Doug Clark	Nokia
Kaisa Nyberg	Nokia
James Steele	Nokia
Joel Linsky	Qualcomm
John Saad	Qualcomm

2.4.6 Part F: Attribute Protocol Specification

Version 4.0

Alexandre Gimard	Alpwise
Xavier Boniface	Alpwise
Ash Kapur	Broadcom
Norbert Grunert	Broadcom
Srikanth Uppala	Broadcom
Burch Seymour	Continental Automotive
Joe Decuir	CSR
Magnus Sommansson	CSR
Nick Hunn	CSR
Robin Heydon	CSR
Morgan Lindqvist	Ericsson
Dave Suvak	iAnywhere
Chris Hansen	Intel
Marcel Holtmann	Intel
Anindya Bakshi	MindTree
John Barr	Motorola
Daidi Zhong	Nokia
James Dent	Nokia
James Steele	Nokia
Jonathan Tanner	Nokia
Juha Salokannel	Nokia
Kanji Kerai	Nokia
Miika Laaksonen	Nokia
Päivi Ruuska	Nokia
Steve Davies	Nokia
Tim Howes	Nokia
David Lopes	Nordic Semiconductor
Niclas Granqvist	Polar
Brian Redding	Qualcomm
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Harsha Master	Sasken
Len Ott	Socket Mobile
Pär-Gunnar Hjälmdahl	ST Ericsson
Aaron Atlas	Texas Instruments

2.4.7 Part G: Generic Attribute Protocol Specification

Alexandre Gimard	Alpwise
Xavier Boniface	Alpwise
Ash Kapur	Broadcom
Norbert Grunert	Broadcom
Srikanth Uppala	Broadcom
Mats Andersson	connectBlue
Burch Seymour	Continental Automotive
Joe Decuir	CSR
Laurence Jupp	CSR
Magnus Sommansson	CSR
Nick Hunn	CSR
Robin Heydon	CSR
Dave Suvak	iAnywhere
Marcel Holtmann	Intel
Anindya Bakshi	MindTree
Krishna Shingala	MindTree
Daidi Zhong	Nokia
James Dent	Nokia
James Steele	Nokia
Jonathan Tanner	Nokia
Juha Salokannel	Nokia
Kanji Kerai	Nokia
Miika Laaksonen	Nokia
Päivi Ruuska	Nokia
Steve Davies	Nokia
Tim Howes	Nokia
David Lopes	Nordic Semiconductor
Sebastien Mackaie-Blanchi	Nordic Semiconductor
Niclas Granqvist	Polar
Brian Redding	Qualcomm
Joel Linksy	Qualcomm
Terry Bourk	Qualcomm
Pär-Gunnar Hjälmdahl	ST Ericsson
Aaron Atlas	Texas Instruments

2.4.8 Part H: Security Manager Specification

Version 4.0

Alexandre Gimard	Alpwise
Mike Tsai	Atheros
John Padgette	Booz Allen Hamilton
Chaojing Sun	Broadcom
Norbert Grunert	Broadcom

Prasanna Desai	Broadcom
Robert Hulvey	Broadcom
Hermanni Suominen	CSR
Laurence Jupp	CSR
Magnus Sommansson	CSR
Robin Heydon	CSR
Steven Wenham	CSR
Patrick Reinelt	Frontline
Chiu-Mien Chi	ISSC
Anindya Bakshi	MindTree
Ashok Kelur	MindTree
Michael Russell	Motorola
James Dent	Nokia
James Steele	Nokia
Jonathan Tanner	Nokia
Kanji Kerai	Nokia
Miika Laaksonen	Nokia
Noel Lobo	Nokia
Steve Davies	Nokia
Tim Howes	Nokia
David Lopes	Nordic Semiconductor
Brian Redding	Qualcomm
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Christian Zechlin	Sasken
Frédéric Virot	ST Ericsson
Geert Sonck	ST Ericsson
Alon Paycher	Texas Instruments
Helge Coward	Texas Instruments

2.5 [VOL 4] HOST CONTROLLER INTERFACE [TRANSPORT LAYER]

Toru Aihara	IBM Corporation
Edgar Kerstan	IBM Corporation
Nathan Lee	IBM Corporation
Kris Fleming	Intel Corporation
Robert Hunter	Intel Corporation
Patrick Kane	Motorola, Inc.
Uwe Gondrum	Nokia Corporation
Thomas Müller	Nokia Corporation
Christian Zechlin	Nokia Corporation
Johannes Elg	Telefonaktiebolaget LM Ericsson
Sven Jerlhagen	Telefonaktiebolaget LM Ericsson
Christian Johansson	Telefonaktiebolaget LM Ericsson
Patrik Lundin	Telefonaktiebolaget LM Ericsson
Lars Novak	Telefonaktiebolaget LM Ericsson
Masahiro Tada	Toshiba Corporation
Steve Ross	Digianswer A/S
Chatschik Bisdikian	IBM Corporation
Les Cline	Intel Corporation
Brad Hosler	Intel Corporation
John Howard	Intel Corporation
Srikanth Kambhatla	Intel Corporation
Kosta Koeman	Intel Corporation
John McGrath	Intel Corporation
Patrik Lundin	Telefonaktiebolaget LM Ericsson
Leonard Ott	Socket Communications
Rebecca O'Dell	Signia Technologies
Tsuyoshi Okada	Matsushita Electric
Robin Heydon	CSR
Toru Aihara	IBM Corporation
Edgar Kerstan	IBM Corporation
Nathan Lee	IBM Corporation
Kris Fleming	Intel Corporation
Robert Hunter	Intel Corporation
Patrick Kane	Motorola, Inc.
Uwe Gondrum	Nokia Corporation
Thomas Müller	Nokia Corporation
Christian Zechlin	Nokia Corporation
Johannes Elg	Telefonaktiebolaget LM Ericsson
Sven Jerlhagen	Telefonaktiebolaget LM Ericsson
Christian Johansson	Telefonaktiebolaget LM Ericsson
Patrik Lundin	Telefonaktiebolaget LM Ericsson
Lars Novak	Telefonaktiebolaget LM Ericsson
Masahiro Tada	Toshiba Corporation

Steve Ross	Digianswer A/S
Chatschik Bisdikian	IBM Corporation
Les Cline	Intel Corporation
Brad Hosler	Intel Corporation
John Howard	Intel Corporation
Srikanth Kambhatla	Intel Corporation
Kosta Koeman	Intel Corporation
John McGrath	Intel Corporation
Patrik Lundin	Telefonaktiebolaget LM Ericsson
Leonard Ott	Socket Communications
Rebecca O'Dell	Signia Technologies
Tsuyoshi Okada	Matsushita Electric
Robin Heydon	CSR

2.6 [VOL 5] CORE SYSTEM PACKAGE [AMP CONTROLLER VOLUME]

2.6.1 Part A: 802.11 PAL

Version 3.0 +HS

Angus Robinson	Anritsu
Edward Harrison	Anritsu
Phil Hough	Anritsu
Stratos Chatzikyriakos	Artimi
Kevin Hayes	Atheros
Prerepa Viswanadham	Atheros
Chris Hansen	Broadcom
Raymond Hayes	Broadcom
Joe Decuir	CSR
Nick Jackson	CSR
David Suvak	iAnywhere
Ganesh Venkatesan	Intel
Quinton Yuan	Marvell
Raja Banerjea	Marvell
John Barr	Motorola
Janne Marin	Nokia
Mika Kasslin	Nokia
Joel Linsky	Qualcomm
Krishnan Rajamani	Qualcomm
Terry Bourk	Qualcomm
Ignacio Gimeno	ST-NXP Wireless
William Stoye	Staccato Communications
Amir Yassur	TI
Yossi Peery	TI

2.7 [VOL 6] LOW ENERGY SPECIFICATION

2.7.1 Part A: Physical Layer Specification

Version 4.0

Edward Harrison	Anritsu
Robert Hulvey	Broadcom
Burch Seymour	Continental Automotive
Magnus Sommansson	CSR
Robin Heydon	CSR
Steven Wenham	CSR
Henrik Arfwedson	Infineon
Jukka Reunamaki	Nokia
Mika Kasslin	Nokia
Steve Davies	Nokia
Frank Karlsen	Nordic Semiconductor
Øyvind Vedal	Nordic Semiconductor
Brian Redding	Qualcomm
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Don Sturek	Texas Instruments

2.7.2 Part B: Link Layer Specification

Version 4.0

Angel Polo	Broadcom
Prasanna Desai	Broadcom
Robert Hulvey	Broadcom
Yuan Zhuang	Broadcom
Burch Seymour	Continental Automotive
Joe Decuir	CSR
Magnus Sommansson	CSR
Robin Heydon	CSR
Steven Singer	CSR
Steven Wenham	CSR
Robert Kvacek	EM Microelectronic
Henrik Arfwedson	Infineon
James Dent	Nokia
Jonathan Tanner	Nokia
Kanji Kerai	Nokia
Mika Kasslin	Nokia
Steve Davies	Nokia

Asbjørn Sæbø	Nordic Semiconductor
David Lopes	Nordic Semiconductor
Frank Karlsen	Nordic Semiconductor
Torstein Nesje	Nordic Semiconductor
Brian Redding	Qualcomm
Joel Linsky	Qualcomm
Terry Bourk	Qualcomm
Harsha Master	Sasken
Geert Sonck	ST Ericsson
Alon Paycher	Texas Instruments
Anthony Viscardi	Texas Instruments
Don Sturek	Texas Instruments
Helge Coward	Texas Instruments

2.7.3 Part C: Sample Data

Version 4.0

Robin Heydon	CSR
Steven Wenham	CSR
Joel Linsky	Qualcomm
Geert Sonck	ST Ericsson

2.7.4 Part D: Message Sequence Charts

Version 4.0

Robin Heydon	CSR
Steven Wenham	CSR
Steve Davies	Nokia
Joel Linsky	Qualcomm
Geert Sonck	ST Ericsson

2.7.5 Part E: Low Energy Security Specification

Version 4.0

Robin Heydon	CSR
Steven Singer	CSR
Steven Wenham	CSR
Mika Kasslin	Nokia
Joel Linsky	Qualcomm
Geert Sonck	ST Ericsson
Alon Paycher	Texas Instruments

2.7.6 Part F: Direct Test Mode

Version 4.0

Edward Harrison	Anritsu
Robert Hulvey	Broadcom
Magnus Sommansson	CSR
Robin Heydon	CSR
Steven Wenham	CSR
Steve Davies	Nokia
Frank Karlsen	Nordic Semiconductor
Joel Linsky	Qualcomm
Helge Coward	Texas Instruments

Ford Ex. 1052



Specification Volume 1

SPECIFICATION OF THE *BLUETOOTH* SYSTEM

Experience More



Architecture & Terminology Overview

Covered Core Package version:
4.0

Bluetooth®

Ford Ex. 1052

Revision History

The Revision History is shown in the [Vol 0] Part C Appendix on page 55.

Contributors

The persons who contributed to this specification are listed in the [Vol 0] Part C Appendix on page 55.

Web Site

This specification can also be found on the official Bluetooth web site:
http://www.bluetooth.com

Disclaimer and Copyright Notice

The copyright in these specifications is owned by the Promoter Members of Bluetooth SIG, Inc. (“Bluetooth SIG”). Use of these specifications and any related intellectual property (collectively, the “Specification”), is governed by the Promoters Membership Agreement among the Promoter Members and Bluetooth SIG (the “Promoters Agreement”), certain membership agreements between Bluetooth SIG and its Adopter and Associate Members (the “Membership Agreements”) and the Bluetooth Specification Early Adopters Agreements (“1.2 Early Adopters Agreements”) among Early Adopter members of the unincorporated Bluetooth special interest group and the Promoter Members (the “Early Adopters Agreement”). Certain rights and obligations of the Promoter Members under the Early Adopters Agreements have been assigned to Bluetooth SIG by the Promoter Members.

Use of the Specification by anyone who is not a member of Bluetooth SIG or a party to an Early Adopters Agreement (each such person or party, a “Member”), is prohibited. The legal rights and obligations of each Member are governed by their applicable Membership Agreement, Early Adopters Agreement or Promoters Agreement. No license, express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

Any use of the Specification not in compliance with the terms of the applicable Membership Agreement, Early Adopters Agreement or Promoters Agreement is prohibited and any such prohibited use may result in termination of the applicable Membership Agreement or Early Adopters Agreement and other liability permitted by the applicable agreement or by applicable law to Bluetooth SIG or any of its members for patent, copyright and/or trademark infringement.

THE SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, SATISFACTORY QUALITY, OR REASONABLE SKILL OR CARE, OR ANY WARRANTY ARISING OUT OF ANY COURSE OF DEALING, USAGE, TRADE PRACTICE, PROPOSAL, SPECIFICATION OR SAMPLE.

Each Member hereby acknowledges that products equipped with the Bluetooth® technology (“Bluetooth® Products”) may be subject to various regulatory controls under the laws and regulations of various governments worldwide. Such laws and regulatory controls may govern, among other things, the combination, operation, use, implementation and distribution of Bluetooth® Products. Examples of such laws and regulatory controls include, but are not limited to, airline regulatory controls, telecommunications regulations, technology transfer controls and health and safety regulations. Each Member is solely responsible for the compliance by their Bluetooth® Products with any such laws and regulations and for obtaining any and all required authorizations, permits, or licenses for their Bluetooth® Products related to such regulations within the applicable jurisdictions. Each Member acknowledges that nothing in the Specification provides any information or assistance in connection with securing such compliance, authorizations or licenses. NOTHING IN THE SPECIFICATION CREATES ANY WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING SUCH LAWS OR REGULATIONS.

ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHTS OR FOR NONCOMPLIANCE WITH LAWS, RELATING TO USE OF THE SPECIFICATION IS EXPRESSLY DISCLAIMED. BY USE OF THE SPECIFICATION, EACH MEMBER EXPRESSLY WAIVES ANY CLAIM AGAINST BLUETOOTH SIG AND ITS PROMOTER MEMBERS RELATED TO USE OF THE SPECIFICATION.

Bluetooth SIG reserves the right to adopt any changes or alterations to the Specification as it deems necessary or appropriate.

Copyright © 1999 - 2010

Ericsson AB,
Lenovo,
Intel Corporation,
Microsoft Corporation,
Motorola, Inc.,
Nokia Corporation,
Toshiba Corporation

*Third-party brands and names are the property of their respective owners.

TABLE OF CONTENTS

Part A
ARCHITECTURE

1 General Description 17
1.1 Overview of BR/EDR Operation 18
1.2 Overview Of Bluetooth Low Energy Operation 20
1.3 Overview Of AMP Operation 22
1.4 Nomenclature 24

2 Core System Architecture 30
2.1 Core Architectural Blocks 34
2.1.1 Host Architectural Blocks 34
2.1.1.1 Channel Manager 34
2.1.1.2 L2CAP Resource Manager 34
2.1.1.3 Security Manager Protocol 34
2.1.1.4 Attribute Protocol 35
2.1.1.5 AMP Manager Protocol 35
2.1.1.6 Generic Attribute Profile 35
2.1.1.7 Generic Access Profile 35
2.1.2 BR/EDR/LE Controller Architectural Blocks 35
2.1.2.1 Device Manager 36
2.1.2.2 Link Manager 36
2.1.2.3 Baseband Resource Manager 36
2.1.2.4 Link Controller 37
2.1.2.5 PHY 37
2.1.3 AMP Controller architectural blocks 37
2.1.3.1 AMP HCI 37
2.1.3.2 AMP PAL 38
2.1.3.3 AMP MAC 38
2.1.3.4 AMP PHY 38

3 Data Transport Architecture 39
3.1 Core Traffic Bearers 40
3.1.1 Framed Data Traffic 41
3.1.2 Unframed Data Traffic 42
3.1.3 Reliability of traffic bearers 43
3.1.3.1 BR/EDR Reliability 43
3.1.3.2 LE reliability 44
3.1.3.3 AMP Reliability 45
3.2 Transport Architecture Entities 45
3.2.1 BR/EDR Generic Packet Structure 46
3.2.2 LE Generic Packet Structure 47
3.3 Physical Channels 49

3.3.1 BR/EDR Physical Channels 49
3.3.1.1 Basic Piconet Channel 50
3.3.1.2 Adapted Piconet Channel 51
3.3.1.3 Inquiry scan channel 52
3.3.1.4 Page Scan Channel 53
3.3.2 LE Physical Channels 54
3.3.2.1 LE piconet channel 55
3.3.2.2 Advertisement broadcast channel 56
3.3.3 AMP physical channel 57
3.3.3.1 Overview 57
3.3.3.2 Characteristics 57
3.3.3.3 Topology 57
3.3.3.4 Supported Layers 58
3.4 Physical Links 58
3.4.1 BR/EDR Links Supported By The Basic And Adapted Piconet Physical Channel58
3.4.1.1 Active Physical Link 58
3.4.1.2 Parked Physical Link 59
3.4.2 BR/EDR Links Supported by the Scanning Physical Channels60
3.4.3 LE Links Supported by the LE Physical Channels 60
3.4.3.1 Active physical link 60
3.4.3.2 Advertising physical link 60
3.4.4 Links Supported by the AMP Physical Channels 61
3.5 Logical Links and Logical Transports 61
3.5.1 Casting 62
3.5.2 Scheduling and Acknowledgement Scheme 63
3.5.3 Class of Data 63
3.5.4 Logical Transports 64
3.5.4.1 BR/EDR Asynchronous Connection-oriented (ACL) 64
3.5.4.2 BR/EDR Synchronous Connection-Oriented (SCO) 65
3.5.4.3 BR/EDR Extended Synchronous Connection-Oriented (eSCO) 65
3.5.4.4 BR/EDR Active Slave Broadcast (ASB) 66
3.5.4.5 BR/EDR Parked Slave Broadcast (PSB) 67
3.5.4.6 LE Asynchronous Connection (LE ACL) 68
3.5.4.7 LE Advertising Broadcast (ADVB) 68
3.5.5 Logical Links 69
3.5.5.1 BR/EDR Logical Links 69
3.5.5.2 LE Logical Links 70
3.5.5.3 AMP Logical Links 71
3.6 L2CAP Channels 72

4 Communication Topology and Operation 73
4.1 Piconet Topology 73
4.1.1 BR/EDR Topology 73
4.1.2 LE Topology 75
4.2 Operational Procedures and Modes 76
4.2.1 BR/EDR Procedures 76
4.2.1.1 Inquiry (Discovering) Procedure 76
4.2.1.2 Paging (Connecting) Procedure 77
4.2.1.3 Connected Mode 77
4.2.1.4 Hold Mode 78
4.2.1.5 Sniff Mode 78
4.2.1.6 Parked State 79
4.2.1.7 Role switch procedure 79
4.2.1.8 Enhanced Data Rate 80
4.2.2 LE Procedures 80
4.2.2.1 Device Filtering Procedure 80
4.2.2.2 Advertising Procedure 81
4.2.2.3 Scanning Procedure 81
4.2.2.4 Discovering Procedure 82
4.2.2.5 Connecting Procedure 82
4.2.2.6 Connected Mode 83
4.2.3 AMP Procedures 83
4.2.3.1 AMP Discovery Procedures 83
4.2.3.2 Physical Link Creation Procedure 84
4.2.3.3 Logical Link Creation Procedure 84
5 Security Overview 85
5.1 BR/EDR Secure Simple Pairing 85
5.1.1 Security Goals 85
5.1.2 Passive Eavesdropping Protection 85
5.1.3 Man-In-The-Middle Protection 86
5.1.4 Association Models 87
5.1.4.1 Numeric Comparison 87
5.1.4.2 Just Works 87
5.1.4.3 Out of Band 88
5.1.4.4 Passkey Entry 88
5.1.4.5 Association Model Overview 89
5.2 LE Security 89
5.2.1 Association Models 90
5.2.2 Key Generation 90
5.2.3 Encryption 90
5.2.4 Signed Data 90
5.2.5 Privacy Feature 91
5.3 AMP Security 91

6 Bluetooth Application Architecture 93
6.1 Bluetooth Profiles 93
6.2 Generic Access Profile 93
6.3 Profile Hierarchy 94
6.4 Generic Attribute Profile 95
6.5 GATT-based Profile Hierarchy 96
6.5.1 Service 96
6.5.2 Referenced Services 97
6.5.3 Characteristic 97

Part B
ACRONYMS & ABBREVIATIONS

1 List of Acronyms and Abbreviations 101

Part C
CORE SPECIFICATION CHANGE HISTORY

1 Deprecated Features 116
2 Changes from V1.1 to V1.2 117
2.1 New Features 117
2.2 Structure Changes 117
2.3 Deprecated Features list 117
2.4 Changes in Wording 118
2.5 Nomenclature Changes 118
3 Changes from V1.2 to V2.0 + EDR 119
3.1 New Features 119
3.2 Deprecated Features 119
4 Changes from V2.0 + EDR to V2.1 + EDR 120
4.1 New features 120
4.2 Deprecated Features 120
5 Changes From V2.1 + EDR To V3.0 + HS 121
5.1 New Features 121
5.2 Deprecated Features 121
6 Changes From V3.0 + HS To v4.0 122
6.1 New Features 122
6.2 Deprecated Features 122

Part D
MIXING OF SPECIFICATION VERSIONS

1 Mixing of Specification Versions 126
1.1 Features and their Types 127

1.2 Core Specification Addendums 128

Part E
IEEE LANGUAGE

1 Use of IEEE Language 135
1.1 Shall 135
1.2 Must 136
1.3 Will 136
1.4 Should 136
1.5 May 136
1.6 Can 137

ARCHITECTURE

CONTENTS

1 General Description 17
1.1 Overview of BR/EDR Operation 18
1.2 Overview Of Bluetooth Low Energy Operation 20
1.3 Overview Of AMP Operation 22
1.4 Nomenclature 24
2 Core System Architecture 30
2.1 Core Architectural Blocks 34
2.1.1 Host Architectural Blocks 34
2.1.1.1 Channel Manager 34
2.1.1.2 L2CAP Resource Manager 34
2.1.1.3 Security Manager Protocol 34
2.1.1.4 Attribute Protocol 35
2.1.1.5 AMP Manager Protocol 35
2.1.1.6 Generic Attribute Profile 35
2.1.1.7 Generic Access Profile 35
2.1.2 BR/EDR/LE Controller Architectural Blocks 35
2.1.2.1 Device Manager 36
2.1.2.2 Link Manager 36
2.1.2.3 Baseband Resource Manager 36
2.1.2.4 Link Controller 37
2.1.2.5 PHY 37
2.1.3 AMP Controller architectural blocks 37
2.1.3.1 AMP HCI 37
2.1.3.2 AMP PAL 38
2.1.3.3 AMP MAC 38
2.1.3.4 AMP PHY 38
3 Data Transport Architecture 39
3.1 Core Traffic Bearers 40
3.1.1 Framed Data Traffic 41
3.1.2 Unframed Data Traffic 42
3.1.3 Reliability of traffic bearers 43
3.1.3.1 BR/EDR Reliability 43
3.1.3.2 LE reliability 44
3.1.3.3 AMP Reliability 45
3.2 Transport Architecture Entities 45
3.2.1 BR/EDR Generic Packet Structure 46
3.2.2 LE Generic Packet Structure 47
3.3 Physical Channels 49
3.3.1 BR/EDR Physical Channels 49
3.3.1.1 Basic Piconet Channel 50

3.3.1.2 Adapted Piconet Channel 51
3.3.1.3 Inquiry scan channel 52
3.3.1.4 Page Scan Channel 53
3.3.2 LE Physical Channels 54
3.3.2.1 LE piconet channel 55
3.3.2.2 Advertisement broadcast channel 56
3.3.3 AMP physical channel 57
3.3.3.1 Overview 57
3.3.3.2 Characteristics 57
3.3.3.3 Topology 57
3.3.3.4 Supported Layers 58
3.4 Physical Links 58
3.4.1 BR/EDR Links Supported By The Basic And Adapted Piconet Physical Channel58
3.4.1.1 Active Physical Link 58
3.4.1.2 Parked Physical Link 59
3.4.2 BR/EDR Links Supported by the Scanning Physical Channels60
3.4.3 LE Links Supported by the LE Physical Channels 60
3.4.3.1 Active physical link 60
3.4.3.2 Advertising physical link 60
3.4.4 Links Supported by the AMP Physical Channels 61
3.5 Logical Links and Logical Transports 61
3.5.1 Casting 62
3.5.2 Scheduling and Acknowledgement Scheme 63
3.5.3 Class of Data 63
3.5.4 Logical Transports 64
3.5.4.1 BR/EDR Asynchronous Connection-oriented (ACL)64
3.5.4.2 BR/EDR Synchronous Connection-Oriented (SCO)65
3.5.4.3 BR/EDR Extended Synchronous Connection-Oriented (eSCO)65
3.5.4.4 BR/EDR Active Slave Broadcast (ASB) 66
3.5.4.5 BR/EDR Parked Slave Broadcast (PSB) 67
3.5.4.6 LE Asynchronous Connection (LE ACL) 68
3.5.4.7 LE Advertising Broadcast (ADVB) 68
3.5.5 Logical Links 69
3.5.5.1 BR/EDR Logical Links 69
3.5.5.2 LE Logical Links 70
3.5.5.3 AMP Logical Links 71
3.6 L2CAP Channels 72

4 Communication Topology and Operation 73

4.1 Piconet Topology 73
4.1.1 BR/EDR Topology 73
4.1.2 LE Topology 75
4.2 Operational Procedures and Modes 76
4.2.1 BR/EDR Procedures 76
4.2.1.1 Inquiry (Discovering) Procedure 76
4.2.1.2 Paging (Connecting) Procedure 77
4.2.1.3 Connected Mode 77
4.2.1.4 Hold Mode 78
4.2.1.5 Sniff Mode 78
4.2.1.6 Parked State 79
4.2.1.7 Role switch procedure 79
4.2.1.8 Enhanced Data Rate 80
4.2.2 LE Procedures 80
4.2.2.1 Device Filtering Procedure 80
4.2.2.2 Advertising Procedure 81
4.2.2.3 Scanning Procedure 81
4.2.2.4 Discovering Procedure 82
4.2.2.5 Connecting Procedure 82
4.2.2.6 Connected Mode 83
4.2.3 AMP Procedures 83
4.2.3.1 AMP Discovery Procedures 83
4.2.3.2 Physical Link Creation Procedure 84
4.2.3.3 Logical Link Creation Procedure 84

5 Security Overview 85

5.1 BR/EDR Secure Simple Pairing 85
5.1.1 Security Goals 85
5.1.2 Passive Eavesdropping Protection 85
5.1.3 Man-In-The-Middle Protection 86
5.1.4 Association Models 87
5.1.4.1 Numeric Comparison 87
5.1.4.2 Just Works 87
5.1.4.3 Out of Band 88
5.1.4.4 Passkey Entry 88
5.1.4.5 Association Model Overview 89
5.2 LE Security 89
5.2.1 Association Models 90
5.2.2 Key Generation 90
5.2.3 Encryption 90
5.2.4 Signed Data 90
5.2.5 Privacy Feature 91
5.3 AMP Security 91

6 Bluetooth Application Architecture 93

6.1 Bluetooth Profiles 93

6.2 Generic Access Profile 93

6.3 Profile Hierarchy 94

6.4 Generic Attribute Profile 95

6.5 GATT-based Profile Hierarchy[#E3624] 96

6.5.1 Service 96

6.5.2 Referenced Services 97

6.5.3 Characteristic 97

1 GENERAL DESCRIPTION

Bluetooth wireless technology is a short-range communications system intended to replace the cable(s) connecting portable and/or fixed electronic devices. The key features of Bluetooth wireless technology are robustness, low power consumption, and low cost. Many features of the core specification are optional, allowing product differentiation.

There are two forms of Bluetooth wireless technology systems: Basic Rate (BR) and Low Energy (LE). Both systems include device discovery, connection establishment and connection mechanisms. The Basic Rate system includes optional Enhanced Data Rate (EDR) Alternate Media Access Control (MAC) and Physical (PHY) layer extensions. The Basic Rate system offers synchronous and asynchronous connections with data rates of 721.2 kbps for Basic Rate, 2.1 Mbps for Enhanced Data Rate and high speed operation up to 24 Mbps with the 802.11 AMP. The LE system includes features designed to enable products that require lower current consumption, lower complexity and lower cost than BR/EDR. The LE system is also designed for use cases and applications with lower data rates and has lower duty cycles. Depending on the use case or application, one system including any optional parts may be more optimal than the other.

Devices implementing both systems can communicate with other devices implementing both systems as well as devices implementing either system. Some profiles and use cases will be supported by only one of the systems. Therefore, devices implementing both systems have the ability to support the most use cases.

The Bluetooth core system consists of a Host and one or more Controllers. A Host is a logical entity defined as all of the layers below the non-core profiles and above the Host Controller Interface (HCI). A Controller is a logical entity defined as all of the layers below HCI. An implementation of the Host and Controller may contain the respective parts of the HCI. Two types of Controllers are defined in this version of the Core Specification: Primary Controllers and Secondary Controllers.

An implementation of the Bluetooth Core has only one Primary Controller which may be one of the following configurations:

- BR/EDR Controller including the Radio, Baseband, Link Manager and optionally HCI.
- an LE Controller including the LE PHY, Link Layer and optionally HCI.
- a combined BR/EDR Controller portion and LE controller portion (as identified in the previous two bullets) into a single Controller. This configuration has only one Bluetooth device address shared by the combination in the combined Controller.

A Bluetooth core system may additionally have one or more Secondary Controllers described by the following configuration:

- an Alternate MAC/PHY (AMP) Controller including an 802.11 PAL (Protocol Adaptation Layer), 802.11 MAC and PHY, and optionally HCI.

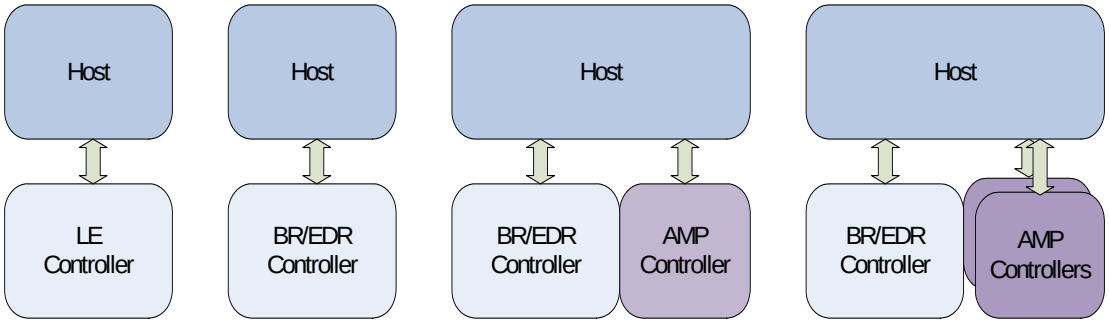


Figure 1.1: Bluetooth Host and Controller Combinations: (from left to right): LE Only Primary Controller, BR/EDR only Primary Controller, BR/EDR Primary Controller, with one AMP Secondary Controller, and BR/EDR with multiple AMP Secondary Controllers

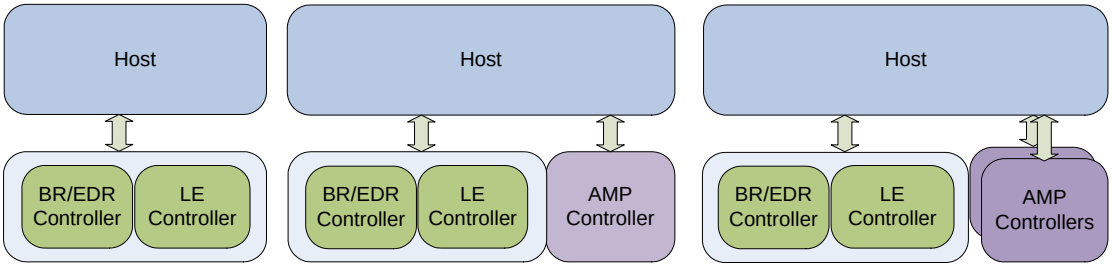


Figure 1.2: Bluetooth Host and Controller Combinations (from left to right): BR/EDR and LE Primary Controller, BR/EDR and LE Primary Controller with one AMP Secondary Controller, and BR/EDR and LE Primary Controller with multiple AMP Secondary Controllers

This volume of the specification provides an overview of the Bluetooth system architecture, communication topologies and data transport features. The text in this volume of the specification should be treated as informational and used as a background and for context-setting.

1.1 OVERVIEW OF BR/EDR OPERATION

The Basic Rate / Enhanced Data Rate (BR/EDR) radio (physical layer or PHY) operates in the unlicensed ISM band at 2.4 GHz. The system employs a frequency hop transceiver to combat interference and fading and provides many FHSS carriers. Basic Rate radio operation uses a shaped, binary frequency modulation to minimize transceiver complexity. The symbol rate is 1 Megasymbol per second (Ms/s) supporting the bit rate of 1 Megabit per second (Mb/s) or, with Enhanced Data Rate, a gross air bit rate of 2 or 3Mb/s. These modes are known as Basic Rate and Enhanced Data Rate respectively.

During typical operation a physical radio channel is shared by a group of devices that are synchronized to a common clock and frequency hopping pattern. One device provides the synchronization reference and is known as the master. All other devices synchronized to a master's clock and frequency hop-

ping pattern are known as slaves. A group of devices synchronized in this fashion form a piconet. This is the fundamental form of communication in the Bluetooth BR/EDR wireless technology.

Devices in a piconet use a specific frequency hopping pattern, which is algorithmically determined by certain fields in the Bluetooth address and clock of the master. The basic hopping pattern is a pseudo-random ordering of the 79 frequencies, separated by 1 MHz, in the ISM band. The hopping pattern can be adapted to exclude a portion of the frequencies that are used by interfering devices. The adaptive hopping technique improves Bluetooth co-existence with static (non-hopping) ISM systems when they are co-located.

The physical channel is sub-divided into time units known as slots. Data is transmitted between Bluetooth devices in packets that are positioned in these slots. When circumstances permit, a number of consecutive slots may be allocated to a single packet. Frequency hopping takes place between the transmission or reception of packets. Bluetooth technology provides the effect of full duplex transmission through the use of a Time-Division Duplex (TDD) scheme.

Above the physical channel there is a layering of links and channels and associated control protocols. The hierarchy of channels and links from the physical channel upwards is physical channel, physical link, logical transport, logical link and L2CAP channel. These are discussed in more detail in Section 3.3 on page 49 to Section 3.6 on page 72 but are introduced here to aid the understanding of the remainder of this section.

Within a physical channel, a physical link is formed between a master device and slave devices. The physical link provides bidirectional packet transport between the master and slave devices. Since a physical channel could include multiple slave devices, there are restrictions on which devices may form a physical link. There is a physical link between each slave and the master. Physical links are not formed directly between the slaves in a piconet.

The physical link is used as a transport for one or more logical links that support unicast synchronous, asynchronous and isochronous traffic, and broadcast traffic. Traffic on logical links is multiplexed onto the physical link by occupying slots assigned by a scheduling function in the resource manager.

A control protocol for the baseband and physical layers is carried over logical links in addition to user data. This is the link manager protocol (LMP). Devices that are active in a piconet have a default asynchronous connection-oriented logical transport that is used to transport the LMP protocol signaling. For historical reasons this is known as the ACL logical transport. The default ACL logical transport is the one that is created whenever a device joins a piconet. Additional logical transports may be created to transport synchronous data streams when this is required.

The Link Manager function uses LMP to control the operation of devices in the piconet and provide services to manage the lower architectural layers (radio

layer and baseband layer). The LMP protocol is only carried on the default ACL logical transport and the default broadcast logical transport.

Above the baseband layer the L2CAP layer provides a channel-based abstraction to applications and services. It carries out segmentation and reassembly of application data and multiplexing and de-multiplexing of multiple channels over a shared logical link. L2CAP has a protocol control channel that is carried over the default ACL logical transport. Application data submitted to the L2CAP protocol may be carried on any logical link that supports the L2CAP protocol.

1.2 OVERVIEW OF BLUETOOTH LOW ENERGY OPERATION

Like the BR/EDR radio, the LE radio operates in the unlicensed 2.4 GHz ISM band. The LE system employs a frequency hopping transceiver to combat interference and fading and provides many FHSS carriers. LE radio operation uses a shaped, binary frequency modulation to minimize transceiver complexity. The symbol rate is 1 Megasymbol per second (Ms/s) supporting the bit rate of 1 Megabit per second (Mb/s).

LE employs two multiple access schemes: Frequency division multiple access (FDMA) and time division multiple access (TDMA). Forty (40) physical channels, separated by 2 MHz, are used in the FDMA scheme. Three (3) are used as advertising channels and 37 are used as data channels. A TDMA based polling scheme is used in which one device transmits a packet at a predetermined time and a corresponding device responds with a packet after a predetermined interval.

The physical channel is sub-divided into time units known as events. Data is transmitted between LE devices in packets that are positioned in these events. There are two types of events: Advertising and Connection events.

Devices that transmit advertising packets on the advertising PHY channels are referred to as **advertisers**. Devices that receive advertising on the advertising channels without the intention to connect to the advertising device are referred to as **scanners**. Transmissions on the advertising PHY channels occur in advertising events. At the start of each advertising event, the advertiser sends an advertising packet corresponding to the advertising event type. Depending on the type of advertising packet, the scanner may make a request to the advertiser on the same advertising PHY channel which may be followed by a response from the advertiser on the same advertising PHY channel. The advertising PHY channel changes on the next advertising packet sent by the advertiser in the same advertising event. The advertiser may end the advertising event at any time during the event. The first advertising PHY channel is used at the start of the next advertising event.

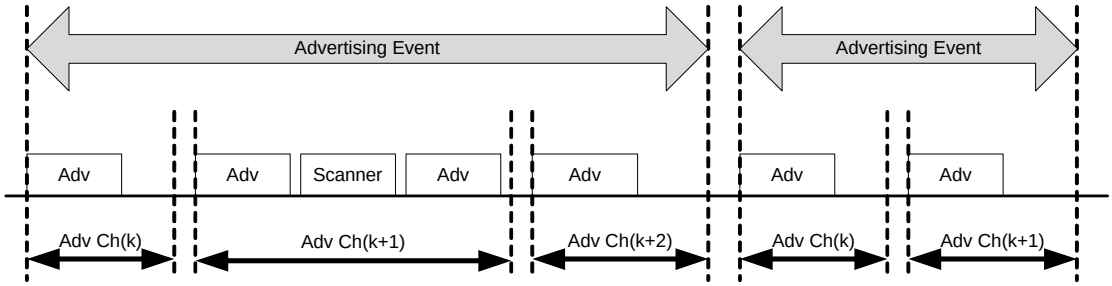


Figure 1.3: Advertising Events

LE devices may fulfill the entire communication in the case of unidirectional or broadcast communication between two or more devices using advertising events. They may also use advertising events to establish pair-wise bi-directional communication between two or more devices using data channels.

Devices that need to form a connection to another device listen for connectable advertising packets. Such devices are referred to as **initiators**. If the advertiser is using a connectable advertising event, an initiator may make a connection request using the same advertising PHY channel on which it received the connectable advertising packet. The advertising event is ended and connection events begin if the advertiser receives and accepts the request for a connection be initiated. Once a connection is established, the initiator becomes the **master** device in what is referred to as a **piconet** and the advertising device becomes the **slave** device. Connection events are used to send data packets between the master and slave devices. In connection events, channel hopping occurs at the start of each connection event. Within a connection event, the master and slave alternate sending data packets using the same data PHY channel. The master initiates the beginning of each connection event and can end each connection event at any time.

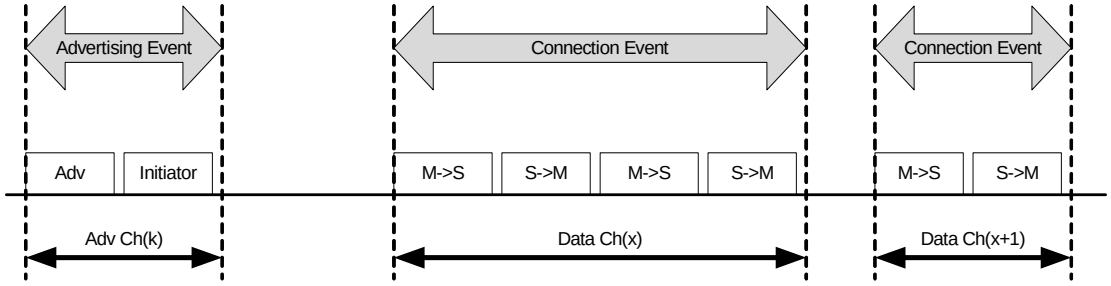


Figure 1.4: Connection Events

Devices in a piconet use a specific frequency hopping pattern, which is algorithmically determined by a field contained in the connection request sent by an initiating device. The initiating device provides the synchronization reference known as a hop interval. The hopping pattern used in LE is a pseudo-random ordering of the 37 frequencies in the ISM band. The hopping pattern can be adapted to exclude a portion of the frequencies that are used by interfering devices. The adaptive hopping technique improves Bluetooth co-existence with

static (non-hopping) ISM systems when these are co-located and have access to information about the local radio environment, or detected by other means.

Above the physical channel there are concepts of links, channels and associated control protocols. The hierarchy is physical channel, physical link, logical transport, logical link and L2CAP channel. These are discussed in more detail in Section 3.3 on page 49 to Section 3.6 on page 72 but are introduced here to aid the understanding of the remainder of this section.

Within a physical channel, a physical link is formed between a master and each slave. Physical links between slaves in a piconet are not supported. At this time, slaves are not permitted to have physical links to more than one master at a time. Role changes between a master and slave device are also not permitted at this time.

The physical link is used as a transport for one or more logical links that support asynchronous traffic. Traffic on logical links is multiplexed onto the physical link assigned by a scheduling function in the resource manager.

A control protocol for the link and physical layers is carried over logical links in addition to user data. This is the link layer protocol (LL). Devices that are active in a piconet have a default LE asynchronous connection logical transport (LE ACL) that is used to transport the LL protocol signaling. The default LE ACL is the one that is created whenever a device joins a piconet.

The Link Layer function uses the LL protocol to control the operation of devices in the piconet and provide services to manage the lower architectural layers (PHY and LL).

Just as in BR/EDR, above the link layer the L2CAP layer provides a channel-based abstraction to applications and services. It carries out fragmentation and de-fragmentation of application data and multiplexing and de-multiplexing of multiple channels over a shared logical link. L2CAP has a protocol control channel that is carried over the default logical transport.

In addition to L2CAP, LE provides two additional protocol layers that reside on top of L2CAP. The Security Manager protocol (SMP) uses a fixed L2CAP channel to implement the security functions between devices. The other is the Attribute protocol (ATT) that provides a method to communicate small amounts of data over a fixed L2CAP channel. The Attribute protocol is also used by devices to determine the services and capabilities of other devices. The Attribute protocol may also be used over BR/EDR.

1.3 OVERVIEW OF AMP OPERATION

Alternate MAC/PHYs (AMP) are secondary Controllers in the Bluetooth core system. The BR/EDR radio, the primary radio, is used to perform discovery, association, connection establishment, and connection maintenance. Once an L2CAP connection has been established between two devices over the BR/

EDR radio, the AMP Managers can discover the AMPs that are available on the other device. When an AMP is common between the two devices, the Core system provides mechanisms for moving data traffic from BR/EDR Controller to an AMP Controller.

Each AMP consists of a Protocol Adaptation Layer (PAL) on top of a MAC and PHY. The PAL is responsible for mapping the Bluetooth protocols and behavior (as specified by HCI) to the specific protocols of the underlying MAC and PHY.

L2CAP channels may be created on, or moved to, an AMP. L2CAP channels may also be moved back to the BR/EDR radio when those capabilities are not necessary or when the AMP physical link has a link supervision timeout. A link supervision timeout on the BR/EDR radio forces a disconnection of all AMP physical links between those devices.

AMPs may be enabled or disabled as needed in order to minimize power consumption in the system.

1.4 NOMENCLATURE

Where the following terms appear in the specification they have the meaning given in Table 1.1 on page 24.

Ad Hoc Network	A network typically created in a spontaneous manner. An ad hoc network requires no formal infrastructure and is limited in temporal and spatial extent.
Advertiser	A Bluetooth low energy device that broadcasts advertising packets during advertising events on advertising channels
Advertising Event	A series of between one and three advertising packets on different advertising channels sent by an advertiser.
Active Slave Broadcast (ASB)	The Active Slave Broadcast logical transport that is used to transport L2CAP user traffic to all active devices in the piconet over the BR/EDR Controller. See Section 3.5.4.4 on page 66
AMP	Alternate Media Access Controller (MAC) and Physical Layer (PHY) or Alternate MAC/PHY.
AMP Controller	A term referring to AMP PHY, AMP MAC, Protocol Adaptation Layer, and HCI layers.
Beacon Train	A pattern of reserved slots within a basic or adapted piconet physical channel. Transmissions starting in these slots are used to resynchronize parked devices.
Bluetooth	Bluetooth is a wireless communication link, operating in the unlicensed ISM band at 2.4 GHz using a frequency hopping transceiver. It allows real-time AV and data communications between Bluetooth Hosts. The link protocol is based on time slots.
Bluetooth Baseband	The part of the Bluetooth system that specifies or implements the medium access and physical layer procedures to support the exchange of real-time voice, data information streams, and ad hoc networking between Bluetooth Devices.
Bluetooth Clock	A 28 bit clock internal to a BR/EDR Controller sub-system that ticks every 312.5µs. The value of this clock defines the slot numbering and timing in the various physical channels.
Bluetooth Controller	A generic term referring to a Primary Controller with or without a Secondary Controller.
Bluetooth Device	A device that is capable of short-range wireless communications using the Bluetooth system.
Bluetooth Device Address	A 48 bit address used to identify each Bluetooth device.

Table 1.1: Nomenclature.

BD_ADDR	The Bluetooth device Address, BD_ADDR, is used to identify a Bluetooth device.
BR/EDR	Bluetooth basic rate (BR) and enhanced data rate (EDR).
BR/EDR Controller	A term referring to the Bluetooth Radio, Baseband, Link Manager, and HCI layers.
BR/EDR/LE	Bluetooth basic rate (BR), enhanced data rate (EDR) and low energy (LE).
BR/EDR Piconet Physical Channel	A Channel that is divided into time slots in which each slot is related to an RF hop frequency. Consecutive hops normally correspond to different RF hop frequencies and occur at a standard hop rate of 1600 hops/s. These consecutive hops follow a pseudo-random hopping sequence, hopping through a 79 RF channel set, or optionally fewer channels when Adaptive Frequency Hopping (AFH) is in use.
C-plane	Control plane
Channel	Either a physical channel or an L2CAP channel, depending on the context.
Connect (to service)	The establishment of a connection to a service. If not already done, this also includes establishment of a physical link, logical transport, logical link and L2CAP channel.
Connectable device	A BR/EDR device in range that periodically listens on its page scan physical channel and will respond to a page on that channel. An LE device that is advertising using a connectable advertising event.
Connected devices	Two BR/EDR devices and with a physical link between them.
Connecting	A phase in the communication between devices when a connection between the devices is being established. (Connecting phase follows after the link establishment phase is completed.)
Connection	A connection between two peer applications or higher layer protocols mapped onto an L2CAP channel.
Connection establishment	A procedure for creating a connection mapped onto a channel.
Connection Event	A series of one or more pairs of interleaving data packets sent between a master and a slave on the same physical channel.
Controller	A collective term referring to all of the layers below HCI.
Coverage area	The area where two Bluetooth devices can exchange messages with acceptable quality and performance.

Table 1.1: Nomenclature.

Creation of a secure connection	A procedure of establishing a connection, including authentication and encryption.
Creation of a trusted relationship	A procedure where the remote device is marked as a trusted device. This includes storing a common link key for future authentication, or pairing, when a link key is not available.
Device discovery	A procedure for retrieving the Bluetooth device address, clock, class-of-device field and used page scan mode from discoverable devices.
Discoverable device	A BR/EDR device in range that periodically listens on an inquiry scan physical channel and will respond to an inquiry on that channel. An LE device in range that is advertising with a connectable or scannable advertising event with a discoverable flag set in the advertising data. This device is in the discoverable mode.
Discoverable Mode	A Bluetooth device that is performing inquiry scans in BR/EDR or advertising with a discoverable or connectable advertising event with a discoverable flag set in LE.
Discovery procedure	A Bluetooth device that is carrying out the inquiry procedure in BR/EDR or scanning for advertisers using a discoverable or connectable advertising event with a discoverable flag set in LE.
HCI	The Host Controller Interface (HCI) provides a command interface to the baseband controller and link manager and access to hardware status and control registers. This interface provides a uniform method of accessing the Bluetooth baseband capabilities.
Host	A logical entity defined as all of the layers below the non-core profiles (e.g.Volume 3) and above the Host Controller Interface (HCI). A Bluetooth Host attached to a Bluetooth Controller may communicate with other Bluetooth Hosts attached to their Controllers as well.
Initiator	A Bluetooth low energy device that listens on advertising channels for connectable advertising events to form connections.
Inquiring device	A BR/EDR device that is carrying out the inquiry procedure. This device is performing the discovery procedure.
Inquiry	A procedure where a Bluetooth device transmits inquiry messages and listens for responses in order to discover the other Bluetooth devices that are within the coverage area.
Inquiry scan	A procedure where a Bluetooth device listens for inquiry messages received on its inquiry scan physical channel.

Table 1.1: Nomenclature.

Interoperability	The ability of two or more systems or components to exchange information and to use the information that has been exchanged.
Isochronous data	Information in a stream where each information entity in the stream is bound by a time relationship to previous and successive entities.
Known device	A Bluetooth device for which at least the BD_ADDR is stored.
L2CAP	Logical Link Control and Adaptation Protocol
L2CAP Channel	A logical connection on L2CAP level between two devices serving a single application or higher layer protocol.
L2CAP Channel establishment	A procedure for establishing a logical connection on L2CAP level.
LE	Bluetooth low energy (LE)
Link establishment	A procedure for establishing the default ACL link and hierarchy of links and channels between devices.
Link	Shorthand for a logical link.
Link key	A secret key that is known by two devices and is used to authenticate the link.
LMP authentication	An LMP level procedure for verifying the identity of a remote device.
LMP pairing	A procedure that authenticates two devices and creates a common link key that can be used as a basis for a trusted relationship or a (single) secure connection.
Logical link	The lowest architectural level used to offer independent data transport services to clients of the Bluetooth system.
Logical transport	Shared acknowledgement protocol and link identifiers between different logical links.
Name discovery	A procedure for retrieving the user-friendly name (the Bluetooth device name) of a connectable device.
Packet	Format of aggregated bits that are transmitted on a physical channel.
Page	The initial phase of the connection procedure where a device transmits a train of page messages until a response is received from the target device or a time-out occurs.
Page scan	A procedure where a device listens for page messages received on its page scan physical channel.
Paging device	A Bluetooth device that is carrying out the page procedure.

Table 1.1: Nomenclature.

Paired device	A Bluetooth device for which a link key has been created (either before connection establishment was requested or during connecting phase).
Parked device	A device operating in a basic mode piconet that is synchronized to the master but has given up its default ACL logical transport.
Passkey	A 6-digit number used to authenticate connections when Secure Simple Pairing is used.
Physical Channel	Characterized by synchronized occupancy of a sequence of RF carriers by one or more devices. A number of physical channel types exist with characteristics defined for their different purposes.
Physical Link	A Baseband or Link Layer level connection between two devices.
Piconet	A collection of devices occupying a shared physical channel where one of the devices is the Piconet Master and the remaining devices are connected to it.
Piconet Master	The BR/EDR device in a piconet whose Bluetooth Clock and Bluetooth Device Address are used to define the piconet physical channel characteristics.
Piconet Slave	Any BR/EDR device in a piconet that is not the Piconet Master, but is connected to the Piconet Master.
PIN	A user-friendly number that can be used to authenticate connections to a device before pairing has taken place.
PMP	A Participant in Multiple Piconets. A device that is concurrently a member of more than one piconet, which it achieves using time division multiplexing (TDM) to interleave its activity on each piconet physical channel.
The Parked Slave Broadcast (PSB)	The Parked Slave Broadcast logical transport that is used for communications between the master and parked devices. These communications may also be received by active devices. See Section 3.5.4.5 on page 67.
Scanner	A Bluetooth low energy device that listens for advertising events on the advertising channels.
Scatternet	Two or more piconets that include one or more devices acting as PMPs.
Service Layer Protocol	A protocol that uses an L2CAP channel for transporting PDUs.
Service discovery	Procedures for querying and browsing for services offered by or through another Bluetooth device.
Silent device	A Bluetooth enabled device appears as silent to a remote device if it does not respond to inquiries made by the remote device.

Table 1.1: Nomenclature.

U-plane	User plane
Unknown device	A Bluetooth device for which no information (Bluetooth Device Address, link key or other) is stored.

Table 1.1: Nomenclature.

2 CORE SYSTEM ARCHITECTURE

The Bluetooth Core system consists of a Host, a Primary Controller and zero or more Secondary Controllers. A minimal implementation of the Bluetooth BR/EDR core system covers the four lowest layers and associated protocols defined by the Bluetooth specification as well as one common service layer protocol; the Service Discovery Protocol (SDP) and the overall profile requirements are specified in the Generic Access Profile (GAP). The BR/EDR Core system includes support of Alternate MAC/PHYs (AMPs) including an AMP Manager Protocol and Protocol Adaptation Layers (PALs) supporting externally referenced MAC/PHYs. A minimal implementation of a Bluetooth LE only core system covers the four lowest layers and associated protocols defined by the Bluetooth specification as well as two common service layer protocols; the Security Manager (SM) and Attribute Protocol (ATT) and the overall profile requirements are specified in the Generic Attribute Profile (GATT) and Generic Access Profile (GAP). Implementations combining Bluetooth BR/EDR and LE include both of the minimal implementations described above.

A complete Bluetooth application requires a number of additional service and higher layer protocols that are defined in the Bluetooth specification, but are not described here. The core system architecture is shown in Figure 2.1 on page 31.

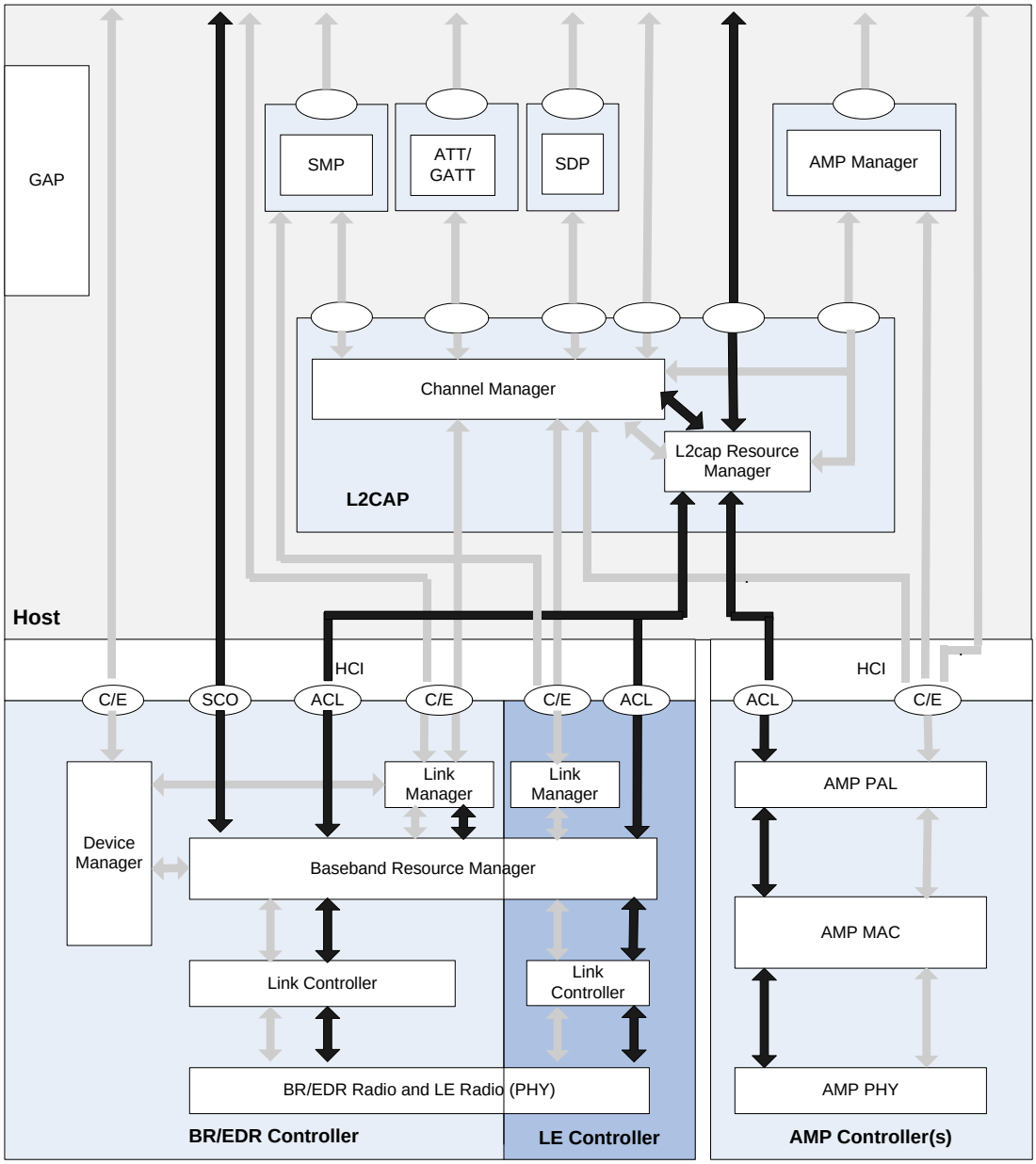


Figure 2.1: Bluetooth core system architecture

Figure 2.1 on page 31 shows the Core blocks, each with its associated communication protocol. Link Manager, Link Controller and BR/EDR Radio blocks comprise a BR/EDR Controller. An AMP PAL, AMP MAC, and AMP PHY comprise an AMP Controller. Link Manager, Link Controller and LE Radio blocks comprise an LE Controller. L2CAP, SDP and GAP blocks comprise a BR/EDR Host. L2CAP, SMP, Attribute protocol, GAP and Generic Attribute Profile (GATT) blocks comprise an LE Host. A BR/EDR/LE Host combines the set of blocks from each respective Host. This is a common implementation involving a standard physical communication interface between the Controller and the Host. Although this interface is optional the architecture is designed to allow for its existence and characteristics. The Bluetooth specification enables interoperability between independent Bluetooth systems by defining the protocol mes-

sages exchanged between equivalent layers, and also interoperability between independent Bluetooth subsystems by defining a common interface between Bluetooth controllers and Bluetooth hosts.

A number of functional blocks and the path of services and data between them are shown. The functional blocks shown in the diagram are informative; in general the Bluetooth specification does not define the details of implementations except where this is required for interoperability. Thus the functional blocks in Figure 2.1 on page 31 are shown in order to aid description of the system behavior. An implementation may be different from the system shown in Figure 2.1 on page 31.

Standard interactions are defined for all inter-device operation, where Bluetooth devices exchange protocol signaling according to the Bluetooth specification. The Bluetooth core system protocols are the Radio (PHY) protocol, Link Control (LC) and Link Manager (LM) protocol or Link Layer (LL) protocol, AMP PAL, Logical Link Control and Adaptation protocol (L2CAP), and AMP Manager Protocol, all of which are fully defined in subsequent parts of the Bluetooth specification. In addition, the Service Discovery Protocol (SDP) and the Attribute Protocol (ATT) are service layer protocols that may be required by some Bluetooth applications.

The Bluetooth core system offers services through a number of service access points that are shown in the diagram as ellipses. These services consist of the basic primitives that control the Bluetooth core system. The services can be split into three types. There are device control services that modify the behavior and modes of a Bluetooth device, transport control services that create, modify and release traffic bearers (channels and links), and data services that are used to submit data for transmission over traffic bearers. It is common to consider the first two as belonging to the C-plane and the last as belonging to the U-plane.

A service interface to the Bluetooth Controller subsystem is defined such that the Primary Controller may be considered a standard part. In this configuration the Bluetooth Controller operates the lowest four layers. The Bluetooth Host operates the L2CAP layer and other higher layers. The standard interface is called the Host Controller Interface (HCI) and its service access points are represented by the ellipses on the upper edge of the Bluetooth controller subsystem in Figure 2.1 on page 31. Implementation of this standard service interface is optional.

As the Bluetooth architecture is defined with the possibility of separate host and controller(s) communicating through one or more HCI transports, a number of general assumptions are made. Bluetooth controllers are assumed to have limited data buffering capabilities in comparison with the host. Therefore the L2CAP layer is expected to carry out some simple resource management when submitting L2CAP PDUs to the controller for transport to a peer device. This includes segmentation of L2CAP SDUs into more manageable PDUs and then the fragmentation of PDUs into start and continuation packets of a size

suitable for the controller buffers, and management of the use of controller buffers to ensure availability for channels with Quality of Service (QoS) commitments.

The BR/EDR Baseband, LE Link Layer, and AMP MAC layers provides the basic acknowledgement/repeat request (ARQ) protocol in Bluetooth. The L2CAP layer can optionally provide a further error detection and retransmission to the L2CAP PDUs. This feature is recommended for applications with requirements for a low probability of undetected errors in the user data. A further optional feature of L2CAP is a window-based flow control that can be used to manage buffer allocation in the receiving device. Both of these optional features augment the QoS performance in certain scenarios. Not all of the L2CAP capabilities are available when using the LE system.

Although these assumptions may not be required for embedded Bluetooth implementations that combine all layers in a single system, the general architectural and QoS models are defined with these assumptions in mind, in effect a lowest common denominator.

Automated conformance testing of implementations of the Bluetooth core system is required. This is achieved by allowing the tester to control the implementation through the PHY interface, and test interfaces such as Direct Test Mode (DTM), Generic Test Methodology (GTM), Test Control Interface (TCI), and test commands and events over HCI which are only required for conformance testing.

The tester exchanges with the Implementation Under Test (IUT) through the PHY interface to ensure the correct responses to requests from remote devices. The tester controls the IUT through HCI, DTM, GTM, or TCI to cause the IUT to originate exchanges through the PHY interface so that these can also be verified as conformant.

The TCI uses a different command-set (service interface) for the testing of each architectural layer and protocol. A subset of the HCI command-set is used as the TCI service interface for each of the layers and protocols within the BR/EDR Controller subsystem. A separate service interface is used for testing the L2CAP layer and protocol. As an L2CAP service interface is not defined in the Bluetooth core specification it is defined separately in the Test Control Interface specification. Implementation of the L2CAP service interface is only required for conformance testing.

2.1 CORE ARCHITECTURAL BLOCKS

This section describes the function and responsibility of each of the blocks shown in Figure 2.1 on page 31. An implementation is not required to follow the architecture described above, though every implementation shall conform to the protocol specifications described in subsequent parts of the Bluetooth specification, and shall implement the behavioral aspects of the system outlined below and specified in subsequent parts of the Bluetooth specification.

2.1.1 Host Architectural Blocks

2.1.1.1 Channel Manager

The channel manager is responsible for creating, managing and closing L2CAP channels for the transport of service protocols and application data streams. The channel manager uses the L2CAP protocol to interact with a channel manager on a remote (peer) device to create these L2CAP channels and connect their endpoints to the appropriate entities. The channel manager interacts with its local link manager or an AMP PAL to create new logical links (if necessary) and to configure these links to provide the required quality of service for the type of data being transported.

2.1.1.2 L2CAP Resource Manager

The L2CAP resource manager block is responsible for managing the ordering of submission of PDU fragments to the baseband and some relative scheduling between channels to ensure that L2CAP channels with QoS commitments are not denied access to the physical channel due to Controller resource exhaustion. This is required because the architectural model does not assume that a Controller has limitless buffering, or that the HCI is a pipe of infinite bandwidth.

L2CAP Resource Managers may also carry out traffic conformance policing to ensure that applications are submitting L2CAP SDUs within the bounds of their negotiated QoS settings. The general Bluetooth data transport model assumes well-behaved applications, and does not define how an implementation is expected to deal with this problem.

2.1.1.3 Security Manager Protocol

The Security Manager Protocol (SMP) is the peer-to-peer protocol used to generate encryption keys and identity keys. The protocol operates over a dedicated fixed L2CAP channel. The SMP block also manages storage of the encryption keys and identity keys and is responsible for generating random addresses and resolving random addresses to known device identities. The SMP block interfaces directly with the controller to provide stored keys used for encryption and authentication during the encryption or pairing procedures.

This block is only used in LE systems. Similar functionality in the BR/EDR system is contained in the Link Manager block in the Controller. SMP functionality

is in the Host on LE systems to reduce the implementation cost of the LE only controllers.

2.1.1.4 Attribute Protocol

The Attribute Protocol (ATT) block implements the peer-to-peer protocol between an attribute server and an attribute client. The ATT client communicates with an ATT server on a remote device over a dedicated fixed L2CAP channel. The ATT client sends commands, requests, and confirmations to the ATT server. The ATT server sends responses, notifications and indications to the client. These ATT client commands and requests provide a means to read and write values of attributes on a peer device with an ATT server.

2.1.1.5 AMP Manager Protocol

The AMP manager is a layer that uses L2CAP to communicate with a peer AMP Manager on a remote device. It also interfaces directly with the AMP PAL for AMP control purposes. The AMP manager is responsible for discovering remote AMP(s) and determining their availability. It also collects information about remote AMP(s). This information is used to set up and manage AMP physical links. The AMP manager uses a dedicated L2CAP signaling channel to communicate with remote AMP manager(s).

2.1.1.6 Generic Attribute Profile

The Generic Attribute Profile (GATT) block represents the functionality of the attribute server and, optionally, the attribute client. The profile describes the hierarchy of services, characteristics and attributes used in the attribute server. The block provides interfaces for discovering, reading, writing and indicating of service characteristics and attributes. GATT is used on LE devices for LE profile service discovery.

2.1.1.7 Generic Access Profile

The Generic Access Profile (GAP) block represents the base functionality common to all Bluetooth devices such as modes and access procedures used by the transports, protocols and application profiles. GAP services include device discovery, connection modes, security, authentication, association models and service discovery.

2.1.2 BR/EDR/LE Controller Architectural Blocks

In implementations where the BR/EDR and LE systems are combined, the architectural blocks may be shared between systems or each system may have their own instantiation of the block.

2.1.2.1 Device Manager

The device manager is the functional block in the baseband that controls the general behavior of the Bluetooth device. It is responsible for all operations of the Bluetooth system that are not directly related to data transport, such as inquiring for the presence of nearby Bluetooth devices, connecting to Bluetooth devices, or making the local Bluetooth device discoverable or connectable by other devices.

The device manager requests access to the transport medium from the baseband resource controller in order to carry out its functions.

The device manager also controls local device behavior implied by a number of the HCI commands, such as managing the device local name, any stored link keys, and other functionality.

2.1.2.2 Link Manager

The link manager is responsible for the creation, modification and release of logical links (and, if required, their associated logical transports), as well as the update of parameters related to physical links between devices. The link manager achieves this by communicating with the link manager in remote Bluetooth devices using the Link Management Protocol (LMP) in BR/EDR and the Link Layer Protocol (LL) in LE.

The LM or LL protocol allows the creation of new logical links and logical transports between devices when required, as well as the general control of link and transport attributes such as the enabling of encryption on the logical transport, the adapting of transmit power in BR/EDR on the physical link, or the adjustment of QoS settings in BR/EDR for a logical link.

2.1.2.3 Baseband Resource Manager

The baseband resource manager is responsible for all access to the radio medium. It has two main functions. At its heart is a scheduler that grants time on the physical channels to all of the entities that have negotiated an access contract. The other main function is to negotiate access contracts with these entities. An access contract is effectively a commitment to deliver a certain QoS that is required in order to provide a user application with an expected performance.

The access contract and scheduling function must take account of any behavior that requires use of the Primary Controller. This includes (for example) the normal exchange of data between connected devices over logical links, and logical transports, as well as the use of the radio medium to carry out inquiries, make connections, be discoverable or connectable, or to take readings from unused carriers during the use of adaptive frequency hopping mode.

In some cases in BR/EDR systems the scheduling of a logical link results in changing a logical link to a different physical channel from the one that was

previously used. This may be (for example) due to involvement in scatternet, a periodic inquiry function, or page scanning. When the physical channels are not time slot aligned, then the resource manager also accounts for the realignment time between slots on the original physical channel and slots on the new physical channel. In some cases the slots will be naturally aligned due to the same device clock being used as a reference for both physical channels.

2.1.2.4 Link Controller

The link controller is responsible for the encoding and decoding of Bluetooth packets from the data payload and parameters related to the physical channel, logical transport and logical link.

The link controller carries out the link control protocol signaling in BR/EDR and link layer protocol in LE (in close conjunction with the scheduling function of the resource manager), which is used to communicate flow control and acknowledgement and retransmission request signals. The interpretation of these signals is a characteristic of the logical transport associated with the baseband packet. Interpretation and control of the link control signaling is normally associated with the resource manager's scheduler.

2.1.2.5 PHY

The PHY block is responsible for transmitting and receiving packets of information on the physical channel. A control path between the baseband and the PHY block allows the baseband block to control the timing and frequency carrier of the PHY block. The PHY block transforms a stream of data to and from the physical channel and the baseband into required formats.

2.1.3 AMP Controller architectural blocks

2.1.3.1 AMP HCI

The AMP HCI is the logical interface between an AMP Controller and Host (L2CAP and AMP manager). HCI is an optional layer used when the Host and AMP Controller(s) are physically separated. Support for AMPs requires additional HCI commands and events. These new commands and events are related to AMP physical and logical link management, QoS, as well as flow control.

There is one logical instantiation of an HCI per AMP Controller, and another logical instantiation for the BR/EDR Controller. The physical HCI transport layer manages the multiplexing of several Controllers over the same physical transport bus in cases where multiple Controllers exist in a single physical unit.

2.1.3.2 AMP PAL

The AMP PAL is the AMP layer interfacing the AMP MAC with the Host (L2CAP and AMP Manager). It translates commands from the Host into the specific MAC service primitives and primitives into commands, and it translates primitives from the AMP MAC into understandable event(s) for the Host. The AMP PAL provides support for AMP channel management, data traffic according to specified flow specifications, and power efficiency.

There is not a requirement that the AMP PAL has exclusive access to the AMP itself. Implementers may choose to allow other protocols to be concurrent clients of the AMP. Specific mechanisms for allowing other protocols to share access to the AMP are outside the scope of this specification.

A generic PAL architecture is shown in Figure 2.2.

.

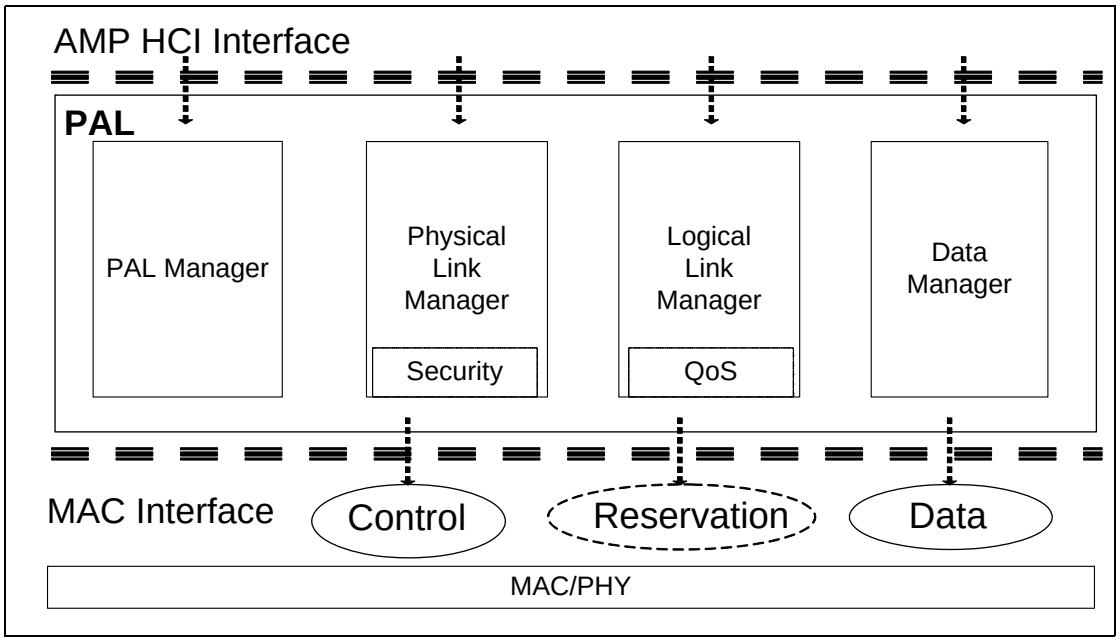


Figure 2.2: Generic PAL architecture

Capabilities of the PAL and the underlying MAC/PHY are exchanged via the AMP_Info and AMP_Assoc structures by the AMP Manager Protocol. The AMP_Info contains generic capabilities of the PAL such as bandwidth availability, maximum PDU size, PAL capabilities, length of the AMP_Assoc and flush timeout information. The contents of the AMP_Assoc are dependent on the PAL and are related to the underlying MAC/PHY

2.1.3.3 AMP MAC

The AMP MAC is the MAC layer as defined in the IEEE 802 reference layer model. It provides services such as addressing and mechanisms to control and access channels. The AMP MAC is in between AMP PHY and AMP PAL layers.

2.1.3.4 AMP PHY

The AMP PHY is the AMP physical layer.

3 DATA TRANSPORT ARCHITECTURE

The Bluetooth data transport system follows a layered architecture. This description of the Bluetooth system describes the Bluetooth core transport layers up to and including L2CAP channels. All Bluetooth operational modes follow the same generic transport architecture, which is shown in Figure 3.1 on page 39.

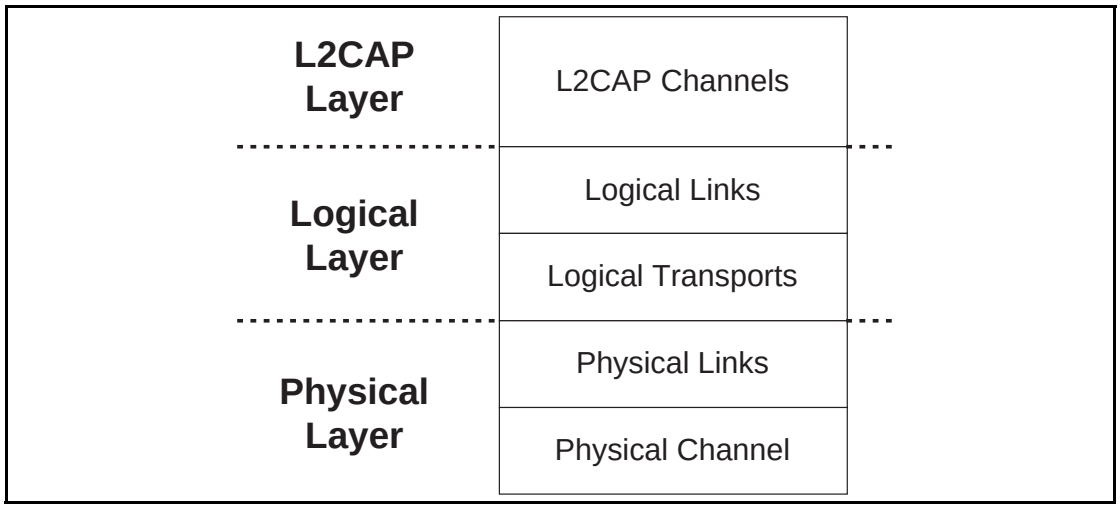


Figure 3.1: Bluetooth generic data transport architecture

For efficiency and legacy reasons, the Bluetooth transport architecture includes a sub-division of the logical layer, distinguishing between logical links and logical transports. This sub-division provides a general (and commonly understood) concept of a logical link that provides an independent transport between two or more devices. The logical transport sub-layer is required to describe the inter-dependence between some of the logical link types (mainly for reasons of legacy behavior.)

Prior to Version 1.2, the Bluetooth Core specification described the ACL and SCO links as physical links. With the addition of Extended SCO (eSCO) and for future expansion it is better to consider these as logical transport types, which more accurately encapsulates their purpose. However, they are not as independent as might be desired, due to their shared use of resources such as the LT_ADDR and acknowledgement/repeat request (ARQ) scheme. Hence the architecture is incapable of representing these logical transports with a single transport layer. The additional logical transport layer goes some way towards describing this behavior.

3.1 CORE TRAFFIC BEARERS

The Bluetooth core system provides a number of standard traffic bearers for the transport of service protocol and application data. These are shown in Figure 3.2 on page 40 below (for ease of representation this is shown with higher layers to the left and lower layers to the right).

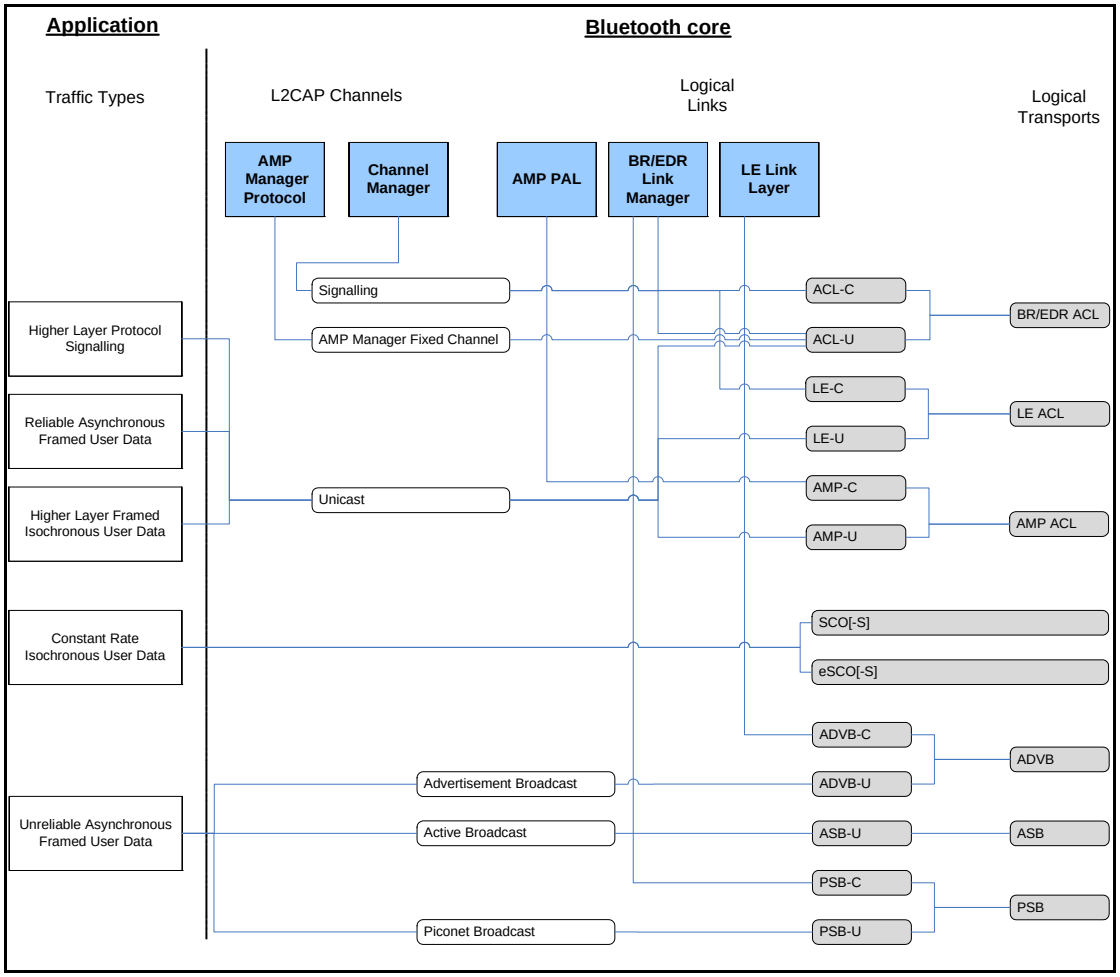


Figure 3.2: Bluetooth traffic bearers

The core traffic bearers that are available to applications are shown in Figure 3.2 on page 40 as the shaded rounded rectangles. The architectural layers that are defined to provide these services are described in Section 2 on page 30. A number of data traffic types are shown on the left of the diagram linked to the traffic bearers that are typically suitable for transporting that type of data traffic.

The logical links are named using the names of the associated logical transport and a suffix that indicates the type of data that is transported. (C for control links carrying LMP or LL messages, U for L2CAP links carrying user data (L2CAP PDUs) and S for stream links carrying unformatted synchronous or isochronous data.) It is common for the suffix to be removed from the logical link without introducing ambiguity, thus a reference to the default ACL logical transport can be resolved to mean the ACL-C logical link in cases where the

LMP protocol is being discussed, the LE-C logical link in cases where LL protocol is being discussed, the AMP-C logical link in cases where a PAL protocol is being discussed, or the ACL-U,LE-U, or AMP-U logical links when the L2CAP layer is being discussed.

The mapping of application traffic types to Bluetooth core traffic bearers in Figure 3.2 on page 40 is based on matching the traffic characteristics with the bearer characteristics. It is recommended to use these mappings as they provide the most natural and efficient method of transporting the data with its given characteristics.

However, an application (or an implementation of the Bluetooth core system) may choose to use a different traffic bearer, or a different mapping to achieve a similar result. For example, in a BR/EDR piconet with only one slave, the master may choose to transport L2CAP broadcasts over the ACL-U logical link rather than over the ASB-U or PSB-U logical links. This will probably be more efficient in terms of bandwidth (if the physical channel quality is not too degraded.) Use of alternative transport paths to those in Figure 3.2 on page 40 is only acceptable if the characteristics of the application traffic type are preserved.

Figure 3.2 on page 40 shows a number of application traffic types. These are used to classify the types of data that may be submitted to the Bluetooth core system. The original data traffic type may not be the same as the type that is submitted to the Bluetooth core system if an intervening process modifies it. For example, video data is generated at a constant rate but an intermediate coding process may alter this to variable rate, e.g. by MPEG4 encoding. For the purposes of the Bluetooth core system, only the characteristic of the submitted data is of interest.

3.1.1 Framed Data Traffic

The L2CAP layer services provide a frame-oriented transport for asynchronous and isochronous user data. The application submits data to this service in variable-sized frames (up to a negotiated maximum for the channel) and these frames are delivered in the same form to the corresponding application on the remote device. There is no requirement for the application to insert additional framing information into the data, although it may do so if this is required (such framing is invisible to the Bluetooth core system.)

Connection-oriented BR/EDR L2CAP channels may be created for transport of unicast (point-to-point) data between two Bluetooth devices. Connection-oriented channels provide a context within which specific properties may be applied to data transported on the channel. For example, quality of service parameters or flow and error control modes may be applied. Connection-oriented L2CAP channels are created using the L2CAP connection procedure.

A connectionless BR/EDR L2CAP channel exists for broadcasting data or for transport of unicast data. In the case of piconet topologies the master device is always the source of broadcast data and the slave device(s) are the recipients.

Broadcast traffic on the connectionless L2CAP channel is uni-directional. Unicast data sent on the connectionless L2CAP channels may be uni-directional or bi-directional. Unicast data sent on the L2CAP connectionless channel provides an alternate mechanism to send data with the same level of reliability as an L2CAP connection-oriented channel operating in Basic mode but without the additional latency incurred by opening an L2CAP connection-oriented channel.

BR/EDR L2CAP channels have an associated QoS setting that defines constraints on the delivery of the frames of data. These QoS settings may be used to indicate (for example) that the data is isochronous, and therefore has a limited lifetime after which it becomes invalid, or that the data should be delivered within a given time period, or that the data is reliable and should be delivered without error, however long this takes.

Only fixed channels are supported in the LE system; connection-oriented and connectionless L2CAP channels are not supported.

Some L2CAP channels are fixed channels created when the ACL-U and/or LE-U logical links are established. These fixed channels have fixed channel identifiers and fixed configurations and do not permit negotiation of the configuration after they are created. These fixed channels are used for BR/EDR and LE L2CAP signaling (ACL-U or LE-U), connectionless channel (ACL-U), AMP manager protocol (ACL-U), Security Manager Protocol (LE-U), Attribute Protocol (ACL-U or LE-U) and AMP Test Manager (ACL-U).

The L2CAP channel manager is responsible for arranging to transport the L2CAP channel data frames on an appropriate baseband logical link, possibly multiplexing this onto the baseband logical link with other L2CAP channels with similar characteristics.

3.1.2 Unframed Data Traffic

If the application does not require delivery of data in frames, possibly because it includes in-stream framing, or because the data is a pure stream, then it may avoid the use of L2CAP channels and make direct use of a baseband logical link.

The Bluetooth core system supports the direct transport of application data that is isochronous and of a constant rate (either bit-rate, or frame-rate for pre-framed data), using a SCO-S or eSCO-S logical link. These logical links reserve physical channel bandwidth and provide a constant rate transport locked to the piconet clock. Data is transported in fixed size packets at fixed intervals with both of these parameters negotiated during channel establishment. eSCO links provide a greater choice of bit-rates and also provide greater reliability by using limited retransmission in case of error. Enhanced Data Rate operation is supported for eSCO, but not for SCO logical transports. SCO and eSCO logical transports do not support multiplexed logical links or any further layering within the Bluetooth core. An application may choose to layer a num-

ber of streams within the submitted SCO/eSCO stream, provided that the submitted stream is, or has the appearance of being, a constant rate stream.

The application chooses the most appropriate type of logical link from those available at the baseband, and creates and configures it to transport the data stream, and releases it when completed. (The application will normally also use a framed L2CAP unicast channel to transport its C-plane information to the peer application on the remote device.)

If the application data is isochronous and of a variable rate, then this may only be carried by the L2CAP unicast channel, and hence will be treated as framed data.

Unframed data traffic is not supported in the LE system.

3.1.3 Reliability of traffic bearers

3.1.3.1 BR/EDR Reliability

Bluetooth is a wireless communications system. In poor RF environments, this system should be considered inherently unreliable. To counteract this the system provides levels of protection at each layer. The baseband packet header uses forward error correcting (FEC) coding to allow error correction by the receiver and a header error check (HEC) to detect errors remaining after correction. Certain Baseband packet types include FEC for the payload. Furthermore, some Baseband packet types include a cyclic redundancy error check (CRC).

On ACL logical transports the results of the error detection algorithm are used to drive a simple ARQ protocol. This provides an enhanced reliability by retransmitting packets that do not pass the receiver's error checking algorithm. It is possible to modify this scheme to support latency-sensitive packets by discarding an unsuccessfully transmitted packet at the transmitter if the packet's useful life has expired. eSCO links use a modified version of this scheme to improve reliability by allowing a limited number of retransmissions.

The resulting reliability gained by this ARQ scheme is only as dependable as the ability of the HEC and CRC codes to detect errors. In most cases this is sufficient, however it has been shown that for the longer packet types the probability of an undetected error is too high to support typical applications, especially those with a large amount of data being transferred.

The L2CAP layer provides an additional level of error control that is designed to detect the occasional undetected errors in the baseband layer and request retransmission of the affected data. This provides the level of reliability required by typical Bluetooth applications.

Broadcast links have no feedback route, and are unable to use the ARQ scheme (although the receiver is still able to detect errors in received packets.)

Instead each packet is transmitted several times in the hope that the receiver is able to receive at least one of the copies successfully. Despite this approach there are still no guarantees of successful receipt, and so these links are considered unreliable.

In summary, if a link or channel is characterized as reliable this means that the receiver is capable of detecting errors in received packets and requesting retransmission until the errors are removed. Due to the error detection system used some residual (undetected) errors may still remain in the received data. For L2CAP channels the level of these is comparable to other communication systems, although for logical links the residual error level is somewhat higher.

The transmitter may remove packets from the transmit queue such that the receiver does not receive all the packets in the sequence. If this happens detection of the missing packets is delegated to the L2CAP layer.

On an unreliable link the receiver is capable of detecting errors in received packets but cannot request retransmission. The packets passed on by the receiver may be without error, but there is no guarantee that all packets in the sequence are received. Hence the link is considered fundamentally unreliable. There are limited uses for such links, and these uses are normally dependent on the continuous repetition of data from the higher layers while it is valid.

Stream links have a reliability characteristic somewhere between a reliable and an unreliable link, depending on the current operating conditions.

3.1.3.2 LE reliability

Like BR/EDR, in poor RF environments, the LE system should be considered inherently unreliable. To counteract this, the system provides levels of protection at each layer. The LL packet uses a 24-bit cyclic redundancy error check (CRC) to cover the contents of the packet payload. If the CRC verification fails on the packet payload, the packet is not acknowledged by the receiver and the packet gets retransmitted by the sender.

Broadcast links have no feedback route, and are unable to use the acknowledgement scheme (although the receiver is still able to detect errors in received packets.) Instead, each packet is transmitted several times to increase the probability that the receiver is able to receive at least one of the copies successfully. Despite this approach there are still no guarantees of successful receipt, and so these links are considered unreliable.

In summary, if a link or channel is characterized as reliable this means that the receiver is capable of detecting errors in received packets and requesting retransmission until the errors are removed.

The transmitter may remove packets from the transmit queue such that the receiver does not receive all the packets in the sequence. If this happens detection of the missing packets is delegated to the L2CAP layer.

On an unreliable link the receiver is capable of detecting errors in received packets but cannot request retransmission. The packets passed on by the receiver may be without error, but there is no guarantee that all packets in the sequence are received. Hence the link is considered fundamentally unreliable. There are limited uses for such links, and these uses are normally dependent on the continuous repetition of data from the higher layers while it is valid.

3.1.3.3 AMP Reliability

The reliability of each AMP depends on the underlying MAC/PHY technology. Some AMPs only flush user traffic when marked as flushable whereas other AMPs can flush user traffic.

The Bluetooth Core maintains a level of reliability for protocols and profiles above the Core by mandating the use of Enhanced Retransmission Mode or Streaming Mode for any L2CAP channel used over the AMP.

3.2 TRANSPORT ARCHITECTURE ENTITIES

The Bluetooth transport architecture entities are shown in Figure 3.3 on page 45 and are described from the lowest layer upwards in the subsequent sections.

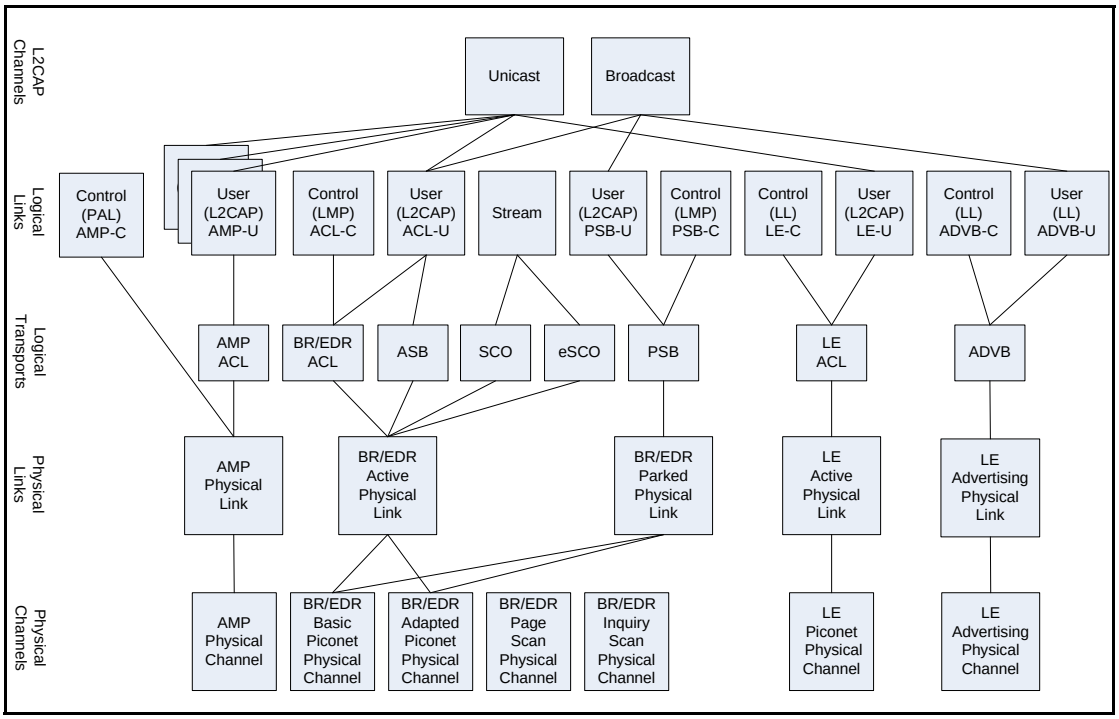


Figure 3.3: Overview of transport architecture entities and hierarchy

3.2.1 BR/EDR Generic Packet Structure

The general packet structure nearly reflects the architectural layers found in the Bluetooth BR/EDR system. The BR/EDR packet structure is designed for optimal use in normal operation. It is shown in Figure 3.4 on page 46.

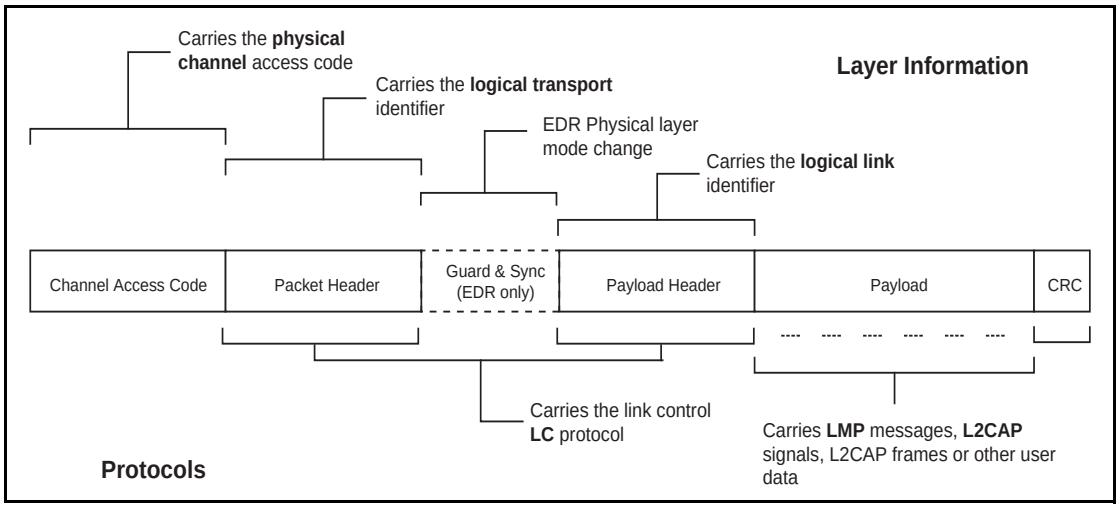


Figure 3.4: BR/EDR packet structure

Packets normally only include the fields that are necessary to represent the layers required by the transaction. Thus a simple inquiry request over an inquiry scan physical channel does not create or require a logical link or higher layer and therefore consists only of the channel access code (associated with the physical channel.) General communication within a piconet uses packets that include all of the fields, as all of the architectural layers are used.

All packets include the channel access code. This is used to identify communications on a particular physical channel, and to exclude or ignore packets on a different physical channel that happens to be using the same RF carrier in physical proximity.

There is no direct field within the BR/EDR packet structure that represents or contains information relating to physical links. This information is implied in the logical transport address (LT_ADDR) carried in the packet header.

Most BR/EDR packets include a packet header. The packet header is always present in packets transmitted on physical channels that support physical links, logical transports and logical links. The packet header carries the LT_ADDR, which is used by each receiving device to determine if the packet is addressed to the device and is used to route the packet internally.

The BR/EDR packet header also carries part of the link control (LC) protocol that is operated per logical transport (except for ACL and SCO transports that operate a shared LC protocol carried on either logical transport.)

The Enhanced Data Rate (EDR) packets have a guard time and synchronization sequence before the payload. This is a field used for physical layer change of modulation scheme.

The payload header is present in all packets on logical transports that support multiple logical links. The payload header includes a logical link identifier field used for routing the payload, and a field indicating the length of the payload. Some packet types also include a CRC after the packet payload that is used to detect most errors in received packets.

EDR packets have a trailer after the CRC.

The packet payload is used to transport the user data. The interpretation of this data is dependent on the logical transport and logical link identifiers. For ACL logical transports Link Manager Protocol (LMP) messages and L2CAP signals are transported in the packet payload, along with general user data from applications. For SCO and eSCO logical transports the payload contains the user data for the logical link.

3.2.2 LE Generic Packet Structure

The general structure of the link layer air interface packet closely reflects the architectural layers found in the LE system. The LE packet structure is designed for optimal use in normal operation. It is shown in Figure 3.5.

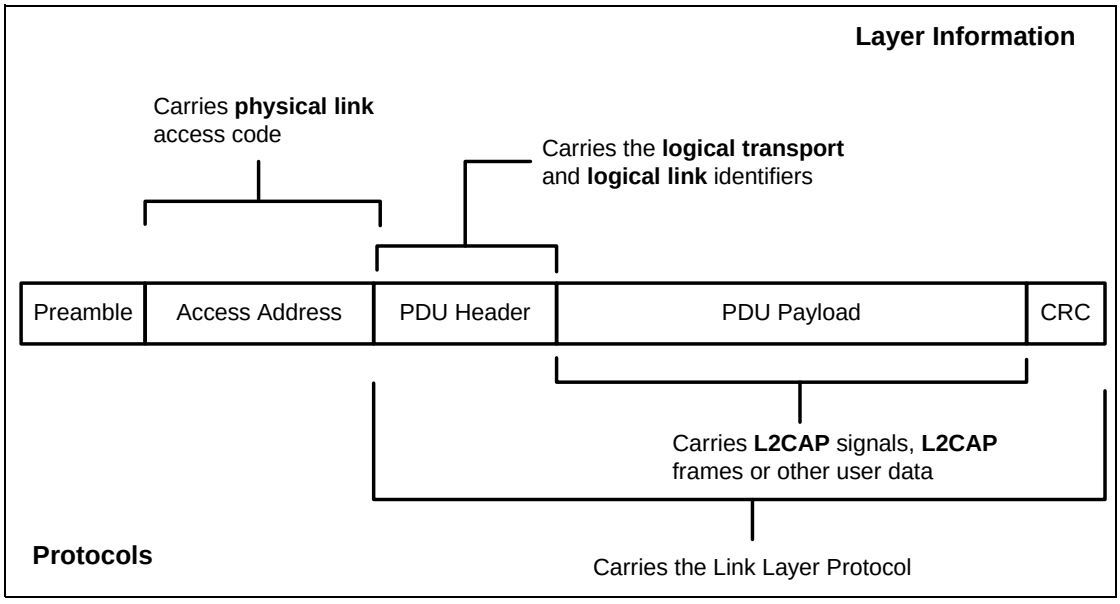


Figure 3.5: LE packet structure

The physical channel identifier is not contained in the link layer air interface packet. The physical channel identifiers are either fixed or are determined at connection setup. All LE packets include the Access Address. This is used to identify communications on a physical link, and to exclude or ignore packets on different physical links that are using the same PHY channels in physical prox-

imity. The Access Address determines whether the packet is directed to the advertising physical link or to an active physical link to a device. All LE advertising physical links use a fixed Access Address. LE active physical links use a randomly generated 32-bit value as their Access Address. This provides an high number of active devices that can be addressed in an LE piconet.

All LE packets include a PDU header. The PDU header determines the type of advertisement broadcast or logical link carried over the physical channel.

For advertising channel PDUs, the PDU header contains the type of advertisement payload, the device address type for addresses contained in the advertisement and advertising channel PDU payload length. Most advertising channel PDU payloads contain the advertiser's address and advertising data. One advertising channel PDU payload only contains the advertiser's device address and the initiator's device address in which the advertisement is directed. Advertising channel PDUs with scan requests payloads contain the scanner's device address and the advertiser's device address. Advertising channel PDUs with scan responses contain advertiser's device address and the scan response data. Advertising channel PDUs with connection request payloads contain the initiator's device address, advertiser's device address and connection setup parameters.

For data channel PDUs, the PDU header contains the Logical Link Identifier (LLID), the Next Expected Sequence Number (NESN), Sequence Number (SN), More Data (MD) and payload length. For data channel PDUs that contain control commands, the data channel PDU payload contains a command opcode and control data that is specific to the command. There is an optional Message Integrity Check (MIC) value that is used to authenticate the data PDU. For data channel PDUs that are data, the data channel PDU payload contains L2CAP data.

3.3 PHYSICAL CHANNELS

The lowest architectural layer in the Bluetooth system is the physical channel. A number of types of physical channel are defined. All Bluetooth physical channels are characterized by a PHY frequency combined with temporal parameters and restricted by spatial considerations. For the basic and adapted piconet physical channels frequency hopping is used to change frequency periodically to reduce the effects of interference and for regulatory reasons.

The Bluetooth BR/EDR system and LE system differ slightly in the way they use physical channels.

3.3.1 BR/EDR Physical Channels

In the BR/EDR core system, peer devices use a shared physical channel for communication. To achieve this their transceivers need to be tuned to the same PHY frequency at the same time, and they need to be within a nominal range of each other.

Given that the number of RF carriers is limited and that many Bluetooth devices may be operating independently within the same spatial and temporal area there is a strong likelihood of two independent Bluetooth devices having their transceivers tuned to the same RF carrier, resulting in a physical channel collision. To mitigate the unwanted effects of this collision each transmission on a physical channel starts with an access code that is used as a correlation code by devices tuned to the physical channel. This channel access code is a property of the physical channel. The access code is present at the start of every transmitted packet.

Four BR/EDR channels are defined. Each is optimized and used for a different purpose. Two of these physical channels (the basic piconet channel and adapted piconet channel) are used for communication between connected devices and are associated with a specific piconet. The remaining BR/EDR physical channels are used for discovering Bluetooth devices and for connecting Bluetooth devices. In BR/EDR this is accomplished through the use of inquiry scan and page scan channels respectively.

A Bluetooth device can only use one BR/EDR physical channel at any given time. In order to support multiple concurrent operations the device uses time-division multiplexing between the channels. In this way a Bluetooth device can appear to operate simultaneously in several piconets, as well as being discoverable and connectable.

Whenever a Bluetooth device is synchronized to the timing, frequency and access code of a physical channel it is said to be ‘connected’ to this channel (whether or not it is actively involved in communications over the channel). The Bluetooth specification assumes that a device is only capable of connecting to one physical channel at any time. Advanced devices may be capable of con-

necting simultaneously to more than one physical channel, but the specification does not assume that this is possible.

3.3.1.1 Basic Piconet Channel

3.3.1.1.1 Overview

The basic piconet channel is used for communication between connected devices during normal operation.

3.3.1.1.2 Characteristics

The basic piconet channel is characterized by a pseudo-random sequence hopping through the PHY channels. The hopping sequence is unique for the piconet and is determined by the Bluetooth device address of the master. The phase in the hopping sequence is determined by the Bluetooth clock of the master. All Bluetooth devices participating in the piconet are time- and hop-synchronized to the channel.

The channel is divided into time slots where each slot corresponds to an PHY hop frequency. Consecutive hops correspond to different PHY hop frequencies. The time slots are numbered according to the Bluetooth clock of the piconet master. Packets are transmitted by Bluetooth devices participating in the piconet aligned to start at a slot boundary. Each packet starts with the channel's access code, which is derived from the Bluetooth device address of the piconet.

On the basic piconet channel the master controls access to the channel. The master starts its transmission in even-numbered time slots only. Packets transmitted by the master are aligned with the slot start and define the piconet timing. Packets transmitted by the master may occupy up to five time slots depending on the packet type.

Each master transmission is a packet carrying information on one of the logical transports. Slave devices may transmit on the physical channel in response. The characteristics of the response are defined by the logical transport that is addressed.

For example, on the asynchronous connection-oriented logical transport (ACL), the addressed slave device responds by transmitting a packet containing information for the same logical transport that is nominally aligned with the next (odd-numbered) slot start. Such a packet may occupy up to five time slots, depending on the packet type. On a broadcast logical transport no slaves are allowed to respond.

A special characteristic of the basic piconet physical channel is the use of some reserved slots to transmit a beacon train. The beacon train is only used if the piconet physical channel has parked slaves connected to it. In this situation the master transmits a packet in the reserved beacon train slots (these packets are used by the slave to resynchronize to the piconet physical channel.) The

master may transmit packets from any logical transport onto these slots, providing there is a transmission starting in each of the slots. In the case where there is information from the parked slave broadcast (PSB) logical transport to be transmitted then this is transmitted in the beacon train slots and takes priority over any other logical transport.

3.3.1.1.3 Topology

A basic piconet channel may be shared by any number of Bluetooth devices, limited only by the resources available on the piconet master device. Only one device is the piconet master, all others being piconet slaves. All communication is between the master and slave devices. There is no direct communication between slave devices on the piconet channel.

There is, however, a limitation on the number of logical transports that can be supported within a piconet. This means that although there is no theoretical limit to the number of Bluetooth devices that share a channel there is a limit to the number of these devices that can be actively involved in exchanging data with the master.

3.3.1.1.4 Supported layers

The basic piconet channel supports a number of physical links, logical transports, logical links and L2CAP channels used for general purpose communications.

3.3.1.2 Adapted Piconet Channel

3.3.1.2.1 Overview

The adapted piconet channel differs from the basic piconet channel in two ways. First, the frequency on which a slave transmits is the same as the frequency used by its master in the preceding transmission. In other words the frequency is not recomputed between master and subsequent slave packets. Second, the adapted piconet channel may be based on fewer than the full 79 frequencies. A number of frequencies may be excluded from the hopping pattern by being marked as “unused”. The remainder of the 79 frequencies are included. The two sequences are the same except that whenever the basic pseudo-random hopping sequence selects an unused frequency, it is replaced with an alternative chosen from the used set.

Because the adapted piconet channel uses the same timing and access code as the basic piconet channel, the two channels are often coincident. This provides a deliberate benefit as it allows slaves in either the basic piconet channel or the adapted piconet channel to adjust their synchronization to the master.

The topology and supported layers of the adapted piconet physical channel are identical to the basic piconet physical channel.

3.3.1.3 Inquiry scan channel

3.3.1.3.1 Overview

In order for a device to be discovered, an inquiry scan channel is used. A discoverable device listens for inquiry requests on its inquiry scan channel and then sends a response to that request. In order for a device to discover other devices, it iterates (hops) through all possible inquiry scan channel frequencies in a pseudo-random fashion, sending an inquiry request on each frequency and listening for any response.

3.3.1.3.2 Characteristics

Inquiry scan channels follow a slower hopping pattern and use an access code to distinguish between occasional occupancy of the same radio frequency by two co-located devices using different physical channels.

The access code used on the inquiry scan channel is taken from a reserved set of inquiry access codes that are shared by all Bluetooth devices. One access code is used for general inquiries, and a number of additional access codes are reserved for limited inquiries. Each device has access to a number of different inquiry scan channels. As all of these channels share an identical hopping pattern, a device may concurrently occupy more than one inquiry scan channel if it is capable of concurrently correlating more than one access code.

A device using one of its inquiry scan channels remains passive on that channel until it receives an inquiry message on this channel from another Bluetooth device. This is identified by the appropriate inquiry access code. The inquiry scanning device will then follow the inquiry response procedure to return a response to the inquiring device.

In order for a device to discover other Bluetooth devices it uses the inquiry scan channel to send inquiry requests. As it has no prior knowledge of the devices to discover, it cannot know the exact characteristics of the inquiry scan channel.

The device takes advantage of the fact that inquiry scan channels have a reduced number of hop frequencies and a slower rate of hopping. The inquiring device transmits inquiry requests on each of the inquiry scan hop frequencies and listens for an inquiry response. Transmissions are done at a faster rate, allowing the inquiring device to cover all inquiry scan frequencies in a reasonably short time period.

3.3.1.3.3 Topology

Inquiring and discoverable devices use a simple exchange of packets to fulfill the inquiring function. The topology formed during this transaction is a simple and transient point-to-point connection.

3.3.1.3.4 Supported Layers

During the exchange of packets between an inquiring and discoverable device it may be considered that a temporary physical link exists between these devices. However, the concept is quite irrelevant as it has no physical representation but is only implied by the brief transaction between the devices. No further architectural layers are considered to be supported.

3.3.1.4 Page Scan Channel

3.3.1.4.1 Overview

A connectable device (one that is prepared to accept connections) does so using a page scan channel. A connectable device listens for a page request on its page scan channel and, once received, enters into a sequence of exchanges with this device. In order for a device to connect to another device, it iterates (hops) through all page scan channel frequencies in a pseudo-random fashion, sending a page request on each frequency and listening for a response.

3.3.1.4.2 Characteristics

The page scan channel uses an access code derived from the scanning device's Bluetooth device address to identify communications on the channel. The page scan channel uses a slower hopping rate than the hop rate of the basic and adapted piconet channels. The hop selection algorithm uses the Bluetooth device clock of the scanning device as an input.

A device using its page scan channel remains passive until it receives a page request from another Bluetooth device. This is identified by the page scan channel access code. The two devices will then follow the page procedure to form a connection. Following a successful conclusion of the page procedure both devices switch to the basic piconet channel that is characterized by having the paging device as master.

In order for a device to connect to another Bluetooth device it uses the page scan channel of the target device in order to send page requests. If the paging device does not know the phase of the target device's page scan channel it therefore does not know the current hop frequency of the target device. The paging device transmits page requests on each of the page scan hop frequencies and listens for a page response. This is done at a faster hop rate, allowing the paging device to cover all page scan frequencies in a reasonably short time period.

The paging device may have some knowledge of the target device's Bluetooth clock (indicated during a previous inquiry transaction between the two devices, or as a result of a previous involvement in a piconet with the device), in this case it is able to predict the phase of the target device's page scan channel. It

may use this information to optimize the synchronization of the paging and page scanning process and speed up the formation of the connection.

3.3.1.4.3 Topology

Paging and connectable devices use a simple exchange of packets to fulfill the paging function. The topology formed during this transaction is a simple and transient point-to-point connection.

3.3.1.4.4 Supported Layers

During the exchange of packets between a paging and connectable device it may be considered that a temporary physical link exists between these devices. However, the concept is quite irrelevant as it has no physical representation but is only implied by the brief transaction between the devices. No further architectural layers are considered to be supported.

3.3.2 LE Physical Channels

In the LE core system, two Bluetooth devices use a shared physical channel for communication. To achieve this, their transceivers need to be tuned to the same PHY frequency at the same time, and they need to be within a nominal range of each other.

Given that the number of PHY channels is limited, and many Bluetooth devices can be operating independently within the same spatial and temporal area, there is a strong likelihood of two pairs of independent Bluetooth devices having their transceivers tuned to the same PHY channel, resulting in a physical channel collision. Unlike BR/EDR where an access code is used to identify the piconet, LE uses a randomly generated Access Address to identify a physical link between devices. In the event that two devices happen to share the same PHY channel in the same area, the targeted device Access Address is used as a correlator to determine to which device the communication is directed.

Two LE physical channels are defined. Each is optimized and used for a different purpose. The LE piconet channel is used for communication between connected devices and is associated with a specific piconet. The LE advertisement broadcast channel is used for broadcasting advertisements to LE devices. These advertisements can be used to discover, connect or send user data to scanner or initiator devices.

An LE device can only use one of these LE physical channels at any given time. In order to support multiple concurrent operations the device uses time-division multiplexing between the channels. In this way a Bluetooth device can appear to support connected devices while simultaneously sending advertising broadcasts.

Whenever an LE device is synchronized to the timing and frequency of the physical channel it is said to be connected to this channel (whether or not it is

actively involved in communications over the channel). The Bluetooth specification assumes that a device is only capable of connecting to one physical channel at a time. Advanced devices may be capable of connecting simultaneously to more than one physical channel, but the specification does not make this assumption.

3.3.2.1 LE piconet channel

3.3.2.1.1 Overview

The LE piconet channel is used for communication between connected LE devices during normal operation.

3.3.2.1.2 Characteristics

The LE piconet channel is characterized by a pseudo-random sequence of PHY channels and by three additional parameters provided by the master. The first is the channel map that indicates the set of PHY channels used in the piconet. The second is a pseudo random number used as an index into the complete set of PHY channels. The third is the timing of the first data packet sent by the master after the connection request.

The channel is divided into connection events where each connection event corresponds to a PHY hop channel. Consecutive connection events correspond to different PHY hop channels. The first packet sent by the master after the connection establishment sets an anchor point for the timing of all future connection events. In a connection event the master transmits packets to a slave in the piconet and the slave may or may not respond with a packet of its own.

On the LE piconet channel the master controls access to the channel. The master starts its transmission in a connection event that occurs at regular intervals. Packets transmitted by the master are aligned with the connection event start and define the piconet timing. Packets sent by the master may vary in length from 10 to 47 octets.

Each master transmission contains a packet carrying information on one of the logical transports. Slave devices can transmit on the physical channel in response. The characteristics of the response are defined by the logical transport that is addressed.

The LE piconet channel is similar to the BR/EDR adapted piconet channel in that the set of PHY channels used can be modified to avoid interference. The set of used channels in the channel map is established by the master during connection setup. While in a connection the master can change the channel map when necessary to avoid new interferers.

There are 37 LE piconet channels. The master can reduce this number through the channel map indicating the used channels. When the hopping pattern hits

an unused channel the unused channel is replaced with an alternate from the set of used channels.

3.3.2.1.3 Topology

An LE piconet channel can be shared by any number of LE devices, limited only by the resources available on the piconet master device. Only one device is the piconet master; all others being piconet slaves. All communication is between the master and slave devices. There is no direct communication between slave devices on the piconet channel.

There is virtually no limit on the number of logical transports that can be supported within a piconet. This means that there is no theoretical limit to the number of Bluetooth devices that share a channel with the master.

At present an LE device is only permitted to belong to one piconet at a time.

3.3.2.1.4 Supported layers

The LE piconet channel supports a number of physical links, logical transports, logical links and L2CAP channels used for general purpose communications.

3.3.2.2 Advertisement broadcast channel

3.3.2.2.1 Overview

The LE advertising channel is used to set up connections between two devices or to communicate broadcast information between unconnected devices.

3.3.2.2.2 Characteristics

The LE advertisement broadcast channel is a set of three fixed PHY channels spread evenly across the LE frequency spectrum. The number of advertising PHY channels can be reduced by the advertising device in order to reduce interference. All advertising packets use a fixed Access Address.

The channel is divided into advertising events where each advertising event can hop on all three advertising PHY channels. Consecutive advertising events begin on the first advertising PHY channel. The advertising events occur at regular intervals which are slightly modified with a random delay to aid in interference avoidance.

On the LE advertising broadcast channel the advertising device controls access to the channel. The advertising device starts its transmission in an advertising event and transmits advertising packets on one or more of the three advertising PHY channels. Each advertising packet is sent on a different advertising PHY channel at a fixed interval. Four types of advertising events can be

used, with each advertising event type having different sized advertising packets. These advertising packets can vary in length from 8 to 39 octets.

Some advertising events sent by the advertising device permit the listening device to concurrently send scan requests or connection requests packets on the same advertising PHY channel in which the advertising packet was received. The advertising device can send a scan response packet again on the same advertising PHY channel within the same advertising event. The scan response packet can vary in length from 8 to 39 octets.

3.3.2.2.3 Topology

An LE advertising channel can be shared by any number of LE devices. Any number of LE devices can transmit advertising packets while sharing the same three advertising PHY channels. Any number of scanning devices can listen on the advertising channel. An advertising device can advertise and be connected on an LE piconet channel simultaneously. If the advertising device is connected on an LE piconet channel then restrictions apply to the types of advertising events the device can use. Scanning devices may also be connected to an LE piconet channel simultaneously; however, if the device is connected to an LE piconet channel as a slave, it cannot make connection requests

3.3.3 AMP physical channel

3.3.3.1 Overview

An AMP physical channel is used for communication between connected devices during normal operation. It is used in parallel with an adapted piconet channel between the associated BR/EDR Controllers.

3.3.3.2 Characteristics

The AMP physical channel characteristics depend on the referenced MAC and PHY. See Volume 5 for the referenced MAC and PHY for each PAL.

3.3.3.3 Topology

An AMP physical channel may be shared by any number of Bluetooth devices, limited only by the resources available on the piconet BR/EDR master device and the number of LT_ADDRs available. Roles within the AMP physical channel are not used. All communication is between the BR/EDR piconet master and a single BR/EDR piconet slave. There is no direct communication between BR/EDR piconet slave devices over the AMP physical channel.

Although there is no theoretical limit to the number of Bluetooth devices that can share an AMP physical channel, similar to the basic and adapted piconet channels, there is a limit to the number of these devices that can be actively involved in exchanging data with the master.

3.3.3.4 Supported Layers

The AMP physical channel supports a number of physical links, logical transports, logical links and L2CAP channels used for general purpose communications.

3.4 PHYSICAL LINKS

A physical link represents a baseband connection between Bluetooth devices. A physical link is always associated with exactly one physical channel (although a physical channel may support more than one physical link.) Within the Bluetooth system a physical link is a virtual concept that has no direct representation within the structure of a transmitted packet.

In BR/EDR the access code packet field, together with the clock and address of the master Bluetooth device, is used to identify a physical channel. However, in BR/EDR, there is no subsequent part of the packet that directly identifies the physical link. Instead the physical link may be identified by association with the logical transport, as each logical transport is only received on one physical link.

In LE the hop index and channel map are used to identify a physical channel. The physical link is identified by the access address in the LE packet.

Some physical link types have properties that may be modified. An example of this is the transmit power for the link. Other physical link types have no modifiable properties. In the case of physical links with modifiable properties the LM protocol (BR/EDR) is used to adapt these properties. As the LM protocol (BR/EDR) is supported at a higher layer (by a logical link) the appropriate physical link is identified by implication from the logical link that transports the LM signaling.

In the situation where a transmission is broadcast over a number of different physical links, then the transmission parameters are selected to be suitable for all of the physical links.

3.4.1 BR/EDR Links Supported By The Basic And Adapted Piconet Physical Channel

The basic and adapted piconet physical channels support a physical link which may be active or parked. The physical link is a point-to-point link between the master and a slave. It is always present when the slave is synchronized in the piconet.

3.4.1.1 Active Physical Link

The physical link between a master and a slave device is active if a default ACL logical transport exists between the devices. Active physical links have no direct identification of their own, but are identified by association with the default ACL logical transport ID with which there is a one-to-one correspondence.

An active physical link has the associated property of radio transmit power in each direction. Transmissions from slave devices are always directed over the active physical link to the master, and use the transmit power that is a property of this link in the slave to master direction. Transmissions from the master may be directed over a single active physical link (to a specific slave) or over a number of physical links (to a group of slaves in the piconet.) In the case of point-to-point transmissions the master uses the appropriate transmit power for the physical link in question. (In the case of point-to-multipoint transmissions the master uses a transmit power appropriate for the set of devices addressed.)

Active physical links may be placed into Hold or Sniff mode. The effect of these modes is to modify the periods when the physical link is active and may carry traffic. Logical transports that have defined scheduling characteristics are not affected by these modes and continue according to their pre-defined scheduling behavior. The default ACL logical transport and other links with undefined scheduling characteristics are subject to the mode of the active physical link.

3.4.1.2 Parked Physical Link

The physical link between a master and a slave device is parked when the slave remains synchronized in the piconet, but has no default ACL logical transport. Such a slave is also said to be parked. A beacon train is used to provide regular synchronization to all parked slaves connected to the piconet physical channel. A parked slave broadcast (PSB) logical transport is used to allow communication of a subset of LMP signaling and broadcast L2CAP to parked slaves. The PSB logical transport is closely associated with the beacon train.

A slave is parked (its active link is changed to a parked link) using the park procedure. The master is not allowed to park a slave that has any user created logical transport supported by the physical link. These logical transports are first removed, and any L2CAP channels that are built on these logical transports are also removed. The broadcast logical transport and default ACL logical transports are not considered as user created and are not explicitly removed. When the active link is replaced with a parked link the default ACL logical transport is implicitly removed. The supported logical links and L2CAP channels remain in existence, but become suspended. It is not possible to use these links and L2CAP channels to transport signaling or data while the active link is absent.

A parked slave may become active using the unpark procedure. This procedure is requested by the slave at an access window and initiated by the master. Following the unpark procedure the parked physical link is changed to an active physical link and the default ACL logical transport is re-created. L2CAP channels that were suspended during the most recent park procedure are associated with the new default ACL logical transport and become active again.

Parked links do not support radio power control, as there is no feedback path from parked slaves to the piconet master that can be used to provide the received signal strength at the slave or for the master to measure received signal strength from the slave. Transmissions are carried out at nominal power on parked links.

Parked links use the same physical channel as their associated active link. If a master manages a piconet that contains parked slaves using the basic piconet physical channel and also parked slaves using the adapted piconet physical channel then it must create a parked slave broadcast logical transport (and associated transport) for each of these physical channels.

A parked slave may use the inactive periods of the parked slave broadcast logical transport to save power, or it may carry out activities on other physical channels unrelated to the piconet within which it is parked.

3.4.2 BR/EDR Links Supported by the Scanning Physical Channels

In the case of inquiry scan and page scan channels, the physical link exists for a relatively short time and cannot be controlled or modified in any way. These types of physical links are not further elaborated.

3.4.3 LE Links Supported by the LE Physical Channels

The LE piconet physical channels support an LE active physical link. The physical link is a point-to-point link between the master and a slave. It is always present when the slave is in a connection with the master.

The LE advertising physical channels support an LE advertising physical link. The physical link is a broadcast between the advertiser device and one or more scanner or initiator devices. It is always present when the advertiser is broadcasting advertisement events.

3.4.3.1 Active physical link

The physical link between a master and a slave device is active if a default LE ACL logical transport exists between the devices. Active physical links are identified by the randomly generated Access Address used in the Link Layer packet. Each Access Address has a one-to-one relationship with the master and the slave of the active physical link.

3.4.3.2 Advertising physical link

An advertising physical link between an advertising device and an initiating device for the purposes of forming a connection (active physical link) can exist for a relatively short period of time. These advertising physical links cannot be controlled or modified in any way and these types of physical links are not further elaborated.

An advertising physical link between an advertising device and a scanning device used for periodic broadcasting of user data can exist for longer periods of time. There is no identification information about the physical link within the protocol. The relationship between the advertising and scanning device shall be established through the use of the Bluetooth device address.

3.4.4 Links Supported by the AMP Physical Channels

An AMP physical link is always associated with exactly one AMP physical channel (although an AMP physical channel may support more than one AMP physical link). The AMP physical link is a point-to-point link between two PALs. The characteristics of an AMP physical link depend on the underlying AMP technology.

3.5 LOGICAL LINKS AND LOGICAL TRANSPORTS

A variety of logical links are available to support different application data transport requirements. Each logical link is associated with a logical transport, which has a number of characteristics. These characteristics include flow control, acknowledgement/repeat mechanisms, sequence numbering and scheduling behavior. Logical transports are able to carry different types of logical links (depending on the type of the logical transport). In the case of some of the Bluetooth logical links these are multiplexed onto the same logical transport. Logical transports may be carried by active physical links on either the basic or the adapted piconet physical channel.

Logical transport identification and real-time (link control) signaling are carried in the packet header, and for some logical links identification is carried in the payload header. Control signaling that does not require single slot response times is carried out using the LMP protocol.

Table 3.1 on page 61 lists all of the logical transport types, the supported logical link types, which type of physical links and physical channels can support them, and a brief description of the purpose of the logical transport.

Logical transport	Links supported	Supported by	Bearer	Overview
Asynchronous Connection-Oriented (ACL)	Control (LMP) ACL-C or (PAL) AMP-C User (L2CAP) ACL-U or AMP-U	BR/EDR active physical link, BR/EDR basic or adapted piconet physical channel, AMP physical link, AMP physical channel.	BR/EDR, AMP	Reliable or time-bounded, bi-directional, point-to-point.

Table 3.1: Logical transport types.

Logical transport	Links supported	Supported by	Bearer	Overview
Synchronous Connection-Oriented (SCO)	Stream (unframed) SCO-S	BR/EDR active physical link, BR/EDR basic or adapted piconet physical channel	BR/EDR	Bi-directional, symmetric, point-to-point, AV channels. Used for 64Kb/s constant rate data.
Extended Synchronous Connection-Oriented (eSCO)	Stream (unframed) eSCO-S	BR/EDR active physical link, BR/EDR basic or adapted piconet physical channel	BR/EDR	Bi-directional, symmetric or asymmetric, point-to-point, general regular data, limited retransmission. Used for constant rate data synchronized to the master Bluetooth clock.
Active Slave Broadcast (ASB)	User (L2CAP) ASB-U	BR/EDR active physical link, basic or adapted physical channel.	BR/EDR	Unreliable, uni-directional broadcast to any devices synchronized with the physical channel. Used for broadcast L2CAP groups.
Parked Slave Broadcast (PSB)	Control (LMP) PSB-C, User (L2CAP) PSB-U	BR/EDR parked physical link, basic or adapted piconet physical channel.	BR/EDR	Unreliable, uni-directional broadcast to all piconet devices. Used for LMP and L2CAP traffic to parked devices, and for access requests from parked devices.
LE asynchronous connection (LE ACL	Control (LL) LE-C, User (L2CAP) LE-U	LE active physical link, LE piconet physical channel.	LE	Reliable or time-bounded, bi-directional, point-to-point.
LE Advertising Broadcast (ADVB)	Control (LL) ADVB-C, User (LL) ADVB-U	LE advertising physical link, LE piconet physical channel.	LE	LE advertising physical link, LE piconet physical channel.

Table 3.1: Logical transport types.

The classification of each link type follows from a selection procedure within three categories.

3.5.1 Casting

The first category is that of casting. This may be either unicast or broadcast.

- *Unicast links.* Unicast links exist between exactly two endpoints. Traffic may be sent in either direction on unicast links.
- *Broadcast links.* Broadcast links exist between one source device and zero or more receiver devices. Traffic is unidirectional, i e., only sent from the source devices to the receiver devices. Broadcast links are connectionless, meaning there is no procedure to create these links, and data may be sent over them at any time. Broadcast links are unreliable, and there is no guarantee that the data will be received.

3.5.2 Scheduling and Acknowledgement Scheme

The second category relates to the scheduling and acknowledgement scheme of the link, and implies the type of traffic that is supported by the link. These are synchronous, isochronous or asynchronous. There are no specific isochronous links defined, though the default ACL link can be configured to operate in this fashion.

- *Synchronous links.* Synchronous links provide a method of associating the transported data with the Bluetooth piconet clock. This is achieved by reserving regular slots on the physical channel, and transmitting fixed size packets at these regular intervals. Such links are suitable for constant rate isochronous data.
- *Asynchronous links.* Asynchronous links provide a method for transporting data that has no time-based characteristics. The data is normally expected to be retransmitted until successfully received, and each data entity can be processed at any time after receipt, without reference to the time of receipt of any previous or successive entity in the stream (providing the ordering of data entities is preserved.)
- *Isochronous links.* Isochronous links provide a method for transporting data that has time-based characteristics. The data may be retransmitted until received or expired. The data rate on the link need not be constant (this being the main difference from synchronous links.)

3.5.3 Class of Data

The final category is related to the class of data that is carried by the link. This is either control (LMP or PAL) data or user data. The user data category is subdivided into L2CAP (or framed) data and stream (or unframed) data.

- *Control links.* Control links are only used for transporting LMP messages between two link managers. These links are invisible above the baseband layer, and cannot be directly instantiated, configured or released by applications, other than by the use of the connection and disconnection services that have this effect implicitly. Control links are always multiplexed with an equivalent L2CAP link onto an ACL logical transport. Subject to the rules defining the ARQ scheme, the control link traffic always takes priority over the L2CAP link traffic.

- *L2CAP links.* L2CAP links are used to transport L2CAP PDUs, which may carry the L2CAP signaling channel (on the default ACL-U logical link only) or framed user data submitted to user-instantiated L2CAP channels. L2CAP frames submitted to the baseband may be larger than the available baseband packets. A link control protocol embedded within the LLID field preserves the frame-start and frame-continuation semantics when the frame is transmitted in a number of fragments to the receiver.
- *Stream links.* Stream links are used to transport user data that has no inherent framing that should be preserved when delivering the data. Lost data may be replaced by padding at the receiver.

3.5.4 Logical Transports

3.5.4.1 BR/EDR Asynchronous Connection-oriented (ACL)

The asynchronous connection-oriented (ACL) logical transport is used to carry LMP and L2CAP control signaling and best effort asynchronous user data. The ACL logical transport uses a 1-bit ARQN/SEQN scheme to provide simple channel reliability. Every active slave device within a piconet has one ACL logical transport to the piconet master, known as the default ACL.

The default ACL is created between the master and the slave when a device joins a piconet (connects to the basic piconet physical channel). This default ACL is assigned a logical transport address (LT_ADDR) by the piconet master. This LT_ADDR is also used to identify the active physical link when required (or as a piconet active member identifier, effectively for the same purpose.)

The LT_ADDR for the default ACL is reused for synchronous connection-oriented logical transports between the same master and slave. (This is for reasons of compatibility with earlier Bluetooth specifications.) Thus the LT_ADDR is not sufficient on its own to identify the default ACL. However the packet types used on the ACL are different from those used on the synchronous connection-oriented logical transport. Therefore, the ACL logical transport can be identified by the LT_ADDR field in the packet header in combination with the packet type field.

The default ACL may be used for isochronous data transport by configuring it to automatically flush packets after the packets have expired. Asynchronous traffic can be sent over an ACL logical transport configured for isochronous traffic by marking the asynchronous packets as non-automatically-flushable. This allows both isochronous and asynchronous traffic to be transferred at the same time to a single device.

If the default ACL is removed from the active physical link then all other logical transports that exist between the master and the slave are also removed. In the case of unexpected loss of synchronization to the piconet physical channel the physical link and all logical transports and logical links cease to exist at the time that this synchronization loss is detected.

A device may remove its default ACL (and by implication its active physical link) but remain synchronized to the piconet. This procedure is known as parking, and a device that is synchronized to the piconet, but has no active physical link is parked within that piconet.

When the device transitions to the parked state the default ACL logical links that are transported on the default ACL logical transport remain in existence, but become suspended. No data may be transferred across a suspended logical link. When the device transitions from the parked state back into connection state, a new default ACL logical transport is created (it may have a different LT_ADDR from the previous one) and the suspended logical links are attached to this default ACL and become active once again.

3.5.4.2 BR/EDR Synchronous Connection-Oriented (SCO)

The synchronous connection-oriented (SCO) logical transport is a symmetric, point-to-point transport between the master and a specific slave. The SCO logical transport reserves slots on the physical channel and can therefore be considered as a circuit-switched connection between the master and the slave. SCO logical transports carry 64 kb/s of information synchronized with the piconet clock. Typically this information is an encoded voice stream. Three different SCO configurations exist, offering a balance between robustness, delay and bandwidth consumption.

Each SCO-S logical link is supported by a single SCO logical transport, which is assigned the same LT_ADDR as the default ACL logical transport between the devices. Therefore the LT_ADDR field is not sufficient to identify the destination of a received packet. Because the SCO links use reserved slots, a device uses a combination of the LT_ADDR, the slot numbers (a property of the physical channel) and the packet type to identify transmissions on the SCO link.

Although slots are reserved for the SCO, it is permissible to use a reserved slot for traffic from another channel that has a higher priority. This may be required as a result of QoS commitments, or to send LMP signaling on the default ACL when the physical channel bandwidth is fully occupied by SCOs. As SCOs carry different packet types to ACLs, the packet type is used to identify SCO traffic (in addition to the slot number and LT_ADDR.)

There are no further architectural layers defined by the Bluetooth core specification that are transported over an SCO link. A number of standard formats are defined for the 64 kb/s stream that is transported, or an unformatted stream is allowed where the application is responsible for interpreting the encoding of the stream.

3.5.4.3 BR/EDR Extended Synchronous Connection-Oriented (eSCO)

The extended synchronous connection-oriented (eSCO) logical transport is a symmetric or asymmetric, point-to-point transport between the master and a

specific slave. The eSCO reserves slots on the physical channel and can therefore be considered as a circuit-switched connection between the master and the slave. eSCO links offer a number of extensions over the standard SCO links, in that they support a more flexible combination of packet types and selectable data contents in the packets and selectable slot periods, allowing a range of synchronous bit rates to be supported.

eSCO links also can offer limited retransmission of packets (unlike SCO links where there is no retransmission.) If these retransmissions are required they take place in the slots that follow the reserved slots, otherwise the slots may be used for other traffic.

Each eSCO-S logical link is supported by a single eSCO logical transport, identified by a LT_ADDR that is unique within the piconet for the duration of the eSCO. eSCO-S links are created using LM signaling and follow scheduling rules similar to SCO-S links.

There are no further architectural layers defined by the Bluetooth core specification that are transported over an eSCO-S link. Instead applications may use the data stream for whatever purpose they require, subject to the transport characteristics of the stream being suitable for the data being transported.

3.5.4.4 BR/EDR Active Slave Broadcast (ASB)

The active slave broadcast logical transport is used to transport L2CAP user traffic to all devices in the piconet that are currently connected to the physical channel that is used by the ASB. There is no acknowledgement protocol and the traffic is uni-directional from the piconet master to the slaves. The ASB channel may be used for L2CAP group traffic (a legacy of the 1.1 specification), and is never used for L2CAP connection-oriented channels, L2CAP control signaling or LMP control signaling.

The ASB logical transport is inherently unreliable because of the lack of acknowledgement. To improve the reliability, each packet is transmitted a number of times. An identical sequence number is used to assist with filtering retransmissions at the slave device.

The ASB logical transport is identified by a reserved LT_ADDR. (The reserved LT_ADDR address is also used by the PSB logical transport.) An active slave will receive traffic on both logical transports, and cannot readily distinguish between them. As the ASB logical transport does not carry LMP traffic an active slave can ignore packets received over the LMP logical link on the ASB logical transport. However L2CAP traffic transmitted over the PSB logical transport is also received by active slaves on the ASB logical transport and cannot be distinguished from L2CAP traffic sent on the ASB transport.

An ASB is implicitly created whenever a piconet exists, and there is always one ASB associated with each of the basic and adapted piconet physical channels that exist within the piconet. Because the basic and adapted piconet physical

channels are mostly coincident, a slave device cannot distinguish which of the ASB channels is being used to transmit the packets. This adds to the general unreliability of the ASB channel. (Although it is, perhaps, no more unreliable than general missed packets.)

A master device may decide to use only one of its two possible ASBs (when it has both a basic and adapted piconet physical channel), as with sufficient retransmissions it is possible to address both groups of slaves on the same ASB channel.

The ASB channel is never used to carry LMP or L2CAP control signals.

3.5.4.5 BR/EDR Parked Slave Broadcast (PSB)

The parked slave broadcast logical transport is used for communications between the master and slaves that are parked (have given up their default ACL logical transport.) The parked slave broadcast link is the only logical transport that exists between the piconet master and parked slaves.

The PSB logical transport is more complex than the other logical transports as it consists of a number of phases, each having a different purpose. These phases are the control information phase (used to carry the LMP logical link), the user information phase (used to carry the L2CAP logical link), and the access phase (carrying baseband signaling.) The control information and broadcast information phases are usually mutually exclusive as only one of them can be supported in a single beacon interval. (Even if there is no control or user information phase, the master is still required to transmit a packet in the beacon slots so that the parked slaves can resynchronize.) The access phase is normally present unless cancelled in a control information message.

The control information phase is used for the master to send information to the parked slaves containing modifications to the PSB transport attributes, modifications to the beacon train attributes, or a request for a parked slave to become active in the piconet (known as unparking). This control information is carried in LMP messages on the LMP logical link. (The control information phase is also present in the case of a user information phase where the user information requires more than one baseband packet.)

Packets in the control information phase are always transmitted in the physical channel beacon train slots, and cannot be transmitted on any other slots. The control information occupies a single DM1 packet and is repeated in every beacon train slot within a single beacon interval. (If there is no control information then there may be a user information phase that uses the beacon slots. If neither phase is used then the beacon slots are used for other logical transport traffic or for NULL packets.)

The user information phase is used for the master to send L2CAP packets that are destined for all piconet slaves. User information may occupy one or more

baseband packets. If the user information occupies a single packet then the user information packet is repeated in each of the piconet channel beacon train slots.

If the user information occupies more than one baseband packet then it is transmitted in slots after the beacon train (the broadcast scan window) and the beacon slots are used to transmit a control information phase message that contains the timing attributes of this broadcast scan window. This is required so that the parked slaves remain connected to the piconet physical channel to receive the user information.

The access phase is normally present unless temporarily cancelled by a control message carried in the control information broadcast phase. The access window consists of a sequence of slots that follow the beacon train. In order for a parked slave to become active in the piconet, it may send such an access request to the piconet master during the access window. Each parked slave is allocated an access request address (not necessarily unique) that controls when during the access window the slave requests access.

The PSB logical transport is identified by the reserved LT_ADDR of 0. This reserved LT_ADDR address is also used by the ASB logical transport. Parked slaves are not normally confused by the duplicated use of the LT_ADDR as they are only connected to the piconet physical channel during the time that the PSB transport is being used.

3.5.4.6 LE Asynchronous Connection (LE ACL)

The LE asynchronous connection (LE ACL) logical transport is used to carry LL and L2CAP control signaling and best effort asynchronous user data. The LE ACL logical transport uses a 2-bit NESN/SN scheme to provide simple channel reliability. Every active slave device within an LE piconet has one LE ACL logical transport to the piconet master, known as the default LE ACL.

The default LE ACL is automatically created between the master and the slave when a device joins a piconet (connects to the LE piconet physical channel). This default LE ACL is assigned an access channel number by the piconet master. This access address is also used to identify the active physical link when required.

If the default LE ACL is removed from the LE active physical link then all other LE logical transports that exist between the master and the slave are also removed. In the case of unexpected loss of synchronization to the LE piconet physical channel the LE physical link and all LE logical transports and LE logical links cease to exist at the time that this synchronization loss is detected.

3.5.4.7 LE Advertising Broadcast (ADVB)

The LE advertising broadcast logical transport is used to transport broadcast control and user data to all scanning devices in a given area. There is no acknowledgement protocol and the traffic is predominately unidirectional from

the advertising device. A scanning device can send requests over the logical transport to get additional broadcast user data, or to form an LE ACL logical transport connection. The LE Advertising Broadcast logical transport data is carried only over the LE advertising broadcast link.

The ADVB logical transport is inherently unreliable because of the lack of acknowledgement. To improve the reliability, each packet is transmitted a number of times over the LE advertising broadcast link.

An ADVB is created whenever an advertising device begins advertising. The ADVB logical transport is identified by the advertiser's Bluetooth device address.

3.5.5 Logical Links

Some logical transports are capable of supporting different logical links, either concurrently multiplexed, or one of the choice.

3.5.5.1 BR/EDR Logical Links

Within BR/EDR logical transports, the logical link is identified by the logical link identifier (LLID) bits in the payload header of baseband packets that carry a data payload. The logical links distinguish between a limited set of core protocols that are able to transmit and receive data on the logical transports. Not all of the logical transports are able to carry all of the logical links (the supported mapping is shown in Figure 3.2 on page 40.) In particular the BR/EDR SCO and eSCO logical transports are only able to carry constant data rate streams, and these are uniquely identified by the LT_ADDR. Such logical transports only use packets that do not contain a payload header, as their length is known in advance, and no LLID is necessary.

3.5.5.1.1 ACL Control Logical Link (ACL-C)

The ACL Control Logical Link (ACL-C) is used to carry BR/EDR LMP signaling between devices in the piconet. The control link is only carried on the default ACL logical transport and on the PSB logical transport (in the control information phase). The ACL-C link is always given priority over the ACL-U (see below) link when carried on the same logical transport.

3.5.5.1.2 User Asynchronous/Isochronous Logical Link (ACL-U)

The user asynchronous/isochronous logical link (ACL-U) is used to carry all asynchronous and isochronous framed user data. The ACL-U link is carried on all but the synchronous logical transports. Packets on the ACL-U link are identified by one of two reserved LLID values. One value is used to indicate whether the baseband packet contains the start of an L2CAP frame and the other indicates a continuation of a previous frame. This ensures correct synchronization of the L2CAP reassembler following flushed packets. The use of

this technique removes the need for a more complex L2CAP header in every baseband packet (the header is only required in the L2CAP start packets), but adds the requirement that a complete L2CAP frame shall be transmitted before a new one is transmitted. (An exception to this rule being the ability to flush a partially transmitted L2CAP frame in favor of another L2CAP frame.)

3.5.5.1.3 User Synchronous/Extended Synchronous Logical Links (SCO-S/ eSCO-S)

Synchronous (SCO-S) and extended synchronous (eSCO-S) logical links are used to support isochronous data delivered in a stream without framing. These links are associated with a single logical transport, where data is delivered in constant sized units at a constant rate. There is no LLID within the packets on these transports, as only a single logical link can be supported, and the packet length and scheduling period are pre-defined and remain fixed during the life-time of the link.

Variable rate isochronous data cannot be carried by the SCO-S or eSCO-S logical links. In this case the data must be carried on ACL-U logical links, which use packets with a payload header.

3.5.5.2 LE Logical Links

Within LE logical transports, the logical link is identified by the logical link identifier (LLID) bits in the payload header of baseband packets that carry a data payload. The logical links distinguish between a limited set of core protocols that are able to transmit and receive data on the logical transports. Not all of the logical transports are able to carry all of the logical links (the supported mapping is shown in Figure 3.2 on page 40).

3.5.5.2.1 Control Logical Link (LE-C)

The LE ACL Control Logical Link (LE-C) is used to carry LE LL signaling between devices in the piconet. The control link is only carried on the default LE ACL logical transport.

3.5.5.2.2 User Asynchronous Logical Link (LE-U)

The user asynchronous logical link (LE-U) is used to carry all asynchronous and framed user data. The LE-U link is carried on the LE logical transport. Packets on the LE-U link are identified by one of two reserved LLID values. One value is used to indicate whether the baseband packet contains the start of an L2CAP frame and the other indicates a continuation of a previous frame or empty PDU. This ensures correct synchronization of the L2CAP re-assembler. The use of this technique removes the need for a more complex L2CAP header in every baseband packet, but adds the requirement that a complete L2CAP frame shall be transmitted before a new one is transmitted.

3.5.5.2.3 Advertising Broadcast Control Logical Link (ADVB-C)

The LE Advertising Broadcast Control Logical Link (ADVB-C) is used to carry LE LL signaling between unconnected devices in a given area. This signaling is the control commands for gathering additional broadcast user data (scan requests) or connection requests. The control link is only carried on the default LE Advertising Broadcast logical transport.

3.5.5.2.4 Advertising Broadcast User Data Logical Link (ADVB-U)

The LE Advertising Broadcast User Data Logical Link (ADVB-U) is used to carry LE advertisement broadcast user data used between devices without the need for a connection or LE-U between the devices. The user data link is only carried on the default LE Advertising Broadcast logical transport.

3.5.5.3 AMP Logical Links

3.5.5.3.1 AMP Control Logical Link (AMP-C)

Each PAL functions differently with respect to a control logical link. When one exists, this logical link is referred to as the AMP-C logical link.

3.5.5.3.2 AMP User Asynchronous/Isochronous Logical Link (AMP-U)

The AMP user asynchronous/isochronous logical link (AMP-U) is used to carry all asynchronous and isochronous framed user data over an AMP. Unlike the ACL-U logical link, the AMP-U supports multiple logical link handles per physical link and each logical link handle may have different quality of service capabilities. The details of how the AMP-U logical link is mapped to the underlying MAC/PHY is described by the PAL.

3.6 L2CAP CHANNELS

L2CAP provides a multiplexing role allowing many different applications to share an ACL-U or AMP-U logical link. Applications and service protocols interface with L2CAP using a channel-oriented interface to create connections to equivalent entities on other devices.

L2CAP channel endpoints are identified to their clients by a Channel Identifier (CID). This is assigned by L2CAP, and each L2CAP channel endpoint on any device has a different CID.

L2CAP channels may be configured to provide an appropriate Quality of Service (QoS) to the application. L2CAP maps the channel onto the ACL-U logical link, LE-U, or one of the AMP-U logical links.

L2CAP supports channels that are connection-oriented and others that are group-oriented. Group-oriented channels may be mapped onto the ASB-U logical link, or implemented as iterated transmission to each member in turn over an ACL-U logical link.

Apart from the creation, configuration and dismantling of channels, the main role of L2CAP is to multiplex service data units (SDUs) from the channel clients onto the ACL-U, LE-U, or AMP-U logical links, and to carry out a simple level of scheduling, selecting SDUs according to relative priority.

L2CAP can provide per channel flow control with the peer L2CAP layer. This option is selected by the application when the channel is established. L2CAP can also provide enhanced error detection and retransmission to (a) reduce the probability of undetected errors being passed to the application and (b) recover from loss of portions of the user data when the Baseband layer performs a flush on the ACL-U logical link.

In the case where an HCI is present, the L2CAP is also required to segment L2CAP SDUs into fragments that will fit into the baseband buffers, and also to operate a token based flow control procedure over the HCI, submitting fragments to the baseband only when allowed to do so. This may affect the scheduling algorithm.

4 COMMUNICATION TOPOLOGY AND OPERATION

4.1 PICONET TOPOLOGY

4.1.1 BR/EDR Topology

Any time a link is created using the BR/EDR Controller it is within the context of a piconet. A piconet consists of two or more devices that occupy the same BR/EDR physical channel.

Connected BR/EDR devices communicate on the same physical channel by synchronizing with a common clock and hopping sequence. The common (piconet) clock is identical to the Bluetooth clock of one of the devices in the piconet, known as the master of the piconet, and the hopping sequence is derived from the master's clock and the master's Bluetooth device address. All other synchronized devices are referred to as slaves in the piconet.

The terms master and slave are only used when describing these roles in a piconet.

A number of independent piconets may exist in close proximity. Each piconet has a different physical channel (that is a different master device and an independent timing and hopping sequence.)

A Bluetooth device may participate concurrently in two or more piconets. It does this on a time-division multiplexing basis. A Bluetooth device can never be a master of more than one piconet. (Since in BR/EDR the piconet is defined by synchronization to the master's Bluetooth clock it is impossible to be the master of two or more piconets.) A Bluetooth device may be a slave in many independent piconets.

A Bluetooth device that is a member of two or more piconets is said to be involved in a scatternet. Involvement in a scatternet does not necessarily imply any network routing capability or function in the Bluetooth device. The Bluetooth core protocols do not, and are not intended to offer such functionality, which is the responsibility of higher level protocols and is outside the scope of the Bluetooth core specification.

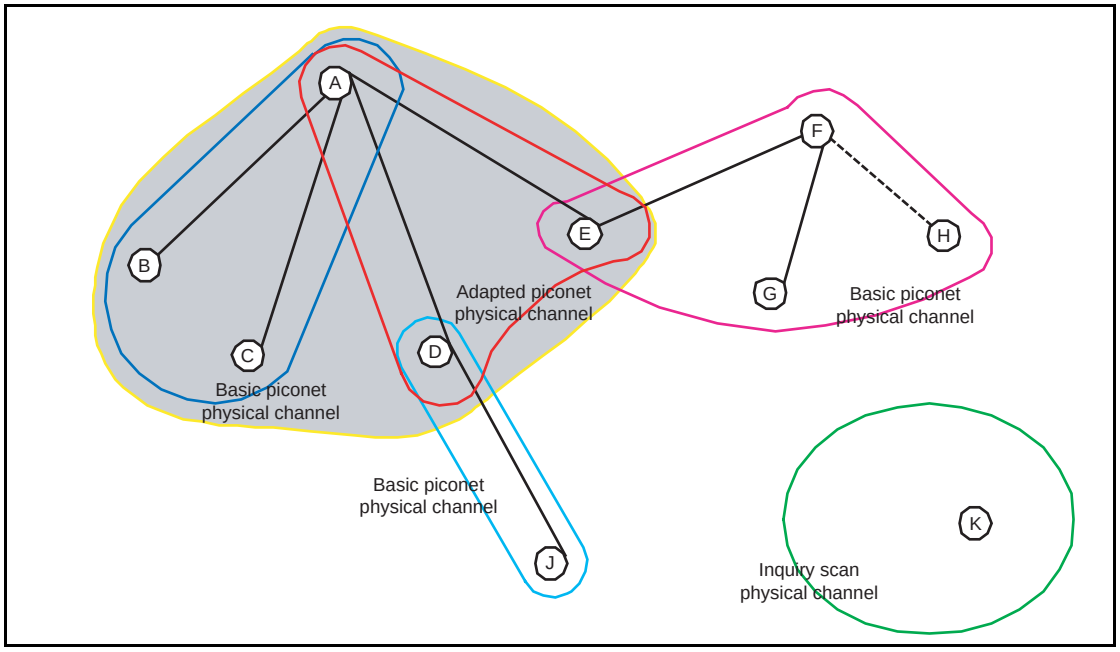


Figure 4.1: Example Bluetooth BR/EDR topology

In Figure 4.1 on page 74 an example topology is shown that demonstrates a number of the architectural features described below. Device A is a master in a piconet (represented by the shaded area, and known as piconet A) with devices B, C, D and E as slaves. Two other piconets are shown: a) one piconet with device F as master (known as piconet F) and devices E, G and H as slaves and b) one piconet with device D as master (known as piconet D) and device J as slave.

In piconet A there are two physical channels. Devices B and C are using the basic piconet physical channel (represented by the blue enclosure) as they do not support adaptive frequency hopping. Devices D and E are capable of supporting adaptive frequency hopping, and are using the adapted piconet physical channel (represented by the red enclosure.) Device A is capable of adaptive frequency hopping, and operates in a TDM basis on both physical channels according to which slave is being addressed.

Piconet D and piconet F are both using only a basic piconet physical channel (represented by the cyan and magenta enclosures respectively.) In the case of piconet D this is because device J does not support the adaptive hopping mode. Although device D supports adaptive hopping it cannot use it in this piconet. In piconet F device F does not support adaptive hopping, and therefore it cannot be used in this piconet.

Device K is shown in the same locality as the other devices. It is not currently a member of a piconet, but has services that it offers to other Bluetooth devices. It is currently listening on its inquiry scan physical channel (represented by the green enclosure), awaiting an inquiry request from another device.

4.1.2 LE Topology

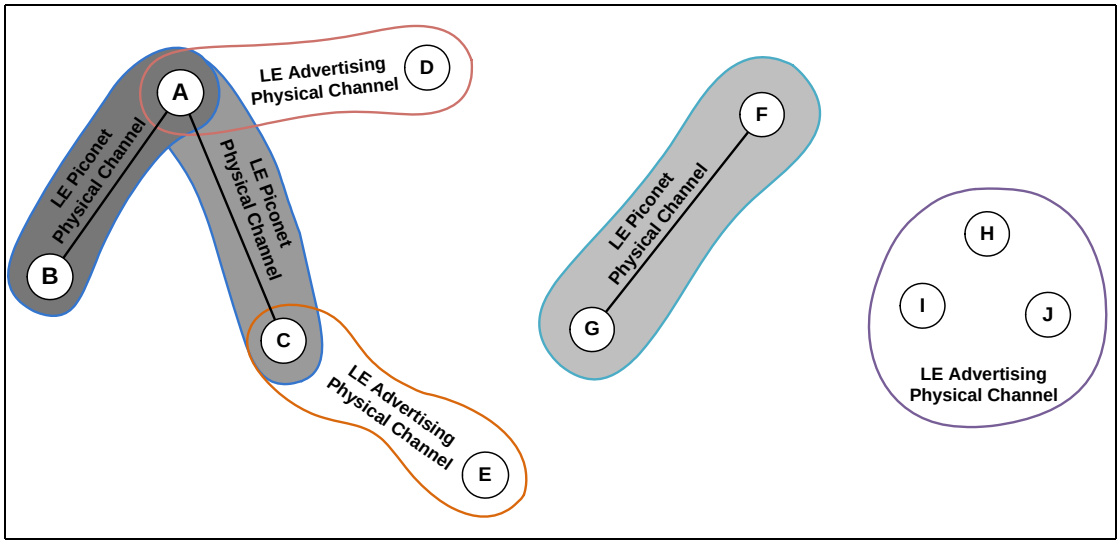


Figure 4.2: Example of Bluetooth LE topology

In Figure 4.2 on page 75 an example topology is shown that demonstrates a number of the LE architectural features described below. Device A is a master in a piconet (represented by the shaded area, and known as piconet A) with devices B and C as slaves, unlike BR/EDR the slaves do not share a common physical channel with the master. Each slave communicates on a separate physical channel with the master. One other piconet is shown with device F as master (known as piconet F) and device G as a slave. There are three other broadcast groups shown: a) device D is an advertiser and device A is also an initiator (known as broadcast group D), b) device E is a scanner and device C is also an advertiser (known as broadcast group C), c) device H is an advertiser and devices I and J are scanners (known as broadcast group H).

Devices A and B are using one LE physical channel (represented by the blue enclosure and a dark gray background). Devices B and C are using another LE piconet physical channel (represented by the blue enclosure and a lighter gray background). In broadcast group D, device D is advertising using a connectable event on the advertising physical channel (represented by the green enclosure) and device A is an initiator. Device A can form a connection with device D and add the device to piconet A. In broadcast group E, device C is also advertising on the advertising physical channel (represented by the orange enclosure) using non-connectable events which are being captured by device E as a scanner. Broadcast group D and broadcast group E may be using different advertising PHY channels or different timings to avoid collisions.

In piconet F there is one physical channel. Devices F and G are using an LE piconet physical channel (represented by the aqua enclosure). Device F is the master and device G is the slave.

In broadcast group H there is one physical channel. Devices H, I and J are using an LE advertising physical channel (represented by the purple enclosure). Devices H and I are advertisers and device J is a scanner.

4.2 OPERATIONAL PROCEDURES AND MODES

The typical operational mode of a Bluetooth device is to be connected to other Bluetooth devices (in a piconet) and exchanging data with those Bluetooth devices. As Bluetooth is an ad-hoc wireless communications technology, there are a number of operational procedures that enable piconets to be formed so that the subsequent communications can take place. Procedures and modes are applied at different layers in the architecture and therefore a device may be engaged in a number of these procedures and modes concurrently.

4.2.1 BR/EDR Procedures

4.2.1.1 Inquiry (Discovering) Procedure

Bluetooth devices use the inquiry procedure to discover nearby devices, or to be discovered by devices in their locality.

The inquiry procedure is asymmetrical. A Bluetooth device that tries to find other nearby devices is known as an inquiring device and actively sends inquiry requests. Bluetooth devices that are available to be found are known as discoverable devices and listen for these inquiry requests and send responses. The inquiry procedure uses a special physical channel for the inquiry requests and responses.

Both inquiring and discoverable devices may already be connected to other Bluetooth devices in a piconet. Any time spent inquiring or occupying the inquiry scan physical channel needs to be balanced with the demands of the QoS commitments on existing logical transports.

The inquiry procedure does not make use of any of the architectural layers above the physical channel, although a transient physical link may be considered to be present during the exchange of inquiry and inquiry response information.

4.2.1.1.1 Extended Inquiry Response

An Extended Inquiry Response can be used to provide miscellaneous information during the inquiry response procedure. Data types are defined for such things as local name and supported services, information that otherwise would have to be obtained by establishing a connection. A device that receives a local name and a list of supported services in an extended inquiry response does not have to connect to do a remote name request and an SDP service search, thereby shortening the time to useful information. It is recommended that a device includes all supported services and a significant portion of its local

name, if that name is too long to be sent in its entirety, in the extended inquiry response.

The extended inquiry response procedure is backwards compatible with the standard inquiry response procedure.

4.2.1.2 Paging (Connecting) Procedure

The procedure for forming connections is asymmetrical and requires that one Bluetooth device carries out the page (connection) procedure while the other Bluetooth device is connectable (page scanning.) The procedure is targeted, so that the page procedure is only responded to by one specified Bluetooth device.

The connectable device uses a special physical channel to listen for connection request packets from the paging (connecting) device. This physical channel has attributes that are specific to the connectable device, hence only a paging device with knowledge of the connectable device is able to communicate on this channel.

Both paging and connectable devices may already be connected to other Bluetooth devices. Any time spent paging or occupying the page scan physical channel needs to be balanced with the demands of the QoS commitments on existing logical transports.

4.2.1.3 Connected Mode

After a successful connection procedure over the BR/EDR Controller, there is a piconet physical channel to which both devices are connected, there is a physical link between the devices, and there are default ACL-C and ACL-U logical links. When in the connected mode it is possible to create and release additional logical links, and to change the modes of the physical and logical links while remaining connected to the piconet physical channel. It is also possible for the device to carry out inquiry, paging or scanning procedures or to be connected to other piconets without needing to disconnect from the original piconet physical channel.

Additional logical links are created using the Link Manager that exchanges Link Manager Protocol messages with the remote Bluetooth device to negotiate the creation and settings for these links. One of these links (ACL-C) transports the LMP control protocol and is invisible to the layers above the Link Manager. The other link (ACL-U) transports the L2CAP signaling protocol and any multiplexed L2CAP best-effort channels. It is common to refer to a default ACL logical transport, which can be resolved by context, but typically refers to the default ACL-U logical link. Also note that these two logical links share a logical transport.

AMP physical links may be established in connected mode. Once an AMP physical link is created, one or more AMP-U logical links may be established to transport L2CAP user data.

During the time that a slave device is actively connected to a piconet there is always a default ACL logical transport between the slave and the master device. There are two methods of deleting the default ACL logical transport. The first method is to detach the device from the piconet physical channel, at which time the entire hierarchy of L2CAP channels, logical links, and logical transports between the devices is deleted.

The second method is to place the physical link to the slave device in the park state, at which time it gives up its default ACL logical transport. This is only allowed if all other logical transports have been deleted (except for the ASB logical transport that cannot be explicitly created or deleted.) It is not allowed to park a device while it has any logical transports other than the default ACL and ASB logical transports.

When the slave device physical link is parked, its default ACL logical transport is released and the ASB logical transport is replaced with a PSB logical transport. The ACL-C and ACL-U logical links that are multiplexed onto the default ACL logical transport remain in existence but cannot be used for the transport of data. The Link Manager on the master device restricts itself to the use of LMP messages that are allowed to be transported over the PSB-C logical link. The Channel Manager and L2CAP Resource Manager ensure that no L2CAP unicast data traffic is submitted to the controller while the device is parked. The Channel Manager may decide to manage the parking and unparking of the device as necessary to allow data to be transported.

4.2.1.4 Hold Mode

Hold mode is not a general device mode, but applies to unreserved slots on the physical link. When in this mode, the physical link is only active during slots that are reserved for the operation of the synchronous link types SCO and eSCO. All asynchronous links are inactive. Hold modes operate once for each invocation and are then exited when complete, returning to the previous mode.

4.2.1.5 Sniff Mode

Sniff mode is not a general device mode, but applies to the default ACL logical transports. When in this mode the availability of these logical transports is modified by defining a duty cycle consisting of periods of presence and absence. Devices that have their default ACL logical transports in sniff mode may use the absent periods to engage in activity on another physical channel, or to enter reduced power mode. Sniff mode only affects the default ACL logical transports (i.e. their shared ACL logical transport), and does not apply to any additional SCO or eSCO logical transports that may be active. The periods of presence and absence of the physical link on the piconet physical channel is derived as a union of all logical transports that are built on the physical link.

Sniff subrating provides a mechanism for further reducing the active duty cycle, thereby enhancing the power-saving capability of sniff mode. Sniff subrating allows a Host to create a guaranteed access-like connection by specifying maximum transmit and receive latencies. This allows the basebands to optimize the low power performance without having to exit and re-enter sniff mode using Link Manager commands.

Note that broadcast logical transports have no defined expectations for presence or absence. A master device should aim to schedule broadcasts to coincide with periods of physical link presence within the piconet physical channel, but this may not always be possible or practical. Repetition of broadcasts is defined to improve the possibilities for reaching multiple slaves without overlapping presence periods. However, broadcast logical transports cannot be considered to be reliable.

4.2.1.6 Parked State

A slave device may remain connected to a piconet but have its physical link in the parked state. In this state the device cannot support any logical links to the master with the exception of the PSB-C and PSB-U logical links that are used for all communication between the piconet master and the parked slave.

When the physical link to a slave device is parked this means that there are restrictions on when the master and slave may communicate, defined by the PSB logical transport parameters. During times when the PSB logical transport is inactive (or absent) then the devices may engage in activity on other physical channels, or enter reduced power mode.

4.2.1.7 Role switch procedure

The role switch procedure is a method for swapping the roles of two devices connected in a piconet. The procedure involves moving from the physical channel that is defined by the original master device to the physical channel that is defined by the new master device. In the process of swapping from one physical channel to the next, the hierarchy of physical links and logical transports over the BR/EDR Controller are removed and rebuilt, with the exception of the ASB and PSB logical transports that are implied by the topology and are not preserved. Note that the role switch procedure does not affect AMP physical channels. After the role switch, the original piconet physical channel may cease to exist or may be continued if the original master had other slaves that are still connected to the channel.

The procedure only copies the default ACL logical links and supporting layers to the new physical channel. Any additional logical transports are not copied by this procedure, and if required this must be carried out by higher layers. The LT_ADDRs of any affected transports may not be preserved as the values may already be in use on the new physical channel.

If there are any QoS commitments or modes such as sniff mode on the original logical transports, then these are not preserved after a role switch. These must be renegotiated after the role switch has completed.

4.2.1.8 Enhanced Data Rate

Enhanced Data Rate is a method of extending the capacity and types of Bluetooth packets for the purposes of increasing the maximum throughput, providing better support for multiple connections, and lowering power consumption, while the remainder of the architecture is unchanged.

Enhanced Data Rate may be selected as a mode that operates independently on each logical transport. Once enabled, the packet type bits in the packet header are interpreted differently from their meaning in Basic Rate mode. This different interpretation is clarified in conjunction with the logical transport address field in the header. The result of this interpretation allows the packet payload header and payload to be received and demodulated according to the packet type. Enhanced Data Rate can be enabled only for ACL-U, eSCO-S logical transports and cannot be enabled for ACL-C, SCO-S, and the broadcast logical transports.

4.2.2 LE Procedures

4.2.2.1 Device Filtering Procedure

The device filtering procedure is a method for controllers to reduce the number of devices requiring communication responses. Since it is not required to respond to requests from every device, it reduces the number of transmissions an LE Controller is required to make which reduces power consumption. It also reduces the communication the controller would be required to make with the host. This results in additional power savings since the Host does not have to be involved.

An advertising or scanning device may employ device filtering to restrict the devices in which it receives advertising packets, scan requests or connection requests. In LE, some advertising packets received by a scanning device require that the scanning device send a request to the advertising device. This advertisement can be ignored if device filtering is used and the advertising device is being filtered. A similar situation occurs with connection requests. Connection requests must be responded to by initiators unless a device filter is used to limit the devices the initiator is required to respond. Advertisers can also use device filters to limit the devices in which it will accept a scan request or connection request.

This device filtering is accomplished through the use of a “White List” located in the LL block of the controller. A white list enumerates the remote devices that are allowed to communicate with the local device. When a white list is in effect, transmissions from devices that are not in the white list will be ignored by the

LL. Since device filtering occurs in the LL it can have a significant impact on power consumption by filtering (or ignoring) advertising packets, scan requests or connection requests from being sent to the higher layers for handling.

The use of device filtering during certain procedures needs to be evaluated carefully to ensure devices are not unintentionally ignored, which may cause interoperability problems when attempting to establish connections or receive advertising broadcasts.

4.2.2.2 Advertising Procedure

An advertiser uses the advertising procedure to perform unidirectional broadcasts to devices in the area. The unidirectional broadcast occurs without a connection between the advertising device and the listening devices. The advertising procedure can be used to establish connections with nearby initiating devices or used to provide periodic broadcast of user data to scanning devices listening on the advertising channel. The advertising procedure uses the advertising physical channel for all advertising broadcasts.

Advertising devices can be connected to other LE devices in an LE piconet. However advertisers are limited to using non-connectable advertising events if they are already connected to another LE device. Time spent advertising while connected needs to be balanced with the connection requirements needed to maintain the already established connection(s) (if the device is a slave in the piconet then it needs to maintain its connection with the master and if the device is the master it needs to maintain its connection(s) with the one or more slaves in the piconet).

Advertising devices may receive scan requests from listening devices in order to get additional user data from the advertising device. Scan responses are sent by the advertising device to the device making the scan request over the same advertising physical channel. Whereas the broadcast user data sent as part of the advertising packets is typically dynamic in nature, scan response data is generally static in nature.

An advertising device may receive connection requests from initiator devices on the advertising broadcast physical channel. If the advertising device was using a connectable advertising event and the initiating device is not being filtered by the device filtering procedure, the advertising device ceases advertising and enters the connected mode. The device can begin advertising again after it is in the connected mode but it can only use non-connected advertising events.

4.2.2.3 Scanning Procedure

A scanning device uses the scanning procedure to listen for unidirectional broadcasts of user data from advertising devices using the advertising physical channel. A scanning device can request additional user data from an advertising device by making a scan request over the advertising physical channel.

The advertising device responds to these requests with additional user data sent to the scanning device over the advertising physical channel.

The scanning procedure can be used while connected to other LE devices in an LE piconet. Time spent scanning while connected needs to be balanced with the connection requirements needed to maintain the already established connection with the other LE devices in the piconet.

If the broadcasts are connectable advertising events and the scanning device is in the initiator mode, it can initiate a connection by sending a connection request on the advertising broadcast physical channel to the advertising device. Once the connection request is sent, the scanning device ceases the initiator mode scanning for additional broadcasts and enters the connected mode. The device can use the scanning procedure after it enters the connected mode. For a master device, using the initiator mode and scanning for connectable advertisements is how additional devices can be added to the master's LE piconet.

4.2.2.4 Discovering Procedure

Bluetooth devices use the advertising procedure and scanning procedure to discover nearby devices, or to be discovered by devices in a given area.

The discovery procedure is asymmetrical. A Bluetooth device that tries to find other nearby devices is known as a discovering device and listens for devices advertising scannable advertising events. Bluetooth devices that are available to be found are known as discoverable devices and actively broadcast scannable advertising events over the advertising broadcast physical channel.

Both discovering and discoverable devices may already be connected to other Bluetooth devices in a piconet. Any time spent inquiring or occupying the advertising broadcast physical channel needs to be balanced with the connection requirements needed to maintain the already established connection with the other LE devices in the piconet.

Using device filtering by the scanning device will prevent the scanning device from discovering all the devices in a given area.

4.2.2.5 Connecting Procedure

The procedure for forming connections is asymmetrical and requires that one Bluetooth device carries out the advertising procedure while the other Bluetooth device carries out the scanning procedure. The advertising procedure can be targeted, so that only one device will respond to the advertising. The scanning device can also target an advertising device by first discovering that the advertising device is present in a connectable manner, and in the given area, and then scanning only connectable advertising events from that device using the device filter. After receiving connectable advertising events from the targeted advertising device, it can initiate a connection by sending the connec-

tion request to the targeted advertising device over the advertising broadcast physical channel.

Time spent scanning while connected needs to be balanced with the connection requirements needed to maintain the already established connection with the other LE devices in the piconet.

4.2.2.6 Connected Mode

After a successful connection procedure, the devices are physically connected to each other within a piconet. This means that there is a piconet physical channel to which they are both connected, there is a physical link between the devices, and there are default LE-C and LE-U logical links. When in the connected mode it is possible to change the properties of the physical and logical links while remaining connected to the piconet physical channel. It is also possible for the device to carry out advertising, scanning or discovery procedures without needing to disconnect from the original piconet physical channel.

Additional logical links are created using the Link Manager that exchanges LL Protocol messages with the remote Bluetooth device to negotiate the creation and settings for these links. One of these links (LE-C) transports the LL control protocol and is invisible to the layers above the Link Manager. The other link (LE-U) transports the L2CAP signaling protocol and any multiplexed L2CAP best-effort channels. It is common to refer to a default LE ACL logical transport, which can be resolved by context, but typically refers to the default LE-U logical link. Also note that these two logical links share a logical transport.

During the time that a slave device is actively connected to a piconet there is always a default LE ACL logical transport between the slave and the master device. The method of deleting the default LE ACL logical transport is to detach the device from the piconet physical channel, at which time the entire hierarchy of L2CAP channels, logical links, and logical transports between the devices is deleted.

4.2.3 AMP Procedures

4.2.3.1 AMP Discovery Procedures

The AMP Manager is responsible for discovering local AMP Controller(s) and maintaining that list over time as AMPs may be added or removed dynamically from the system. The local AMP Manager may request a list of AMPs from the remote AMP Manager. After the list of AMPs has been requested by the remote device, when an AMP is added or removed from the system, the local AMP Manager will notify that remote AMP Manager of the change.

Each AMP Is identified with an ID and a type. Once the list of AMPs has been received, the AMP Manager may request information (AMP_Info) for each AMP.

4.2.3.2 Physical Link Creation Procedure

To set up an L2CAP channel with a remote device over an AMP physical link (AMP channel), the AMP Manager first discovers the remote AMP(s), collects the necessary information about the remote AMP(s) to set up a physical link in an optimal way and then initiates the physical link creation. This is done over a dedicated L2CAP channel.

The AMP Manager provides the remote AMP information that it collected about the remote AMP to the local AMP PAL. The local AMP PAL then uses this information to create the AMP physical link.

4.2.3.3 Logical Link Creation Procedure

Once a physical link exists, L2CAP creates an AMP logical link with the desired QoS. An L2CAP channel is then created over the logical link, at which point the channel is ready for data communication.

5 SECURITY OVERVIEW

5.1 BR/EDR SECURE SIMPLE PAIRING

The primary goal of Secure Simple Pairing is to simplify the pairing procedure for the user. Secondary goals are to maintain or improve the security in Bluetooth wireless technology. Since high levels of security and ease-of-use are often at opposite ends of the spectrum in many technologies and products, much care has been taken to maximize security while minimizing complexity from the end user's point of view.

5.1.1 Security Goals

Secure Simple Pairing has two security goals: protection against passive eavesdropping and protection against man-in-the-middle (MITM) attacks (active eavesdropping). It is a goal of Secure Simple Pairing to exceed the maximum security level provided by the use of a 16 alphanumeric PIN with the pairing algorithm used in Bluetooth Core Specification version 2.0 + EDR and earlier versions. Note that many Bluetooth devices compliant with Bluetooth Core Specification 2.0 + EDR and earlier versions use a 4-digit PIN or a fixed PIN of commonly known values significantly limiting the security on the link.

5.1.2 Passive Eavesdropping Protection

A strong link key coupled with a strong encryption algorithm is necessary to give the user protection against passive eavesdropping. The strength of the link key is based on the amount of entropy (or randomness) in its generation process which would not be known by an attacker. Using legacy pairing, the only source of entropy is the PIN which, in many use cases, is typically four digits either selected by the user or fixed for a given product. Therefore, if the pairing procedure and one authentication exchange is recorded one can run an exhaustive search to find the PIN in a very short amount of time on commonly available computing hardware. With Secure Simple Pairing, the recording attack becomes much harder as the attacker must have solved a hard problem in public key cryptography in order to derive the link key from the recorded information. This protection is independent of the length of the passkey or other numeric values that the user must handle. Secure Simple Pairing gives the same resistance against the recording and passive eavesdropping attacks even when the user is not required to do anything.

Secure Simple Pairing uses Elliptic Curve Diffie Hellman (ECDH) public key cryptography as a means to thwart passive eavesdropping attacks. ECDH provides a very high degree of strength against passive eavesdropping attacks but it may be subject to MITM attacks, which however, are much harder to perform in practice than the passive eavesdropping attack (see Section 5.1.3, “Man-In-The-Middle Protection,” on page 86).

Using the security protocols in the Bluetooth Core Specification version 2.0 + EDR and earlier with a 16 numeric digit PIN achieves about 53 bits of entropy whereas a 16 character alphanumeric, case sensitive PIN yields about 95 bits of entropy when the entire 62 character set is used ([0, … 9, 'A', … 'Z', 'a', … 'z']). Secure Simple Pairing has approximately 95 bits of entropy using the FIPS approved P192 elliptic curve which is at least as good as the entropy in Bluetooth Core Specification 2.0 + EDR and earlier using a 16 character, alphanumeric, case sensitive PIN. Secure Simple Pairing, therefore, exceeds the security requirements of the Bluetooth SIM Access Profile (SAP) which is the profile with the most stringent security requirements. ECDH cryptography was selected over standard Diffie Hellman (often referred to as DH76) since it is computationally less complex and less likely to exceed the low computational capacity in common Bluetooth Controllers.

5.1.3 Man-In-The-Middle Protection

A man-in-the-middle (MITM) attack occurs when a user wants to connect two devices but instead of connecting directly with each other they unknowingly connect to a third (attacking) device that plays the role of the device they are attempting to pair with. The third device then relays information between the two devices giving the illusion that they are directly connected. The attacking device may even eavesdrop on communication between the two devices (known as active eavesdropping) and is able to insert and modify information on the connection. In this type of attack, all of the information exchanged between the two devices are compromised and the attacker may inject commands and information into each of the devices thus potentially damaging the function of the devices. Devices falling victim to the attack are capable of communicating only when the attacker is present. If the attacker is not active or out range, the two victim devices will not be able to communicate directly with each other and the user will notice it.

To prevent MITM attacks, Secure Simple Pairing offers two user assisted numeric methods: numerical comparison or passkey entry. If Secure Simple Pairing would use 16 decimal digit numbers, then the usability would be the same as using legacy pairing with 16 decimal digit PIN. The chance for a MITM to succeed inserting its own link keys in this case is a 1 in $10^{16} = 2^{53}$ pairing instances, which is an unnecessarily low probability.

Secure Simple Pairing protects the user from MITM attacks with a goal of offering a 1 in 1,000,000 chance that a MITM could mount a successful attack. The strength of the MITM protections was selected to minimize the user impact by using a six digit number for numerical comparison and Passkey entry. This level of MITM protection was selected since, in most cases, users can be alerted to the potential presence of a MITM attacker when the connection process fails as a result of a failed MITM attack. While most users feel that provided that they have not compromised their passkey, a 4-digit key is sufficient for authentication (i.e. bank card PIN codes), the use of six digits allows Secure

Simple Pairing to be FIPS compliant and this was deemed to have little perceivable usability impact.

5.1.4 Association Models

Secure Simple Pairing uses four association models referred to as Numeric Comparison, Just Works, Out Of Band, and Passkey Entry. Each of these association models are described in more detail in the following sections.

The association model used is deterministic based on the I/O capabilities of the two devices.

5.1.4.1 Numeric Comparison

The Numeric Comparison association model is designed for scenarios where both devices are capable of displaying a six digit number and both are capable of having the user enter "yes" or "no". A good example of this model is the cell phone / PC scenario.

The user is shown a six digit number (from "000000" to "999999") on both displays and then asked whether the numbers are the same on both devices. If "yes" is entered on both devices, the pairing is successful.

The numeric comparison serves two purposes. First, since many devices do not have unique names, it provides confirmation to the user that the correct devices are connected with each other. Second, the numeric comparison provides protection against MITM attacks (see Section 5.1.3 on page 86).

Note that there is a significant difference from a cryptographic point of view between Numeric Comparison and the PIN entry model used by Bluetooth Core Specification and earlier versions. In the Numeric Comparison association model, the six digit number is an artifact of the security algorithm and not an input to it, as is the case in the Bluetooth security model. Knowing the displayed number is of no benefit in decrypting the encoded data exchanged between the two devices.

5.1.4.2 Just Works

The Just Works association model is primarily designed for scenarios where at least one of the devices does not have a display capable of displaying a six digit number nor does it have a keyboard capable of entering six decimal digits. A good example of this model is the cell phone/mono headset scenario where most headsets do not have a display.

The Just Works association model uses the Numeric Comparison protocol but the user is never shown a number and the application may simply ask the user to accept the connection (exact implementation is up to the end product manufacturer).

The Just Works association model provides the same protection as the Numeric Comparison association model against passive eavesdropping but offers no protection against the MITM attack.

When compared against today's experience of a headset with a fixed PIN, the security level of the Just Works association model is considerably higher since a high degree of protection against passive eavesdropping is realized.

5.1.4.3 Out of Band

The Out of Band (OOB) association model is primarily designed for scenarios where an Out of Band mechanism is used to both discover the devices as well as to exchange or transfer cryptographic numbers used in the pairing process. In order to be effective from a security point of view, the Out of Band channel should provide different properties in terms of security compared to the Bluetooth radio channel. The Out of Band channel should be resistant to MITM attacks. If it is not, security may be compromised during authentication.

The user's experience differs a bit depending on the Out of Band mechanism. As an example, with a Near Field Communication (NFC) solution, the user(s) will initially touch the two devices together, and is given the option to pair the first device with the other device. If "yes" is entered, the pairing is successful. This is a single touch experience where the exchanged information is used in both devices. The information exchanged includes discovery information (such as the Bluetooth Device Address) as well as cryptographic information. One of the devices will use a Bluetooth Device Address to establish a connection with the other device. The rest of the exchanged information is used during authentication.

The OOB mechanism may be implemented as either read only or read/write. If one side is read only, a one-way authentication is performed. If both sides are read/write, a two-way authentication is performed.

The OOB protocol is selected only when the pairing process has been activated by previous OOB exchange of information and one (or both) of the device(s) gives OOB as the IO capabilities. The protocol uses the information which has been exchanged and simply asks the user to confirm connection.

The OOB association model supports any OOB mechanism where cryptographic information and the Bluetooth Device Address can be exchanged. The OOB association model does not support a solution where the user has activated a Bluetooth connection and would like to use OOB for authentication only.

5.1.4.4 Passkey Entry

The Passkey Entry association model is primarily designed for the scenario where one device has input capability but does not have the capability to dis-

play six digits and the other device has output capabilities. A good example of this model is the PC and keyboard scenario.

The user is shown a six digit number (from "000000" to "999999") on the device with a display, and is then asked to enter the number on the other device. If the value entered on the second device is correct, the pairing is successful. Note that there is a significant difference from a cryptographic point of view between Passkey Entry and the PIN entry model used by Bluetooth Core Specification 2.0 + EDR and earlier versions. In the Passkey Entry association model, the six digit number is independent of the security algorithm and not an input to it, as is the case in the 2.0 + EDR security model. Knowing the entered number is of no benefit in decrypting the encoded data exchanged between the two devices.

5.1.4.5 Association Model Overview

The following diagram shows Secure Simple Pairing from the point of view of the technology used for discovery and then the different association possibilities.

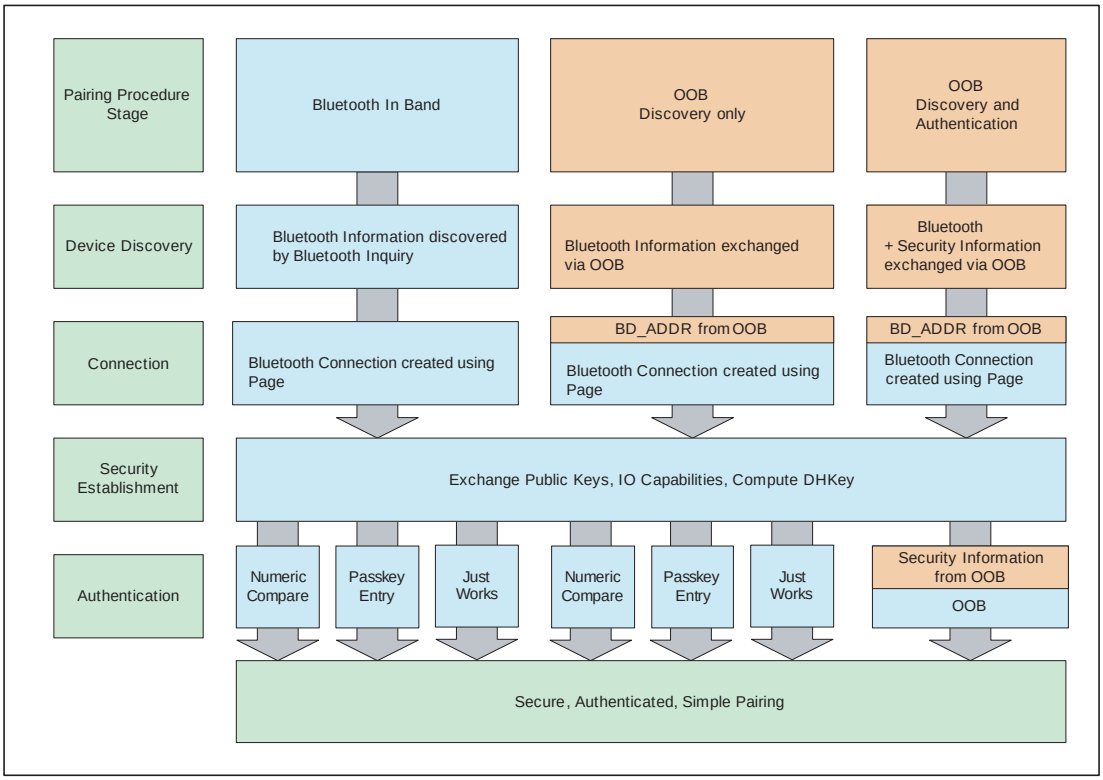


Figure 5.1: Secure Simple Pairing Association Models

5.2 LE SECURITY

Bluetooth LE has some differences in security aspects with respect to BR/EDR security features such as Secure Simple Pairing. The association models are

similar to Secure Simple Pairing from the user perspective and have the same names but have differences in the quality of the protection provided.

The overall goal of keeping the cost of the Controller and the complexity of a slave device to a minimum was used in making compromises on security capabilities in LE.

5.2.1 Association Models

Bluetooth LE uses three association models referred to as Just Works, Out of Band and Passkey Entry. Bluetooth LE does not have an equivalent of Numeric Comparison. Each of these association models is similar to Secure Simple Pairing with the following exceptions.

- Just Works and Passkey Entry do not provide any passive eavesdropping protection. This is because Secure Simple Pairing uses Elliptic Curve Diffie-Hellman and LE does not.

The use of each association model is based on the I/O capabilities of the devices in a similar manner as Secure Simple Pairing.

5.2.2 Key Generation

Key generation in Bluetooth LE is performed by the Host on each LE device independent of any other LE device. Note: Key generation in BR/EDR is performed in the Controller. By performing key generation in the Host, the key generation algorithms can be upgraded without the need to change the Controller.

Bluetooth LE uses multiple keys, each for a specific purpose, as follows:

- Confidentiality of data and device authentication
- Authentication of unencrypted data
- Device Identity

In LE a single link key is generated by combining contributions from each device into a link key used during pairing. In BR/EDR key generation is performed in the Controller.

5.2.3 Encryption

Encryption in Bluetooth LE uses AES-CCM cryptography. Like BR/EDR, in LE encryption is performed in the Controller.

5.2.4 Signed Data

Bluetooth LE supports the ability to send authenticated data over an unencrypted ATT bearer between two devices with a trusted relationship. This is accomplished by signing the data with a Connection Signature Resolving Key

(CSRK). The sending devices places a signature after the Data PDU. The receiving verifies the signature and if the signature is verified the Data PDU is assumed to come from the trusted source. The signature is composed of a Message Authentication Code generated by the signing algorithm and a counter. The counter is used to protect against a replay attack and is incremented on each signed Data PDU sent.

5.2.5 Privacy Feature

Bluetooth LE supports a feature that reduces the ability to track a LE device over a period of time by changing the Bluetooth device address on a frequent basis. The privacy feature is not used in the GAP discovery mode and procedures but it is used when supported during connection mode and connection procedures.

In order for devices using the privacy feature to reconnect to known devices, the device addresses used when the privacy feature is enabled, private address, must be resolvable to the other devices' identity. The private address is generated using the device's identity key exchanged during the bonding procedure. Use of the private address for reconnection is limited to use cases where filtering of devices disabled. Disabling device filtering may increase the power consumption of the device since the host will need to process all device requests. Situations where the device filtering is disabled should only be used for limited periods of time to minimize the power consumption in this mode.

The privacy feature also defines a reconnection address which allows for bonded device to reconnect while also filtering devices to known devices. The reconnect address is exchanged between the two devices at each connection. Since reconnect addresses only change between connections, device filtering can be used to minimize processing of excessive requests.

5.3 AMP SECURITY

AMP security does not change the user experience because it utilizes the same Secure Simple Pairing association models that were introduced in the Bluetooth 2.1 + EDR Core Specification. From the user's point of view, all radios are "paired" in one process.

AMP security starts during the Secure Simple Pairing process. The link key for BR/EDR is generated during Phase 4 of Secure Simple Pairing. A 256-bit Generic AMP Link Key (GAMP_LK) is generated from the BR/EDR Link Key. The Generic AMP Link Key is stored in the security database alongside the BR/EDR Link Key once pairing has completed.

AMP security does not affect the BR/EDR Link Key so backwards compatibility is maintained for devices that support the Generic AMP feature with devices that do not.

When an AMP is used for the first time, a dedicated AMP key is created by the AMP Manager for that AMP type using the new Secure Simple Pairing function h2 and a KeyID for the AMP type. The length of the Dedicated AMP Link Key depends on the AMP Type. If the Pair-wise Master key already exists due to a previous connection the AMP link key is not created and the stored key is reused.

The Dedicated AMP Link Key is sent to the PAL during the Physical Link creation process. Each PAL is responsible for using the Dedicated AMP Link Key during the security phase of the physical link establishment process. Note that a Dedicated AMP link key is used for multiple sessions over the same AMP.

Each time a dedicated AMP key is successfully created, the Generic AMP Link Key is updated. This is performed using the h2 function with KeyID "gamp."

6 BLUETOOTH APPLICATION ARCHITECTURE

6.1 BLUETOOTH PROFILES

Application interoperability in the Bluetooth system is accomplished by Bluetooth profiles. Bluetooth profiles define the required functions and features of each layer in the Bluetooth system from the PHY to L2CAP and any other protocols outside of the Core specification. The profile defines the vertical interactions between the layers as well as the peer-to-peer interactions of specific layers between devices.

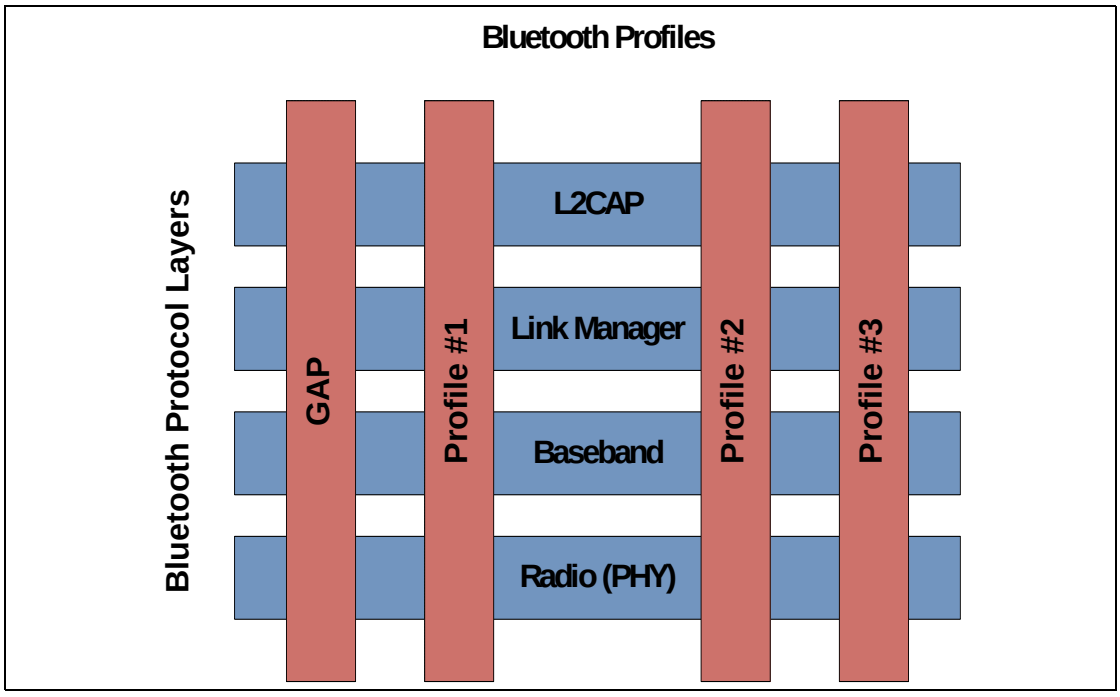


Figure 6.1: Bluetooth Profiles

In addition, application behaviors and data formats are also defined by the profile. When two devices comply with all the requirements of a Bluetooth profile, interpretability of an application is enabled.

All profiles describe service discovery requirements necessary for devices to connect, find available application services and connection information necessary for making application level connections.

6.2 GENERIC ACCESS PROFILE

The Bluetooth system defines a base profile which all Bluetooth devices implement. This profile is the Generic Access Profile (GAP), which defines the basic requirements of a Bluetooth device. For instance, for BR/EDR, it defines a Bluetooth device to include the Radio, Baseband, Link Manager, L2CAP, and the service discovery protocol functionality; for LE, it defines the Physical

Layer, Link Layer, L2CAP, Security Manager, Attribute Protocol and Generic Attribute Profile. This ties all the various layers together to form the basic requirements for a Bluetooth device. It also describes the behaviors and methods for device discovery, connection establishment, security, authentication, association models and service discovery.

In BR/EDR, GAP defines a single role with functionality that may be present in each device. This functionality includes how devices discovery each other, establish connections and describes security association models used for authentication. In BR/EDR this functionality may be present in both devices. It may be necessary for a device to implement both the initiating and accepting functionality if the device wants to discover or establish connections with all devices. A device may only include either the initiating or the accepting functionality but it requires the remote device to support the complimentary functionality to discovery or establish connections with the device. For BR/EDR, the Controller is required to support all the functionality, however the Host may limit this functionality based on the other profiles or use cases supported by the device.

In LE, GAP defines four specific roles: Broadcaster, Observer, Peripheral, and Central. A device may support multiple LE GAP roles provided that the underlying Controller supports those roles or role combinations. However, only one LE GAP role may be supported at a given time. Each role specifies the requirements for the underlying Controller. This allows for Controllers to be optimized for specific use cases.

The **Broadcaster** role is optimized for transmitter only applications. Devices supporting the broadcaster role use advertising to broadcast data. The broadcaster role does not support connections. The **Observer** role is optimized for receiver only applications. Devices supporting the observer role are the complementary device for a broadcaster and receives broadcast data contained in advertisements. The observer role does not support connections. The **Peripheral** role is optimized for devices that support a single connection and are less complex than central devices. Devices supporting the peripheral role only require Controllers that support the Controller's slave role. The **Central** role supports multiple connections and is the initiator for all connections with devices in the peripheral role. Devices supporting the central role require a Controller that support the Controller's master role and generally supports more complex functions compared to the other LE GAP roles.

6.3 PROFILE HIERARCHY

Since all Bluetooth devices are required to implement GAP, any additional profiles implemented by a Bluetooth device become supersets of GAP. Depending on the complexity of an application or the ability to reuse common requirements of functionality of the Bluetooth system between many applications, additional generic profiles can be created that are both a superset of GAP as well as being a superset of another profile. A top level profile that describes application interoperability is called an Application Profile.

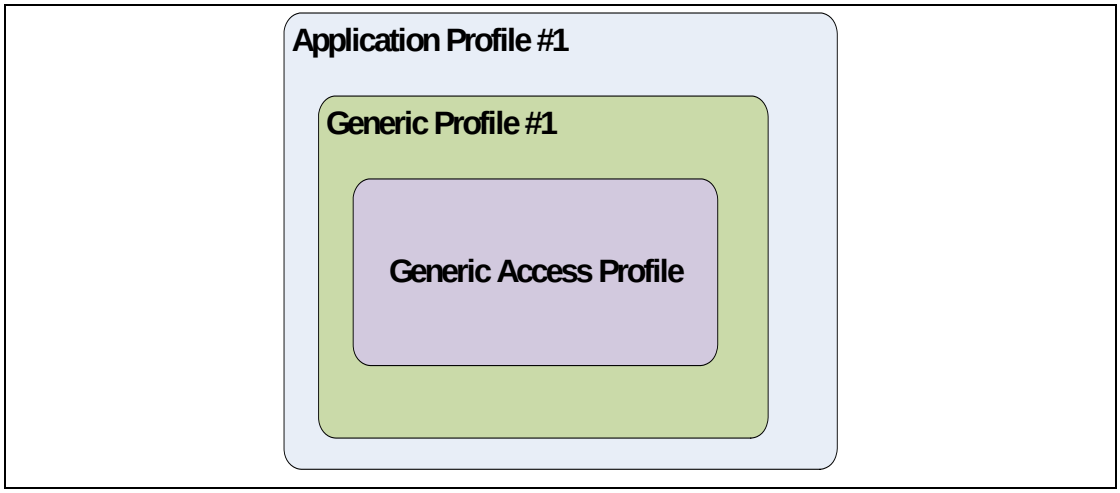


Figure 6.2: Profile Hierarchy

Application profiles contain by reference GAP and any other generic profile that describes a set of common requirements of the Bluetooth system.

6.4 GENERIC ATTRIBUTE PROFILE

Generic Attribute Profile (GATT) is built on top of the Attribute Protocol (ATT) and establishes common operations and a framework for the data transported and stored by the Attribute Protocol. GATT defines two roles: Server and Client. The GATT roles are not necessarily tied to specific GAP roles and but may be specified by higher layer profiles. GATT and ATT are not transport specific and can be used in both BR/EDR and LE. However, GATT and ATT are mandatory to implement in LE since it is used for discovering services

The GATT server stores the data transported over the Attribute Protocol and accepts Attribute Protocol requests, commands and confirmations from the GATT client. The GATT server sends responses to requests and when configured, sends indication and notifications asynchronously to the GATT client when specified events occur on the GATT server.

GATT also specifies the format of data contained on the GATT server. Attributes, as transported by the Attribute Protocol, are formatted as **Services** and **Characteristics**. Services may contain a collection of characteristics. Characteristics contain a single value and any number of descriptors describing the characteristic value.

With the defined structure of services, characteristics and characteristic descriptors a GATT client that is not specific to a profile can still traverse the GATT server and display characteristic values to the user. The characteristic descriptors can be used to display descriptions of the characteristic values that may make the value understandable by the user.

6.5 GATT-BASED PROFILE HIERARCHY

The GATT Profile specifies the structure in which profile data is exchanged. This structure defines basic elements such as services and characteristics, used in a profile.

The top level of the hierarchy is a profile. A profile is composed of one or more services necessary to fulfill a use case. A service is composed of characteristics or references to other services. Each characteristic contains a value and may contain optional information about the value. The service and characteristic and the components of the characteristic (i.e., value and descriptors) contain the profile data and are all stored in Attributes on the server.

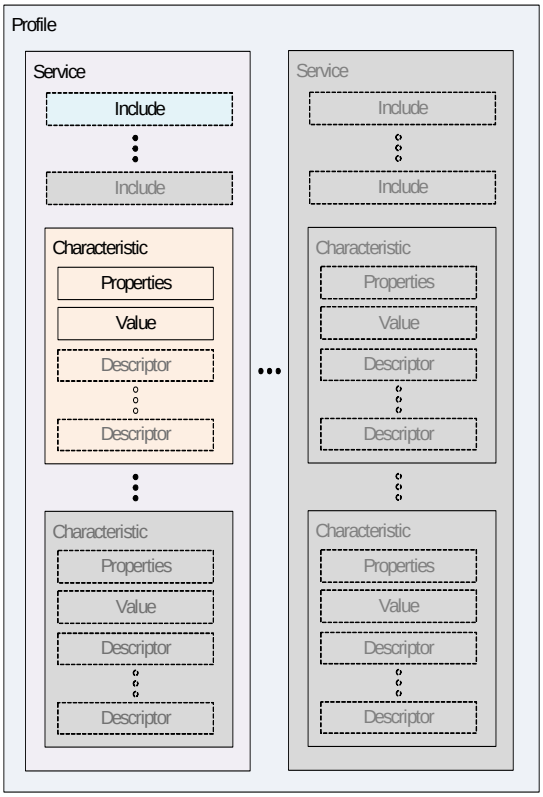


Figure 6.3: GATT-Based Profile Hierarchy

6.5.1 Service

A service is a collection of data and associated behaviors to accomplish a particular function or feature of a device or portions of a device. A service may reference other primary or secondary services and/or a set of characteristics that make up the service.

There are two types of services: primary and secondary. A primary service is a service that provides the primary functionality of a device. A secondary service is a service that provides auxiliary functionality of a device and is referenced from at least one primary service on the device.

To maintain backward compatibility with earlier clients, later revisions of a service definition can only add new referenced services or optional characteristics. Later revisions of a service definition are also forbidden from changing behaviors from previous revision of the service definition.

Services may be used in one or more profiles to fulfill a particular use case.

6.5.2 Referenced Services

A referenced service is a method to incorporate another service definition on the server as part of the service referencing it. When a service references another service, the entire referenced service becomes part of the new service including any nested referenced services and characteristics. The referenced service still exists as an independent service. There are no limits to the depth of nested references.

6.5.3 Characteristic

A characteristic is a value used in a service along with properties and configuration information about how the value is accessed and information about how the value is displayed or represented. A characteristic definition contains a characteristic declaration, characteristic properties, and a value. It may also contain descriptors that describe the value or permit configuration of the server with respect to the characteristic value.

ACRONYMS & ABBREVIATIONS

CONTENTS

1 List of Acronyms and Abbreviations 101

1 LIST OF ACRONYMS AND ABBREVIATIONS

Acronym or abbreviation	Writing out in full	Which means
8DPSK	8 phase Differential Phase Shift Keying	3 Mbps modulation type used by Enhanced Data rate
π/4 DQPSK	pi/4 Rotated Differential Quaternary Phase Shift Keying	2 Mbps modulation type used by Enhanced Data Rate
A		
A2MP	AMP Manager Protocol	
AAD	Additional Authentical Data	
AC	Access Category	
ACI	Access Category Index	
ACK	Acknowledge/ACKnowledgement	
ACL	Asynchronous Connection-oriented [logical transport]	Reliable or time-bounded, bi-directional, point-to-point.
ACL-C	ACL Control [logical link] (LMP)	
ACL-U	ACL User [logical link] (L2CAP)	
ACO	Authenticated Ciphering Offset	
AD	Advertising Data	
ADVB	LE Advertising Broadcast	
ADVB-C	LE Advertising Broadcast Control (Logical Link)	
ADVB-U	LE Advertising Broadcast User Data (Logical Link)	
Adv_idx	Advertising channel index	
AES	Advanced Encryption Standard	
AGC	Automatic Gain Control	
AFH	Adaptive Frequency Hopping	
AGC	Automatic Gain Control	
AHS	Adapted Hop Sequence	
AIFSN	Arbitration Interframe Space Number	
AMP	Alternate MAC PHY	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
AMP_ASSOC	Alternate MAC PHY association information	
AMP-U	AMP User Asynchronous/Isochronous Logical Link	
AR_ADDR	Access Request Address	
ARQ	Automatic Repeat Request	
ASB	Active Slave Broadcast [logical transport]	Unreliable, uni-directional broadcast to any devices synchronized with the physical channel.
ASB-U	ASB User [logical link] (L2CAP)	
ATT	Attribute Protocol	
B		
BB	Baseband	
BCH	Bose, Chaudhuri & Hocquenghem	Type of code The persons who discovered these codes in 1959 (H) and 1960 (B&C)
BD_ADDR	Bluetooth Device Address	
BER	Bit Error Rate	
BT	Bandwidth Time	
C		
CAC	Channel Access Code	
CL	Connectionless	
CCM	Counter with Cipher Block Chaining-Message Authentication Code	
CCMP	CTR with CBC-MAC Protocol	
CKLN	Native Clock	
CLK	Master Clock	
CLKE	Estimated Clock	
CODEC	COder DECoder	
COF	Ciphering Offset	
CRC	Cyclic Redundancy Check	
CTS	Clear to send	
CVSD	Continuous Variable Slope Delta Modulation	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
D		
DA	Destination address	
DAC	Device Access Code	
DCE	Data Communication Equipment	
DCE	Data Circuit-Terminating Equipment	In serial communications, DCE refers to a device between the communication endpoints whose sole task is to facilitate the communications process; typically a modem
DCI	Default Check Initialization	
DEVM	Differential Error Vector Magnitude	Measure of modulation error used for Enhanced Data Rate transmitter testing
DFS	Dynamic Frequency Selection	
DH	Data-High Rate	Data packet type for high rate data
DHK	Diversifier Hiding Key	
DIAC	Dedicated Inquiry Access Code	
DIV	Diversifier	
DM	Data - Medium Rate	Data packet type for medium rate data
DPSK	Differential Phase Shift Keying	Generic description of Enhanced Data Rate modulation
DQPSK	Differential Quaternary Phase Shift Keying	Modulation type used by Enhanced Data Rate
DSAP	Destination Service Access Point	
DTE	Data Terminal Equipment	In serial communications, DTE refers to a device at the endpoint of the communications path; typically a computer or terminal.
DTM	Direct Test Mode	
DUT	Device Under Test	
DV	Data Voice	Data packet type for data and voice
E		
ECWmax	Enhanced Contention Window maximum	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
ECWmim	Enhanced Contention Window minimum	
ED	Erroneous Data	
EDCA	Enhanced Distributed Channel Access	
EDIV	Encrypted Diversifier	
EDR	Enhanced Data Rate	
EFS	Extended Flow Specification	
EIR	Extended Inquiry Response	Host supplied information transmitted in the inquiry responses substate.
EIRP	Effective Isotropic Radiated Power	Equivalent power that an isotropic antenna must transmit to provide the same field power density
EOC	Extreme Operating Conditions	
ER	Encryption Root	
EPR	Encryption Pause Resume	Mode allowing the device to initiate the pause and resume encryption sequence
ERP	Extended Rate PHY conforming to Clause 19	
eSCO	Extended Synchronous Connection Oriented [logical transport]	Bi-directional, symmetric or asymmetric, point-to-point, general regular data, limited retransmission.
eSCO-S	Stream eSCO (unframed)	used to support isochronous data delivered in a stream without framing
ESS	Extended Service Set	
ETSI	European Telecommunications Standards Institute	
F		
FCC	Federal Communications Commission	
FCS	Frame Check Sequence	
FDMA	Frequency Division Multiple Access	
FEC	Forward Error Correction code	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
FH	Frequency Hopping	
FHS	Frequency Hop Synchronization	
FHSS	Frequency Hopping Spread Spectrum	
FIFO	First In First Out	
FIPS	Federal Information Processing Standards	
FM	Frequency Modulation	Modulation Type
G		
GAP	Generic Access Profile	
GATT	Generic Attribute Protocol	
GFSK	Gaussian Frequency Shift Keying	
GIAC	General Inquiry Access Code	
GTM	Generic Test Methodology	
H		
HCI	Host Controller Interface	
HEC	Header-Error-Check	
HID	Human Interface Device	
HV	High quality Voice	e.g. HV1 packet
HW	Hardware	
I		
IAC	Inquiry Access Code	
IC	Industry Canada	
IEC	International Electrotechnical Commission	
IEEE	Institute of Electronic and Electrical Engineers	
IETF	Internet Engineering Task Force	
IFS	Inter Frame Space	
IP	Internet Protocol	
IPv4	Internet Protocol version 4	
IPv6	Internet Protocol version 6	
IR	Identity Root	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
IrDA	Infra-red Data Association	
IRK	Identity Root Key	
ISM	Industrial, Scientific, Medical	
ISO	International Organization for Standardization	
IUT	Implementation Under Test	
ITU	International Telecommunication Union	
IV	Initialization Vector	
IVm	Initialization Vector (master)	
IVs	Initialization Vector (slave)	
J		
JRL	Japanese Radio Law	
K		
KCC	Korea Communications Commission	
L		
L2CAP	Logical Link Control and Adaptation Protocol	
LAP	Lower Address Part	
LC	Link Controller	Link Controller (or baseband) part of the Bluetooth protocol stack. Low level Baseband protocol handler
LC	Link Control [logical link]	The control logical links LC and ACL-C are used at the link control level and link manager level, respectively.
LCP	Link Control Protocol	
LE	Low Energy	
LE-C	Low Energy Control (link)	
LE-U	LE User [logical link]	
LFSR	Linear Feedback Shift Register	
LL	Link Layer	
LLCP	Link Layer Control Protocol	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
LLID	Logical Link Identifier	
LM	Link Manager	
LMP	Link Manager Protocol	For LM peer to peer communication
LR	Loudness Rating	
LSB	Least Significant Bit	
LSO	Least Significant Octet	
LSTO	Link Supervision Timeout Event	Controller can send LSTO event to Host
LT_ADDR	Logical Transport ADDRess	
LTK	Long-Term Key	
M		
M	Master or Mandatory	
MAC	Medium Access Control	
MAC	Message Authentication Code	Usually meaning the Medium Access Control code
MD	More Data	
Mbps	Million (Mega) bits per second	
MIC	Message Integrity Check	
MIIT	Ministry of Industry and Information Technology	
MITM	Man-in-the-middle	
MMI	Man Machine Interface	
MS	Mobile Station	
MSC	Message Sequence Chart	
MSB	Most Significant Bit	
MSC	Message Sequence Chart	
MSO	Most Significant Octet	
MTU	Maximum Transmission Unit	
N		
NAK	Negative Acknowledge	
NAP	Non-significant Address Part	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
NCC	National Communications Commission	
NESN	Next Expected Sequence Number	
NIST	National Institute of Standards and Technology	
NOC	Normal Operating Conditions	
O		
O	Optional	
OBEX	OBject EXchange protocol	
OCF	OpCode Command Field	
OGF	OpCode Group Field	
OOB	Out of Band	
P		
PBF	Packet Boundary Flag	The device supports the capability to correctly handle HCI ACL Data Packets
PCM	Pulse Coded Modulation	
PDU	Protocol Data Unit	a message
PHY	Physical Layer	
PIN	Personal Identification Number	
PM_ADDR	Parked Member Address	
PN	Pseudo-random Noise	
PPM	Part Per Million	
PPP	Point-to-Point Protocol	
PRBS	Pseudo Random Bit Sequence	
PRNG	Pseudo Random Noise Generation	
PSB	Parked Slave Broadcast [logical transport]	Unreliable, uni-directional broadcast to all piconet devices.
PSB-C	PSB Control [logical link] (LMP)	
PSB-U	PSB User [logical link] (L2CAP)	
PSK	Phase Shift Keying	Class of modulation types
PSTN	Public Switched Telephone Network	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
ptt	Packet Type Table	The ptt parameter is used to select the logical transport types via LMP.
Q		
QoS	Quality of Service	
R		
RAND	Random number	
RF	Radio Frequency	
RFC	Request For Comments	
RFCMode	Retransmission and Flow Control Mode	
RFCOMM		Serial cable emulation protocol based on ETSI TS 07.10
RFU	Reserved for Future Use	
RMS	Root Mean Square	
RSSI	Received Signal Strength Indication	
RX	Receive	
S		
S	Slave	
SAP	Service Access Points	
SAR	Segmentation and Reassembly	
SCA	Sleep Clock Accuracy	
SCO	Synchronous Connection-Oriented [logical transport]	Bi-directional, symmetric, point-to-point, AV channels.
SCO-S	Stream SCO (unframed)	
SCO-S	Synchronous logical link	used to support isochronous data delivered in a stream without framing
SD	Service Discovery	
SDP	Service Discovery Protocol	
SDU	Service Data Unit	
SEQN	Sequential Numbering scheme	
SK	Session Key	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
SKDm	Session Key Diversifier (master)	Master portion of the Session Key Diversifier
SKDs	Session Key Diversifier (slave)	Slave portion of the Session Key Diversifier
SLR	Send Loudness Ring	
SM	Security Manager	
SMP	Security Manager Protocol	
SN	Sequence Number	
SRES	Signed Response	
SRK	Signature Resolving Key	
SS	Supplementary Services	
SSI	Signal Strength Indication	
SSP	Secure Simple Pairing	Simplifies the pairing process and improves Bluetooth security.
SSR	Sniff Subrating	A mode that defines the anchor points at which the master transmits to the slave.
STK	Short Term Key	
SW	Software	
T		
TC	Test Control	Test Control layer for the test interface
TCI	Test Control Interface	
TCP/IP	Transport Control Protocol/Internet Protocol	
TCS	Telephony Control protocol Specification	
TDD	Time-Division Duplex	
TDMA	Time Division Multiple Access	
T_IFS	Time Inter Frame Space	Time interval between consecutive packets on same channel index
TK	Temporary Key	
TX	Transmit	
U		
UAP	Upper Address Part	

Table 1.1: Acronyms and Abbreviations

Acronym or abbreviation	Writing out in full	Which means
UART	Universal Asynchronous receiver Transmitter	
UI	User Interface	
UI	Unnumbered Information	
ULAP	Upper and Lower Address Parts	
USB	Universal Serial Bus	
UTF-8	8-bit UCS/Unicode Transformation Format	
UUID	Universal Unique Identifier	
W		
WAP	Wireless Application Protocol	

Table 1.1: Acronyms and Abbreviations

CORE SPECIFICATION CHANGE HISTORY

Ford Ex. 1052

CONTENTS

11 Deprecated Features 116

2 Changes from V1.1 to V1.2 117

2.1 New Features 117
2.2 Structure Changes 117
2.3 Deprecated Features list 117
2.4 Changes in Wording 118
2.5 Nomenclature Changes 118

3 Changes from V1.2 to V2.0 + EDR 119

3.1 New Features 119
3.2 Deprecated Features 119

4 Changes from V2.0 + EDR to V2.1 + EDR 120

4.1 New features 120
4.2 Deprecated Features 120

5 Changes From V2.1 + EDR To V3.0 + HS 121

5.1 New Features 121
5.2 Deprecated Features 121

6 Changes From V3.0 + HS To v4.0 122

6.1 New Features 122
6.2 Deprecated Features 122

1 DEPRECATED FEATURES

Some features have been deemed no longer useful and have been deprecated. The term *deprecation* does not mean that they are no longer allowed, but that they are no longer recommended as the best way of performing a given function.

Features that have been deprecated are not included in this specification or corresponding test documentation. Deprecated features may be implemented using the latest qualifiable version of the Bluetooth core specification that specifies the deprecated features.

2 CHANGES FROM V1.1 TO V1.2

2.1 NEW FEATURES

Several new features were introduced in the Bluetooth Core Specification 1.2. The major areas of improvement are:

- Architectural overview
- Faster connection
- Adaptive frequency hopping
- Extended SCO links
- Enhanced error detection and flow control
- Enhanced synchronization capability
- Enhanced flow specification

The feature descriptions are incorporated into the existing text in different core parts, described in Volumes 2 and 3.

2.2 STRUCTURE CHANGES

The Bluetooth Core Specification 1.2 was significantly restructured for better consistency and readability. The most important structure changes have been performed in Baseband, LMP, HCI and L2CAP. The text in these sections has been rearranged to provide:

- Presentation of the information in a more logical progression
- Removal of redundant text and requirements
- Consolidation of baseband related requirements (for example, the *Baseband Timers* and *Bluetooth Audio* sections into the Baseband Specification)
- Alignment of the specification with the new architecture and terminology presented in the Architecture Overview (see “IEEE Language” on page 131 [vol. 1]).

In addition, new text and requirements were added for the new features as well as many changes throughout the specification to update the text to use IEEE language (see “IEEE Language” on page 131 [vol. 1]).

2.3 DEPRECATED FEATURES LIST

Features deprecated in version 1.2 are:

- The use of Unit Keys for security
- Optional Paging schemes
- 23 channel hopping sequence

- Page scan period mode and associated commands

2.4 CHANGES IN WORDING

Two general classes of changes to the wording of the Bluetooth Specification have been done for version 1.2. They are a formalization of the language by using conventions established by the Institute of Electrical and Electronic Engineers (IEEE), and a regularization of Bluetooth wireless technology-specific terms. Many portions of the version 1.1 specification use imprecise or inaccurate terms to describe attributes of the protocol. A more accurate terminology described in Part E was introduced into the version 1.2 specification and have been applied in future versions.

2.5 NOMENCLATURE CHANGES

The nomenclature in Bluetooth 1.2 was also updated due to new concepts that are introduced together with the new features and the new architecture. See “Architecture” on page 11.

3 CHANGES FROM V1.2 TO V2.0 + EDR

3.1 NEW FEATURES

The Bluetooth Core Specification version 2.0 + EDR introduces Enhanced Data Rate (EDR). EDR provides a set of additional packet types that use the new 2 Mbps and 3 Mbps modes.

In addition to EDR a set of errata provided in ESR02 has been incorporated into this version and revised to include changes caused by the addition of EDR.

These additions are incorporated into the existing text in different core parts described in Volumes 2 and 3.

3.2 DEPRECATED FEATURES

The only feature deprecated in version 2.0 + EDR is the Page Scan Period Mode and associated commands (based on Erratum 694 which is also included in ESR02).

4 CHANGES FROM V2.0 + EDR TO V2.1 + EDR

4.1 NEW FEATURES

Several new features are introduced in Bluetooth Core Specification 2.1 + EDR. The major areas of improvement are:

- Erroneous Data Reporting
- Encryption Pause and Resume
- Extended Inquiry Response
- Link Supervision Timeout Changed Event
- Non-Automatically-Flushable Packet Boundary Flag
- Secure Simple Pairing
- Sniff Subrating
- Security Mode 4

4.2 DEPRECATED FEATURES

No features were deprecated in v2.1 + EDR.

5 CHANGES FROM V2.1 + EDR TO V3.0 + HS

5.1 NEW FEATURES

Several new features are introduced in Bluetooth Core Specification v3.0 + HS. The major areas of improvement are:

- AMP Manager Protocol (A2MP)
- Enhancements to L2CAP including
  - •Enhanced Retransmission Mode and Streaming Mode
  - •Improvements to the L2CAP state machine for AMP channels
  - •Fixed channel support
- Enhancements to HCI for AMP
- Enhancements to Security for AMP
- 802.11 Protocol Adaptation Layer
- Enhanced Power Control
- Unicast Connectionless Data
- HCI Read Encryption Key Size command
- Generic Test Methodology for AMP
- Enhanced USB and SDIO HCI Transports

5.2 DEPRECATED FEATURES

No features were deprecated in v3.0 + HS.

6 CHANGES FROM V3.0 + HS TO V4.0

6.1 NEW FEATURES

Several new features are introduced in Bluetooth Core Specification v4.0 Release. The major areas of improvement are:

- Bluetooth Low Energy including
  - Low Energy Physical Layer
  - Low Energy Link Layer
  - Enhancements to HCI for Low Energy
  - Low Energy Direct Test Mode
  - AES Encryption
  - Enhancements to L2CAP for Low Energy
  - Enhancements to GAP for Low Energy
  - Attribute Protocol (ATT)
  - Generic Attribute Profile (GATT)
  - Security Manager (SM)

6.2 DEPRECATED FEATURES

No features were deprecated in v4.0.

MIXING OF SPECIFICATION VERSIONS

Ford Ex. 1052

CONTENTS

1 Mixing of Specification Versions 126

1.1 Features and their Types 127

1.2 Core Specification Addendums 128

1 MIXING OF SPECIFICATION VERSIONS

This part describes how volumes, and parts within volumes, of different versions of the Core Specification may be mixed in Bluetooth implementations. The Core System consists of a BR/EDR Controller Package (see Volume 2), a Low Energy Controller Package (see Volume 6), a Host Package (see Volume 3) and AMP Protocol Adaptation Layers (see Volume 5).

A Core Specification Addendum contains one or more parts of a single volume or one or more parts in multiple volumes. Addendums may be used to supersede a part in a volume or may be used to add a part to a volume according to the rules in Section 1.2.

- All parts within a Primary Controller implementation shall be the same version of Volume 2 and Volume 6, and shall support at least one new Type 1, -2, or -3 feature from Table 1.2.

  All parts within a Host implementation of Volume 3 shall be the same version. An AMP Controller implementation shall contain parts of Volume 2 and Volume 5 from the same version and shall support at least one new Type 3 or Type 4 feature from Table 1.2.
- The Primary Controller, AMP Controller, and Host may be different versions within a single implementation.

In order to describe how these volumes and parts within volumes can be mixed, one needs to distinguish between four categories of features specified in the different specification versions. The four categories are:

Category	Description
Type 1	Feature that exists below HCI and cannot be configured/enabled via HCI
Type 2	Feature that exists below HCI and can be configured/enabled via HCI
Type 3	Feature that exists below and above HCI and requires HCI command/events to function
Type 4	Feature that exists only above HCI

Table 1.1: Feature type definitions

The outcome of mixing different core system packages are derived from the feature definitions in the table above:

- If an implementation contains features of type 1 or type 4, these features can function with any combination of Host Package and Controller Package or AMP Protocol Adaptation Layer (PAL) versions.
- If an implementation contains features of type 2, these features can only be used under a default condition if a Host Package of an older version is mixed with a Controller Package or AMP PAL of this version.

- In order to fully use the feature under all conditions, the Host Package, Controller Package, and AMP PAL must be of the same or later version.
- If an implementation contains features of type 3, these features can only function if the Host Package supports this version or a later version and if the Controller Package supports this version or a later version.

See the Bluetooth Brand Book for specification naming requirements.

1.1 FEATURES AND THEIR TYPES

The following table lists the features and their types.

Feature	Version	Type
Basic AFH operation	1.2	1
Enhanced inquiry	1.2	1
Configuration of AFH (setting channels and enabling/disabling channel assessment)	1.2	2
Enhanced synchronization capability	1.2	2
Interlaced inquiry scan	1.2	2
Interlaced page scan	1.2	2
Broadcast encryption	1.2	2
Enhanced flow specification and flush time-out	1.2	3
Extended SCO links	1.2	3
Inquiry Result with RSSI	1.2	3
L2CAP flow and error control	1.2	4
2 Mbps EDR	2.0 + EDR	2
3 Mbps EDR	2.0 + EDR	2
3 slot packets in EDR	2.0 + EDR	2
5 slot packets in EDR	2.0 + EDR	2
2 Mbps eSCO	2.0 + EDR	2[1]
3 Mbps eSCO	2.0 + EDR	2*
3 slot packets for EDR eSCO	2.0 + EDR	2*
Erroneous Data Reporting	2.1 + EDR	3
Extended Inquiry Response	2.1 + EDR	3
Encryption Pause and Resume	2.1 + EDR	1
Link Supervision Timeout Changed Event	2.1 + EDR	3

Table 1.2: Features and their types

Feature	Version	Type
Non-Flushable Packet Boundary Flag	2.1 + EDR	3
Sniff subrating	2.1+ EDR	3
Secure Simple Pairing	2.1.+ EDR	3
L2CAP Enhanced Retransmission Mode	Addendum 1/ 3.0 + HS	4
L2CAP Streaming Mode	Addendum 1/ 3.0 + HS	4
Enhanced Power Control	3.0 + HS	2
AMP Manager Protocol (A2MP)	3.0 + HS	4
L2CAP Enhancements for AMP	3.0 + HS	4
802.11 PAL	3.0 + HS	3
Generic Test Methodology	3.0 + HS	3
Unicast Connectionless Data	3.0 + HS	4
Low Energy (up through L2CAP)	4.0	3
Attribute Protocol	4.0	4
Generic Attribute Profile	4.0	4
Security Manager	4.0	3

Table 1.2: Features and their types

1. The EDR eSCO options are marked as 2* because eSCO requires profile support, but if a product includes the eSCO option from V1.2, then EDR eSCO will be supported without any new support above HCI.

1.2 CORE SPECIFICATION ADDENDUMS

The following table contains a list of Core Specification Addendums and Core Specification versions they are allowed to be used with.

Each part within an addendum is identified by a type indicating whether it may replace a part in an allowed core specification version or whether it may add to a package within a core specification version.

Type	Description
Replacement	The part in the addendum is used instead of the equivalent part in the allowed specification versions.
Addition	The Part in the Addendum is used in addition to the existing parts in the allowed specification versions.

The following table lists adopted addendums.

Addendum	Volume and Part	Type	Allowed Versions
1	Volume 3, Part A	Replacement	1.2, 2.0 + EDR, 2.1 + EDR

IEEE LANGUAGE

Ford Ex. 1052

CONTENTS

1 Use of IEEE Language 135

1.1 Shall 135
1.2 Must 136
1.3 Will 136
1.4 Should 136
1.5 May 136
1.6 Can 137

1 USE OF IEEE LANGUAGE

One of the purposes of this terminology is to make it easy for the reader to identify text that describes requirements as opposed to background information. The general term for text that describes attributes that are required for proper implementation of Bluetooth wireless technology is normative. The general term for language that provides background and context for normative text is informative. These terms are used in various sections to clarify implementation requirements.

Many portions of the Bluetooth Specification use imprecise or inaccurate terms to describe attributes of the protocol. This subsection describes the correct usage of key terms that indicate degree of requirements for processes and data structures. The information here was derived from the Institute of Electrical and Electronics Engineers (IEEE) Style Guide, see http://standards.ieee.org/guides/style/.

The following list is a summary of the terms to be discussed in more detail below:

shall	is required to – used to define requirements
must	is a natural consequence of -- used only to describe unavoidable situations
will	it is true that -- only used in statements of fact
should	is recommended that – used to indicate that among several possibilities one is recommended as particularly suitable, but not required
may	is permitted to – used to allow options
can	is able to – used to relate statements in a causal fashion
is	is defined as – used to further explain elements that are previously required or allowed
note	<informational text ONLY>

For clarity of the definition of those terms, the following sections document why and how they are used. For these sections only, the IEEE terms are italicized to indicate their use as a noun. Uses and examples of the use of the terms in this section are underlined.

1.1 SHALL

The word *shall* is used to indicate mandatory requirements that shall be followed in order to conform to the specification and from which no deviation is permitted.

There is a strong implication that the presence of the word *shall* indicates a testable requirement. All testable requirements shall be reflected in the Protocol Implementation Conformance Statement (PICS). In turn, all PICS indicators should be reflected in the Test Cases (TCs) either directly or indirectly.

A direct reference is a specific test for the attribute cited in the text. For example, a minimum value for a given parameter may be an entry in the TCs. Indirect test coverage may be appropriate if the existence of the attribute is requisite for passing a higher level test.

1.2 MUST

The word *must* shall not be used when stating mandatory requirements. *Must* is used only to describe unavoidable situations and is seldom appropriate for the text of a Specification.

An example of an appropriate use of the term *must* is: “the Bluetooth radios must be in range of each other to communicate”.

1.3 WILL

The use of the word *will* shall not be used when stating mandatory requirements. The term *will* is only used in statements of fact. As with the term *must*, *will* is not generally applicable to the description of a protocol. An example of appropriate use of will is: “when power is removed from the radio, it can be assumed that communications will fail”

1.4 SHOULD

Should equals *is recommended that*. The word *should* is used to indicate that among several possibilities one is recommended as particularly suitable without mentioning or excluding others. Alternatively it may indicate that a certain course of action is preferred but not necessarily required. Finally, in the negative form, it indicates a certain course of action is deprecated but not prohibited.

In the Bluetooth Specification the term designates an optional attribute that may require an entry in the PICS.

Explicit specification of alternatives should be done when using *should*.

1.5 MAY

The word *may* is used to indicate a course of action permissible within the limits of the specification. The term *may* equals *is permitted*. This is generally used when there is a single, optional attribute described, but multiple alternatives may be cited.

The use of *may* implies an optional condition in the PICS and therefore may need to be reflected in the corresponding test cases.

1.6 CAN

The word *can* is used for statements of possibility and capability, whether material, physical, or causal. The term *can* equals *is able to*.

The term *can* shall be used only in informative text. It describes capabilities by virtue of the rules established by normative text.



Specification Volume 2

SPECIFICATION OF THE *BLUETOOTH* SYSTEM

Experience More



Core System Package

[BR/EDR Controller volume]

Covered Core Package version:
4.0

Bluetooth®

Ford Ex. 1052

Revision History

The Revision History is shown in the [Vol 0] Part C, “Appendix” on page 55.

Contributors

The persons who contributed to this specification are listed in the [Vol 0] Part C, “Appendix” on page 55.

Web Site

This specification can also be found on the official Bluetooth web site:
http://www.bluetooth.com

Disclaimer and Copyright Notice

The copyright in these specifications is owned by the Promoter Members of Bluetooth SIG, Inc. (“Bluetooth SIG”). Use of these specifications and any related intellectual property (collectively, the “Specification”), is governed by the Promoters Membership Agreement among the Promoter Members and Bluetooth SIG (the “Promoters Agreement”), certain membership agreements between Bluetooth SIG and its Adopter and Associate Members (the “Membership Agreements”) and the Bluetooth Specification Early Adopters Agreements (“1.2 Early Adopters Agreements”) among Early Adopter members of the unincorporated Bluetooth special interest group and the Promoter Members (the “Early Adopters Agreement”). Certain rights and obligations of the Promoter Members under the Early Adopters Agreements have been assigned to Bluetooth SIG by the Promoter Members.

Use of the Specification by anyone who is not a member of Bluetooth SIG or a party to an Early Adopters Agreement (each such person or party, a “Member”), is prohibited. The legal rights and obligations of each Member are governed by their applicable Membership Agreement, Early Adopters Agreement or Promoters Agreement. No license, express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

Any use of the Specification not in compliance with the terms of the applicable Membership Agreement, Early Adopters Agreement or Promoters Agreement is prohibited and any such prohibited use may result in termination of the applicable Membership Agreement or Early Adopters Agreement and other liability permitted by the applicable agreement or by applicable law to Bluetooth SIG or any of its members for patent, copyright and/or trademark infringement.

THE SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, SATISFACTORY QUALITY, OR REASONABLE SKILL OR CARE, OR ANY WARRANTY ARISING OUT OF ANY COURSE OF DEALING, USAGE, TRADE PRACTICE, PROPOSAL, SPECIFICATION OR SAMPLE.

Each Member hereby acknowledges that products equipped with the Bluetooth® technology (“Bluetooth® Products”) may be subject to various regulatory controls under the laws and regulations of various governments worldwide. Such laws and regulatory controls may govern, among other things, the combination, operation, use, implementation and distribution of Bluetooth® Products. Examples of such laws and regulatory controls include, but are not limited to, airline regulatory controls, telecommunications regulations, technology transfer controls and health and safety regulations. Each Member is solely responsible for the compliance by their Bluetooth® Products with any such laws and regulations and for obtaining any and all required authorizations, permits, or licenses for their Bluetooth® Products related to such regulations within the applicable jurisdictions. Each Member acknowledges that nothing in the Specification provides any information or assistance in connection with securing such compliance, authorizations or licenses. NOTHING IN THE SPECIFICATION CREATES ANY WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING SUCH LAWS OR REGULATIONS.

ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHTS OR FOR NONCOMPLIANCE WITH LAWS, RELATING TO USE OF THE SPECIFICATION IS EXPRESSLY DISCLAIMED. BY USE OF THE SPECIFICATION, EACH MEMBER EXPRESSLY WAIVES ANY CLAIM AGAINST BLUETOOTH SIG AND ITS PROMOTER MEMBERS RELATED TO USE OF THE SPECIFICATION.

Bluetooth SIG reserves the right to adopt any changes or alterations to the Specification as it deems necessary or appropriate.

Copyright © 1999 - 2010

Ericsson AB,
Lenovo,
Intel Corporation,
Microsoft Corporation,
Motorola, Inc.,
Nokia Corporation,
Toshiba Corporation

*Third-party brands and names are the property of their respective owners.

CONTENTS

Part A
RADIO SPECIFICATION

1 Scope 33
2 Frequency Bands and Channel Arrangement 35
3 Transmitter Characteristics 36
3.1 Basic Rate 37
3.1.1 Modulation Characteristics 37
3.1.2 Spurious Emissions 38
3.1.3 Radio Frequency Tolerance 39
3.2 Enhanced Data Rate 39
3.2.1 Modulation Characteristics 40
3.2.2 Spurious Emissions 42
3.2.3 Radio Frequency Tolerance 44
3.2.4 Relative Transmit Power 44
4 Receiver Characteristics 46
4.1 Basic Rate 46
4.1.1 Actual Sensitivity Level 46
4.1.2 Interference Performance 46
4.1.3 Out-of-Band Blocking 47
4.1.4 Intermodulation Characteristics 47
4.1.5 Maximum Usable Level 48
4.1.6 Receiver Signal Strength Indicator 48
4.1.7 Reference Signal Definition 48
4.2 Enhanced Data Rate 48
4.2.1 Actual Sensitivity Level 48
4.2.2 BER Floor Performance 48
4.2.3 Interference Performance 48
4.2.4 Maximum Usable Level 49
4.2.5 Out-of-Band and Intermodulation Characteristics 50
4.2.6 Reference Signal Definition 50
5 Appendix A 51
5.1 Nominal Test Conditions 51
5.1.1 Nominal temperature 51
5.1.2 Nominal power source 51
5.2 Extreme Test Conditions 52

5.2.1 Extreme temperatures 52
5.2.2 Extreme power source voltages 52
6 Appendix B 53
7 Appendix C 54
7.1 Enhanced Data Rate Modulation Accuracy 54

Part B
BASEBAND SPECIFICATION

1 General Description 65
1.1 Bluetooth Clock 66
1.2 Bluetooth Device Addressing 68
1.2.1 Reserved Addresses 68
1.3 Access Codes 69
2 Physical Channels 70
2.1 Physical Channel Definition 71
2.2 Basic Piconet Physical Channel 71
2.2.1 Master-slave Definition 71
2.2.2 Hopping Characteristics 72
2.2.3 Time Slots 72
2.2.4 Piconet Clocks 73
2.2.5 Transmit/receive Timing 73
2.3 Adapted Piconet Physical Channel 76
2.3.1 Hopping Characteristics 76
2.4 Page Scan Physical Channel 77
2.4.1 Clock Estimate for Paging 77
2.4.2 Hopping Characteristics 77
2.4.3 Paging Procedure Timing 78
2.4.4 Page Response Timing 79
2.5 Inquiry Scan Physical Channel 81
2.5.1 Clock for Inquiry 81
2.5.2 Hopping Characteristics 81
2.5.3 Inquiry Procedure Timing 81
2.5.4 Inquiry Response Timing 81
2.6 Hop Selection 83
2.6.1 General Selection Scheme 83
2.6.2 Selection Kernel 87
2.6.3 Adapted Hop Selection Kernel 90
2.6.4 Control Word 91
3 Physical Links 96

3.1 Link Supervision 96

4 Logical Transports 97

4.1 General 97
4.2 Logical Transport Address (LT_ADDR) 97
4.3 Synchronous Logical Transports 98
4.4 Asynchronous Logical Transport 98
4.5 Transmit/Receive Routines 99
4.5.1 TX Routine 99
4.5.2 RX Routine 102
4.5.3 Flow Control 103
4.6 Active Slave Broadcast Transport 104
4.7 Parked Slave Broadcast Transport 105
4.7.1 Parked Member Address (PM_ADDR) 105
4.7.2 Access Request Address (AR_ADDR) 105

5 Logical Links 106

5.1 Link Control Logical Link (LC) 106
5.2 ACL Control Logical Link (ACL-C) 106
5.3 User Asynchronous/Isochronous Logical Link (ACL-U) 106
5.3.1 Pausing the ACL-U logical link 107
5.4 User Synchronous Data Logical Link (SCO-S) 107
5.5 User Extended Synchronous Data Logical Link (eSCO-S) 107
5.6 Logical Link Priorities 107

6 Packets 108

6.1 General Format 108
6.1.1 Basic Rate 108
6.1.2 Enhanced Data Rate 108
6.2 Bit Ordering 109
6.3 Access Code 110
6.3.1 Access Code Types 110
6.3.2 Preamble 111
6.3.3 Sync Word 111
6.3.4 Trailer 114
6.4 Packet Header 115
6.4.1 LT_ADDR 115
6.4.2 TYPE 115
6.4.3 FLOW 116
6.4.4 ARQN 116
6.4.5 SEQN 116
6.4.6 HEC 116

6.5 Packet Types 117
6.5.1 Common Packet Types 118
6.5.2 SCO Packets 122
6.5.3 eSCO Packets 123
6.5.4 ACL Packets 125
6.6 Payload Format 127
6.6.1 Synchronous Data Field 127
6.6.2 Asynchronous Data Field 129
6.7 Packet Summary 132
7 Bitstream Processing 134
7.1 Error Checking 135
7.1.1 HEC Generation 135
7.1.2 CRC Generation 136
7.2 Data Whitening 138
7.3 Error Correction 139
7.4 FEC Code: Rate 1/3 139
7.5 FEC Code: Rate 2/3 140
7.6 ARQ Scheme 141
7.6.1 Unnumbered ARQ 141
7.6.2 Retransmit Filtering 144
7.6.3 Flushing Payloads 147
7.6.4 Multi-slave Considerations 147
7.6.5 Broadcast Packets 148
7.7 Erroneous Synchronous Data Reporting 149
8 Link Controller Operation 150
8.1 Overview of States 150
8.2 Standby State 151
8.3 Connection Establishment Substates 151
8.3.1 Page Scan Substate 151
8.3.2 Page substate 153
8.3.3 Page Response Substates 156
8.4 Device Discovery Substates 160
8.4.1 Inquiry scan substate 161
8.4.2 Inquiry Substate 162
8.4.3 Inquiry Response Substate 163
8.5 Connection State 165
8.6 Active Mode 166
8.6.1 Polling in the Active Mode 167
8.6.2 SCO 167

8.6.3 eSCO 169
8.6.4 Broadcast Scheme 171
8.6.5 Role Switch 172
8.6.6 Scatternet 174
8.6.7 Hop Sequence Switching 175
8.6.8 Channel Classification and Channel Map Selection 178
8.6.9 Power Management 179
8.7 Sniff Mode 180
8.7.1 Sniff Transition Mode 181
8.7.2 Sniff Subrating 182
8.8 Hold Mode 183
8.9 Park State 184
8.9.1 Beacon Train 184
8.9.2 Beacon Access Window 187
8.9.3 Parked Slave Synchronization 188
8.9.4 Parking 189
8.9.5 Master-initiated Unparking 190
8.9.6 Slave-initiated Unparking 190
8.9.7 Broadcast Scan Window 191
8.9.8 Polling in the Park State 191

9 Audio 192
9.1 LOG PCM CODEC 192
9.2 CVSD CODEC 192
9.3 Error Handling 195
9.4 General Audio Requirements 195
9.4.1 Signal Levels 195
9.4.2 CVSD Audio Quality 195

10 List of Figures 196

11 List of Tables 199
Appendix A: General Audio Recommendations 200
Appendix B: Timers 203
Appendix C: Recommendations for AFH Operation in Park, Hold, and Sniff 205

Part C
LINK MANAGER PROTOCOL SPECIFICATION

1 Introduction 211

2 General Rules 212
2.1 Message Transport 212

2.2 Synchronization 212
2.3 Packet Format 213
2.4 Transactions 214
2.4.1 LMP Response Timeout 215
2.5 Error Handling 215
2.5.1 Transaction Collision Resolution 216
2.6 Procedure Rules 216
2.7 General Response Messages 217
2.8 LMP Message Constraints 217
3 Device Features 218
3.1 General Description 218
3.2 Feature Definitions 218
3.3 Feature Mask Definition 225
3.4 Link Manager Interoperability policy 227
4 Procedure Rules 228
4.1 Connection Control 228
4.1.1 Connection Establishment 228
4.1.2 Detach 229
4.1.3 Power Control 230
4.1.4 Adaptive Frequency Hopping 234
4.1.5 Channel Classification 237
4.1.6 Link Supervision 239
4.1.7 Channel Quality Driven Data Rate Change (CQDDR) 240
4.1.8 Quality of Service (QoS) 241
4.1.9 Paging Scheme Parameters 243
4.1.10 Control of Multi-slot Packets 244
4.1.11 Enhanced Data Rate 244
4.1.12 Encapsulated LMP PDUs 246
4.2 Security 247
4.2.1 Authentication 247
4.2.2 Pairing 249
4.2.3 Change Link Key 252
4.2.4 Change Current Link Key Type 253
4.2.5 Encryption 255
4.2.6 Request Supported Encryption Key Size 262
4.2.7 Secure Simple Pairing 262
4.3 Informational Requests 275
4.3.1 Timing Accuracy 275
4.3.2 Clock Offset 277

4.3.3 LMP version 277
4.3.4 Supported Features....278
4.3.5 Name Request 280
4.4 Role Switch 281
4.4.1 Slot Offset....281
4.4.2 Role Switch 282
4.5 Modes of Operation 284
4.5.1 Hold Mode 284
4.5.2 Park State....286
4.5.3 Sniff Mode 293
4.6 Logical Transports....297
4.6.1 SCO Logical Transport....297
4.6.2 eSCO Logical Transport....300
4.7 Test Mode 305
4.7.1 Activation and Deactivation of Test Mode 305
4.7.2 Control of Test Mode 306
4.7.3 Summary of Test Mode PDUs....306

5 Summary....310
5.1 PDU Summary 310
5.2 Parameter Definitions 320
5.3 LMP Encapsulated....331
5.4 Default Values....332

6 List of Figures....333

7 List of Tables 337

Part D
ERROR CODES

1 Overview of Error Codes 343
1.1 Usage Descriptions....343
1.2 HCI Command Errors....343
1.3 List of Error Codes 344

2 Error Code Descriptions....347
2.1 Unknown HCI Command (0X01)....347
2.2 Unknown Connection Identifier (0X02) 347
2.3 Hardware Failure (0X03)....347
2.4 Page Timeout (0X04) 347
2.5 Authentication Failure (0X05)....347
2.6 PIN or key Missing (0X06) 347
2.7 Memory Capacity Exceeded (0X07) 347
2.8 Connection Timeout (0X08) 348

2.9 Connection Limit Exceeded (0X09) 348
2.10 Synchronous Connection Limit to a Device Exceeded (0X0A) 348
2.11 ACL Connection Already Exists (0X0B) 348
2.12 Command Disallowed (0X0C) 348
2.13 Connection Rejected due to Limited Resources (0X0D) 348
2.14 Connection Rejected due to Security Reasons (0X0E) 348
2.15 Connection Rejected due to Unacceptable BD_ADDR (0X0F) 349
2.16 Connection Accept Timeout Exceeded (0X10) 349
2.17 Unsupported Feature or Parameter Value (0X11) 349
2.18 Invalid HCI Command Parameters (0X12) 349
2.19 Remote User Terminated Connection (0X13) 349
2.20 Remote Device Terminated Connection due to Low Resources (0X14) 350
2.21 Remote Device Terminated Connection due to Power Off (0X15) 350
2.22 Connection Terminated by Local Host (0X16) 350
2.23 Repeated Attempts (0X17) 350
2.24 Pairing not Allowed (0X18) 350
2.25 Unknown LMP PDU (0X19) 350
2.26 Unsupported Remote Feature / Unsupported LMP Feature (0X1A) 350
2.27 SCO Offset Rejected (0X1B) 350
2.28 SCO Interval Rejected (0X1C) 351
2.29 SCO Air Mode Rejected (0X1D) 351
2.30 Invalid LMP Parameters (0X1E) 351
2.31 Unspecified Error (0X1F) 351
2.32 Unsupported LMP Parameter Value (0X20) 351
2.33 Role Change Not Allowed (0X21) 351
2.34 LMP Response Timeout / LL Response Timeout (0X22) 351
2.35 LMP Error Transaction Collision (0X23) 352
2.36 LMP PDU Not Allowed (0X24) 352
2.37 Encryption Mode Not Acceptable (0X25) 352
2.38 Link Key Can Notcannot be Changed (0X26) 352
2.39 Requested QoS Not Supported (0X27) 352
2.40 Instant Passed (0X28) 352
2.41 Pairing with Unit Key Not Supported (0X29) 352
2.42 Different Transaction Collision (0x2A) 352
2.43 QoS Unacceptable Parameter (0X2C) 352
2.44 QoS Rejected (0X2D) 353
2.45 Channel Assessment Not Supported (0X2E) 353
2.46 Insufficient Security (0X2F) 353
2.47 Parameter out of Mandatory Range (0X30) 353
2.48 Role Switch Pending (0X32) 353

2.49 Reserved Slot Violation (0X34) 353
2.50 Role Switch Failed (0X35) 353
2.51 Extended Inquiry Response Too Large (0x36) 354
2.52 Simple Pairing Not Supported By Host (0X37) 354
2.53 Host Busy–Pairing(0X38) 354
2.54 Connection Rejected Due To No Suitable Channel Found (0X39) 354
2.55 Controller Busy (0X3A) 354
2.56 Unacceptable Connection Interval (0X3B) 354
2.57 Directed Advertising Timeout (0X3C) 354
2.58 Connection Terminated Due To MIC Failure (0X3D) 355
2.59 Connection Failed To Be Established (0X3E) 355
2.60 MAC Connection Failed (0x3F) 355

Part E
HOST CONTROLLER INTERFACE FUNCTIONAL SPECIFICATION

1 Introduction 369
1.1 Lower Layers of the Bluetooth Software Stack 370

2 Overview of Host Controller Transport Layer 372
2.1 Host Controller Transport Layer and AMPS 372

3 Overview of Commands and Events 373
3.1 Generic Events 374
3.2 Device Setup 374
3.3 Controller Flow Control 375
3.4 Controller Information 376
3.5 Controller Configuration 377
3.6 Device Discovery 380
3.7 Connection Setup 382
3.8 Remote Information 387
3.9 Synchronous Connections 388
3.10 Connection State 389
3.11 Piconet Structure 392
3.12 Quality of Service 393
3.13 Physical Links 395
3.14 Host Flow Control 397
3.15 Link Information 398
3.16 Authentication and Encryption 400
3.17 Testing 405
3.18 Alphabetical List of Commands and Events 408
3.19 LE Controller Requirements 415

4 HCI Flow Control 418

4.1 Host to Controller Data Flow Control 418
4.1.1 Packet-based Data Flow Control 418
4.1.2 Data-Block-Based Data Flow Control 420
4.2 Controller to Host Data Flow Control 421
4.3 Disconnection Behavior 421
4.4 Command Flow Control 422
4.5 Command Error Handling 422
5 HCI Data Formats 424
5.1 Introduction 424
5.2 Data and Parameter Formats 424
5.3 Handles 425
5.3.1 Primary Controller Handles 425
5.3.2 AMP Controller Handles 426
5.4 Exchange of HCI-Specific Information 427
5.4.1 HCI Command Packet 427
5.4.2 HCI ACL Data Packets 428
5.4.3 HCI Synchronous Data Packets 431
5.4.4 HCI Event Packet 434
6 HCI Configuration Parameters 435
6.1 Scan Enable 435
6.2 Inquiry Scan Interval 435
6.3 Inquiry Scan Window 436
6.4 Inquiry Scan Type 436
6.5 Inquiry Mode 436
6.6 Page Timeout 437
6.7 Connection Accept Timeout 437
6.8 Page Scan Interval 438
6.9 Page Scan Window 438
6.10 Page Scan Period Mode (Deprecated) 438
6.11 Page Scan Type 439
6.12 Voice Setting 439
6.13 PIN Type 440
6.14 Link Key 440
6.15 Failed Contact Counter 440
6.16 Authentication Enable 441
6.17 Hold Mode Activity 441
6.18 Link Policy Settings 443
6.19 Flush Timeout 444
6.20 Num Broadcast Retransmissions 444
6.21 Link Supervision Timeout 445
6.22 Synchronous Flow Control Enable 445

6.23 Local Name 446
6.24 Extended Inquiry Response 446
6.25 Erroneous Data Reporting 446
6.26 Class Of Device 447
6.27 Supported Commands 447
6.28 Logical Link Accept Timeout 454
6.29 Location Domain Aware 455
6.30 Location Domain 455
6.31 Location Domain Options 455
6.32 Location Options 456
6.33 Flow Control Mode 456
6.34 LE Supported Host 457
6.35 Simultaneous LE Host 457

7 HCI Commands and Events 458

7.1 Link Control Commands 458
7.1.1 Inquiry Command 459
7.1.2 Inquiry Cancel Command 461
7.1.3 Periodic Inquiry Mode Command 462
7.1.4 Exit Periodic Inquiry Mode Command 465
7.1.5 Create Connection Command 466
7.1.6 Disconnect Command 469
7.1.7 Create Connection Cancel Command 470
7.1.8 Accept Connection Request Command 472
7.1.9 Reject Connection Request Command 474
7.1.10 Link Key Request Reply Command 475
7.1.11 Link Key Request Negative Reply Command 477
7.1.12 PIN Code Request Reply Command 478
7.1.13 PIN Code Request Negative Reply Command 480
7.1.14 Change Connection Packet Type Command 481
7.1.15 Authentication Requested Command 484
7.1.16 Set Connection Encryption Command 486
7.1.17 Change Connection Link Key Command 487
7.1.18 Master Link Key Command 488
7.1.19 Remote Name Request Command 489
7.1.20 Remote Name Request Cancel Command 491
7.1.21 Read Remote Supported Features Command 493
7.1.22 Read Remote Extended Features Command 494
7.1.23 Read Remote Version Information Command 496
7.1.24 Read Clock Offset Command 497

7.1.25 Read LMP Handle Command 498
7.1.26 Setup Synchronous Connection Command 500
7.1.27 Accept Synchronous Connection Request Command 504
7.1.28 Reject Synchronous Connection Request Command . 508
7.1.29 IO Capability Request Reply Command 509
7.1.30 User Confirmation Request Reply Command 512
7.1.31 User Confirmation Request Negative Reply Command 513
7.1.32 User Passkey Request Reply Command 514
7.1.33 User Passkey Request Negative Reply Command 515
7.1.34 Remote OOB Data Request Reply Command 516
7.1.35 Remote OOB Data Request Negative Reply Command 518
7.1.36 IO Capability Request Negative Reply Command 519
7.1.37 Create Physical Link Command 520
7.1.38 Accept Physical Link Command 523
7.1.39 Disconnect Physical Link Command 525
7.1.40 Create Logical Link Command 527
7.1.41 Accept Logical Link Command 529
7.1.42 Disconnect Logical Link Command 531
7.1.43 Logical Link Cancel Command 532
7.1.44 Flow Spec Modify Command 534
7.2 Link Policy Commands 535
7.2.1 Hold Mode Command 536
7.2.2 Sniff Mode Command 538
7.2.3 Exit Sniff Mode Command 541
7.2.4 Park State Command 542
7.2.5 Exit Park State Command 544
7.2.6 QoS Setup Command 545
7.2.7 Role Discovery Command 548
7.2.8 Switch Role Command 549
7.2.9 Read Link Policy Settings Command 551
7.2.10 Write Link Policy Settings Command 553
7.2.11 Read Default Link Policy Settings Command 555
7.2.12 Write Default Link Policy Settings Command 556
7.2.13 Flow Specification Command 557
7.2.14 Sniff Subrating Command 559
7.3 Controller & Baseband Commands 560

7.3.1 Set Event Mask Command....561
7.3.2 Reset Command....564
7.3.3 Set Event Filter Command....565
7.3.4 Flush Command....571
7.3.5 Read PIN Type Command....573
7.3.6 Write PIN Type Command....574
7.3.7 Create New Unit Key Command....575
7.3.8 Read Stored Link Key Command....576
7.3.9 Write Stored Link Key Command....578
7.3.10 Delete Stored Link Key Command....580
7.3.11 Write Local Name Command....582
7.3.12 Read Local Name Command....583
7.3.13 Read Connection Accept Timeout Command....584
7.3.14 Write Connection Accept Timeout Command....585
7.3.15 Read Page Timeout Command....586
7.3.16 Write Page Timeout Command....587
7.3.17 Read Scan Enable Command....588
7.3.18 Write Scan Enable Command....589
7.3.19 Read Page Scan Activity Command....590
7.3.20 Write Page Scan Activity Command....592
7.3.21 Read Inquiry Scan Activity Command....593
7.3.22 Write Inquiry Scan Activity Command....595
7.3.23 Read Authentication Enable Command....596
7.3.24 Write Authentication Enable Command....597
7.3.25 Read Class of Device Command....598
7.3.26 Write Class of Device Command....599
7.3.27 Read Voice Setting Command....600
7.3.28 Write Voice Setting Command....601
7.3.29 Read Automatic Flush Timeout Command....602
7.3.30 Write Automatic Flush Timeout Command....604
7.3.31 Read Num Broadcast Retransmissions Command....606
7.3.32 Write Num Broadcast Retransmissions Command....607
7.3.33 Read Hold Mode Activity Command....608
7.3.34 Write Hold Mode Activity Command....609
7.3.35 Read Transmit Power Level Command....610
7.3.36 Read Synchronous Flow Control Enable Command....612
7.3.37 Write Synchronous Flow Control Enable Command....613
7.3.38 Set Controller To Host Flow Control Command....614

7.3.39 Host Buffer Size Command....616
7.3.40 Host Number Of Completed Packets Command....618
7.3.41 Read Link Supervision Timeout Command....620
7.3.42 Write Link Supervision Timeout Command....622
7.3.43 Read Number Of Supported IAC Command....624
7.3.44 Read Current IAC LAP Command....625
7.3.45 Write Current IAC LAP Command....626
7.3.46 Set AFH Host Channel Classification Command....628
7.3.47 Read Inquiry Scan Type Command....629
7.3.48 Write Inquiry Scan Type Command....630
7.3.49 Read Inquiry Mode Command....631
7.3.50 Write Inquiry Mode Command....632
7.3.51 Read Page Scan Type Command....633
7.3.52 Write Page Scan Type Command....634
7.3.53 Read AFH Channel Assessment Mode Command....635
7.3.54 Write AFH Channel Assessment Mode Command....636
7.3.55 Read Extended Inquiry Response Command....638
7.3.56 Write Extended Inquiry Response Command....639
7.3.57 Refresh Encryption Key Command....640
7.3.58 Read Simple Pairing Mode Command....641
7.3.59 Write Simple Pairing Mode Command....642
7.3.60 Read Local OOB Data Command....644
7.3.61 Read Inquiry Response Transmit Power Level Command....646
7.3.62 Write Inquiry Transmit Power Level Command....647
7.3.63 Send Keypress Notification Command....648
7.3.64 Read Default Erroneous Data Reporting....650
7.3.65 Write Default Erroneous Data Reporting....651
7.3.66 Enhanced Flush Command....652
7.3.67 Read Logical Link Accept Timeout Command....654
7.3.68 Write Logical Link Accept Timeout Command....655
7.3.69 Set Event Mask Page 2 Command....656
7.3.70 Read Location Data Command....658
7.3.71 Write Location Data Command....659
7.3.72 Read Flow Control Mode Command....660
7.3.73 Write Flow Control Mode Command....661
7.3.74 Read Enhanced Transmit Power Level Command....662
7.3.75 Read Best Effort Flush Timeout Command....665

7.3.76 Write Best Effort Flush Timeout Command 666
7.3.77 Short Range Mode Command 667
7.3.78 Read LE Host Supported Command 668
7.3.79 Write LE Host Supported Command 669
7.4 Informational Parameters 670
7.4.1 Read Local Version Information Command 670
7.4.2 Read Local Supported Commands Command 672
7.4.3 Read Local Supported Features Command 673
7.4.4 Read Local Extended Features Command 674
7.4.5 Read Buffer Size Command 676
7.4.6 Read BD_ADDR Command 678
7.4.7 Read Data Block Size Command 679
7.5 Status Parameters 680
7.5.1 Read Failed Contact Counter Command 681
7.5.2 Reset Failed Contact Counter Command 683
7.5.3 Read Link Quality Command 685
7.5.4 Read RSSI Command 687
7.5.5 Read AFH Channel Map Command 689
7.5.6 Read Clock Command 691
7.5.7 Read Encryption Key Size Command 693
7.5.8 Read Local AMP Info Command 695
7.5.9 Read Local AMP ASSOC Command 700
7.5.10 Write Remote AMP ASSOC Command 703
7.6 Testing Commands 704
7.6.1 Read Loopback Mode Command 705
7.6.2 Write Loopback Mode Command 706
7.6.3 Enable Device Under Test Mode Command 709
7.6.4 Write Simple Pairing Debug Mode Command 710
7.6.5 Enable AMP Receiver Reports Command 712
7.6.6 AMP Test End Command 713
7.6.7 AMP Test Command 714
7.7 Events 715
7.7.1 Inquiry Complete Event 715
7.7.2 Inquiry Result Event 716
7.7.3 Connection Complete Event 718
7.7.4 Connection Request Event 719
7.7.5 Disconnection Complete Event 721
7.7.6 Authentication Complete Event 722

7.7.7 Remote Name Request Complete Event 723
7.7.8 Encryption Change Event 724
7.7.9 Change Connection Link Key Complete Event 725
7.7.10 Master Link Key Complete Event 726
7.7.11 Read Remote Supported Features Complete Event ... 727
7.7.12 Read Remote Version Information Complete Event ... 728
7.7.13 QoS Setup Complete Event 730
7.7.14 Command Complete Event 732
7.7.15 Command Status Event 733
7.7.16 Hardware Error Event 734
7.7.17 Flush Occurred Event 735
7.7.18 Role Change Event 736
7.7.19 Number Of Completed Packets Event 737
7.7.20 Mode Change Event 739
7.7.21 Return Link Keys Event 741
7.7.22 PIN Code Request Event 742
7.7.23 Link Key Request Event 743
7.7.24 Link Key Notification Event 744
7.7.25 Loopback Command Event 746
7.7.26 Data Buffer Overflow Event 747
7.7.27 Max Slots Change Event 748
7.7.28 Read Clock Offset Complete Event 749
7.7.29 Connection Packet Type Changed Event 750
7.7.30 QoS Violation Event 752
7.7.31 Page Scan Repetition Mode Change Event 753
7.7.32 Flow Specification Complete Event 754
7.7.33 Inquiry Result with RSSI Event 756
7.7.34 Read Remote Extended Features Complete Event 758
7.7.35 Synchronous Connection Complete Event 760
7.7.36 Synchronous Connection Changed Event 762
7.7.37 Sniff Subrating Event 764
7.7.38 Extended Inquiry Result Event 766
7.7.39 Encryption Key Refresh Complete Event 768
7.7.40 IO Capability Request Event 769
7.7.41 IO Capability Response Event 770
7.7.42 User Confirmation Request Event 772
7.7.43 User Passkey Request Event 773
7.7.44 Remote OOB Data Request Event 774

7.7.45 Simple Pairing Complete Event....775
7.7.46 Link Supervision Timeout Changed Event....776
7.7.47 Enhanced Flush Complete Event....777
7.7.48 User Passkey Notification Event....778
7.7.49 Keypress Notification Event....779
7.7.50 Remote Host Supported Features Notification Event..780
7.7.51 Physical Link Complete Event....781
7.7.52 Channel Selected Event....782
7.7.53 Disconnection Physical Link Complete Event....783
7.7.54 Physical Link Loss Early Warning Event....784
7.7.55 Physical Link Recovery Event....785
7.7.56 Logical Link Complete Event....786
7.7.57 Disconnection Logical Link Complete Event....787
7.7.58 Flow Spec Modify Complete Event....788
7.7.59 Number Of Completed Data Blocks Event....789
7.7.60 Short Range Mode Change Complete Event....791
7.7.61 AMP Status Change Event....792
7.7.62 AMP Start Test Event....794
7.7.63 AMP Test End Event....795
7.7.64 AMP Receiver Report Event....796
7.7.65 LE Meta Event....798
7.8 LE Controller Commands....806
7.8.1 LE Set Event Mask Command....806
7.8.2 LE Read Buffer Size Command....808
7.8.3 LE Read Local Supported Features Command....810
7.8.4 LE Set Random Address Command....811
7.8.5 LE Set Advertising Parameters Command....812
7.8.6 LE Read Advertising Channel Tx Power Command....815
7.8.7 LE Set Advertising Data Command....816
7.8.8 LE Set Scan Response Data Command....817
7.8.9 LE Set Advertise Enable Command....818
7.8.10 LE Set Scan Parameters Command....820
7.8.11 LE Set Scan Enable Command....823
7.8.12 LE Create Connection Command....825
7.8.13 LE Create Connection Cancel Command....829
7.8.14 LE Read White List Size Command....830
7.8.15 LE Clear White List Command....831
7.8.16 LE Add Device To White List Command....832

7.8.17 LE Remove Device From White List Command.....833
7.8.18 LE Connection Update Command 835
7.8.19 LE Set Host Channel Classification Command.....838
7.8.20 LE Read Channel Map Command 839
7.8.21 LE Read Remote Used Features Command 841
7.8.22 LE Encrypt Command.....842
7.8.23 LE Rand Command.....844
7.8.24 LE Start Encryption Command.....845
7.8.25 LE Long Term Key Request Reply Command 847
7.8.26 LE Long Term Key Request Negative Reply Command 849
7.8.27 LE Read Supported States Command.....850
7.8.28 LE Receiver Test Command 853
7.8.29 LE Transmitter Test Command 854
7.8.30 LE Test End Command 856

8 List of Figures 857

9 List of Tables 858

10 Appendix A: Deprecated Commands, Events and Configuration Parameters 859

10.1 Read Page Scan Mode Command 860
10.2 Write Page Scan Mode Command 861
10.3 Read Page Scan Period Mode Command.....862
10.4 Write Page Scan Period Mode Command.....863
10.5 Add SCO Connection Command.....864
10.6 Page Scan Mode Change Event 866
10.7 Read Country Code Command 867
10.8 Read Encryption Mode Command.....868
10.9 Write Encryption Mode Command.....868
10.10 Deprecated Parameters.....869
10.10.1 Encryption Mode 869
10.10.2 Page Scan Mode 871

Part F
MESSAGE SEQUENCE CHARTS

1 Introduction 877

1.1 Notation 877
1.2 Flow of Control.....878
1.3 Example MSC.....878

2 Services Without Connection Request.....879

2.1 Remote Name Request....879
2.2 One-time Inquiry....881
2.3 Periodic Inquiry....883

3 ACL Connection Establishment and Detachment....885
3.1 Connection Setup....886

4 Optional Activities After ACL Connection Establishment....893
4.1 Authentication Requested....893
4.2 Simple Pairing Message Sequence Charts....894
4.2.1 Optional OOB Information Collection....894
4.2.2 Enable Simple Pairing....895
4.2.3 Connection Establishment....895
4.2.4 L2CAP Connection Request for a Secure Service....895
4.2.5 Optional OOB Information Transfer....896
4.2.6 Start Simple Pairing....896
4.2.7 IO Capability Exchange....897
4.2.8 Public Key Exchange....897
4.2.9 Authentication....898
4.2.10 Numeric Comparison....898
4.2.11 Numeric Comparison Failure on Initiating Side....899
4.2.12 Numeric Comparison Failure on Responding Side....900
4.2.13 Passkey Entry....901
4.2.14 Passkey Entry Failure on Responding Side....902
4.2.15 Passkey Entry Failure on Initiator Side....903
4.2.16 Out of Band....903
4.2.17 OOB Failure on Initiator Side....904
4.2.18 DHKey Checks....905
4.2.19 Calculate Link Key....905
4.2.20 Enable Encryption....906
4.2.21 L2CAP Connection Response....907
4.3 Link Supervision Timeout Changed Event....907
4.4 Set Connection Encryption....908
4.5 Change Connection Link Key....909
4.6 Change Connection Link Key with Encryption Pause and Resume....909
4.7 Master Link Key....911
4.8 Read Remote Supported Features....913
4.9 Read Remote Extended Features....913
4.10 Read Clock Offset....914
4.11 Role Switch on an Encrypted Link using Encryption Pause and Resume....914

4.12 Refreshing Encryption Keys 915
4.13 Read Remote Version Information 916
4.14 QOS Setup 916
4.15 Switch Role 917
4.16 AMP Physical Link Creation and Disconnect 918
4.16.1 Physical Link Establishment 919
4.16.2 Logical Link Creation 922
4.17 AMP Test Mode Sequence Charts 925
4.17.1 Discover the AMP Present and Running Transmitter and Receiver Tests 925

5 Synchronous Connection Establishment and Detachment 929
5.1 Synchronous Connection Setup 929

6 Sniff, Hold and Park 934
6.1 Sniff Mode 934
6.2 Hold Mode 935
6.3 Park State 937

7 Buffer Management, Flow Control 940

8 Loopback Mode 942
8.1 Local Loopback Mode 942
8.2 Remote Loopback Mode 944

9 List of Figures 946

Part G
SAMPLE DATA

1 Encryption Sample Data 953
1.1 Generating Kc' from Kc, ... 953
1.2 First Set of Sample Data 956
1.3 Second Set of Sample Data 964
1.4 Third Set of Samples 972
1.5 Fourth Set of Samples 980

2 Frequency Hopping Sample Data 988
2.1 First set 989
2.2 Second set 995
2.3 Third set 1001

3 Access Code Sample Data 1007

4 HEC and Packet Header Sample Data 1010

5 CRC Sample Data 1011

6 Complete Sample Packets 1012
6.1 Example of DH1 Packet 1012

6.2 Example of DM1 Packet....1013
7 Simple Pairing Sample Data....1014
7.1 P-192 Sample Data....1014
7.1.1 Data Set 1....1014
7.1.2 Data Set 2....1014
7.1.3 Data Set 3....1014
7.1.4 Data Set 4....1014
7.1.5 Data Set 5....1014
7.1.6 Data Set 6....1015
7.1.7 Data Set 7....1015
7.1.8 Data Set 8....1015
7.1.9 Data Set 9....1015
7.1.10 Data Set 10....1015
7.2 Hash Functions Sample Data....1015
7.2.1 f1()....1016
7.2.2 g()....1017
7.2.3 f2()....1017
7.2.4 f3()....1017
7.2.5 h2()....1024
8 Whitening Sequence Sample Data....1026
9 FEC Sample Data....1029
10 Encryption Key Sample Data....1030
10.1 Four Tests of E1....1030
10.2 Four Tests of E21....1035
10.3 Three Tests of E22....1037
10.4 Tests of E22 With Pin Augmenting....1039
10.5 Four Tests of E3....1049

Part H
SECURITY SPECIFICATION

1 Security Overview....1059
1.1 Pausing Encryption and Role Switch....1060
1.2 Change Connection Link Keys....1060
1.3 Periodically Refreshing Encryption Keys....1060
2 Random Number Generation....1061
3 Key Management....1063
3.1 Key Types....1063
3.2 Key Generation and Initialization....1065
3.2.1 Generation of initialization key,....1066

3.2.2 Authentication 1066
3.2.3 Generation of a unit key 1066
3.2.4 Generation of a combination key 1067
3.2.5 Generating the encryption key 1068
3.2.6 Point-to-multipoint configuration 1069
3.2.7 Modifying the link keys 1069
3.2.8 Generating a master key 1070

4 Encryption 1072
4.1 Encryption Key Size Negotiation 1073
4.2 Encryption of Broadcast Messages 1073
4.3 Encryption Concept 1074
4.4 Encryption Algorithm 1075
4.4.1 The operation of the cipher 1077
4.5 LFSR Initialization 1078
4.6 Key Stream Sequence 1081

5 Authentication 1082
5.1 Repeated Attempts 1084

6 The Authentication And Key-Generating Functions 1085
6.1 The Authentication Function E1 1085
6.2 The Functions Ar and A'r 1087
6.2.1 The round computations 1087
6.2.2 The substitution boxes "e" and "l" 1087
6.2.3 Key scheduling 1088
6.3 E2-Key Generation Function for Authentication 1089
6.4 E3-Key Generation Function for Encryption 1091

7 Secure Simple Pairing 1092
7.1 Phase 1: Public Key Exchange 1093
7.2 Phase 2: Authentication Stage 1 1094
7.2.1 Authentication Stage 1: Numeric Comparison Protocol 1094
7.2.2 Authentication Stage 1: Out of Band Protocol 1096
7.2.3 Authentication Stage 1: Passkey Entry Protocol 1098
7.3 Phase 3: Authentication Stage 2 1100
7.4 Phase 4: Link Key Calculation 1101
7.5 Phase 5: LMP Authentication and Encryption 1101
7.6 Elliptic Curve Definition 1101
7.7 Cryptographic Function Definitions 1102
7.7.1 The Simple Pairing Commitment Function f1 1102
7.7.2 The Simple Pairing Numeric Verification Function g 1103
7.7.3 The Simple Pairing Key Derivation Function f2 1104

7.7.4 The Simple Pairing Check Function f3 1105
7.7.5 The Simple Pairing AMP Key Derivation Function h2 1106
8 AMP Security 1108
8.1 Creation of the Initial Generic AMP Link Key 1108
8.2 Creation of Dedicated AMP Link Keys 1108
8.3 Debug Considerations 1109
9 List of Figures 1111

RADIO SPECIFICATION

Ford Ex. 1052

CONTENTS

1 Scope 33

2 Frequency Bands and Channel Arrangement 35

3 Transmitter Characteristics 36

3.1 Basic Rate 37

3.1.1 Modulation Characteristics 37

3.1.2 Spurious Emissions 38

3.1.3 Radio Frequency Tolerance 39

3.2 Enhanced Data Rate 39

3.2.1 Modulation Characteristics 40

3.2.2 Spurious Emissions 42

3.2.3 Radio Frequency Tolerance 44

3.2.4 Relative Transmit Power 44

4 Receiver Characteristics 46

4.1 Basic Rate 46

4.1.1 Actual Sensitivity Level 46

4.1.2 Interference Performance 46

4.1.3 Out-of-Band Blocking 47

4.1.4 Intermodulation Characteristics 47

4.1.5 Maximum Usable Level 48

4.1.6 Receiver Signal Strength Indicator 48

4.1.7 Reference Signal Definition 48

4.2 Enhanced Data Rate 48

4.2.1 Actual Sensitivity Level 48

4.2.2 BER Floor Performance 48

4.2.3 Interference Performance 48

4.2.4 Maximum Usable Level 49

4.2.5 Out-of-Band and Intermodulation Characteristics 50

4.2.6 Reference Signal Definition 50

5 Appendix A 51

5.1 Nominal Test Conditions 51

5.1.1 Nominal temperature 51

5.1.2 Nominal power source 51

5.2 Extreme Test Conditions 52

5.2.1 Extreme temperatures 52

5.2.2 Extreme power source voltages 52

6 Appendix B 53

7 Appendix C 54

7.1 Enhanced Data Rate Modulation Accuracy 54

1 SCOPE

Bluetooth devices operate in the unlicensed 2.4 GHz ISM (Industrial Scientific Medical) band. A frequency hop transceiver is applied to combat interference and fading.

Two modulation modes are defined. A mandatory mode, called Basic Rate, uses a shaped, binary FM modulation to minimize transceiver complexity. An optional mode, called Enhanced Data Rate, uses PSK modulation and has two variants: π/4-DQPSK and 8DPSK. The symbol rate for all modulation modes is 1 Ms/s. The gross air data rate is 1 Mbps for Basic Rate, 2 Mbps for Enhanced Data Rate using π/4-DQPSK and 3 Mbps for Enhanced Data Rate using 8DPSK.

For full duplex transmission, a Time Division Duplex (TDD) scheme is used in both modes. This specification defines the requirements for a Bluetooth radio for the Basic Rate and Enhanced Data Rate modes.

Requirements are defined for two reasons:

- Provide compatibility between radios used in the system
- Define the quality of the system

The Bluetooth radio shall fulfil the stated requirements under the operating conditions specified in Appendix A and Appendix B. The radio parameters shall be measured according to the methods described in the RF Test Specification.

This specification is based on the established regulations for Europe, Japan and North America. The standard documents listed below are only for information, and are subject to change or revision at any time.

The Bluetooth SIG maintains an online database of regulations that apply to Bluetooth technology in the 2.4 GHz ISM band, posted at https://www.bluetooth.org/regulatory/newindex.cfm.

Europe:

Approval Standards: European Telecommunications Standards Institute, ETSI
Documents: EN 300 328, ETS 300-826
Approval Authority: National Type Approval Authorities

Japan:

Approval Standards: Association of Radio Industries and Businesses, ARIB
Documents: ARIB STD-T66
Approval Authority: Ministry of Post and Telecommunications, MPT.

North America:

Approval Standards: Federal Communications Commission, FCC, USA
Documents: CFR47, Part 15, Sections 15.205, 15.209, 15.247 and 15.249

Approval Standards: Industry Canada, IC, Canada
Documents: RSS-139 and RSS-210

Approval Authority: FCC (USA), Industry Canada (Canada)

2 FREQUENCY BANDS AND CHANNEL ARRANGEMENT

The Bluetooth system operates in the 2.4 GHz ISM band. This frequency band is 2400 - 2483.5 MHz.

Regulatory Range	RF Channels
2.400-2.4835 GHz	f=2402+k MHz, k=0,…,78

Table 2.1: Operating frequency bands

RF channels are spaced 1 MHz and are ordered in channel number k as shown in Table 2.1.

3 TRANSMITTER CHARACTERISTICS

The requirements stated in this section are given as power levels at the antenna connector of the Bluetooth device. If the device does not have a connector, a reference antenna with 0 dBi gain is assumed.

Due to difficulty in measurement accuracy in radiated measurements, systems with an integral antenna should provide a temporary antenna connector during type approval.

If transmitting antennas of directional gain greater than 0 dBi are used, the power level delivered to the antenna shall be compensated to comply with the applicable paragraphs in EN 300 328, EN 301 489-17and FCC part 15.

Bluetooth devices are classified into three power classes based on the modulation mode with the highest output power. That modulation mode shall meet the specification requirements for the power class. The other modulation modes do not need to meet the power class requirements of the applicable device power class, but the maximum output power used for the other modulation modes shall not exceed the maximum power of the modulation mode used for classification.

Power Class	Maximum Output Power (Pmax)	Nominal Output Power	Minimum Output Power[1]	Power Control
1	100 mW (20 dBm)	N/A	1 mW (0 dBm)	Pmin<+4 dBm to Pmax Optional: Pmin[2] to Pmax
2	2.5 mW (4 dBm)	1 mW (0 dBm)	0.25 mW (-6 dBm)	Optional: Pmin[2] to Pmax
3	1 mW (0 dBm)	N/A	N/A	Optional: Pmin[2] to Pmax

Table 3.1: Power classes

1. Minimum output power at maximum power setting.
2. The lower power limit Pmin<-30dBm is suggested but is not mandatory, and may be chosen according to application needs.

A power class 1 device shall support received power control requests. Power control may be used to limit the transmitted power of a remote device to no more than +4 dBm to prevent saturation of the receiver of the local device. Support of received power control requests is optional for class 2 and class 3 devices but may be used to optimize the power consumption and reduce the overall interference level for all devices that use the same spectrum that Bluetooth devices use. The power steps shall form a monotonic sequence, with a maximum step size of 8 dB and a minimum step size of 2 dB.

A class 1 device shall support handling of power control requests and shall be able to adjust its transmit power down to 4 dBm or less. A device that supports changing its transmit power controls the output power in a physical link in response to LMP commands received from a peer device that is capable of sending such requests (see Link Manager Protocol Specification).

In a connection, the output power shall not exceed the maximum output power of power class 2 for transmitting packets if the receiving device does not support the necessary messaging for sending the power control messages, see Link Manager Protocol Section 4.1.3 on page 230. In this case, the transmitting device shall comply with the rules of a class 2 or class 3 device.

If a class 1 device is paging or inquiring very close to another device, the input power can be larger than the requirement in Section 4.1.5 on page 48. This can cause the receiving device to fail to respond. It may therefore be useful to page at class 2 or 3 power in addition to paging at power class 1.

The transmit power level difference between the packet headers of all supported packet types at any given power step shall not exceed 10dB.

When communicating with a device that does not support enhanced power control, an enhanced power control device shall have an equal number of steps for each supported modulation scheme so that all supported modulation modes shall reach their respective maximum (or minimum) levels at the same time. The power levels may vary per modulation mode.

Devices shall not exceed the maximum allowed transmit power levels set by the regulatory bodies that have jurisdiction over the locales in which the device is to be sold or intended to operate. Implementers should be aware that the maximum transmit power level permitted under a given set of regulations might not be the same for all modulation modes.

3.1 BASIC RATE

3.1.1 Modulation Characteristics

The Modulation is GFSK (Gaussian Frequency Shift Keying) with a bandwidth-bit period product BT=0.5. The Modulation index shall be between 0.28 and 0.35. A binary one shall be represented by a positive frequency deviation, and a binary zero shall be represented by a negative frequency deviation. The symbol timing shall be less than ±20 ppm.

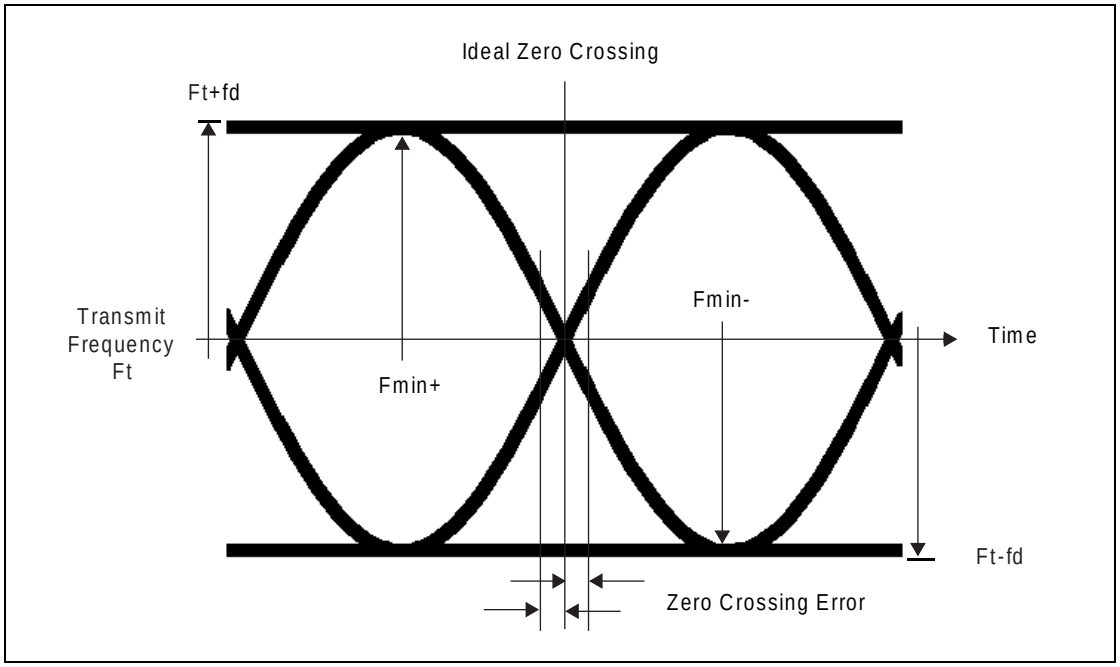


Figure 3.1: GFSK parameters definition.

For each transmission, the minimum frequency deviation, Fmin = min{|Fmin+|, Fmin-}, which corresponds to 1010 sequence shall be no smaller than ±80% of the frequency deviation (fd) with respect to the transmit frequency Ft, which corresponds to a 00001111 sequence.

In addition, the minimum frequency deviation shall never be smaller than 115 kHz. The data transmitted has a symbol rate of 1 Ms/s.

The zero crossing error is the time difference between the ideal symbol period and the measured crossing time. This shall be less than ± 1/8 of a symbol period.

3.1.2 Spurious Emissions

In-band spurious emissions shall be measured with a frequency hopping radio transmitting on one RF channel and receiving on a second RF channel; this means that the synthesizer may change RF channels between reception and transmission, but always returns to the same transmit RF channel. The equipment manufacturer is responsible for compliance in the intended country of use.

3.1.2.1 In-band spurious emission

Within the ISM band the transmitter shall pass a spectrum mask, given in Table 3.2. The spectrum shall comply with the 20 dB bandwidth definition in FCC part 15.247 and shall be measured accordingly. In addition to the FCC requirement an adjacent channel power on adjacent channels with a difference

in RF channel number of two or greater is defined. This adjacent channel power is defined as the sum of the measured power in a 1 MHz bandwidth. The transmitted power shall be measured in a 100 kHz bandwidth using maximum hold. The device shall transmit on RF channel M and the adjacent channel power shall be measured on RF channel number N. The transmitter shall transmit a pseudo random data pattern in the payload throughout the test.

Frequency offset	Transmit Power
± 500 kHz	-20 dBc
2MHz (|M-N| = 2)	-20 dBm
3MHz or greater (|M-N| ≥ 3)	-40 dBm

Table 3.2: Transmit Spectrum mask.

Note: If the output power is less than 0dBm then, wherever appropriate, the FCC's 20 dB relative requirement overrules the absolute adjacent channel power requirement stated in the above table.

Exceptions are allowed in up to three bands of 1 MHz width centered on a frequency which is an integer multiple of 1 MHz. They shall comply with an absolute value of –20 dBm.

3.1.3 Radio Frequency Tolerance

The transmitted initial center frequency shall be within ±75 kHz from F_c. The initial frequency accuracy is defined as being the frequency accuracy before any packet information is transmitted. Note that the frequency drift requirement is not included in the ±75 kHz.

The limits on the transmitter center frequency drift within a packet are specified in Table 3.3. The different packets are defined in the Baseband Specification.

Duration of Packet	Frequency Drift
Max length one slot packet	±25 kHz
Max length three slot packet	±40 kHz
Max length five slot packet	±40 kHz
Maximum drift rate[1]	400 Hz/µs

Table 3.3: Maximum allowable frequency drifts in a packet.

1. The maximum drift rate is allowed anywhere in a packet.

3.2 ENHANCED DATA RATE

A key characteristic of the Enhanced Data Rate mode is that the modulation mode is changed within the packet. The access code and packet header, as defined in Table 6.1 in the Baseband Specification, are transmitted with the

Basic Rate 1 Mbps GFSK modulation mode, whereas the subsequent synchronization sequence, payload, and trailer sequence are transmitted using the Enhanced Data Rate PSK modulation mode.

3.2.1 Modulation Characteristics

During access code and packet header transmission the Basic Rate GFSK modulation mode shall be used. During the transmission of the synchronization sequence, payload, and trailer sequence a PSK type of modulation with a data rate of 2 Mbps or optionally 3 Mbps shall be used. The following subsections specify the PSK modulation for this transmission.

3.2.1.1 Modulation Method Overview

The PSK modulation format defined for the 2 Mbps transmission shall be $\pi/4$ rotated differential encoded quaternary phase shift keying ($\pi/4$-DQPSK).

The PSK modulation format defined for the 3 Mbps transmission shall be differential encoded 8-ary phase shift keying (8DPSK).

The modulation shall employ square-root raised cosine pulse shaping to generate the equivalent lowpass information-bearing signal v(t). The output of the transmitter shall be a bandpass signal that can be represented as

$$S(t) = Re\left[v(t)e^{j2\pi F_c t}\right] \qquad \textit{(EQ 1)}$$

with F_c denoting the carrier frequency.

3.2.1.2 Differential Phase Encoding

For the M-ary modulation, the binary data stream $\{b_n\}$, n=1,2,3, …N, shall be mapped onto a corresponding sequence {Sk}, k=1,2, …$N/\log_2(M)$ of complex valued signal points. M=4 applies for 2 Mbps and M=8 applies for 3 Mbps. Gray coding shall be applied as shown in Table 3.4 and Table 3.5. In the event that the length of the binary data stream N is not an integer multiple of log2(M), the last symbol of the sequence $\{S_k\}$ shall be formed by appending data zeros to the appropriate length. The signal points S_k shall be defined by:

$$S_k = S_{k-1}e^{j\varphi_k} \qquad k = 1, 2, ..N/\log_2(M) \qquad \text{(EQ 2)}$$

$$S_0 = e^{j\phi} \qquad \text{with } \phi \in [0, 2\pi) \qquad \text{(EQ 3)}$$

The relationship between the binary input b_k and the phase ϕ_k shall be as defined in Table 3.4 for the 2 Mbps transmission and in Table 3.5 for the 3 Mbps transmission.

b_{2k-1}	b_{2k}	φ_k
0	0	$\pi/4$
0	1	$3\pi/4$
1	1	$-3\pi/4$
1	0	$-\pi/4$

Table 3.4: $\pi/4$-DQPSK mapping.

b_{3k-2}	b_{3k-1}	b_{3k}	φ_k
0	0	0	0
0	0	1	$\pi/4$
0	1	1	$\pi/2$
0	1	0	$3\pi/4$
1	1	0	π
1	1	1	$-3\pi/4$
1	0	1	$-\pi/2$
1	0	0	$-\pi/4$

Table 3.5: 8DPSK mapping.

3.2.1.3 Pulse Shaping

The lowpass equivalent information-bearing signal v(t) shall be generated according to

$$v(t) = \sum_k S_k p(t - kT) \qquad \text{(EQ 4)}$$

in which the symbol period T shall be 1μs.

The frequency spectrum P(f), which corresponds to the square-root raised cosine pulse p(t) of the pulse shaping filter is:

$$|P(f)| = \begin{cases} 1 & 0 \le |f| \le \frac{1-\beta}{2T} \\ \sqrt{\frac{1}{2}\left(1 - \sin\left(\frac{\pi(2\ |f|T - 1)}{2\beta}\right)\right)} & \frac{1-\beta}{2T} \le |f| \le \frac{1+\beta}{2T} \\ 0 & \text{elsewhere} \end{cases} \quad \text{(EQ 5)}$$

The roll off factor β shall be 0.4.

3.2.1.4 Modulation Accuracy

The measurement of modulation accuracy utilizes differential error vector magnitude (DEVM) with tracking of the carrier frequency drift. The definition of DEVM and the derivation of the RMS and peak measures of DEVM are given in Appendix C.

The DEVM shall be measured over the synchronization sequence and payload portions of the packet, but not the trailer symbols. For each modulation method and each measurement carrier frequency, DEVM measurement is made over a total of 200 non-overlapping blocks, where each block contains 50 symbols.

The transmitted packets shall be the longest supported packet type for each modulation method, as defined in Table 6.9 and Table 6.10 in the Baseband part.

The DEVM is measured using a square-root raised cosine filter, with a roll-off of 0.4 and a 3 dB bandwidth of ±500 kHz.

3.2.1.4.1 RMS DEVM

The RMS DEVM for any of the measured blocks shall not exceed 0.20 for π/4-DQPSK and 0.13 for 8DPSK.

3.2.1.4.2 99% DEVM

The 99% DEVM (defined as the DEVM value for which 99% of measured symbols have a lower DEVM) shall not exceed 0.30 for π/4-DQPSK and 0.20 for 8DPSK.

3.2.1.4.3 Peak DEVM

The Peak DEVM shall not exceed 0.35 for π/4-DQPSK and 0.25 for 8DPSK.

3.2.2 Spurious Emissions

In-band spurious emissions shall be measured with a frequency hopping radio transmitting on one RF channel and receiving on a second RF channel; this means that the synthesizer may change RF channels between reception and

transmission, but always returns to the same transmit RF channel. The equipment manufacturer is responsible for compliance in the intended country of use.

3.2.2.1 In-band Spurious Emission

Within the ISM band the power spectral density of the transmitter shall comply with the following requirements when sending pseudo random data. All power measurements shall use a 100 kHz bandwidth with maximum hold. The power measurements between 1 MHz and 1.5 MHz from the carrier shall be at least 26 dB below the maximum power measurement up to 500 kHz from the carrier. The adjacent channel power for channels at least 2 MHz from the carrier is defined as the sum of the power measurements over a 1 MHz channel and shall not exceed -20 dBm for the second adjacent channels and -40 dBm for the third and subsequent adjacent channels. These requirements shall apply to the transmitted signal from the start of the guard time to the end of the power down ramp. The spectral mask is illustrated in Figure 3.2.

Exceptions are allowed in up to 3 bands of 1 MHz width centered on a frequency which is an integer multiple of 1 MHz. They shall comply with an absolute value of –20 dBm.

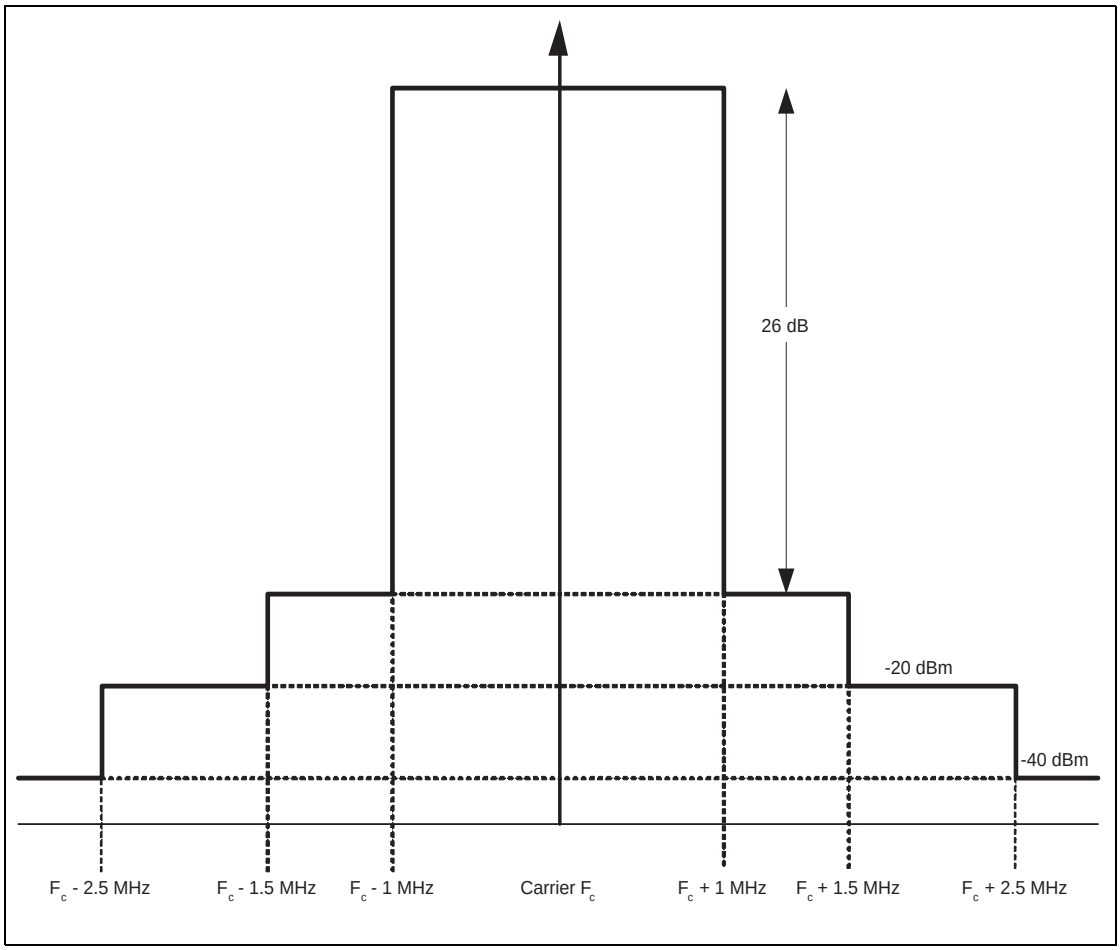


Figure 3.2: Transmitter spectrum mask

3.2.3 Radio Frequency Tolerance

The same carrier frequencies F_c as used for Basic Rate transmissions shall be used for the Enhanced Data Rate transmissions. The transmitted initial center frequency accuracy shall be within ±75 kHz from F_c. The maximum excursion from F_c (frequency offset plus drift) shall not exceed ±75 kHz.

The initial frequency accuracy is defined as being the frequency accuracy before any information is transmitted.

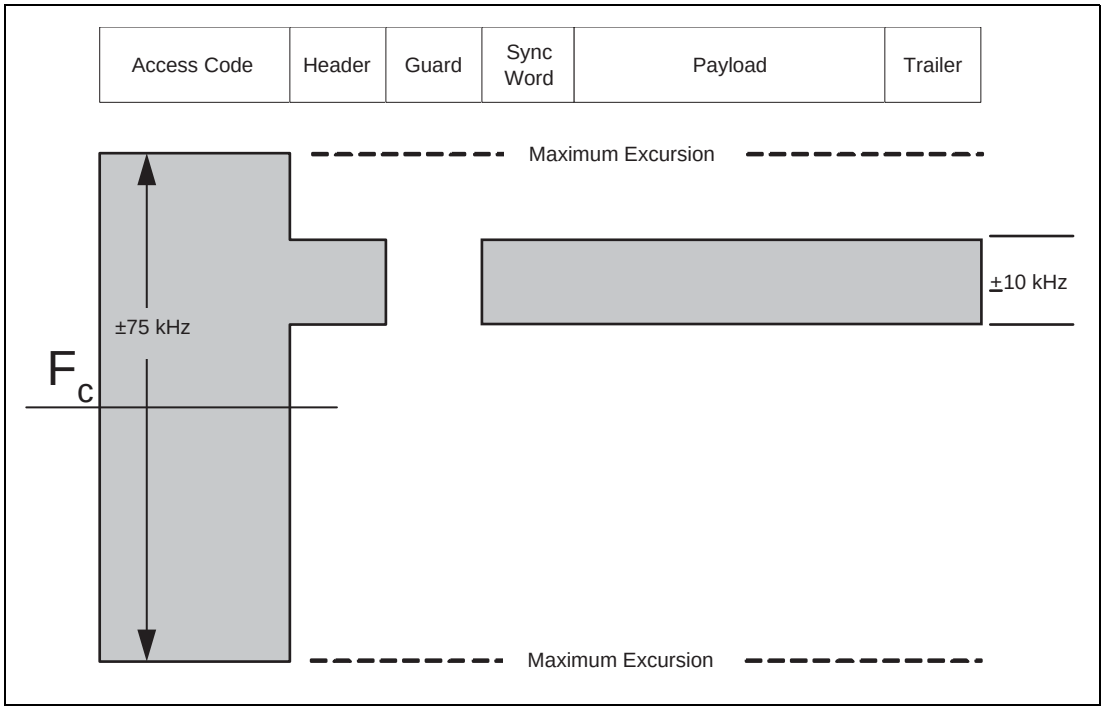


Figure 3.3: Carrier frequency mask

The requirements on accuracy and stability are illustrated by Figure 3.3 for the Enhanced Data Rate packet format defined in Baseband Specification. The higher frequency accuracy requirement shall start at the first symbol of the header. The maximum drift over the header, synchronization sequence and payload shall be ±10 kHz.

3.2.4 Relative Transmit Power

The requirement on the relative power of the GFSK and DPSK portions of the Enhanced Data Rate packet is defined as follows. The average power level during the transmission of access code and header is denoted as P_{GFSK} and the average power level during the transmission of the synchronization sequence and the payload is denoted as P_{DPSK}. The following inequalities

shall be satisfied independently for each Enhanced Data Rate packet transmitted:

$(P_{GFSK} - 4\ dB) < P_{DPSK} < (P_{GFSK} + 1\ dB)$

4 RECEIVER CHARACTERISTICS

The receiver characteristics shall be measured using loopback as defined in Section 1, “Test Methodology,” on page 411.

The reference sensitivity level referred to in this chapter is -70 dBm.

4.1 BASIC RATE

4.1.1 Actual Sensitivity Level

The actual sensitivity level is defined as the input level for which a raw bit error rate (BER) of 0.1% is met. The receiver sensitivity shall be below or equal to –70 dBm with any Bluetooth transmitter compliant to the transmitter specification specified in Section 3 on page 36.

4.1.2 Interference Performance

The interference performance on Co-channel and adjacent 1 MHz and 2 MHz shall be measured with the wanted signal 10 dB over the reference sensitivity level. For interference performance on all other RF channels the wanted signal shall be 3 dB over the reference sensitivity level. If the frequency of an interfering signal is outside of the band 2400-2483.5 MHz, the out-of-band blocking specification (see Section 4.1.3 on page 47) shall apply. The interfering signal shall be Bluetooth-modulated (see Section 4.1.7 on page 48). The BER shall be ≤0.1% for all the signal-to-interference ratios listed in Table 4.1:

Frequency of Interference	Ratio
Co-Channel interference, $C/I_{co\text{-}channel}$	11 dB
Adjacent (1 MHz) interference[1], C/I_{1MHz}	0 dB
Adjacent (2 MHz) interference[1], C/I_{2MHz}	-30 dB
Adjacent (≥3 MHz) interference[1], $C/I_{\geq 3MHz}$	-40 dB
Image frequency Interference[1 2 3], C/I_{Image}	-9 dB
Adjacent (1 MHz) interference to in-band image frequency[1], $C/I_{Image\pm 1MHz}$	-20 dB

Table 4.1: Interference performance

1. If two adjacent channel specifications from Table 4.1 are applicable to the same channel, the more relaxed specification applies.
2. In-band image frequency
3. If the image frequency ≠ n*1 MHz, then the image reference frequency is defined as the closest n*1 MHz frequency for integer n.

.

These specifications are only to be tested at nominal temperature conditions with a device receiving on one RF channel and transmitting on a second RF channel; this means that the synthesizer may change RF channels between reception and transmission, but always returns to the same receive RF channel.

RF channels where the requirements are not met are called spurious response RF channels. Five spurious response RF channels are allowed at RF channels with a distance of ≥2 MHz from the wanted signal. On these spurious response RF channels a relaxed interference requirement C/I = -17 dB shall be met.

4.1.3 Out-of-Band Blocking

The out-of-band suppression (or rejection) shall be measured with the wanted signal 3 dB over the reference sensitivity level. The interfering signal shall be a continuous wave signal. The BER shall be ≤ 0.1%. The out-of-band blocking shall fulfil the following requirements:

Interfering Signal Frequency	Interfering Signal Power Level
30 MHz - 2000 MHz	-10 dBm
2000 - 2399 MHz	-27 dBm
2484 – 3000 MHz	-27 dBm
3000 MHz – 12.75 GHz	-10 dBm

Table 4.2: Out-of-band suppression (or rejection) requirements.

24 exceptions are permitted which are dependent upon the given RF channel and are centered at a frequency which is an integer multiple of 1 MHz. For at least 19 of these spurious response frequencies, a reduced interference level of at least -50dBm is allowed in order to achieve the required BER=0.1% performance, whereas for a maximum of 5 of the spurious frequencies the interference level may be assumed arbitrarily lower.

4.1.4 Intermodulation Characteristics

The reference sensitivity performance, BER = 0.1%, shall be met under the following conditions:

- The wanted signal shall be at frequency f_0 with a power level 6 dB over the reference sensitivity level.
- A static sine wave signal shall be at a frequency f_1 with a power level of –39 dBm.
- A Bluetooth modulated signal (see Section 4.1.7 on page 48) shall be at f_2 with a power level of -39 dBm.

Frequencies f_0, f_1 and f_2 shall be chosen such that $f_0=2f_1-f_2$ and $|f_2-f_1|=n*1$ MHz, where n can be 3, 4, or 5. The system shall fulfill at least one of the three alternatives (n=3, 4, or 5).

4.1.5 Maximum Usable Level

The maximum usable input level that the receiver operates at shall be greater than -20 dBm. The BER shall be less than or equal to 0.1% at –20 dBm input power.

4.1.6 Receiver Signal Strength Indicator

If a device supports Receive Signal Strength Indicator (RSSI) the accuracy shall be +/- 6 dBm.

4.1.7 Reference Signal Definition

A Bluetooth modulated interfering signal shall be defined as:

Modulation = GFSK

Modulation index = 0.32±1%

BT= 0.5±1%

Bit Rate = 1 Mbps ±1 ppm

Modulating Data for wanted signal = PRBS9

Modulating Data for interfering signal = PRBS 15

Frequency accuracy better than ±1 ppm.

4.2 ENHANCED DATA RATE

4.2.1 Actual Sensitivity Level

The actual sensitivity level shall be defined as the input level for which a raw bit error rate (BER) of 0.01% is met. The requirement for a Bluetooth π/4-DQPSK and 8DPSK Enhanced Data Rate receiver shall be an actual sensitivity level of –70 dBm or better. The receiver shall achieve the –70 dBm sensitivity level with any Bluetooth transmitter compliant to the Enhanced Data Rate transmitter specification as defined in Section 3.2.

4.2.2 BER Floor Performance

The receiver shall achieve a BER less than 0.001% at 10 dB above the reference sensitivity level.

4.2.3 Interference Performance

The interference performance for co-channel and adjacent 1 MHz and 2 MHz channels shall be measured with the wanted signal 10 dB above the reference

sensitivity level. On all other frequencies the wanted signal shall be 3 dB above the reference sensitivity level. The requirements in this section shall only apply if the frequency of the interferer is inside of the band 2400-2483.5 MHz.

The interfering signal for co-channel interference shall be similarly modulated as the desired signal. The interfering signal for other channels shall be equivalent to a nominal Bluetooth Basic Rate GFSK transmitter. The interfering signal shall carry random data.

A BER of 0.1% or better shall be achieved for the signal to interference ratios defined in Table 4.3.

Frequency of Interference	**π/4-DQPSK Ratio**	**8DPSK Ratio**
Co-Channel interference, $C/I_{co\text{-}channel}$	13 dB	21 dB
Adjacent (1 MHz) interference[1], C/I_{1MHz}	0 dB	5 dB
Adjacent (2MHz) interference[1], C/I_{2MHz}	-30 dB	-25 dB
Adjacent (≥3MHz) interference[1]	-40 dB	-33 dB
Image frequency interference[1,2,3], C/I_{Image}	-7 dB	0 dB
Adjacent (1 MHz) interference to in-band image frequency[1,2,3], $C/I_{Image \pm 1MHz}$	-20 dB	-13 dB

Table 4.3: Interference Performance

1. If two adjacent channel specifications from Table 4.3 are applicable to the same channel, the more relaxed specification applies.
2. In-band image frequency.
3. If the image frequency is not equal to n*1 MHz, then the image reference frequency is defined as the closest n*1 MHz frequency for integer n.

These specifications are only to be tested at nominal temperature conditions with a receiver hopping on one frequency; this means that the synthesizer may change frequency between receive slot and transmit slot, but always returns to the same receive frequency.

Frequencies where the requirements are not met are called spurious response frequencies. Five spurious response frequencies are allowed at frequencies with a distance of ≥2 MHz from the wanted signal. On these spurious response frequencies a relaxed interference requirement C/I = -15 dB for π/4-DQPSK and C/I = -10 dB for 8DPSK shall be met.

4.2.4 Maximum Usable Level

The maximum usable input level that the receiver operates at shall be greater than -20 dBm. The BER shall be less than or equal to 0.1% at -20 dBm input power.

4.2.5 Out-of-Band and Intermodulation Characteristics

Note: The Basic Rate out-of-band blocking and intermodulation requirements ensure adequate Enhanced Data Rate performance, and therefore there are no specific requirements for Enhanced Data Rate.

4.2.6 Reference Signal Definition

A 2 Mbps Bluetooth signal used as "wanted" or "interfering signal" is defined as:

Modulation = π/4-DQPSK
Symbol Rate = 1 Msym/s ± 1 ppm
Frequency accuracy better than ±1 ppm
Modulating Data for wanted signal = PRBS9
Modulating Data for interfering signal = PRBS15
RMS Differential Error Vector Magnitude < 5%
Average power over the GFSK and DPSK portions of the packet shall be equal to within +/- 1 dB

A 3 Mbps Bluetooth signal used as "wanted" or "interfering signal" is defined as:

Modulation = 8DPSK
Symbol Rate = 1 Msym/s ± 1 ppm
Frequency accuracy better than ±1 ppm
Modulating Data for wanted signal = PRBS9
Modulating Data for interfering signal = PRBS15
RMS Differential Error Vector Magnitude < 5%
Average power over the GFSK and DPSK portions of the packet shall be equal to within +/- 1 dB

5 APPENDIX A

5.1 NOMINAL TEST CONDITIONS

5.1.1 Nominal temperature

The nominal temperature conditions for tests shall be +15 to +35 ^{o}C. When it is impractical to carry out the test under this condition a note to this effect, stating the ambient temperature, shall be recorded. The actual value during the test shall be recorded in the test report.

5.1.2 Nominal power source

5.1.2.1 Mains voltage

The nominal test voltage for equipment to be connected to the mains shall be the nominal mains voltage. The nominal voltage shall be the declared voltage or any of the declared voltages for which the equipment was designed. The frequency of the test power source corresponding to the AC mains shall be within 2% of the nominal frequency.

5.1.2.2 Lead-acid battery power sources used in vehicles

When radio equipment is intended for operation from the alternator-fed lead-acid battery power sources which are standard in vehicles, then the nominal test voltage shall be 1.1 times the nominal voltage of the battery (6V, 12V, etc.).

5.1.2.3 Other power sources

For operation from other power sources or types of battery (primary or secondary), the nominal test voltage shall be as declared by the equipment manufacturer. This shall be recorded in the test report.

5.2 EXTREME TEST CONDITIONS

5.2.1 Extreme temperatures

The extreme temperature range shall be the largest temperature range given by the combination of:

- The minimum temperature range 0 °C to +35 °C
- The product operating temperature range declared by the manufacturer.

This extreme temperature range and the declared operating temperature range shall be recorded in the test report.

5.2.2 Extreme power source voltages

Tests at extreme power source voltages specified below are not required when the equipment under test is designed for operation as part of and powered by another system or piece of equipment. Where this is the case, the limit values of the host system or host equipment shall apply. The appropriate limit values shall be declared by the manufacturer and recorded in the test report.

5.2.2.1 Mains voltage

The extreme test voltage for equipment to be connected to an AC mains source shall be the nominal mains voltage ±10%.

5.2.2.2 Lead-acid battery power source used on vehicles

When radio equipment is intended for operation from the alternator-fed lead-acid battery power sources which are standard in vehicles, then extreme test voltage shall be 1.3 and 0.9 times the nominal voltage of the battery (6V, 12V etc.)

5.2.2.3 Power sources using other types of batteries

The lower extreme test voltage for equipment with power sources using the following types of battery, shall be

a) for Leclanché, alkaline, or lithium type battery: 0.85 times the nominal voltage of the battery

b) for mercury or nickel-cadmium types of battery: 0.9 times the nominal voltage of the battery.

In both cases, the upper extreme test voltage shall be 1.15 times the nominal voltage of the battery.

5.2.2.4 Other power sources

For equipment using other power sources, or capable of being operated from a variety of power sources (primary or secondary), the extreme test voltages shall be those declared by the manufacturer. These shall be recorded in the test report.

6 APPENDIX B

The Basic Rate radio parameters shall be tested in the following conditions:

Parameter	Temperature	Power source
Output Power	ETC	ETC
Power control	NTC	NTC
Modulation index	ETC	ETC
Initial Carrier Frequency accuracy	ETC	ETC
Carrier Frequency drift	ETC	ETC
Conducted in-band spurious emissions	ETC	ETC
Radiated in-band emissions	NTC	NTC
Sensitivity	ETC	ETC
Interference Performance	NTC	NTC
Intermodulation Characteristics	NTC	NTC
Out-of-band blocking	NTC	NTC
Maximum Usable Level	NTC	NTC
Receiver Signal Strength Indicator	NTC	NTC

ETC = Extreme Test Conditions
NTC = Nominal Test Conditions

The Enhanced Data Rate radio parameters shall be tested in the following conditions:

Parameter	Temperature	Power source
Modulation accuracy	ETC	ETC
Carrier frequency stability	ETC	ETC
In-band spurious emissions	ETC	ETC
Relative transmit power	ETC	ETC
Sensitivity	ETC	ETC
BER floor sensitivity	NTC	NTC
Interference Performance	NTC	NTC
Maximum usable level	NTC	NTC

ETC = Extreme Test Conditions
NTC = Nominal Test Conditions

7 APPENDIX C

7.1 ENHANCED DATA RATE MODULATION ACCURACY

The Enhanced Data Rate modulation accuracy is defined by the differential error vector, being the difference between the vectors representing consecutive symbols of the transmitted signal, after passing the signal through a specified measurement filter, sampling it at the symbol rate with an optimum sampling phase and compensating it for carrier frequency error and for the ideal carrier phase changes. The magnitude of the normalized differential error vector is called the Differential Error Vector Magnitude (DEVM). The objective of the DEVM is to estimate the modulation errors that would be perceived by a differential receiver.

In an ideal transmitter, the input bit sequence $\{b_j\}$ is mapped onto a complex valued symbol sequence $\{S_k\}$. Subsequently, this symbol sequence is transformed into a baseband signal S(t) by means of a pulse-shaping filter.

In an actual transmitter implementation, the bit sequence $\{b_j\}$ generates a baseband equivalent transmitted signal Y(t). This signal Y(t) contains, besides the desired component S(t), multiple distortion components. This is illustrated in Figure 7.1.

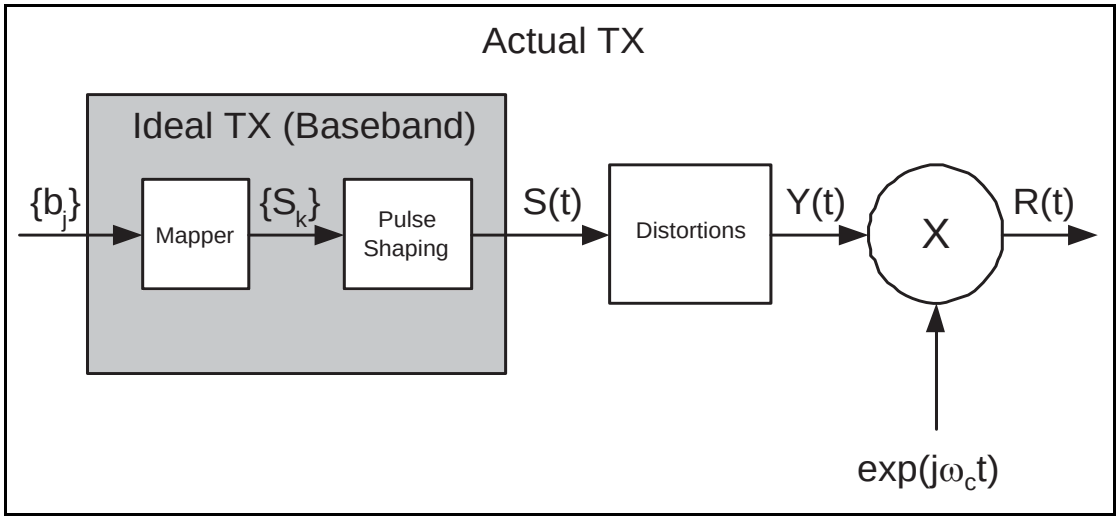


Figure 7.1: TX model.

Let Z(t) be the output of the measurement filter after first compensating the received signal for the initial center frequency error, ω_i, of the received packet, i.e. the output after down conversion and filtering the transmit signal R(t) (see Figure 7.2).The measurement filter is defined by a square-root raised cosine shaping filter with a roll-off factor equal to 0.4 and 3 dB bandwidth of ±500 kHz.

Let $\{Z_k(\varepsilon)\}$ be the sequence of samples obtained by sampling the signal Z(t) with a sampling period equal to the symbol period T and a sampling phase equal to ε such that $Z_k(\varepsilon)=Z((k+\varepsilon)T)$. Note that this sequence $\{Z_k(\varepsilon)\}$ would coincide with the symbol sequence $\{S_k\}$ if no distortion is present and the correct timing phase ε is chosen.

To reflect the behavior of a typical differential receiver, the sample sequence $\{Z_k(\varepsilon)\}$ should be compensated for carrier frequency drift. Therefore, the sequence $\{Z_k(\varepsilon)\}$ is multiplied by a factor W^{-k} in which $W = e^{j\omega T}$ accounts for the frequency offset ω. A constant value of ω is used for each DEVM block of N = 50 symbols, but ω may vary between DEVM blocks (note that the values of ω can be used to measure carrier frequency drift.

In addition, $\{Z_k(\varepsilon)\}$ is compensated for the ideal phase changes between symbols by multiplying it with the complex conjugate of the symbol sequence $\{S_k\}$. However, it is not necessary to compensate $\{Z_k(\varepsilon)\}$ for initial carrier phase or output power of the transmitter.

Let $\{Q_k(\varepsilon,\omega)\}$ denote the compensated sequence $\{Z_k(\varepsilon)\}$, where the ideal phase changes have been removed and ε and ω are chosen optimally to minimize the DEVM, (i.e. remove time and frequency uncertainty). For a transmitter with no distortions other than a constant frequency error, $\{Q_k(\varepsilon,\omega)\}$ is a complex constant that depends on the initial carrier phase and the output power of the transmitter.

The differential error sequence $\{E_k(\varepsilon,\omega)\}$ is defined as the difference between $\{Q_k(\varepsilon,\omega)\}$ and $\{Q_{k-1}(\varepsilon,\omega)\}$. This reflects the modulation errors that would be perceived by a differential receiver. For a transmitter with no distortions other than a constant frequency error, $\{E_k(\varepsilon,\omega)\}$ is zero.

The definitions of the DEVM measures are based upon this differential error sequence $\{E_k(\varepsilon,\omega)\}$. The generation of the error sequence is depicted in Figure 7.2.

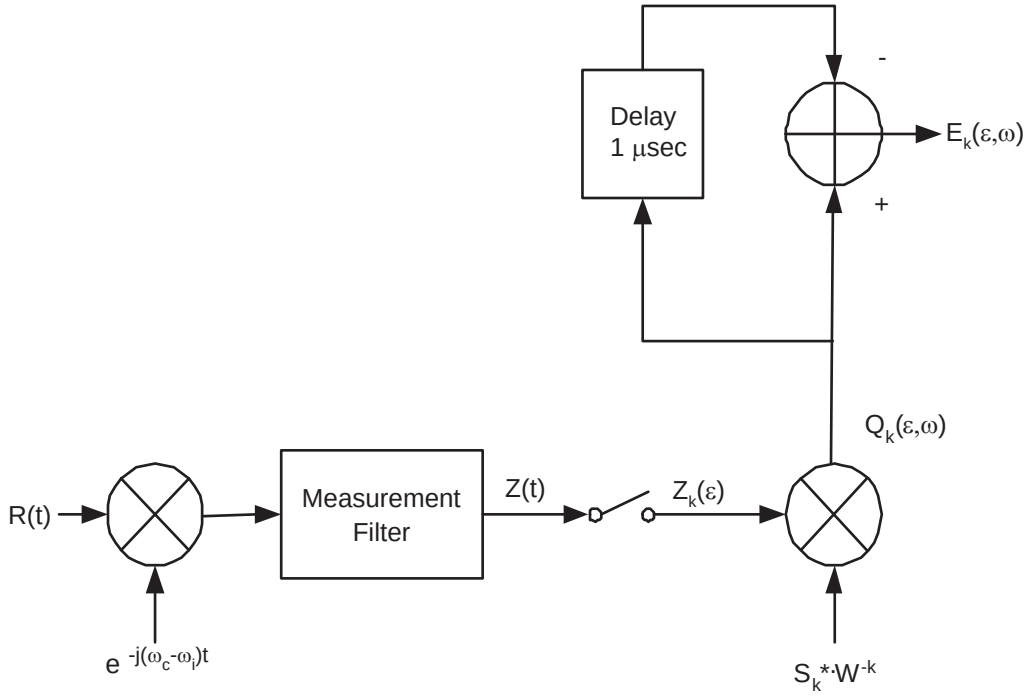


Figure 7.2: Error sequence generation.

RMS DEVM

The root mean squared DEVM (RMS DEVM) computed over N = 50 symbols is defined as:

$$RMS\ DEVM = \min_{\varepsilon,\omega} \left\{ \sqrt{\sum_{k=1}^{N} \left|E_k(\varepsilon,\omega)\right|^2 \Big/ \sum_{k=1}^{N} \left|Q_k(\varepsilon,\omega)\right|^2} \right\}$$

As can be seen from the expression above, the RMS DEVM is the square-root of the normalized power of the error.

Peak DEVM

The DEVM at symbol k is defined as

$$DEVM\,(k) = \sqrt{\frac{\left|E_k\,(\varepsilon_0, \omega_0)\right|^2}{\sum_{j=1}^{N}\left|Q_j\,(\varepsilon_0, \omega_0)\right|^2 / N}} \qquad \text{(EQ 6)}$$

where ε_0 and ω_0 are the values for ε and ω used to calculate the RMS DEVM.

The peak DEVM is defined as:

$$Peak\ DEVM = \max_{k}\{\mathrm{DEVM\ (k)}\} \qquad \text{(EQ 7)}$$

BASEBAND SPECIFICATION

This document describes the specification of the Bluetooth link controller which carries out the baseband protocols and other low-level link routines.

Ford Ex. 1052

CONTENTS

1 General Description 65
1.1 Bluetooth Clock 66
1.2 Bluetooth Device Addressing 68
1.2.1 Reserved Addresses 68
1.3 Access Codes 69
2 Physical Channels 70
2.1 Physical Channel Definition 71
2.2 Basic Piconet Physical Channel 71
2.2.1 Master-slave Definition 71
2.2.2 Hopping Characteristics 72
2.2.3 Time Slots 72
2.2.4 Piconet Clocks 73
2.2.5 Transmit/receive Timing 73
2.3 Adapted Piconet Physical Channel 76
2.3.1 Hopping Characteristics 76
2.4 Page Scan Physical Channel 77
2.4.1 Clock Estimate for Paging 77
2.4.2 Hopping Characteristics 77
2.4.3 Paging Procedure Timing 78
2.4.4 Page Response Timing 79
2.5 Inquiry Scan Physical Channel 81
2.5.1 Clock for Inquiry 81
2.5.2 Hopping Characteristics 81
2.5.3 Inquiry Procedure Timing 81
2.5.4 Inquiry Response Timing 81
2.6 Hop Selection 83
2.6.1 General Selection Scheme 83
2.6.2 Selection Kernel 87
2.6.3 Adapted Hop Selection Kernel 90
2.6.4 Control Word 91
3 Physical Links 96
3.1 Link Supervision 96
4 Logical Transports 97
4.1 General 97
4.2 Logical Transport Address (LT_ADDR) 97
4.3 Synchronous Logical Transports 98
4.4 Asynchronous Logical Transport 98

4.5 Transmit/Receive Routines 99
4.5.1 TX Routine 99
4.5.2 RX Routine 102
4.5.3 Flow Control 103
4.6 Active Slave Broadcast Transport 104
4.7 Parked Slave Broadcast Transport 105
4.7.1 Parked Member Address (PM_ADDR) 105
4.7.2 Access Request Address (AR_ADDR) 105

5 Logical Links 106
5.1 Link Control Logical Link (LC) 106
5.2 ACL Control Logical Link (ACL-C) 106
5.3 User Asynchronous/Isochronous Logical Link (ACL-U) 106
5.3.1 Pausing the ACL-U logical link 107
5.4 User Synchronous Data Logical Link (SCO-S) 107
5.5 User Extended Synchronous Data Logical Link (eSCO-S) 107
5.6 Logical Link Priorities 107

6 Packets 108
6.1 General Format 108
6.1.1 Basic Rate 108
6.1.2 Enhanced Data Rate 108
6.2 Bit Ordering 109
6.3 Access Code 110
6.3.1 Access Code Types 110
6.3.2 Preamble 111
6.3.3 Sync Word 111
6.3.4 Trailer 114
6.4 Packet Header 115
6.4.1 LT_ADDR 115
6.4.2 TYPE 115
6.4.3 FLOW 116
6.4.4 ARQN 116
6.4.5 SEQN 116
6.4.6 HEC 116
6.5 Packet Types 117
6.5.1 Common Packet Types 118
6.5.2 SCO Packets 122
6.5.3 eSCO Packets 123
6.5.4 ACL Packets 125
6.6 Payload Format 127

6.6.1 Synchronous Data Field 127
6.6.2 Asynchronous Data Field 129
6.7 Packet Summary 132
7 Bitstream Processing 134
7.1 Error Checking 135
7.1.1 HEC Generation 135
7.1.2 CRC Generation 136
7.2 Data Whitening 138
7.3 Error Correction 139
7.4 FEC Code: Rate 1/3 139
7.5 FEC Code: Rate 2/3 140
7.6 ARQ Scheme 141
7.6.1 Unnumbered ARQ 141
7.6.2 Retransmit Filtering 144
7.6.3 Flushing Payloads 147
7.6.4 Multi-slave Considerations 147
7.6.5 Broadcast Packets 148
7.7 Erroneous Synchronous Data Reporting 149
8 Link Controller Operation 150
8.1 Overview of States 150
8.2 Standby State 151
8.3 Connection Establishment Substates 151
8.3.1 Page Scan Substate 151
8.3.2 Page substate 153
8.3.3 Page Response Substates 156
8.4 Device Discovery Substates 160
8.4.1 Inquiry scan substate 161
8.4.2 Inquiry Substate 162
8.4.3 Inquiry Response Substate 163
8.5 Connection State 165
8.6 Active Mode 166
8.6.1 Polling in the Active Mode 167
8.6.2 SCO 167
8.6.3 eSCO 169
8.6.4 Broadcast Scheme 171
8.6.5 Role Switch 172
8.6.6 Scatternet 174
8.6.7 Hop Sequence Switching 175
8.6.8 Channel Classification and Channel Map Selection 178

8.6.9 Power Management 179
8.7 Sniff Mode 180
8.7.1 Sniff Transition Mode 181
8.7.2 Sniff Subrating 182
8.8 Hold Mode 183
8.9 Park State 184
8.9.1 Beacon Train 184
8.9.2 Beacon Access Window 187
8.9.3 Parked Slave Synchronization 188
8.9.4 Parking 189
8.9.5 Master-initiated Unparking 190
8.9.6 Slave-initiated Unparking 190
8.9.7 Broadcast Scan Window 191
8.9.8 Polling in the Park State 191

9 Audio 192
9.1 LOG PCM CODEC 192
9.2 CVSD CODEC 192
9.3 Error Handling 195
9.4 General Audio Requirements 195
9.4.1 Signal Levels 195
9.4.2 CVSD Audio Quality 195

10 List of Figures 196

11 List of Tables 199
Appendix A: General Audio Recommendations 200
Appendix B: Timers 203
Appendix C: Recommendations for AFH Operation in Park, Hold, and Sniff 205

1 GENERAL DESCRIPTION

This part specifies the normal operation of a Bluetooth baseband.

The Bluetooth system provides a point-to-point connection or a point-to-multi-point connection, see (a) and (b) in Figure 1.1 on page 65. In a point-to-point connection the physical channel is shared between two Bluetooth devices. In a point-to-multipoint connection, the physical channel is shared among several Bluetooth devices. Two or more devices sharing the same physical channel form a *piconet*. One Bluetooth device acts as the master of the piconet, whereas the other device(s) act as slave(s). Up to seven slaves can be active in the piconet. Additionally, many more slaves can remain connected in a parked state. These parked slaves are not active on the channel, but remain synchronized to the master and can become active without using the connection establishment procedure. Both for active and parked slaves, the channel access is controlled by the master.

Piconets that have common devices are called a *scatternet,* see (c) in Figure 1.1 on page 65. Each piconet only has a single master, however, slaves can participate in different piconets on a time-division multiplex basis. In addition, a master in one piconet can be a slave in other piconets. Piconets shall not be frequency synchronized and each piconet has its own hopping sequence.

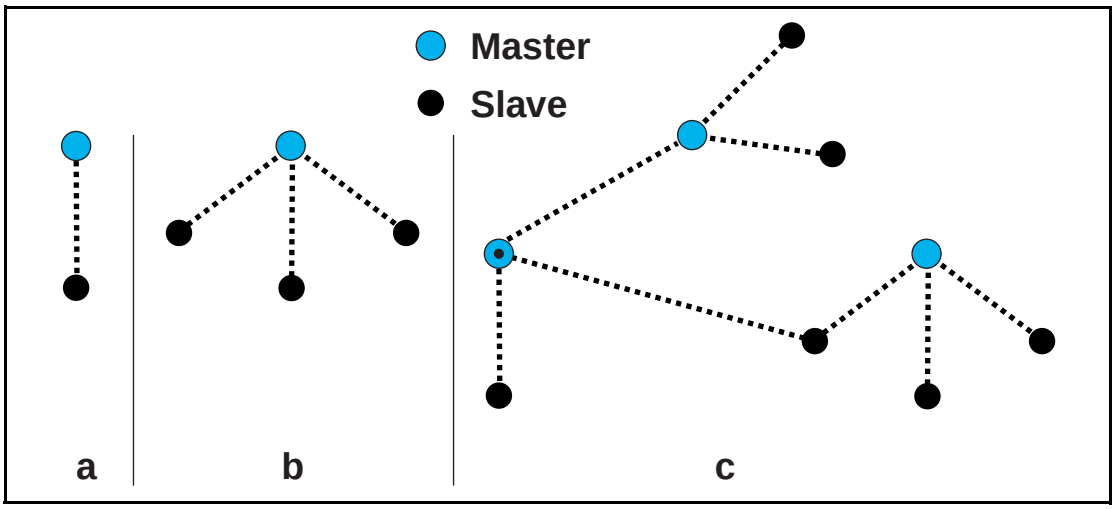


Figure 1.1: Piconets with a single slave operation (a), a multi-slave operation (b) and a scatternet operation (c).

Data is transmitted over the air in packets. Two modes are defined: a mandatory mode called Basic Rate and an optional mode called Enhanced Data Rate. The symbol rate for all modulation modes is 1 Ms/s. The gross air data rate is 1 Mbps for Basic Rate. Enhanced Data Rate has a primary modulation mode that provides a gross air data rate of 2 Mbps, and a secondary modulation mode that provides a gross air data rate of 3 Mbps.

The general Basic Rate packet format is shown in Figure 1.2. Each packet consists of 3 entities: the access code, the header, and the payload.

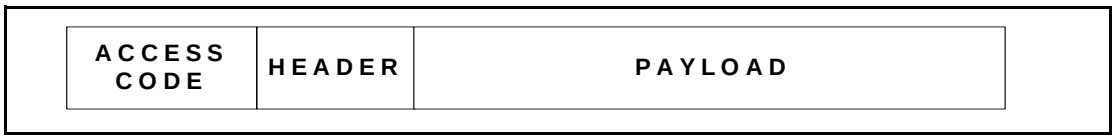


Figure 1.2: Standard Basic Rate packet format.

The general Enhanced Data Rate packet format is shown in Figure 1.3. Each packet consists of 6 entities: the access code, the header, the guard period, the synchronization sequence, the Enhanced Data Rate payload and the trailer. The access code and header use the same modulation mode as for Basic Rate packets while the synchronization sequence, the Enhanced Data Rate payload and the trailer use the Enhanced Data Rate modulation mode. The guard time allows for the transition between the modulation modes.

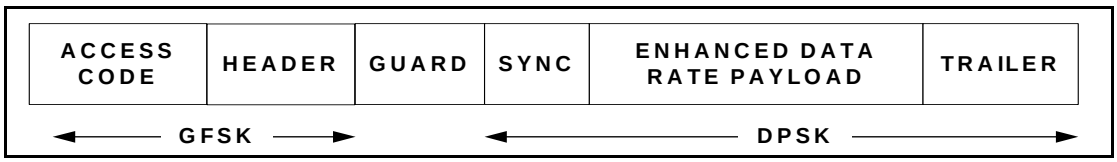


Figure 1.3: Standard Enhanced Data Rate packet format

1.1 BLUETOOTH CLOCK

Every Bluetooth device shall have a native clock that shall be derived from a free running system clock. For synchronization with other devices, offsets are used that, when added to the native clock, provide temporary Bluetooth clocks that are mutually synchronized. It should be noted that the Bluetooth clock has no relation to the time of day; it may therefore be initialized to any value. The clock has a cycle of about a day. If the clock is implemented with a counter, a 28-bit counter is required that shall wrap around at 2^{28}-1. The least significant bit (LSB) shall tick in units of 312.5 μs (i.e. half a time slot), giving a clock rate of 3.2 kHz.

The clock determines critical periods and triggers the events in the device. Four periods are important in the Bluetooth system: 312.5 μs, 625 μs, 1.25 ms, and 1.28 s; these periods correspond to the timer bits CLK_0, CLK_1, CLK_2, and CLK_{12}, respectively, see Figure 1.4 on page 66.

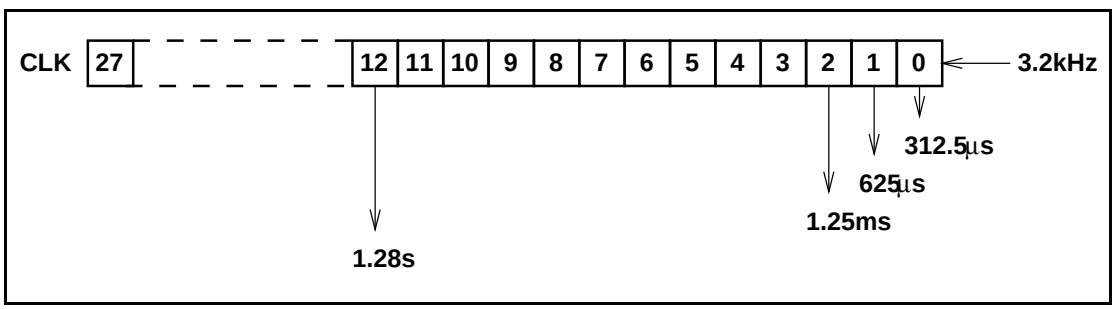


Figure 1.4: Bluetooth clock.

In the different modes and states a device can reside in, the clock has different appearances:

- CLKN native clock
- CLKE estimated clock
- CLK master clock

CLKN is the native clock and shall be the reference to all other clock appearances. In STANDBY and in Park, Hold, and Sniff modes the native clock may be driven by a low power oscillator (LPO) with worst case accuracy (+/-250ppm). Otherwise, the native clock shall be driven by the reference crystal oscillator with worst case accuracy of +/-20ppm.

See Section 2.2.4 on page 73 for the definition of CLK and Section 2.4.1 on page 77 for the definition of CLKE.

The master shall never adjust its native clock during the existence of the piconet.

1.2 BLUETOOTH DEVICE ADDRESSING

Each Bluetooth device shall be allocated a unique 48-bit Bluetooth device address (BD_ADDR). This address shall be obtained from the IEEE Registration Authority. The address shall be created in accordance with section 9.2 ("48-bit universal LAN MAC addresses") of the IEEE 802-2001 standard (http://standards.ieee.org/getieee802/download/802-2001.pdf) and using a valid Organizationally Unique Identifier (OUI) obtained from the IEEE Registration Authority (see http://standards.ieee.org/regauth/oui/forms/ and sections 9 and 9.1 of the IEEE 802-2001 specification).The LAP and UAP form the significant part of the BD_ADDR. The bit pattern in Figure 1.5 is an example BD_ADDR.

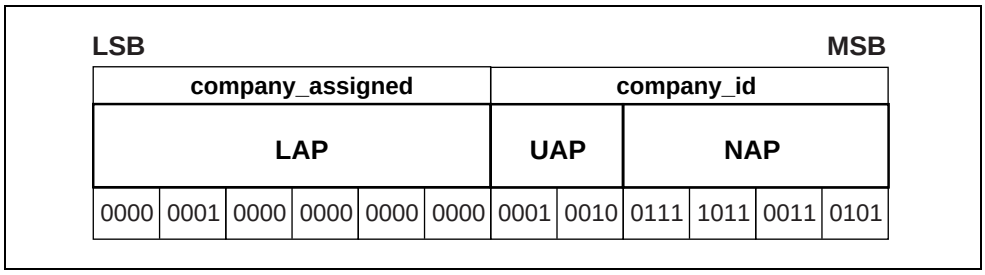


Figure 1.5: Format of BD_ADDR.

The BD_ADDR may take any values except the 64 reserved LAP values for general and dedicated inquiries (see Section 1.2.1 on page 68).

1.2.1 Reserved Addresses

A block of 64 contiguous LAPs is reserved for inquiry operations; one LAP common to all devices is reserved for general inquiry, the remaining 63 LAPs are reserved for dedicated inquiry of specific classes of devices (see Assigned Numbers). The same LAP values are used regardless of the contents of UAP and NAP. Consequently, none of these LAPs can be part of a user BD_ADDR.

The reserved LAP addresses are 0x9E8B00-0x9E8B3F. The general inquiry LAP is 0x9E8B33. All addresses have the LSB at the rightmost position, hexadecimal notation. The default check initialization (DCI) is used as the UAP whenever one of the reserved LAP addresses is used. The DCI is defined to be 0x00 (hexadecimal).

1.3 ACCESS CODES

In the Bluetooth system all transmissions over the physical channel begin with an access code. Three different access codes are defined, see also Section 6.3.1 on page 110:

- device access code (DAC)
- channel access code (CAC)
- inquiry access code (IAC)

All access codes are derived from the LAP of a device address or an inquiry address. The device access code is used during **page**, **page scan** and **page response** substates and shall be derived from the paged device's BD_ADDR. The channel access code is used in the **CONNECTION** state and forms the beginning of all packets exchanged on the piconet physical channel. The channel access code shall be derived from the LAP of the master's BD_ADDR. Finally, the inquiry access code shall be used in the **inquiry** substate. There is one general IAC (GIAC) for general inquiry operations and there are 63 dedicated IACs (DIACs) for dedicated inquiry operations.

The access code also indicates to the receiver the arrival of a packet. It is used for timing synchronization and offset compensation. The receiver correlates against the entire synchronization word in the access code, providing very robust signaling.

2 PHYSICAL CHANNELS

The lowest architectural layer in the Bluetooth system is the physical channel. A number of types of physical channels are defined. All Bluetooth physical channels are characterized by the combination of a pseudo-random frequency hopping sequence, the specific slot timing of the transmissions, the access code and packet header encoding. These aspects, together with the range of the transmitters, define the signature of the physical channel. For the basic and adapted piconet physical channels frequency hopping is used to change frequency periodically to reduce the effects of interference and to satisfy local regulatory requirements.

Two devices that wish to communicate use a shared physical channel for this communication. To achieve this, their transceivers must be tuned to the same RF frequency at the same time, and they must be within a nominal range of each other.

Given that the number of RF carriers is limited and that many Bluetooth devices could be operating independently within the same spatial and temporal area there is a strong likelihood of two independent Bluetooth devices having their transceivers tuned to the same RF carrier, resulting in a physical channel collision. To mitigate the unwanted effects of this collision each transmission on a physical channel starts with an access code that is used as a correlation code by devices tuned to the physical channel. This channel access code is a property of the physical channel. The access code is always present at the start of every transmitted packet.

Four Bluetooth physical channels are defined. Each is optimized and used for a different purpose. Two of these physical channels (the basic piconet channel and adapted piconet channel) are used for communication between connected devices and are associated with a specific piconet. The remaining physical channels are used for discovering (the inquiry scan channel) and connecting (the page scan channel) Bluetooth devices.

A Bluetooth device can only use one of these physical channels at any given time. In order to support multiple concurrent operations the device uses time-division multiplexing between the channels. In this way a Bluetooth device can appear to operate simultaneously in several piconets, as well as being discoverable and connectable.

Whenever a Bluetooth device is synchronized to the timing, frequency and access code of a physical channel it is said to be 'connected' to this channel (whether or not it is actively involved in communications over the channel.) At a minimum, a device need only be capable of connection to one physical channel at a time, however, advanced devices may be capable of connecting simultaneously to more than one physical channel, but the specification does not assume that this is possible.

2.1 PHYSICAL CHANNEL DEFINITION

Physical channels are defined by a pseudo-random RF channel hopping sequence, the packet (slot) timing and an access code. The hopping sequence is determined by the UAP and LAP of a Bluetooth device address and the selected hopping sequence. The phase in the hopping sequence is determined by the Bluetooth clock. All physical channels are subdivided into time slots whose length is different depending on the physical channel. Within the physical channel, each reception or transmission event is associated with a time slot or time slots. For each reception or transmission event an RF channel is selected by the hop selection kernel (see Section 2.6 on page 83).The maximum hop rate is 1600 hops/s in the **CONNECTION** state and the maximum is 3200 hops/s in the **inquiry** and **page** substates.

The following physical channels are defined:

- basic piconet physical channel
- adapted piconet physical channel
- page scan physical channel
- inquiry scan physical channel

2.2 BASIC PICONET PHYSICAL CHANNEL

During the **CONNECTION** state the basic piconet physical channel is used by default. The adapted piconet physical channel may also be used. The adapted piconet physical channel is identical to the basic piconet physical channel except for the differences listed in Section 2.3 on page 76.

2.2.1 Master-slave Definition

The basic piconet physical channel is defined by the master of the piconet. The master controls the traffic on the piconet physical channel by a polling scheme (see Section 8.5 on page 165).

By definition, the device that initiates a connection by paging is the master. Once a piconet has been established, master-slave roles may be exchanged. This is described in Section 8.6.5 on page 172.

2.2.2 Hopping Characteristics

The basic piconet physical channel is characterized by a pseudo-random hopping through all 79 RF channels. The frequency hopping in the piconet physical channel is determined by the Bluetooth clock and BD_ADDR of the master. When the piconet is established, the master clock is communicated to the slaves. Each slave shall add an offset to its native clock to synchronize with the master clock. Since the clocks are independent, the offsets must be updated regularly. All devices participating in the piconet are time-synchronized and hop-synchronized to the channel.

The basic piconet physical channel uses the basic channel hopping sequence and is described in Section 2.6 on page 83.

2.2.3 Time Slots

The basic piconet physical channel is divided into time slots, each 625 μs in length. The time slots are numbered according to the most significant 27 bits of the Bluetooth clock CLK_{28-1} of the piconet master. The slot numbering ranges from 0 to 2^{27}-1 and is cyclic with a cycle length of 2^{27}. The time slot number is denoted as k.

A TDD scheme is used where master and slave alternately transmit, see Figure 2.1 on page 72. The packet start shall be aligned with the slot start. Packets may extend over up to five time slots.

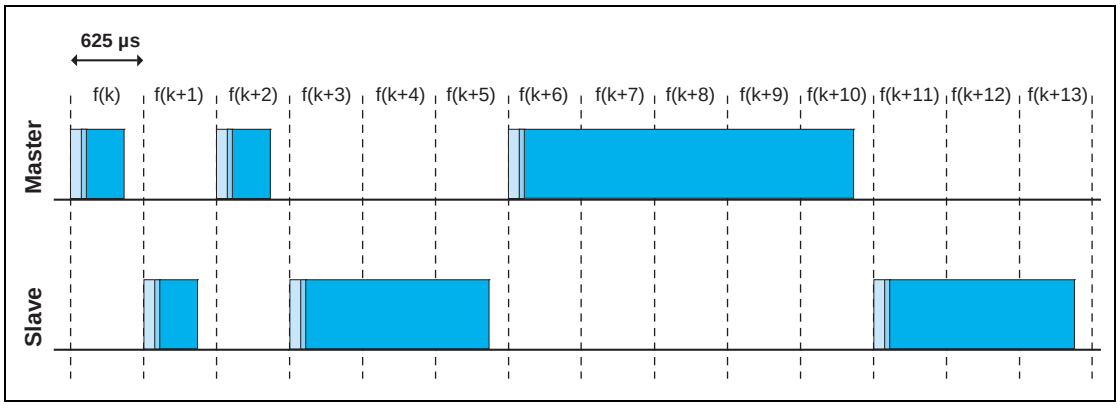


Figure 2.1: Multi-slot packets

The term *slot pairs* is used to indicate two adjacent time slots starting with a master-to-slave transmission slot.

2.2.4 Piconet Clocks

CLK is the master clock of the piconet. It shall be used for all timing and scheduling activities in the piconet. All devices shall use the CLK to schedule their transmission and reception. The CLK shall be derived from the native clock CLKN (see Section 1.1 on page 66) by adding an offset, see Figure 2.2 on page 73. The offset shall be zero for the master since CLK is identical to its own native clock CLKN. Each slave shall add an appropriate offset to its CLKN such that the CLK corresponds to the CLKN of the master. Although all CLKNs in the devices run at the same nominal rate, mutual drift causes inaccuracies in CLK. Therefore, the offsets in the slaves must be regularly updated such that CLK is approximately CLKN of the master.

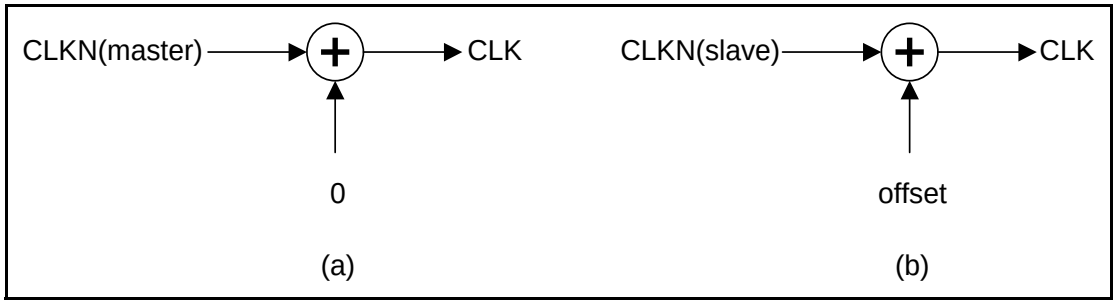


Figure 2.2: Derivation of CLK in master (a) and in slave (b).

2.2.5 Transmit/receive Timing

The master transmission shall always start at even numbered time slots (CLK_1=0) and the slave transmission shall always start at odd numbered time slots (CLK_1=1). Due to packet types that cover more than a single slot, master transmission may continue in odd numbered slots and slave transmission may continue in even numbered slots, see Figure 2.1 on page 72.

All timing diagrams shown in this chapter are based on the signals as present at the antenna. The term “exact” when used to describe timing refers to an ideal transmission or reception and neglects timing jitter and clock frequency imperfections.

The average timing of packet transmission shall not drift faster than 20 ppm relative to the ideal slot timing of 625 µs. The instantaneous timing shall not deviate more than 1 µs from the average timing. Thus, the absolute packet transmission timing t_k of slot boundary k shall fulfill the equation:

$$t_k = \left(\sum_{i=1}^{k} (1 + d_i) T_N \right) + j_k + \text{offset}, \qquad \text{(EQ 1)}$$

where T_N is the nominal slot length (625 µs), j_k denotes jitter ($|j_k| \leq 1$ µs) at the start of slot k, and, d_k, denotes the drift ($|d_k| \leq 20$ ppm) within slot k. The jitter and drift can vary arbitrarily within the given limits for every slot, while *offset* is an

arbitrary but fixed constant. For hold, Park, and sniff the drift and jitter parameters specified in Link Manager Protocol [Part C] Section 4.3.1 on page 275 apply.

2.2.5.1 Piconet physical channel timing

In the figures, only single-slot packets are shown as an example.

The master TX/RX timing is shown in Figure 2.3 on page 74. In Figure 2.3 and Figure 2.4 the channel hopping frequencies are indicated by *f(k)* where k is the time slot number. After transmission, a return packet is expected N × 625 µs after the start of the TX packet where N is an odd, integer larger than 0. N depends on the type of the transmitted packet.

To allow for some time slipping, an uncertainty window is defined around the exact receive timing. During normal operation, the window length shall be 20 µs, which allows the RX packet to arrive up to 10 µs too early or 10 µs too late. It is recommended that slaves implement variable sized windows or time tracking to accommodate a master's absence of more than 250ms.

During the beginning of the RX cycle, the access correlator shall search for the correct channel access code over the uncertainty window. If an event trigger does not occur the receiver may go to sleep until the next RX event. If in the course of the search, it becomes apparent that the correlation output will never exceed the final threshold, the receiver may go to sleep earlier. If a trigger event occurs, the receiver shall remain open to receive the rest of the packet unless the packet is for another device, a non-recoverable header error is detected, or a non-recoverable payload error is detected.

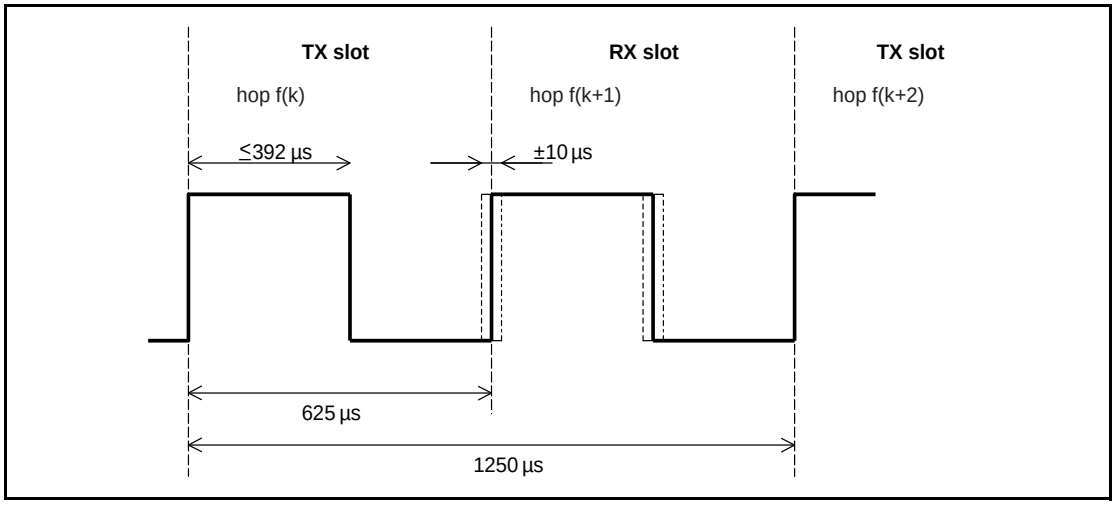


Figure 2.3: RX/TX cycle of master transceiver in normal mode for single-slot packets.

Each master transmission shall be derived from bit 2 of the Master's native Bluetooth clock, thus the current transmission will be scheduled Mx1250µs after the start of the previous master TX burst where M depends on the transmitted and received packet type and is an even, integer larger than 0. The

master TX timing shall be derived from the master's native Bluetooth clock, and thus it will not be affected by time drifts in the slave(s).

Slaves maintain an estimate of the master's native clock by adding a timing offset to the slave's native clock (see Section 2.2.4 on page 73). This offset shall be updated each time a packet is received from the master. By comparing the exact RX timing of the received packet with the estimated RX timing, slaves shall correct the offset for any timing misalignments. Since only the channel access code is required to synchronize the slave, slave RX timing can be corrected with any packet sent in the master-to-slave transmission slot.

The slave's TX/RX timing is shown in Figure 2.4 on page 75. The slave's transmission shall be scheduled N × 625µs after the start of the slave's RX packet where N is an odd, positive integer larger than 0. If the slave's RX timing drifts, so will its TX timing. During periods when a slave is in the active mode (see Section 8.6 on page 166) and is prevented from receiving valid channel access codes from the master due to local RF interference, slave activity in a different piconet, or any other reason, the slave may increase its receive uncertainty window and/or use predicted timing drift to increase the probability of receiving the master's bursts when reception resumes.

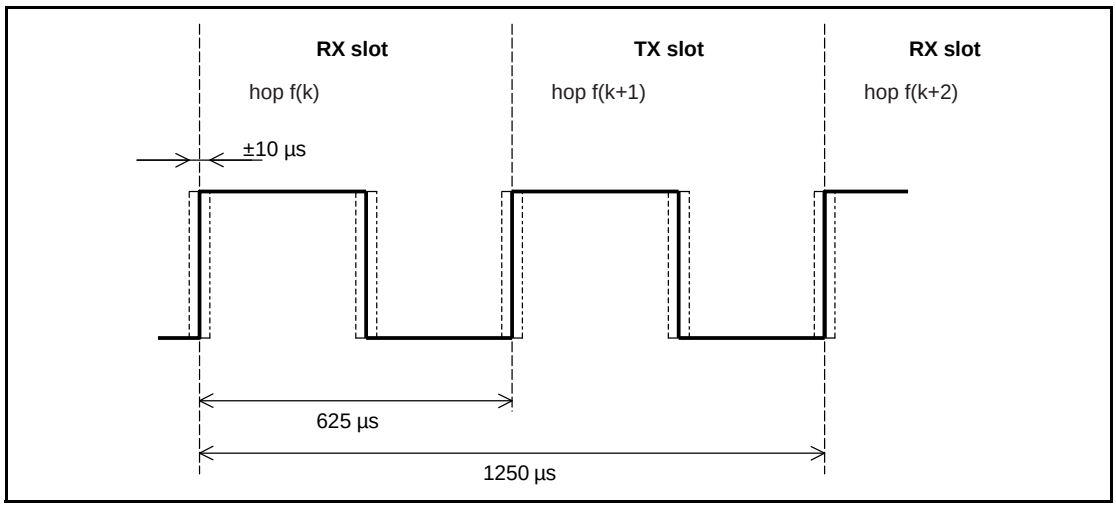


Figure 2.4: RX/TX cycle of slave transceiver in normal mode for single-slot packets.

2.2.5.2 Piconet physical channel re-synchronization

In the piconet physical channel, a slave can lose synchronization if it does not receive a packet from the master at least every 250ms (or less if the low power clock is used). This can occur in sniff, hold, and Park in a scatternet or due to interference. When re-synchronizing to the piconet physical channel a slave device shall listen for the master before it may send information. In this case, the length of the search window in the slave device may be increased from 20 µs to a larger value X µs as illustrated in Figure 2.5 on page 76. Note that only RX hop frequencies are used. The hop frequency used in the master-to-slave (RX) slot shall also be used in the uncertainty window, even when it is

extended into the preceding time interval normally used for the slave-to-master (TX) slot.

If the length of search window, X, exceeds 1250 µs, consecutive windows shall avoid overlapping search windows. Consecutive windows should instead be centered at *f(k), f(k+4),... f(k+4i)* (where 'i' is an integer), which gives a maximum value X=2500 µs, or even at *f(k), f(k+6),...f(k+6i)* which gives a maximum value X=3750 µs. The RX hop frequencies used shall correspond to the master-to-slave transmission slots.

It is recommended that single slot packets are transmitted by the master during slave re-synchronization.

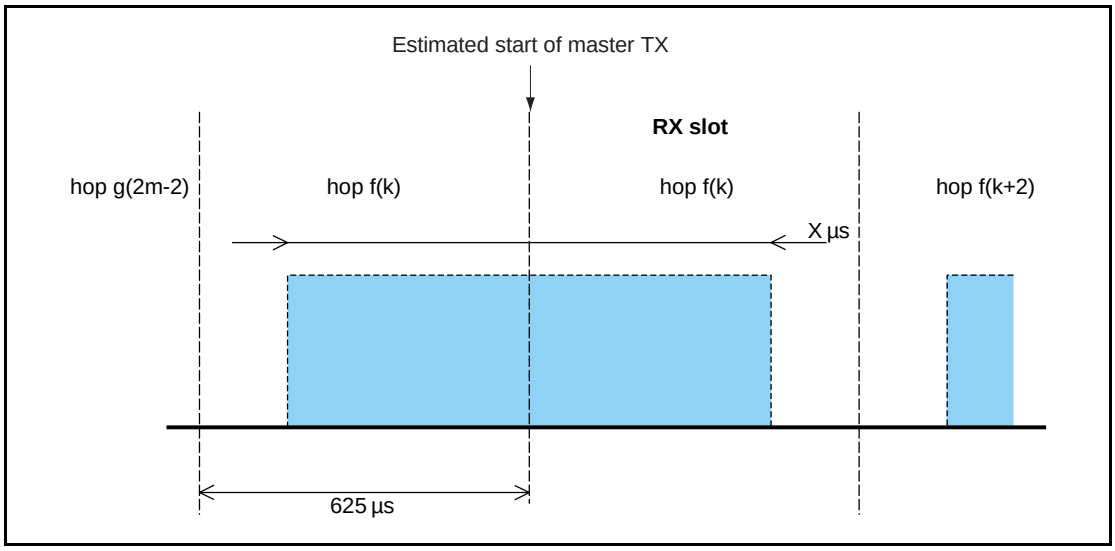


Figure 2.5: RX timing of slave returning from hold mode.

2.3 ADAPTED PICONET PHYSICAL CHANNEL

2.3.1 Hopping Characteristics

The adapted piconet physical channel shall use at least N_{min} RF channels (where N_{min} is 20).

The adapted piconet physical channel uses the adapted channel hopping sequence described in Section 2.6 on page 83.

Adapted piconet physical channels can be used for connected devices that have adaptive frequency hopping (AFH) enabled. There are two distinctions between basic and adapted piconet physical channels. The first is the same channel mechanism that makes the slave frequency the same as the preceding master transmission. The second aspect is that the adapted piconet physical channel may be based on less than the full 79 frequencies of the basic piconet physical channel.

2.4 PAGE SCAN PHYSICAL CHANNEL

Although master and slave roles are not defined prior to a connection, the term *master* is used for the paging device (that becomes a master in the **CONNECTION** state) and *slave* is used for the page scanning device (that becomes a slave in the **CONNECTION** state).

2.4.1 Clock Estimate for Paging

A paging device uses an estimate of the native clock of the page scanning device, CLKE; i.e. an offset shall be added to the CLKN of the pager to approximate the CLKN of the recipient, see Figure 2.6 on page 77. CLKE shall be derived from the reference CLKN by adding an offset. By using the CLKN of the recipient, the pager might be able to speed up the connection establishment.

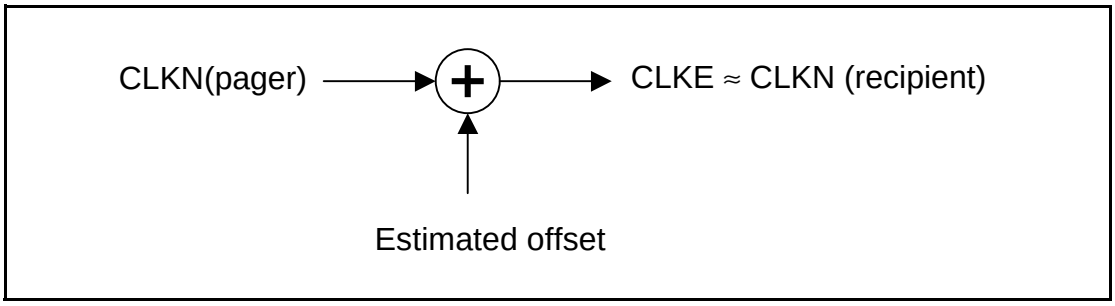


Figure 2.6: Derivation of CLKE.

2.4.2 Hopping Characteristics

The page scan physical channel follows a slower hopping pattern than the basic piconet physical channel and is a short pseudo-random hopping sequence through the RF channels. The timing of the page scan channel shall be determined by the native Bluetooth clock of the scanning device. The frequency hopping sequence is determined by the Bluetooth address of the scanning device.

The page scan physical channel uses the page, master page response, slave page response, and page scan hopping sequences specified in Section 2.6 on page 83.

2.4.3 Paging Procedure Timing

During the paging procedure, the master shall transmit paging messages (see Table 8.3 on page 156) corresponding to the slave to be connected. Since the paging message is a very short packet, the hop rate is 3200 hops/s. In a single TX slot interval, the paging device shall transmit on two different hop frequencies. In Figure 2.7 through Figure 2.11, *f(k)* is used for the frequencies of the page hopping sequence and *f'(k)* denotes the corresponding page response sequence frequencies. The first transmission starts where $CLK_0 = 0$ and the second transmission starts where $CLK_0 = 1$.

In a single RX slot interval, the paging device shall listen for the slave page response message on two different hop frequencies. Similar to transmission, the nominal reception starts where $CLK_0 = 0$ and the second reception nominally starts where $CLK_0 = 1$; see Figure 2.7 on page 78. During the TX slot, the paging device shall send the paging message at the TX hop frequencies *f(k)* and *f*(*k*+1*)*. In the RX slot, it shall listen for a response on the corresponding RX hop frequencies *f'(k)* and *f'*(*k*+1*)*. The listening periods shall be exactly timed 625 µs after the corresponding paging packets, and shall include a ±10 µs uncertainty window.

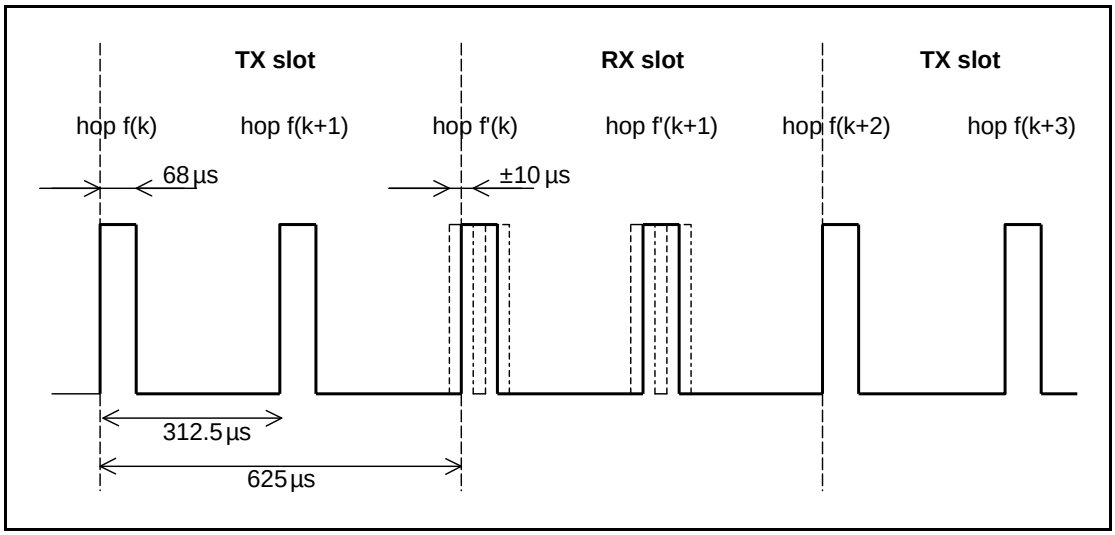


Figure 2.7: RX/TX cycle of transceiver in PAGE mode.

2.4.4 Page Response Timing

At connection setup a master page response packet is transmitted from the master to the slave (see Table 8.3 on page 156). This packet establishes the timing and frequency synchronization. After the slave device has received the page message, it shall return a response message that consists of the slave page response packet and shall follow 625 μs after the receipt of the page message. The master shall send the master page response packet in the TX slot following the RX slot in which it received the slave response, according to the RX/TX timing of the master. The time difference between the slave page response and master page response message will depend on the timing of the page message the slave received. In Figure 2.8 on page 79, the slave receives the paging message sent **first** in the master-to-slave slot. It then responds with a first slave page response packet in the first half of the slave-to-master slot. The timing of the master page response packet is based on the timing of the page message sent first in the preceding master-to-slave slot: there is an exact 1250 μs delay between the first page message and the master page response packet. The packet is sent at the hop frequency *f(k+1)* which is the hop frequency following the hop frequency *f(k)* the page message was received in.

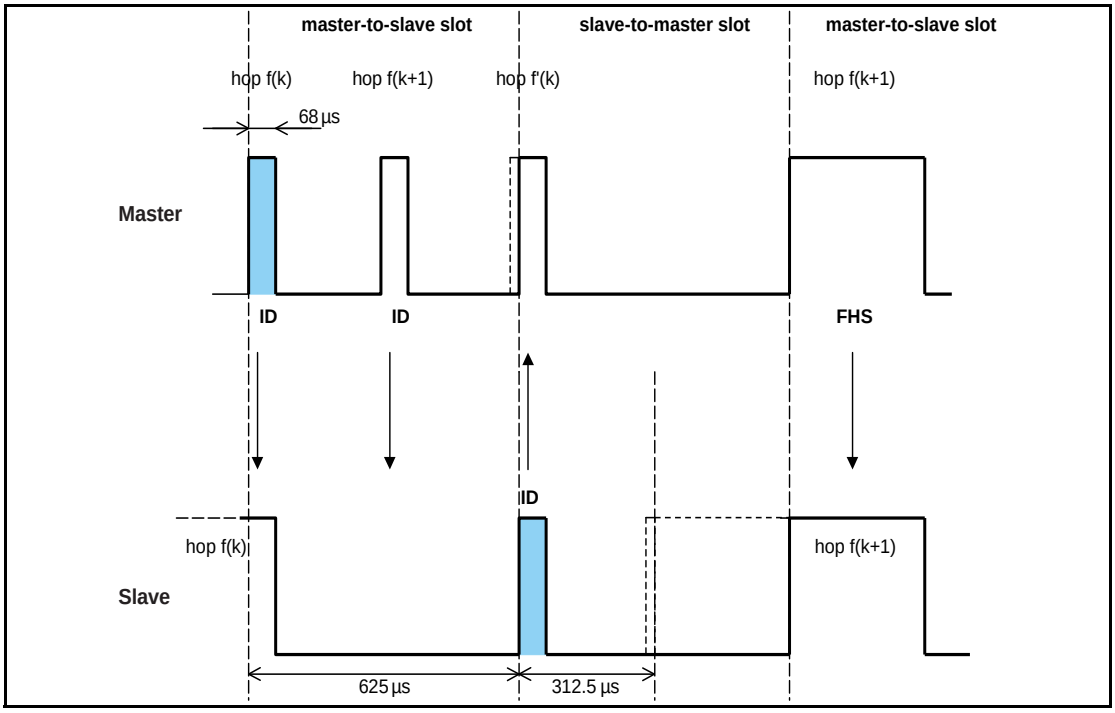


Figure 2.8: Timing of page response packets on successful page in first half slot

In Figure 2.9 on page 80, the slave receives the paging message sent **second** in the master-to-slave slot. It then responds with a slave page response packet in the second half of the slave-to-master slot exactly 625 μs after the receipt of the page message. The timing of the master page response packet is still based on the timing of the page message sent **first** in the preceding master-to-slave slot: there is an exact 1250 μs delay between the **first** page message and the master page response packet. The packet is sent at the hop frequency *f(k+2)* which is the hop frequency following the hop frequency *f(k+1)* the page message was received in.

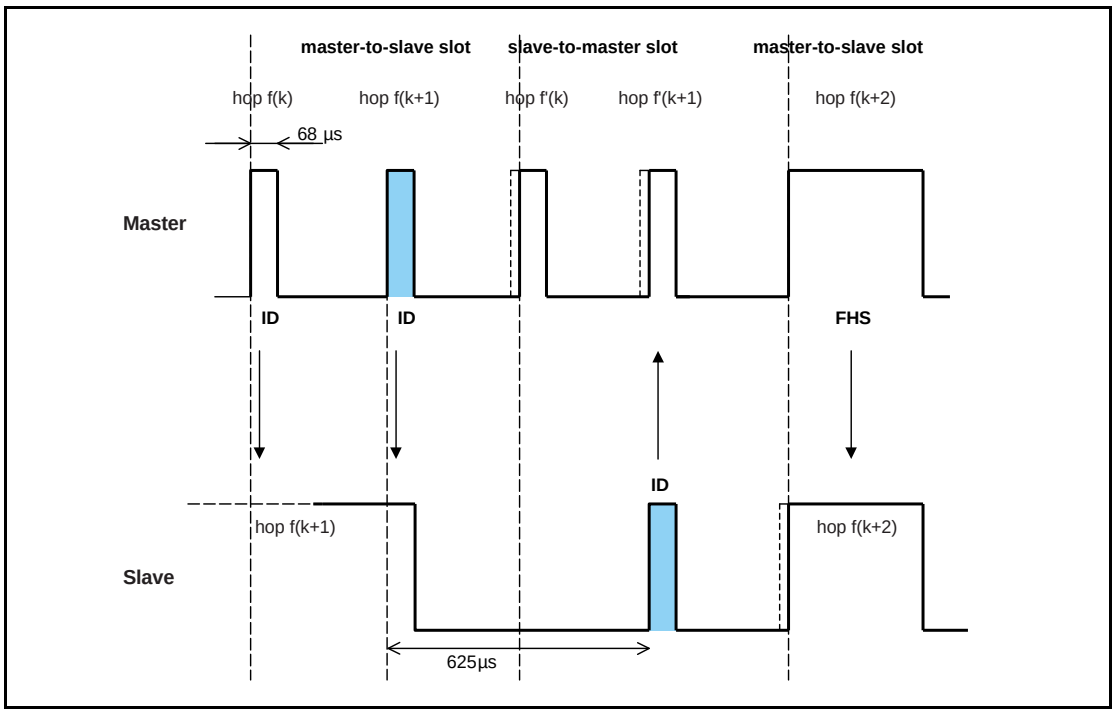


Figure 2.9: Timing of page response packets on successful page in second half slot

The slave shall adjust its RX/TX timing according to the reception of the master page response packet (and not according to the reception of the page message). That is, the second slave page response message that acknowledges the reception of the master page response packet shall be transmitted 625 µs after the start of the master page response packet.

2.5 INQUIRY SCAN PHYSICAL CHANNEL

Although master and slave roles are not defined prior to a connection, the term *master* is used for the inquiring device and *slave* is used for the inquiry scanning device.

2.5.1 Clock for Inquiry

The clock used for inquiry and inquiry scan shall be the device's native clock.

2.5.2 Hopping Characteristics

The inquiry scan channel follows a slower hopping pattern than the piconet physical channel and is a short pseudo-random hopping sequence through the RF channels. The timing of the inquiry scan channel is determined by the native Bluetooth clock of the scanning device while the frequency hopping sequence is determined by the general inquiry access code.

The inquiry scan physical channel uses the inquiry, inquiry response, and inquiry scan hopping sequences described in Section 2.6 on page 83.

2.5.3 Inquiry Procedure Timing

During the inquiry procedure, the master shall transmit inquiry messages with the general or dedicated inquiry access code. The timing for inquiry is the same as for paging (see Section 2.4.3 on page 78).

2.5.4 Inquiry Response Timing

An inquiry response packet is transmitted from the slave to the master after the slave has received an inquiry message (see Table 8.5 on page 165). This packet contains information necessary for the inquiring master to page the slave (see definition of the FHS packet in Section 6.5.1.4 on page 119) and follows 625μs after the receipt of the inquiry message. If the slave transmits an extended inquiry response packet, it shall be transmitted1250μs after the start of the inquiry response packet.

In Figure 2.10 and Figure 2.11, *f(k)* is used for the frequencies of the inquiry hopping sequence and f'(k) denotes the corresponding inquiry response sequence frequency. The inquiry response packet and the extended inquiry response packet are received by the master at the hop frequency *f'(k)* when the inquiry message received by the slave was first in the master-to-slave slot.

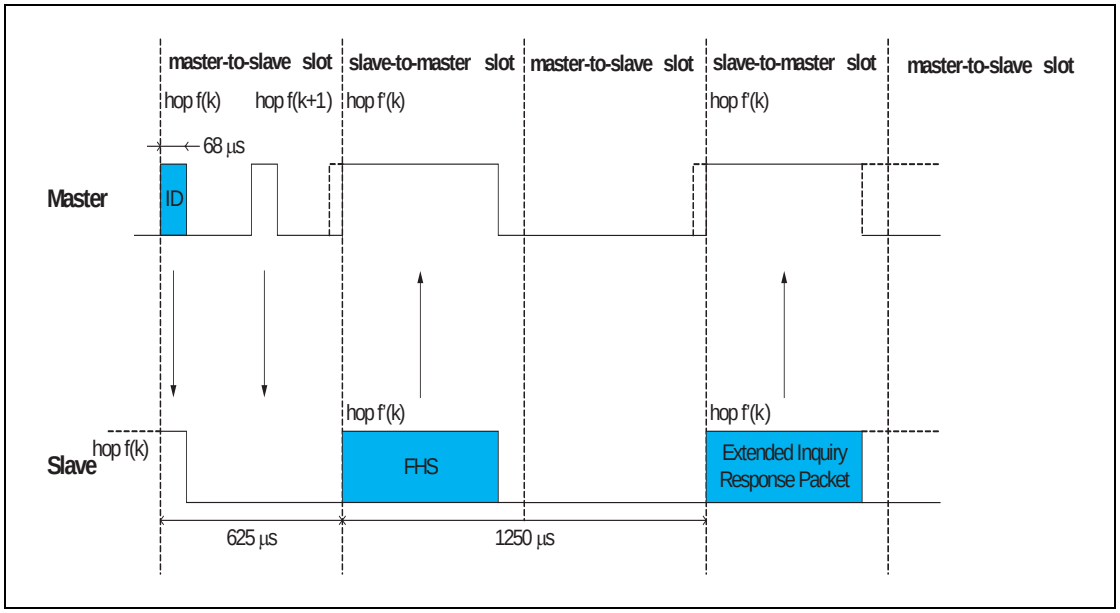


Figure 2.10: Timing of inquiry response packets on successful inquiry in first half slot

When the inquiry message received by the slave was the second in the master-to-slave slot the inquiry response packet and the extended inquiry response packet are received by the master at the hop frequency *f'(k+1)*.

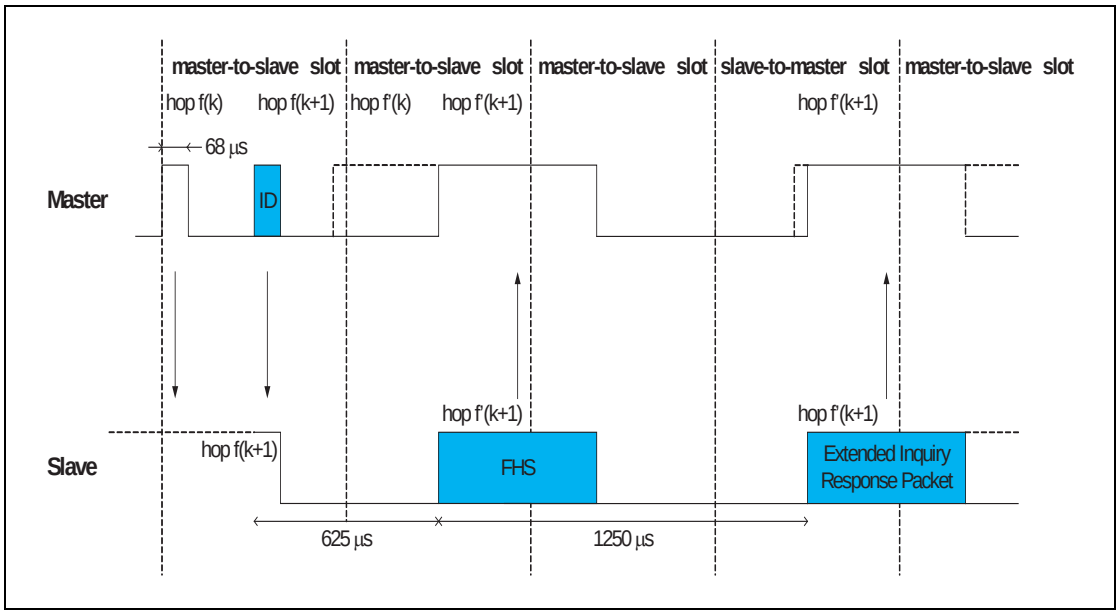


Figure 2.11: Timing of inquiry response packets on successful inquiry in second half slot

2.6 HOP SELECTION

Bluetooth devices shall use the hopping kernel as defined in the following sections.

In total, six types of hopping sequence are defined – five for the basic hop system and one for an adapted set of hop locations used by adaptive frequency hopping (AFH). These sequences are:

- A **page hopping sequence** with 32 wake-up frequencies distributed equally over the 79 MHz, with a period length of 32;
- A **page response hopping sequence** covering 32 response frequencies that are in a one-to-one correspondence to the current page hopping sequence. The master and slave use different rules to obtain the same sequence;
- An **inquiry hopping sequence** with 32 wake-up frequencies distributed equally over the 79 MHz, with a period length of 32;
- An **inquiry response hopping sequence** covering 32 response frequencies that are in a one-to-one correspondence to the current inquiry hopping sequence.
- A **basic channel hopping sequence** which has a very long period length, which does not show repetitive patterns over a short time interval, and which distributes the hop frequencies equally over the 79 MHz during a short time interval.
- An **adapted channel hopping sequence** derived from the basic channel hopping sequence which uses the same channel mechanism and may use fewer than 79 frequencies. The adapted channel hopping sequence is only used in place of the basic channel hopping sequence. All other hopping sequences are not affected by hop sequence adaptation.

2.6.1 General Selection Scheme

The selection scheme consists of two parts:

- selecting a sequence;
- mapping this sequence onto the hop frequencies;

The general block diagram of the hop selection scheme is shown in Figure 2.12 on page 84. The mapping from the input to a particular RF channel index is performed in the selection box.

The inputs to the selection box are the selected clock, frozen clock, N, k_{offset}, address, sequence selection and AFH_channel_map. The source of the clock input depends on the hopping sequence selected. Additionally, each hopping sequence uses different bits of the clock (see Table 2.2 on page 92). N and k_{offset} are defined in Section 2.6.4 on page 91.

The *sequence selection* input can be set to the following values:

- page scan
- inquiry scan
- page
- inquiry
- master page response
- slave page response
- inquiry response
- basic channel
- adapted channel

The address input consists of 28 bits including the entire LAP and the 4 LSBs of the UAP. This is designated as the UAP/LAP. When the basic or adapted channel hopping sequence is selected, the Bluetooth device address of the master (BD_ADDR) shall be used. When the page, master page response, slave page response, or page scan hopping sequences are selected the BD_ADDR given by the Host of the paged device shall be used (see HCI Create Connection Command [Part E] Section 7.1.5 on page 466). When the inquiry, inquiry response, or inquiry scan hopping sequences are selected, the UAP/LAP corresponding to the GIAC shall be used even if it concerns a DIAC. Whenever one of the reserved BD_ADDRs (see Section 1.2.1 on page 68) is used for generating a frequency hop sequence, the UAP shall be replaced by the default check initialization (DCI, see Section 7.1 on page 135). The hopping sequence is selected by the sequence selection input to the selection box.

When the adapted channel hopping sequence is selected, the *AFH_channel_map* is an additional input to the selection box. The *AFH_channel_map* indicates which channels shall be *used* and which shall be *unused*. These terms are defined in Section 2.6.3 on page 90.

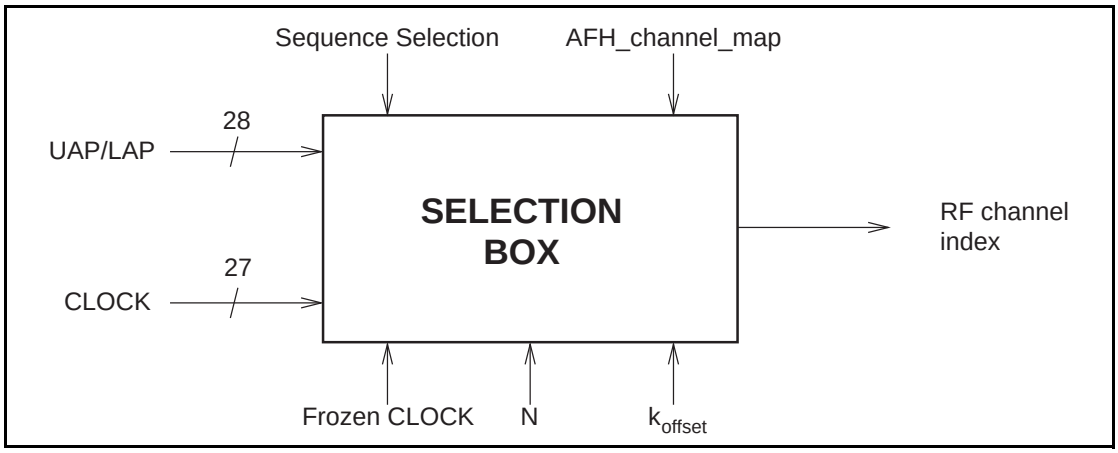


Figure 2.12: General block diagram of hop selection scheme.

The output, *RF channel index*, constitutes a pseudo-random sequence. The RF channel index is mapped to RF channel frequencies using the equation in Table 2.1 on page 35 in the Radio Specification.

The selection scheme chooses a segment of 32 hop frequencies spanning about 64 MHz and visits these hops in a pseudo-random order. Next, a different 32-hop segment is chosen, etc. In the page, master page response, slave page response, page scan, inquiry, inquiry response and inquiry scan hopping sequences, the same 32-hop segment is used all the time (the segment is selected by the address; different devices will have different paging segments). When the basic channel hopping sequence is selected, the output constitutes a pseudo-random sequence that slides through the 79 hops. The principle is depicted in Figure 2.13 on page 85.

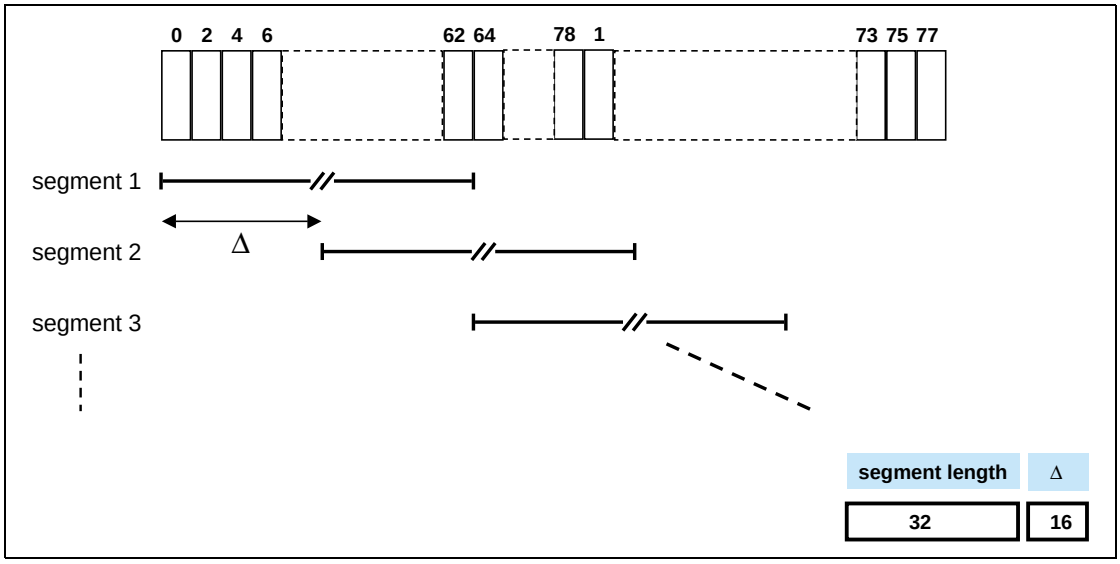


Figure 2.13: Hop selection scheme in CONNECTION state.

The RF frequency shall remain fixed for the duration of the packet. The RF frequency for the packet shall be derived from the Bluetooth clock value in the first slot of the packet. The RF frequency in the first slot after a multi-slot packet shall use the frequency as determined by the Bluetooth clock value for that slot. Figure 2.14 on page 86 illustrates the hop definition on single- and multi-slot packets.

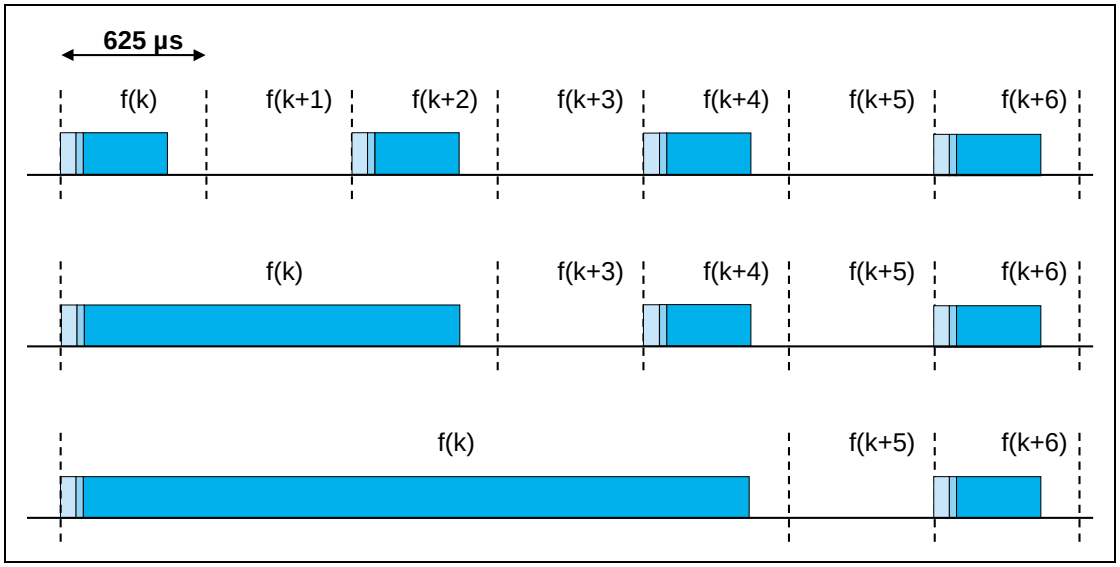


Figure 2.14: Single- and multi-slot packets.

When the adapted channel hopping sequence is used, the pseudo-random sequence contains only frequencies that are in the RF channel set defined by the *AFH_channel_map* input. The adapted sequence has similar statistical properties to the non-adapted hop sequence. In addition, the slave responds with its packet on the same RF channel that was used by the master to address that slave (or would have been in the case of a synchronous reserved slot without a validly received master-to-slave transmission). This is called the *same channel mechanism* of AFH. Thus, the RF channel used for the master to slave packet is also used for the immediately following slave to master packet. An example of the same channel mechanism is illustrated in Figure 2.15 on page 86. The same channel mechanism shall be used whenever the adapted channel hopping sequence is selected.

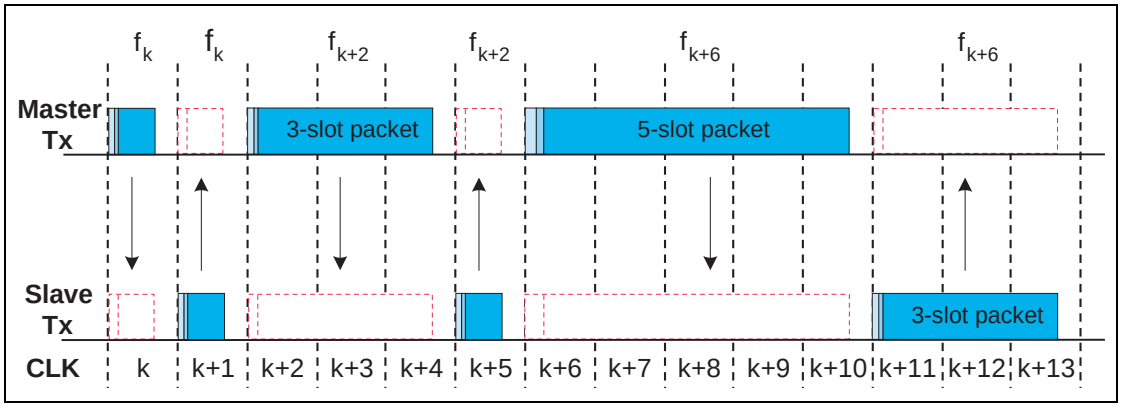


Figure 2.15: Example of the same channel mechanism.

2.6.2 Selection Kernel

The basic hop selection kernel shall be as shown in Figure 2.16 on page 87 and is used for the page, page response, inquiry, inquiry response and basic channel hopping selection kernels. In these substates the AFH_channel_map input is unused. The adapted channel hopping selection kernel is described in Section 2.6.3 on page 90. The X input determines the phase in the 32-hop segment, whereas Y1 and Y2 selects between master-to-slave and slave-to-master. The inputs A to D determine the ordering within the segment, the inputs E and F determine the mapping onto the hop frequencies. The kernel addresses a register containing the RF channel indices. This list is ordered so that first all even RF channel indices are listed and then all odd hop frequencies. In this way, a 32-hop segment spans about 64 MHz.

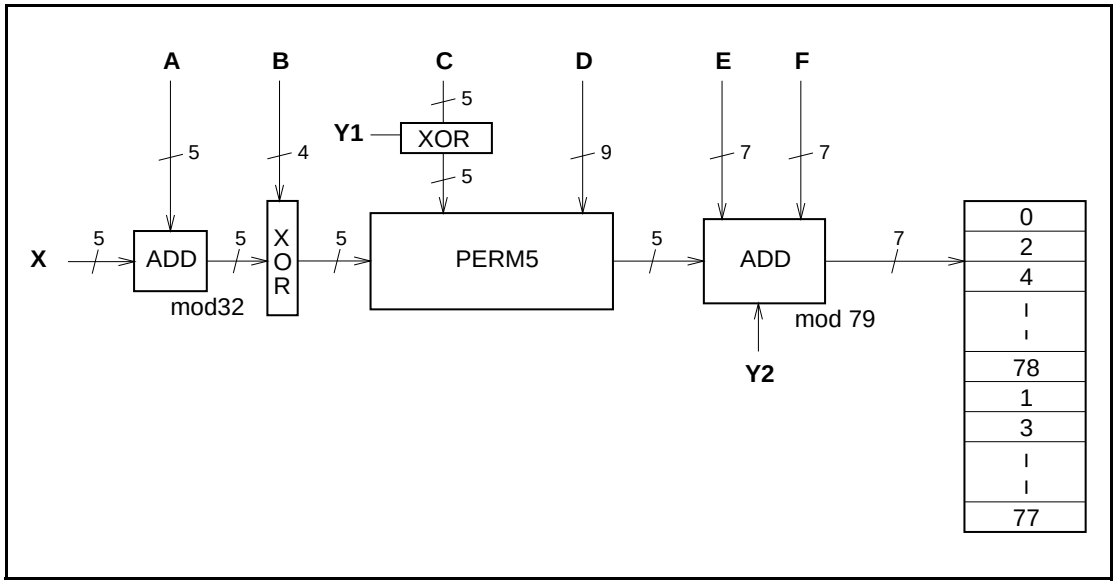


Figure 2.16: Block diagram of the basic hop selection kernel for the hop system.

The selection procedure consists of an addition, an XOR operation, a permutation operation, an addition, and finally a register selection. In the remainder of this chapter, the notation A_i is used for bit i of the BD_ADDR.

2.6.2.1 First addition operation

The first addition operation only adds a constant to the phase and applies a modulo 32 operation. For the page hopping sequence, the first addition is redundant since it only changes the phase within the segment. However, when different segments are concatenated (as in the basic channel hopping sequence), the first addition operation will have an impact on the resulting sequence.

2.6.2.2 XOR operation

Let Z' denote the output of the first addition. In the XOR operation, the four LSBs of Z' are modulo-2 added to the address bits $A_{22\text{-}19}$. The operation is illustrated in Figure 2.17 on page 88.

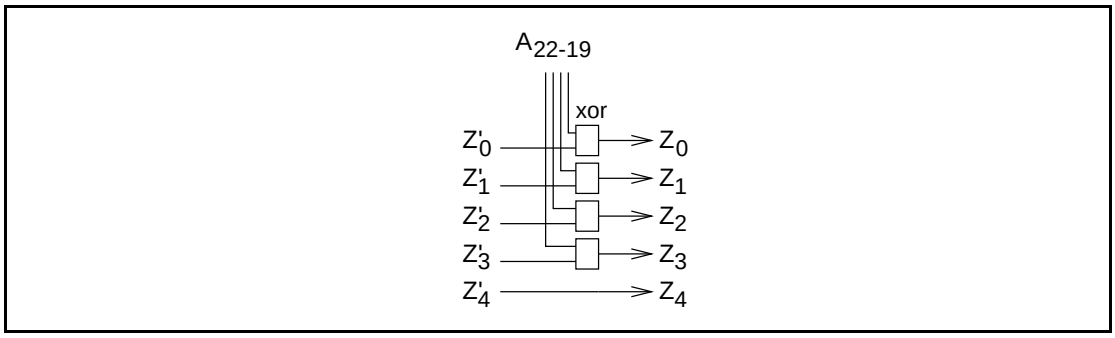


Figure 2.17: XOR operation for the hop system.

2.6.2.3 Permutation operation

The permutation operation involves the switching from 5 inputs to 5 outputs for the hop system, controlled by the control word. The permutation or switching box shall be as shown in Figure 2.18 on page 89. It consists of 7 stages of butterfly operations. The control of the butterflies by the control signals P is shown in Table 2.1. $P_{0\text{-}8}$ corresponds to $D_{0\text{-}8}$, and, P_{i+9} corresponds to $C_i \oplus Y1$ for $i = 0 \ldots 4$ in Figure 2.16.

Control signal	Butterfly		Control signal	Butterfly
P_0	$\{Z_0,Z_1\}$		P_8	$\{Z_1,Z_4\}$
P_1	$\{Z_2,Z_3\}$		P_9	$\{Z_0,Z_3\}$
P_2	$\{Z_1,Z_2\}$		P_{10}	$\{Z_2,Z_4\}$
P_3	$\{Z_3,Z_4\}$		P_{11}	$\{Z_1,Z_3\}$
P_4	$\{Z_0,Z_4\}$		P_{12}	$\{Z_0,Z_3\}$
P_5	$\{Z_1,Z_3\}$		P_{13}	$\{Z_1,Z_2\}$
P_6	$\{Z_0,Z_2\}$			
P_7	$\{Z_3,Z_4\}$			

Table 2.1: Control of the butterflies for the hop system

The Z input is the output of the XOR operation as described in the previous section. The butterfly operation can be implemented with multiplexers as depicted in Figure 2.19 on page 89.

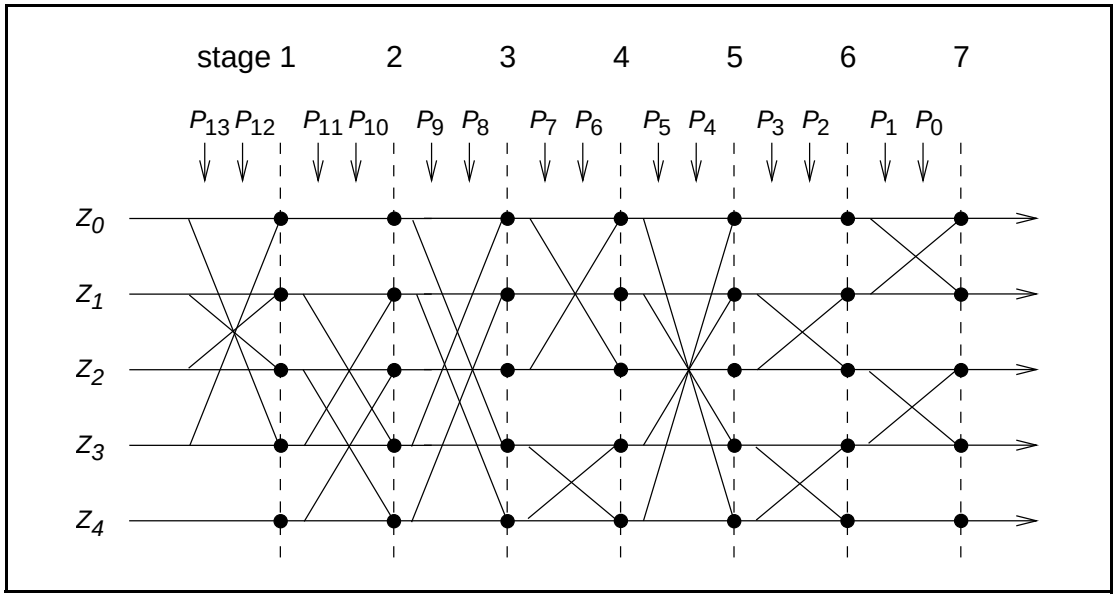


Figure 2.18: Permutation operation for the hop system.

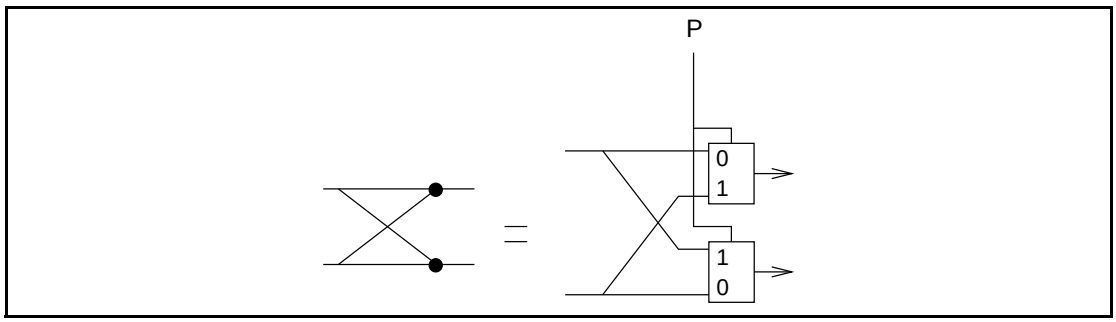


Figure 2.19: Butterfly implementation.

2.6.2.4 Second addition operation

The addition operation only adds a constant to the output of the permutation operation. The addition is applied modulo 79.

2.6.2.5 Register bank

The output of the adder addresses a bank of 79 registers. The registers are loaded with the synthesizer code words corresponding to the hop frequencies 0 to 78. Note that the upper half of the bank contains the even hop frequencies, whereas the lower half of the bank contains the odd hop frequencies.

2.6.3 Adapted Hop Selection Kernel

The adapted hop selection kernel is based on the basic hop selection kernel defined in the preceding sections.

The inputs to the adapted hop selection kernel are the same as for the basic hop system kernel except that the input AFH_channel_map (defined in Link Manager Protocol [Part C] Section 5.2 on page 320) is used. The *AFH_channel_map* indicates which RF channels shall be *used* and which shall be *unused*. When hop sequence adaptation is enabled, the number of *used* RF channels may be reduced from 79 to some smaller value N. All devices shall be capable of operating on an adapted hop sequence (AHS) with $N_{min} \leq N \leq 79$, with any combination of *used* RF channels within the *AFH_channel_map* that meets this constraint. N_{min} is defined in Section 2.3.1 on page 76.

Adaptation of the hopping sequence is achieved through two additions to the basic channel hopping sequence according to Figure 2.16 on page 87:

- *Unused* RF channels are re-mapped uniformly onto *used* RF channels. That is, if the hop selection kernel of the basic system generates an *unused* RF channel, an alternative RF channel out of the set of *used* RF channels is selected pseudo-randomly.
- The *used* RF channel generated for the master-to-slave packet is also used for the immediately following slave-to-master packet (see Section 2.6.1 on page 83).

2.6.3.1 Channel re-mapping function

When the adapted hop selection kernel is selected, the basic hop selection kernel according to Figure 2.16 on page 87 is initially used to determine an RF channel. If this RF channel is *unused* according to the *AFH_channel_map*, the *unused* RF channel is re-mapped by the re-mapping function to one of the *used* RF channels. If the RF channel determined by the basic hop selection kernel is already in the set of *used* RF channels, no adjustment is made. The hop sequence of the (non-adapted) basic hop equals the sequence of the adapted selection kernel on all locations where *used* RF channels are generated by the basic hop. This property facilitates non-AFH slaves remaining synchronized while other slaves in the piconet are using the adapted hopping sequence.

A block diagram of the re-mapping mechanism is shown in Figure 2.20 on page 91. The re-mapping function is a post-processing step to the selection kernel from Figure 2.16 on page 87, denoted as ‘Hop selection of the basic hop’. The output f_k of the basic hop selection kernel is an RF channel number that ranges between 0 and 78. This RF channel will either be in the set of *used* RF channels or in the set of *unused* RF channels.

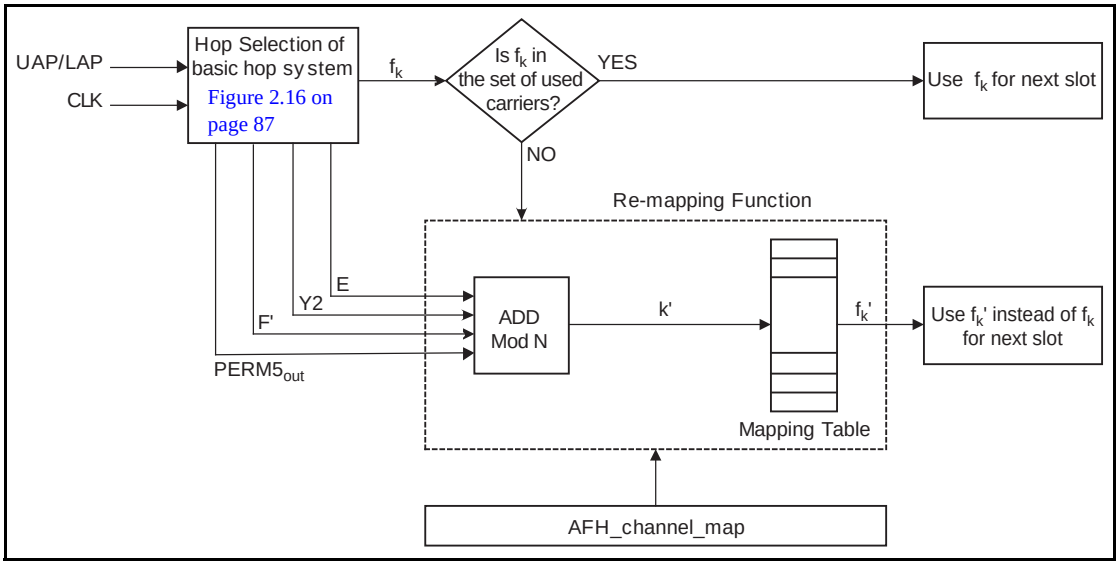


Figure 2.20: Block diagram of adaptive hop selection mechanism

When an unused RF channel is generated by the basic hop selection mechanism, it is re-mapped to the set of *used* RF channels as follows. A new index *k'* $\in$ {0, 1,..., *N*-1} is calculated using some of the parameters from the basic hop selection kernel:

$$k' = (PERM5_{out} + E + F' + Y2) \bmod N$$

where *F'* is defined in Table 2.2 on page 92. The index *k'* is then used to select the re-mapped channel from a mapping table that contains all of the even *used* RF channels in ascending order followed by all the odd *used* RF channels in ascending order (i.e., the mapping table of Figure 2.16 on page 87 with all the *unused* RF channels removed).

2.6.4 Control Word

In the following section X_{j-i}, $i<j$, denotes bits i, i+1,...,j of the bit vector X. By convention, X_0 is the least significant bit of the vector X.

The control word of the kernel is controlled by the overall control signals X, Y1, Y2, A to F, and F' as illustrated in Figure 2.16 on page 87 and Figure 2.20 on page 91. During paging and inquiry, the inputs A to E use the address values as given in the corresponding columns of Table 2.2 on page 92. In addition, the inputs X, Y1 and Y2 are used. The F and F' inputs are unused. The clock bits CLK_{6-2} (i.e., input X) specifies the phase within the length 32 sequence. CLK_1 (i.e., inputs Y1 and Y2) is used to select between TX and RX. The address inputs determine the sequence order within segments. The final mapping onto the hop frequencies is determined by the register contents.

During the **CONNECTION** state (see Section 8.5 on page 165), the inputs A, C and D shall be derived from the address bits being bit-wise XORed with the

clock bits as shown in the “Connection state” column of Table 2.2 on page 92 (the two most significant bits, MSBs, are XORed together, the two second MSBs are XORed together, etc.).

	Page scan / Interlaced Page Scan / Inquiry scan / Interlaced Inquiry Scan	**Page/Inquiry**	**Master/Slave page response and Inquiry response**	**Connection state**
X	$CLKN_{16-12}$ / $(CLKN_{16-12}+16) \bmod 32$ / Xir_{4-0} / $Xir_{4-0}+16) \bmod 32$	Xp_{4-0}/Xi_{4-0}	$Xprm_{4-0}$ / $Xprs_{4-0}$ / Xir_{4-0}	CLK_{6-2}
Y1	0	$CLKE_1/CLKN_1$	$CLKE_1/CLKN_1/1$	CLK_1
Y2	0	$32 \times CLKE_1/$ $32 \times CLKN_1$	$32 \times CLKE_1$ / $32 \times CLKN_1$ / 32×1	$32 \times CLK_1$
A	A_{27-23}	A_{27-23}	A_{27-23}	$A_{27-23} \oplus CLK_{25-21}$
B	A_{22-19}	A_{22-19}	A_{22-19}	A_{22-19}
C	$A_{8,6,4,2,0}$	$A_{8,6,4,2,0}$	$A_{8,6,4,2,0}$	$A_{8,6,4,2,0} \oplus CLK_{20-16}$
D	A_{18-10}	A_{18-10}	A_{18-10}	$A_{18-10} \oplus CLK_{15-7}$
E	$A_{13,11,9,7,5,3,1}$	$A_{13,11,9,7,5,3,1}$	$A_{13,11,9,7,5,3,1}$	$A_{13,11,9,7,5,3,1}$
F	0	0	0	$16 \times CLK_{27-7} \bmod 79$
F’	n/a	n/a	n/a	$16 \times CLK_{27-7} \bmod N$

Table 2.2: Control for hop system.

The five X input bits vary depending on the current state of the device. In the **page scan** and **inquiry scan** substates, the native clock (CLKN) shall be used. In **CONNECTION** state the master clock (CLK) shall be used as input. The situation is somewhat more complicated for the other states.

2.6.4.1 Page scan and inquiry scan hopping sequences

When the sequence selection input is set to page scan, the Bluetooth device address of the scanning device shall be used as address input. When the sequence selection input is set to inquiry scan, the GIAC LAP and the four LSBs of the DCI (as A_{27-24}), shall be used as address input for the hopping sequence. For the transmitted access code and in the receiver correlator, the appropriate GIAC or DIAC shall be used. The application decides which inquiry access code to use depending on the purpose of the inquiry.

2.6.4.2 Page hopping sequence

When the sequence selection input is set to page, the paging device shall start using the **A**-train, i.e., $\{f(k-8), \ldots, f(k), \ldots, f(k+7)\}$, where $f(k)$ is the source's estimate of the current receiver frequency in the paged device. The index k is a function of all the inputs in Figure 2.16. There are 32 possible paging frequencies within each 1.28 second interval. Half of these frequencies belong to the **A**-train, the rest (i.e., $\{f(k+8), \ldots, f(k+15), f(k-16), \ldots, f(k-9)\}$) belong to the **B**-train. In order to achieve the -8 offset of the **A**-train, a constant of 24 shall be added to the clock bits (which is equivalent to -8 due to the modulo 32 operation). The **B**-train is obtained by setting the offset to 8. A cyclic shift of the order within the trains is also necessary in order to avoid a possible repetitive mismatch between the paging and scanning devices. Thus,

$$Xp = [CLKE_{16-12} + k_{offset} + (CLKE_{4-2,0} - CLKE_{16-12}) \bmod 16] \bmod 32, \quad \text{(EQ 2)}$$

where

$$k_{offset} = \begin{cases} 24 & \text{A-train,} \\ 8 & \text{B-train.} \end{cases} \quad \text{(EQ 3)}$$

Alternatively, each switch between the **A**- and **B**-trains may be accomplished by adding 16 to the current value of k_{offset} (originally initialized with 24).

2.6.4.3 Slave page response hopping sequence

When the sequence selection input is set to *slave page response,* in order to eliminate the possibility of losing the link due to discrepancies of the native clock CLKN and the master's clock estimate CLKE, the four bits $CLKN_{16-12}$ shall be frozen at their current value. The value shall be frozen at the content it has in the slot where the recipient's access code is detected. The native clock shall *not* be stopped; it is merely the values of the bits used for creating the X-input that are kept fixed for a while. A frozen value is denoted by an asterisk (*) in the discussion below.

For each response slot the paged device shall use an X-input value one larger (modulo 32) than in the preceding response slot. However, the first response shall be made with the X-input kept at the same value as it was when the access code was recognized. Let N be a counter starting at zero. Then, the X-input in the $(N+1)$-th response slot (the first response slot being the one immediately following the page slot now responding to) of the **slave response** substate is:

$$Xprs = [CLKN^*_{16-12} + N] \bmod 32, \quad \text{(EQ 4)}$$

The counter N shall be set to zero in the slot where the slave acknowledges the page (see Figure 8.3 on page 157 and Figure 8.4 on page 157). Then, the value of N shall be increased by one each time $CLKN_1$ is set to zero, which corresponds to the start of a master TX slot. The X-input shall be constructed this way until the first **FHS** packet is received *and* the immediately following response packet has been transmitted. After this the slave shall enter the **CONNECTION** state using the parameters received in the **FHS** packet.

2.6.4.4 Master page response hopping sequence

When the sequence selection input is set to *master page response*, the master shall freeze its estimated slave clock to the value that triggered a response from the paged device. It is equivalent to using the values of the clock estimate when receiving the slave response (since only $CLKE_1$ will differ from the corresponding page transmission). Thus, the values are frozen when the slave **ID** packet is received. In addition to the clock bits used, the current value of k_{offset} shall also be frozen. The master shall adjust its X-input in the same way the paged device does, i.e., by incrementing this value by one for each time $CLKE_1$ is set to zero. The first increment shall be done before sending the **FHS** packet to the paged device. Let N be a counter starting at one. The rule for forming the X-input is:

$$Xprm = [CLKE^*_{16-12} + k_{offset}{}^* + (CLKE^*_{4-2,0} - CLKE^*_{16-12}) \bmod 16 + N] \bmod 32, \quad \text{(EQ 5)}$$

The value of N shall be increased each time $CLKE_1$ is set to zero, which corresponds to the start of a master TX slot.

2.6.4.5 Inquiry hopping sequence

When the sequence selection input is set to *inquiry*, the X-input is similar to that used in the *page hopping sequence*. Since no particular device is addressed, the native clock CLKN of the inquirer shall be used. Moreover, which of the two train offsets to start with is of no real concern in this state. Consequently,

$$Xi = [CLKN_{16-12} + k_{offset} + (CLKN_{4-2,0} - CLKN_{16-12}) \bmod 16] \bmod 32, \quad \text{(EQ 6)}$$

where k_{offset} is defined by (EQ 3) on page 93. The initial choice of the offset is arbitrary.

The GIAC LAP and the four LSBs of the DCI (as A_{27-24}) shall be used as address input for the hopping sequence generator.

2.6.4.6 Inquiry response hopping sequence

The *inquiry response* hopping sequence is similar to the *slave page response* hopping sequence with respect to the X-input. The clock input shall not be frozen, thus the following equation apply:

$$Xir = [CLKN_{16-12} + N] \mod 32, \quad \text{(EQ 7)}$$

Furthermore, the counter N is increased not on $CLKN_1$ basis, but rather after each **FHS** packet has been transmitted in response to the inquiry. There is no restriction on the initial value of N as it is independent of the corresponding value in the inquiring unit.

The X_{ir} value used for the extended inquiry response packet shall be the same X_{ir} value as calculated for the immediately preceding FHS packet.

The GIAC LAP and the four LSBs of the DCI (as A_{27-24}) shall be used as address input for the hopping sequence generator. The other input bits to the generator shall be the same as for page response.

2.6.4.7 Basic and adapted channel hopping sequence

In the *basic* and *adapted channel hopping sequences*, the clock bits to use in the basic or adapted hopping sequence generation shall always be derived from the master clock, CLK. The address bits shall be derived from the Bluetooth device address of the master.

3 PHYSICAL LINKS

A physical link represents a baseband connection between devices. A physical link is always associated with exactly one physical channel. Physical links have common properties that apply to all logical transports on the physical link.

The common properties of physical links are:

- Power control (see Link Manager Protocol Section 4.1.3 on page 230)
- Link supervision (see Section 3.1 on page 96 and Link Manager Protocol Section 4.1.6 on page 239)
- Encryption (see Security [Part H] Section 4 on page 1072 and Link Manager Protocol [Part C] Section 4.2.5 on page 255)
- Channel quality-driven data rate change (see Link Manager Protocol Section 4.1.7 on page 240)
- Multi-slot packet control (see Link Manager Protocol Section 4.1.10 on page 244)

3.1 LINK SUPERVISION

A connection can break down due to various reasons such as a device moving out of range, encountering severe interference or a power failure condition. Since this can happen without any prior warning, it is important to monitor the link on both the master and the slave side to avoid possible collisions when the logical transport address (see Section 4.2 on page 97) or parked member address (see Section 4.7.1 on page 105) is reassigned to another slave.

To be able to detect link loss, both the master and the slave shall use a link supervision timer, $T_{supervision}$. Upon reception of a valid packet header with one of the slave's addresses (see Section 4.2 on page 97) on the physical link, the timer shall be reset. If at any time in **CONNECTION** state, the timer reaches the *supervisionTO* value, the connection shall be considered disconnected. The same link supervision timer shall be used for SCO, eSCO, and ACL logical transports.

The timeout period, *supervisionTO*, is negotiated by the Link Manager. Its value shall be chosen so that the supervision timeout will be longer than hold and sniff periods. Link supervision of a parked slave shall be done by unparking and re-parking the slave.

4 LOGICAL TRANSPORTS

4.1 GENERAL

Between master and slave(s), different types of logical transports may be established. Five logical transports have been defined:

- Synchronous Connection-Oriented (SCO) logical transport
- Extended Synchronous Connection-Oriented (eSCO) logical transport
- Asynchronous Connection-Oriented (ACL) logical transport
- Active Slave Broadcast (ASB) logical transport
- Parked Slave Broadcast (PSB) logical transport

The synchronous logical transports are point-to-point logical transports between a master and a single slave in the piconet. The synchronous logical transports typically support time-bounded information like voice or general synchronous data. The master maintains the synchronous logical transports by using reserved slots at regular intervals. In addition to the reserved slots the eSCO logical transport may have a retransmission window after the reserved slots.

The ACL logical transport is also a point-to-point logical transport between the master and a slave. In the slots not reserved for synchronous logical transport(s), the master can establish an ACL logical transport on a per-slot basis to any slave, including the slave(s) already engaged in a synchronous logical transport.

The ASB logical transport is used by a master to communicate with active slaves. The PSB logical transport is used by a master to communicate with parked slaves.

4.2 LOGICAL TRANSPORT ADDRESS (LT_ADDR)

Each slave active in a piconet is assigned a primary 3-bit logical transport address (LT_ADDR). The all-zero LT_ADDR is reserved for broadcast messages. The master does not have an LT_ADDR. A master's timing relative to the slaves distinguishes it from the slaves. A secondary LT_ADDR is assigned to the slave for each eSCO logical transport in use in the piconet. Only eSCO traffic (i.e. NULL, POLL, and one of the EV packet types as negotiated at eSCO logical transport setup) may be sent on these LT_ADDRs. ACL traffic (including LMP) shall always be sent on the primary LT_ADDR. A slave shall only accept packets with matching primary or secondary LT_ADDR and broadcast packets. The LT_ADDR is carried in the packet header (see Section 6.4 on page 115). The LT_ADDR shall only be valid for as long as a slave is in the active mode. As soon as it is disconnected or parked, the slave shall lose all of its LT_ADDRs.

The primary LT_ADDR shall be assigned by the master to the slave when the slave is activated. This is either at connection establishment, or role switch, or when the slave is unparked. At connection establishment and at role switch, the primary LT_ADDR is carried in the **FHS** payload. When unparking, the primary LT_ADDR is carried in the unpark message.

4.3 SYNCHRONOUS LOGICAL TRANSPORTS

The first type of synchronous logical transport, the SCO logical transport, is a symmetric, point-to-point transport between the master and a specific slave. The SCO logical transport reserves slots and can therefore be considered as a circuit-switched connection between the master and the slave. The master may support up to three SCO links to the same slave or to different slaves. A slave may support up to three SCO links from the same master, or two SCO links if the links originate from different masters. SCO packets are never retransmitted.

The second type of synchronous logical transport, the eSCO logical transport, is a point-to-point logical transport between the master and a specific slave. eSCO logical transports may be symmetric or asymmetric. Similar to SCO, eSCO reserves slots and can therefore be considered a circuit-switched connection between the master and the slave. In addition to the reserved slots, eSCO supports a retransmission window immediately following the reserved slots. Together, the reserved slots and the retransmission window form the complete eSCO window.

4.4 ASYNCHRONOUS LOGICAL TRANSPORT

In the slots not reserved for synchronous logical transports, the master may exchange packets with any slave on a per-slot basis. The ACL logical transport provides a packet-switched connection between the master and all active slaves participating in the piconet. Both asynchronous and isochronous services are supported. Between a master and a slave only a single ACL logical transport shall exist. For most ACL packets, packet retransmission is applied to assure data integrity.

ACL packets not addressed to a specific slave are considered as broadcast packets and should be read by every slave. If there is no data to be sent on the ACL logical transport and no polling is required, no transmission is required.

4.5 TRANSMIT/RECEIVE ROUTINES

This section describes the way to use the packets as defined in Section 6 in order to support the traffic on the ACL, SCO and eSCO logical transports. Both single-slave and multi-slave configurations are considered. In addition, the use of buffers for the TX and RX routines are described.

The TX and RX routines described in sections 4.5.1 and 4.5.2 are informative only.

4.5.1 TX Routine

The TX routine is carried out separately for each asynchronous and synchronous logical transport. Figure 4.1 on page 99 shows the asynchronous and synchronous buffers as used in the TX routine. In this figure, only a single TX asynchronous buffer and a single TX synchronous buffer are shown. In the master, there is a separate TX asynchronous buffer for each slave. In addition there can be one or more TX synchronous buffers for each synchronous slave (different SCO or eSCO logical transports could either reuse the same TX synchronous buffer, or each have their own TX synchronous buffer). Each TX buffer consists of two FIFO registers: one **current** register which can be accessed and read by the Link Controller in order to compose the packets, and one **next** register that can be accessed by the Baseband Resource Manager to load new information. The positions of the switches S1 and S2 determine which register is current and which register is next; the switches are controlled by the Link Controller. The switches at the input and the output of the FIFO registers can never be connected to the same register simultaneously.

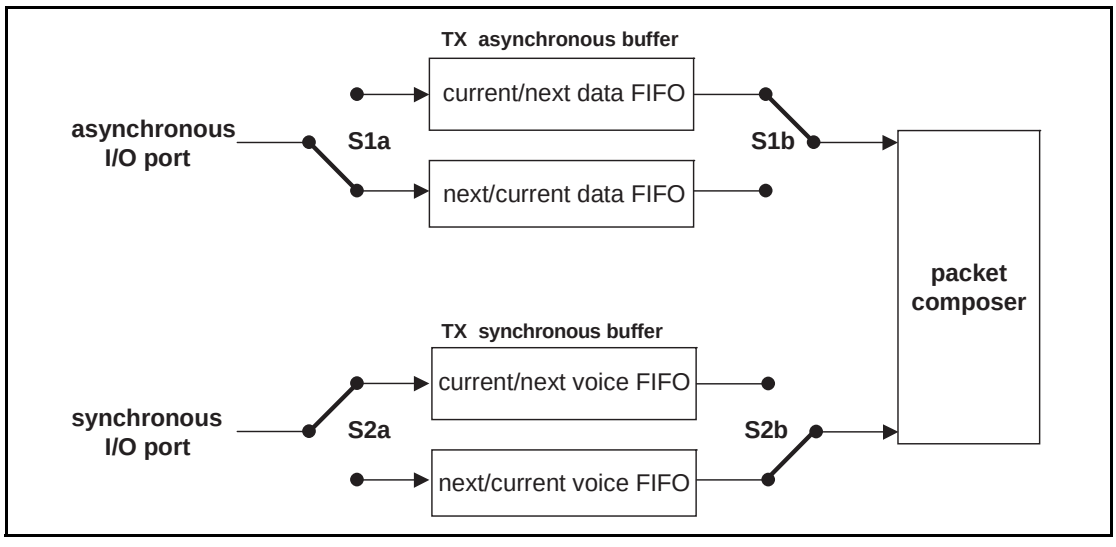


Figure 4.1: Functional diagram of TX buffering.

Of the packets common on the ACL and SCO logical transports (**NULL**, **POLL and DM1**) only the **DM1** packet carries a payload that is exchanged between the Link Controller and the Link Manager; this common packet makes use of the asynchronous buffer. All ACL packets make use of the asynchronous buf-

fer. All SCO and eSCO packets make use of the synchronous buffer except for the **DV** packet where the synchronous data part is handled by the synchronous buffer and the data part is handled by the asynchronous buffer. In the next sections, the operation for ACL traffic, SCO traffic, eSCO traffic, and combined data-voice traffic on the SCO logical transport are described.

4.5.1.1 ACL traffic

In the case of asynchronous data only the TX ACL buffer in Figure 4.1 on page 99 has to be considered. In this case, only packet types **DM** or **DH** are used, and these can have different lengths. The length is indicated in the payload header. The selection of **DM** or **DH** packets should depend on the quality of the link. See [Part C] Section 4.1.7 on page 240.

The default packet type in pure data traffic is **NULL** (see Section 6.5.1.2 on page 119). This means that, if there is no data to be sent (the data traffic is asynchronous, and therefore pauses occur in which no data is available) or no slaves need to be polled, **NULL** packets are sent instead – in order to send link control information to the other device (e.g. ACK/STOP information for received data). When no link control information is available (either no need to acknowledge and/or no need to stop the RX flow) no packet is sent at all.

The TX routine works as follows. The Baseband Resource Manager loads new data information in the register to which the switch S1a points. Next, it gives a command to the Link Controller, which forces the switch S1 to change (both S1a and S1b switch synchronously). When the payload needs to be sent, the packet composer reads the current register and, depending on the packet type, builds a payload which is appended to the channel access code and the header and is subsequently transmitted. In the response packet (which arrives in the following RX slot if it concerned a master transmission, or may be postponed until some later RX slot if it concerned a slave transmission), the result of the transmission is reported back. In case of an ACK, the switch S1 changes position; if a NAK (explicit or implicit) is received instead, the switch S1 will not change position. In that case, the same payload is retransmitted at the next TX occasion.

As long as the Baseband Resource Manager keeps loading the registers with new information, the Link Controller will automatically transmit the payload; in addition, retransmissions are performed automatically in case of errors. The Link Controller will send **NULL** or nothing when no new data is loaded. If no new data has been loaded in the **next** register, during the last transmission, the packet composer will be pointing to an empty register after the last transmission has been acknowledged and the **next** register becomes the **current** register. If new data is loaded in the **next** register, a **flush** command is required to switch the S1 switch to the proper register. As long as the Baseband Resource Manager keeps loading the data and type registers before each TX slot, the data is automatically processed by the Link Controller since the S1 switch is controlled by the ACK information received in response. However, if the traffic from the Baseband Resource Manager is interrupted once and a default packet is sent instead, a **flush** command is necessary to continue the flow in the Link Controller.

The **flush** command may also be used in case of time-bounded (isochronous) data. In case of a bad link, many retransmissions are necessary. In certain applications, the data is time-bounded: if a payload is retransmitted all the time because of link errors, it may become outdated, and the system might decide to continue with more recent data instead and skip the payload that does not come through. This is accomplished by the **flush** command as well. With the **flush**, the switch S1 is forced to change and the Link Controller is forced to consider the next data payload and overrules the ACK control. Any ACL type of packet can be used to send data or link control information to any other ACL slave.

4.5.1.2 SCO traffic

On the SCO logical transport only **HV** and **DV** packet types are used, See Section 6.5.2 on page 122. The synchronous port may continuously load the **next** register in the synchronous buffer. The S2 switches are changed according to the T_{sco} interval. This T_{sco} interval is negotiated between the master and the slave at the time the SCO logical transport is established.

For each new SCO slot, the packet composer reads the **current** register after which the S2 switch is changed. If the SCO slot has to be used to send control information with high priority concerning a control packet between the master and the SCO slave, or a control packet between the master and any other slave, the packet composer will discard the SCO information and use the control information instead. This control information shall be sent in a DM1 packet. Data or link control information may also be exchanged between the master and the SCO slave by using the **DV** or **DM1** packets.

4.5.1.3 Mixed data/voice traffic

In Section 6.5.2 on page 122, a **DV** packet has been defined that can support both data and voice simultaneously on a single SCO logical transport. When the TYPE is **DV**, the Link Controller reads the data register to fill the data field and the voice register to fill the voice field. Thereafter, the switch S2 is changed. However, the position of S1 depends on the result of the transmission as on the ACL logical transport: only if an ACK has been received will the S1 switch change its position. In each **DV** packet, the voice information is new, but the data information might be retransmitted if the previous transmission failed. If there is no data to be sent, the SCO logical transport will automatically change from **DV** packet type to the current **HV** packet type used before the mixed data/voice transmission. Note that a **flush** command is necessary when the data stream has been interrupted and new data has arrived.

Combined data-voice transmission can also be accomplished by using a separate ACL logical transport in addition to the SCO logical transport(s) if channel capacity permits this.

4.5.1.4 eSCO Traffic

On the eSCO logical transport only **EV**, **POLL** and **NULL** packet types are used, see Section 6.5.3 on page 123. The synchronous port may continuously load the next register in the synchronous buffer. The S2 switches are changed according to the T_{eSCO} interval. This T_{eSCO} interval is negotiated between the master and the slave at the time the eSCO logical transport is established.

For each new eSCO slot, the packet composer reads the current register after which the S2 switch is changed. If the eSCO slot has to be used to send control information with high priority concerning a control packet between the master and the eSCO slave, or an ACL packet between the master and any other slave, the packet composer will discard the eSCO information and use the control information instead. Control information to the eSCO slave is sent in a DM1 packet on the primary LT_ADDR.

4.5.1.5 Default packet types

On the ACL links, the default type is always **NULL** both for the master and the slave. This means that if no user information needs to be sent, either a **NULL** packet is sent if there is **ACK** or **STOP** information, or no packet is sent at all. The **NULL** packet can be used by the master to allocate the next slave-to-master slot to a certain slave (namely the one addressed). However, the slave is not forced to respond to the **NULL** packet from the master. If the master requires a response, it sends a **POLL** packet.

The SCO and eSCO packet types are negotiated at the LM level when the SCO or eSCO logical transport is established. The agreed packet type is also the default packet type for the reserved SCO or eSCO slots.

4.5.2 RX Routine

The RX routine is carried out separately for the ACL logical transport and the synchronous logical transports. However, in contrast to the master TX asynchronous buffer, a single RX buffer is shared among all slaves. For the synchronous buffer, how the different synchronous logical transports are distinguished depends on whether extra synchronous buffers are required or not. Figure 4.2 on page 103 shows the asynchronous and synchronous buffers as used in the RX routine. The RX asynchronous buffer consists of two FIFO registers: one register that can be accessed and loaded by the Link Controller with the payload of the latest RX packet, and one register that can be accessed by the Baseband Resource Manager to read the previous payload. The RX synchronous buffer also consists of two FIFO registers: one register which is filled with newly arrived voice information, and one register which can be read by the voice processing unit.

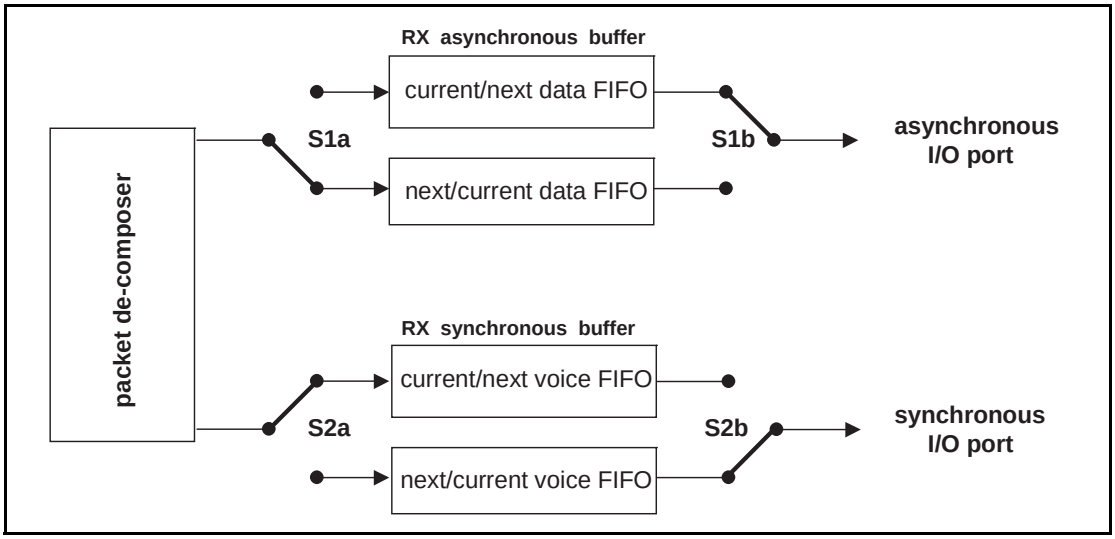


Figure 4.2: Functional diagram of RX buffering

Since the TYPE indication in the header (see Section 6.4.2 on page 115) of the received packet indicates whether the payload contains data and/or voice, the packet de-composer can automatically direct the traffic to the proper buffers. The switch S1 changes every time the Baseband Resource Manager reads the old register. If the next payload arrives before the RX register is emptied, a STOP indication is included in the packet header of the next TX packet that is returned. The STOP indication is removed again as soon as the RX register is emptied. The SEQN field is checked before a new ACL payload is stored into the asynchronous register (flush indication in LLID and broadcast messages influence the interpretation of the SEQN field see Section 7.6 on page 141).

The S2 switch is changed every T_{SCO} or T_{eSCO} for SCO and eSCO respectively. If, due to errors in the header, no new synchronous payload arrives, the switch still changes. The synchronous data processing unit then processes the synchronous data to account for the missing parts.

4.5.3 Flow Control

Since the RX ACL buffer can be full while a new payload arrives, flow control is required. The header field FLOW in the return TX packet may use STOP or GO in order to control the transmission of new data.

4.5.3.1 Destination control

As long as data cannot be received, a STOP indication shall be transmitted which is automatically inserted by the Link Controller into the header of the return packet. STOP shall be returned as long as the RX ACL buffer is not emptied by the Baseband Resource Manager. When new data can be accepted again, the GO indication shall be returned. GO shall be the default value. All packet types not including data can still be received. Voice communication for example is not affected by the flow control. Although a device cannot receive new information, it may still continue to transmit information: the flow control shall be separate for each direction.

4.5.3.2 Source control

On the reception of a STOP signal, the Link Controller shall automatically switch to the default packet type. The ACL packet transmitted just before the reception of the STOP indication shall be kept until a GO signal is received. It may be retransmitted as soon as a GO indication is received. Only default packets shall be sent as long as the STOP indication is received. When no packet is received, GO shall be assumed implicitly. Note that the default packets contain link control information (in the header) for the receive direction (which may still be open) and may contain synchronous data (**HV** or **EV** packets). When a GO indication is received, the Link Controller may resume transmitting the data that is present in the TX ACL buffers.

In a multi-slave configuration, only the transmission to the slave that issued the STOP signal shall be stalled. This means that the master shall only stop transmission from the TX ACL buffer corresponding to the slave that momentarily cannot accept data.

4.6 ACTIVE SLAVE BROADCAST TRANSPORT

The active slave broadcast logical transport is used to transport L2CAP user traffic to all devices in the piconet that are currently connected to the piconet physical channel that is used by the ASB. There is no acknowledgement protocol and the traffic is uni-directional from the piconet master to the slaves. The ASB logical transport may only be used for L2CAP group traffic and shall never be used for L2CAP connection-oriented channels, L2CAP control signaling or LMP control signaling.

The ASB logical transport is unreliable. To improve reliability somewhat each packet is transmitted a number of times. An identical sequence number is used to assist with filtering retransmissions at the slave device.

The ASB logical transport is identified by the reserved, all-zero, LT_ADDR. Packets on the ASB logical transport may be sent by the master at any time.

4.7 PARKED SLAVE BROADCAST TRANSPORT

The parked slave broadcast logical transport is used for communication from the master to the slaves that are parked. The PSB logical transport is more complex than the other logical transports as it consists of a number of phases, each having a different purpose. These phases are the control information phase (used to carry the LMP logical link), the user information phase (used to carry the L2CAP logical link), and the access phase (carrying baseband signaling).

The PSB logical transport is identified by the reserved, all-zero, LT_ADDR.

4.7.1 Parked Member Address (PM_ADDR)

A slave in the **PARK** state can be identified by its BD_ADDR or by a dedicated parked member address (PM_ADDR). This latter address is an 8-bit member address that separates the parked slaves. The PM_ADDR shall only be valid as long as the slave is parked. When the slave is activated it shall be assigned an LT_ADDR but shall lose the PM_ADDR. The PM_ADDR is assigned to the slave by the master during the parking procedure (see [Part C] Section 4.5.2 on page 286).

The all-zero PM_ADDR shall be reserved for parked slaves that only use their BD_ADDR to be unparked.

4.7.2 Access Request Address (AR_ADDR)

The access request address (AR_ADDR) is used by the parked slave to determine the slave-to-master half slot in the access window where it is allowed to send access request messages, see also Section 8.9.6 on page 190. The AR_ADDR shall be assigned to the slave when it enters the **PARK** state and shall only be valid as long as the slave is parked. The AR_ADDR is not necessarily unique; i.e. different parked slaves may have the same AR_ADDR.

5 LOGICAL LINKS

Five logical links are defined:

- Link Control (LC)
- ACL Control (ACL-C)
- User Asynchronous/Isochronous (ACL-U)
- User Synchronous (SCO-S)
- User Extended Synchronous (eSCO-S)

The control logical links LC and ACL-C are used at the link control level and link manager level, respectively. The ACL-U logical link is used to carry either asynchronous or isochronous user information. The SCO-S, and eSCO-S logical links are used to carry synchronous user information. The LC logical link is carried in the packet header, all other logical links are carried in the packet payload. The ACL-C and ACL-U logical links are indicated in the logical link ID, LLID, field in the payload header. The SCO-S and eSCO-S logical links are carried by the synchronous logical transports only; the ACL-U link is normally carried by the ACL logical transport; however, it may also be carried by the data in the DV packet on the SCO logical transport. The ACL-C link may be carried either by the SCO or the ACL logical transport.

5.1 LINK CONTROL LOGICAL LINK (LC)

The LC control logical link shall be mapped onto the packet header. This logical link carries low level link control information like ARQ, flow control, and payload characterization. The LC logical link is carried in every packet except in the **ID** packet which does not have packet header.

5.2 ACL CONTROL LOGICAL LINK (ACL-C)

The ACL-C logical link shall carry control information exchanged between the link managers of the master and the slave(s). The ACL-C logical link shall use DM1or DV packets. DV packets shall only be used on the ACL-C link if the ACL-C message is less than or equal to 9 bytes and an HV1 synchronous logical transport is in use. The ACL-C logical link is indicated by the LLID code 11 in the payload header.

5.3 USER ASYNCHRONOUS/ISOCHRONOUS LOGICAL LINK (ACL-U)

The ACL-U logical link shall carry L2CAP asynchronous and isochronous user data. These messages may be transmitted in one or more baseband packets. For fragmented messages, the start packet shall use an LLID code of 10 in the payload header. Remaining continuation packets shall use LLID code 01. If there is no fragmentation, all packets shall use the LLID start code 10.

5.3.1 Pausing the ACL-U logical link

When paused by LM, the Link Controller transmits the current packet with ACL-U information, if any, until an ACK is received or, optionally, until an explicit NACK is received. While the ACL-U logical link is paused, the Link Controller shall not transmit any packets with ACL-U logical link information.

If the ACL-U was paused after an ACK, the next sequence number shall be used on the next packet. If the ACL-U was paused after a NAK, the same sequence number shall be used on the next packet and the un-acknowledged packet shall be transmitted once the ACL-U logical link is un-paused.

When the ACL-U logical link is un-paused by LM, the Link Controller may resume transmitting packets with ACL-U information.

5.4 USER SYNCHRONOUS DATA LOGICAL LINK (SCO-S)

The SCO-S logical link carries transparent synchronous user data. This logical link is carried over the synchronous logical transport SCO.

5.5 USER EXTENDED SYNCHRONOUS DATA LOGICAL LINK (eSCO-S)

The eSCO-S logical link also carries transparent synchronous user data. This logical link is carried over the extended synchronous logical transport eSCO.

5.6 LOGICAL LINK PRIORITIES

The ACL-C logical link shall have a higher priority than the ACL-U logical link when scheduling traffic on the shared ACL logical transport, except in the case when retransmissions of unacknowledged ACL packets shall be given priority over traffic on the ACL-C logical link. The ACL-C logical link should also have priority over traffic on the SCO-S and eSCO-S logical links but opportunities for interleaving the logical links should be taken.

6 PACKETS

Bluetooth devices shall use the packets as defined in the following sections.

6.1 GENERAL FORMAT

6.1.1 Basic Rate

The general packet format of Basic Rate packets is shown in Figure 6.1 on page 108. Each packet consists of 3 entities: the access code, the header, and the payload. In the figure, the number of bits per entity is indicated.

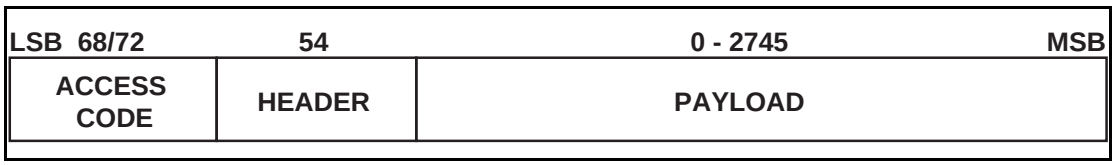


Figure 6.1: General Basic Rate packet format.

The access code is 72 or 68 bits and the header is 54 bits. The payload ranges from zero to a maximum of 2745 bits. Different packet types have been defined. Packet may consist of:

- the shortened access code only (see ID packet on page 116)
- the access code and the packet header
- the access code, the packet header and the payload.

6.1.2 Enhanced Data Rate

The general format of Enhanced Data Rate packets is shown in Figure 6.2 on page 108. The access code and packet header are identical in format and modulation to Basic Rate packets. Enhanced Data Rate packets have a guard time and synchronization sequence following the packet header. Following the payload are two trailer symbols. The guard time, synchronization sequence and trailer are defined in section 6.6.

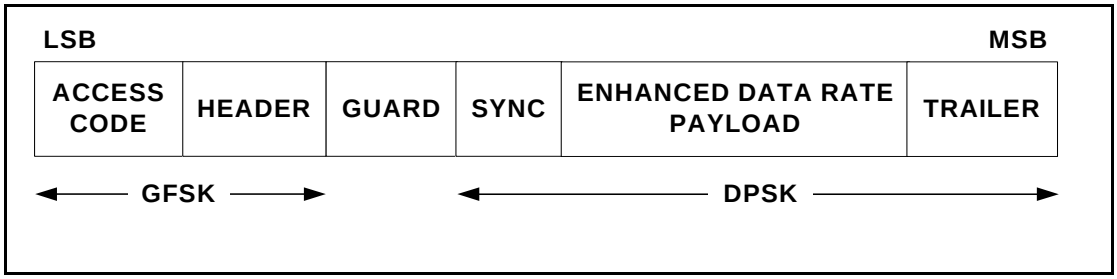


Figure 6.2: General enhanced data rate packet format

6.2 BIT ORDERING

The bit ordering when defining packets and messages in the *Baseband Specification*, follows the *Little Endian* format. The following rules apply:

- The *least significant bit* (LSB) corresponds to b_0;
- The LSB is the first bit sent over the air;
- In illustrations, the LSB is shown on the left side;

Furthermore, data fields generated internally at baseband level, such as the packet header fields and payload header length, shall be transmitted with the LSB first. For instance, a 3-bit parameter X=3 is sent as:

$$b_0 b_1 b_2 = 110$$

over the air where 1 is sent first and 0 is sent last.

6.3 ACCESS CODE

Every packet starts with an access code. If a packet header follows, the access code is 72 bits long, otherwise the access code is 68 bits long and is known as a shortened access code. The shortened access code does not contain a trailer. This access code is used for synchronization, DC offset compensation and identification. The access code identifies all packets exchanged on a physical channel: all packets sent in the same physical channel are preceded by the same access code. In the receiver of the device, a sliding correlator correlates against the access code and triggers when a threshold is exceeded. This trigger signal is used to determine the receive timing.

The shortened access code is used in paging, inquiry, and park. In this case, the access code itself is used as a signaling message and neither a header nor a payload is present.

The access code consists of a preamble, a sync word, and possibly a trailer, see Figure 6.3 on page 110. For details see Section 6.3.1 on page 110.

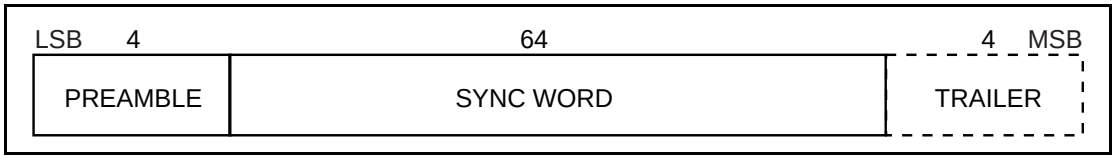


Figure 6.3: Access code format

6.3.1 Access Code Types

The different access code types use different Lower Address Parts (LAPs) to construct the sync word. The LAP field of the BD_ADDR is explained in Section 1.2 on page 68. A summary of the different access code types is in Table 6.1 on page 110.

Code type	LAP	Code length	Comments
CAC	Master	72	See also Section 1.3 on page 69
DAC	Paged device	68/72[1]	
GIAC	Reserved	68/72*	
DIAC	Dedicated	68/72*	

Table 6.1: Summary of access code types.

1. length 72 is only used in combination with FHS packets

The CAC consists of a **preamble**, **sync word**, and **trailer** and its total length is 72 bits. When used as self-contained messages without a header, the DAC and IAC do not include the trailer bits and are of length 68 bits.

6.3.2 Preamble

The preamble is a fixed zero-one pattern of 4 symbols used to facilitate DC compensation. The sequence is either 1010 or 0101, depending on whether the LSB of the following sync word is 1 or 0, respectively. The preamble is shown in Figure 6.4 on page 111.

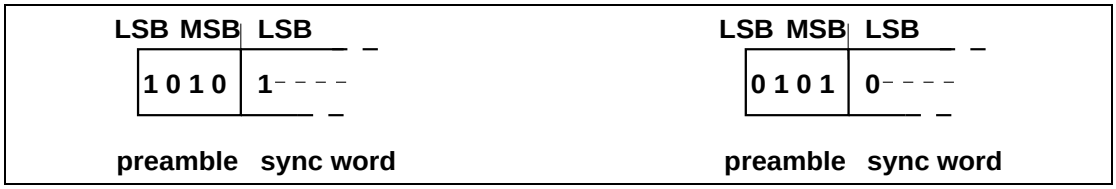


Figure 6.4: Preamble

6.3.3 Sync Word

The sync word is a 64-bit code word derived from a 24 bit address (LAP); for the CAC the master's LAP is used; for the GIAC and the DIAC, reserved, dedicated LAPs are used; for the DAC, the slave LAP is used. The construction guarantees large Hamming distance between sync words based on different LAPs. In addition, the good auto correlation properties of the sync word improve timing acquisition.

6.3.3.1 Synchronization word definition

The sync words are based on a (64,30) expurgated block code with an overlay (bit-wise XOR) of a 64 bit full length pseudo-random noise (PN) sequence. The expurgated code guarantees large Hamming distance ($d_{min} = 14$) between sync words based on different addresses. The PN sequence improves the auto correlation properties of the access code. The following steps describe how the sync word shall be generated:

1. Generate information sequence;
2. XOR this with the "information covering" part of the PN overlay sequence;
3. Generate the codeword;
4. XOR the codeword with all 64 bits of the PN overlay sequence;

The information sequence is generated by appending 6 bits to the 24 bit LAP (step 1). The appended bits are 001101 if the MSB of the LAP equals 0. If the MSB of the LAP is 1 the appended bits are 110010. The LAP MSB together with the appended bits constitute a length-seven Barker sequence. The purpose of including a Barker sequence is to further improve the auto correlation properties. In step 2 the information is pre-scrambled by XORing it with the bits $p_{34} \cdots p_{63}$ of the PN sequence (defined in Section 6.3.3.2 on page 114). After generating the codeword (step 3), the complete PN sequence is XORed to the

codeword (step 4). This step de-scrambles the information part of the codeword. At the same time the parity bits of the codeword are scrambled. Consequently, the original LAP and Barker sequence are ensured a role as a part of the access code sync word, and the cyclic properties of the underlying code is removed. The principle is depicted in Figure 6.5 on page 112.

In the following discussion, binary sequences will be denoted by their corresponding D-transform (in which D^i represents a delay of *i* time units). Let $p'(D) = p'_0 + p'_1 D + \ldots + p'_{62} D^{62}$ be the 63 bit PN sequence, where p'_0 is the first bit (LSB) leaving the PRNG (see Figure 6.6 on page 114), and, p'_{62} is the last bit (MSB). To obtain 64 bits, an extra zero is appended at the *end* of this sequence (thus, $p'(D)$ is unchanged). For notational convenience, the reciprocal of this extended polynomial, $p(D) = D^{63} p'(1/D)$, will be used in the following discussion. This is the sequence $p'(D)$ in reverse order. We denote the 24 bit lower address part (LAP) of the Bluetooth device address by $a(D) = a_0 + a_1 D + \ldots + a_{23} D^{23}$ (a_0 is the LSB of the Bluetooth device address).

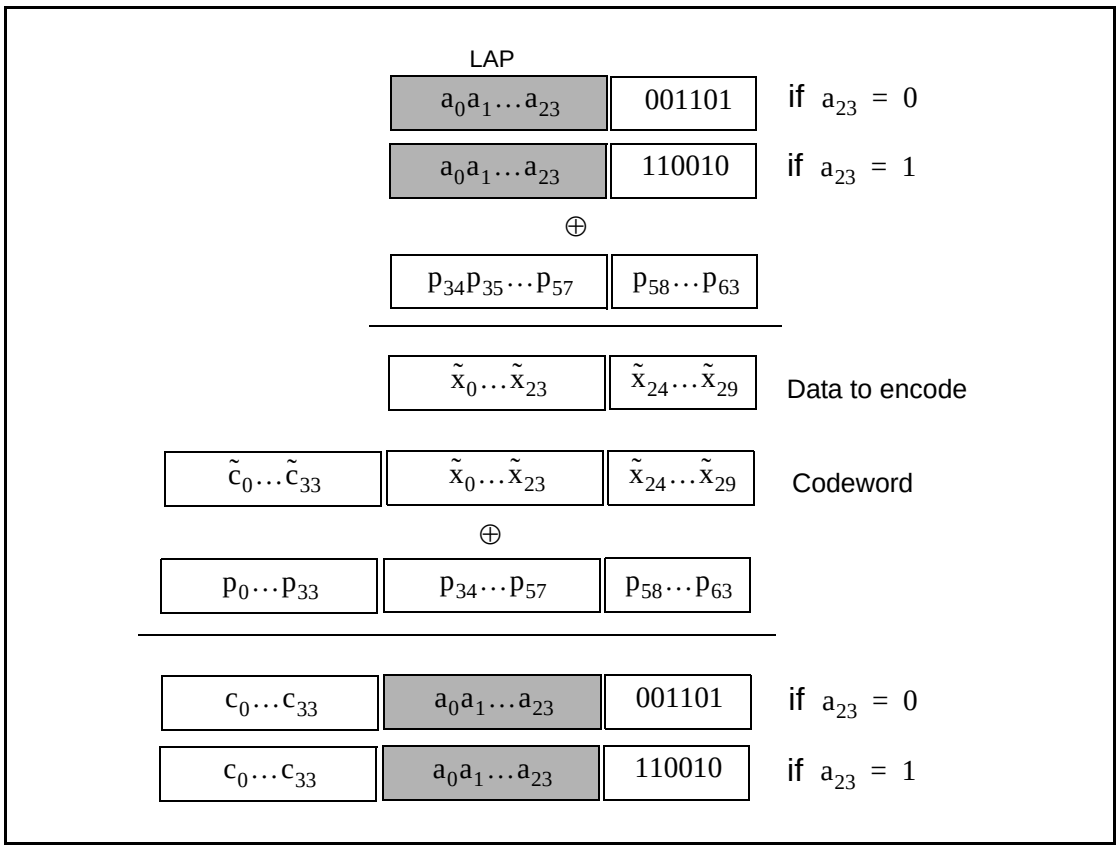


Figure 6.5: Construction of the sync word.

The (64,30) block code generator polynomial is denoted $g(D) = (1 + D)g'(D)$, where $g'(D)$ is the generator polynomial 157464165547 (octal notation) of a primitive binary (63,30) BCH code. Thus, in octal notation *g(D)* is

$$g(D) = 260534236651, \quad \text{(EQ 8)}$$

the left-most bit corresponds to the high-order (g_{34}) coefficient. The DC-free four bit sequences `0101` and `1010` can be written

$$\begin{cases} F_0(D) = D + D^3, \\ F_1(D) = 1 + D^2, \end{cases} \quad \text{(EQ 9)}$$

respectively. Furthermore,

$$\begin{cases} B_0(D) = D^2 + D^3 + D^5, \\ B_1(D) = 1 + D + D^4, \end{cases} \quad \text{(EQ 10)}$$

which are used to create the length seven Barker sequences. Then, the access code shall be generated by the following procedure:

1. Format the 30 information bits to encode:

$$x(D) = a(D) + D^{24}B_{a_{23}}(D).$$

2. Add the information covering part of the PN overlay sequence:

$$\tilde{x}(D) = x(D) + p_{34} + p_{35}D + \ldots + p_{63}D^{29}.$$

3. Generate parity bits of the (64,30) expurgated block code:[1]

$$\tilde{c}(D) = D^{34}\tilde{x}(D) \mod g(D).$$

4. Create the codeword:

$$\tilde{s}(D) = D^{34}\tilde{x}(D) + \tilde{c}(D).$$

5. Add the PN sequence:

$$s(D) = \tilde{s}(D) + p(D).$$

6. Append the (DC-free) preamble and trailer:

$$y(D) = F_{c_0}(D) + D^4 s(D) + D^{68}F_{a_{23}}(D).$$

1. $x(D)$ mod $y(D)$ denotes the remainder when $x(D)$ is divided by $y(D)$.

6.3.3.2 Pseudo-random noise sequence generation

To generate the PN sequence the primitive polynomial $h(D) = 1 + D + D^3 + D^4 + D^6$ shall be used. The LFSR and its starting state are shown in Figure 6.6 on page 114. The PN sequence generated (including the extra terminating zero) becomes (hexadecimal notation) 83848D96BBCC54FC. The LFSR output starts with the left-most bit of this PN sequence. This corresponds to $p'(D)$ of the previous section. Thus, using the reciprocal $p(D)$ as overlay gives the 64 bit sequence:

$$\mathbf{p} = 3F2A33DD69B121C1, \quad \text{(EQ 11)}$$

where the left-most bit is $p_0 = 0$ (there are two initial zeros in the binary representation of the hexadecimal digit 3), and $p_{63} = 1$ is the right-most bit.

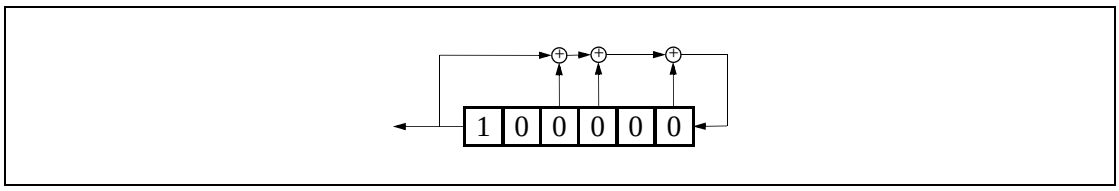


Figure 6.6: LFSR and the starting state to generate $p'(D)$.

6.3.4 Trailer

The trailer is appended to the sync word as soon as the packet header follows the access code. This is typically the case with the CAC, but the trailer is also used in the DAC and IAC when these codes are used in FHS packets exchanged during page response and inquiry response.

The trailer is a fixed zero-one pattern of four symbols. The trailer together with the three MSBs of the syncword form a 7-bit pattern of alternating ones and zeroes which can be used for extended DC compensation. The trailer sequence is either 1010 or 0101 depending on whether the MSB of the sync word is 0 or 1, respectively. The choice of trailer is illustrated in Figure 6.7 on page 114.



Figure 6.7: Trailer in CAC when MSB of sync word is 0 (a), and when MSB of sync word is 1 (b).

6.4 PACKET HEADER

The header contains link control (LC) information and consists of 6 fields:

- LT_ADDR: 3-bit logical transport address
- TYPE: 4-bit type code
- FLOW: 1-bit flow control
- ARQN: 1-bit acknowledge indication
- SEQN: 1-bit sequence number
- HEC: 8-bit header error check

The total header, including the HEC, consists of 18 bits, see Figure 6.8 on page 115, and is encoded with a rate 1/3 FEC (not shown but described in Section 7.4 on page 139) resulting in a 54-bit header. The LT_ADDR and TYPE fields shall be sent LSB first.

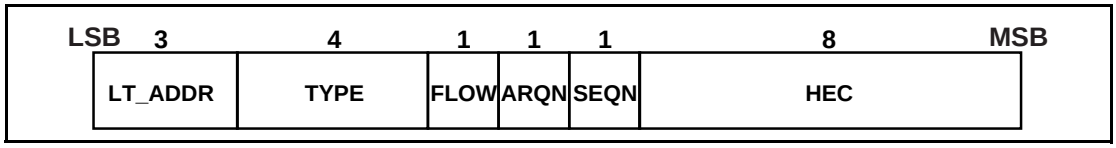


Figure 6.8: Header format.

6.4.1 LT_ADDR

The 3-bit LT_ADDR field contains the logical transport address for the packet (see Section 4.2 on page 97). This field indicates the destination slave for a packet in a master-to-slave transmission slot and indicates the source slave for a slave-to-master transmission slot.

6.4.2 TYPE

Sixteen different types of packets can be distinguished. The 4-bit TYPE code specifies which packet type is used. The interpretation of the TYPE code depends on the logical transport address in the packet. First, it shall be determined whether the packet is sent on an SCO logical transport, an eSCO logical transport, or an ACL logical transport. Second, it shall be determined whether Enhanced Data Rate has been enabled for the logical transport (ACL or eSCO) indicated by LT_ADDR. It can then be determined which type of SCO packet, eSCO packet, or ACL packet has been received. The TYPE code determines how many slots the current packet will occupy (see the slot occupancy column in Table 6.2 on page 118). This allows the non-addressed receivers to refrain from listening to the channel for the duration of the remaining slots. In Section 6.5 on page 117, each packet type is described in more detail.

6.4.3 FLOW

The FLOW bit is used for flow control of packets over the ACL logical transport. When the RX buffer for the ACL logical transport in the recipient is full, a STOP indication (FLOW=0) shall be returned to stop the other device from transmitting data temporarily. The STOP signal only affects ACL packets. Packets including only link control information (ID, POLL, and NULL packets), SCO packets or eSCO packets can still be received. When the RX buffer can accept data, a GO indication (FLOW=1) shall be returned. When no packet is received, or the received header is in error, a GO shall be assumed implicitly. In this case, the slave can receive a new packet with CRC although its RX buffer is still not emptied. The slave shall then return a NAK in response to this packet even if the packet passed the CRC check.

The FLOW bit is not used on the eSCO logical transport or the ACL-C logical link and shall be set to one on transmission and ignored upon receipt.

6.4.4 ARQN

The 1-bit acknowledgment indication ARQN is used to inform the source of a successful transfer of payload data with CRC, and can be positive acknowledge ACK or negative acknowledge NAK. See Section 7.6 on page 141 for initialization and usage of this bit.

6.4.5 SEQN

The SEQN bit provides a sequential numbering scheme to order the data packet stream. See Section 7.6.2 on page 144 for initialization and usage of the SEQN bit. For broadcast packets, a modified sequencing method is used, see Section 7.6.5 on page 148.

6.4.6 HEC

Each header has a header-error-check to check the header integrity. The HEC is an 8-bit word (generation of the HEC is specified in Section 7.1.1 on page 135). Before generating the HEC, the HEC generator is initialized with an 8-bit value. For FHS packets sent in **master response** substate, the slave upper address part (UAP) shall be used. For FHS packets and extended inquiry response packets sent in **inquiry response**, the default check initialization (DCI, see Section 1.2.1 on page 68) shall be used. In all other cases, the UAP of the master device shall be used.

After the initialization, a HEC shall be calculated for the 10 header bits. Before checking the HEC, the receiver shall initialize the HEC check circuitry with the proper 8-bit UAP (or DCI). If the HEC does not check, the entire packet shall be discarded. More information can be found in Section 7.1 on page 135.

6.5 PACKET TYPES

The packets used on the piconet are related to the logical transports they are used in. Three logical transports with distinct packet types are defined (see Section 4 on page 97): the SCO logical transport, the eSCO logical transport, and the ACL logical transport. For each of these logical transports, 15 different packet types can be defined.

To indicate the different packets on a logical transport, the 4-bit TYPE code is used. The packet types are divided into four segments. The first segment is reserved for control packets. All control packets occupy a single time slot. The second segment is reserved for packets occupying a single time slot. The third segment is reserved for packets occupying three time slots. The fourth segment is reserved for packets occupying five time slots. The slot occupancy is reflected in the segmentation and can directly be derived from the type code. Table 6.2 on page 118 summarizes the packets defined for the SCO, eSCO, and ACL logical transport types.

All packet types with a payload shall use GFSK modulation unless specified otherwise in the following sections.

ACL logical transports Enhanced Data Rate packet types are explicitly selected via LMP using the *packet_type_table* (ptt) parameter. eSCO Enhanced Data Rate packet types are selected when the eSCO logical transport is established.

.

Segment	TYPE code $b_3b_2b_1b_0$	Slot occupancy	SCO logical transport (1 Mbps)	eSCO logical transport (1 Mbps)	eSCO logical transport (2-3 Mbps)	ACL logical transport (1 Mbps) ptt=0	ACL logical transport (2-3 Mbps) ptt=1
1	0000	1	NULL	NULL	NULL	NULL	NULL
	0001	1	POLL	POLL	POLL	POLL	POLL
	0010	1	FHS	reserved	reserved	FHS	FHS
	0011	1	DM1	reserved	reserved	DM1	DM1
2	0100	1	undefined	undefined	undefined	DH1	2-DH1
	0101	1	HV1	undefined	undefined	undefined	undefined
	0110	1	HV2	undefined	2-EV3	undefined	undefined
	0111	1	HV3	EV3	3-EV3	undefined	undefined
	1000	1	DV	undefined	undefined	undefined	3-DH1
	1001	1	undefined	undefined	undefined	AUX1	AUX1
3	1010	3	undefined	undefined	undefined	DM3	2-DH3
	1011	3	undefined	undefined	undefined	DH3	3-DH3
	1100	3	undefined	EV4	2-EV5	undefined	undefined
	1101	3	undefined	EV5	3-EV5	undefined	undefined
4	1110	5	undefined	undefined	undefined	DM5	2-DH5
	1111	5	undefined	undefined	undefined	DH5	3-DH5

Table 6.2: Packets defined for synchronous and asynchronous logical transport types.

6.5.1 Common Packet Types

There are five common kinds of packets. In addition to the types listed in segment 1 of the previous table, the ID packet is also a common packet type but is not listed in segment 1 because it does not have a packet header.

6.5.1.1 ID Packet

The identity or ID packet consists of the device access code (DAC) or inquiry access code (IAC). It has a fixed length of 68 bits. It is a very robust packet since the receiver uses a bit correlator to match the received packet to the known bit sequence of the ID packet.

6.5.1.2 NULL Packet

The NULL packet has no payload and consists of the channel access code and packet header only. Its total (fixed) length is 126 bits. The NULL packet may be used to return link information to the source regarding the success of the previous transmission (ARQN), or the status of the RX buffer (FLOW). The NULL packet may not have to be acknowledged.

6.5.1.3 POLL Packet

The POLL packet is very similar to the NULL packet. It does not have a payload. In contrast to the NULL packet, it requires a confirmation from the recipient. It is not a part of the ARQ scheme. The POLL packet does not affect the ARQN and SEQN fields. Upon reception of a POLL packet the slave shall respond with a packet even when the slave does not have any information to send unless the slave has scatternet commitments in that timeslot. This return packet is an implicit acknowledgement of the POLL packet. This packet can be used by the master in a piconet to poll the slaves. Slaves shall not transmit the POLL packet.

6.5.1.4 FHS Packet

The FHS packet is a special control packet containing, among other things, the Bluetooth device address and the clock of the sender. The payload contains 144 information bits plus a 16-bit CRC code. The payload is coded with a rate 2/3 FEC with a gross payload length of 240 bits.

Figure 6.9 on page 119 illustrates the format and contents of the FHS payload. The payload consists of eleven fields. The FHS packet is used in page master response, inquiry response and in role switch.

The FHS packet contains real-time clock information. This clock information shall be updated before each retransmission. The retransmission of the FHS payload is different than retransmissions of ordinary data payloads where the same payload is used for each retransmission. The FHS packet is used for frequency hop synchronization before the piconet channel has been established, or when an existing piconet changes to a new piconet.

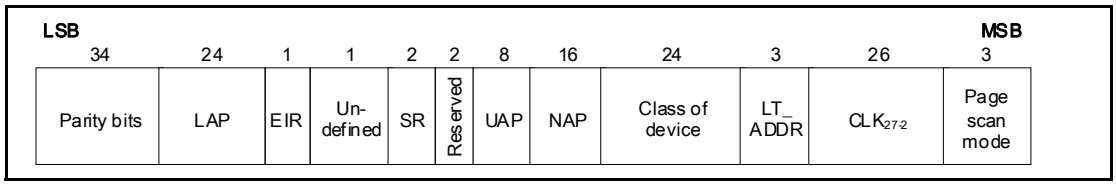


Figure 6.9: Format of the FHS payload.

Each field is described in more detail below:

Parity bits	This 34-bit field contains the parity bits that form the first part of the sync word of the access code of the device that sends the FHS packet. These bits are derived from the LAP as described in Section 1.2 on page 68.
LAP	This 24-bit field shall contain the lower address part of the device that sends the FHS packet.
EIR	This bit shall indicate that an extended inquiry response packet may follow. See Section 8.4.3 on page 163.
Undefined	This 1-bit field is reserved for future use and shall be set to zero.
SR	This 2-bit field is the scan repetition field and indicates the interval between two consecutive page scan windows, see also Table 6.4 and Table 8.1 on page 152
Reserved	This 2-bit field shall be set to 10.
UAP	This 8-bit field shall contain the upper address part of the device that sends the FHS packet.
NAP	This 16-bit field shall contain the non-significant address part of the device that sends the FHS packet (see also Section 1.2 on page 68 for LAP, UAP, and NAP).
Class of device	This 24-bit field shall contain the class of device of the device that sends the FHS packet. The field is defined in Bluetooth Assigned Numbers.
LT_ADDR	This 3-bit field shall contain the logical transport address the recipient shall use if the FHS packet is used at connection setup or role switch. A slave responding to a master or a device responding to an inquiry request message shall include an all-zero LT_ADDR field if it sends the FHS packet.
CLK_{27-2}	This 26-bit field shall contain the value of the native clock of the device that sends the FHS packet, sampled at the beginning of the transmission of the access code of this FHS packet. This clock value has a resolution of 1.25ms (two-slot interval). For each new transmission, this field is updated so that it accurately reflects the real-time clock value.
Page scan mode	This 3-bit field shall indicate which scan mode is used by default by the sender of the FHS packet. The interpretation of the page scan mode is illustrated in Table 6.5.

Table 6.3: Description of the FHS payload

The device sending the FHS shall set the SR bits according to Table 6.4.

SR bit format b_1b_0	SR mode
00	R0
01	R1
10	R2
11	reserved

Table 6.4: Contents of SR field

The device sending the FHS shall set the page scan mode bits according to Table 6.5.

Bit format $b_2b_1b_0$	Page scan mode
000	Mandatory scan mode
001	Reserved for future use
010	Reserved for future use
011	Reserved for future use
100	Reserved for future use
101	Reserved for future use
110	Reserved for future use
111	Reserved for future use

Table 6.5: Contents of page scan mode field

The LAP, UAP, and NAP together form the 48-bit Bluetooth Device Address of the device that sends the FHS packet. Using the parity bits and the LAP, the recipient can directly construct the channel access code of the sender of the FHS packet.

When initializing the HEC and CRC for the FHS packet of inquiry response, the UAP shall be the DCI.

6.5.1.5 DM1 Packet

DM1 is part of segment 1 in order to support control messages in any logical transport that allows the DM1 packet (see Table 6.2 on page 118). However, it may also carry regular user data. Since the DM1 packet can be regarded as an ACL packet, it will be discussed in Section 6.5.4 on page 125.

6.5.2 SCO Packets

HV and DV packets are used on the synchronous SCO logical transport. The HV packets do not include a CRC and shall not be retransmitted. DV packets include a CRC on the data section, but not on the synchronous data section. The data section of DV packets shall be retransmitted. SCO packets may be routed to the synchronous I/O port. Four packets are allowed on the SCO logical transport: HV1, HV2, HV3 and DV. These packets are typically used for 64kb/s speech transmission but may be used for transparent synchronous data.

6.5.2.1 HV1 Packet

The **HV1** packet has 10 information bytes. The bytes are protected with a rate 1/3 FEC. No CRC is present. The payload length is fixed at 240 bits. There is no payload header present.

6.5.2.2 HV2 Packet

The **HV2** packet has 20 information bytes. The bytes are protected with a rate 2/3 FEC. No CRC is present. The payload length is fixed at 240 bits. There is no payload header present.

6.5.2.3 HV3 Packet

The **HV3** packet has 30 information bytes. The bytes are not protected by FEC. No CRC is present. The payload length is fixed at 240 bits. There is no payload header present.

6.5.2.4 DV Packet

The DV packet is a combined data - voice packet. The DV packet shall only be used in place of an HV1 packet. The payload is divided into a voice field of 80 bits and a data field containing up to 150 bits, see Figure 6.10. The voice field is not protected by FEC. The data field has between 1 and 10 information bytes (including the 1-byte payload header) plus a 16-bit CRC code. The data field (including the CRC) is encoded with a rate 2/3 FEC. Since the **DV** packet has to be sent at regular intervals due to its synchronous contents, it is listed under the SCO packet types. The voice and data fields shall be treated separately. The voice field shall be handled in the same way as normal SCO data and shall never be retransmitted; that is, the voice field is always new. The data field is checked for errors and shall be retransmitted if necessary. When the asynchronous data field in the DV packet has not be acknowledged before the SCO logical transport is terminated, the asynchronous data field shall be retransmitted in a DM1 packet.

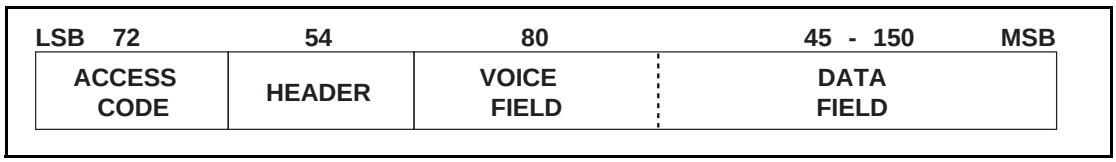


Figure 6.10: DV packet format

6.5.3 eSCO Packets

EV packets are used on the synchronous eSCO logical transport. The packets include a CRC and retransmission may be applied if no acknowledgement of proper reception is received within the retransmission window. eSCO packets may be routed to the synchronous I/O port. Three eSCO packet types (EV3, EV4, EV5) are defined for Basic Rate operation and four additional eSCO packet types (2-EV3, 3-EV3, 2-EV5, 3-EV5) for Enhanced Data Rate operation. The eSCO packets may be used for 64kb/s speech transmission as well as transparent data at 64kb/s and other rates.

6.5.3.1 EV3 Packet

The **EV3** packet has between 1 and 30 information bytes plus a 16-bit CRC code. The bytes are not protected by FEC. The EV3 packet may cover up to a single time slot. There is no payload header present. The payload length is set during the LMP eSCO setup and remains fixed until the link is removed or re-negotiated.

6.5.3.2 EV4 Packet

The **EV4** packet has between 1 and 120 information bytes plus a 16-bit CRC code. The EV4 packet may cover to up three time slots. The information plus CRC bits are coded with a rate 2/3 FEC. There is no payload header present. The payload length is set during the LMP eSCO setup and remains fixed until the link is removed or re-negotiated.

6.5.3.3 EV5 Packet

The **EV5** packet has between 1 and 180 information bytes plus a 16-bit CRC code. The bytes are not protected by FEC. The EV5 packet may cover up to three time slots. There is no payload header present. The payload length is set during the LMP eSCO setup and remains fixed until the link is removed or re-negotiated.

6.5.3.4 2-EV3 Packet

The **2-EV3** packet is similar to the EV3 packet except that the payload is modulated using π/4-DQPSK. It has between 1 and 60 information bytes plus a 16-

bit CRC code. The bytes are not protected by FEC. The 2-EV3 packet covers a single time slot. There is no payload header present. The payload length is set during the LMP eSCO setup and remains fixed until the link is removed or re-negotiated.

6.5.3.5 2-EV5 Packet

The **2-EV5** packet is similar to the EV5 packet except that the payload is modulated using π/4-DQPSK. It has between 1 and 360 information bytes plus a 16-bit CRC code. The bytes are not protected by FEC. The 2-EV5 packet may cover up to three time slots. There is no payload header present. The payload length is set during the LMP eSCO setup and remains fixed until the link is removed or re-negotiated.

6.5.3.6 3-EV3 Packet

The **3-EV3** packet is similar to the EV3 packet except that the payload is modulated using 8DPSK. It has between 1 and 90 information bytes plus a 16-bit CRC code. The bytes are not protected by FEC. The 3-EV3 packet covers a single time slot. There is no payload header present. The payload length is set during the LMP eSCO setup and remains fixed until the link is removed or re-negotiated.

6.5.3.7 3-EV5 Packet

The **3-EV5** packet is similar to the EV5 packet except that the payload is modulated using 8DPSK. It has between 1 and 540 information bytes plus a 16-bit CRC code. The bytes are not protected by FEC. The 3-EV5 packet may cover up to three time slots. There is no payload header present. The payload length is set during the LMP eSCO setup and remains fixed until the link is removed or re-negotiated.

6.5.4 ACL Packets

ACL packets are used on the asynchronous logical transport. The information carried may be user data or control data.

Seven packet types are defined for Basic Rate operation: DM1, DH1, DM3, DH3, DM5, DH5 and AUX1. Six additional packets are defined for Enhanced Data Rate operation: 2-DH1, 3-DH1, 2-DH3, 3-DH3, 2-DH5 and 3-DH5.

6.5.4.1 DM1 Packet

The DM1 packet carries data information only. The payload has between 1 and 18 information bytes (including the 1-byte payload header) plus a 16-bit CRC code. The DM1 packet occupies a single time slot. The information plus CRC bits are coded with a rate 2/3 FEC. The payload header in the DM1 packet is 1 byte long, see Figure 6.12 on page 129. The length indicator in the payload header specifies the number of user bytes (excluding payload header and the CRC code).

6.5.4.2 DH1 Packet

This packet is similar to the DM1 packet, except that the information in the payload is not FEC encoded. As a result, the DH1 packet has between 1 and 28 information bytes (including the 1-byte payload header) plus a 16-bit CRC code. The DH1 packet occupies a single time slot.

6.5.4.3 DM3 Packet

The DM3 packet may occupy up to three time slots. The payload has between 2 and 123 information bytes (including the 2-byte payload header) plus a 16-bit CRC code. The information plus CRC bits are coded with a rate 2/3 FEC. The payload header in the DM3 packet is 2 bytes long, see Figure 6.13 on page 129. The length indicator in the payload header specifies the number of user bytes (excluding payload header and the CRC code).

6.5.4.4 DH3 Packet

This packet is similar to the DM3 packet, except that the information in the payload is not FEC encoded. As a result, the DH3 packet has between 2 and 185 information bytes (including the 2-byte payload header) plus a 16-bit CRC code. The DH3 packet may occupy up to three time slots.

6.5.4.5 DM5 Packet

The DM5 packet may occupy up to five time slots. The payload has between 2 and 226 information bytes (including the 2-byte payload header) plus a 16-bit

CRC code. The payload header in the DM5 packet is 2 bytes long. The information plus CRC bits are coded with a rate 2/3 FEC. The length indicator in the payload header specifies the number of user bytes (excluding payload header and the CRC code).

6.5.4.6 DH5 Packet

This packet is similar to the DM5 packet, except that the information in the payload is not FEC encoded. As a result, the DH5 packet has between 2 and 341 information bytes (including the 2-byte payload header) plus a 16-bit CRC code. The DH5 packet may occupy up to five time slots.

6.5.4.7 AUX1 Packet

This packet resembles a DH1 packet but has no CRC code. The AUX1 packet has between 1 and 30 information bytes (including the 1-byte payload header). The AUX1 packet occupies a single time slot. The AUX1 packet shall not be used for the ACL-U or ACL-C logical links. An AUX1 packet may be discarded.

6.5.4.8 2-DH1 packet

This packet is similar to the DH1 packet except that the payload is modulated using π/4-DQPSK. The 2-DH1 packet has between 2 and 56 information bytes (including the 2-byte payload header) plus a 16-bit CRC code. The 2-DH1 packet occupies a single time slot.

6.5.4.9 2-DH3 packet

This packet is similar to the DH3 packet except that the payload is modulated using π/4-DQPSK. The 2-DH3 packet has between 2 and 369 information bytes (including the 2-byte payload header) plus a 16-bit CRC code. The 2-DH3 packet may occupy up to three time slots.

6.5.4.10 2-DH5 packet

This packet is similar to the DH5 packet except that the payload is modulated using π/4-DQPSK. The 2-DH5 packet has between 2 and 681 information bytes (including the 2-byte payload header) plus a 16-bit CRC code. The 2-DH5 packet may occupy up to five time slots.

6.5.4.11 3-DH1 packet

This packet is similar to the DH1 packet except that the payload is modulated using 8DPSK. The 3-DH1 packet has between 2 and 85 information bytes

(including the 2-byte payload header) plus a 16-bit CRC code. The 3-DH1 packet occupies a single time slot.

6.5.4.12 3-DH3 packet

This packet is similar to the DH3 packet except that the payload is modulated using 8DPSK. The 3-DH3 packet has between 2 and 554 information bytes (including the 2-byte payload header) plus a 16-bit CRC code. The 3-DH3 packet may occupy up to three time slots.

6.5.4.13 3-DH5 packet

This packet is similar to the DH5 packet except that the payload is modulated using 8DPSK. The 3-DH5 packet has between 2 and 1023 information bytes (including the 2-byte payload header) plus a 16-bit CRC code. The 3-DH5 packet may occupy up to five time slots.

6.6 PAYLOAD FORMAT

In the payload, two fields are distinguished: the synchronous data field and the asynchronous data field. The ACL packets only have the asynchronous data field and the SCO and eSCO packets only have the synchronous data field – with the exception of the DV packets which have both.

6.6.1 Synchronous Data Field

In SCO, which is only supported in Basic Rate mode, the synchronous data field has a fixed length and consists only of the synchronous data body portion. No payload header is present.

In Basic Rate eSCO, the synchronous data field consists of two segments: a synchronous data body and a CRC code. No payload header is present.

In Enhanced Data Rate eSCO, the synchronous data field consists of five segments: a guard time, a synchronization sequence, a synchronous data body, a CRC code and a trailer. No payload header is present.

1. Enhanced Data Rate Guard Time

For Enhanced Data Rate packets the guard time is defined as the period starting at the end of the last GFSK symbol of the header and ending at the start of the reference symbol of the synchronization sequence. The length of the guard time shall be between 4.75 µsec and 5.25 µsec.

2. Enhanced Data Rate Synchronization Sequence

For Enhanced Data Rate packets the symbol timing at the start of the synchronization sequence shall be within ±¼ µsec of the symbol timing of the last GFSK symbol of the packet header. The length of the synchronization sequence is 11 µsec (11 DPSK symbols) and consists of a reference symbol (with arbitrary phase) followed by ten DPSK symbols.

The phase changes between the DPSK symbols (shown in Synchronization sequence) shall be

$$\{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5, \varphi_6, \varphi_7, \varphi_8, \varphi_9, \varphi_{10}\} =$$
$$\{3\pi/4, -3\pi/4, 3\pi/4, -3\pi/4, 3\pi/4, -3\pi/4, -3\pi/4, 3\pi/4, 3\pi/4, 3\pi/4\} \quad \text{(EQ 12)}$$

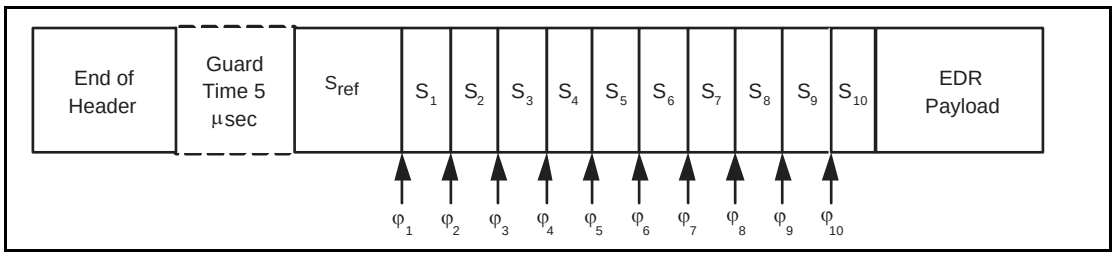


Figure 6.11: Synchronization sequence

Sref is the reference symbol. φ_1 is the phase change between the reference symbol and the first DPSK symbol S_1. φ_k is the phase change between the k-1th symbol S_{k-1} and the kth symbol S_k

Note: the synchronization sequence may be generated using the modulator by pre-pending the data with bits that generate the synchronization sequence.

For $\pi/4$-DQPSK, the bit sequence used to generate the synchronization sequence is 0,1,1,1,0,1,1,1,0,1,1,1,1,1,0,1,0,1,0,1.

For 8DPSK, the bit sequence used to generate the synchronization sequence is 0,1,0,1,1,1,0,1,0,1,1,1,0,1,0,1,1,1,1,1,1,1,0,1,0,0,1,0,0,1,0.

3. Synchronous data body

For HV and DV packets, the synchronous data body length is fixed. For EV packets, the synchronous data body length is negotiated during the LMP eSCO setup. Once negotiated, the synchronous data body length remains constant unless re-negotiated. The synchronous data body length may be different for each direction of the eSCO logical transport.

4. CRC code

The 16-bit CRC in the payload is generated as specified in Section 7.1 on page 135. The 8-bit UAP of the master is used to initialize the CRC generator.

Only the Synchronous data body segment is used to generate the CRC code.

5. Enhanced Data Rate Trailer

For Enhanced Data Rate packets, two trailer symbols shall be added to the end of the payload. The trailer bits shall be all zero, i.e. {00, 00} for the π/4-DQPSK and {000, 000} for the 8DPSK.

6.6.2 Asynchronous Data Field

Basic rate ACL packets have an asynchronous data field consisting of two or three segments: a payload header, a payload body, and possibly a CRC code (the AUX1 packet does not carry a CRC code).

Enhanced Data Rate ACL packets have an asynchronous data field consisting of six segments: a guard time, a synchronization sequence, a payload header, a payload body, a CRC and a trailer.

1. Enhanced Data Rate Guard time

This is the same as defined for the Synchronous data field in Section 6.6.1.

2. Enhanced Data Rate Synchronization sequence

This is the same as defined for the Synchronous data field in Section 6.6.1.

3. Payload header

The payload header is one or two bytes long. Basic rate packets in segments one and two have a 1-byte payload header; Basic Rate packets in segments three and four and all Enhanced Data Rate packets have a 2-byte payload header. The payload header specifies the logical link (2-bit LLID indication), controls the flow on the logical channels (1-bit FLOW indication), and has a payload length indicator (5 bits and 10 bits for 1-byte and 2-byte payload headers, respectively). In the case of a 2-byte payload header, the length indicator is extended by five bits into the next byte. The remaining three bits of the second byte are reserved for future use and shall be set to zero. The formats of the 1-byte and 2-byte payload headers are shown in Figure 6.12 on page 129 and Figure 6.13 on page 129.

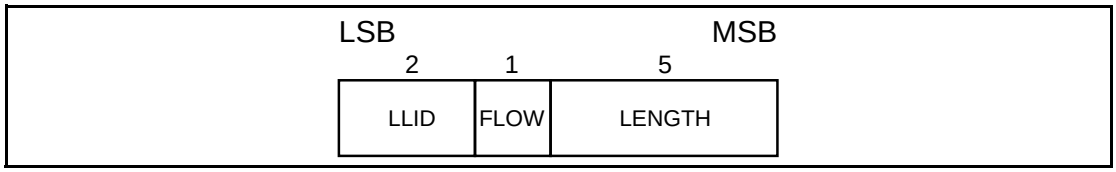


Figure 6.12: Payload header format for Basic Rate single-slot ACL packets.

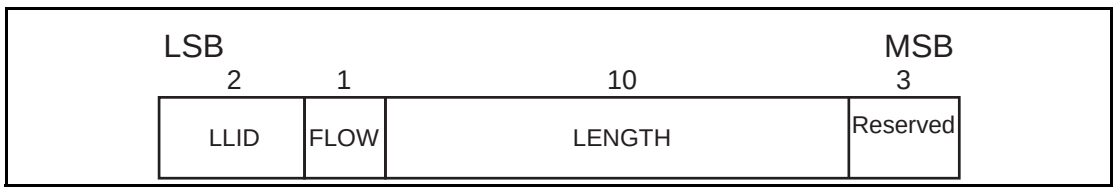


Figure 6.13: Payload header format for multi-slot ACL packets and all EDR ACL packets.

The LLID field shall be transmitted first, the length field last. In Table 6.6 on page 130, more details about the contents of the LLID field are listed.

LLID code b_1b_0	**Logical Link**	**Information**
00	NA	undefined
01	ACL-U	Continuation fragment of an L2CAP message
10	ACL-U	Start of an L2CAP message or no fragmentation
11	ACL-C	LMP message

Table 6.6: Logical link LLID field contents

An L2CAP message may be fragmented into several packets. Code 10 shall be used for an ACL-U packet carrying the first fragment of such a message; code 01 shall be used for continuing fragments. The first fragment of an L2CAP message sent over HCI is identified by having a Packet_Boundary_Flag value of 00 or 10 both of which are mapped to LLID code 10. If there is no fragmentation, code 10 shall be used for every packet. Code 11 shall be used for LMP messages. Code 00 is reserved for future use.

The flow indicator in the payload is used to control the flow at the L2CAP level. It is used to control the flow per logical link. FLOW=1 means flow-on (GO) and FLOW=0 means flow-off (STOP). After a new connection has been established the flow indicator shall be set to GO. When a device receives a payload header with the flow bit set to STOP, it shall stop the transmission of ACL-U packets before an additional amount of payload data is sent. This amount is defined as the flow control lag, expressed as a number of bytes. The shorter the flow control lag, the less buffering the other device must dedicate to this function. The flow control lag shall not exceed 1792 bytes (7×256 bytes). In order to allow devices to optimize the selection of packet length and buffer space, the flow control lag of a given implementation shall be provided in the LMP_features_res message.

If a packet containing the payload flow bit of STOP is received, with a valid packet header but bad payload, the payload flow control bit shall be ignored. The baseband ACK contained in the packet header will be received and a further ACL packet may be transmitted. Each occurrence of this situation allows a further ACL packet to be sent in spite of the flow control request being sent via the payload header flow control bit. It is recommended that devices that use the payload header flow bit should ensure that no further ACL packets are sent until the payload flow bit has been correctly received. This can be accomplished by simultaneously turning on the flow bit in the packet header and keeping it on until an ACK is received back (ARQN=1). This will typically be only one round trip time. Since they lack a payload CRC, AUX1 packets should not be used with a payload flow bit of STOP.

The Baseband Resource Manager is responsible for setting and processing the flow bit in the payload header. Real-time flow control shall be carried out at the packet level by the link controller via the flow bit in the packet header (see Section 6.4.3 on page 116). With the payload flow bit, traffic from the remote end can be controlled. It is allowed to generate and send an ACL packet with payload length zero irrespective of flow status. L2CAP start-fragment and continue-fragment indications (LLID=10 and LLID=01) also retain their meaning when the payload length is equal to zero (i.e. an empty start-fragment shall not be sent in the middle of an on-going ACL-U packet transmission). It is always safe to send an ACL packet with length=0 and LLID=01. The payload flow bit has its own meaning for each logical link (ACL-U or ACL-C), see Table 6.7 on page 131. On the ACL-C logical link, no flow control is applied and the payload FLOW bit shall always be set to one.

LLID code b_1b_0	**Usage and semantics of the ACL payload header FLOW bit**
00	Not defined, reserved for future use.
01 or 10	Flow control of the ACL-U channel (L2CAP messages)
11	Always set FLOW=1 on transmission and ignore the bit on reception

Table 6.7: Use of payload header flow bit on the logical links.

The length indicator shall be set to the number of bytes (i.e. 8-bit words) in the payload excluding the payload header and the CRC code; i.e. the payload body only. With reference to Figure 6.12 and Figure 6.13, the MSB of the length field in a 1-byte header is the last (right-most) bit in the payload header; the MSB of the length field in a 2-byte header is the fourth bit to the left from the right-most end of the second byte in the payload header.

4. Payload body

The payload body includes the user information and determines the effective user throughput. The length of the payload body is indicated in the length field of the payload header.

5. CRC code generation

The 16-bit cyclic redundancy check code in the payload is generated as specified in Section 7.1 on page 135. Before determining the CRC code, an 8-bit value is used to initialize the CRC generator. For the CRC code in the FHS packets sent in **master response** substate, the UAP of the slave is used. For the FHS packet and the extended inquiry response packet sent in **inquiry response** substate, the DCI (see Section 1.2.1 on page 68) is used. For all other packets, the UAP of the master is used.

Only the Payload header and Payload body segments are used to generate the CRC code.

6. Enhanced Data Rate Trailer

This is the same as defined for the Synchronous data field in section 6.6.1.

6.7 PACKET SUMMARY

A summary of the packets and their characteristics is shown in Table 6.8, Table 6.9 and Table 6.10. The payload represents the packet payload excluding FEC, CRC, and payload header.

Type	Payload (bytes)	FEC	CRC	Symmetric Max. Rate	Asymmetric Max. Rate
ID	na	na	na	na	na
NULL	na	na	na	na	na
POLL	na	na	na	na	na
FHS	18	2/3	yes	na	na

Table 6.8: Link control packets

Type	Payload Header (bytes)	User Payload (bytes)	FEC	CRC	Symmetric Max. Rate (kb/s)	Asymmetric Max. Rate (kb/s)	
						Forward	Reverse
DM1	1	0-17	2/3	yes	108.8	108.8	108.8
DH1	1	0-27	no	yes	172.8	172.8	172.8
DM3	2	0-121	2/3	yes	258.1	387.2	54.4
DH3	2	0-183	no	yes	390.4	585.6	86.4
DM5	2	0-224	2/3	yes	286.7	477.8	36.3
DH5	2	0-339	no	yes	433.9	723.2	57.6
AUX1	1	0-29	no	no	185.6	185.6	185.6
2-DH1	2	0-54	no	yes	345.6	345.6	345.6
2-DH3	2	0-367	no	yes	782.9	1174.4	172.8
2-DH5	2	0-679	no	yes	869.1	1448.5	115.2
3-DH1	2	0-83	no	yes	531.2	531.2	531.2
3-DH3	2	0-552	no	yes	1177.6	1766.4	235.6
3-DH5	2	0-1021	no	yes	1306.9	2178.1	177.1

Table 6.9: ACL packets

Type	Payload Header (bytes)	User Payload (bytes)	FEC	CRC	Symmetric Max. Rate (kb/s)
HV1	na	10	1/3	no	64.0
HV2	na	20	2/3	no	64.0
HV3	na	30	no	no	64.0
DV[1]	1 D	10+(0-9) D	2/3 D	yes D	64.0+57.6 D
EV3	na	1-30	No	Yes	96
EV4	na	1-120	2/3	Yes	192
EV5	na	1-180	No	Yes	288
2-EV3	na	1-60	No	Yes	192
2-EV5	na	1-360	No	Yes	576
3-EV3	na	1-90	No	Yes	288
3-EV5	na	1-540	No	Yes	864

Table 6.10: Synchronous packets

1. Items followed by ‘D’ relate to data field only.

7 BITSTREAM PROCESSING

Bluetooth devices shall use the bitstream processing schemes as defined in the following sections.

Before the payload is sent over the air interface, several bit manipulations are performed in the transmitter to increase reliability and security. An HEC is added to the packet header, the header bits are scrambled with a whitening word, and FEC coding is applied. In the receiver, the inverse processes are carried out. Figure 7.1 on page 134 shows the processes carried out for the packet header both at the transmit and the receive side. All header bit processes are mandatory.

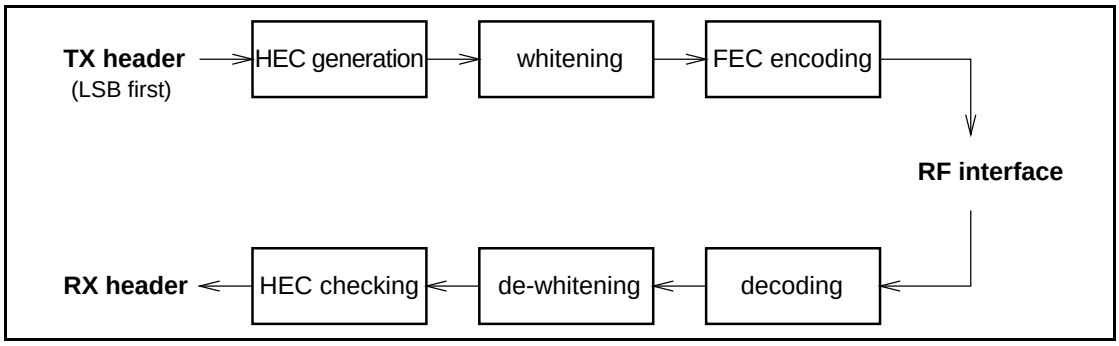


Figure 7.1: Header bit processes.

Figure 7.2 on page 134 shows the processes that may be carried out on the payload. In addition to the processes defined for the packet header, encryption can be applied on the payload. Only whitening and de-whitening, as explained in Section 7.2 on page 138, are mandatory for every payload; all other processes are optional and depend on the packet type (see Section 6.6 on page 127) and whether encryption is enabled. In Figure 7.2 on page 134, optional processes are indicated by dashed blocks.

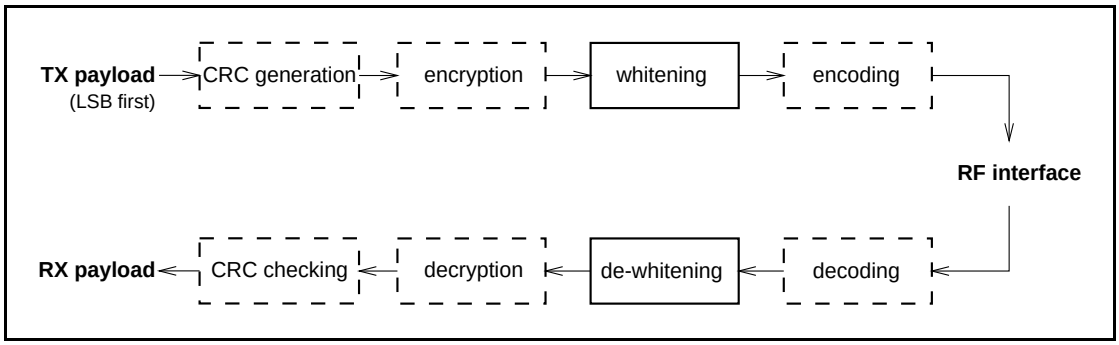


Figure 7.2: Payload bit processes.

7.1 ERROR CHECKING

The packet can be checked for errors or wrong delivery using the channel access code, the HEC in the header, and the CRC in the payload. At packet reception, the access code is checked first. Since the 64-bit sync word in the channel access code is derived from the 24-bit master LAP, this checks if the LAP is correct, and prevents the receiver from accepting a packet of another piconet (provided the LAP field of the master's BD_ADDR is different).

The HEC and CRC computations are normally initialized with the UAP of the master. Even though the access code may be the same for two piconets the different UAP values will typically cause the HEC and CRC to fail. However, there is an exception where no common UAP is available in the transmitter and receiver. This is the case when the HEC and CRC are generated for the FHS packet and the extended inquiry response packet in inquiry response substate. In this case the DCI value shall be used.

The generation and check of the HEC and CRC are summarized in Figure 7.5 on page 136 and Figure 7.8 on page 137. Before calculating the HEC or CRC, the shift registers in the HEC/CRC generators shall be initialized with the 8-bit UAP (or DCI) value. Then the header and payload information shall be shifted into the HEC and CRC generators, respectively (with the LSB first).

7.1.1 HEC Generation

The HEC generating LFSR is depicted in Figure 7.3 on page 135. The generator polynomial is

$g(D) = (D + 1)(D^7 + D^4 + D^3 + D^2 + 1) = D^8 + D^7 + D^5 + D^2 + D + 1$. Initially this circuit shall be pre-loaded with the 8-bit UAP such that the LSB of the UAP (denoted UAP_0) goes to the left-most shift register element, and, UAP_7 goes to the right-most element. The initial state of the HEC LFSR is depicted in Figure 7.4 on page 136. Then the data shall be shifted in with the switch S set in position 1. When the last data bit has been clocked into the LFSR, the switch S shall be set in position 2, and, the HEC can be read out from the register. The LFSR bits shall be read out from right to left (i.e., the bit in position 7 is the first to be transmitted, followed by the bit in position 6, etc.).

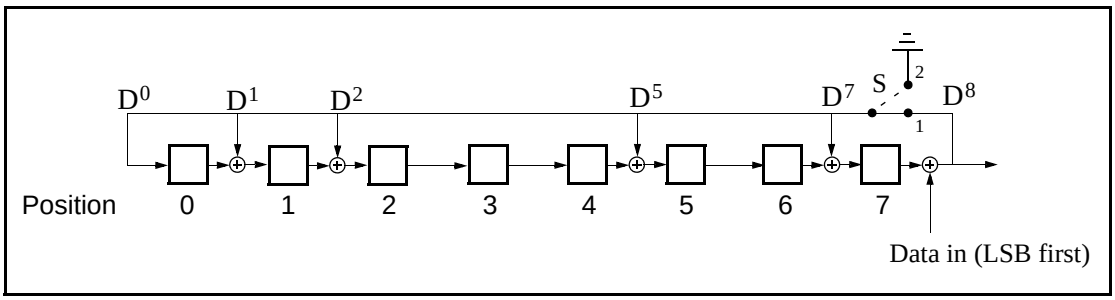


Figure 7.3: The LFSR circuit generating the HEC.

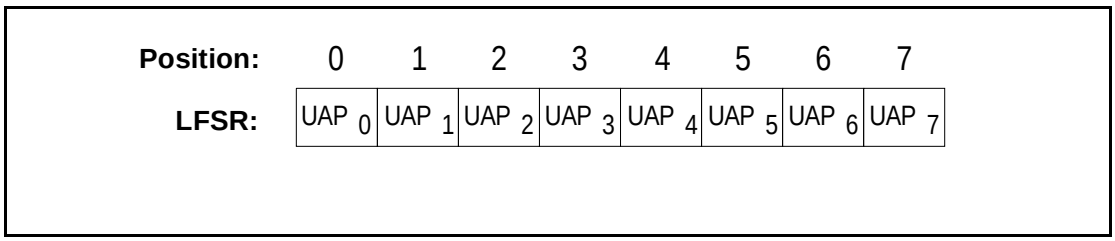


Figure 7.4: Initial state of the HEC generating circuit.

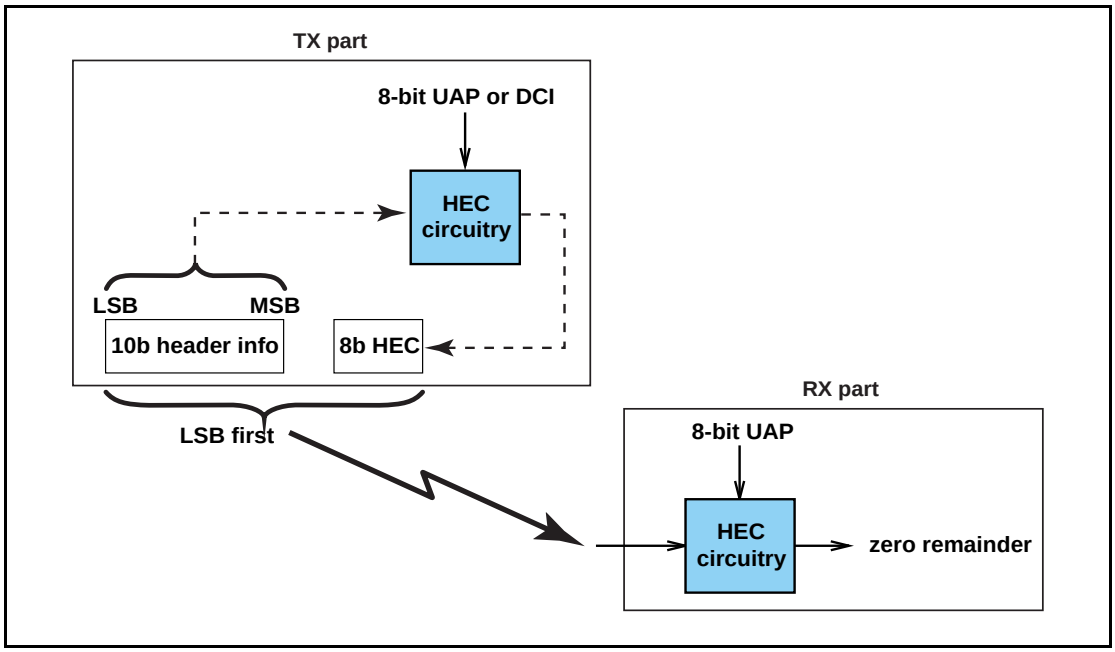


Figure 7.5: HEC generation and checking.

7.1.2 CRC Generation

The 16 bit LFSR for the CRC is constructed similarly to the HEC using the CRC-CCITT generator polynomial $g(D) = D^{16} + D^{12} + D^5 + 1$ (i.e. 210041 in octal representation) (see Figure 7.6 on page 137). For this case, the 8 left-most bits shall be initially loaded with the 8-bit UAP (UAP_0 to the left and UAP_7 to the right) while the 8 right-most bits shall be reset to zero. The initial state of the 16 bit LFSR is specified in Figure 7.7 on page 137. The switch S shall be set in position 1 while the data is shifted in. After the last bit has entered the LFSR, the switch shall be set in position 2, and, the register's contents shall be transmitted, from right to left (i.e., starting with position 15, then position 14, etc.).

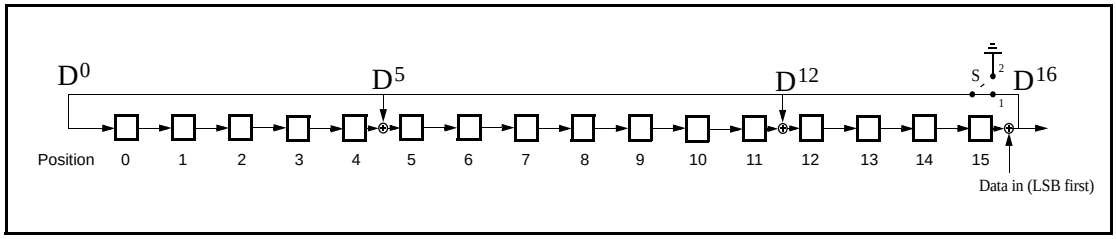


Figure 7.6: The LFSR circuit generating the CRC.

Position:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
LFSR:	UAP_0	UAP_1	UAP_2	UAP_3	UAP_4	UAP_5	UAP_6	UAP_7	0	0	0	0	0	0	0	0

Figure 7.7: Initial state of the CRC generating circuit.

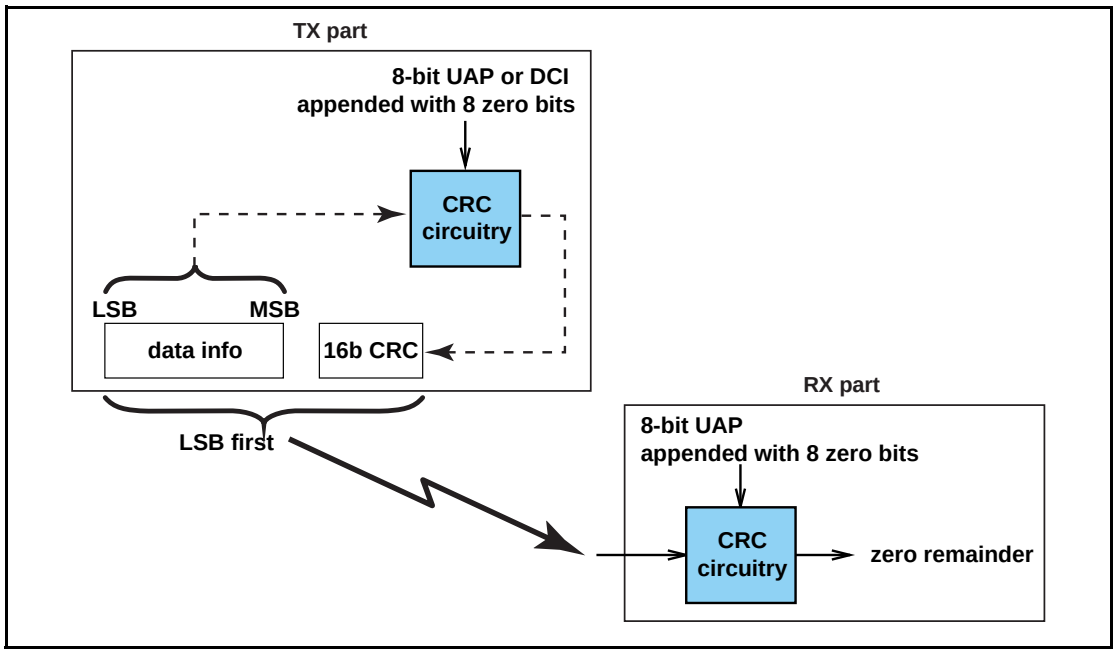


Figure 7.8: CRC generation and checking.

7.2 DATA WHITENING

Before transmission, both the header and the payload shall be scrambled with a data whitening word in order to randomize the data from highly redundant patterns and to minimize DC bias in the packet. The scrambling shall be performed prior to the FEC encoding.

At the receiver, the received data shall be descrambled using the same whitening word generated in the recipient. The descrambling shall be performed after FEC decoding.

The whitening word is generated with the polynomial $g(D) = D^7 + D^4 + 1$ (i.e., 221 in octal representation) and shall be subsequently XORed with the header and the payload. The whitening word is generated with the linear feedback shift register shown in Figure 7.9 on page 138. Before each transmission, the shift register shall be initialized with a portion of the master Bluetooth clock, CLK6-1, extended with an MSB of value one. This initialization shall be carried out with CLK1 written to position 0, CLK2 written to position 1, etc. Exceptions are the FHS packet sent during inquiry response or page response, and the extended inquiry response packet sent during inquiry response, where initialization of the whitening register shall be carried out differently. Instead of the master clock, the X-input used in the inquiry or page response (depending on current state) routine shall be used, see Table 2.2. The 5-bit value shall be extended with two MSBs of value 1. During register initialization, the LSB of X (i.e., X0) shall be written to position 0, X1 shall be written to position 1, etc.

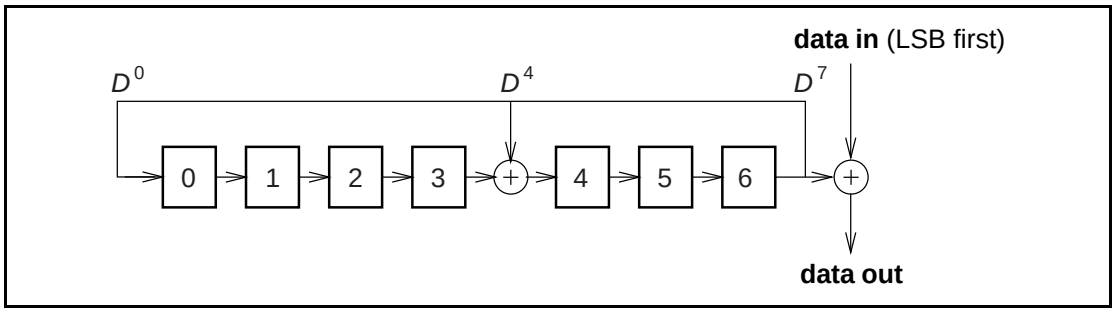


Figure 7.9: Data whitening LFSR.

After initialization, the packet header and the payload (including the CRC) are whitened. The payload whitening shall continue from the state the whitening LFSR had at the end of HEC. There shall be no re-initialization of the shift register between packet header and payload. The first bit of the “data in” sequence shall be the LSB of the packet header.

For Enhanced Data Rate packets, whitening shall not be applied to the guard, synchronization and trailer portions of the Enhanced Data Rate packets. During the periods where whitening is not applied the LFSR shall be paused.

7.3 ERROR CORRECTION

There are three error correction schemes defined for Bluetooth:

- 1/3 rate FEC
- 2/3 rate FEC
- ARQ scheme for the data

The purpose of the FEC scheme on the data payload is to reduce the number of retransmissions. However, in a reasonably error-free environment, FEC gives unnecessary overhead that reduces the throughput. Therefore, the packet definitions given in Section 6 have been kept flexible to use FEC in the payload or not, resulting in the **DM** and **DH** packets for the ACL logical transport, **HV** packets for the SCO logical transport, and **EV** packets for the eSCO logical transport. The packet header is always protected by a 1/3 rate FEC since it contains valuable link information and is designed to withstand more bit errors.

Correction measures to mask errors in the voice decoder are not included in this section. This matter is discussed in Section 9.3 on page 195.

7.4 FEC CODE: RATE 1/3

A simple 3-times repetition FEC code is used for the header. The repetition code is implemented by repeating each bit three times, see the illustration in Figure 7.10 on page 139. The 3-times repetition code is used for the entire header, as well as for the synchronous data field in the **HV1** packet.

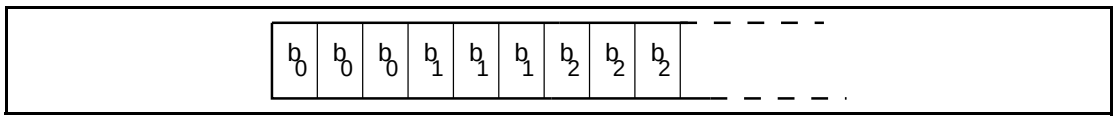


Figure 7.10: Bit-repetition encoding scheme.

7.5 FEC CODE: RATE 2/3

The other FEC scheme is a (15,10) shortened Hamming code. The generator polynomial is $g(D) = (D + 1)(D^4 + D + 1)$. This corresponds to 65 in octal notation. The LFSR generating this code is depicted in Figure 7.11 on page 140. Initially all register elements are set to zero. The 10 information bits are sequentially fed into the LFSR with the switches S1 and S2 set in position 1. Then, after the final input bit, the switches S1 and S2 are set in position 2, and the five parity bits are shifted out. The parity bits are appended to the information bits. Subsequently, each block of 10 information bits is encoded into a 15 bit codeword. This code can correct all single errors and detect all double errors in each codeword. This 2/3 rate FEC is used in the **DM** packets, in the data field of the **DV** packet, in the **FHS** packet, in the **HV2** packet, and in the **EV4** packet. Since the encoder operates with information segments of length 10, tail bits with value zero shall be appended after the CRC bits to bring the total number of bits equal to a multiple of 10. The number of tail bits to append shall be the least possible that achieves this (i.e., in the interval 0...9). These tail bits are not included in the payload length indicator for ACL packets or in the payload length field of the eSCO setup LMP command.

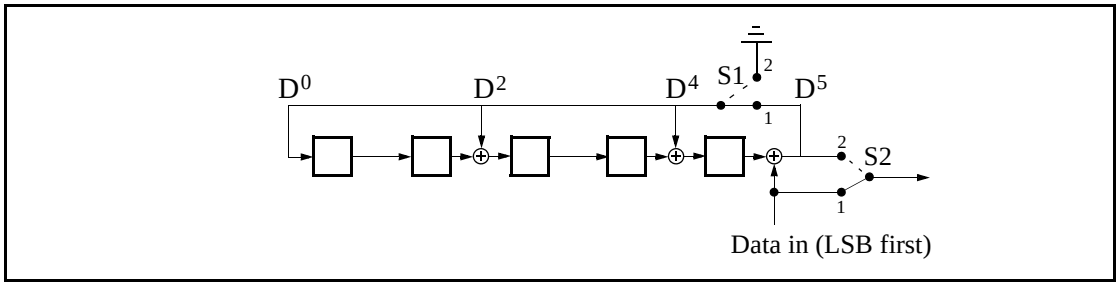


Figure 7.11: LFSR generating the (15,10) shortened Hamming code.

7.6 ARQ SCHEME

With an automatic repeat request scheme, **DM, DH** the data field of **DV** packets, and **EV** packets shall be transmitted until acknowledgement of a successful reception is returned by the destination (or timeout is exceeded). The acknowledgement information shall be included in the header of the return packet. The ARQ scheme is only used on the payload in the packet and only on packets that have a CRC. The packet header and the synchronous data payload of HV and DV packets are not protected by the ARQ scheme.

7.6.1 Unnumbered ARQ

Bluetooth uses a fast, unnumbered acknowledgment scheme. An ACK (ARQN=1) or a NAK (ARQN=0) is returned in response to the receipt of previously received packet. The slave shall respond in the slave-to-master slot directly following the master-to-slave slot unless the slave has scatternet commitments in that timeslot; the master shall respond at the next event addressing the same slave (the master may have addressed other slaves between the last received packet from the considered slave and the master response to this packet). For a packet reception to be successful, at least the HEC must pass. In addition, the CRC must pass if present.

In the first POLL packet at the start of a new connection (as a result of a page, page scan, role switch, or unpark) the master shall initialize the ARQN bit to NAK. The response packet sent by the slave shall also have the ARQN bit set to NAK. The subsequent packets shall use the following rules. The initial value of the master's eSCO ARQN at link set-up shall be NAK.

The ARQ bit shall only be affected by data packets containing CRC and empty slots. As shown in Figure 7.12 on page 142, upon successful reception of a CRC packet, the ARQN bit shall be set to ACK. If, in any receive slot in the slave, or, in a receive slot in the master following transmission of a packet, one of these events applies:

1. no access code is detected,
2. the HEC fails,
3. the CRC fails,

then the ARQN bit shall be set to NAK. In eSCO the ARQN bit may be set to ACK even when the CRC on an EV packet has failed thus enabling delivery of erroneous packets.

Packets that have correct HEC but that are addressed to other slaves, or packets other than DH, DM, DV or EV packets, shall not affect the ARQN bit, except as noted in Section 7.6.2.2 on page 145. In these cases the ARQN bit shall be left as it was prior to reception of the packet. For ACL packets, if a CRC packet with a correct header has the same SEQN as the previously received CRC packet, the ARQN bit shall be set to ACK and the payload shall be ignored without checking the CRC. For eSCO packets, the SEQN shall not be used

when determining the ARQN. If an eSCO packet has been received successfully within the eSCO window subsequent receptions within the eSCO window shall be ignored. At the end of the eSCO window, the master's ARQN shall be retained for the first master-to-slave transmission in the next eSCO window.

The ARQ bit in the FHS packet is not meaningful. Contents of the ARQN bit in the FHS packet shall not be checked.

The ARQ bit in the extended inquiry response packet is not used. The bit shall be set to zero and ignored upon receipt.

Broadcast packets shall be checked on errors using the CRC, but no ARQ scheme shall be applied. Broadcast packets shall never be acknowledged.

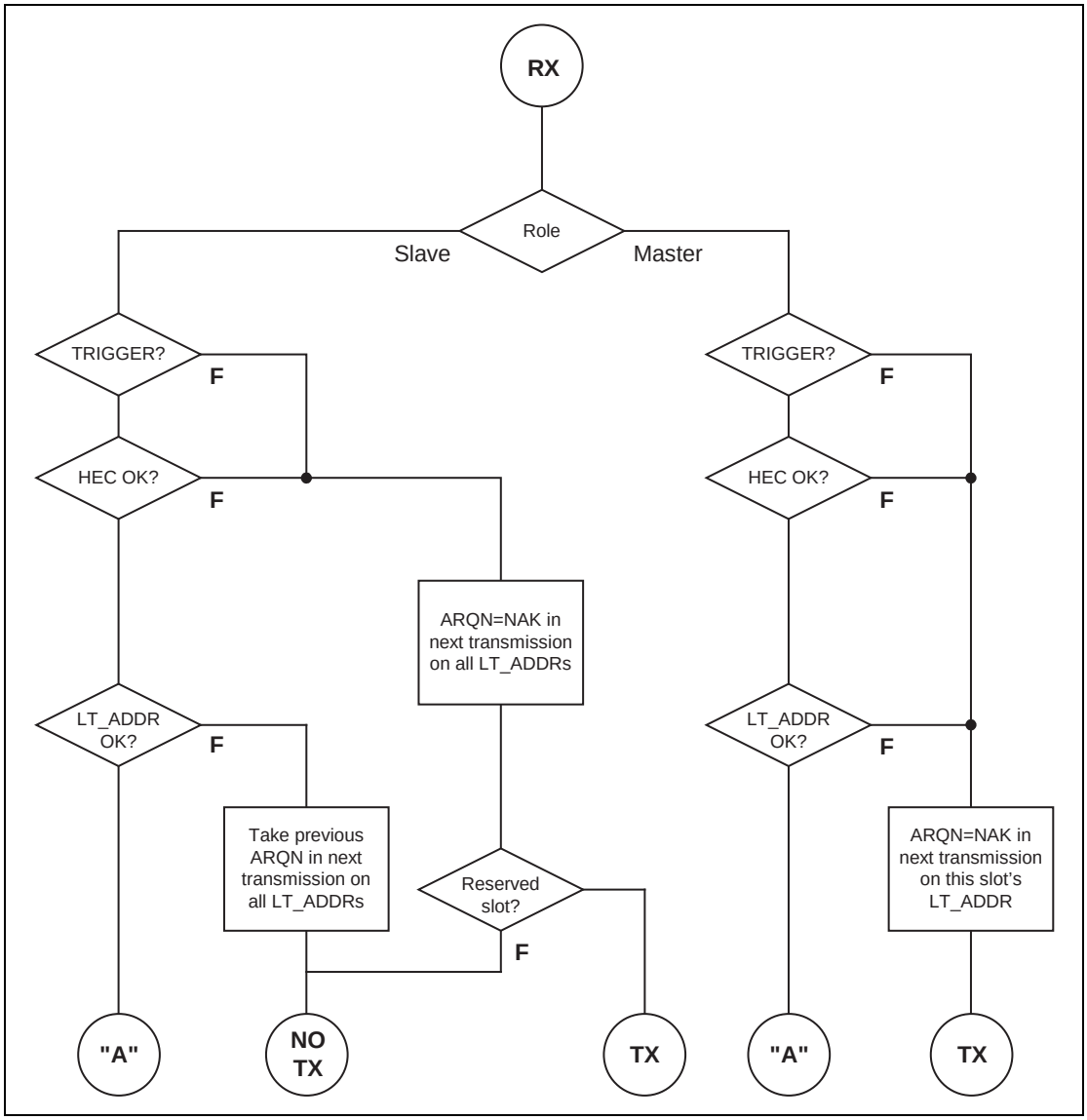


Figure 7.12: Stage 1 of the receive protocol for determining the ARQN bit.

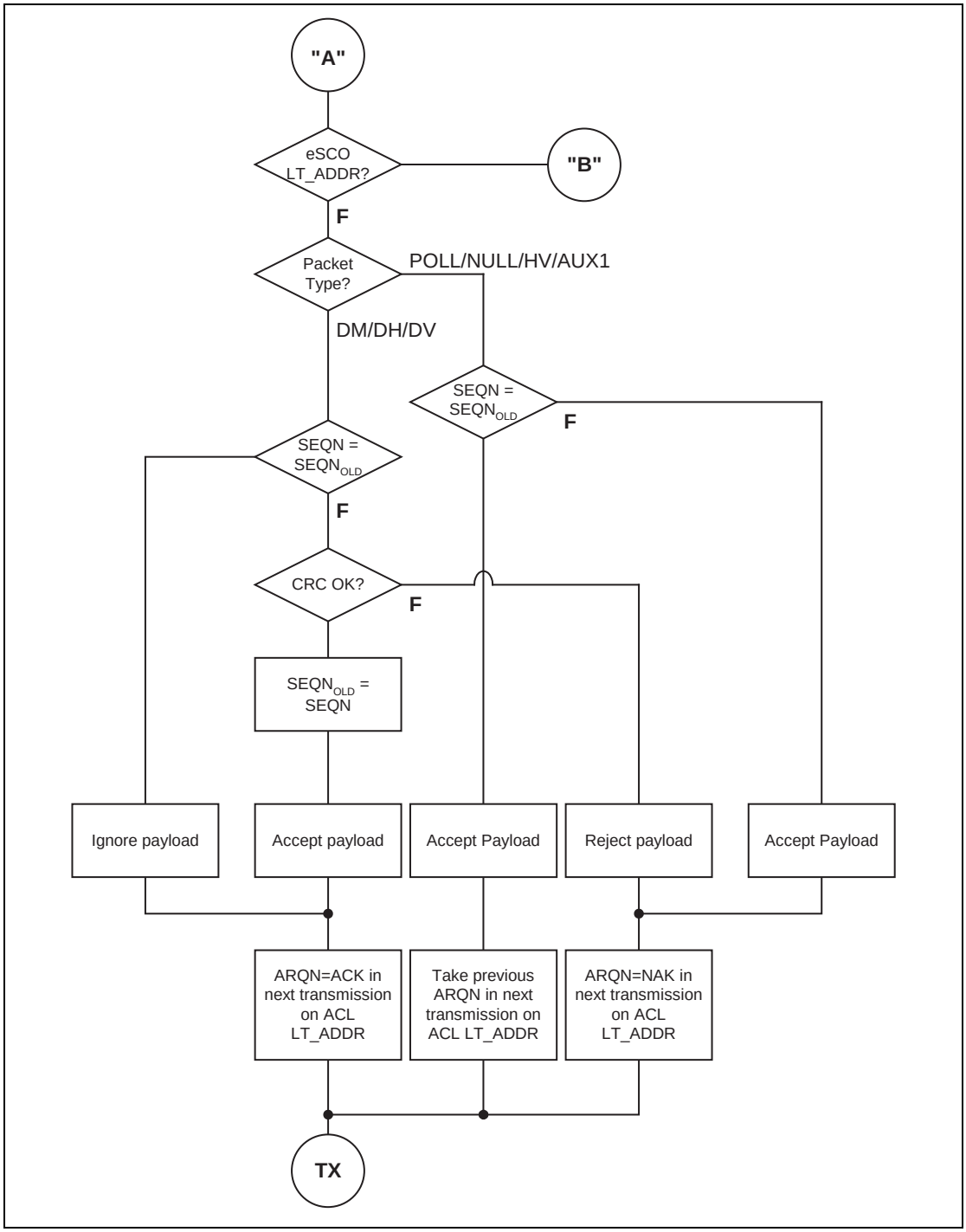


Figure 7.13: Stage 2 (ACL) of the receive protocol for determining the ARQN bit.

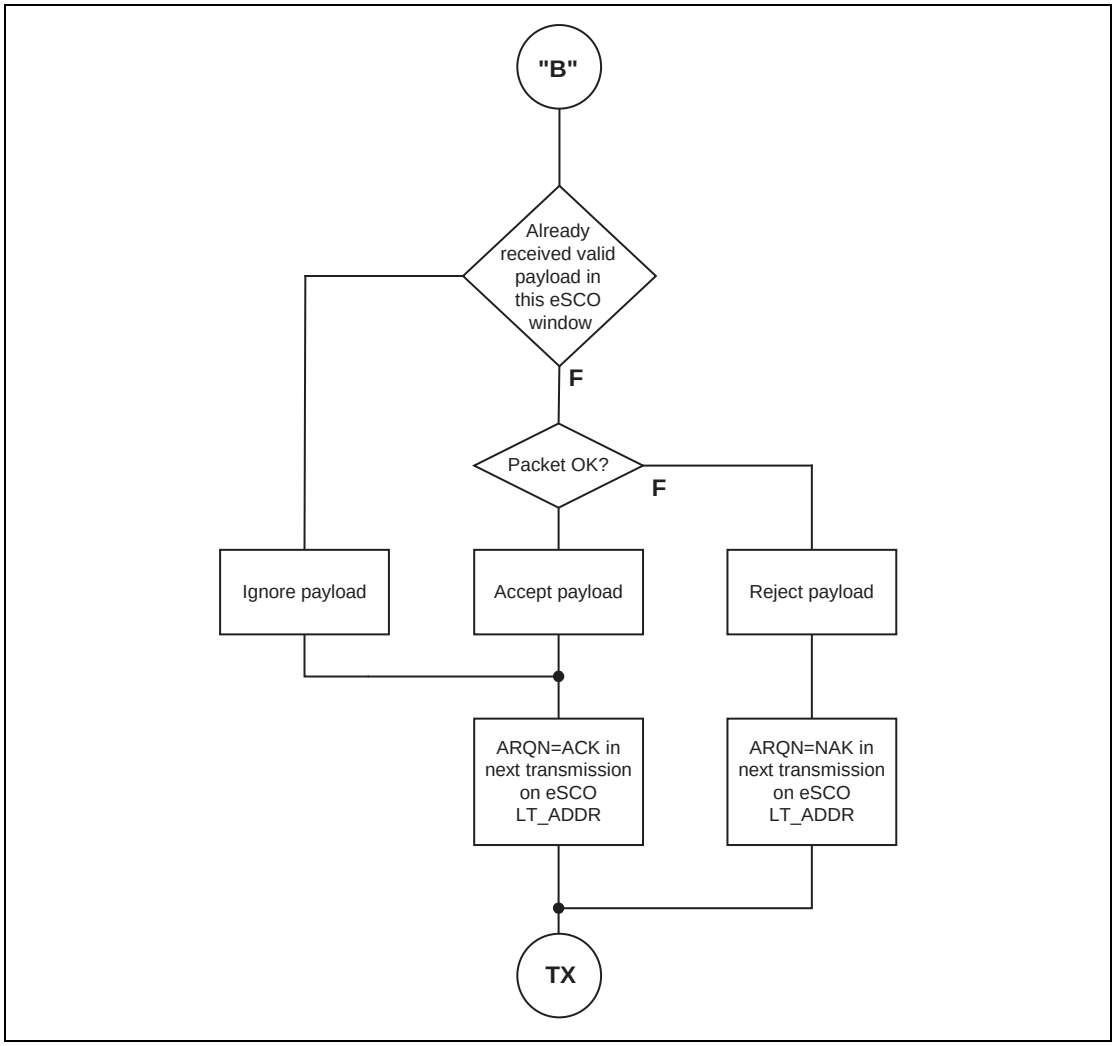


Figure 7.14: Stage 2 (eSCO) of the receive protocol for determining the ARQN bit.

7.6.2 Retransmit Filtering

The data payload shall be transmitted until a positive acknowledgment is received or a timeout is exceeded. A retransmission shall be carried out either because the packet transmission itself failed, or because the acknowledgment transmitted in the return packet failed (note that the latter has a lower failure probability since the header is more heavily coded). In the latter case, the destination keeps receiving the same payload over and over again. In order to filter out the retransmissions in the destination, the SEQN bit is present in the header. Normally, this bit is alternated for every new CRC data payload transmission. In case of a retransmission, this bit shall not be changed so the destination can compare the SEQN bit with the previous SEQN value. If different, a new data payload has arrived; otherwise it is the same data payload and may be ignored. Only new data payloads shall be transferred to the Baseband Resource Manager. Note that CRC data payloads can be carried only by **DM**, **DH**, **DV** or **EV** packets.

7.6.2.1 Initialization of SEQN at start of new connection

The SEQN bit of the first CRC data packet at the start of a connection (as a result of page, page scan, or role switch or unpark) on both the master and the slave sides shall be set to 1. The subsequent packets shall use the rules in the following sections.

7.6.2.2 ACL and SCO retransmit filtering

The SEQN bit shall only be affected by the CRC data packets as shown in Figure 7.15. It shall be inverted every time a new CRC data packet is sent. The CRC data packet shall be retransmitted with the same SEQN number until an ACK is received or the packet is flushed. When an ACK is received, a new payload may be sent and on that transmission the SEQN bit shall be inverted. If a device decides to flush (see Section 7.6.3 on page 147), and it has not received an acknowledgement for the current packet, it shall replace the current packet with an ACL-U continuation packet with the same sequence number as the current packet and length zero. If it replaces the current packet in this way it shall not move on to transmit the next packet until it has received an ACK.

If the slave receives a packet other than DH, DM, DV or EV with the SEQN bit inverted from that in the last header successfully received on the same LT_ADDR, it shall set the ARQN bit to NAK until a DH, DM, DV or EV packet is successfully received.

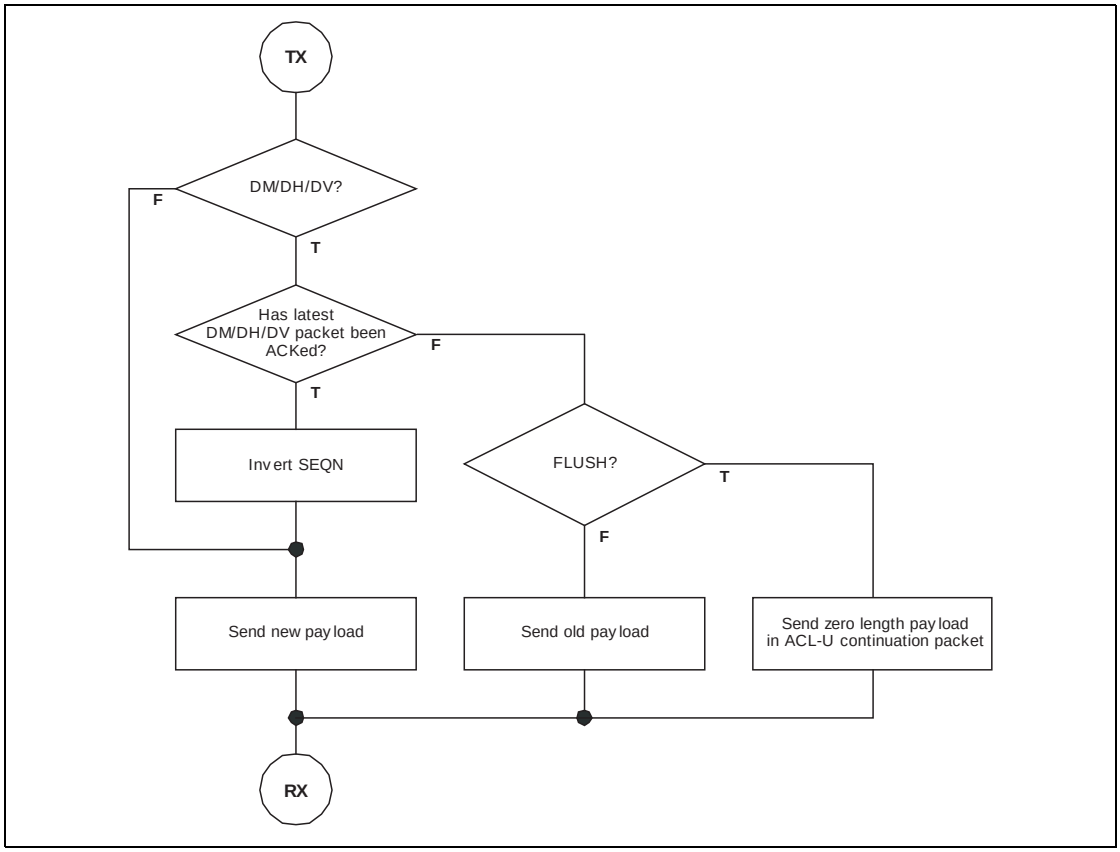


Figure 7.15: Transmit filtering for packets with CRC.

7.6.2.3 eSCO retransmit filtering

In eSCO, the SEQN bit shall be toggled every eSCO window. The value shall be constant for the duration of the eSCO window. The initial value of SEQN shall be zero.

For a given eSCO window the SEQN value shall be constant.

7.6.2.4 FHS retransmit filtering

The SEQN bit in the FHS packet is not meaningful. This bit may be set to any value. Contents of the SEQN bit in the FHS packet shall not be checked.

7.6.2.5 Extended Inquiry Response retransmit filtering

The SEQN bit in the extended inquiry response packet is not used. The bit shall be set to zero and ignored upon receipt.

7.6.2.6 Packets without CRC retransmit filtering

During transmission of packets without a CRC the SEQN bit shall remain the same as it was in the previous packet.

7.6.3 Flushing Payloads

In ACL, the ARQ scheme can cause variable delay in the traffic flow since retransmissions are inserted to assure error-free data transfer. For certain communication links, only a limited amount of delay is allowed: retransmissions are allowed up to a certain limit at which the current payload shall be ignored. This data transfer is indicated as **isochronous traffic**. This means that the retransmit process must be overruled in order to continue with the next data payload. Aborting the retransmit scheme is accomplished by *flushing* the old data and forcing the link controller to take the next data instead.

Flushing results in loss of remaining portions of an L2CAP message. Therefore, the packet following the flush shall have a start packet indication of LLID = 10 in the payload header for the next L2CAP message. This informs the destination of the flush (see Section 6.6). Flushing will not necessarily result in a change in the SEQN bit value, see the previous section.

The Flush Timeout defines a maximum period after which all segments of the ACL-U packet with a Packet_Boundary_Flag value of 10 are flushed from the Controller buffer. The Flush Timeout shall start when the First segment of the ACL-U packet is stored in the Controller buffer. If the First segment of an ACL-U packet has a Packet_Boundary_Flag value of 00, it is non-automatically-flushable and shall not cause the Flush Timeout to start. After the Flush timeout has expired the Link Controller may continue transmissions according to the procedure described in Section 7.6.2.2 on page 145, however the Baseband Resource Manager shall not continue the transmission of the ACL-U packet to the Link Controller. If the Baseband Resource Manager has further segments of the packet queued for transmission to the Link Controller it shall delete the remaining segments of the ACL-U packet from the queue. In case the complete ACL-U packet was not stored in the Controller buffer yet, any Continuation segments, received for the ACL logical transport, shall be flushed, until a First segment is received. When the complete ACL-U packet has been flushed, the Link Manager shall continue transmission of the next ACL-U packet for the ACL logical transport. The default Flush Timeout shall be infinite, i.e. re-transmissions are carried out until physical link loss occurs. This is also referred to as a 'reliable channel.' All devices shall support the default Flush Timeout. Reliable data shall be sent over a channel with a finite flush timeout by marking reliable packets as non-automatically-flushable.

In eSCO, packets shall be automatically flushed at the end of the eSCO window.

7.6.4 Multi-slave Considerations

In a piconet with multiple logical transports, the master shall carry out the ARQ protocol independently on each logical transport.

7.6.5 Broadcast Packets

Broadcast packets are packets transmitted by the master to all the slaves simultaneously (see Section 8.6.4) If multiple hop sequences are being used each transmission may only be received by some of the slaves. In this case the master shall repeat the transmission on each hop sequence. A broadcast packet shall be indicated by the all-zero LT_ADDR (note; the FHS packet and the extended inquiry response packet are the only packets which may have an all-zero LT_ADDR but are not broadcast packets). Broadcast packets shall not be acknowledged (at least not at the LC level).

Since broadcast messages are not acknowledged, each broadcast packet is transmitted at least a fixed number of times. A broadcast packet should be transmitted N_{BC} times before the next broadcast packet of the same broadcast message is transmitted, see Figure 7.16 on page 149. Optionally, a broadcast packet may be transmitted N_{BC} + 1 times. Note: N_{BC}=1 means that each broadcast packet should be sent only once, but optionally may be sent twice. However, time-critical broadcast information may abort the ongoing broadcast train. For instance, unpark messages sent at beacon instances may do this, see Section 8.9.5.

If multiple hop sequences are being used then the master may transmit on the different hop sequences in any order, providing that transmission of a new broadcast packet shall not be started until all transmissions of any previous broadcast packet have completed on all hop sequences. The transmission of a single broadcast packet may be interleaved among the hop sequences to minimize the total time to broadcast a packet. The master has the option of transmitting only N_{BC} times on channels common to all hop sequences.

Broadcast packets with a CRC shall have their own sequence number. The SEQN of the first broadcast packet with a CRC shall be set to SEQN = 1 by the master and shall be inverted for each new broadcast packet with CRC thereafter. Broadcast packets without a CRC have no influence on the sequence number. The slave shall accept the SEQN of the first broadcast packet it receives in a connection and shall check for change in SEQN for subsequent broadcast packets. Since there is no acknowledgement of broadcast messages and there is no end packet indication, it is important to receive the start packets correctly. To ensure this, repetitions of the broadcast packets that are L2CAP start packets and LMP packets shall not be filtered out. These packets shall be indicated by LLID=1X in the payload header as explained in Table 6.6 on page 130. Only repetitions of the L2CAP continuation packets shall be filtered out.

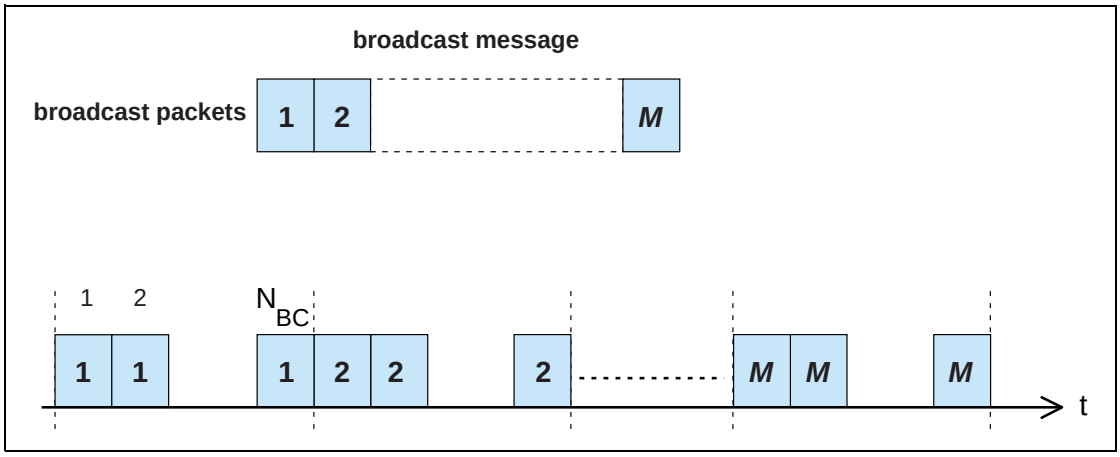


Figure 7.16: Broadcast repetition scheme

7.7 ERRONEOUS SYNCHRONOUS DATA REPORTING

Erroneous data reporting may be enabled for synchronous links. When enabled, synchronous data shall be processed as follows:

- If, during an (e)SCO interval, an eSCO packet was received with valid HEC and valid CRC or a SCO packet was received with valid HEC, the received payload data shall be sent to the upper-edge of the Controller with a "good data" indication.
- If, during an eSCO interval, eSCO packets with valid HEC were received, but none of them had a valid CRC, the best known data available (e.g. data derived from the received data or the actual payload data that was received with a CRC error) shall be sent to the upper-edge of the Controller with a "data with possible errors" indication.
- If, during an (e)SCO interval, no SCO or eSCO packet with valid HEC has been received, a "lost data" indication shall be sent to the upper-edge of the Controller.

8 LINK CONTROLLER OPERATION

This section describes how a piconet is established and how devices can be added to and released from the piconet. Several states of operation of the devices are defined to support these functions. In addition, the operation of several piconets with one or more common members, the scatternet, is discussed.

8.1 OVERVIEW OF STATES

Figure 8.1 on page 150 shows a state diagram illustrating the different states used in the link controller. There are three major states: **STANDBY**, **CONNECTION**, and **PARK**; in addition, there are seven substates, **page**, **page scan**, **inquiry**, **inquiry scan**, **master response**, **slave response**, and **inquiry response**. The substates are interim states that are used to establish connections and enable device discovery. To move from one state or substate to another, either commands from the link manager are used, or internal signals in the link controller are used (such as the trigger signal from the correlator and the timeout signals).

.

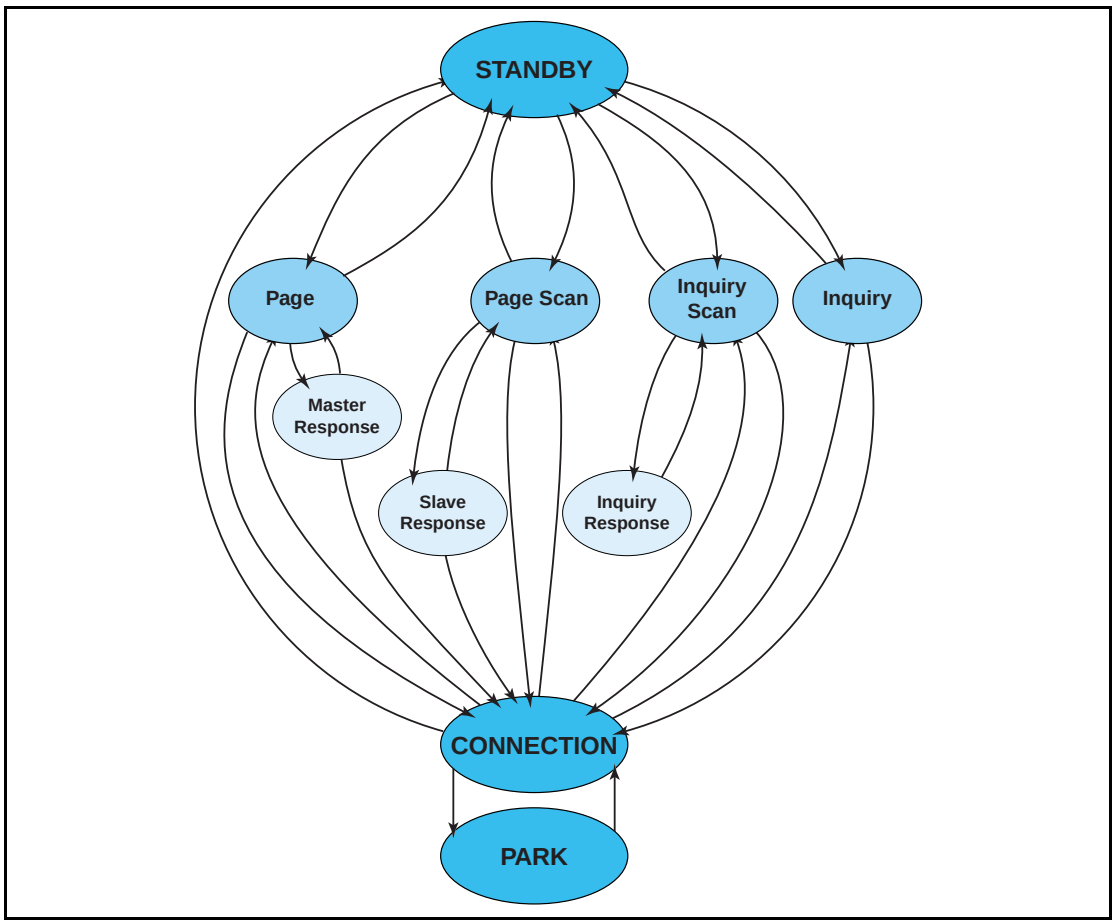


Figure 8.1: State diagram of link controller.

8.2 STANDBY STATE

The **STANDBY** state is the default state in the device. In this state, the device may be in a low-power mode. Only the native clock is running at the accuracy of the LPO (or better).

The controller may leave the **STANDBY** state to scan for page or inquiry messages, or to page or inquiry itself.

8.3 CONNECTION ESTABLISHMENT SUBSTATES

In order to establish new connections the paging procedure is used. Only the Bluetooth device address is required to set up a connection. Knowledge about the clock, obtained from the inquiry procedure (see Section 8.4 on page 160) or from a previous connection with this device, and the page scanning mode of the other device will accelerate the setup procedure. A device that establishes a connection carries out a page procedure and will automatically become the master of the connection.

8.3.1 Page Scan Substate

In the **page scan** substate, a device may be configured to use either the standard or interlaced scanning procedure. During a standard scan, a device listens for the duration of the scan window $T_{w\_page\_scan}$ (11.25ms default, see HCI [Part E] Section 7.3.20 on page 592), while the interlaced scan is performed as two back to back scans of $T_{w\_page\_scan}$. If the scan interval is not at least twice the scan window, then interlaced scan shall not be used. During each scan window, the device shall listen at a single hop frequency, its correlator matched to its device access code (DAC). The scan window shall be long enough to completely scan 16 page frequencies.

When a device enters the **page scan** substate, it shall select the scan frequency according to the page hopping sequence determined by the device's Bluetooth device address, see Section 2.6.4.1 on page 92. The phase in the sequence shall be determined by $CLKN_{16\text{-}12}$ of the device's native clock; that is, every 1.28s a different frequency is selected.

In the case of a standard scan, if the correlator exceeds the trigger threshold during the **page scan**, the device shall enter the **slave response** substate described in Section 8.3.3.1 on page 157. The scanning device may also use interlaced scan. In this case, if the correlator does not exceed the trigger threshold during the first scan it shall scan a second time using the phase in the sequence determined by $[CLKN_{16\text{-}12} + 16] \bmod 32$. If on this second scan the correlator exceeds the trigger threshold the device shall enter the **slave response** substate using $[CLKN_{16\text{-}12} + 16] \bmod 32$ as the frozen CLKN* in the calculation for Spars(79), see Section 2.6.4.3 on page 93 for details. If the correlator does not exceed the trigger threshold during a scan in normal mode or

during the second scan in interlaced scan mode it shall return to either the **STANDBY** or **CONNECTION** state.

The **page scan** substate can be entered from the **STANDBY** state or the **CONNECTION** state. In the **STANDBY** state, no connection has been established and the device can use all the capacity to carry out the **page scan**. Before entering the **page scan** substate from the **CONNECTION** state, the device should reserve as much capacity as possible for scanning. If desired, the device may place ACL connections in Hold, Park, or Sniff (see Section 8.8 on page 183 and Section 8.9 on page 184). Synchronous connections should not be interrupted by the page scan, although eSCO retransmissions should be paused during the scan. The page scan may be interrupted by the reserved synchronous slots which should have higher priority than the **page scan**. SCO packets should be used requiring the least amount of capacity (**HV3** packets). The scan window shall be increased to minimize the setup delay. If one SCO logical transport is present using **HV3** packets and T_{SCO}=6 slots or one eSCO logical transport is present using **EV3** packets and T_{ESCO}=6 slots, a total scan window $T_{w\ page\ scan}$ of at least 36 slots (22.5ms) is recommended; if two SCO links are present using **HV3** packets and T_{SCO}=6 slots or two eSCO links are present using **EV3** packets and T_{ESCO}=6 slots, a total scan window of at least 54 slots (33.75ms) is recommended.

The scan interval $T_{page\ scan}$ is defined as the interval between the beginnings of two consecutive page scans. A distinction is made between the case where the scan interval is equal to the scan window $T_{w\ page\ scan}$ (continuous scan), the scan interval is maximal 1.28s, or the scan interval is maximal 2.56s. These three cases shall determine the behavior of the paging device; that is, whether the paging device shall use R0, R1 or R2, see also Section 8.3.2 on page 153. Table 8.1 illustrates the relationship between $T_{page\ scan}$ and modes R0, R1 and R2. Although scanning in the R0 mode is continuous, the scanning may be interrupted for example by reserved synchronous slots. The scan interval information is included in the SR field in the FHS packet.

SR mode	$T_{page\ scan}$
R0	≤ 1.28s and = $T_{w\ page\ scan}$
R1	≤ 1.28s
R2	≤ 2.56s
Reserved	-

Table 8.1: Relationship between scan interval, and paging modes R0, R1, and R2.

8.3.2 Page substate

The **page** substate is used by the master (source) to activate and connect to a slave (destination) in the **page scan** substate. The master tries to coincide with the slave's scan activity by repeatedly transmitting the paging message consisting of the slave's device access code (DAC) in different hop channels. Since the Bluetooth clocks of the master and the slave are not synchronized, the master does not know exactly when the slave wakes up and on which hop frequency. Therefore, it transmits a train of identical page scan messages at different hop frequencies and listens in between the transmit intervals until it receives a response from the slave.

The page procedure in the master consists of a number of steps. First, the Host communicates the BD_ADDR of the slave to the Controller. This BD_ADDR shall be used by the master to determine the page hopping sequence; see Section 2.6.4.2 on page 93. For the phase in the sequence, the master shall use an estimate of the slave's clock. For example, this estimate can be derived from timing information that was exchanged during the last encounter with this particular device (which could have acted as a master at that time), or from an inquiry procedure. With this estimate CLKE of the slave's Bluetooth clock, the master can predict on which hop channel the slave starts page scanning.

The estimate of the Bluetooth clock in the slave can be completely wrong. Although the master and the slave use the same hopping sequence, they use different phases in the sequence and might never select the same frequency. To compensate for the clock drifts, the master shall send its page message during a short time interval on a number of wake-up frequencies. It shall transmit also on hop frequencies just before and after the current, predicted hop frequency. During each TX slot, the master shall sequentially transmit on two different hop frequencies. In the following RX slot, the receiver shall listen sequentially to two corresponding RX hops for an ID packet. The RX hops shall be selected according to the page response hopping sequence. The page response hopping sequence is strictly related to the page hopping sequence: for each page hop there is a corresponding page response hop. The RX/TX timing in the **page** substate is described in Section 2.2.5, see also Figure 2.7 on page 78. In the next TX slot, the master shall transmit on two hop frequencies different from the former ones. Note: The hop rate is increased to 3200 hops/s.

With the increased hopping rate as described above, the transmitter can cover 16 different hop frequencies in 16 slots or 10 ms. The page hopping sequence is divided over two paging trains **A** and **B** of 16 frequencies. Train **A** includes the 16 hop frequencies surrounding the current, predicted hop frequency *f(k)*, where k is determined by the clock estimate $CLKE_{16\text{-}12}$. The first train consists of hops

f(k-8), f(k-7),...,f(k),...,f(k+7)

When the difference between the Bluetooth clocks of the master and the slave is between -8x1.28 s and +7x1.28 s, one of the frequencies used by the master will be the hop frequency the slave will listen to. Since the master does not know when the slave will enter the **page scan** substate, the master has to repeat this train **A** N_{page} times or until a response is obtained, whichever is shorter. If the slave scan interval corresponds to R1, the repetition number is at least 128; if the slave scan interval corresponds to R2 or if the master has not previously read the slave's SR mode, the repetition number is at least 256. If the master has not previously read the slave's SR mode it shall use Npage >= 256. Note that $CLKE_{16-12}$ changes every 1.28 s; therefore, every 1.28 s, the trains will include different frequencies of the page hopping set.

When the difference between the Bluetooth clocks of the master and the slave is less than -8x1.28 s or larger than +7x1.28 s, the remaining 16 hops are used to form the new 10 ms train **B.** The second train consists of hops

f(k-16), f(k-15),...,f(k-9),f(k+8)...,f(k+15)

Train **B** shall be repeated for N_{page} times. If no response is obtained, train **A** shall be tried again N_{page} times. Alternate use of train A and train B shall be continued until a response is received or the timeout *pageTO* is exceeded. If a response is returned by the slave, the master device enters the **master response** substate.

The **page** substate may be entered from the **STANDBY** state or the **CONNECTION** state. In the **STANDBY** state, no connection has been established and the device can use all the capacity to carry out the page. Before entering the **page** substate from the **CONNECTION** state, the device should free as much capacity as possible for paging. To ensure this, it is recommended that the ACL connections are put on hold or park. However, the synchronous connections shall not be disturbed by the page. This means that the page will be interrupted by the reserved SCO and eSCO slots which have higher priority than the page. In order to obtain as much capacity for paging, it is recommended to use the SCO packets which use the least amount of capacity (**HV3** packets). If SCO or eSCO links are present, the repetition number N_{page} of a single train shall be increased, see Table 8.2. Here it has been assumed that the **HV3** packet are used with an interval T_{SCO}=6 slots or **EV3** packets are used with an interval of T_{ESCO}=6 slots, which would correspond to a 64 kb/s synchronous link.

.

SR mode	no synchronous link	one synchronous link (HV3)	two synchronous links (HV3)
R0	$N_{page} \geq 1$	$N_{page} \geq 2$	$N_{page} \geq 3$
R1	$N_{page} \geq 128$	$N_{page} \geq 256$	$N_{page} \geq 384$
R2	$N_{page} \geq 256$	$N_{page} \geq 512$	$N_{page} \geq 768$

Table 8.2: Relationship between train repetition, and paging modes R0, R1 and R2 when synchronous links are present.

The construction of the page train shall be independent of the presence of synchronous links; that is, synchronous packets are sent on the reserved slots but shall not affect the hop frequencies used in the unreserved slots, see Figure 8.2 on page 155.

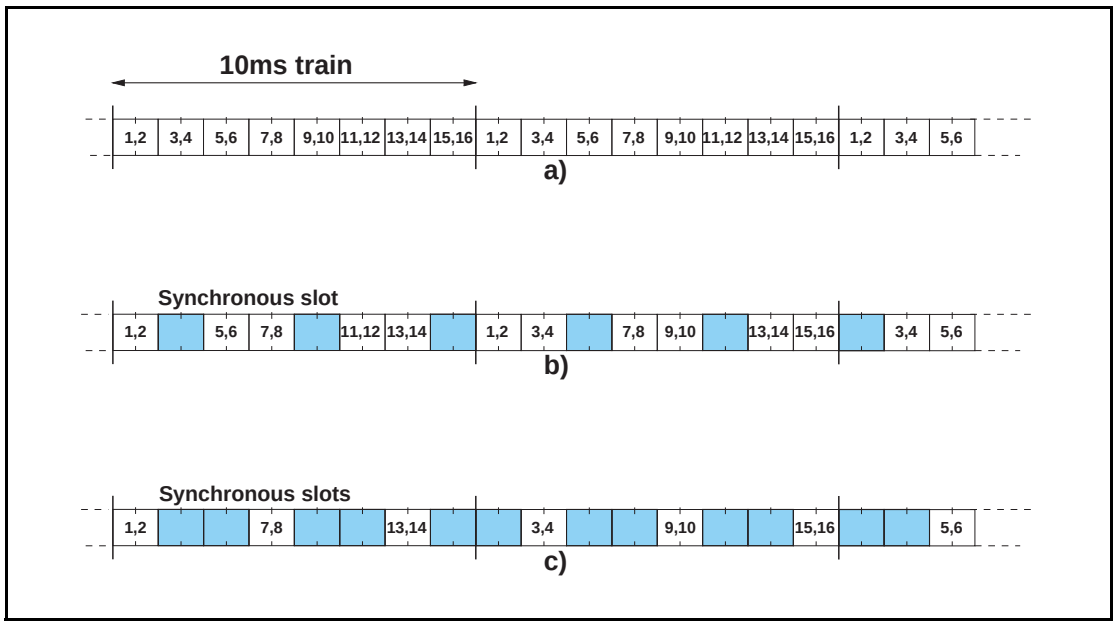


Figure 8.2: Conventional page (a), page while one synchronous link present (b), page while two synchronous links present (c).

8.3.3 Page Response Substates

When a page message is successfully received by the slave, there is a coarse FH synchronization between the master and the slave. Both the master and the slave enter a response substate to exchange vital information to continue the connection setup. It is important for the piconet connection that both devices shall use the same channel access code, use the same channel hopping sequence, and their clocks are synchronized. These parameters shall be derived from the master device. The device that initializes the connection (starts paging) is defined as the master device (which is thus only valid during the time the piconet exists). The channel access code and channel hopping sequence shall be derived from the Bluetooth device address (BD_ADDR) of the master. The timing shall be determined by the master clock. An offset shall be added to the slave's native clock to temporarily synchronize the slave clock to the master clock. At start-up, the master parameters are transmitted from the master to the slave. The messaging between the master and the slave at start-up is specified in this section.

The initial messaging between master and slave is shown in Table 8.3 on page 156 and in Figure 8.3 on page 157 and Figure 8.4 on page 157. In those two figures frequencies f (k), f(k+1), etc. are the frequencies of the page hopping sequence determined by the slave's BD_ADDR. The frequencies f'(k), f'(k+1), etc. are the corresponding page_response frequencies (slave-to-master). The frequencies g(m) belong to the basic channel hopping sequence.

Step	Message	Packet Type	Direction	Hopping Sequence	Access Code and Clock
1	Page	ID	Master to slave	Page	Slave
2	First slave page response	ID	Slave to master	Page response	Slave
3	Master page response	FHS	Master to slave	Page	Slave
4	Second slave page response	ID	Slave to master	Page response	Slave
5	1st packet master	POLL	Master to slave	Channel	Master
6	1st packet slave	Any type	Slave to master	Channel	Master

Table 8.3: Initial messaging during start-up.

In step 1 (see Table 8.3 on page 156), the master device is in **page** substate and the slave device in the **page scan** substate. Assume in this step that the page message sent by the master reaches the slave. On receiving the page message, the slave enters the **slave response** in step 2. The master waits for a reply from the slave and when this arrives in step 2, it shall enter the **master response** in step 3. Note that during the initial message exchange, all parame-

ters are derived from the slave's device address, and that only the page hopping and page response hopping sequences are used (are also derived from the slave's device address). Note that when the master and slave enter the response states, their clock input to the page and page response hop selection is frozen as is described in Section 2.6.4.3 on page 93.

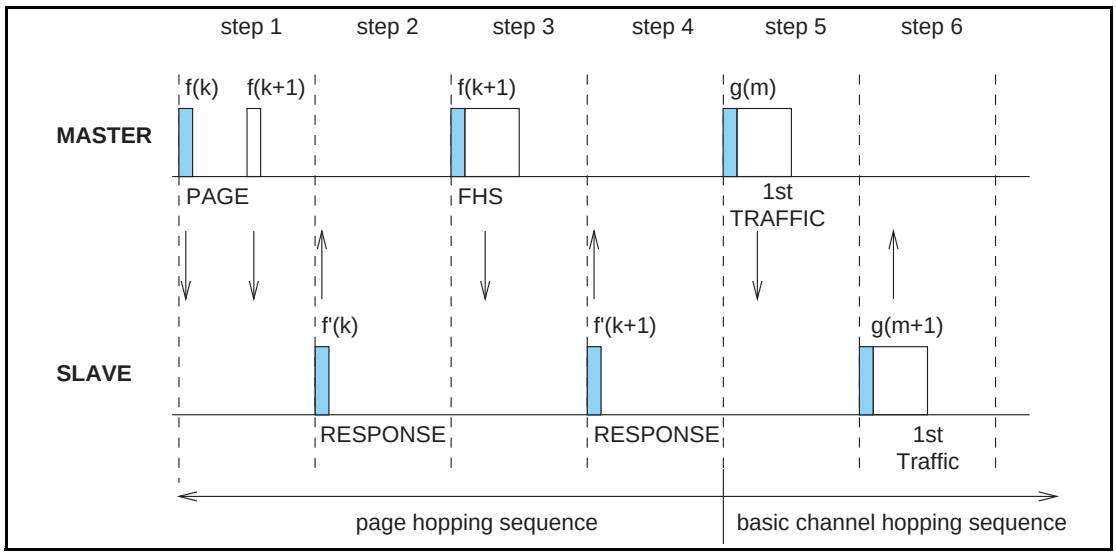


Figure 8.3: Messaging at initial connection when slave responds to first page message.

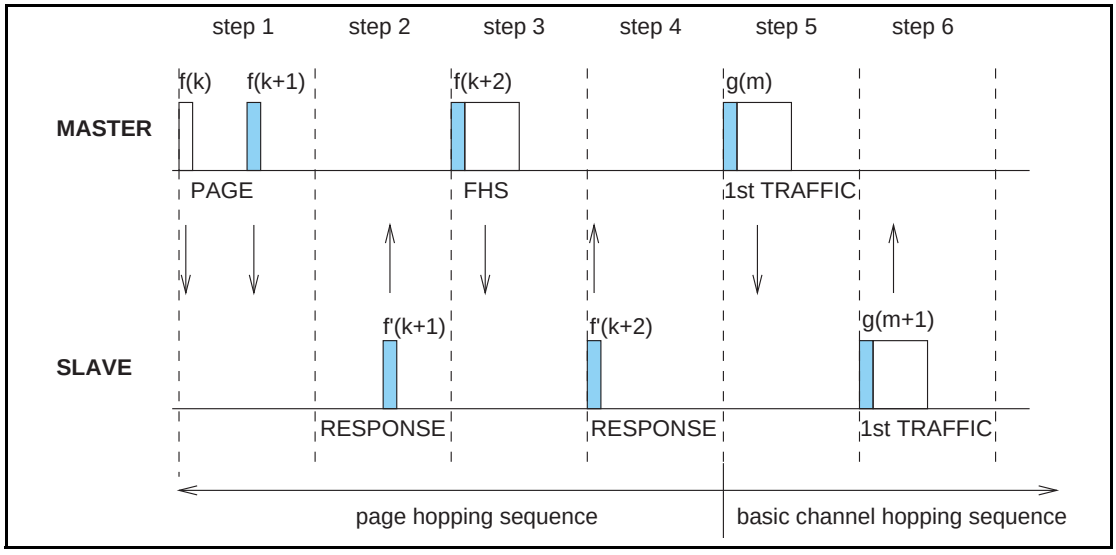


Figure 8.4: Messaging at initial connection when slave responds to second page message.

8.3.3.1 Slave response substate

After having received the page message in step 1, the slave device shall transmit a slave page response message (the slave's device access code) in step 2. This response message shall be the slave's device access code. The slave shall transmit this response 625 μs after the beginning of the received page message and at the response hop frequency that corresponds to the hop fre-

quency in which the page message was received. The slave transmission is therefore time aligned to the master transmission. During initial messaging, the slave shall still use the page response hopping sequence to return information to the master. The clock input $CLKN_{16-12}$ shall be frozen at the value it had at the time the page message was received.

After having sent the response message, the slave's receiver shall be activated 312.5 μs after the start of the response message and shall await the arrival of an **FHS** packet. Note that an **FHS** packet can arrive 312.5 μs after the arrival of the page message as shown in Figure 8.4 on page 157, and not after 625 μs as is usually the case in the piconet physical channel RX/TX timing. More details about the timing can be found in Section 2.4.4 on page 79.

If the setup fails before the **CONNECTION** state has been reached, the following procedure shall be carried out. The slave shall listen as long as no **FHS** packet is received until *pagerespTO* is exceeded. Every 1.25 ms, however, it shall select the next master-to-slave hop frequency according to the page hop sequence. If nothing is received after *pagerespTO*, the slave shall return back to the **page scan** substate for one scan period. Length of the scan period depends on the synchronous reserved slots present. If no page message is received during this additional scan period, the slave shall resume scanning at its regular scan interval and return to the state it was in prior to the first page scan state.

If an **FHS** packet is received by the slave in the **slave response** substate, the slave shall return a slave page response message in step 4 to acknowledge reception of the **FHS** packet. This response shall use the page response hopping sequence. The transmission of the slave page response packet is based on the reception of the **FHS** packet. Then the slave shall change to the master's channel access code and clock as received from the **FHS** packet. Only the 26 MSBs of the master clock are transferred: the timing shall be such that CLK_1 and CLK_0 are both zero at the time the **FHS** packet was received as the master transmits in even slots only. The offset between the master's clock and the slave's clock shall be determined from the master's clock in the **FHS** packet and reported to the slave's Baseband Resource Manager.

Finally, the slave enters the **CONNECTION** state in step 5. From then on, the slave shall use the master's clock and the master's BD_ADDR to determine the basic channel hopping sequence and the channel access code. The slave shall use the LT_ADDR in the **FHS** payload as the primary LT_ADDR in the **CONNECTION** state. The connection mode shall start with a POLL packet transmitted by the master. The slave may respond with any type of packet. If the POLL packet is not received by the slave, or the response packet is not received by the master, within *newconnectionTO* number of slots after FHS packet acknowledgement, the master and the slave shall return to **page** and **page scan** substates, respectively. See Section 8.5 on page 165

8.3.3.2 Master response substate

When the master has received a slave page response message in step 2, it shall enter the **master response** routine. It shall freeze the current clock input to the page hop selection scheme. The master shall then transmit an **FHS** packet in step 3 containing the master's real-time Bluetooth clock, the master's BD_ADDR, the BCH parity bits, and the class of device. The **FHS** packet contains all information to construct the channel access code without requiring a mathematical derivation from the master's Bluetooth device address. The LT_ADDR field in the packet header of FHS packets in the master response substate shall be set to all-zeros. The **FHS** packet shall be transmitted at the beginning of the master-to-slave slot following the slot in which the slave responded. The FHS packet shall carry the all-zero LT_ADDR. The TX timing of the **FHS** is not based on the reception of the response packet from the slave. The **FHS** packet may therefore be sent 312.5 μs after the reception of the response packet like shown in Figure 8.4 on page 157 and not 625 μs after the received packet as is usual in the piconet physical channel RX/TX timing, see also Section 2.4.4 on page 79.

After the master has sent its **FHS** packet, it shall wait for a second slave page response message in step 4 acknowledging the reception of the **FHS** packet. This response shall be the slave's device access code. If no response is received, the master shall retransmit the **FHS** packet with an updated clock and still using the slave's parameters. It shall retransmit the FHS packet with the clock updated each time until a second slave page response message is received, or the timeout of *pagerespTO* is exceeded. In the latter case, the master shall return to the **page** substate and send an error message to the Baseband Resource Manager. During the retransmissions of the **FHS** packet, the master shall use the page hopping sequence.

If the slave's response is received, the master shall change to using the master parameters, so it shall use the channel access code and the master clock. The lower clock bits CLK_0 and CLK_1 shall be reset to zero at the start of the **FHS** packet transmission and are not included in the **FHS** packet. Finally, the master enters the **CONNECTION** state in step 5. The master BD_ADDR shall be used to change to a new hopping sequence, the *basic channel hopping sequence*. The basic channel hopping sequence uses all 79 hop channels in a pseudo-random fashion, see also Section 2.6.4.7 on page 95. The master shall now send its first traffic packet in a hop determined with the new (master) parameters. This first packet shall be a POLL packet. See Section 8.5 on page 165. This packet shall be sent within *newconnectionTO* number of slots after reception of the FHS packet acknowledgement. The slave may respond with any type of packet. If the POLL packet is not received by the slave or the POLL packet response is not received by the master within *newconnectionTO* number of slots, the master and the slave shall return to **page** and **page scan** substates, respectively.

8.4 DEVICE DISCOVERY SUBSTATES

In order to discover other devices a device shall enter **inquiry** substate. In this substate, it shall repeatedly transmit the inquiry message (ID packet, see Section 6.5.1.1 on page 118) at different hop frequencies. The **inquiry** hop sequence is derived from the LAP of the GIAC. Thus, even when DIACs are used, the applied hopping sequence is generated from the GIAC LAP. A device that allows itself to be discovered, shall regularly enter the **inquiry scan** substate to respond to inquiry messages. The following sections describe the message exchange and contention resolution during inquiry response. The inquiry response is optional: a device is not forced to respond to an inquiry message.

During the **inquiry** substate, the discovering device collects the Bluetooth device addresses and clocks of all devices that respond to the inquiry message. In addition, the discovering device also collects extended information (e.g. local name and supported services) from devices that respond with an extended inquiry response packet. It can then, if desired, make a connection to any one of the discovered devices by means of the previously described page procedure.

The inquiry message broadcast by the source does not contain any information about the source. However, it may indicate which class of devices should respond. There is one general inquiry access code (GIAC) to inquire for any device, and a number of dedicated inquiry access codes (DIAC) that only inquire for a certain type of device. The inquiry access codes are derived from reserved Bluetooth device addresses and are further described in Section 6.3.1.

8.4.1 Inquiry scan substate

The **inquiry scan** substate is very similar to the **page scan** substate. However, instead of scanning for the device's device access code, the receiver shall scan for the inquiry access code long enough to completely scan for 16 inquiry frequencies. Two types of scans are defined: standard and interlaced. In the case of a standard scan the length of this scan period is denoted $T_{w\_inquiry\_scan}$ (11.25ms default, see HCI [Part E] Section 7.3.22 on page 595). The standard scan is performed at a single hop frequency as defined by Xir_{4-0} (see Section 2.6.4.6 on page 95). The interlaced scan is performed as two back to back scans of $T_{w\_inquiry\_scan}$ where the first scan is on the normal hop frequency and the second scan is defined by $[Xir_{4-0} + 16] \bmod 32$. If the scan interval is not at least twice the scan window then interlaced scan shall not be used. The inquiry procedure uses 32 dedicated inquiry hop frequencies according to the inquiry hopping sequence. These frequencies are determined by the general inquiry address. The phase is determined by the native clock of the device carrying out the **inquiry scan**; the phase changes every 1.28s.

Instead of, or in addition to, the general inquiry access code, the device may scan for one or more dedicated inquiry access codes. However, the scanning shall follow the inquiry scan hopping sequence determined by the general inquiry address. If an inquiry message is received during an inquiry wake-up period, the device shall enter the **inquiry response** substate.

The **inquiry scan** substate can be entered from the **STANDBY** state or the **CONNECTION** state. In the **STANDBY** state, no connection has been established and the device can use all the capacity to carry out the **inquiry scan**. Before entering the **inquiry scan** substate from the **CONNECTION** state, the device should reserve as much capacity as possible for scanning. If desired, the device may place ACL logical transports in Sniff, Hold, or Park. Synchronous logical transports are preferably not interrupted by the **inquiry scan**, although eSCO retransmissions should be paused during the scan. In this case, the **inquiry scan** may be interrupted by the reserved synchronous slots. SCO packets should be used requiring the least amount of capacity (**HV3** packets). The scan window, $T_{w\ inquiry\ scan}$, shall be increased to increase the probability to respond to an inquiry message. If one SCO logical transport is present using HV3 packets and T_{SCO}=6 slots or one eSCO logical transport is present using EV3 packets and T_{ESCO}=6 slots, a total scan window of at least 36 slots (22.5ms) is recommended; if two SCO links are present using HV3 packets and T_{SCO}=6 slots or two eSCO links are present using EV3 packets and T_{ESCO}=6 slots, a total scan window of at least 54 slots (33.75ms) is recommended.

The scan interval $T_{inquiry\ scan}$ is defined as the interval between two consecutive inquiry scans. The **inquiry scan** interval shall be less than or equal to 2.56 s.

8.4.2 Inquiry Substate

The **inquiry** substate is used to discover new devices. This substate is very similar to the **page** substate; the TX/RX timing shall be the same as in paging, see Section 2.4.4 on page 79 and Figure 2.7 on page 78. The TX and RX frequencies shall follow the inquiry hopping sequence and the inquiry response hopping sequence, and are determined by the general inquiry access code and the native clock of the discovering device. In between inquiry transmissions, the receiver shall scan for inquiry response messages. When a response is received, the entire packet (an FHS packet) shall be read. If the EIR bit in the FHS packet is set to one, the device should try to receive the extended inquiry response packet 1250μs after the start of the FHS packet. After this, the device shall continue with inquiry transmissions. The device in an **inquiry** substate shall not acknowledge the inquiry response messages. If enabled by the Host (see HCI [Part E] Section 7.3.50 on page 632), the RSSI value of the inquiry response message shall be measured. It shall keep probing at different hop channels and in between listening for response packets. As in the **page** substate, two 10 ms trains **A** and **B** are defined, splitting the 32 frequencies of the inquiry hopping sequence into two 16-hop parts. A single train shall be repeated for at least $N_{inquiry}$=256 times before a new train is used. In order to collect all responses in an error-free environment, at least three train switches must have taken place. As a result, the **inquiry** substate may have to last for 10.24 s unless the inquirer collects enough responses and aborts the **inquiry** substate earlier. If desired, the inquirer may also prolong the **inquiry** substate to increase the probability of receiving all responses in an error-prone environment. If an inquiry procedure is automatically initiated periodically (say a 10 s period every minute), then the interval between two inquiry instances shall be determined randomly. This is done to avoid two devices synchronizing their inquiry procedures.

The **inquiry** substate is continued until stopped by the Baseband Resource Manager (when it decides that it has sufficient number of responses), when a timeout has been reached (*inquiryTO*), or by a command from the host to cancel the inquiry procedure.

The **inquiry** substate can be entered from the **STANDBY** state or the **CONNECTION** state. In the **STANDBY** state, no connection has been established and the device can use all the capacity to carry out the inquiry. Before entering the **inquiry** substate from the **CONNECTION** state, the device should free as much capacity as possible for scanning. To ensure this, it is recommended that the ACL logical transports are placed in Sniff, Hold, or Park. However, the reserved slots of synchronous logical transports shall not be disturbed by the inquiry. This means that the inquiry will be interrupted by the reserved SCO and eSCO slots which have higher priority than the inquiry. In order to obtain as much capacity as possible for inquiry, it is recommended to use the SCO packets which use the least amount of capacity (**HV3** packets). If SCO or eSCO links are present, the repetition number $N_{inquiry}$ shall be increased, see Table 8.4 on page 163.

Here it has been assumed that **HV3** packets are used with an interval T_{SCO}=6 slots or **EV3** packets are used with an interval of T_{ESCO}=6 slots, which would correspond to a 64 kb/s synchronous link.

	No synchronous links	One synchronous link (HV3)	Two synchronous links (HV3)
$N_{inquiry}$	≥ 256	≥ 512	≥ 768

Table 8.4: Increase of train repetition when synchronous links are present.

If an extended inquiry response packet could not be received because of higher priority traffic, the reception failed due to HEC or CRC failure, or because the packet type is not supported by the device, the inquiry response shall be reported to higher layers as an extended inquiry response with all-zero data.

8.4.3 Inquiry Response Substate

The slave response substate for inquiries differs completely from the slave response substate applied for pages. When the inquiry message is received in the inquiry scan substate, the recipient shall return an inquiry response (FHS) packet containing the recipient's device address (BD_ADDR) and other parameters. If the recipient has non-zero extended inquiry response data to send it shall return an extended inquiry response packet after the FHS packet.

The following protocol in the slave's **inquiry response** shall be used. On the first inquiry message received in the inquiry scan substate the slave shall enter the inquiry response substate. If the slave has non-zero extended inquiry response data to send it shall return an FHS packet with the EIR bit set to one to the master 625µs after the inquiry message was received. It shall then return an extended inquiry response packet 1250µs after the start of the FHS packet. If the slave's extended inquiry response data is all zeroes the slave shall only return an FHS packet with the EIR bit set to zero. A contention problem could arise when several devices are in close proximity to the inquiring device and all respond to an inquiry message at the same time. However, because every device has a free running clock it is highly unlikely that they all use the same phase of the inquiry hopping sequence. In order to avoid repeated collisions between devices that wake up in the same inquiry hop channel simultaneously, a device shall back-off for a random period of time. Thus, if the device receives an inquiry message and returns an FHS packet, it shall generate a random number, RAND, between 0 and MAX_RAND. For scanning intervals ≥ 1.28s MAX_RAND shall be 1023, however, for scanning intervals < 1.28s MAX_RAND may be as small as 127. A profile that uses a special DIAC may choose to use a smaller MAX_RAND than 1023 even when the scanning interval is ≥ 1.28s. The slave shall return to the **CONNECTION** or **STANDBY** state for the duration of at least RAND time slots. Before returning to the **CONNECTION** and **STANDBY** state, the device may go through the **page scan** substate. After at least RAND slots, the device shall add an offset of 1 to the phase

in the inquiry hop sequence (the phase has a 1.28 s resolution) and return to the **inquiry scan** substate again. If the slave is triggered again, it shall repeat the procedure using a new RAND. The offset to the clock accumulates each time an **FHS** packet is returned. During a probing window, a slave may respond multiple times, but on different frequencies and at different times. Reserved synchronous slots should have priority over response packets; that is, if a response packet overlaps with a reserved synchronous slot, it shall not be sent but the next inquiry message is awaited. If a device has extended inquiry response data to send but the extended inquiry response packet overlaps with a reserved synchronous slot the FHS packet may be sent with the EIR bit set to zero.

The messaging during the inquiry routines is summarized in Table 8.5 on page 165. In step 1, the master transmits an inquiry message using the inquiry access code and its own clock. The slave responds with the **FHS** packet containing the slave's Bluetooth device address, native clock and other slave information. This **FHS** packet is returned at times that tend to be random. The **FHS** packet is not acknowledged in the inquiry routine, but it is retransmitted at other times and frequencies as long as the master is probing with inquiry messages. If the slave has non-zero extended inquiry response data it sends an extended inquiry response packet to the master in step 3.

The extended inquiry response packet is an ACL packet with type DM1, DM3, DM5, DH1, DH3 or DH5. To minimize interference it is recommended to use the shortest packet that fits the data. The packet shall be sent on the same frequency as the FHS packet, 1250 µs after the start of the FHS packet. In the packet header, LT_ADDR shall be set to zero. TYPE shall be one of DM1, DM3, DM5, DH1, DH3 or DH5. FLOW, ARQN and SEQN shall all be set to zero and ignored during receipt. The HEC LFSR shall be initialized with the same DCI (default check initialization) as for the FHS packet. In the payload header, LLID shall contain the value 10 (start of an L2CAP message or no fragmentation). FLOW shall be set to zero and ignored upon receipt. The length of the payload body (LENGTH) shall be smaller than or equal to 240 bytes. The CRC LFSR shall be initialized with the same DCI as for the FHS packet. The data whitening LFSR shall be initialized with the same value as for the FHS packet.

The payload data has two parts, a significant part followed by a non-significant part. The significant part contains a sequence of data structures as defined in [Vol. 3, Part C] Section 8 on page 330. The non-significant part contains all-zero octets.

The baseband shall not change any octets in the significant part. When transmitting data, the non-significant part octets may be omitted from the payload.

A device shall store a single extended inquiry response packet. This packet shall be used with all IACs.

Step	Message	Packet Type	Direction	Hopping Sequence	Access Code and Clock
1	Inquiry	ID	master to slave	inquiry	inquiry
2	Inquiry response	FHS	slave to master	inquiry response	inquiry
3	Extended Inquiry Response	DM1, DM3, DM5, DH1, DH3, DH5	slave to master	inquiry response *	inquiry

Table 8.5: Messaging during inquiry routines.

* The extended inquiry response packet is sent on the same frequency as the inquiry response (FHS) packet.

8.5 CONNECTION STATE

In the **CONNECTION** state, the connection has been established and packets can be sent back and forth. In both devices, the channel (master) access code, the master's Bluetooth clock, and the AFH_channel_map are used. **CONNECTION** state uses the *basic* or *adapted channel hopping sequence*.

The **CONNECTION** state starts with a POLL packet sent by the master to verify the switch to the master's timing and channel frequency hopping. The slave may respond with any type of packet. If the slave does not receive the POLL packet or the master does not receive the response packet for *newconnectionTO* number of slots, both devices shall return to **page**/**page scan** substates.

The first information packets in the **CONNECTION** state contain control messages that characterize the link and give more details regarding the devices. These messages are exchanged between the link managers of the devices. For example, they may define the SCO logical transport and the sniff parameters. Then the transfer of user information can start by alternately transmitting and receiving packets.

The **CONNECTION** state is left through a **detach** or **reset** command. The **detach** command is used if the link has been disconnected in the normal way; all configuration data in the link controller shall remain valid. The **reset** command is a soft reset of the link controller. The functionality of the soft reset is described in [Part E] Section 7.3.2 on page 564.

In the **CONNECTION** state, if a device is not going to be nominally present on the channel at all times it may describe its unavailability by using sniff mode or hold mode (see Figure 8.5 on page 166).

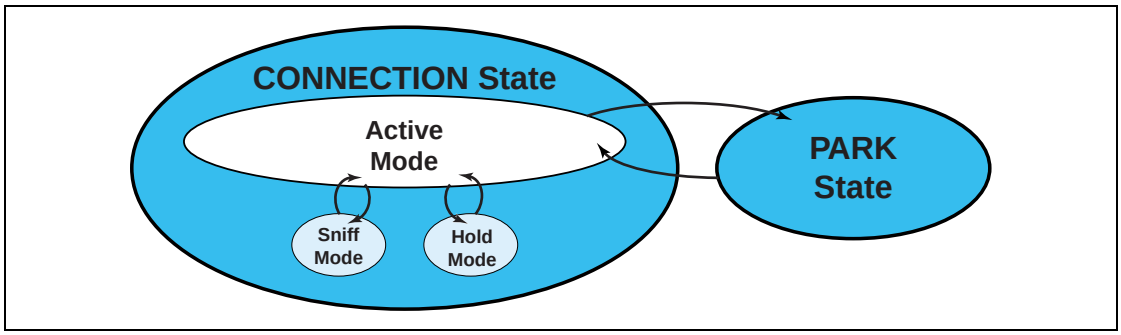


Figure 8.5: Connection state.

8.6 ACTIVE MODE

In the active mode, both master and slave actively participate on the channel. Up to seven slaves may be in the active mode at any given time in a piconet. The master schedules the transmission based on traffic demands to and from the different slaves. In addition, it supports regular transmissions to keep slaves synchronized to the channel. Slaves in the active mode listen in the master-to-slave slots for packets. These devices are known as *active slaves*. If an active slave is not addressed, it may sleep until the next new master transmission. Slaves can derive the number of slots the master has reserved for transmission from the TYPE field in the packet header; during this time, the non-addressed slaves do not have to listen on the master-to-slave slots. When a device is participating in multiple piconets it should listen in the master-to-slave slot for the current piconet. It is recommended that a device not be away from each piconet in which it is participating for more than T_{poll} slots. A periodic master transmission is required to keep the slaves synchronized to the channel. Since the slaves only need the channel access code to synchronize, any packet type can be used for this purpose.

Only the slave that is addressed by one of its LT_ADDRs (primary or secondary) may return a packet in the next slave-to-master slot. If no valid packet header is received, the slave may only respond in its reserved SCO or eSCO slave-to-master slot. In the case of a broadcast message, no slave shall return a packet (an exception is the access window for access requests in the **PARK** state, see Section 8.9 on page 184).

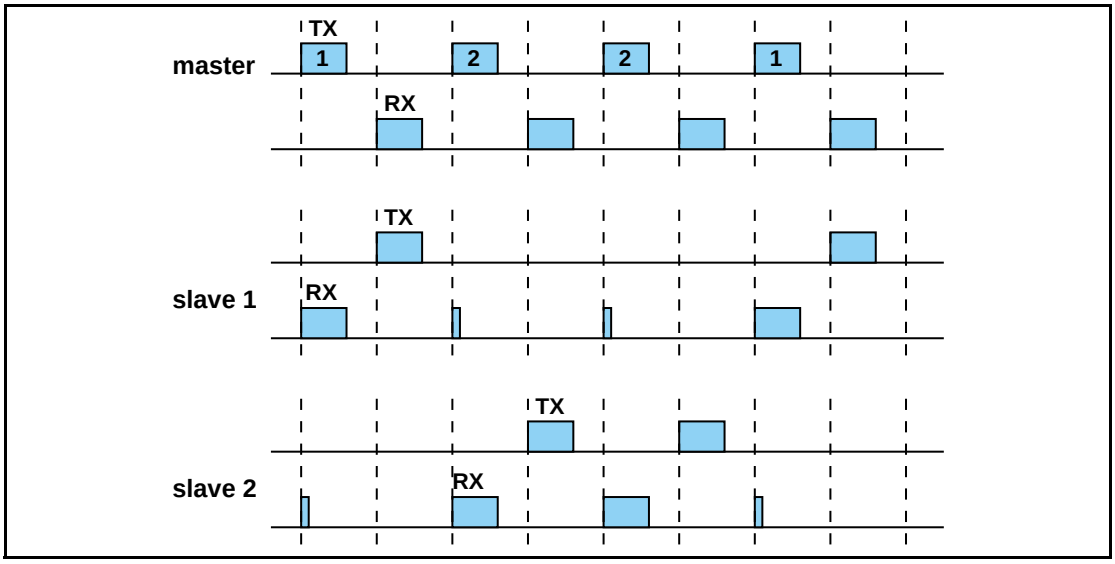


Figure 8.6: RX/TX timing in multi-slave configuration

For ACL logical transports the mode selection may be left to real time packet type selections. The packet type table (ptt) in section 6.5 allows the selection of Basic Rate or Enhanced Data Rate for each of the packet type codes, however; the DM1 packet is available in all packet type tables. ACL traffic over this given physical or logical link shall utilize the packet types in the given column of Packets defined for synchronous and asynchronous logical transport types.

8.6.1 Polling in the Active Mode

The master always has full control over the piconet. Due to the TDD scheme, slaves can only communicate with the master and not other slaves. In order to avoid collisions on the ACL logical transport, a slave is only allowed to transmit in the slave-to-master slot when addressed by the LT_ADDR in the packet header in the preceding master-to-slave slot. If the LT_ADDR in the preceding slot does not match, or a valid packet header was not received, the slave shall not transmit.

The master normally attempts to poll a slave's ACL logical transport no less often than once every T_{poll} slots. T_{poll} is set by the Link Manager (see [Part C] Section 4.1.8 on page 241).

The slave's ACL logical transport may be polled with any packet type except for FHS and ID. For example, polling during SCO may use HV packets, since the slave may respond to an HV packet with a DM1 packet (see Section 8.6.2 on page 167).

8.6.2 SCO

The SCO logical transport shall be established by the master sending a SCO setup message via the LM protocol. This message contains timing parameters

including the SCO interval T_{SCO} and the offset D_{SCO} to specify the reserved slots.

In order to prevent clock wrap-around problems, an initialization flag in the LMP setup message indicates whether initialization procedure 1 or 2 is being used. The slave shall apply the initialization method as indicated by the initialization flag. The master shall use initialization 1 when the MSB of the current master clock (CLK_{27}) is 0; it shall use initialization 2 when the MSB of the current master clock (CLK_{27}) is 1. The master-to-slave SCO slots reserved by the master and the slave shall be initialized on the slots for which the clock satisfies the applicable equation:

$$CLK_{27-1} \bmod T_{SCO} = D_{SCO} \qquad \text{for initialization 1}$$

$$(\overline{CLK_{27}}, CLK_{26-1}) \bmod T_{SCO} = D_{SCO} \qquad \text{for initialization 2}$$

The slave-to-master SCO slots shall directly follow the reserved master-to-slave SCO slots. After initialization, the clock value CLK(k+1) for the next master-to-slave SCO slot shall be derived by adding the fixed interval T_{SCO} to the clock value of the current master-to-slave SCO slot:

$$CLK(k+1) = CLK(k) + T_{SCO}$$

The master will send SCO packets to the slave at regular intervals (the SCO interval T_{SCO} counted in slots) in the reserved master-to-slave slots. An HV1 packet can carry 1.25ms of speech at a 64 kb/s rate. An HV1 packet shall therefore be sent every two time slots (T_{SCO}=2). An HV2 packet can carry 2.5ms of speech at a 64 kb/s rate. An HV2 packet shall therefore be sent every four time slots (T_{SCO}=4). An HV3 packet can carry 3.75ms of speech at a 64 Kobe/s rate. An HV3 packet shall therefore be sent every six time slots (T_{SCO}=6).

The slave is allowed to transmit in the slot reserved for its SCO logical transport unless the (valid) LT_ADDR in the preceding slot indicates a different slave. If no valid LT_ADDR can be derived in the preceding slot, the slave may still transmit in the reserved SCO slot.

Since the DM1 packet is recognized on the SCO logical transport, it may be sent during the SCO reserved slots if a valid packet header with the primary LT_ADDR is received in the preceding slot. DM1 packets sent during SCO reserved slots shall only be used to send ACL-C data.

The slave shall not transmit anything other than an HV packet in a reserved SCO slot unless it decodes its own slave address in the packet header of the packet in the preceding master-to-slave transmission slot.

8.6.3 eSCO

The eSCO logical transport is established by the master sending an eSCO setup message via the LM protocol. This message contains timing parameters including the eSCO interval T_{ESCO} and the offset D_{ESCO} to specify the reserved slots.

To enter eSCO, the master or slave shall send an eSCO setup command via the LM protocol. This message shall contain the eSCO interval T_{ESCO} and an offset D_{ESCO}. In order to prevent clock wrap-around problems, an initialization flag in the LMP setup message indicates whether initialization procedure 1 or 2 shall be used. The initiating device shall use initialization 1 when the MSB of the current master clock (CLK_{27}) is 0; it shall use initialization 2 when the MSB of the current master clock (CLK_{27}) is 1. The responding device shall apply the initialization method as indicated by the initialization flag. The master-to-slave eSCO slots reserved by the master and the slave shall be initialized on the slots for which the clock satisfies the applicable equation:

$$CLK_{27-1} \bmod T_{ESCO} = D_{ESCO} \qquad \text{for initialization 1}$$

$$(\overline{CLK_{27}}, CLK_{26-1}) \bmod T_{ESCO} = D_{ESCO} \qquad \text{for initialization 2}$$

The slave-to-master eSCO slots shall directly follow the reserved master-to-slave eSCO slots. After initialization, the clock value CLK(k+1) for the next master-to-slave eSCO slot shall be found by adding the fixed interval T_{ESCO} to the clock value of the current master-to-slave eSCO slot:

$$CLK(k+1) = CLK(k) + T_{ESCO}$$

When an eSCO logical transport is established, the master shall assign an additional LT_ADDR to the slave. This provides the eSCO logical transport with an ARQ scheme that is separate from that of the ACL logical transport. All traffic on a particular eSCO logical transport, and only that eSCO traffic, is carried on the eSCO LT_ADDR. The eSCO ARQ scheme uses the ARQN bit in the packet header, and operates similarly to the ARQ scheme on ACL links.

There are two different polling rules in eSCO. In the eSCO reserved slots, the polling rule is the same as to the SCO reserved slots. The master may send a packet in the master slot. The slave may transmit on the eSCO LT_ADDR in the following slot either if it received a packet on the eSCO LT_ADDR in the previous slot, or if it did not receive a valid packet header in the previous slot. When the master-to-slave packet type is a three-slot packet, the slave's transmit slot is the fourth slot of the eSCO reserved slots. A master shall transmit in an eSCO retransmission window on a given eSCO LT_ADDR only if it addressed that eSCO LT_ADDR in the immediately preceding eSCO reserved slots. A slave may transmit on an eSCO LT_ADDR in the eSCO reserved slots only if the slave did not received a valid packet header with a different LT_ADDR in the eSCO reserved slots. Inside retransmission windows, the

same polling rule as for ACL traffic is used: the slave shall transmit on the eSCO channel only if it received a valid packet header and correct LT_ADDR on the eSCO channel in the previous master-to-slave transmission slot. The master may transmit on any non-eSCO LT_ADDR in any master-to-slave transmission slot inside the eSCO retransmission window. The master shall only transmit on an eSCO LT_ADDR in the retransmission window if there are enough slots left for both the master and slave packets to complete in the retransmission window. The master may refrain from transmitting in any slot during the eSCO retransmission window. When there is no data to transmit from master to slave, either due to the traffic being unidirectional or due to the master-to-slave packet having been ACK'ed, the master shall use the POLL packet. When the master-to-slave packet has been ACK'ed, and the slave-to-master packet has been correctly received, the master shall not address the slave on the eSCO LT_ADDR until the next eSCO reserved slot, with the exception that the master may transmit a NULL packet with ARQN=ACK on the eSCO LT_ADDR. When there is no data to transmit from slave to master, either due to the traffic being unidirectional or due to the slave-to-master packet having been ACK'ed, the slave shall use NULL packets. eSCO traffic should be given priority over ACL traffic in the retransmission window.

Figure 8.7 on page 170 shows the eSCO window when single slot packets are used.

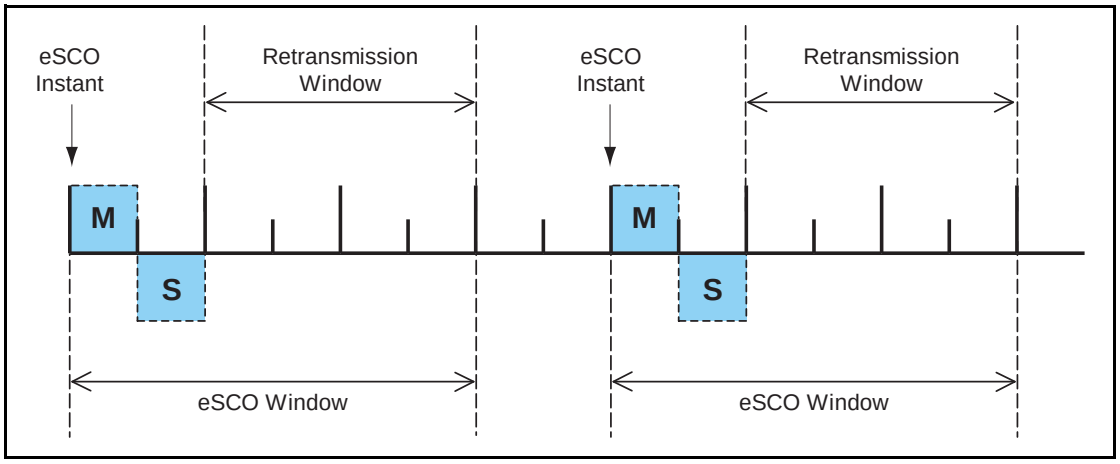


Figure 8.7: eSCO Window Details for Single-Slot Packets

When multi-slot packets are used in either direction of the eSCO logical transport, the first transmission continues into the following slots. The retransmission window in this case starts the slot after the end of the slave-to-master packet, i.e. two, four or six slots immediately following the eSCO instant are reserved and should not be used for other traffic. Figure 8.8 on page 171 shows the eSCO window when multi-slot packets are used in one direction and single-slot packets are used in the other direction.

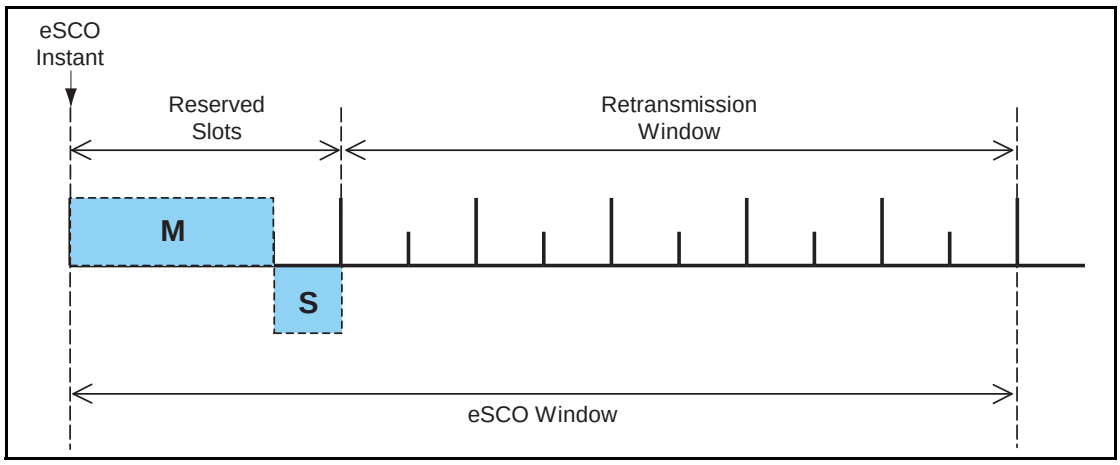


Figure 8.8: eSCO Window Details for Asymmetric Traffic

eSCO windows may overlap on the master, but shall not overlap on an individual slave.

In the reserved slots both master and slave shall only listen and transmit at their allocated slots at the first transmission time of each eSCO window. Intermittent lapses due to, for instance, time-critical signaling during connection establishment are allowed. Both master and slave may refrain from sending data and may use instead POLL and NULL packets respectively. When the master transmits a POLL packet instead of an EV4 or EV5 packet the slave shall transmit starting in the same slot as if the master transmitted an EV4 or EV5 packet. If the slave does not receive anything in the reserved master-to-slave transmission slot it shall transmit in the same slot as if the master had transmitted the negotiated packet type. For example, if the master had negotiated an EV5 packet the slave would transmit three slots later. [If the master does not receive a slave transmission in response to an eSCO packet it causes an implicit NAK of the packet in question. The listening requirements for the slave during the retransmission window are the same as for an active ACL logical transport.

8.6.4 Broadcast Scheme

The master of the piconet can broadcast messages to all slaves on the ASB-U, PSB-C, and PSB-U logical transports. A broadcast packet shall have an LT_ADDR set to all zero. Each new broadcast message (which may be carried by a number of packets) shall start with the start of L2CAP message indication (LLID=10).

The Broadcast LT_ADDR shall use a ptt=0.

A broadcast packet shall never be acknowledged. In an error-prone environment, the master may carry out a number of retransmissions to increase the probability for error-free delivery, see also Section 7.6.5 on page 148.

In order to support the **PARK** state (as described in Section 8.9 on page 184), a master transmission shall take place at fixed intervals. This master transmission will act as a beacon to which slaves can synchronize. If no traffic takes place at the beacon event, broadcast packets shall be sent. More information is given in Section 8.9 on page 184.

8.6.5 Role Switch

There are several occasions when a role switch is used.

- a role switch is necessary when joining an existing piconet by paging, since by definition, the paging device is initially master of a "small" piconet only involving the pager (master) and the paged (slave) device.
- a role switch is necessary in order for a slave in an existing piconet to set up a new piconet with itself as master and the original piconet master as slave. If the original piconet had more than one slave, then this implies a double role for the original piconet master; it becomes a slave in the new piconet while still maintaining the original piconet as master.

Prior to the role switch, encryption if present, shall be paused or disabled in the old piconet. A role switch shall not be performed if the physical link is in Sniff or Hold mode, in the **PARK** state, or has any synchronous logical transports.

For the master and slave involved in the role switch, the switch results in a reversal of their TX and RX timing: a TDD switch. Additionally, since the piconet parameters are derived from the Bluetooth device address and clock of the master, a role switch inherently involves a redefinition of the piconet as well: a piconet switch. The new piconet's parameters shall be derived from the former slave's device address and clock.

Assume device A is to become master; device B was the former master. Then there are two alternatives: either the slave initiates the role switch or the master initiates the role switch. These alternatives are described in Link Manager Protocol, [Part C] Section 4.4.2 on page 282. The baseband procedure is the same regardless of which alternative is used.

In step 1, the slave A and master B shall perform a TDD switch using the former hopping scheme (still using the Bluetooth device address and clock of device B), so there is no piconet switch yet. The slot offset information sent by slave A shall not be used yet but shall be used in step 3. Device A now becomes the master, device B the slave. The LT_ADDR formerly used by device A in its slave role, shall now be used by slave B.

At the moment of the TDD switch, both devices A and B shall start a timer with a time out of *newconnectionTO*. The timer shall be stopped in slave B as soon as it receives an FHS packet from master A on the TDD-switched channel. The timer shall be stopped in master A as soon as it receives an ID packet from slave B. If the *newconnectionTO* expires, the master and slave shall return to the old piconet timing and AFH state, taking their old roles of master and slave. The FHS packet shall be sent by master A using the "old" piconet parameters.

The LT_ADDR in the FHS packet header shall be the former LT_ADDR used by device A. The LT_ADDR carried in the FHS payload shall be the new LT_ADDR intended for device B when operating on the new piconet. After the FHS acknowledgment, which is the ID packet and shall be sent by the slave on the old hopping sequence, both master A and slave B shall use the new channel parameters of the new piconet as indicated by the FHS with the sequence selection set to *basic channel hopping sequence*. If the new master has physical links that are *AFH enabled*, following the piconet switch the new master is responsible for controlling the AFH operational mode of its new slave.

Since the old and new masters' clocks are synchronized, the clock information sent in the FHS payload shall indicate the new master's clock at the beginning of the FHS packet transmission. Furthermore, the 1.25 ms resolution of the clock information given in the FHS packet is not sufficient for aligning the slot boundaries of the two piconets. The slot-offset information in the LMP message previously sent by device A shall be used to provide more accurate timing information. The slot offset indicates the delay between the start of the master-to-slave slots of the old and new piconet channels. This timing information ranges from 0 to 1249 μs with a resolution of 1μs. It shall be used together with the clock information in the FHS packet to accurately position the correlation window when switching to the new master's timing after acknowledgment of the FHS packet.

After reception of the FHS packet acknowledgment, the new master A shall switch to its own timing with the sequence selection set to the *basic channel hopping sequence* and shall send a POLL packet to verify the switch. Both the master and the slave shall start a new timer with a time out of *newconnectionTO* on FHS packet acknowledgment. The start of this timer shall be aligned with the beginning of the first master TX slot boundary of the new piconet, following the FHS packet acknowledgment. The slave shall stop the timer when the POLL packet is received; the master shall stop the timer when the POLL packet is acknowledged. The slave shall respond with any type of packet to acknowledge the POLL. Any pending AFH_Instant shall be cancelled once the POLL packet has been received by the slave. If no response is received, the master shall re-send the POLL packet until *newconnectionTO* is reached. If this timer expires, both the slave and the master shall return to the old piconet timing with the old master and slave roles. Expiry of the timer shall also restore the state associated with AFH (including any pending AFH_Instant), Channel Quality Driven Data Rate (CQDDR, Link Manager Protocol [Part C] Section 4.1.7 on page 240) and power control (Link Manager Protocol [Part C] Section 4.1.3 on page 230). The procedure may then start again beginning at step 1. Aligning the timer with TX boundaries of the new piconet ensures that no device returns to the old piconet timing in the middle of a master RX slot.

After the role switch the ACL logical transport is reinitialized as if it were a new connection. For example, the SEQN of the first data packet containing a CRC on the new piconet channel shall be set according to the rules in Section 7.6.2 on page 144.

A parked slave shall be unparked before it can participate in a role switch.

8.6.6 Scatternet

Multiple piconets can cover the same area. Since each piconet has a different master, the piconets hop independently, each with their own hopping sequence and phase as determined by the respective master. In addition, the packets carried on the channels are preceded by different channel access codes as determined by the master device addresses. As more piconets are added, the probability of collisions increases; a graceful degradation of performance results as is common in frequency-hopping spread spectrum systems.

If multiple piconets cover the same area, a device can participate in two or more overlaying piconets by applying time multiplexing. To participate on the proper channel, it shall use the associated master device address and proper clock offset to obtain the correct phase. A device can act as a slave in several piconets, but only as a master in a single piconet: since two piconets with the same master are synchronized and use the same hopping sequence, they are one and the same piconet. A group of piconets in which connections exist between different piconets is called a *scatternet*.

A master or slave can become a slave in another piconet by being paged by the master of this other piconet. On the other hand, a device participating in one piconet can page the master or slave of another piconet. Since the paging device always starts out as master, a master-slave role switch is required if a slave role is desired. This is described in Section 8.6.5 on page 172.

8.6.6.1 Inter-piconet communications

Time multiplexing must be used to switch between piconets. Devices may achieve the time multiplexing necessary to implement scatternet by using sniff mode or by remaining in an active ACL connection. For an ACL connection in piconets where the device is a slave in the **CONNECTION** state, it may choose not to listen in every master slot. In this case it should be recognized that the quality of service on this link can degrade abruptly if the slave is not present enough to match up with the master polling that slave. Similarly, in piconets where the device is master it may choose not to transmit in every master slot. In this case it is important to honor T_{poll} as much as possible. Devices in sniff mode could have sufficient time to visit another piconet in between sniff slots. When the device is a slave using sniff mode and there are not sufficient idle slots, the device may choose to not listen to all master transmission slots in the sniff_attempts period or during the subsequent sniff_timeout period. A master is not required to transmit during sniff slots and therefore has flexibility for scatternet. If SCO or eSCO links are established, other piconets shall only be visited in the non-reserved slots in between reserved slots. This is only possible if there is a single SCO logical transport using HV3 packets or eSCO links where at least four slots remain in between the reserved slots. Since the multiple piconets are not synchronized, guard time must be left to account for misalignment. This means that only 2 slots can effectively be used to visit another piconet in between the HV3 packets.

Since the clocks of two masters of different piconets are not synchronized, a slave device participating in two piconets shall maintain two offsets that, added to its own native clock, create the two master clocks. Since the two master clocks drift independently, the slave must regularly update the offsets in order to keep synchronization to both masters.

8.6.7 Hop Sequence Switching

Hop sequence adaptation is controlled by the master and can be set to either *enabled* or *disabled*. Once enabled, hop sequence adaptation shall apply to all logical transports on a physical link. Once enabled, the master may periodically update the set of *used* and *unused* channels as well as disable hop sequence adaptation on a physical link. When a master has multiple physical links the state of each link is independent of all other physical links.

When hop sequence adaptation is enabled, the *sequence selection* hop selection kernel input is set to *adapted channel hopping sequence* and the *AFH_channel_map* input is set to the appropriate set of *used* and *unused* channels. Additionally, the *same channel* mechanism shall be used. When hop sequence adaptation is enabled with all channels *used* this is known as AHS(79).

When hop sequence adaptation is disabled, the *sequence selection* input of the hop selection kernel is set to *basic channel hopping sequence* (the *AFH_channel_map* input is unused in this case) and the *same channel* mechanism shall not be used.

The hop sequence adaptation state shall be changed when the master sends the LMP_set_AFH PDU and a baseband acknowledgement is received. When the baseband acknowledgement is received prior to the hop sequence switch instant, *AFH_Instant*, (See Link Manager Protocol [Part C] Section 4.1.4 on page 234) the hop sequence proceeds as shown in Figure 8.9 on page 175.

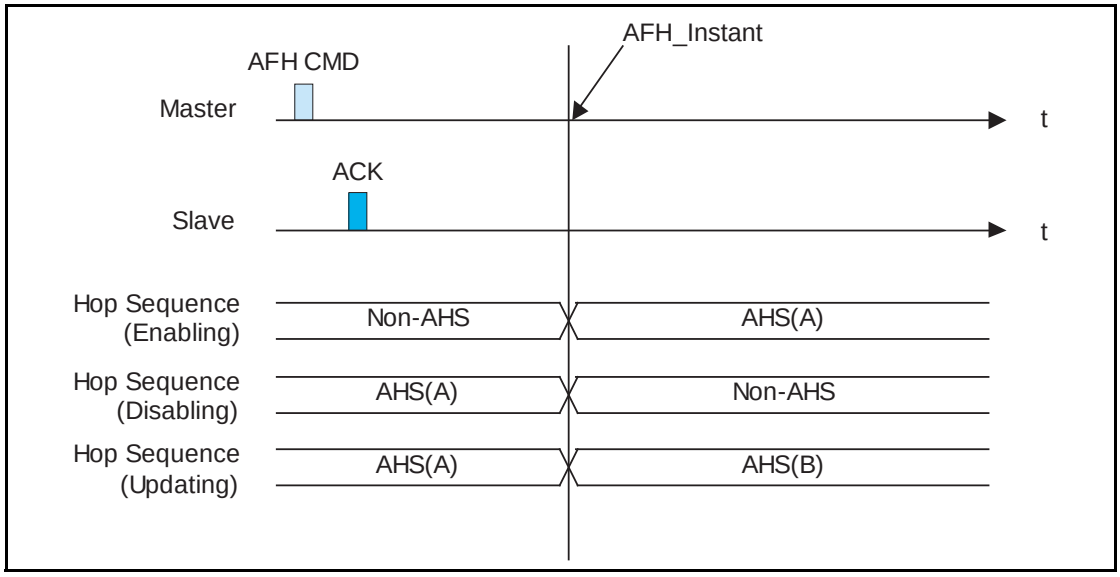


Figure 8.9: Successful hop sequence switching

When the baseband acknowledgement is not received prior to the *AFH_Instant* the master shall use a recovery hop sequence for the slave(s) that did not respond with an acknowledgement (this may be because the slave did not hear the master's transmission or the master did not hear the slave's transmission). When hop sequence adaptation is being enabled or disabled the recovery sequence shall be the *AFH_channel_map* specified in the LMP_set_AFH PDU. When the *AFH_channel_map* is being updated the master shall choose a recovery sequence that includes all of the RF channels marked as *used* in either the old or new *AFH_channel_map,* e.g. AHS(79). Once the baseband acknowledgement is received the master shall use the *AFH_channel_map* in the LMP_set_AFH PDU starting with the next transmission to the slave. See Figure 8.10 on page 176.

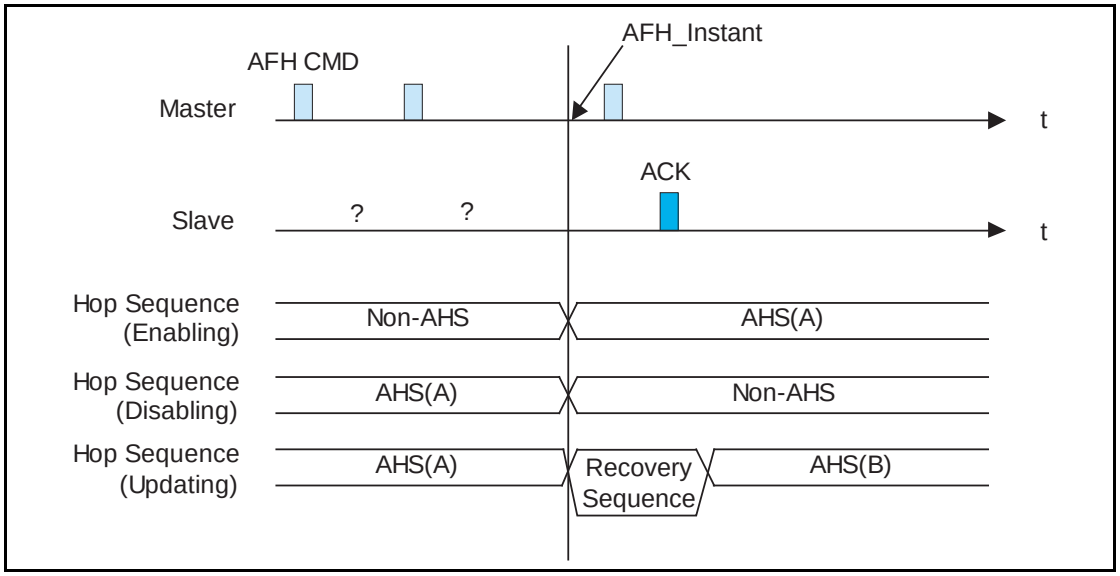


Figure 8.10: Recovery hop sequences

When the *AFH_Instant* occurs during a multi-slot packet transmitted by the master, the slave shall use the same hopping sequence parameters as the master used at the start of the multi-slot packet. An example of this is shown in Figure 8.11 on page 177. In this figure the *basic channel hopping sequence* is designated f. The first *adapted channel hopping sequence* is designated with f', and the second *adapted channel hopping sequence* is designated f".

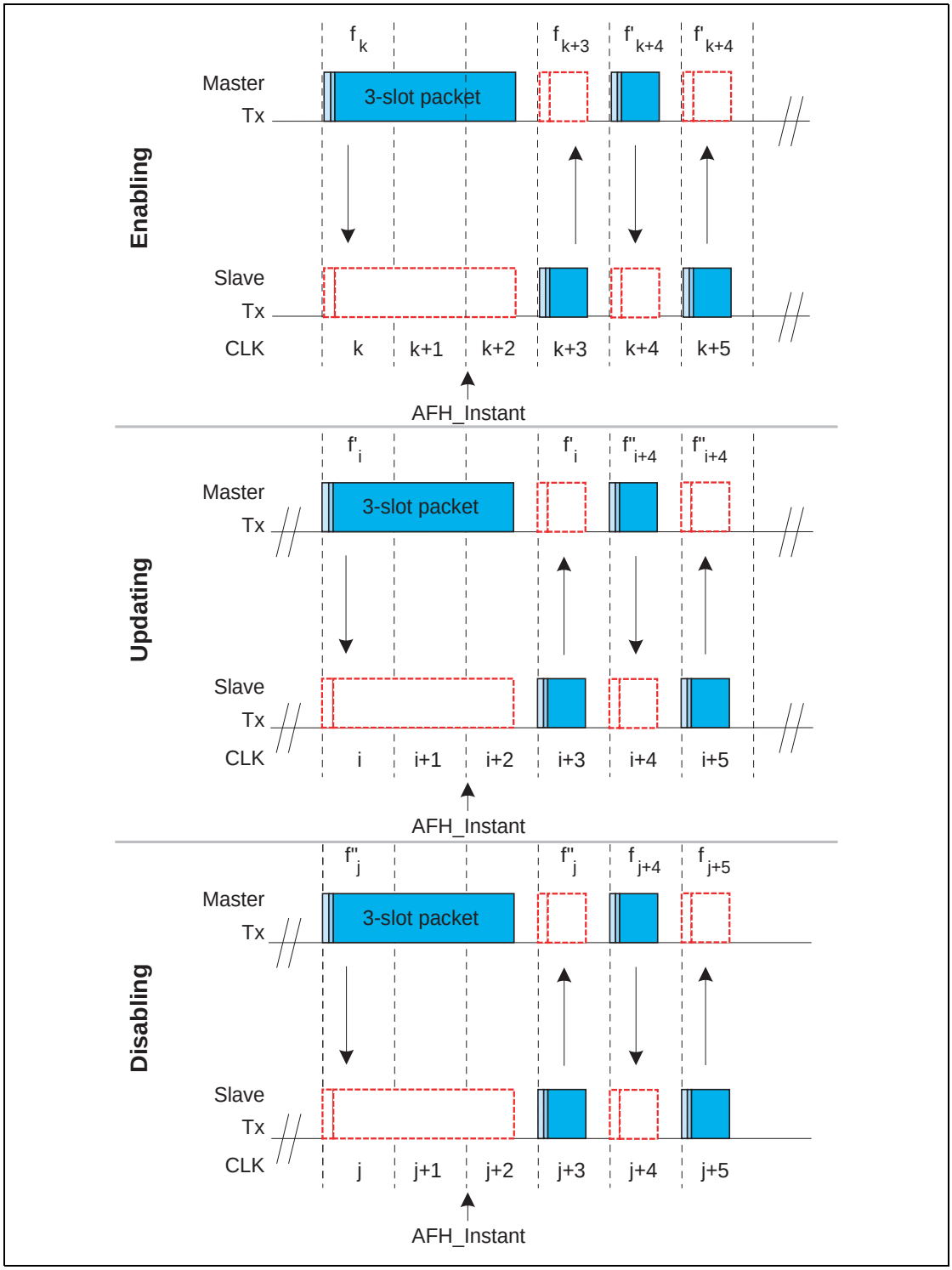


Figure 8.11: AFH_Instant changes during multi-slot packets transmitted by the master.

8.6.8 Channel Classification and Channel Map Selection

RF channels are classified as being *unknown, bad or good*. These classifications are determined individually by the master and slaves based on local information (e.g. active or passive channel assessment methods or from the Host via HCI). Information received from other devices via LMP (for example, an *AFH_channel_map* from a master or a channel classification report from a slave) shall not be included in a device's channel classification.

The three possible channel classifications shall be as defined in Table 8.6 on page 178.

Classification	Definition
unknown	A channel shall be classified as *unknown* if the channel assessment measurements are insufficient to reliably classify the channel, and the channel is not marked as *bad* in the most recent HCI *Set_AFH_Channel_Classification*.
bad	A channel may be classified as *bad* if an ACL or synchronous throughput failure measure associated with it has exceeded a threshold (defined by the particular channel assessment algorithm employed). A channel may also be classified as *bad* if an interference-level measure associated with it, determining the interference level that the link poses upon other systems in the vicinity, has exceeded a threshold (defined by the particular channel assessment algorithm employed). A channel shall be classified as *bad* when it is marked as *bad* in the most recent HCI *Set_AFH_Channel_Classification*
good	A channel shall be classified as *good* if it is not either *unknown* or *bad*.

Table 8.6: Channel classification descriptions

A master with AFH enabled physical links shall determine an *AFH_channel_map* based on any combination of the following information:

- Channel classification from local measurements (e.g. active or passive channel assessment in the Controller), if supported and enabled. The Host may enable or disable local measurements using the HCI *Write_AFH_Channel_Classification_Mode* command, defined in the HCI Functional Specification [Part E] Section 7.3.54 on page 636 if HCI is present.
- Channel classification information from the Host using the HCI *Set_AFH_channel_classification* command, defined in the HCI Functional Specification [Part E] Section 7.3.46 on page 628 if HCI is present. Channels classified as *bad* in the most recent *AFH_Host_Channel_Classification* shall be marked as *unused* in the *AFH_channel_map*.
- Channel classification reports received from slaves in *LMP_channel_classification* PDUs, defined in the LMP Specification [Part C] Section 4.1.5 on page 237.

The algorithm used by the master to combine these information sources and generate the *AFH_channel_map* is not defined in the specification and will be implementation specific. At no time shall the number of channels used be less than N_{min}, defined in Section 2.3.1 on page 76.

If a master that determines that all channels should be *used* it may keep AFH operation enabled using an *AFH_channel_map* of 79 *used* channels, i.e. AHS(79).

8.6.9 Power Management

Features are provided to allow low-power operation. These features are both at the microscopic level when handling the packets, and at the macroscopic level when using certain operation modes.

8.6.9.1 Packet handling

In order to minimize power consumption, packet handling is minimized both at TX and RX sides. At the TX side, power is minimized by only sending useful data. This means that if only link control information needs to be exchanged, **NULL** packets may be used. No transmission is required if there is no link control information to be sent, or if the transmission would only involve a NAK (NAK is implicit on no reply). If there is data to be sent, the payload length shall be adapted in order to send only the valid data bytes. At the RX side, packet processing takes place in different steps. If no valid access code is found in the search window, the transceiver may return to sleep. If an access code is found, the receiver device shall start to process the packet header. If the HEC fails, the device may return to sleep after the packet header. A valid header indicates if a payload will follow and how many slots are involved.

8.6.9.2 Slot occupancy

As was described in Section 6.5 on page 117, the packet type indicates how many slots a packet may occupy. A slave not addressed in the packet header may go to sleep for the remaining slots the packet occupies. This can be read from the TYPE code.

8.6.9.3 Recommendations for low-power operation

The most common and flexible methods for reducing power consumption are the use of sniff and park. Hold can also be used by repeated negotiation of hold periods.

8.6.9.4 Enhanced Data Rate

Enhanced Data Rate provides power saving because of the ability to send a given amount of data in either fewer packets or with the same (or similar) number of packets but with shorter payloads.

8.7 SNIFF MODE

In Sniff mode, the duty cycle of the slave's activity in the piconet may be reduced. If a slave is in active mode on an ACL logical transport, it shall listen in every ACL slot to the master traffic, unless that link is being treated as a scatternet link or is absent due to hold mode. With sniff mode, the time slots when a slave is listening are reduced, so the master shall only transmit to a slave in specified time slots. The sniff anchor points are spaced regularly with an interval of T_{sniff}.

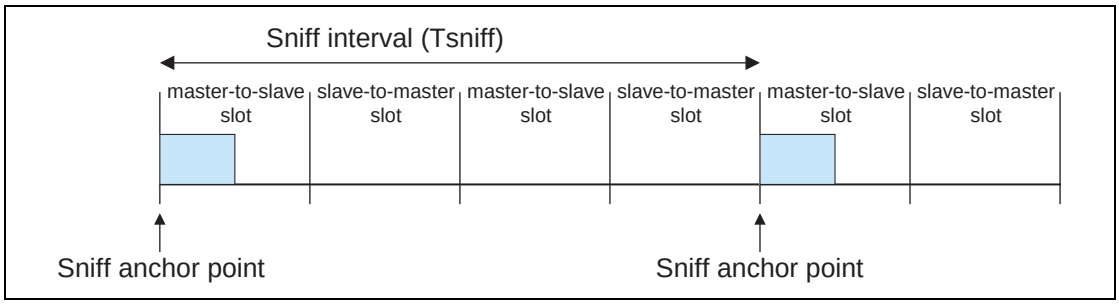


Figure 8.12: Sniff anchor points

The slave listens in master-to-slave transmission slots starting at the sniff anchor point. It shall use the following rules to determine whether to continue listening:

- If fewer than $N_{sniff\ attempt}$ master-to-slave transmission slots have elapsed since the sniff anchor point then the slave shall continue listening.
- If the slave has received a packet with a matching LT_ADDR that contains ACL data (DM, DH, DV, or AUX1 packets) in the preceding $N_{sniff\ timeout}$ master-to-slave transmission slots then it shall continue listening.
- If the slave has transmitted a packet containing ACL data (DM, DH, DV, or AUX1 packets) in the preceding $N_{sniff\ timeout}$ slave-to-master transmission slots then it shall continue listening.
- If the slave has received any packet with a matching LT_ADDR in the preceding $N_{sniff\ timeout}$ master-to-slave transmission slots then it may continue listening.
- A device may override the rules above and stop listening prior to $N_{sniff\ timeout}$ or the remaining $N_{sniff\ attempt}$ slots if it has activity in another piconet.

It is possible that activity from one sniff timeout may extend to the next sniff anchor point. Any activity from a previous sniff timeout shall not affect activity

after the next sniff anchor point. So in the above rules, only the slots since the last sniff anchor point are considered.

Note that $N_{sniff\ attempt}$ =1 and $N_{sniff\ timeout}$ =0 cause the slave to listen only at the slot beginning at the sniff anchor point, irrespective of packets received from the master.

$N_{sniff\ attempt}$ =0 shall not be used.

Sniff mode only applies to asynchronous logical transports and their associated LT_ADDR. Sniff mode shall not apply to synchronous logical transports, therefore, both masters and slaves shall still respect the reserved slots and retransmission windows of synchronous links.

To enter sniff mode, the master or slave shall issue a sniff command via the LM protocol. This message includes the sniff interval T_{sniff} and an offset $D_{sniff.}$ In addition, an initialization flag indicates whether initialization procedure 1 or 2 shall be used. The device shall use initialization 1 when the MSB of the current master clock (CLK_{27}) is 0; it shall use initialization 2 when the MSB of the current master clock (CLK_{27}) is 1. The slave shall apply the initialization method as indicated by the initialization flag irrespective of its clock bit value CLK_{27}. The sniff anchor point determined by the master and the slave shall be initialized on the slots for which the clock satisfies the applicable equation:

$$CLK_{27-1} \bmod T_{sniff} = D_{sniff} \qquad \text{for initialization 1}$$

$$(\overline{CLK_{27}}, CLK_{26-1}) \bmod T_{sniff} = D_{sniff} \qquad \text{for initialization 2}$$

this implies that D_{sniff} must be even

After initialization, the clock value CLK(k+1) for the next sniff anchor point shall be derived by adding the fixed interval T_{sniff} to the clock value of the current sniff anchor point:

$$CLK(k+1) = CLK(k) + T_{sniff}$$

8.7.1 Sniff Transition Mode

Sniff transition mode is a special mode which is used during the transition between sniff and active mode. It is required because during this transition it is unclear which mode (Sniff or Active) the slave is in and it is necessary to ensure that the slave is polled correctly regardless of which mode it is in.

In sniff transition mode the master shall maintain the active mode poll interval in case the slave is in active mode. In addition the master shall poll the slave at least once in the sniff attempt transmit slots starting at each sniff anchor point. Note that this transmission counts for the active mode polling as well. The master shall use its high power accurate clock when in Sniff Transition Mode.

The precise circumstances under which the master enters Sniff Transition Mode are defined in [Part C] Section 4.5.3.1 on page 294.

Only LMP data shall ever be transferred in sniff transition mode. The ACL-C logical link shall carry control information exchanged between the link managers of the master and the slave(s). The ACL-C logical link shall use DM1 packets. The ACL-C logical link is indicated by the LLID code 11 in the payload header.

8.7.2 Sniff Subrating

When sniff subrating is enabled by the Link Manager a device alternates between sniff mode and sniff subrating mode. Sniff subrating mode allows a device to use a reduced number of sniff anchor points. A device shall transition from sniff mode to sniff subrating mode based on the sniff mode timeout value (see Section 8.7.2.1 on page 182). A device shall transition from sniff subrating mode to sniff mode whenever ACL-U or ACL-C data is received from the remote device.

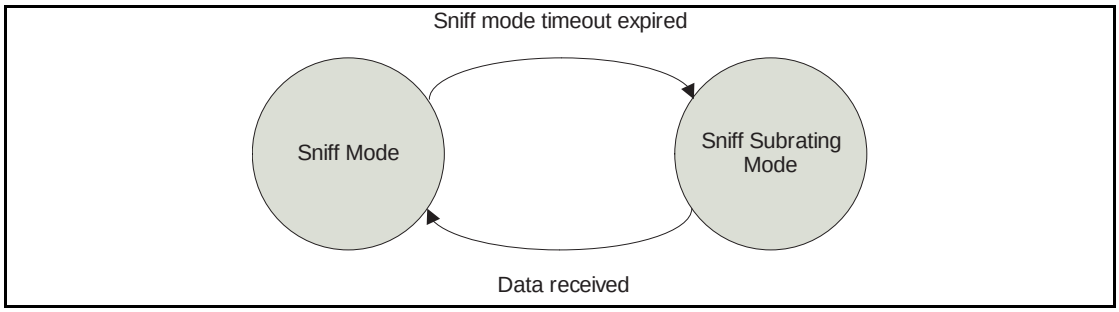


Figure 8.13: Sniff subrating

A slave device in sniff subrating mode shall also temporarily enter sniff mode after transmitting a packet requiring acknowledgement until the baseband acknowledgement is received.

Note: Once sniff subrating is enabled the master and slave devices may be in sniff mode or sniff subrating mode at different times. The rules defined in Section 8.7.2.2 on page 183 describe how the two devices maintain synchronization and can reliably exchange data.

8.7.2.1 Sniff Mode Timeout

Whenever a packet containing ACL-C or ACL-U data is received by a device in sniff subrating mode it shall transition to sniff mode, re-start the sniff mode timeout and all sniff anchor points shall be used until a minimum number of slots equal to the received min_sniff_mode_timeout have expired. If ACL-C or ACL-U data is received before min_sniff_mode_timeout slots have passed since the last ACL-C or ACL-U data was received, the timeout shall be

restarted. NULL and POLL packets do not contain ACL data and shall not restart the sniff mode timeout.

Note that there are two sniff mode timeout values: one for the local device and one for the remote device.

8.7.2.2 Sniff Subrating Mode

When the sniff mode timeout has expired a device shall enter sniff subrating mode. In sniff subrating mode the mandatory sniff subrate anchor points at which the master shall transmit to the slave and the slave shall listen for the master, are defined as follows (where M is the max subrate received by the master, N is the max subrate received by the slave, A is the sniff subrating instant, and k is any positive integer):

	Master	**Slave**
M=N	$CLK_{M(k)} = A + [T_{sniff} \bullet M \bullet k]$	$CLK_{N(k)} = A + [T_{sniff} \bullet N \bullet k]$
M>N	At least once every j anchor points satisfying $CLK_{N(k)} = A + [T_{sniff} \bullet N \bullet k]$, where $j = \lfloor M / N \rfloor$	$CLK_{N(k)} = A + [T_{sniff} \bullet N \bullet k]$
M<N	$CLK_{M(k)} = A + [T_{sniff} \bullet M \bullet k]$	At least once every j anchor points satisfying $CLK_{M(k)} = A + [T_{sniff} \bullet M \bullet k]$, where $j = \lfloor N / M \rfloor$

Table 8.7: Sniff subrate anchor points

1 Note: $\lfloor X \rfloor$ denotes the mathematical “floor” function, equivalent to rounding down to the nearest integer. Hence, $\lfloor X / Y \rfloor$ yields the highest integer j for which $Y \bullet j \leq X$.

2 Note: When sniff subrating is enabled, the rules specified in Section 8.7 for $N_{sniff\ attempt}$ and $N_{sniff\ Timeout}$ shall apply to sniff subrating anchor points.

8.8 HOLD MODE

During the **CONNECTION** state, the ACL logical transport to a slave can be put in a **hold** mode. In **hold** mode the slave temporarily shall not support ACL packets on the channel. Any synchronous packet during reserved synchronous slots (from SCO and eSCO links) shall be supported. With the **hold** mode, capacity can be made free to do other things like scanning, paging, inquiring, or attending another piconet. The device in **hold** mode can also enter a low-power sleep mode. During **hold** mode, the slave device keeps its logical transport address(es) (LT_ADDR).

Prior to entering hold mode, master and slave agree on the time duration the slave remains in hold mode. A timer shall be initialized with the *holdTO* value.

When the timer is expired, the slave shall wake up, synchronize to the traffic on the channel and will wait for further master transmissions.

8.9 PARK STATE

When a slave does not need to participate on the piconet channel, but still needs to remain synchronized to the channel, it can enter **PARK** state. **PARK** state is a state with very little activity in the slave. In the **PARK** state, the slave shall give up its logical transport address LT_ADDR. Instead, it shall receive two new addresses to be used in the **PARK** state

- PM_ADDR: 8-bit Parked Member Address
- AR_ADDR: 8-bit Access Request Address

The PM_ADDR distinguishes a parked slave from the other parked slaves. This address may be used in the master-initiated unpark procedure. In addition to the PM_ADDR, a parked slave may also be unparked by its 48-bit BD_ADDR. The all-zero PM_ADDR is a reserved address: if a parked device has the all-zero PM_ADDR it can only be unparked by the BD_ADDR. In that case, the PM_ADDR has no meaning. The AR_ADDR shall be used by the slave in the slave-initiated unpark procedure. All messages sent to the parked slaves are carried by broadcast packets.

The parked slave wakes up at regular intervals to listen to the channel in order to re-synchronize and to check for broadcast messages. To support the synchronization and channel access of the parked slaves, the master supports a beacon train described in the next section. The beacon structure is communicated to the slave when it is parked. When the beacon structure changes, the parked slaves are updated through broadcast messages.

The master shall maintain separate non-overlapping park beacon structures for each hop sequence. The beacon structures shall not overlap either their beacon slots or access windows.

In addition for using it for low power consumption, park is used to connect more than seven slaves to a single master. At any one time, only seven slaves can be in the **CONNECTION** state. However, by swapping active and parked slaves out respectively in the piconet, the number of slaves can be much larger (255 if the PM_ADDR is used, and an arbitrarily large number if the BD_ADDR is used).

8.9.1 Beacon Train

To support parked slaves, the master establishes a beacon train when one or more slaves are parked. The beacon train consists of one beacon slot or a train of equidistant beacon slots which is transmitted periodically with a constant time interval. The beacon train is illustrated in Figure 8.14 on page 186. A train of N_B ($N_B \geq 1$) beacon slots is defined with an interval of T_B slots. The beacon slots in the train are separated by Δ_B. The start of the first beacon slot is referred to as the **beacon instant** and serves as the beacon timing reference. The beacon parameters N_B and T_B are chosen such that there are sufficient

beacon slots for a parked slave to synchronize to during a certain time window in an error-prone environment.

When parked, the slave shall receive the beacon parameters through an LMP command. In addition, the timing of the beacon instant is indicated through the offset D_B. As with the SCO logical transport (see Section 8.6.2 on page 167), two initialization procedures 1 or 2 are used. The master shall use initialization 1 when the MSB of the current master clock (CLK_{27}) is 0; it shall use initialization 2 when the MSB of the current master clock (CLK_{27}) is 1. The chosen initialization procedure shall also be carried by an initialization flag in the LMP command. The slave shall apply the initialization method as indicated by the initialization flag irrespective of its clock bit CLK_{27}. The master-to-slave slot positioned at the beacon instant shall be initialized on the slots for which the clock satisfies the applicable equation:

$$CLK_{27-1} \bmod T_B = D_B \qquad \text{for initialization 1}$$

$$(\overline{CLK}_{27}, CLK_{26-1}) \bmod T_B = D_B \qquad \text{for initialization 2}$$

this implies that D_B will be even.

After initialization, the clock value CLK(k+1) for the next beacon instant shall be derived by adding the fixed interval T_B to the clock value of the current beacon instant:

$$CLK(k+1) = CLK(k) + T_B$$

The beacon train serves four purposes:

1. Transmission of master-to-slave packets which the parked slaves can use for re-synchronization
2. Carrying messages to the parked slaves to change the beacon parameters
3. Carrying general broadcast messages to the parked slaves
4. Unparking of one or more parked slaves

Since a slave can synchronize to any packet which is preceded by the proper channel access code, the packets carried on the beacon slots do not have to contain specific broadcast packets for parked slaves to be able to synchronize; any packet may be used. The only requirement placed on the beacon slots is that there is a master-to-slave transmission present on the hopping sequence associated with the park structure. If there is no information to be sent, **NULL** packets may be transmitted by the master. If there is indeed broadcast information to be sent to the parked slaves, the first packet of the broadcast message shall be repeated in every beacon slot of the beacon train. However, synchronous traffic in the synchronous reserved slots may interrupt the beacon transmission if it is on the same hopping sequence as the parked slaves. The master shall configure its park beacon structure so that reserved slots of synchronous logical transports do not cause slaves to miss synchronization on a beacon slot. For example, a master that has active slaves using AHS, and parked slaves using Non-AHS shall ensure that the Park beacons cannot be interrupted by AHS synchronous reserved slots.

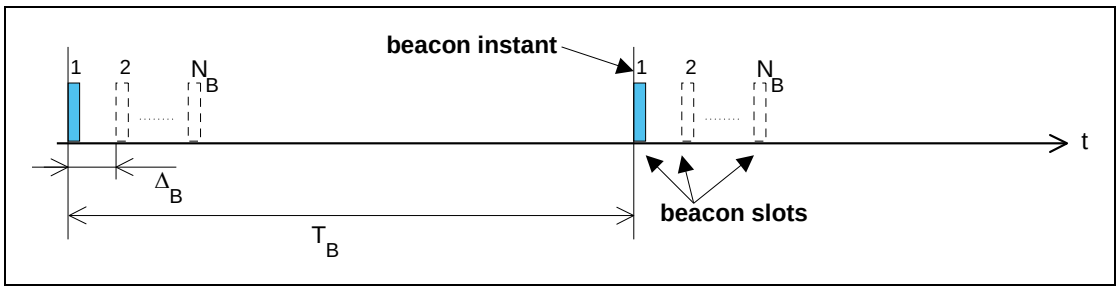


Figure 8.14: General beacon train format

The master can place parked slaves in any of the AFH operating modes, but shall ensure that all parked slaves use the same hop sequence. Masters should use AHS(79) or AHS when all the slaves in the Piconet are AFH capable.

A master that switches a slave between AFH enabled, AFH disabled or AHS(79) operation shall ensure that the AFH_Instant occurs before transmission of the beacon train using this hop sequence.

The master communicates with parked slaves using broadcast messages. Since these messages can be time-critical, an ongoing repetition train of broadcast message may be prematurely aborted by broadcast information destined to parked slaves in beacon slots and in access windows (see Section 8.9.2).

8.9.2 Beacon Access Window

In addition to the beacon slots, an access window is defined where the parked slaves can send requests to be unparked. To increase reliability, the access window may be repeated M_{access} times ($M_{access} \geq 1$), see Figure 8.15 on page 187. The access window starts a fixed delay D_{access} after the beacon instant. The width of the access window is T_{access}.

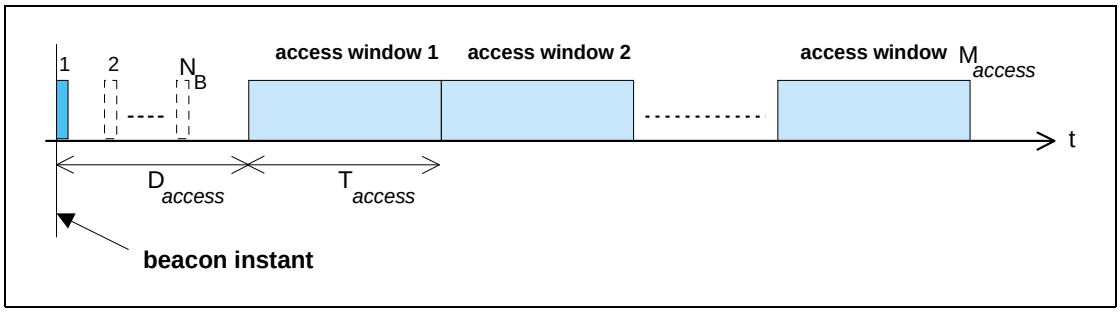


Figure 8.15: Definition of access window

The access window supports a polling slave access technique. The format of the polling technique is shown in Figure 8.16 on page 187. The same TDD structure is used as on the piconet channel, i.e. master-to-slave transmission is alternated with slave-to-master transmission. The slave-to-master slot is divided into two half slots of 312.5 µs each. The half slot a parked slave is allowed to respond in corresponds to its access request address (AR_ADDR), see also Section 8.9.6 on page 190. For counting the half slots to determine the access request slot, the start of the access window is used, see Figure 8.16 on page 187. The slave shall only send an access request in the proper slave-to-master half slot if a broadcast packet has been received in the preceding master-to-slave slot. In this way, the master polls the parked slaves.

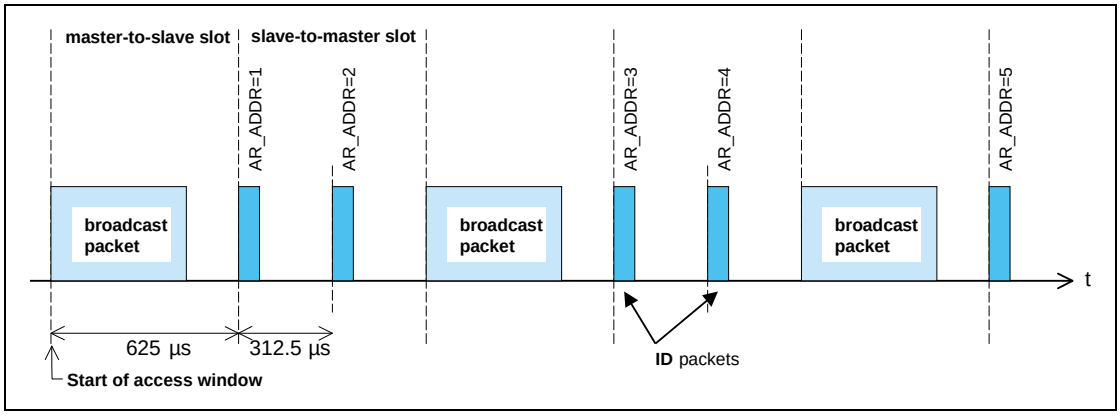


Figure 8.16: Access procedure applying the polling technique.

The slots of the access window may also be used for traffic on the piconet if required. For example, if a synchronous connection has to be supported, the slots reserved for the synchronous link may carry synchronous information

instead of being used for access requests, i.e. if the master-to-slave slot in the access window contains a packet different from a broadcast packet, the following slave-to-master slot shall not be used for slave access requests. If the master transmits a broadcast packet in the access window then it shall use the hop sequence associated with the park structure. Slots in the access window not affected by piconet traffic may still be used according to the defined access structure (an example is shown in Figure 8.17 on page 188). The access procedure shall be continued as if no interruption had taken place.

When the slave is parked, the master shall indicate what type of access scheme will be used. For the polling scheme, the number of slave-to-master access slots N_{acc_slot} is indicated.

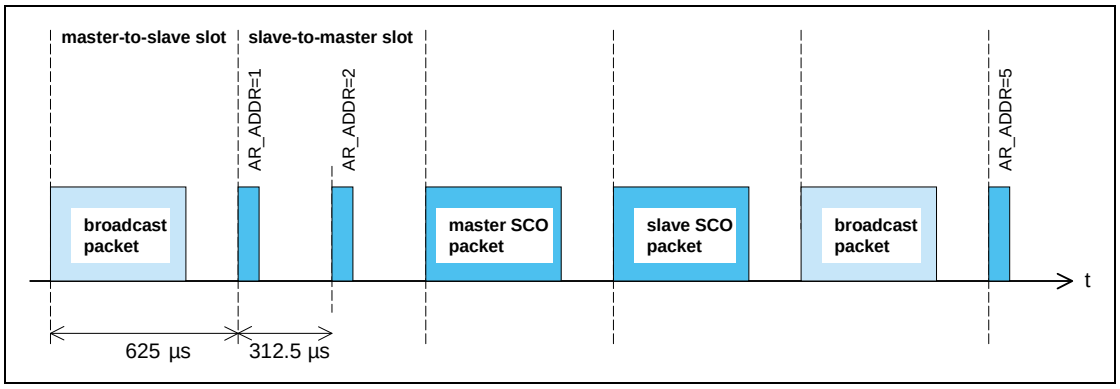


Figure 8.17: Disturbance of access window by SCO traffic

By default, the access window is always present. However, its activation depends on the master sending broadcast messages to the slave at the appropriate slots in the access window. A flag in a broadcast LMP message within the beacon slots may indicate that the access window(s) belonging to this instant will not be activated. This prevents unnecessary scanning of parked slaves that want to request access.

8.9.3 Parked Slave Synchronization

Parked slaves wake up periodically to re-synchronize to the channel. Any packet exchanged on the channel can be used for synchronization. Since master transmission is mandatory on the beacon slots, parked slaves will use the beacon train to re-synchronize. A parked slave may wake-up at the beacon instant to read the packet sent on the first beacon slot. If this fails, it may retry on the next beacon slot in the beacon train; in total, there are N_B opportunities per beacon instant to re-synchronize. During the search, the slave may increase its search window, see also Section 2.2.5.2 on page 75. The separation between the beacon slots in the beacon train Δ_B shall be chosen such that consecutive search windows will not overlap.

The parked slave may not wake up at every beacon instant. Instead, a sleep interval may be applied which is longer than the beacon interval T_B, see Figure 8.18 on page 189. The slave sleep window shall be a multiple N_{B_sleep} of T_B.

The precise beacon instant the slave may wake up on shall be indicated by the master with D_{B_sleep} which indicates the offset (in multiples of T_B) with respect to the beacon instant ($0 < D_{B\_sleep} < N_{B\_sleep}-1$). To initialize the wake-up period, the applicable equation shall be used:

$CLK_{27-1} \bmod (N_{B\_sleep} \bullet T_B) = D_B + D_{B\_sleep} \bullet T_B$ for initialization 1

$(\overline{CLK_{27}}, CLK_{26-1}) \bmod (N_{B\_sleep} \bullet T_B) = D_B + D_{B\_sleep} \bullet T_B$ for initialization 2

where initialization 1 shall be chosen by the master if the MSB in the current master clock is 0 and initialization 2 shall be chosen by the master if the MSB in the current master clock is 1.

When the master needs to send broadcast messages to the parked slaves, it may use the beacon slots for these broadcast messages. However, if $N_B < N_{BC}$, the slots following the last beacon slot in the beacon train shall be used for the remaining $N_{BC}-N_B$ broadcast packets. If $N_B > N_{BC}$, the broadcast message shall be repeated on all N_B beacon slots.

A parked slave shall read the broadcast messages sent in the beacon slot(s) it wakes up in. If the parked slave wakes up, the minimum wake-up activity shall be to read the channel access code for re-synchronization and the packet header to check for broadcast messages.

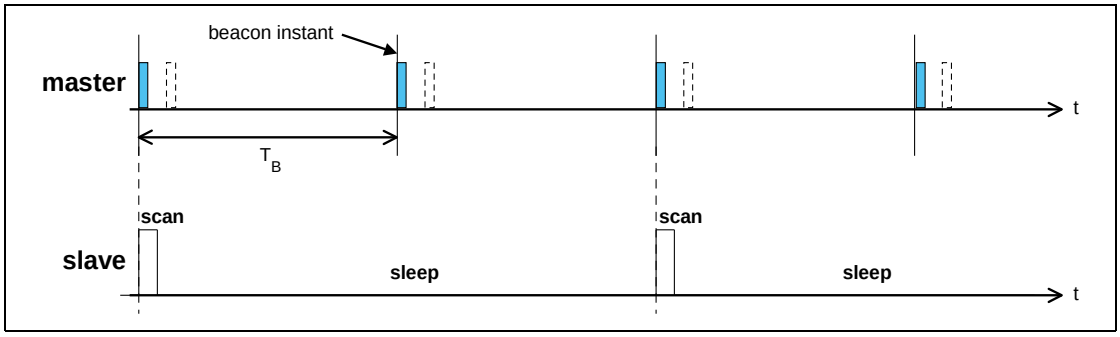


Figure 8.18: Extended sleep interval of parked slaves.

8.9.4 Parking

A master can park an active slave through the exchange of LMP commands. Before being put into park, the slave shall be assigned a PM_ADDR and an AR_ADDR. Every parked slave shall have a unique PM_ADDR or a PM_ADDR of 0. The AR_ADDR is not necessarily unique. The beacon parameters shall be given by the master when the slave is parked. The slave shall then give up its LT_ADDR and shall enter **PARK** state. A master can park only a single slave at a time. The park message is carried with a normal data packet and addresses the slave through its LT_ADDR.

8.9.5 Master-initiated Unparking

The master can unpark a parked slave by sending a dedicated LMP unpark command including the parked slave's address. This message shall be sent in a broadcast packet on the beacon slots. The master shall use either the slave's PM_ADDR, or its BD_ADDR. The message also includes the logical transport address LT_ADDR the slave shall use after it has re-entered the piconet. The unpark message may include a number of slave addresses so that multiple slaves may be unparked simultaneously. For each slave, a different LT_ADDR shall be assigned.

After having received the unpark message, the parked slave matching the PM_ADDR or BD_ADDR shall leave the **PARK** state and enter the **CONNECTION** state. It shall keep listening to the master until it is addressed by the master through its LT_ADDR. The first packet sent by the master shall be a POLL packet. The return packet in response to the POLL packet confirms that the slave has been unparked. If no response packet from the slave is received for *newconnectionTO* number of slots after the end of beacon repetition period, the master shall unpark the slave again. The master shall use the same LT_ADDR on each unpark attempt until it has received a link supervision timeout for that slave or the unpark has completed successfully. If the slave does not receive the POLL packet for *newconnectionTO* number of slots after the end of beacon repetition period, it shall return to park, with the same beacon parameters. After confirming that the slave is in the **CONNECTION** state, the master decides in which mode the slave will continue.

When a device is unparked, the SEQN bit for the link shall be reset to 1 on both the master and the slave (see Section 7.6.2.1 on page 145).

8.9.6 Slave-initiated Unparking

A slave can request access to the channel through the access window defined in Section 8.9.2 on page 187. As shown in Figure 8.16 on page 187, the access window includes several slave-to-master half slots where the slave may send an access request message. The specific half slot the slave is allowed to respond in, corresponds to its access request address (AR_ADDR) which it received when it was parked. The order of the half slots (in Figure 8.16 the AR_ADDR numbers linearly increase from 1 to 5) is not fixed: an LMP command sent in the beacon slots may reconfigure the access window. When a slave desires access to the channel, it shall send an access request message in the proper slave-to-master half slot. The access request message of the slave is the **ID** packet containing the device access code (DAC) of the master (which is the channel access code without the trailer). The parked slave shall only transmit an access request message in the half slot, when in the preceding master-to-slave slot a broadcast packet has been received. This broadcast message may contain any kind of broadcast information not necessarily related to the parked slave(s). If no broadcast information is available, a broadcast **NULL** or broadcast **POLL** packet shall be sent.

After having sent an access request, the parked slave shall listen for an unpark message from the master. As long as no unpark message is received, the slave shall repeat the access requests in the subsequent access windows. After the last access window (there are M_{access} windows in total, see Section 8.9.2 on page 187), the parked slave shall listen for an additional N_{poll} time slots for an unpark message. If no unpark message is received within N_{poll} slots after the end of the last access window, the slave may return to sleep and retry an access attempt after the next beacon instant.

After having received the unpark message, the parked slave matching the PM_ADDR or BD_ADDR shall leave the **PARK** state and enter the **CONNECTION** state. It shall keep listening to the master until it is addressed by the master through its LT_ADDR. The first packet sent by the master shall be a POLL packet. The return packet in response to the POLL packet confirms that the slave has been unparked. After confirming that the slave is in the **CONNECTION** state, the master decides in which mode the slave will continue. If no response packet from the slave is received for *newconnectionTO* number of slots after N_{poll} slots after the end of the last access window, the master shall send the unpark message to the slave again. If the slave does not receive the POLL packet for *newconnectionTO* number of slots after N_{poll} slots after the end of the last access window, it shall return to park, with the same beacon parameters.

When a device is unparked, the SEQN bit for the link shall be reset to 1 on both the master and the slave (see Section 7.6.2.1 on page 145).

8.9.7 Broadcast Scan Window

In the beacon train, the master can support broadcast messages to the parked slaves. However, it may extend its broadcast capacity by indicating to the parked slaves that more broadcast information is following after the beacon train. This is achieved by an LMP command ordering the parked slaves (as well as the active slaves) to listen to the channel for broadcast messages during a limited time window. This time window starts at the beacon instant and continues for the period indicated in the LMP command sent in the beacon train.

8.9.8 Polling in the Park State

In the **PARK** state, parked slaves may send access requests in the access window provided a broadcast packet is received in the preceding master-to-slave slot. Slaves in the **CONNECTION** state shall not send in the slave-to-master slots following the broadcast packet, since they are only allowed to send if addressed specifically.

9 AUDIO

On the air-interface, either a 64 kb/s log PCM (Pulse Code Modulation) format (A-law or μ-law) may be used, or a 64 kb/s CVSD (Continuous Variable Slope Delta Modulation) may be used. The latter format applies an adaptive delta modulation algorithm with syllabic companding.

The voice coding on the line interface is designed to have a quality equal to or better than the quality of 64 kb/s log PCM.

Table 9.1 on page 192 summarizes the voice coding schemes supported on the air interface. The appropriate voice coding scheme is selected after negotiations between the Link Managers.

Voice Codecs	
linear	CVSD
8-bit logarithmic	A-law
	μ-law

Table 9.1: Voice coding schemes supported on the air interface.

9.1 LOG PCM CODEC

Since the synchronous logical transports on the air-interface can support a 64 kb/s information stream, a 64 kb/s log PCM traffic can be used for transmission. Either A-law or μ-law compression may be applied. In the event that the line interface uses A-law and the air interface uses μ-law or vice versa, a conversion from A-law to μ-law shall be performed. The compression method shall follow ITU-T recommendations G. 711.

9.2 CVSD CODEC

A more robust format for voice over the air interface is delta modulation. This modulation scheme follows the waveform where the output bits indicate whether the prediction value is smaller or larger than the input waveform. To reduce slope overload effects, syllabic companding is applied: the step size is adapted according to the average signal slope. The input to the CVSD encoder shall be 64 ksamples/s linear PCM (typically 16 bits, but actual value is implementation specific). Block diagrams of the CVSD encoder and CVSD decoder are shown in Figure 9.1 on page 193, Figure 9.2 on page 193 and Figure 9.3 on page 193. The system shall be clocked at 64 kHz.

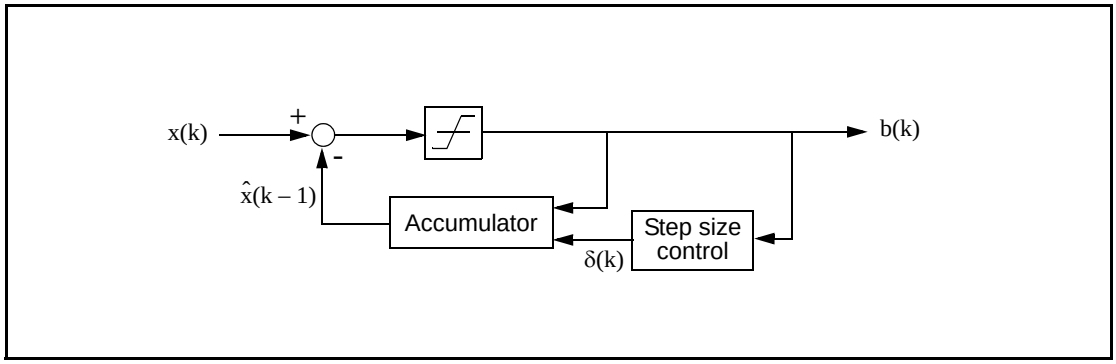


Figure 9.1: Block diagram of CVSD encoder with syllabic companding.

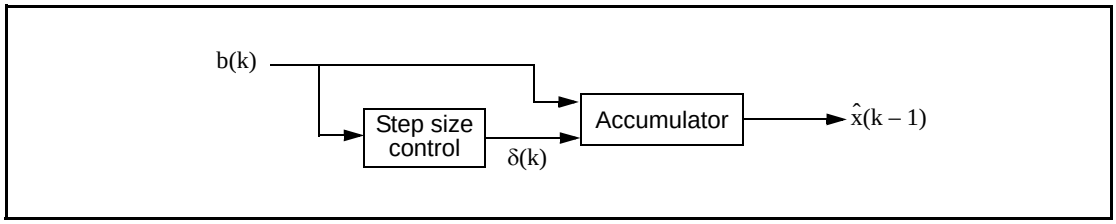


Figure 9.2: Block diagram of CVSD decoder with syllabic companding.

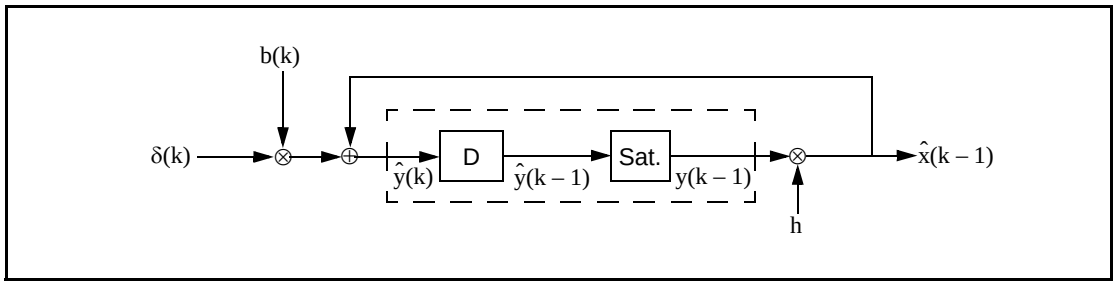


Figure 9.3: Accumulator procedure.

Let $\mathrm{sgn}(x) = 1$ for $x \geq 0$, otherwise $\mathrm{sgn}(x) = -1$. On air these numbers shall be represented by the sign bit; i.e. negative numbers are mapped on “1” and positive numbers are mapped on “0”.

Denote the CVSD encoder output bit $b(k)$, the encoder input $x(k)$, the accumulator contents $y(k)$, and the step size $\delta(k)$. Furthermore, let *h* be the decay factor for the accumulator, let β denote the decay factor for the step size, and, let α be the syllabic companding parameter. The latter parameter monitors the slope by considering the *K* most recent output bits

Let

$$\hat{x}(k) = hy(k). \quad \text{(EQ 13)}$$

Then, the CVSD encoder internal state shall be updated according to the following set of equations:

$$b(k) = \mathrm{sgn}\{x(k) - \hat{x}(k-1)\}, \quad \text{(EQ 14)}$$

$$\alpha = \begin{cases} 1, & \text{if J bits in the last K output bits are equal,} \\ 0, & \text{otherwise,} \end{cases} \quad \text{(EQ 15)}$$

$$\delta(k) = \begin{cases} \min\{\delta(k-1) + \delta_{min}, \delta_{max}\}, & \alpha = 1, \\ \max\{\beta\delta(k-1), \delta_{min}\}, & \alpha = 0, \end{cases} \quad \text{(EQ 16)}$$

$$y(k) = \begin{cases} \min\{\hat{y}(k), y_{max}\}, & \hat{y}(k) \geq 0. \\ \max\{\hat{y}(k), y_{min}\}, & \hat{y}(k) < 0. \end{cases} \quad \text{(EQ 17)}$$

where

$$\hat{y}(k) = \hat{x}(k-1) + b(k)\delta(k). \quad \text{(EQ 18)}$$

In these equations, δ_{min} and δ_{max} are the minimum and maximum step sizes, and, y_{min} and y_{max} are the accumulator's negative and positive saturation values, respectively. Over air, the bits shall be sent in the same order they are generated by the CVSD encoder.

For a 64 kb/s CVSD, the parameters as shown in Table 9.2 shall be used. The numbers are based on a 16 bit signed number output from the accumulator. These values result in a time constant of 0.5 ms for the accumulator decay, and a time constant of 16 ms for the step size decay

Parameter	Value
h	$1 - \frac{1}{32}$
β	$1 - \frac{1}{1024}$
J	4
K	4
δ_{min}	10
δ_{max}	1280
y_{min}	-2^{15} or $-2^{15} + 1$
y_{max}	$2^{15} - 1$

Table 9.2: CVSD parameter values. The values are based on a 16 bit signed number output from the accumulator.

9.3 ERROR HANDLING

In the DV, HV3, EV3, EV5, 2-EV3, 3-EV3, 2-EV5 and 3-EV5 packets, the voice is not protected by FEC. The quality of the voice in an error-prone environment then depends on the robustness of the voice coding scheme and, in the case of eSCO, the retransmission scheme. CVSD, in particular, is rather insensitive to random bit errors, which are experienced as white background noise. However, when a packet is rejected because either the channel access code, the HEC test was unsuccessful, or the CRC has failed, measures have to be taken to "fill" in the lost speech segment.

The voice payload in the **HV2** and **EV4** packets are protected by a 2/3 rate FEC. For errors that are detected but cannot be corrected, the receiver should try to minimize the audible effects. For instance, from the 15-bit FEC segment with uncorrected errors, the 10-bit information part as found before the FEC decoder should be used. The **HV1** packet is protected by a 3 bit repetition FEC. For this code, the decoding scheme will always assume zero or one-bit errors. Thus, there exist no detectable but uncorrectable error events for **HV1** packets.

9.4 GENERAL AUDIO REQUIREMENTS

9.4.1 Signal Levels

For A-law and μ-law log-PCM encoded signals the requirements on signal levels shall follow the ITU-T recommendation G.711.

Full swing at the 16 bit linear PCM interface to the CVSD encoder is defined to be 3 dBm0.

9.4.2 CVSD Audio Quality

For Bluetooth audio quality the requirements are put on the transmitter side. The 64 ksamples/s linear PCM input signal should have negligible spectral power density above 4 kHz. The power spectral density in the 4-32 kHz band of the decoded signal at the 64 ksamples/s linear PCM output, should be more than 20 dB below the maximum in the 0-4 kHz range.

10 LIST OF FIGURES

Figure 1.1: Piconets with a single slave operation (a), a multi-slave operation (b) and a scatternet operation (c). 65
Figure 1.2: Standard Basic Rate packet format. 66
Figure 1.3: Standard Enhanced Data Rate packet format 66
Figure 1.4: Bluetooth clock. 66
Figure 1.5: Format of BD_ADDR. 68
Figure 2.1: Multi-slot packets 72
Figure 2.2: Derivation of CLK in master (a) and in slave (b). 73
Figure 2.3: RX/TX cycle of master transceiver in normal mode for single-slot packets. 74
Figure 2.4: RX/TX cycle of slave transceiver in normal mode for single-slot packets. 75
Figure 2.5: RX timing of slave returning from hold mode. 76
Figure 2.6: Derivation of CLKE. 77
Figure 2.7: RX/TX cycle of transceiver in PAGE mode. 78
Figure 2.8: Timing of page response packets on successful page in first half slot 79
Figure 2.9: Timing of page response packets on successful page in second half slot 80
Figure 2.10: Timing of inquiry response packets on successful inquiry in first half slot 82
Figure 2.11: Timing of inquiry response packets on successful inquiry in second half slot 82
Figure 2.12: General block diagram of hop selection scheme. 84
Figure 2.13: Hop selection scheme in CONNECTION state. 85
Figure 2.14: Single- and multi-slot packets. 86
Figure 2.15: Example of the same channel mechanism. 86
Figure 2.16: Block diagram of the basic hop selection kernel for the hop system. 87
Figure 2.17: XOR operation for the hop system. 88
Figure 2.18: Permutation operation for the hop system. 89
Figure 2.19: Butterfly implementation. 89
Figure 2.20: Block diagram of adaptive hop selection mechanism 91
Figure 4.1: Functional diagram of TX buffering. 99
Figure 4.2: Functional diagram of RX buffering 103
Figure 6.1: General Basic Rate packet format. 108
Figure 6.2: General enhanced data rate packet format 108
Figure 6.3: Access code format 110
Figure 6.4: Preamble 111
Figure 6.5: Construction of the sync word. 112
Figure 6.6: LFSR and the starting state to generate 114

Figure 6.7: Trailer in CAC when MSB of sync word is 0 (a), and when MSB of sync word is 1 (b). 114
Figure 6.8: Header format. 115
Figure 6.9: Format of the FHS payload. 119
Figure 6.10: DV packet format 123
Figure 6.11: **Synchronization sequence 128**
Figure 6.12: Payload header format for Basic Rate single-slot ACL packets. 129
Figure 6.13: Payload header format for multi-slot ACL packets and all EDR ACL packets. 129
Figure 7.1: Header bit processes. 134
Figure 7.2: Payload bit processes. 134
Figure 7.3: The LFSR circuit generating the HEC. 135
Figure 7.4: Initial state of the HEC generating circuit. 136
Figure 7.5: HEC generation and checking. 136
Figure 7.6: The LFSR circuit generating the CRC. 137
Figure 7.7: Initial state of the CRC generating circuit. 137
Figure 7.8: CRC generation and checking. 137
Figure 7.9: Data whitening LFSR. 138
Figure 7.10: Bit-repetition encoding scheme. 139
Figure 7.11: LFSR generating the (15,10) shortened Hamming code. 140
Figure 7.12: Stage 1 of the receive protocol for determining the ARQN bit. 142
Figure 7.13: Stage 2 (ACL) of the receive protocol for determining the ARQN bit. 143
Figure 7.14: Stage 2 (eSCO) of the receive protocol for determining the ARQN bit. 144
Figure 7.15: Transmit filtering for packets with CRC. 145
Figure 7.16: Broadcast repetition scheme 149
Figure 8.1: State diagram of link controller. 150
Figure 8.2: Conventional page (a), page while one synchronous link present (b), page while two synchronous links present (c). 155
Figure 8.3: Messaging at initial connection when slave responds to first page message. 157
Figure 8.4: Messaging at initial connection when slave responds to second page message. 157
Figure 8.5: Connection state. 166
Figure 8.6: RX/TX timing in multi-slave configuration 167
Figure 8.7: eSCO Window Details for Single-Slot Packets 170
Figure 8.8: eSCO Window Details for Asymmetric Traffic 171
Figure 8.9: Successful hop sequence switching 175
Figure 8.10: Recovery hop sequences 176
Figure 8.11: AFH_Instant changes during multi-slot packets transmitted by the master. 177

Figure 8.12: Sniff anchor points 180
Figure 8.13: Sniff subrating 182
Figure 8.14: General beacon train format 186
Figure 8.15: Definition of access window 187
Figure 8.16: Access procedure applying the polling technique. 187
Figure 8.17: Disturbance of access window by SCO traffic 188
Figure 8.18: Extended sleep interval of parked slaves. 189
Figure 9.1: Block diagram of CVSD encoder with syllabic companding. .. 193
Figure 9.2: Block diagram of CVSD decoder with syllabic companding. .. 193
Figure 9.3: Accumulator procedure. 193
Figure 11.1: SLR measurement set-up. 201
Figure 11.2: RLR measurement set-up. 201
Figure 11.3: Plot of recommended frequency mask for Bluetooth. The GSM send frequency mask is given for comparison (dotted line) 202
Figure 11.4: Timing constraint on AFH_Instant with slaves in park, hold and sniff 206

11 LIST OF TABLES

Table 2.1: Control of the butterflies for the hop system 88
Table 2.2: Control for hop system. 92
Table 6.1: Summary of access code types. 110
Table 6.2: Packets defined for synchronous and asynchronous logical transport types. 118
Table 6.3: Description of the FHS payload 120
Table 6.4: Contents of SR field 121
Table 6.5: Contents of page scan mode field....121
Table 6.6: Logical link LLID field contents....130
Table 6.7: Use of payload header flow bit on the logical links. 131
Table 6.8: Link control packets 132
Table 6.9: ACL packets....132
Table 6.10: Synchronous packets....133
Table 8.1: Relationship between scan interval, and paging modes R0, R1, and R2. 152
Table 8.2: Relationship between train repetition, and paging modes R0, R1 and R2 when synchronous links are present. 155
Table 8.3: Initial messaging during start-up. 156
Table 8.4: Increase of train repetition when synchronous links are present. 163
Table 8.5: Messaging during inquiry routines. 165
Table 8.6: Channel classification descriptions 178
Table 8.7: Sniff subrate anchor points 183
Table 9.1: Voice coding schemes supported on the air interface....192
Table 9.2: CVSD parameter values. The values are based on a 16 bit signed number output from the accumulator. 194
Table 11.1: Recommended Frequency Mask for Bluetooth....202

APPENDIX A: GENERAL AUDIO RECOMMENDATIONS

MAXIMUM SOUND PRESSURE

It is the sole responsibility of each manufacturer to design their audio products in a safe way with regards to injury to the human ear. The Bluetooth Specification doesn't specify maximum sound pressure from an audio device.

OTHER TELEPHONY NETWORK REQUIREMENTS

It is the sole responsibility of each manufacturer to design the Bluetooth audio product so that it meets the regulatory requirements of all telephony networks that it may be connected to.

AUDIO LEVELS FOR BLUETOOTH

Audio levels shall be calculated as Send Loudness Rating, SLR, and Receive Loudness Rating, RLR. The calculation methods are specified in ITU-T Recommendation P.79.

The physical test set-up for Handsets and Headsets is described in ITU-T Recommendation P.51 and P.57

The physical test set-up for speakerphones and "Vehicle hands-free systems" is specified in ITU-T Recommendation P.34.

A general equation for computation of loudness rating (LR) for telephone sets is given by ITU-T recommendations P.79 and is given by

$$LR = -\frac{10}{m}\log_{10}\left(\sum_{i=N_1}^{N_2} 10^{m(s_i - w_i)/10}\right), \qquad \text{(EQ 19)}$$

where

m is a constant (~ 0.2).

w_i = weighting coefficient (different for the various LRs).

S_i = the sensitivity at frequency F_i, of the electro-acoustic path

N_1,N_2, consecutive filter bank numbers (Art. Index: 200-4000 Hz)

(EQ 19) on page 200 is used for calculating the (SLR) according to Figure 11.1 on page 201, and (RLR) according to Figure 11.2 on page 201. When calculating LRs one must only include those parts of the frequency band where an actual signal transmission can occur in order to ensure that the additive property of LRs is retained. Therefore ITU-T P.79 uses only the frequency band 200-4000 Hz in LR computations.

MICROPHONE PATH

SLR measurement model

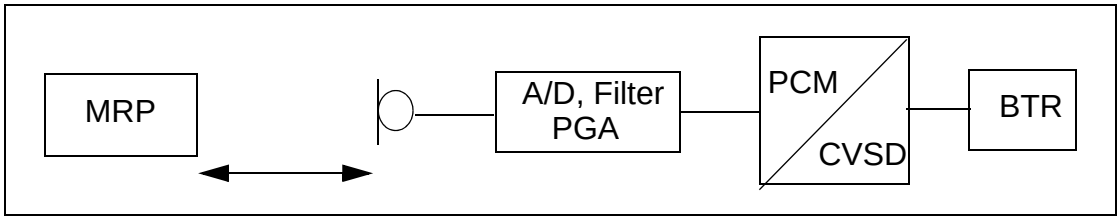


Figure 11.1: SLR measurement set-up.

LOUDSPEAKER PATH

RLR measurement model

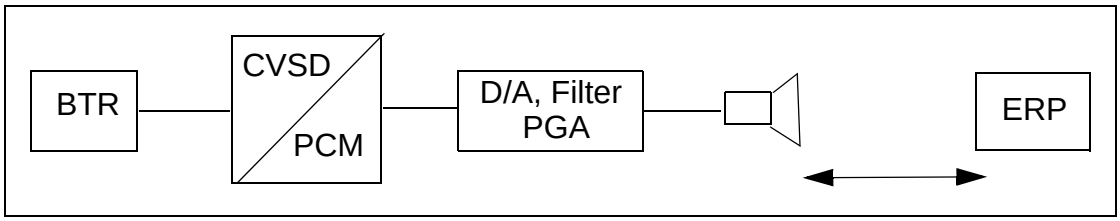


Figure 11.2: RLR measurement set-up.

BLUETOOTH VOICE INTERFACE

The specification for the Bluetooth voice interface should follow in the first place the *ITU-T Recommendations P.79*, which specifies the loudness ratings for telephone sets. These recommendations give general guidelines and specific algorithms used for calculating the loudness ratings of the audio signal with respect to Ear Reference Point (ERP).

For Bluetooth voice interface to the different cellular system terminals, loudness and frequency recommendations based on the cellular standards should be used. For example, GSM 03.50 gives recommendation for both the loudness ratings and frequency mask for a GSM terminal interconnection with Bluetooth.

1- The output of the CVSD decoder are 16-bit linear PCM digital samples, at a sampling frequency of 8 ksample/s. Bluetooth also supports 8-bit log PCM samples of A-law and μ-law type. The sound pressure at the ear reference point for a given 16-bit CVSD sample, should follow the sound pressure level given in the cellular standard specification.

2- A maximum sound pressure which can be represented by a 16-bit linear PCM sample at the output of the CVSD decoder should be specified according to the loudness rating, in ITU P.79 and at PGA value of 0 dB. Programmable Gain Amplifiers (PGAs) are used to control the audio level at the terminals by the user. For conversion between various PCM representations: A-law, μ-law and linear PCM, ITU-T G.711, G.712, G.714 give guidelines and PCM value relationships. Zero-code suppression based on ITU-T G.711 is also recommended to avoid network mismatches.

FREQUENCY MASK

For interfacing a Bluetooth terminal to a digital cellular mobile terminal, a compliance of the CVSD decoder signal to the frequency mask given in the cellular standard, is recommended to guarantee correct function of the speech coders. A recommendation for a frequency mask is given in the Table 11.1 below. The Figure 11.3 below shows a plot of the frequency mask for Bluetooth (solid line). The GSM frequency mask (dotted line) is shown in Figure 11.3 for comparison.

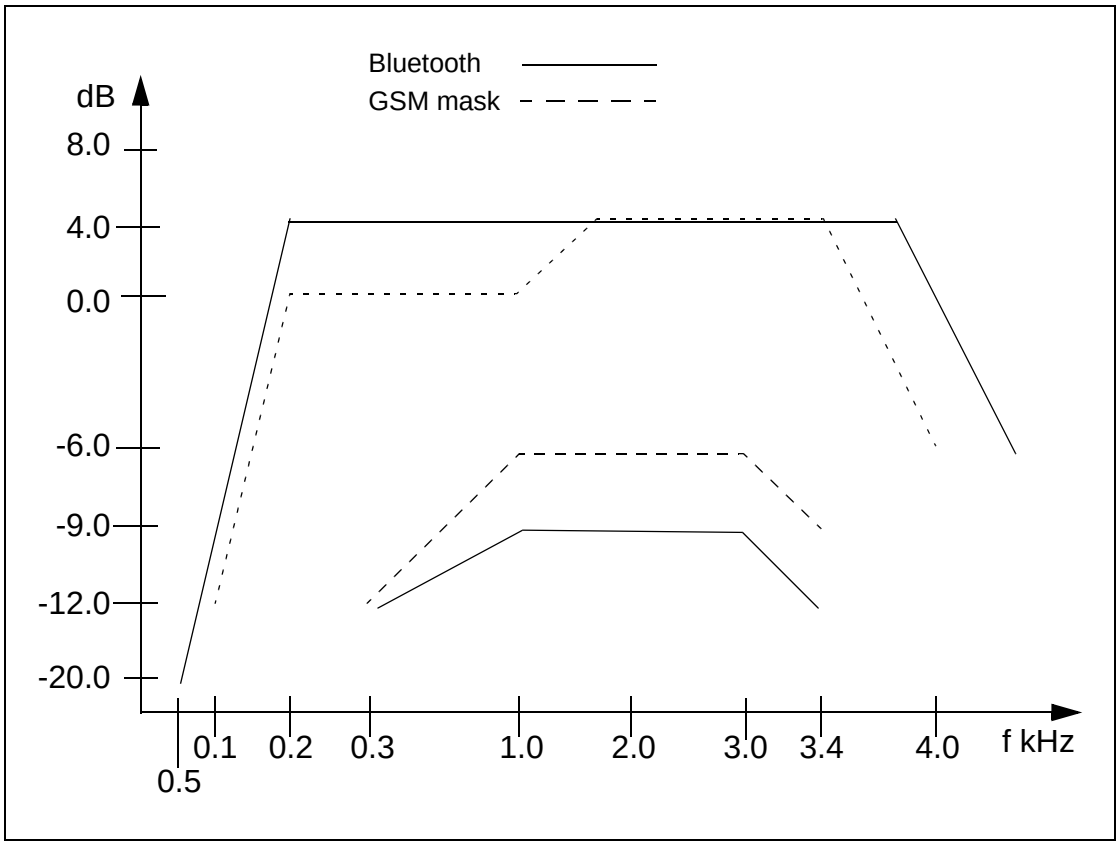


Figure 11.3: Plot of recommended frequency mask for Bluetooth. The GSM send frequency mask is given for comparison (dotted line)

Frequency (Hz)	Upper Limit (dB)	Lower Limit (dB)
50	-20	-
300	4	-12
1000	4	-9
2000	4	-9
3000	4	--9
3400	4	-12
4000	0	-

Table 11.1: Recommended Frequency Mask for Bluetooth

APPENDIX B: TIMERS

This appendix contains a list of Baseband timers related to inactivity timeout defined in this specification. Definitions and default values of the timers are listed below

All timer values are given in slots.

LIST OF TIMERS

inquiryTO

The *inquiryTO* defines the number of slots the **inquiry** substate shall last. The timer value may be changed by the host. HCI provides a command to change the timer value.

pageTO

The *pageTO* defines the number of slots the **page** substate can last before a response is received. The timer value may be changed by the host. HCI provides a command to change the timer value.

pagerespTO

In the slave, it defines the number of slots the slave shall await the master's response, FHS packet, after sending the page acknowledgment ID packet. In the master, *pagerespTO* defines the number of slots the master should wait for the FHS packet acknowledgment before returning to **page** substate. Both master and slave units should use the same value for this timeout, to ensure common page/scan intervals after reaching *pagerespTO*.

The *pagerespTO* value is 8 slots.

newconnectionTO

Every time a new connection is started through paging, scanning, role switch or unparking, the master sends a POLL packet as the first packet in the new connection. Transmission and acknowledgment of this POLL packet is used to confirm the new connection. If the POLL packet is not received by the slave or the response packet is not received by the master for *newconnectionTO* number of slots, both the master and the slave should return to the previous substate.

newconnectionTO value is 32 slots.

supervisionTO

The *supervisionTO* is used by both the master and slave to monitor link loss. If a device does not receive any packets that pass the HEC check and have the proper LT_ADDR for a period of *supervisionTO*, it should reset the link. The supervision timer keeps running in Hold mode, Sniff mode and Park state.

The *supervisionTO* value may be changed by the host. HCI provides a command to change the timer value. At the baseband level a default value equivalent to 20 seconds should be used.

APPENDIX C: RECOMMENDATIONS FOR AFH OPERATION IN PARK, HOLD, AND SNIFF

The three possible AFH operation modes for an AFH capable slave in Park state, Hold mode, and Sniff mode are the same three AFH operation modes used during **CONNECTION** state:

- *Enabled* (using the same AHS as slaves in the **CONNECTION** state)
- *Enabled* (using AHS(79))
- *Disabled*

The master may place an AFH capable slave in any of the three AFH operating modes.

Operation at the Master

A master that has one or more slaves in park, hold, or sniff and decides to update them simultaneously shall schedule an *AFH_Instant* for a time that allows it to update all these slaves (as well as its active slaves) with the new adaptation.

A master that has multiple slaves with non-overlapping “wake” times (e.g. slaves in sniff mode with different timing parameters) may keep them *enabled* on the same adaptation provided that its scheduling of the *AFH_Instant* allows enough time to update them all.

This timing is summarized in the figure below. In this example the master decides that a hop sequence adaptation is required. However it cannot schedule an *AFH_Instant* until it has informed its active slave, its slave in hold, its slave in sniff, and had time to un-park its parked slaves.

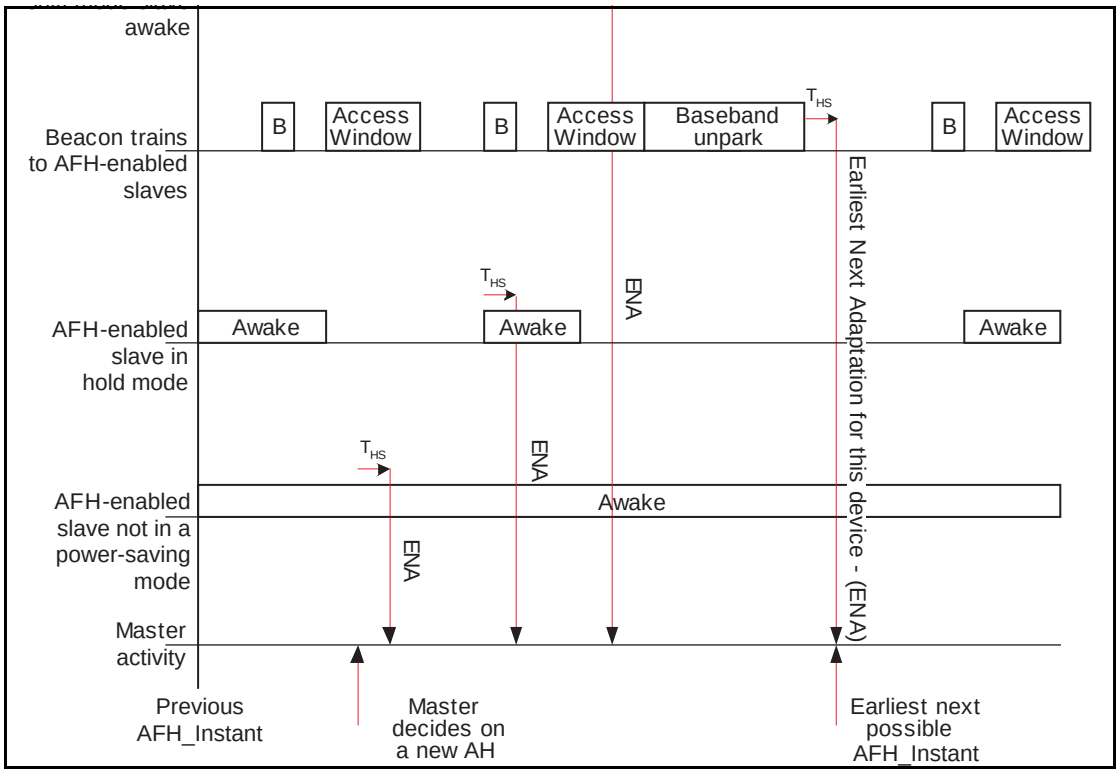


Figure 11.4: Timing constraint on AFH_Instant with slaves in park, hold and sniff

Operation in park

A slave that is in the Park state cannot send or receive any AFH LMP messages. Once the slave has left the Park state the master may subsequently update the slave's adaptation.

AFH Operation in Sniff

Once a slave has been placed in sniff mode, the master may periodically change its AHS without taking the slave out of sniff mode.

AFH Operation in Hold

A slave that is in hold mode cannot send or receive any LMP messages. Once the slave has left hold mode the master may subsequently update the slave's adaptation.

LINK MANAGER PROTOCOL SPECIFICATION

This specification describes the Link Manager Protocol (LMP) which is used for link set-up and control. The signals are interpreted and filtered out by the Link Manager on the receiving side and are not propagated to higher layers.

Ford Ex. 1052

CONTENTS

1 Introduction 211

2 General Rules 212

2.1 Message Transport 212
2.2 Synchronization 212
2.3 Packet Format 213
2.4 Transactions 214
2.4.1 LMP Response Timeout 215
2.5 Error Handling 215
2.5.1 Transaction Collision Resolution 216
2.6 Procedure Rules 216
2.7 General Response Messages 217
2.8 LMP Message Constraints 217

3 Device Features 218

3.1 General Description 218
3.2 Feature Definitions 218
3.3 Feature Mask Definition 225
3.4 Link Manager Interoperability policy 227

4 Procedure Rules 228

4.1 Connection Control 228
4.1.1 Connection Establishment 228
4.1.2 Detach 229
4.1.3 Power Control 230
4.1.4 Adaptive Frequency Hopping 234
4.1.5 Channel Classification 237
4.1.6 Link Supervision 239
4.1.7 Channel Quality Driven Data Rate Change (CQDDR) 240
4.1.8 Quality of Service (QoS) 241
4.1.9 Paging Scheme Parameters 243
4.1.10 Control of Multi-slot Packets 244
4.1.11 Enhanced Data Rate 244
4.1.12 Encapsulated LMP PDUs 246
4.2 Security 247
4.2.1 Authentication 247
4.2.2 Pairing 249
4.2.3 Change Link Key 252
4.2.4 Change Current Link Key Type 253
4.2.5 Encryption 255

4.2.6 Request Supported Encryption Key Size 262
4.2.7 Secure Simple Pairing 262
4.3 Informational Requests 275
4.3.1 Timing Accuracy 275
4.3.2 Clock Offset 277
4.3.3 LMP version 277
4.3.4 Supported Features 278
4.3.5 Name Request 280
4.4 Role Switch 281
4.4.1 Slot Offset 281
4.4.2 Role Switch 282
4.5 Modes of Operation 284
4.5.1 Hold Mode 284
4.5.2 Park State 286
4.5.3 Sniff Mode 293
4.6 Logical Transports 297
4.6.1 SCO Logical Transport 297
4.6.2 eSCO Logical Transport 300
4.7 Test Mode 305
4.7.1 Activation and Deactivation of Test Mode 305
4.7.2 Control of Test Mode 306
4.7.3 Summary of Test Mode PDUs 306

5 Summary 310
5.1 PDU Summary 310
5.2 Parameter Definitions 320
5.3 LMP Encapsulated 331
5.4 Default Values 332

6 List of Figures 333

7 List of Tables 337

1 INTRODUCTION

The Link Manager Protocol (LMP) is used to control and negotiate all aspects of the operation of the Bluetooth connection between two devices. This includes the set-up and control of logical transports and logical links, and for control of physical links.

The Link Manager Protocol is used to communicate between the Link Managers (LM) on the two devices which are connected by the ACL logical transport. All LMP messages shall apply solely to the physical link and associated logical links and logical transports between the sending and receiving devices.

The protocol is made up of a series of messages which shall be transferred over the ACL-C logical link on the default ACL logical transport between two devices. LMP messages shall be interpreted and acted-upon by the LM and shall not be directly propagated to higher protocol layers.

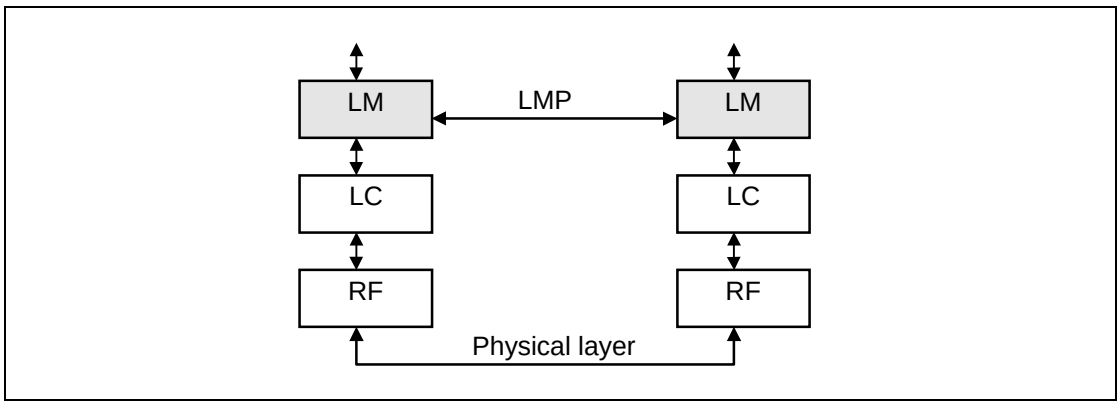


Figure 1.1: Link Manager Protocol signaling layer.

2 GENERAL RULES

2.1 MESSAGE TRANSPORT

LMP messages shall be exchanged over the ACL-C logical link that is carried on the default ACL logical transport (Baseband Specification, Section 4.4, on page 98). The ACL-C logical link is distinguished from the ACL-U (which carries L2CAP and user data) by the Logical Link Identifier (LLID) field carried in the payload header of variable-length packets (Baseband Specification, Section 6.6.2, on page 129).

The ACL-C has a higher priority than other traffic - see Baseband Specification, Section 5.6, on page 107.

The error detection and correction capabilities of the baseband ACL logical transport are generally sufficient for the requirements of the LMP. Therefore LMP messages do not contain any additional error detection information beyond what can be realized by means of sanity checks performed on the contents of LMP messages. Any such checks and protections to overcome undetected errors in LMP messages is an implementation matter.

2.2 SYNCHRONIZATION

This section is informative and explains why many of the LMP message sequences are defined as they are.

LMP messages are carried on the ACL-C logical link, which does not guarantee a time to deliver or acknowledge packets. LMP procedures take account of this when synchronizing state changes in the two devices. For example, criteria are defined that specify when a logical transport address (LT_ADDR) may be re-used after it becomes available due to a device leaving the piconet or entering the park state. Other LMP procedures, such as hold or role switch include the Bluetooth clock as a parameter in order to define a fixed synchronization point. The transitions into and out of sniff mode are protected with a transition mode.

The LC normally attempts to communicate with each slave no less often than every T_{poll} slots (see Section 4.1.8 on page 241) based on the T_{poll} for that slave.

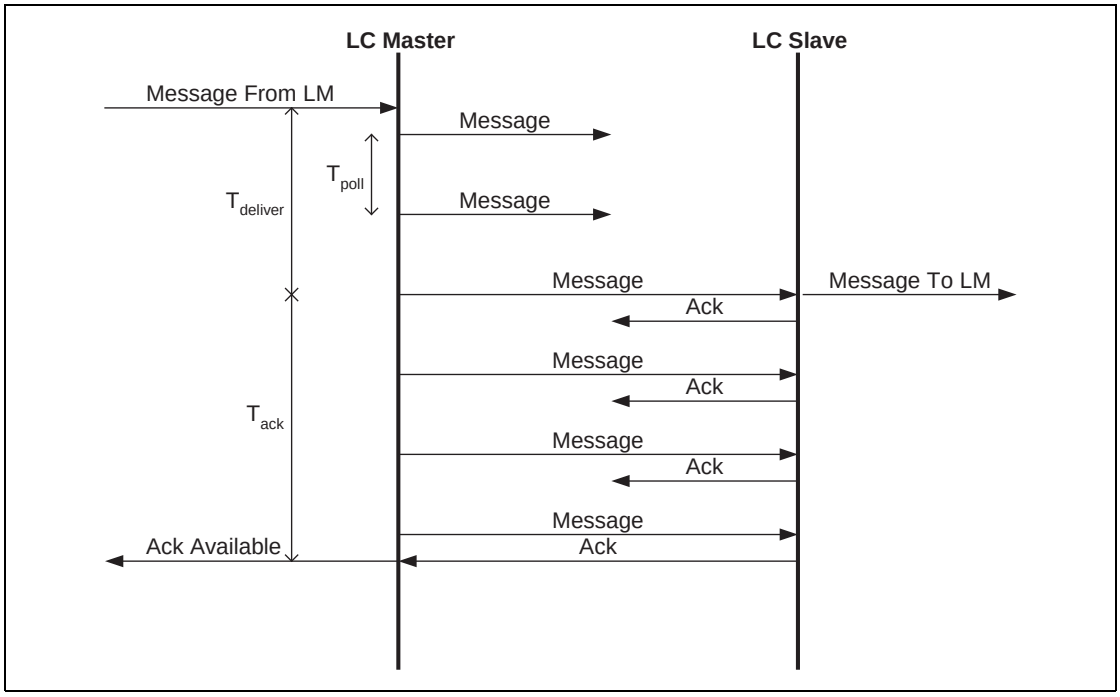


Figure 2.1: Transmission of a message from master to slave[1].

Figure 2.1 on page 213 illustrates the fundamental problem. It shows the transmission of a packet from the master to the slave in conditions of heavy interference for illustrative purposes. It is obvious that neither side can determine the value of either $T_{deliver}$ or T_{ack}. It is therefore not possible to use simple messages to identify uniquely the instant at which a state change occurs in the other device.

2.3 PACKET FORMAT

Each PDU is assigned either a 7 or a 15 bit opcode used to uniquely identify different types of PDUs, see Table 5.1 on page 310. The first 7 bits of the opcode and a transaction ID are located in the first byte of the payload body. If the initial 7 bits of the opcode have one of the special escape values 124-127 then an additional byte of opcode is located in the second byte of the payload and the combination uniquely identifies the PDU.

The FLOW bit in the packet header is always 1 and is ignored on reception.

If the PDU contains one or more parameters these are placed in the payload starting immediately after the opcode, i.e. at byte 2 if the PDU has a 7 bit opcode or byte 3 if the PDU has a 15 bit opcode. The number of bytes used

1. Note the diagram shows the limiting case where the master transmits the message at intervals of T_{poll}. In the case of heavy interference improved performance is gained by transmitting more often.

depends on the length of the parameters. All parameters have a little-endian format, i.e. the least significant byte is transferred first.

LMP messages shall be transmitted using DM1 packets, however if an HV1 SCO link is in use and the length of the payload is less than 9 bytes then DV packets may be used.

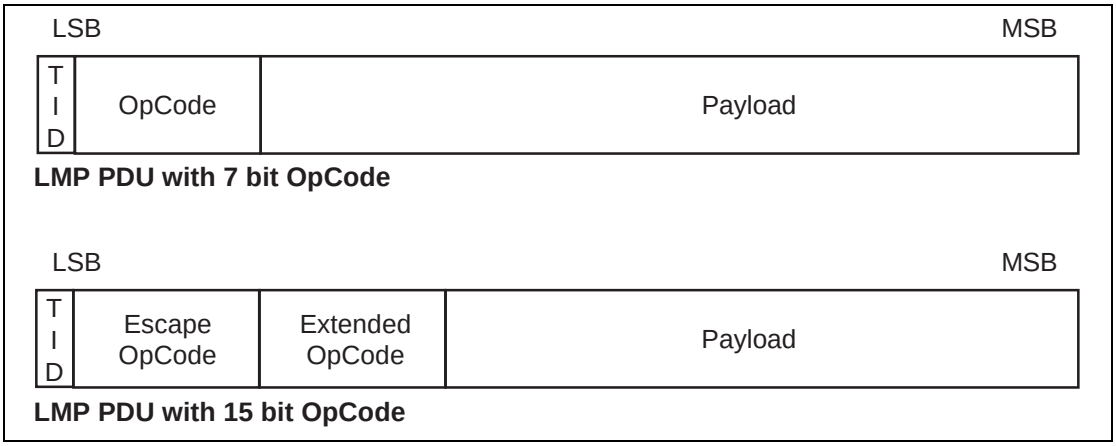


Figure 2.2: Payload body when LMP PDUs are sent.

2.4 TRANSACTIONS

The LMP operates in terms of transactions. A transaction is a connected set of message exchanges which achieve a particular purpose. All PDUs which form part of the same transaction shall have the same value for the transaction ID which is stored as part of the first byte of the OpCode (see Section 2.3 on page 213).

The transaction ID is in the least significant bit. It shall be 0 if the PDU forms part of a transaction that was initiated by the master and 1 if the transaction was initiated by the slave.

Each sequence described in Section 4 on page 228 shall be defined as a transaction. For pairing, see Section 4.2.2 on page 249, and encryption, see Section 4.2.5 on page 255, all sequences belonging to each section shall be counted as one transaction and shall use the same transaction ID. For connection establishment, see Section 4.1.1 on page 228, LMP_host_connection_req and the response with LMP_accepted or LMP_not_accepted shall form one transaction and have the transaction ID of 0. LMP_setup_complete is a stand-alone PDU, which forms a transaction by itself.

For error handling, see Section 2.5 on page 215, the PDU to be rejected and LMP_not_accepted or LMP_not_accepted_ext shall form a single transaction.

2.4.1 LMP Response Timeout

The time between receiving a baseband packet carrying an LMP PDU and sending a baseband packet carrying a valid response PDU, according to the procedure rules in Section 4 on page 228, shall be less than the LMP Response Timeout. The value of this timeout is 30 seconds. Note that the LMP Response Timeout is applied not only to sequences described in Section 4 on page 228, but also to the series of the sequences defined as the transactions in Section 4 on page 228. It shall also be applied to the series of LMP transactions that take place during a period when traffic on the ACL-U logical link is disabled for the duration of the series of LMP transactions, for example during the enabling of encryption.

The LMP Response Timeout shall restart each time an LMP PDU which requires a reply is queued for transmission by the baseband.

2.5 ERROR HANDLING

If the LM receives a PDU with unrecognized opcode, it shall respond with LMP_not_accepted or LMP_not_accepted_ext with the error code *unknown LMP PDU.* The opcode parameter that is echoed back is the unrecognized opcode.

If the LM receives a PDU with invalid parameters, it shall respond with LMP_not_accepted or LMP_not_accepted_ext with the error code *invalid LMP parameters*.

If the maximum response time, see Section 2.4 on page 214, is exceeded or if a link loss is detected (see Baseband Specification, Section 3.1, on page 96), the party that waits for the response shall conclude that the procedure has terminated unsuccessfully.

Erroneous LMP messages can be caused by errors on the channel or systematic errors at the transmit side. To detect the latter case, the LM should monitor the number of erroneous messages and disconnect if it exceeds a threshold, which is implementation-dependent.

When the LM receives a PDU that is not allowed, and the PDU normally expects a PDU reply, for example LMP_host_connection_req or LMP_unit_key, the LM shall return PDU LMP_not_accepted or LMP_not_accepted_ext with the error code *PDU not allowed*. If the PDU normally doesn't expect a reply, for example LMP_sres or LMP_temp_key, the PDU will be ignored.

The reception of an optional PDU which is not supported shall be handled in one of two ways: if the LM simply does not know the opcode (e.g. it was added at a later version of the specification) it shall respond with LMP_not_accepted or LMP_not_accepted_ext with the error code *unknown LMP PDU*. If the LM recognizes the PDU as optional but unsupported then it shall reply with LMP_not_accepted or LMP_not_accepted_ext with the error code *unsupported LMP feature* if the PDU would normally generate a reply otherwise no reply is generated.

2.5.1 Transaction Collision Resolution

Since LMP PDUs are not interpreted in real time, collision situations can occur where both LMs initiate the same procedure and both cannot be completed. In this situation, the master shall reject the slave-initiated procedure by sending LMP_not_accepted or LMP_not_accepted_ext with the error code *LMP error transaction collision*. The master-initiated procedure shall then be completed.

Collision situations can also occur where both LMs initiate different procedures and both cannot be completed. In this situation, the master shall reject the slave-initiated procedure by sending LMP_not_accepted or LMP_not_accepted_ext with the error code *LMP error different transaction collision*. The master- initiated procedure shall then be completed.

2.6 PROCEDURE RULES

Each procedure is described and depicted with a sequence diagram. The following symbols are used in the sequence diagrams:

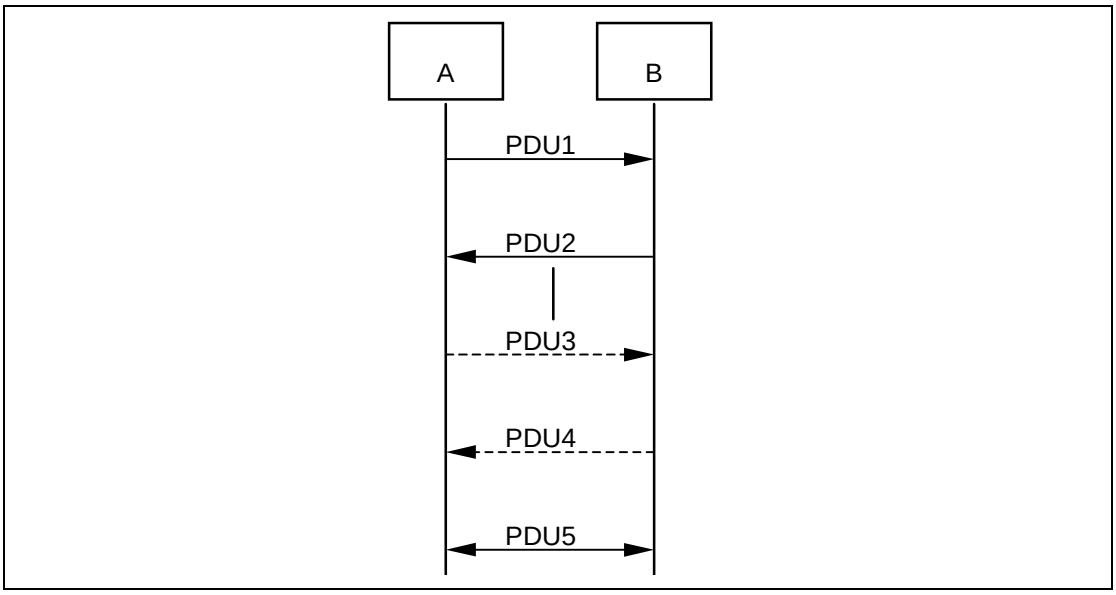


Figure 2.3: Symbols used in sequence diagrams.

PDU1 is a PDU sent from A to B. PDU2 is a PDU sent from B to A. PDU3 is a PDU that is optionally sent from A to B. PDU4 is a PDU that is optionally sent from B to A. PDU5 is a PDU sent from either A or B. A vertical line indicates that more PDUs can optionally be sent.

2.7 GENERAL RESPONSE MESSAGES

The PDUs LMP_accepted, LMP_accepted_ext, LMP_not_accepted and LMP_not_accepted_ext are used as response messages to other PDUs in a number of different procedures. LMP_accepted or LMP_accepted_ext includes the opcode of the message which is accepted. LMP_not_accepted or LMP_not_accepted_ext includes the opcode of the message which is not accepted and the error code why it is not accepted.

LMP_accepted_ext and LMP_not_accepted_ext shall be used when the opcode is 15 bits in length. LMP_accepted and LMP_not_accepted shall be used when the opcode is 7 bits in length.

M/O	PDU	Contents
M	LMP_accepted	op code
M	LMP_not_accepted	op code error code
O	LMP_accepted_ext	escape op code extended op code
O	LMP_not_accepted_ext	escape op code extended op code error code

Table 2.1: General response messages.

2.8 LMP MESSAGE CONSTRAINTS

This section is informative.

- No LMP message shall exceed the maximum payload length of a single DM1 packet i.e. 17 bytes in length (Baseband Specification, Section 6.5.4.1, on page 125).
- All LM messages are of fixed length apart from those sent using broadcast in park state.
- The LMP version number shall not be used to indicate the presence or absence of functionality.

3 DEVICE FEATURES

3.1 GENERAL DESCRIPTION

Each PDU is either Mandatory or Optional as defined by the M/O field in the tables of Section 4 on page 228. An M in this field shall indicate that support for the feature is mandatory. An O in this field shall indicate that support for the PDU is optional and it shall be followed by the number(s) of the feature(s) involved in brackets.

All features added after the 1.1 specification have associated LMP feature bits. Support of these features may be made “mandatory” by the qualification process but the LM still considers them to be optional since it must interoperate with older devices which do not support them.

The LM does not need to be able to transmit a PDU which is optional. Support of optional PDUs is indicated by a device's features mask. The features mask can be read (see Section 4.3.4 on page 278). An LM shall not send or be sent any PDU which is incompatible with the features signalled in its or its peer's features mask, as detailed in Section 3.2.

3.2 FEATURE DEFINITIONS

Feature	Definition
Extended features	This feature indicates whether the device is able to support the extended features mask using the LMP sequences defined in Section 4.3.4 on page 278.
Timing accuracy	This feature indicates whether the LM supports requests for timing accuracy using the sequence defined in Section 4.3.1 on page 275.
Enhanced inquiry scan	This feature indicates whether the device is capable of supporting the enhanced inquiry scan mechanism as defined in Baseband Specification, Section 8.4.1, on page 161. The presence of this feature has only local meaning and does not imply support for any additional LMP PDUs or sequences.
Interlaced inquiry scan	This feature indicates whether the device is capable of supporting the interlaced inquiry scan mechanism as defined in Baseband Specification, Section 8.4.1, on page 161. The presence of this feature has only local meaning and does not imply support for any additional LMP PDUs or sequences.
Interlaced page scan	This feature indicates whether the device is capable of supporting the interlaced page scan mechanism as defined in Baseband Specification, Section 8.3.1, on page 151. The presence of this feature has only local meaning and does not imply support for any additional LMP PDUs or sequences.

Table 3.1: Feature Definitions

Feature	Definition
RSSI with inquiry results	This feature bit shall be set if the "Extended Inquiry Response" feature bit is set. This feature indicates whether the device is capable of reporting the RSSI with inquiry results as defined in Baseband Specification, Section 8.4.2, on page 162. The presence of this feature has only local meaning and does not imply support for any additional LMP PDUs or sequences.
Extended Inquiry Response	This feature indicates whether the device supports extended inquiry response as defined in the Baseband specification. If this bit is set, the "RSSI with inquiry results" feature bit shall also be set.
Paging parameter negotiation	This feature indicates whether the LM is capable of supporting the signaling of changes in the paging scheme as defined in Section 4.1.9 on page 243.
3 slot packets	This feature indicates whether the device supports the transmission and reception of both DM3 and DH3 packets for the transport of traffic on the ACL-U logical link.
5 slot packets	This feature indicates whether the device supports the transmission and reception of both DM5 and DH5 packets for the transport of traffic on the ACL-U logical link.
Flow control lag	This is defined as the total amount of ACL-U data that can be sent following the receipt of a valid payload header with the payload header flow bit set to 0 and is in units of 256 bytes. See further in Baseband Specification, Section 6.6.2, on page 129.
AFH capable slave	This feature indicates whether the device is able to support the Adaptive Frequency Hopping mechanism as a slave as defined in Baseband Specification, Section 2.3, on page 76 using the LMP sequences defined in Section 4.1.4 on page 234.
AFH classification slave	This feature indicates whether the device is able to support the AFH classification mechanism as a slave as defined in Baseband Specification, Section 8.6.8, on page 178 using the LMP sequences defined Section 4.1.5 on page 237.
AFH capable master	This indicates whether the device is able to support the Adaptive Frequency Hopping mechanism as a master as defined in Baseband Specification, Section 2.3, on page 76 using the LMP sequences defined in Section 4.1.4 on page 234.
AFH classification master	This feature indicate whether the device is able to support the AFH classification mechanism as a master as defined in Baseband Specification, Section 8.6.8, on page 178 using the LMP sequences defined Section 4.1.5 on page 237.
Power control	This feature indicates whether the device is capable of adjusting its transmission power. This feature indicates the ability to receive the LMP_incr_power and LMP_decr_power PDUs and transmit the LMP_max_power and LMP_min_power PDUs, using the sequences defined in Section 4.1.3 on page 230. These sequences may be used even if the remote device does not support the power control feature, as long as it supports the Power control requests feature.

Table 3.1: Feature Definitions

Feature	Definition
Power control requests	This feature indicates whether the device is capable of determining if the transmit power level of the other device should be adjusted and will send the LMP_incr_power and LMP_decr_power PDUs to request the adjustments. It indicates the ability to receive the LMP_max_power and LMP_min_power PDUs, using the sequences in Section 4.1.3 on page 230. These sequences may be used even if the remote device does not support the RSSI feature, as long as it supports the power control feature.
Enhanced power control	This feature indicates whether the device is able to support enhanced power control using the LMP sequences as defined in Section 4.1.3.1 on page 232. This feature bit shall only be set if the "power control" feature bit (bit 18) is set and if the "power control requests" feature bit (bit 9) is set.
Channel Quality Driven Data Rate	This feature indicates whether the LM is capable of recommending a packet type (or types) depending on the channel quality using the LMP sequences defined in Section 4.1.7 on page 240.
Broadcast encryption	This feature indicates whether the device is capable of supporting broadcast encryption as defined in [Part H] Section 4.2 on page 1073 and also the sequences defined in Section 4.2.6 on page 262 and Section 4.2.4 on page 253. Note: Devices compliant to versions of this specification 1.1 and earlier may support broadcast encryption even though this feature bit is not set.
Encryption	This feature shall be supported. This feature indicates whether the device supports the encryption of packet contents using the sequence defined in Section 4.2.5 on page 255.
Pause Encryption	When this feature bit is enabled on both sides, then the encryption pause/resume sequences shall be used. If either side does not support the Pause Encryption feature bit, then the encryption pause/resume sequences shall not be used.
Slot offset	This feature indicates whether the LM supports the transfer of the slot offset using the sequence defined in Section 4.4.1 on page 281.
Role switch	This feature indicates whether the device supports the change of master and slave roles as defined by baseband Section 8.6.5 on page 172 using the sequence defined in Section 4.4.2 on page 282.
Hold mode	This feature indicates whether the device is able to support Hold mode as defined in baseband Section 8.8 on page 183 using the LMP sequences defined in Section 4.5.1 on page 284.
Sniff mode	This feature indicates whether the device is able to support sniff mode as defined in baseband Section 8.7 on page 180 using the LMP sequences defined in Section 4.5.3 on page 293.
Sniff subrating	This feature indicates whether the device is able to support sniff subrating as defined in [PART B] Sniff Subrating on page 182 using the LMP sequences as defined in Sniff Subrating on page 295.This feature bit shall be set if the "Sniff mode" feature bit is set.

Table 3.1: Feature Definitions

Feature	Definition
Park state	This feature indicates whether the device is able to support Park state as defined in baseband Section 8.9 on page 184 using the LMP sequences defined in Section 4.5.2 on page 286.
SCO link	This feature shall indicate whether the device is able to support the SCO logical transport as defined in Baseband Specification, Section 4.3, on page 98, the HV1 packet defined in Baseband Specification, Section 6.5.2.1, on page 122 and receiving the DV packet defined in Baseband Specification, Section 6.5.2.4, on page 122 using the LMP sequence in Section 4.6.1 on page 297.
HV2 packets	This feature indicates whether the device is capable of supporting the HV2 packet type as defined in baseband Section 6.5.2.2 on page 122 on the SCO logical transport.
HV3 packets	This feature indicates whether the device is capable of supporting the HV3 packet type as defined in baseband Section 6.5.2.3 on page 122 on the SCO logical transport.
µ-law log synchronous data	This feature indicates whether the device is capable of supporting µ-Law Log format data as defined in Baseband Specification, Section 9.1, on page 192 on the SCO and eSCO logical transports.
A-law log synchronous data	This feature indicates whether the device is capable of supporting A-Law Log format data as defined in Baseband Specification, Section 9.1, on page 192 on the SCO and eSCO logical transports.
CVSD synchronous data	This feature indicates whether the device is capable of supporting CVSD format data as defined in Baseband Specification, Section 9.2, on page 192 on the SCO and eSCO logical transports.
Transparent synchronous data	This feature indicates whether the device is capable of supporting transparent synchronous data as defined in Baseband Specification, Section 6.4.3, on page 116 on the SCO and eSCO logical transports.
Extended SCO link	This feature indicates whether the device is able to support the eSCO logical transport as defined Baseband Specification, Section 5.5, on page 107 and the EV3 packet defined in Baseband Specification, Section 6.5.3.1, on page 123 using the LMP sequences defined in Section 4.6.2 on page 300.
EV4 packets	This feature indicates whether the device is capable of supporting the EV4 packet type defined in Baseband Specification, Section 6.5.3.2, on page 123 on the eSCO logical transport.
EV5 packets	This feature indicates whether the device is capable of supporting the EV5 packet type defined in Baseband Specification, Section 6.5.3.3, on page 123 on the eSCO logical transport.
Enhanced Data Rate ACL 2 Mbps mode	This feature indicates whether the device supports the transmission and reception of 2-DH1 packets for the transport of traffic on the ACL-U logical link.

Table 3.1: Feature Definitions

Feature	Definition
Enhanced Data Rate ACL 3 Mbps mode	This feature indicates whether the device supports the transmission and reception of 3-DH1 packets for the transport of traffic on the ACL-U logical link. This feature bit shall only be set if the “Enhanced Data Rate ACL 2 Mbps mode” feature bit is set.
3-slot Enhanced Data Rate ACL packets	This feature indicates whether the device supports the transmission and reception of three-slot Enhanced Data Rate packets on the ACL-U logical link. This feature bit shall only be set if the “Enhanced Data Rate ACL 2 Mbps mode” feature bit is set.
5-slot Enhanced Data Rate ACL packets	This feature indicates whether the device supports the transmission and reception of 5-slot Enhanced Data Rate packets for the transport of traffic on the ACL-U logical link. This feature bit shall only be set if the “Enhanced Data Rate ACL 2 Mbps mode” feature bit is set.
Enhanced Data Rate eSCO 2 Mbps mode	This feature indicates whether the device supports the transmission and reception of 2-EV3 packets for the transport of traffic on the eSCO logical transport.
Enhanced Data Rate eSCO 3 Mbps mode	This feature indicates whether the device supports the transmission and reception of 3-EV3 packets for the transport of traffic on the eSCO logical transport. This feature bit shall only be set if the “Enhanced Data Rate eSCO 2 Mbps mode” feature bit is set.
3-slot Enhanced Data Rate eSCO packets	This feature indicates whether the device supports the transmission and reception of 3-slot Enhanced Data Rate packets for the transport of traffic on the eSCO logical transport. This feature bit shall only be set if the “Enhanced Data Rate eSCO 2 Mbps mode” feature bit is set.
Erroneous Data Reporting	This feature indicates whether the device is able to support the Packet_Status_Flag and the HCI commands "Write Default Erroneous Data Reporting" and "Read Default Erroneous Data Reporting". This feature bit may be set if at least one of the "SCO link" or "Extended SCO link" feature bits are set.
Link Supervision Timeout Changed Event	This feature bit indicates whether the device supports the sending the HCI Link Supervision Timeout Changed Event to the Host.
Non-flushable Packet Boundary Flag	This feature indicates that the device supports the capability to correctly handle HCI ACL Data Packets with a Packet_Boundary_Flag value of 00 (Non-Automatically-Flushable packet).
Secure Simple Pairing (Controller Support)	This feature indicates whether the Link Manager is capable of supporting the Secure Simple Pairing, Section 4.2.7, “Secure Simple Pairing,” on page 262.

Table 3.1: Feature Definitions

Feature	Definition
Secure Simple Pairing (Host Support)	This feature indicates whether the Host is capable of supporting the Secure Simple Pairing Section 4.2.7, “Secure Simple Pairing,” on page 262. If HCI is supported, this bit shall be set if Write Simple Pairing Mode has been issued to enable the Secure Simple Pairing feature. When HCI is not supported, this bit may be set using a vendor specific mechanism, This bit shall only be set if the Secure Simple Pairing (Controller Support) bit is set.
Encapsulated PDU	This feature indicates whether the device is capable of supporting the Encapsulated mechanism as defined in Section 4.1.12.1, “Sending an Encapsulated PDU,” on page 246. This feature shall be set if Secure Simple Pairing is supported.
Variable Inquiry TX Power Level	This feature indicates whether the device is capable of setting the TX power level for Inquiry.
BR/EDR Not Supported	This feature indicates whether the device does not support BR/EDR. If this feature bit is set, then all other LMP feature bits shall be set to zero except the LE Supported (Controller) feature bit that shall be set to one.
LE Supported (Controller)	This feature indicates whether the Controller supports LE. The local Host uses this feature bit to determine whether the Controller supports LE. A remote device does not use this feature bit.
Simultaneous LE and BR/EDR to Same Device Capable (Controller)	This feature indicates that the Controller supports simultaneous LE and BR/EDR links to the same remote device. This bit shall only be set if the LE Supported (Controller) bit is set. The local Host uses this feature bit to determine whether the Controller is capable of supporting simultaneous LE and BR/EDR connections to a remote device. A remote device does not use this feature bit.
LE Supported (Host)	This feature indicates that the Host supports LE. This bit shall only be set if the LE Supported (Controller) bit is set on the local Controller. The local Host sets this feature bit to indicate to a remote device that the local device is capable of supporting LE. A remote Host uses this feature bit to determine whether an LE connection to the peer device is possible.

Table 3.1: Feature Definitions

Feature	Definition
Simultaneous LE and BR/EDR to Same Device Capable (Host)	This feature indicates that the Host supports simultaneous LE and BR/EDR links to the same remote device. This bit shall only be set if the Simultaneous LE and BR/EDR to Same Device Capable (Controller) bit is set on the local Controller. The local Host sets this feature bit to indicate to a remote device that the local device is capable of supporting simultaneous LE and BR/EDR connections to the peer device. A remote Host uses this feature to determine whether simultaneous LE and BR/EDR connections are possible to the peer device.

Table 3.1: Feature Definitions

3.3 FEATURE MASK DEFINITION

The features are represented as a bit mask when they are transferred in LMP messages. For each feature a single bit is specified which shall be set to 1 if the feature is supported and set to 0 otherwise. The single exception is the flow control lag which is coded as a 3 bit field with the least significant bit in byte 2 bit 4 and the most significant bit in byte 2 bit 6. All removed, unknown. or unassigned feature bits shall be set to 0 and ignored upon receipt.

No.	Supported feature	Byte	Bit
0	3 slot packets	0	0
1	5 slot packets	0	1
2	Encryption	0	2
3	Slot offset	0	3
4	Timing accuracy	0	4
5	Role switch	0	5
6	Hold mode	0	6
7	Sniff mode	0	7
8	Park state	1	0
9	Power control requests	1	1
10	Channel quality driven data rate (CQDDR)	1	2
11	SCO link	1	3
12	HV2 packets	1	4
13	HV3 packets	1	5
14	μ-law log synchronous data	1	6
15	A-law log synchronous data	1	7
16	CVSD synchronous data	2	0
17	Paging parameter negotiation	2	1
18	Power control	2	2
19	Transparent synchronous data	2	3
20	Flow control lag (least significant bit)	2	4
21	Flow control lag (middle bit)	2	5
22	Flow control lag (most significant bit)	2	6
23	Broadcast Encryption	2	7
24	Reserved	3	0
25	Enhanced Data Rate ACL 2 Mbps mode	3	1

Table 3.2: Feature mask definitions (page 0)

No.	Supported feature	Byte	Bit
26	Enhanced Data Rate ACL 3 Mbps mode	3	2
27	Enhanced inquiry scan	3	3
28	Interlaced inquiry scan	3	4
29	Interlaced page scan	3	5
30	RSSI with inquiry results	3	6
31	Extended SCO link (EV3 packets)	3	7
32	EV4 packets	4	0
33	EV5 packets	4	1
34	Reserved	4	2
35	AFH capable slave	4	3
36	AFH classification slave	4	4
37	BR/EDR Not Supported	4	5
38	LE Supported (Controller)	4	6
39	3-slot Enhanced Data Rate ACL packets	4	7
40	5-slot Enhanced Data Rate ACL packets	5	0
41	Sniff subrating	5	1
42	Pause encryption	5	2
43	AFH capable master	5	3
44	AFH classification master	5	4
45	Enhanced Data Rate eSCO 2 Mbps mode	5	5
46	Enhanced Data Rate eSCO 3 Mbps mode	5	6
47	3-slot Enhanced Data Rate eSCO packets	5	7
48	Extended Inquiry Response	6	0
49	Simultaneous LE and BR/EDR to Same Device Capable (Controller)	6	1
50	Reserved	6	2
51	Secure Simple Pairing	6	3
52	Encapsulated PDU	6	4
53	Erroneous Data Reporting	6	5
54	Non-flushable Packet Boundary Flag	6	6
55	Reserved	6	7
56	Link Supervision Timeout Changed Event	7	0

Table 3.2: Feature mask definitions (page 0)

No.	Supported feature	Byte	Bit
57	Inquiry TX Power Level	7	1
58	Enhanced Power Control	7	2
59	Reserved	7	3
60	Reserved	7	4
61	Reserved	7	5
62	Reserved	7	6
63	Extended features	7	7

Table 3.2: Feature mask definitions (page 0)

No.	Supported Feature	Byte	Bit
64	Secure Simple Pairing (Host Support)	0	0
65	LE Supported (Host)	0	1
66	Simultaneous LE and BR/EDR to Same Device Capable (Host)	0	2

Table 3.3: Extended feature mask definition (page 1)

3.4 LINK MANAGER INTEROPERABILITY POLICY

Link managers of any version shall interpret using the lowest common subset of functionality by reading the LMP features mask (defined in Table 3.2 on page 225).

An optional LMP PDU shall only be sent to a device if the corresponding feature bit is set in its feature mask. The exception to this are certain PDUs (see Section 4.1.1 on page 228) which can be sent before the features mask is requested. Note: a later version device with a restricted feature set is indistinguishable from an earlier version device with the same features.

4 PROCEDURE RULES

4.1 CONNECTION CONTROL

4.1.1 Connection Establishment

After the paging procedure, LMP procedures for clock offset request, LMP version, supported features, name request and detach may then be initiated.

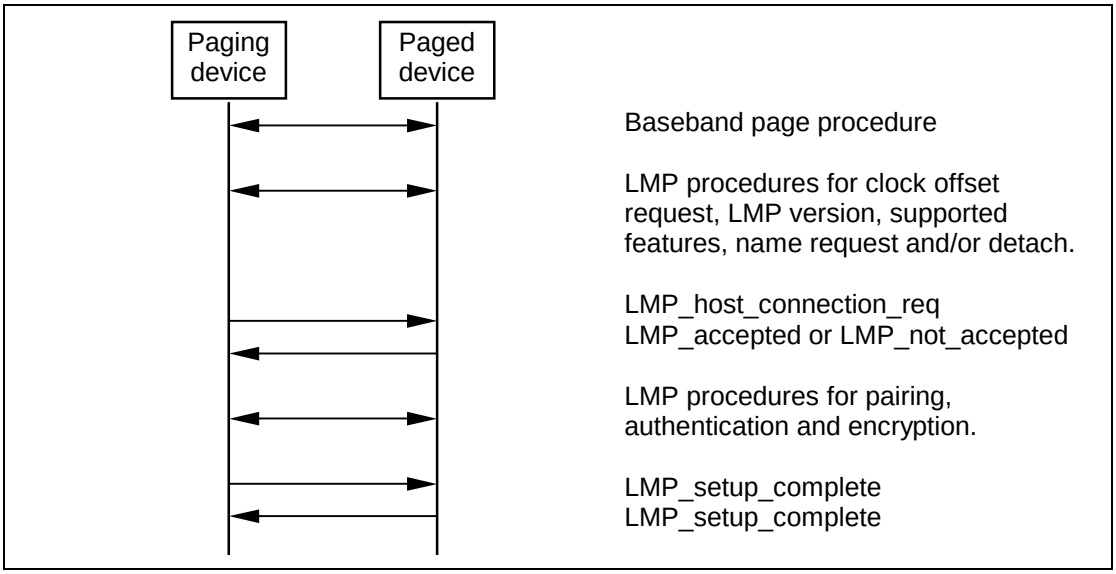


Figure 4.1: Connection establishment.

When the paging device wishes to create a connection involving layers above LM, it sends an LMP_host_connection_req PDU. When the other side receives this message, the host is informed about the incoming connection. The remote device can accept or reject the connection request by sending an LMP_accepted PDU or an LMP_not_accepted PDU. Alternatively, if the slave needs a role switch, see Section 4.4.2 on page 282, it sends an LMP_slot_offset PDU and LMP_switch_req PDU after it has received an LMP_host_connection_req PDU. If the role switch fails the LM shall continue with the creation of the connection unless this cannot be supported due to limited resources in which case the connection shall be terminated with an LMP_detach PDU with error code *other end terminated connection: low resources*. When the role switch has been successfully completed, the old slave will reply with an LMP_accepted PDU or an LMP_not_accepted PDU to the LMP_host_connection_req PDU (with the transaction ID set to 0).

If the paging device receives an LMP_not_accepted PDU in response to LMP_host_connection_req it shall immediately disconnect the link using the mechanism described in Section 4.1.2 on page 229.

If the LMP_host_connection_req PDU is accepted, LMP security procedures (pairing, authentication and encryption) may be invoked. When a device is not going to initiate any more security procedures during connection establishment it sends an LMP_setup_complete PDU. When both devices have sent LMP_setup_complete PDUs the traffic can be transferred on the ACL-U logical transport.

M/O	PDU	Contents
M	LMP_host_connection_req	-
M	LMP_setup_complete	-

Table 4.1: Connection establishment PDU

4.1.2 Detach

The connection between two Bluetooth devices may be detached anytime by the master or the slave. An error code parameter is included in the message to inform the other party of why the connection is detached.

M/O	PDU	Contents
M	LMP_detach	error code

Table 4.2: Detach PDU

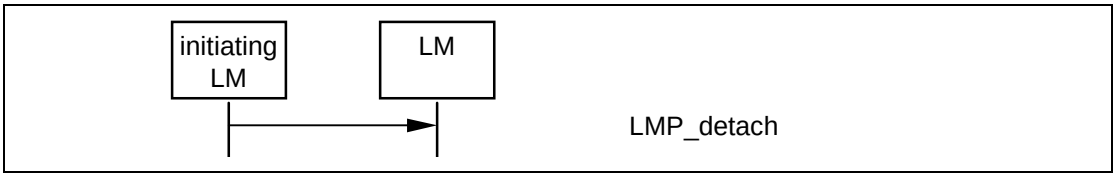
The initiating LM shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107). The initiating LM then queues the LMP_detach for transmission and it shall start a timer for $6*T_{poll}$ slots where T_{poll} is the poll interval for the connection. If the initiating LM receives the baseband acknowledgement before the timer expires it starts a timer for $3*T_{poll}$ slots. When this timer expires, and if the initiating LM is the master, the LT_ADDR(s) may be re-used immediately. If the initial timer expires then the initiating LM drops the link and starts a timer for $T_{linksupervisiontimeout}$ slots after which the LT_ADDR(s) may be re-used if the initiating LM is the master.

When the receiving LM receives the LMP_detach, it shall start a timer for $6*T_{poll}$ slots if it is the master and $3*T_{poll}$ if it is the slave. On timer expiration, the link shall be detached and, if the receiving LM is the master, the LT_ADDR(s) may be re-used immediately. If the receiver never receives the LMP_detach then a link supervision timeout will occur, the link will be detached, and the LT_ADDR may be re-used immediately.

If at any time during this or any other LMP sequence the Link supervision time-out expires then the link shall be terminated immediately and the LT_ADDR(S) may be re-used immediately.

If the connection is in hold mode, the initiating LM shall wait for hold mode to end before initiating the procedure defined above. If the connection is in sniff

mode or park state, the initiating LM shall perform the procedure to exit sniff mode or park state before initiating the procedure defined above. If the procedure to exit sniff mode or park state does not complete within the LMP response timeout (30 seconds) the procedure defined above shall be initiated anyway.



Sequence 1: Connection closed by sending LMP_detach.

4.1.3 Power Control

If the received signal characteristics differ too much from the preferred value of a Bluetooth device, it may request an increase or a decrease of the other device's transmit power level. A device's transmit power level is a property of the physical link, and affects all logical transports carried over the physical link. Power control requests carried over the default ACL-C logical link shall only affect the physical link associated with the requesting device's default ACL-C logical link; they shall not affect the power level used on the physical links to other slaves.

Two power control mechanisms are specified: legacy power control (see Table 4.3) and enhanced power control (see Table 4.4 and Section 4.1.3.1). A device that supports enhanced power control shall also support legacy power control.

M/O	PDU	Contents
O(9)	LMP_incr_power_req	for future use (1 Byte)
O(9)	LMP_decr_power_req	for future use (1 Byte)
O(18)	LMP_max_power	-
O(18)	LMP_min_power	-

Table 4.3: Legacy Power control PDU

M/O	PDU	Contents
O(58)	LMP_power_control_req	power_adjustment_request
O(58)	LMP_power_control_res	power_adjustment_response

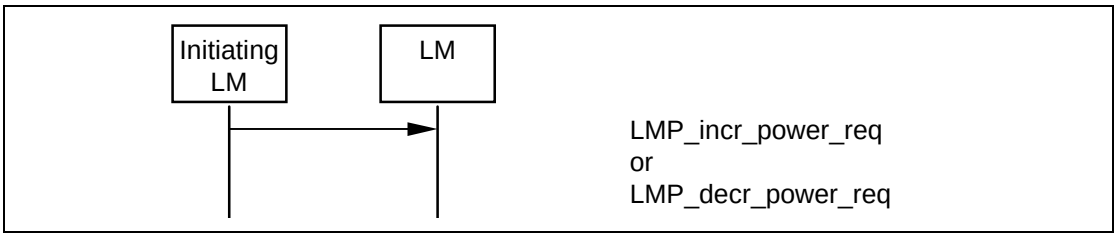
Table 4.4: Enhanced Power control PDU

The power adjustment requests may be made at any time using the legacy power control mechanism following a successful baseband paging procedure and before the Link Manager supported features responses have been processed. After the Link Manager supported features responses have been pro-

cessed, if both devices support enhanced power control (see Section 4.1.3.1) then only enhanced power control shall be used. Otherwise, if either device supports only the legacy power control mechanism then only legacy power control shall be used.

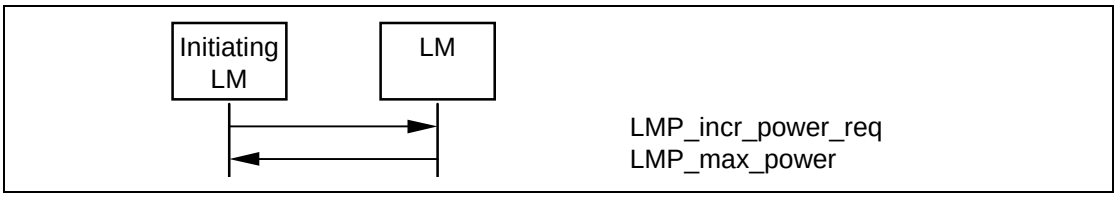
If a device does not support power control requests this is indicated in the supported features list and thus no power control requests shall be sent after the supported features response has been processed. Prior to this time, a power control adjustment might be sent and if the recipient does not support power control it is allowed to send LMP_max_power in response to LMP_incr_power_req and LMP_min_power in response to LMP_decr_power_req. Another possibility is to send LMP_not_accepted with the error code *unsupported LMP feature*.

Upon receipt of an LMP_incr_power_req PDU or LMP_decr_power_req PDU the output power shall be increased or decreased one step. See Radio Specification Section 3 on page 36 for the definition of the step size.

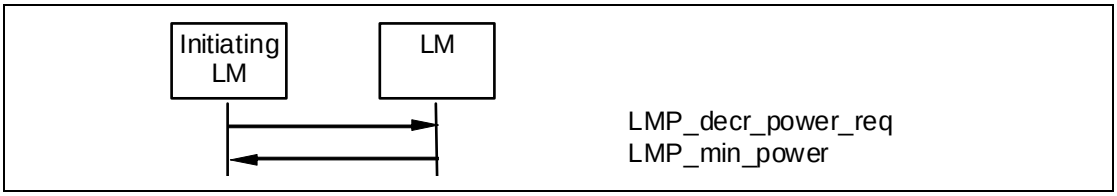


Sequence 2: A device requests a change of the other device's TX power.

If the receiver of LMP_incr_power_req is at maximum power LMP_max_power shall be returned. The device shall only request an increase again after having requested a decrease at least once. If the receiver of LMP_decr_power_req is at minimum power then LMP_min_power shall be returned and the device shall only request a decrease after having requested an increase at least once.



Sequence 3: The TX power cannot be increased.



Sequence 4: The TX power cannot be decreased.

One byte is reserved in LMP_incr/decr_power_req for future use. The parameter value shall be 0x00 and ignored upon receipt.

4.1.3.1 Enhanced Power Control

Enhanced power control shall only be used when both devices support the enhanced power control LMP feature. Legacy power control shall not be used when both devices support the enhanced power control LMP feature.

4.1.3.1.1 Sending Enhanced Power Control Requests

To adjust the remote device's output power, a device shall send the LMP_power_control_req PDU with the power_adjustment_request parameter.

The power_adjustment_request parameter may be set to: one step up, one step down, or all the way to the max power level. The remote device shall respond with an LMP_power_control_res PDU. The responder shall transmit the LMP_power_control_res PDU at the new power level. Upon reception of the LMP_power_control_res PDU, the initiating device should restart any processes used to determine whether additional power level changes are required.

If the receiver of the LMP_power_control_req PDU has indicated that the output power levels for all of the supported modulation modes are at maximum the requesting device shall only request an increase again after having requested a decrease at least once. If the receiver of the LMP_power_control_req PDU has indicated that the output power levels for all of the supported modulation modes are at minimum the requesting device shall only request a decrease after having requested an increase at least once.

A new LMP_power_control_req PDU shall not be sent until an LMP_power_control_res PDU has been received.

4.1.3.1.2 Responding to Enhanced Power Control Requests

When a device receives an LMP_power_control_req PDU with the power_control_adjust parameter set to "increment one step," at least one modulation shall be increased one step unless all supported modulations are at the maximum level. Modulations that would violate the relative power level between modulations (see A, Section 3) shall also increment one step.

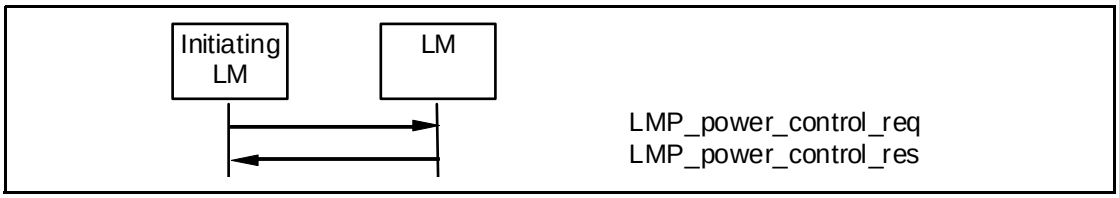
When a device receives an LMP_power_control_req PDU with the power_control_adjust parameter set to "decrement one step", at least one modulation shall be decreased one step unless all supported modulations are at the minimum level. Modulations that would violate the relative power level between modulations (see A, Section 3) shall also decrement one step.

When a device receives an LMP_power_control_req PDU with the power_control_adjust parameter set to "increment power to maximum value", each supported modulation mode shall be set to its maximum power level.

Note: See A, Section 3 for requirements on the relative power levels of different modulation modes.

The responding LM shall send the LMP_power_control_res PDU to indicate the status for every modulation. The power_adjustment_response parameter has three 2-bit fields indicating the status for each modulation mode: GFSK, DQPSK and 8DPSK. Each 2-bit field shall be set to one of the following values: *not supported*, *changed one step, max power*, or *min power*. The changed one step value shall only be used when the power level for that modulation mode has not reached the minimum or maximum level.

The responder shall transmit the LMP_power_control_res PDU at the new transmit power level and shall not change its power level until requested by the remote device by a subsequent LMP_power_control_req PDU.



Sequence 5: A device responds to a power control request.

4.1.4 Adaptive Frequency Hopping

AFH is used to improve the performance of physical links in the presence of interference as well as reducing the interference caused by physical links on other devices in the ISM band. AFH shall only be used during the connection state.

M/O	PDU	Contents
O(35) Rx O(43) Tx	LMP_set_AFH	AFH_Instant, AFH_Mode, AFH_Channel_Map

Table 4.5: AFH PDU

The LMP_set_AFH PDU contains three parameters: AFH_Instant, AFH_Mode, and AFH_Channel_Map. The parameter, AFH_Instant, specifies the instant at which the hopset switch shall become effective. This is specified as a Bluetooth Clock value of the master's clock, that is available to both devices. The AFH instant is chosen by the master and shall be an even value at least $6*T_{poll}$ or 96 slots (whichever is greater) in the future, where T_{poll} is at least the longest poll interval for all AFH enabled physical links. The AFH_instant shall be within 12 hours of the current clock value. The parameter AFH_Mode, specifies whether AFH shall be enabled or disabled. The parameter AFH_Channel_Map, specifies the set of channels that shall be used if AFH is enabled.

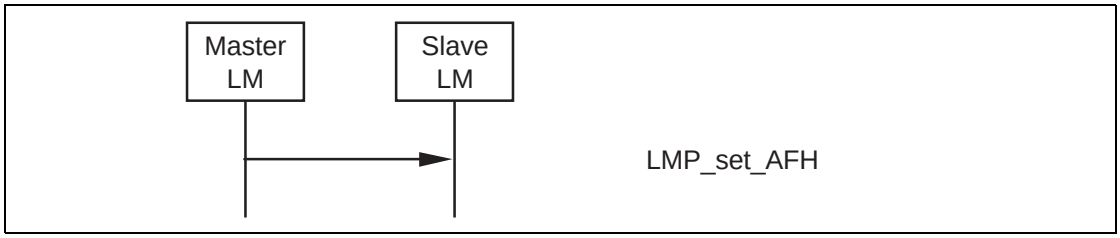
When the LMP_set_AFH PDU is received the AFH instant shall be compared with the current Bluetooth clock value. If it is in the past then the AFH_instant has passed and the slave shall immediately configure the hop selection kernel (see Baseband Specification, Section 2.6.3, on page 90) with the new AFH_mode and AFH_channel_map specified in the LMP_set_AFH PDU. If it is in the future then a timer shall be started to expire at the AFH instant. When this timer expires it shall configure the hop selection kernel with the new AFH_mode and AFH_channel_map.

The master shall not send a new LMP_set_AFH PDU to a slave until it has received the baseband acknowledgement for any previous LMP_set_AFH addressed to that slave and the instant has passed.

Role switch while AFH is enabled shall follow the procedures define by Baseband Specification, Section 8.6.5, on page 172.

4.1.4.1 Master enables AFH

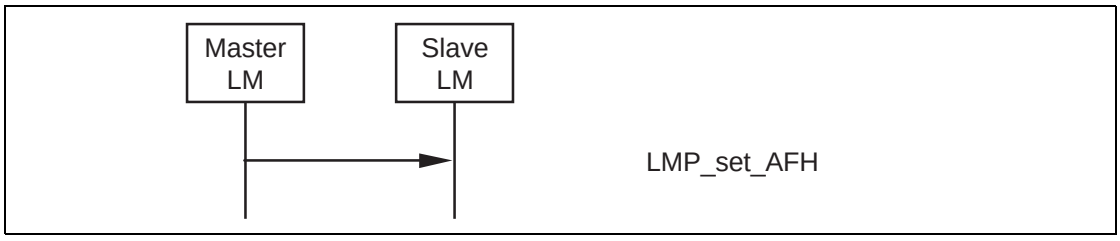
Prior to enabling AFH the master LM shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107). The master shall then enable AFH on a physical link by sending the LMP_set_AFH PDU with AFH_mode set to AFH_enabled, the AFH_channel_map parameter containing the set of used and unused channels, and an AFH_instant. The LM shall not calculate the AFH instant until after traffic on the ACL-U logical link has been stopped. The master considers the physical link to be AFH_enabled once the baseband acknowledgement has been received and the AFH_instant has passed. Once the baseband acknowledgement has been received the master shall restart transmission on the ACL-U logical link.



Sequence 6: Master Enables AFH.

4.1.4.2 Master Disables AFH

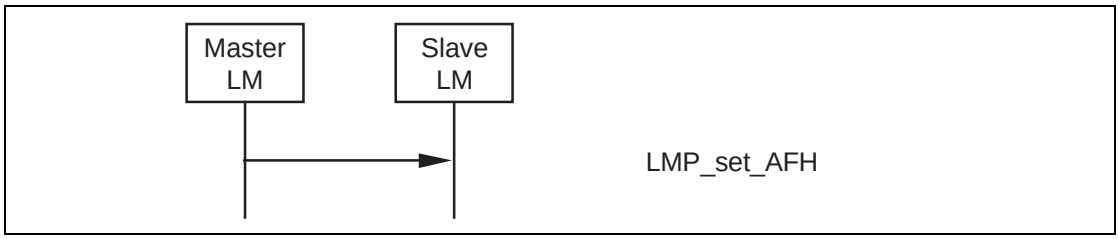
Prior to disabling AFH the master LM shall pause traffic on the ACL-U logical link (Baseband Specification, Section 5.3.1, on page 107). The master shall then disable AFH operation on a physical link by sending the LMP_set_AFH PDU with AFH_mode set to AFH_disabled and an AFH_instant. The AFH_channel_map parameter is not valid when AFH_mode is AFH_disabled. The LM shall not calculate the AFH instant until after traffic on the ACL-U logical link has been stopped. The master considers the physical link to have entered AFH_disabled operation once the baseband acknowledgement has been received and the AFH_instant has passed. Once the baseband acknowledgement has been received the master shall restart transmission on the ACL-U logical link.



Sequence 7: Master disables AFH.

4.1.4.3 Master updates AFH

A master shall update the AFH parameters on a physical link by sending the LMP_set_AFH PDU with AFH_mode set to AFH_enabled, an AFH_instant and a new AFH_channel_map. The master shall consider the slave to have the updated AFH parameters once the baseband acknowledgement has been received and the AFH_instant has passed.



Sequence 8: Master Updates AFH.

4.1.4.4 AFH operation in Park, Hold, and Sniff modes

A slave in Park, Hold or Sniff shall retain the AFH_mode and AFH_channel_map prior to entering those modes. A master may change the AFH_mode while a slave is in sniff.

A master that receives a request from an AFH_enabled slave to enter Park, Hold, or Sniff and decides to operate the slave using a different hop sequence shall respond with an LMP_set_AFH PDU specifying the new hop sequence.

The master continues with the LMP signaling, for Park, Hold or Sniff initiation, once the baseband acknowledgement for the LMP_set_AFH PDU has been received. Optionally, the master may delay the continuation of this LMP signaling until after the instant. An AFH_capable_slave device shall support both of these cases.

A master that receives a request from an AFH_enabled slave to enter Park, Hold or Sniff and decides not to change the slave's hop sequence shall respond exactly as it would do without AFH. In this case, AFH operation has no effect on the LMP signaling.

4.1.5 Channel Classification

A master may request channel classification information from a slave that is AFH_enabled.

A slave that supports the AFH_classification_slave feature shall perform channel classification and reporting according to its AFH_reporting_mode. The master shall control the AFH_reporting_mode using the LMP_channel_classification_req PDU. The slave shall report its channel classification using the LMP_channel_classification PDU.

The slave shall report pairs of channels as *good*, *bad* or *unknown*. See Table 5.2 on page 320 for the detailed format of the AFH_Channel_Classification parameter. When one channel in the nth channel pair is good and the other channel is unknown the nth channel pair shall be reported as good. When one channel in the nth channel pair is bad and the other is unknown the nth channel pair shall be reported as bad. It is implementation dependent what to report when one channel in a channel pair is good and the other is bad.

M/O	PDU	Contents
O(36) Rx O(44) Tx	LMP_channel_classification_req	AFH_Reporting_Mode, AFH_Min_Interval, AFH_Max_Interval
O(36) Tx O(44) Rx	LMP_channel_classification	AFH_Channel_Classification

The LMP_channel_classification_req PDU contains three parameters: AFH_Reporting_Mode, AFH_Min_Interval, and AFH_Max_Interval. In the AFH_reporting_disabled state, the slave shall not generate any channel classification reports. The parameter AFH_min_interval, defines the minimum amount of time from the last LMP_channel_classification command that was sent before the next LMP_channel_classification PDU may be sent. The parameter AFH_max_interval, defines the maximum amount of time between the change in the radio environment being detected by a slave and its generation of an LMP_channel_classification PDU. The AFH_max_interval shall be equal to or larger than AFH_min_interval.

The AFH_reporting_mode parameter shall determine if the slave is in the AFH_reporting_enabled or AFH_reporting_disabled state. The default state, prior to receipt of any LMP_channel_classification_req PDUs, shall be AFH_reporting_disabled.

AFH_reporting_mode is implicitly set to the AFH_reporting_disabled state when any of the following occur:

- Establishment of a connection at the baseband level
- Master-slave role switch
- Entry to park state operation
- Entry to hold mode operation

AFH_reporting_mode is implicitly restored to its former value when any of the following occur:

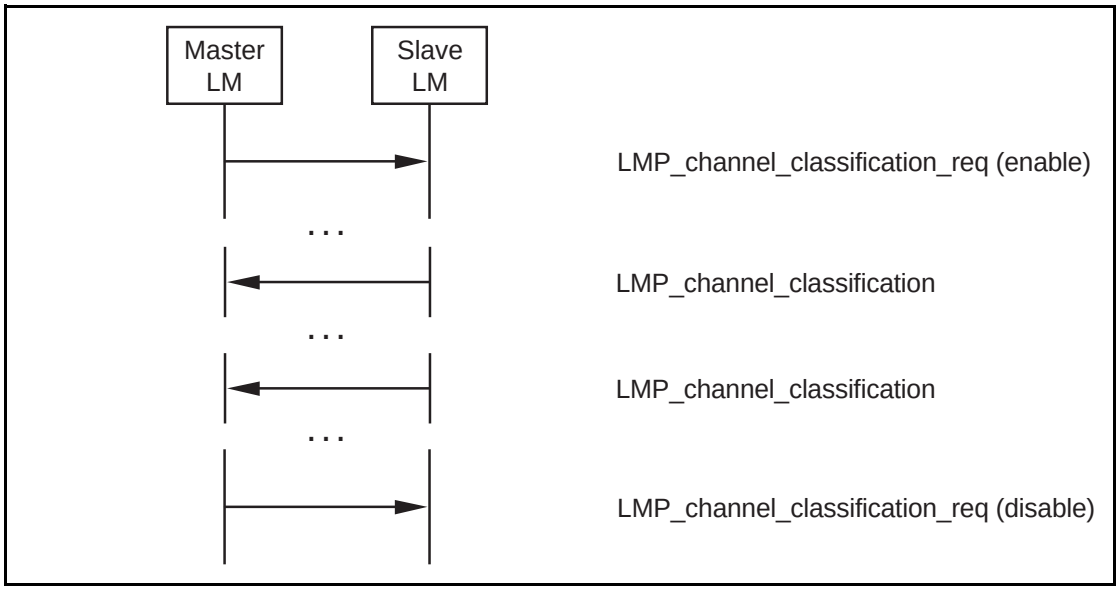
- Exit from park state operation
- Exit from hold mode
- Failure of Master-slave role switch

4.1.5.1 Channel classification reporting enabling and disabling

A master enables slave channel classification reporting by sending the LMP_channel_classification_req PDU with the AFH_reporting_mode parameter set to AFH_reporting_enabled.

When a slave has had classification reporting enabled by the master it shall send the LMP_channel_classification PDU according to the information in the latest LMP_channel_classification_req PDU. The LMP_channel_classification PDU shall not be sent if there has been no change in the slave's channel classification.

A master disables slave channel classification reporting by sending the LMP_channel_classification_req PDU with the AFH_reporting_mode parameter set to AFH_reporting_disabled.



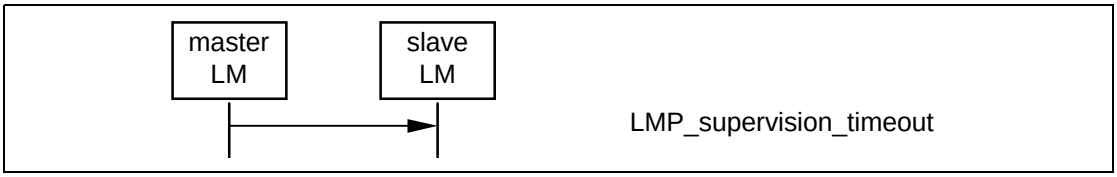
Sequence 9: Channel classification reporting.

4.1.6 Link Supervision

Each physical link has a timer that is used for link supervision. This timer is used to detect physical link loss caused by devices moving out of range, or being blocked by interference, a device's power-down, or other similar failure cases. Link supervision is specified in Baseband Specification, Section 3.1, on page 96.

M/O	PDU	Contents
M	LMP_supervision_timeout	supervision timeout

Table 4.6: Set supervision timeout PDU



Sequence 10: Setting the link supervision timeout.

4.1.7 Channel Quality Driven Data Rate Change (CQDDR)

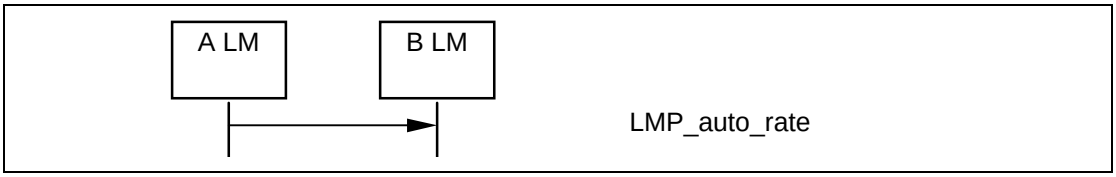
The data throughput for a given packet type depends on the quality of the RF channel. Quality measurements in the receiver of one device can be used to dynamically control the packet type transmitted from the remote device for optimization of the data throughput. Device A sends the LMP_auto_rate PDU once to notify device B to enable this feature. Once enabled, device B may request packet type(s) that A should transmit by sending the LMP_preferred_rate PDU. This PDU has a parameter which determines the preferred error coding (with or without 2/3FEC) and optionally the preferred size in slots of the packets. Device A is not required to change to the packet type specified by this parameter. Device A shall not send a packet that is larger than max slots (see Section 4.1.10 on page 244) even if the preferred size is greater than this value.

The data rate parameter includes the preferred rate for Basic Rate and Enhanced Data Rate modes. When operating in Basic Rate mode, the device shall use bits 0-2 to determine the preferred data rate. When operating in Enhanced Data Rate mode, the device shall use bits 3-6 to determine the preferred data rate. For devices that support Enhanced Data Rate, the preferred rates for both Basic Rate and Enhanced Data Rate modes shall be valid at all times.

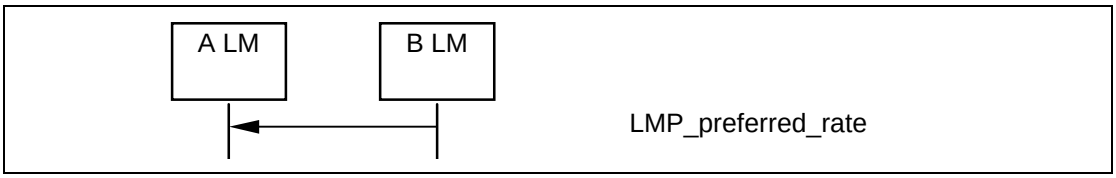
These PDUs may be sent at any time after connection setup is completed.

M/O	PDU	Contents
O(10)	LMP_auto_rate	-
O(10)	LMP_preferred_rate	data rate

Table 4.7: Quality-driven change of the data rate PDU



Sequence 11: A notifies B to enable CQDDR



Sequence 12: B sends A a preferred packet type

4.1.8 Quality of Service (QoS)

The LM provides QoS capabilities. A poll interval, T_{poll}, that is defined as the maximum time between transmissions from the master to a particular slave on the ACL logical transport, is used to support bandwidth allocation and latency control - see Baseband Specification, Section 8.6.1, on page 167 for details. The poll interval is guaranteed in the active and sniff modes except when there are collisions with page, page scan, inquiry and inquiry scan, during time critical LMP sequences in the current piconet and any other piconets in which the Bluetooth device is a member, and during critical baseband sequences (such as the page response, initial connection state until the first POLL, and master slave switch). These PDUs may be sent at any time after connection setup is completed.

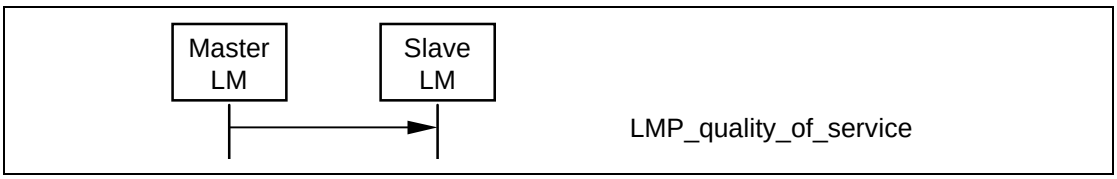
Master and slave negotiate the number of repetitions for broadcast packets (N_{BC}), see Baseband Specification, Section 7.6.5, on page 148.

M/O	PDU	Contents
M	LMP_quality_of_service	poll interval N_{BC}
M	LMP_quality_of_service_req	poll interval N_{BC}

Table 4.8: Quality of Service PDU

4.1.8.1 Master Notifies Slave of the Quality of Service

The master notifies the slave of the new poll interval and N_{BC} by sending the LMP_quality_of_service PDU. The slave cannot reject the notification.

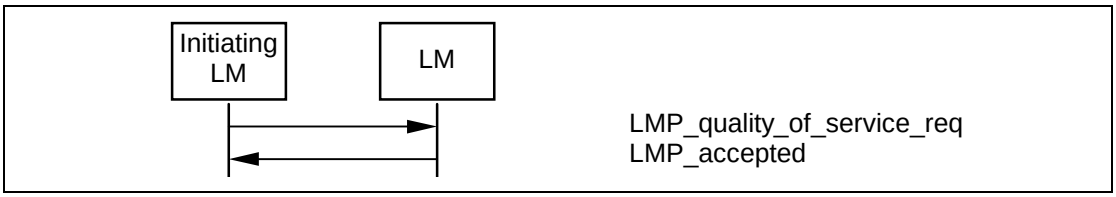


Sequence 13: Master notifies slave of quality of service.

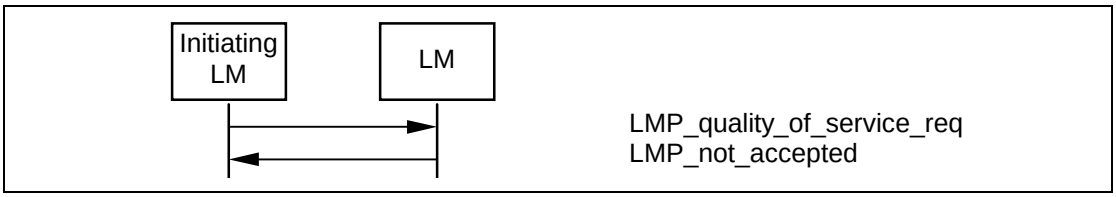
4.1.8.2 Device requests new quality of service

Either the master or the slave may request a new poll interval and N_{BC} by sending an LMP_quality_of_service_req PDU. The parameter N_{BC} is meaningful only when it is sent by a master to a slave. For transmission of LMP_quality_of_service_req PDUs from a slave, this parameter shall be ignored by the master. The request can be accepted or rejected. This allows the master and slave to dynamically negotiate the quality of service as needed.

The selected poll interval by the slave shall be less than or equal to the specified Access Latency for the outgoing traffic of the ACL link (see L2CAP "Quality of Service (QoS) Option" on page 82 in Vol. 3).



Sequence 14: Device accepts new quality of service



Sequence 15: Device rejects new quality of service.

4.1.9 Paging Scheme Parameters

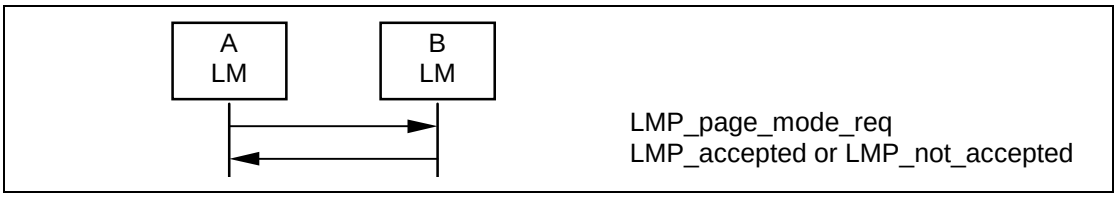
LMP provides a means to negotiate the paging scheme parameters that are used the next time a device is paged.

M/O	PDU	Contents
O(17)	LMP_page_mode_req	paging scheme paging scheme settings
O(17)	LMP_page_scan_mode_req	paging scheme paging scheme settings

Table 4.9: Paging scheme request PDU

4.1.9.1 Page Mode

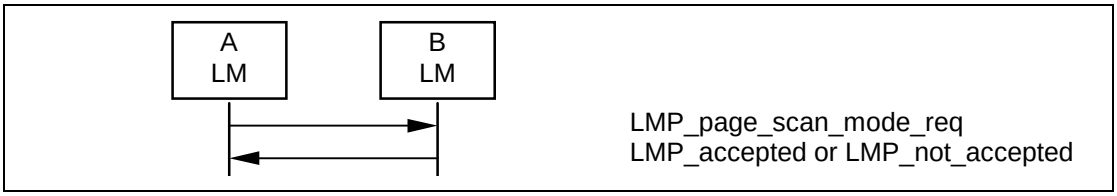
This procedure is initiated from device A and negotiates the paging scheme used when device A pages device B. Device A proposes a paging scheme including the parameters for this scheme and device B can accept or reject. On rejection the old setting shall not be changed. A request to switch to a reserved paging scheme shall be rejected.



Sequence 16: Negotiation for page mode.

4.1.9.2 Page Scan Mode

This procedure is initiated from device A and negotiates the paging scheme and paging scheme settings used when device B pages device A. Device A proposes a paging scheme and paging scheme settings and device B may accept or reject. On reject the old setting is not changed. A request specifying the mandatory scheme shall be accepted. A request specifying a non-mandatory scheme shall be rejected. This procedure should be used when device A changes its paging scheme settings. A slave should also send this message to the master after connection establishment, to inform the master of the slave's current paging scheme and paging scheme settings.



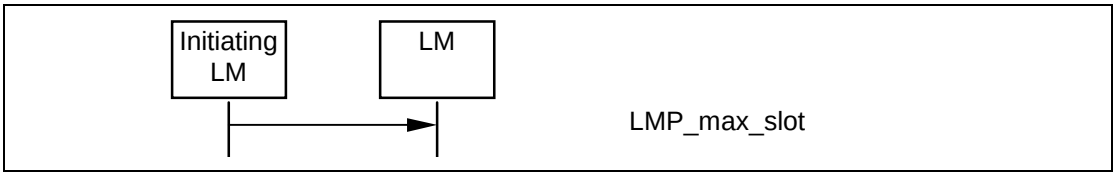
Sequence 17: Negotiation for page scan mode

4.1.10 Control of Multi-slot Packets

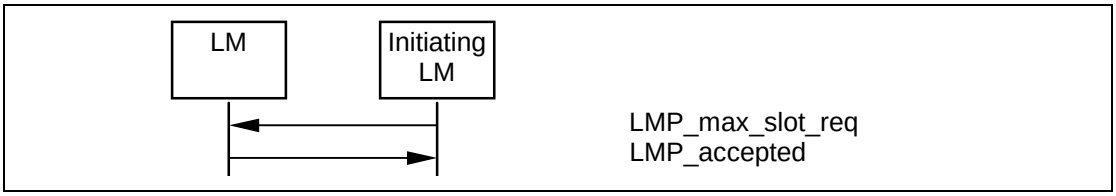
The number of consecutive slots used by a device on an ACL-U logical link can be limited. It does not affect traffic on the eSCO links where the packet sizes are defined as part of link setup. A device allows the remote device to use a maximum number of slots by sending the PDU LMP_max_slot providing max slots as parameter. Each device can request to use a maximal number of slots by sending the PDU LMP_max_slot_req providing max slots as parameter. After a new connection, as a result of page, page scan role switch or unpark, the default value is 1 slot. These PDUs can be sent at any time after connection setup is completed.

M/O	PDU	Contents
M	LMP_max_slot	max slots
M	LMP_max_slot_req	max slots

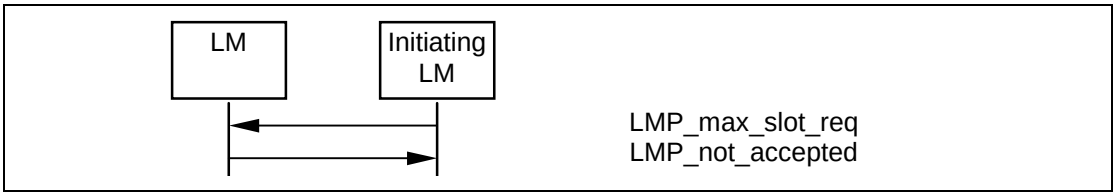
Table 4.10: Multi-slot packet control PDU



Sequence 18: Device allows Remote Device to use a maximum number of slots.



Sequence 19: Device requests a maximum number of slots. Remote Device accepts.



Sequence 20: Device requests a maximum number of slots. Remote Device rejects.

4.1.11 Enhanced Data Rate

A device may change the packet type table, ptt, to select which if any of the packets and optional modulation modes are to be used on an ACL logical transport.

Either the master or the slave may request a new packet type table and therefore the packets and modulation mode to be used on this ACL link. After a new

baseband connection, as a result of page or page scan, the default value for ptt shall be 0.

The change of the modulation mode for an ACL logical transport shall not affect the packet and packet types used for an associated SCO logical transport on the same LT_ADDR.

Note: Enhanced Data Rate eSCO links are negotiated using the LMP eSCO link_req as described in section 4.6.2.

Before changing the packet type table, the initiator shall finalize the transmission of the current ACL packet with ACL-U information and shall stop ACL-U transmissions. It shall then send the LMP_packet_type_table_req PDU.

If the receiver rejects the change, then it shall respond with an LMP_not_accepted_ext PDU.

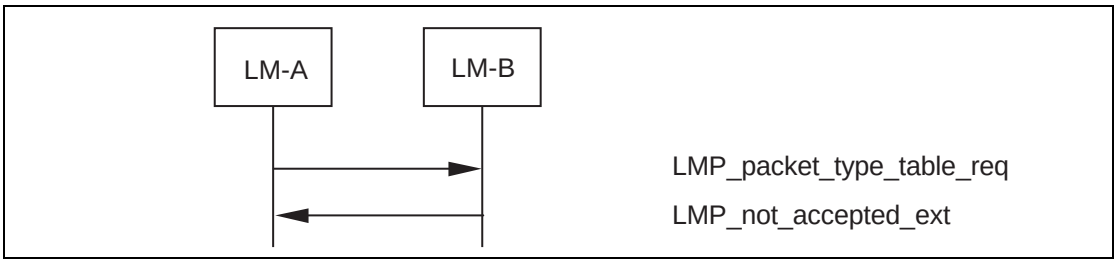
If the receiver accepts the change, then it shall finalize the transmission of the current ACL packet with ACL-U information and shall stop ACL-U transmissions, it shall change to the new packet type table and shall respond with an LMP_accepted_ext PDU. When it receives the baseband level acknowledgement for the LMP_accepted_ext PDU it shall restart ACL-U transmissions.

When the initiator receives an LMP_not_accepted_ext PDU the initiator shall restart ACL-U transmissions.

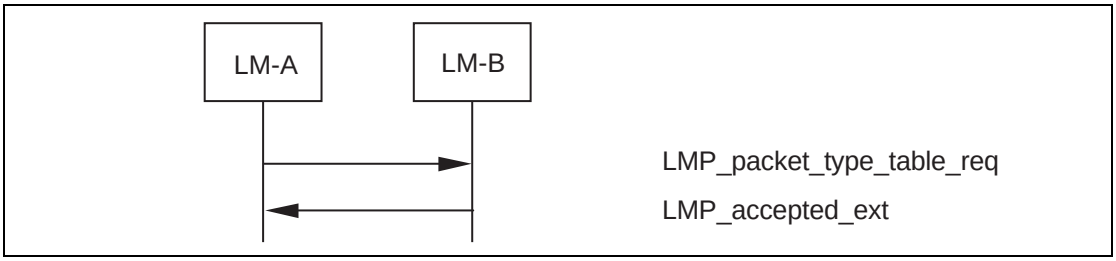
When the initiator receives an LMP_accepted_ext PDU it shall change the packet type table and restart ACL-U transmissions.

M/O	PDU	Contents
O(25)	LMP_packet_type_table_req	packet type table

Table 4.11: Enhanced Data Rate PDUs



Sequence 21: Packet type table change is rejected.



Sequence 22: Packet type table change is accepted.

4.1.12 Encapsulated LMP PDUs

Some transactions require sending LMP payload data that is longer than 16 octets. To enable a link manager to send a large PDU, an encapsulated LMP PDU is defined. An encapsulated LMP PDU is composed of a minimum of two LMP messages, a header PDU and one or more payload PDUs.

M/O	PDU	Contents
O(52)	LMP_encapsulated_header	encapsulated major type encapsulated minor type encapsulated payload length
O(52)	LMP_encapsulated_payload	encapsulated data

Table 4.12: Encapsulated LMP PDUs

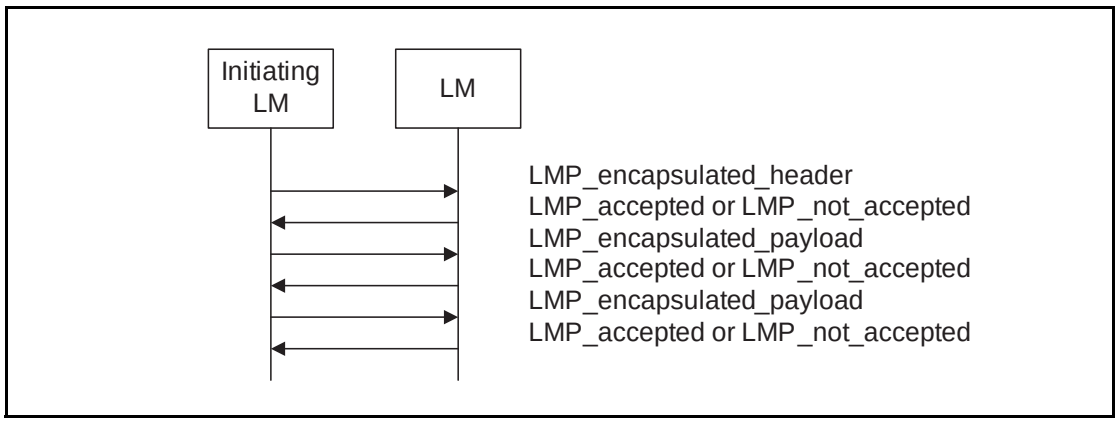
4.1.12.1 Sending an Encapsulated PDU

The LMP_encapsulated_header PDU shall be sent by the initiating device when it needs to send an encapsulated PDU. This PDU shall be either accepted or rejected using the LMP_accepted or LMP_not_accepted PDUs. If the major and minor types are not supported the PDU shall be rejected with error code Unsupported LMP Parameter Value. If the LMP_encapsulated_header PDU is accepted, then one or more LMP_encapsulated_payload PDUs will be sent with the encapsulated data sent in sequence, 16 octets at a time, or if this is last packet, the correct number of octets and then zero padded.

Each LMP_encapsulated_payload PDU shall be accepted or rejected. If the LMP_encapsulated_header PDU is rejected, then the opcode in the LMP_not_accepted PDU shall be the opcode for the LMP_encapsulated_header and not the encapsulated major type or encapsulated minor type. If the LMP_encapsulated_payload PDU is rejected, then the opcode in the LMP_not_accepted PDU shall be the opcode for the LMP_encapsulated_payload.

A responding device may reject the final LMP_encapsulated_payload PDU after accepting the LMP_encapsulated_header PDU. This is so that the link

manager can still reject an encapsulated message after all the data has been received.



Sequence 23: Sending an encapsulated PDU

Between sending an LMP_encapsulated_header PDU and an LMP_encapsulated_payload PDU, or between each of the LMP_encapsulated_payload PDU, either device shall be able to send the following LMP PDUs without causing a "different transaction collision" error.

- LMP_channel_classification
- LMP_decr_power_req
- LMP_detach
- LMP_incr_power_req
- LMP_max_power
- LMP_max_slot
- LMP_min_power
- LMP_preferred_rate
- LMP_set_AFH

4.2 SECURITY

4.2.1 Authentication

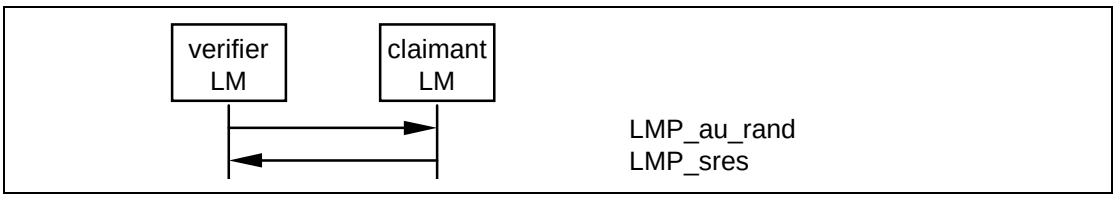
The authentication procedure is based on a challenge-response scheme as described in [Part H] Section 3.2.2 on page 1066. The verifier sends an LMP_au_rand PDU that contains a random number (the challenge) to the claimant. The claimant calculates a response, that is a function of this challenge, the claimant's BD_ADDR and a secret key. The response is sent back to the verifier, that checks if the response was correct or not. The response shall be calculated as described in [Part H] Section 6.1 on page 1085. A successful calculation of the authentication response requires that two devices share a secret key. This key is created as described in Section 4.2.2 on page 249. Both the master and the slave can be verifiers.

M/O	PDU	Contents
M	LMP_au_rand	random number
M	LMP_sres	authentication response

Table 4.13: Authentication PDUs

4.2.1.1 Claimant has Link Key

If the claimant has a link key associated with the verifier, it shall calculate the response and sends it to the verifier with LMP_sres. The verifier checks the response. If the response is not correct, the verifier can end the connection by sending an LMP_detach PDU with the error code *authentication failure,* see Section 4.1.2 on page 229.



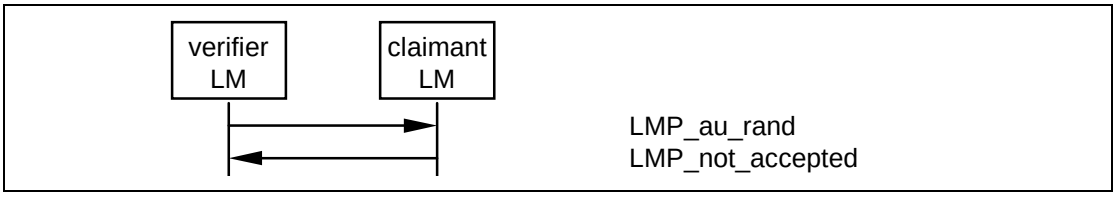
Sequence 24: Authentication. Claimant has link key.

Upon reception of an LMP_au_rand, an LM shall reply with LMP_sres before initiating its own authentication.

Note: There can be concurrent requests caused by the master and slave simultaneously initiating an authentication. The procedures in Section 2.5.1 on page 216 assures that devices will not have different Authenticated Ciphering Offset (ACO, see [Part H] Section 6.1 on page 1085) when they calculate the encryption key.

4.2.1.2 Claimant has no Link Key

If the claimant does not have a link key associated with the verifier it shall send an LMP_not_accepted PDU with the error code **PIN** 0x06 ([Part D] Section 2.6) or *key missing* after receiving an LMP_au_rand PDU.



Sequence 25: Authentication fails. Claimant has no link key.

4.2.1.3 Repeated Attempts

The scheme described in [Part H] Section 5.1 on page 1084 shall be applied when an authentication fails. This will prevent an intruder from trying a large number of keys in a relatively short time.

4.2.2 Pairing

When two devices do not have a common link key an initialization key (K_{init}) shall be created using either the pairing or Secure Simple Pairing procedures. When pairing is used, K_{init} shall be created based on a PIN, and a random number and a BD_ADDR. K_{init} shall be created as specified in [Part H] Section 6.3 on page 1089. When both devices have calculated K_{init} the link key shall be created, and a mutual authentication is performed. The pairing procedure starts with a device sending an LMP_in_rand PDU; this device is referred to as the "initiating LM" or "initiator" in Section 4.2.2.1 on page 249 - Section 4.2.2.5 on page 252. The other device is referred to as the "responding LM" or "responder". The PDUs used in the pairing procedure are:

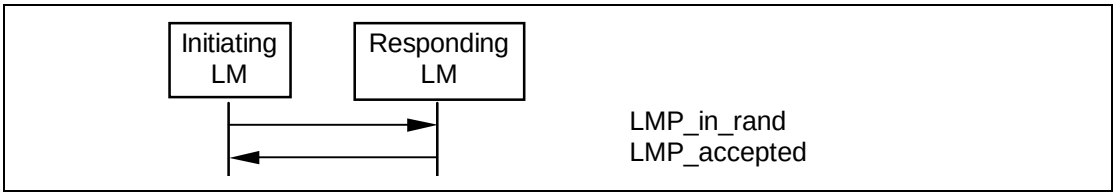
M/O	PDU	Contents
M	LMP_in_rand	random number
M	LMP_au_rand	random number
M	LMP_sres	authentication response
M	LMP_comb_key	random number
M	LMP_unit_key	key

Table 4.14: Pairing PDUs

All sequences described in Section 4 on page 228, including the mutual authentication after the link key has been created, shall form a single transaction. The transaction ID from the first LMP_in_rand shall be used for all subsequent sequences.

4.2.2.1 Responder Accepts Pairing

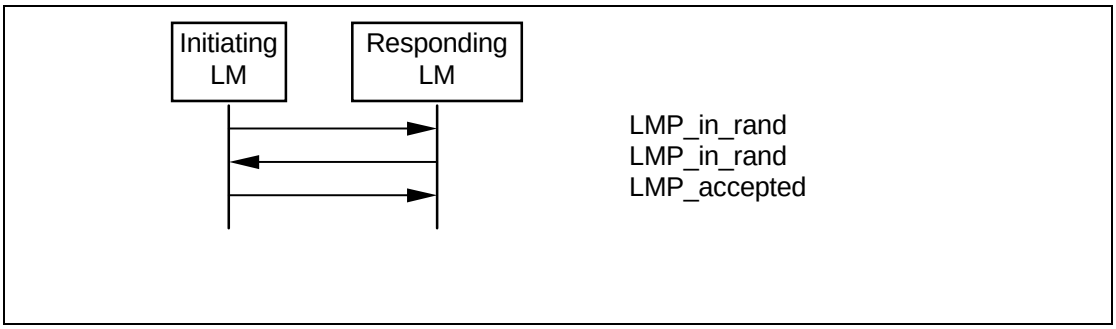
When the initiator sends an LMP_in_rand PDU and the responder shall reply with an LMP_accepted PDU. Both devices shall then calculate K_{init} based on the BD_ADDR of the responder and the procedure continues with creation of the link key; see Section 4.2.2.4 on page 251.



Sequence 26: Pairing accepted. Responder has a variable PIN. Initiator has a variable or fixed PIN.

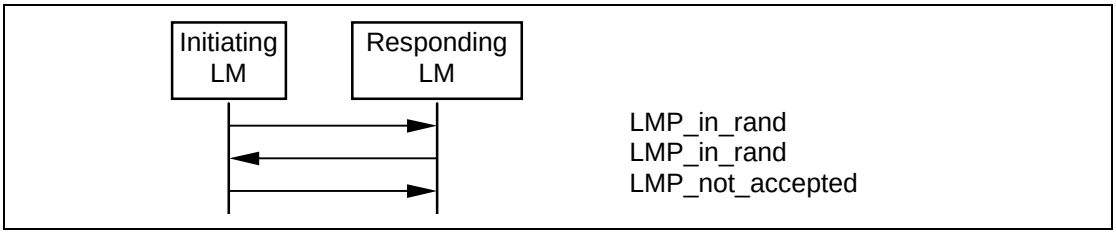
4.2.2.2 Responder has a Fixed PIN

If the responder has a fixed PIN it shall generate a new random number and send it back in an LMP_in_rand PDU. If the initiator has a variable PIN it shall accept the LMP_in_rand PDU and shall respond with an LMP_accepted PDU. Both sides shall then calculate K_{init} based on the last IN_RAND and the BD_ADDR of the initiator. The procedure continues with creation of the link key; see Section 4.2.2.4 on page 251.



Sequence 27: Responder has a fixed PIN and initiator has a variable PIN.

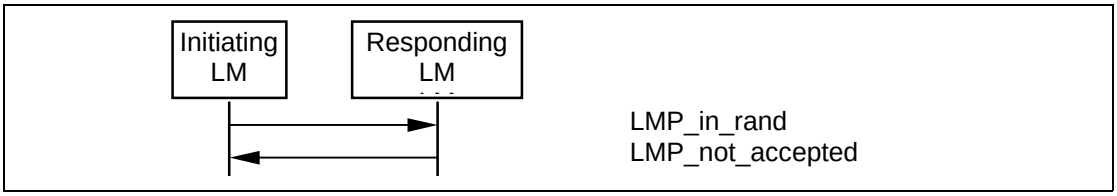
If the responder has a fixed PIN and the initiator also has a fixed PIN, the second LMP_in_rand shall be rejected by the initiator sending an LMP_not_accepted PDU with the error code *pairing not allowed*.



Sequence 28: Both devices have a fixed PIN.

4.2.2.3 Responder Rejects Pairing

If the responder rejects pairing it shall send an LMP_not_accepted PDU with the error code *pairing not allowed* after receiving an LMP_in_rand PDU.



Sequence 29: Responder rejects pairing.

4.2.2.4 Creation of the Link Key

When K_{init} is calculated in both devices the link key shall be created. This link key will be used in the authentication between the two devices for all subsequent connections until it is changed; see Section 4.2.3 on page 252 and Section 4.2.4 on page 253. The link key created in the pairing procedure will either be a combination key or one of the device's unit keys. The following rules shall apply to the selection of the link key:

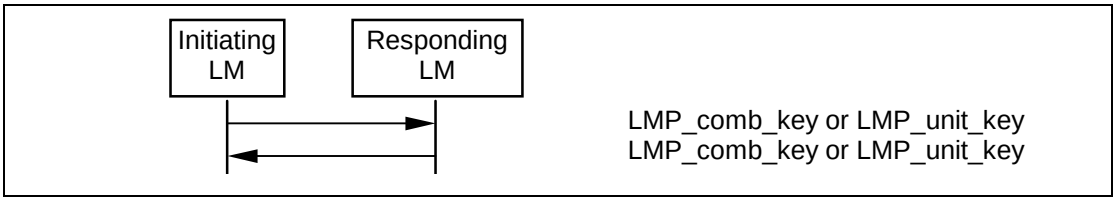
- if one device sends an LMP_unit_key PDU and the other device sends LMP_comb_key, the unit key will be the link key.
- if both devices send an LMP_unit_key PDU, the master's unit key will be the link key.
- if both devices send an LMP_comb_key PDU, the link key shall be calculated as described in [Part H] Section 3.2 on page 1065.

The content of the LMP_unit_key PDU is the unit key bitwise XORed with K_{init}. The content of the LMP_comb_key PDU is LK_RAND bitwise XORed with K_{init}. Any device configured to use a combination key shall store the link key.

The use of unit keys is deprecated since it is implicitly insecure.

When the new link key has been created mutual authentication shall be performed to confirm that the same link key has been created in both devices. After mutual authentication, the initiating device shall pause and immediately resume encryption to produce a new encryption key. Note that this will cause a new encryption key to be generated from the ACO created during the mutual authentication process, and the random number sent in the LMP_start_encryption_req PDU which occurs in response to the resumption of encryption.

After mutual authentication, if encryption is enabled, the initiating device shall pause and immediately resume encryption to produce a new encryption key. Note that this will cause a new encryption key to be generated from the ACO created during the mutual authentication process, and the random number sent in the LMP_start_encryption_req PDU which occurs in response to the resumption of encryption.



Sequence 30: Creation of the link key.

4.2.2.5 Repeated Attempts

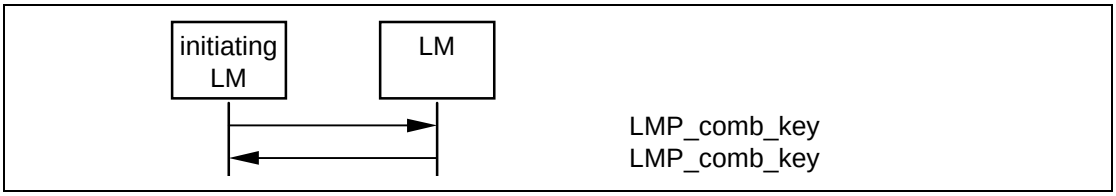
When the authentication after creation of the link key fails because of an incorrect authentication response, the same scheme as in Section 4.2.1.3 on page 249 shall be used. This prevents an intruder from trying a large number of different PINs in a relatively short time.

4.2.3 Change Link Key

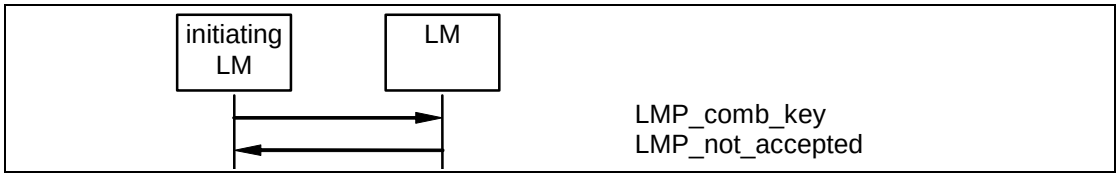
If the link key is derived from combination keys and the current link is the semi-permanent link key, the link key can be changed. If the link key is a unit key, the devices shall go through the pairing procedure in order to change the link key. The contents of the LMP_comb_key PDU is protected by a bitwise XOR with the current link key.

M/O	PDU	Contents
M	LMP_comb_key	random number

Table 4.15: Change link key PDU



Sequence 31: Successful change of the link key



Sequence 32: Change of the link key not possible since the other device uses a unit key.

If the change of link key is successful the new link key shall be stored and the old link key shall be discarded. The new link key shall be used as link key for all the following connections between the two devices until the link key is changed again. The new link key also becomes the current link key. It will remain the

current link key until the link key is changed again, or until a temporary link key is created, see Section 4.2.4 on page 253.

When the new link key has been created mutual authentication shall be performed to confirm that the same link key has been created in both devices. After mutual authentication, if encryption is enabled, the initiating device shall pause and immediately resume encryption to produce a new encryption key. Note that this will cause a new encryption key to be generated from the ACO created during the mutual authentication process, and the random number sent in the LMP_start_encryption_req PDU which occurs in response to the resumption of encryption. The first authentication in the mutual authentication is performed with the device that initiated change link key as verifier. When finalized an authentication in the reverse direction is performed.

4.2.4 Change Current Link Key Type

The current link key can be a semi-permanent link key or a temporary link key. It may be changed temporarily, but the change shall only be valid for the current connection, see [Part H] Section 3.1 on page 1063. Changing to a temporary link key is necessary if the piconet is to support encrypted broadcast. The current link key may not be changed before the connection establishment procedure has completed. This feature is only supported if broadcast encryption is supported as indicated by the LMP features mask.

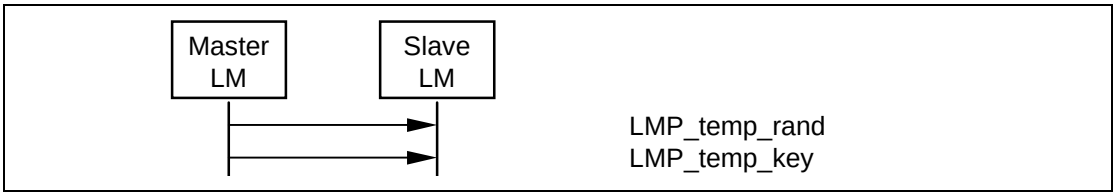
M/O	PDU	Contents
O(23)	LMP_temp_rand	random number
O(23)	LMP_temp_key	key
O(23)	LMP_use_semi_permanent_key	-

Table 4.16: Change current link key PDU

4.2.4.1 Change to a Temporary Link Key

The master starts by creating the master key K_{master} as specified in Security Specification (EQ 4), on page 1070. Then the master shall generate a random number, RAND, and shall send it to the slave in an LMP_temp_rand PDU. Both sides then calculate an overlay denoted OVL as OVL= E_{22}(current link key, RAND, 16). The master shall then send K_{master} protected by a modulo-2 addition with OVL to the slave in an LMP_temp_key PDU. The slave calculates K_{master}, based on OVL, that becomes the current link key. It shall be the current link key until the devices fall back to the semi-permanent link key, see Semi-permanent Link Key Becomes Current Link Key on page 254.

Note: the terminology in this section is the same as used in [Part H] Section 3.2.8 on page 1070.



Sequence 33: Change to a temporary link key

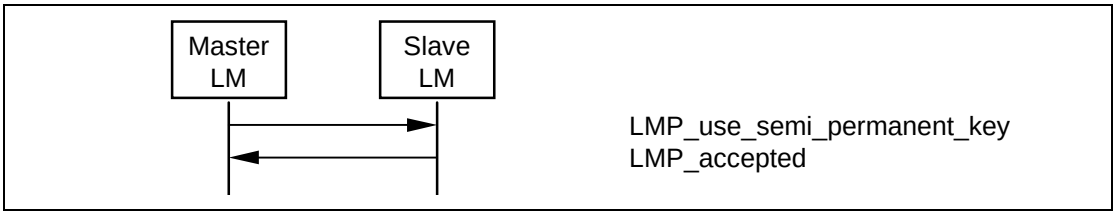
All sequences described in Section 4.2.4.1 on page 253, including the mutual authentication after K_{master} has been created, shall form a single transaction. The transaction ID shall be set to 0.

When the devices have changed to the temporary key, a mutual authentication shall be made to confirm that the same link key has been created in both devices. The first authentication in the mutual authentication shall be performed with the master as verifier. When finalized an authentication in the reverse direction is performed.

Should the mutual authentication fail at either side, the LM of the verifier should start the detach procedure. This will allow the procedure to succeed even though one of the devices may be erroneous.

4.2.4.2 Semi-permanent Link Key Becomes Current Link Key

After the current link key has been changed to K_{master}, this change can be undone and the semi-permanent link key becomes the current link key again. If encryption is used on the link, the procedure to go back to the semi-permanent link key shall be immediately followed by the master stopping encryption using the procedure described in Section 4.2.5.4 on page 258. Encryption may be restarted by the master according to the procedures in Encryption Mode on page 255 subsection 3. This is to assure that encryption with encryption parameters known by other devices in the piconet is not used when the semi-permanent link key is the current link key.



Sequence 34: Link key changed to the semi-permanent link key

4.2.5 Encryption

If at least one authentication has been performed encryption may be used. In order for the master to use the same encryption parameters for all slaves in the piconet it shall issue a temporary key, K_{master}. The master shall make this key the current link key for all slaves in the piconet before encryption is started, see Section 4.2.4 on page 253. This is required if broadcast packets are to be encrypted.

M/O	PDU	Contents
O	LMP_encryption_mode_req	encryption mode
O	LMP_encryption_key_size_req	key size
O	LMP_start_encryption_req	random number
O	LMP_stop_encryption_req	-
O (42)	LMP_pause_encryption_req	
O (42)	LMP_resume_encryption_req	

Table 4.17: Encryption handling PDU

All sequences described in Section 4.2.5 shall form a single transaction. The transaction ID from the LMP_encryption_mode_req PDU shall be used for all start encryption and stop encryption sequences.

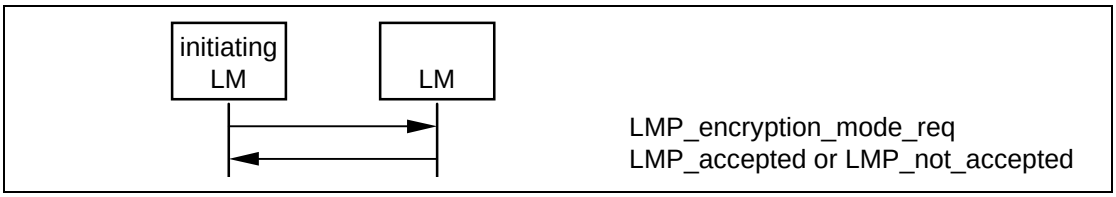
4.2.5.1 Encryption Mode

The master and the slave must agree upon whether to use encryption (encryption mode=1 in LMP_encryption_mode_req) or not (encryption mode=0). If the semi-permanent key is used (Key_Flag=0x00) encryption shall only apply to point-to-point packets. If the master link key is used (Key_Flag=0x01) encryption shall apply to both point-to-point packets and broadcast packets. If master and slave agree on the encryption mode, the master continues to give more detailed information about the encryption.

Devices should never send LMP_encryption_mode_req with an encryption mode value of 2 however for backwards compatibility if the LMP_encryption_mode_req is received with an encryption mode value of 2 then it should be treated the same as an encryption mode value of 1.

The initiating LM shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107). The initiating device shall then send the LMP_encryption_mode_req PDU. If the responding device accepts the change in encryption mode then it shall complete the transmission of the current packet on the ACL logical transport and shall then suspend transmission on the ACL-U logical link. The responding device shall then send the LMP_accepted PDU.

ACL-U logical link traffic shall only be resumed after the attempt to encrypt or decrypt the logical transport is completed i.e. at the end of Sequence 31 (on failure) 34, 35, 36.

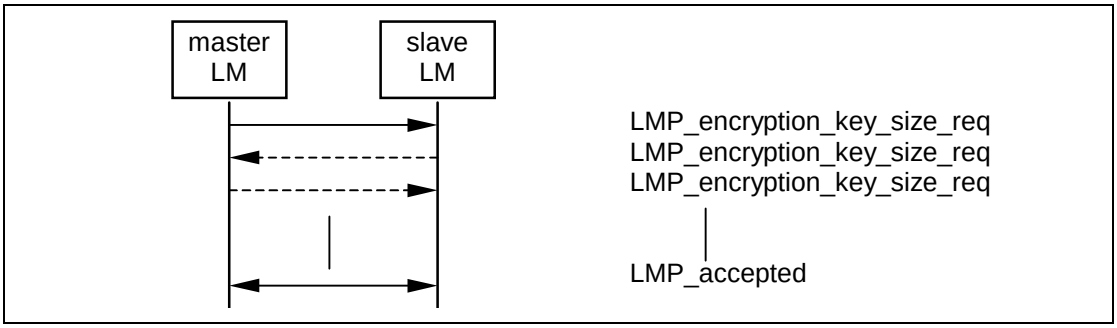


Sequence 35: Negotiation for encryption mode

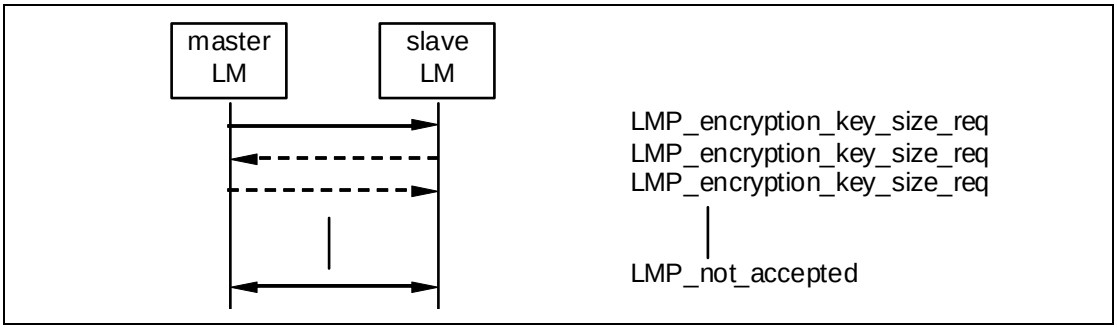
After a device has sent an LMP_encryption_mode_req PDU it shall not send an LMP_au_rand PDU before encryption is started. After a device has received an LMP_encryption_mode_req PDU and sent an LMP_accepted PDU it shall not send an LMP_au_rand PDU before encryption is started. If an LMP_au_rand PDU is sent violating these rules, the claimant shall respond with an LMP_not_accepted PDU with the error code *PDU not allowed*. This assures that devices will not have different ACOs when they calculate the encryption key. If the encryption mode is not accepted or the encryption key size negotiation results in disagreement the devices may send an LMP_au_rand PDU again.

4.2.5.2 Encryption Key Size

Note: this section uses the same terms as in [Part H] Section 4.1 on page 1073. The master sends an LMP_encryption_key_size_req PDU including the suggested key size L_{sug}, m, that is initially equal to $L_{max, m}$. If $L_{min, s} \leq L_{sug, m}$ and the slave supports $L_{sug, m}$ it shall respond with an LMP_accepted PDU and $L_{sug, m}$ shall be used as the key size. If both conditions are not fulfilled the slave sends back an LMP_encryption_key_size_req PDU including the slave's suggested key size $L_{sug, s}$. This value shall be the slave's largest supported key size that is less than $L_{sug, m}$. Then the master performs the corresponding test on the slave's suggestion. This procedure is repeated until a key size agreement is reached or it becomes clear that no such agreement can be reached. If an agreement is reached a device sends an LMP_accepted PDU and the key size in the last LMP_encryption_key_size_req PDU shall be used. After this, encryption is started; see Section 4.2.5.3 on page 257. If an agreement is not reached a device sends an LMP_not_accepted PDU with the error code *unsupported parameter value* and the devices shall not communicate using encryption.



Sequence 36: Encryption key size negotiation successful

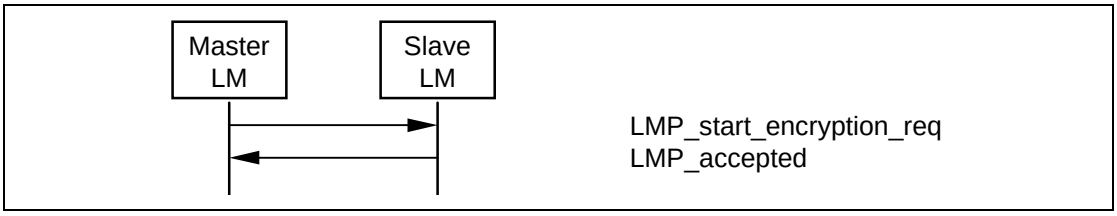


Sequence 37: Encryption key size negotiation failed

4.2.5.3 Start Encryption

To start encryption, the master issues the random number EN_RAND and calculates the encryption key. See [Part H] Section 3.2.5 on page 1068. The random number shall be the same for all slaves in the piconet when broadcast encryption is used. The master then sends an LMP_start_encryption_req PDU, that includes EN_RAND. The slave shall calculate the encryption key when this message is received and shall acknowledge with an LMP_accepted PDU.

If encryption has been paused, then this sequence shall not be used.



Sequence 38: Start of encryption

Starting encryption shall be performed in three steps:

1. Master is configured to transmit unencrypted packets and to receive encrypted packets.
2. Slave is configured to transmit and receive encrypted packets.
3. Master is configured to transmit and receive encrypted packets.

Between step 1 and step 2, master-to-slave transmission is possible. This is when an LMP_start_encryption_req PDU is transmitted. Step 2 is triggered when the slave receives this message. Between step 2 and step 3, slave-to-master transmission is possible. This is when an LMP_accepted PDU is transmitted. Step 3 is triggered when the master receives this message.

4.2.5.4 Stop Encryption

To stop encryption a device shall send an LMP_encryption_mode_req PDU with the parameter encryption mode equal to 0 (no encryption). The other device responds with an LMP_accepted PDU or an LMP_not_accepted PDU (the procedure is described in Sequence 35 in Encryption Mode on page 255). If accepted, encryption shall be stopped by the master sending an LMP_stop_encryption_req PDU and the slave shall respond with an LMP_accepted PDU according to Sequence 39.

If encryption has been paused, then this sequence shall not be used.



Sequence 39: Stop of encryption

Stopping encryption shall be performed in three steps, similar to the procedure for starting encryption.

1. Master is configured to transmit encrypted packets and to receive unencrypted packets.
2. Slave is configured to transmit and receive unencrypted packets.
3. Master is configured to transmit and receive unencrypted packets.

Between step 1 and step 2 master to slave transmission is possible. This is when an LMP_stop_encryption_req PDU is transmitted. Step 2 is triggered when the slave receives this message. Between step 2 and step 3 slave to master transmission is possible. This is when an LMP_accepted PDU is transmitted. Step 3 is triggered when the master receives this message.

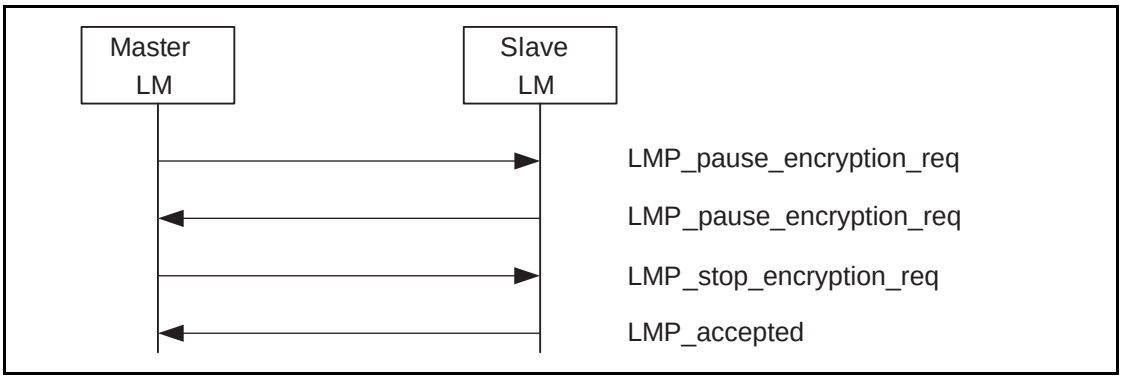
4.2.5.5 Pause Encryption

To pause encryption without disabling encryption, a device shall finalize the transmission of the current ACL-U data packet and then send an LMP_pause_encryption_req PDU (with the transaction ID set to the role of the device at the time the LMP_pause_encryption_req PDU is sent).

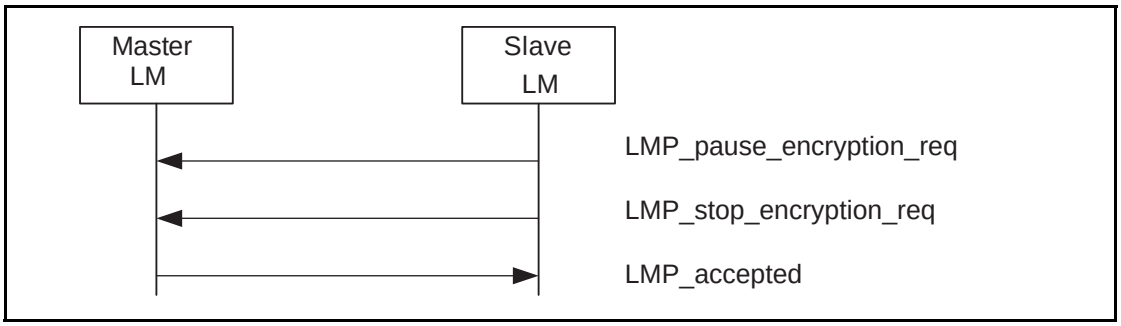
If the responding device is a master, then the master device shall finalize the transmission of the current ACL-U data packet and then respond with an LMP_stop_encryption_req PDU to the slave.

If the responding device is a slave, then the slave device shall finalize the transmission of the current ACL-U data packet and then respond with an LMP_pause_encryption_req PDU. The master device shall respond to the LMP_pause_encryption_req PDU with an LMP_stop_encryption_req PDU to the slave.

When the slave receives the LMP_stop_encryption_req PDU it shall respond with an LMP_accepted PDU.



Sequence 40: Master-initiated pausing of encryption



Sequence 41: Slave-initiated pausing of encryption

The LMP_pause_encryption_req PDU and LMP_stop_encryption_req PDU shall be rejected only when transaction collision needs to be resolved.

Pausing encryption shall be performed in three steps, similar to the procedure for stopping encryption.

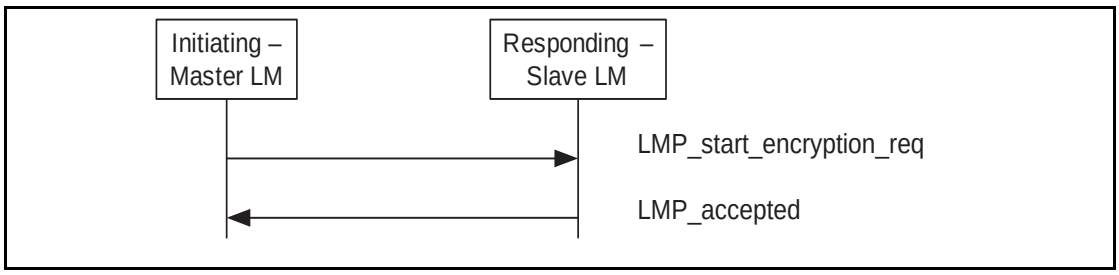
1. Master is configured to transmit encrypted packets and to receive unencrypted packets.
2. Slave is configured to transmit and receive unencrypted packets.
3. Master is configured to transmit and receive unencrypted packets.

Between step 1 and step 2 master-to-slave transmission is possible. This is when the LMP_stop_encryption_req PDU is transmitted from the master to the

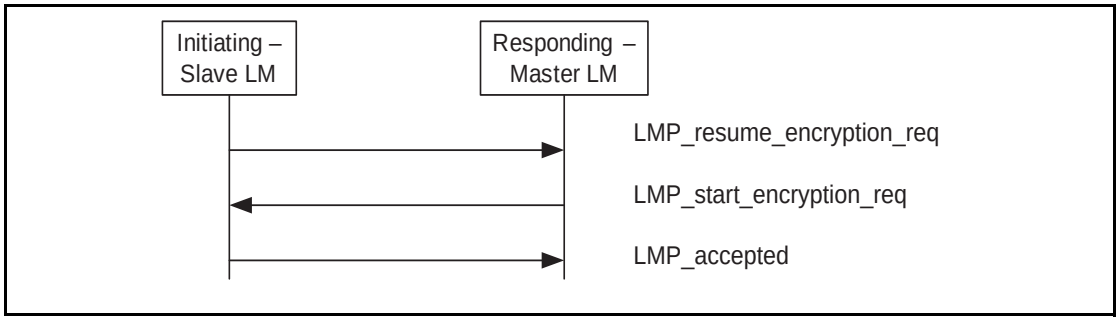
slave. Step 2 is triggered when the slave receives this message. Between step 2 and step 3 slave-to-master transmission is possible. This is when an LMP_accepted PDU is transmitted. Step 3 is triggered when the master receives this message.

Note: A device can only restart ACL-U data traffic by resuming encryption using the procedures in Section 4.2.5.6.

4.2.5.6 Resume Encryption



Sequence 42: Initiating Master LM resumes encryption



Sequence 43: Initiating Slave LM resumes encryption

If the responding device is a slave, then the slave shall calculate the encryption key and respond with an LMP_accepted PDU.

If the responding device is a master, then the master shall respond with an LMP_start_encryption_req PDU. The slave, upon receiving the LMP_start_encryption_req PDU from the master, shall calculate the encryption key and respond with an LMP_accepted PDU.

The LMP_resume_encryption_req PDU and the LMP_start_encryption_req PDU shall not be rejected.

Resuming encryption shall be performed in three steps, similar to the procedure for starting encryption:

1. Master is configured to transmit unencrypted packets and to receive encrypted packets.
2. Slave is configured to transmit and receive encrypted packets.
3. Master is configured to transmit and receive encrypted packets.

Between step 1 and step 2, master-to-slave transmission is possible. This is when the LMP_start_encryption_req PDU is transmitted from the master. Step 2 is triggered when the slave receives this message. Between step 2 and step 3, slave-to-master transmission is possible. This is when an LMP_accepted PDU is transmitted. Step 3 is triggered when the master receives this message.

Note: For a slave initiated starting of encryption, step 1 is not started when the master has received the LMP_resume_encryption_req PDU from the slave, but when the master sends the LMP_start_encryption_req PDU.

4.2.5.7 Change Encryption Mode, Key or Random Number

If the encryption key or encryption random number need to be changed or if the current link key needs to be changed according to the procedures in Section 4.2.4 on page 253, encryption shall be paused and resumed after completion, using the procedures in Section 4.2.5 on page 255, Section 4.2.5.1 and Section 4.2.5.2, and Section 4.2.3 on page 252 for the new parameters to be valid. If the Pause Encryption feature is not supported by both devices, encryption shall be stopped and re-started after completion, using the procedures in Section 4.2.5 on page 255, subsections 3-4 for the new parameters to be valid.

4.2.5.8 Encryption Key Refresh

The Link Manager shall refresh the encryption key within 2^{28} ticks of the Bluetooth clock from the previous start or resume of encryption. To refresh the encryption key, the Link Manager shall Pause encryption using the procedure in Section 4.2.5.5 on page 258 and immediately Resume encryption using the procedure in Section 4.2.5.6 on page 260.

4.2.6 Request Supported Encryption Key Size

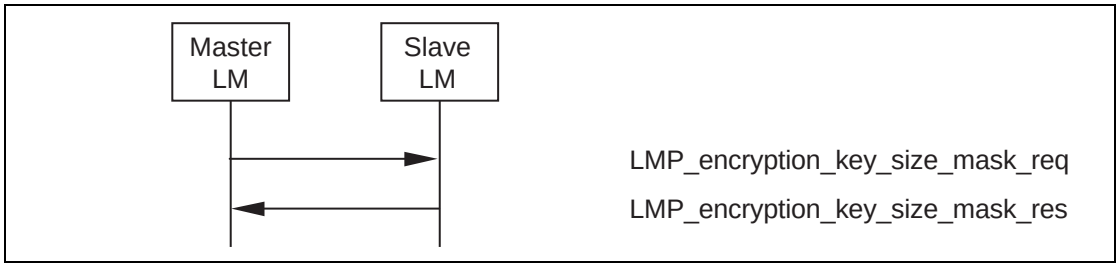
When broadcast encryption is supported via the LMP features mask, it is possible for the master to request a slave's supported encryption key sizes.

M/O	PDU	Contents
O(23)	LMP_encryption_key_size_mask_req	
O(23)	LMP_encryption_key_size_mask_res	key size mask

Table 4.18: Encryption key size request PDU

The master shall send an LMP_encryption_key_mask_req PDU to the slave to obtain the slaves supported encryption key sizes.

The slave shall return a bit mask indicating all broadcast encryption key sizes supported. The least significant bit shall indicate support for a key size of 1, the next most significant bit shall indicate support for a key size of 2 and so on up to a key size of 16. In all cases a bit set to 1 shall indicate support for a key size; a bit set to 0 shall indicate that the key size is not supported.



Sequence 44: Request for supported encryption key sizes.

4.2.7 Secure Simple Pairing

There are four stages defined in the Secure Simple Pairing LM process:

- IO capabilities exchange
- Public key exchange
- Authentication stage 1
- Authentication stage 2

The devices shall first exchange the IO capabilities to determine the proper algorithm to be used. Three algorithms have been specified: Numeric comparison, Passkey entry, Out of band.

In following sections, the device requesting the IO capabilities is referred to as the "Initiating LM" or "Initiator". The other device is referred to as the "LM" or "Responder". This designation remains throughout the entire Secure Simple Pairing procedure.

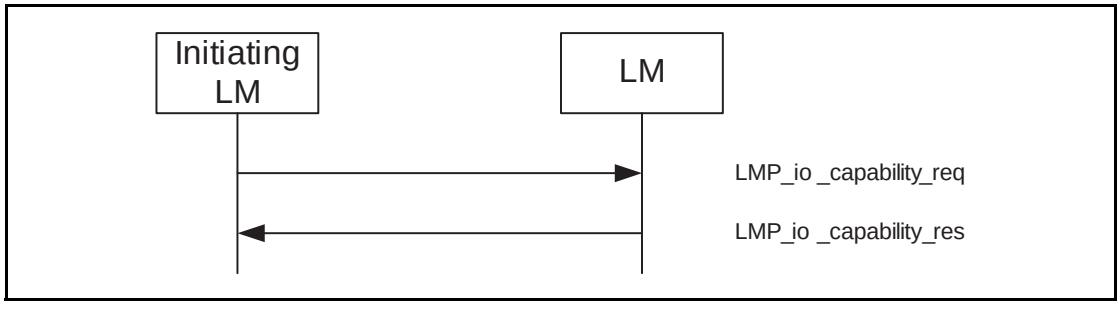
Note that the host shall enable Secure Simple Pairing before the LMP Secure Simple Pairing procedures start.

M/O	PDU	Contents
O(51)	LMP_io_capability_req	IO_capabilities, OOB_Authentication_Data, Authentication_Requirements
O(51)	LMP_io_capability_res	IO_capabilities, OOB_Authentication_Data, Authentication_Requirements
O(51)	LMP_Simple_Pairing_Confirm	Commitment Value
O(51)	LMP_Simple_Pairing_Number	Nonce Value
O(51)	LMP_Dhkey_check	Confirmation value
O(51)	LMP_Numeric_Comparison_Failed	-
O(51)	LMP_OOB_Failed	-
O(51)	LMP_keypress_notification	Notification Type
O(51)	LMP_Passkey_Entry_Failed	-

4.2.7.1 IO Capability Exchange

The Link Managers shall use the local and remote IO capabilities to determine which authentication method shall be used.

If Simple_Pairing_Mode is set to enabled, the Initiator shall request the IO capabilities from the Host. The initiator shall send an LMP_io_capability_req PDU to the Responder. If Simple_Pairing_Mode is set to enabled on the responding device, it shall reply with an LMP_io_capability_res PDU containing its IO capabilities description.



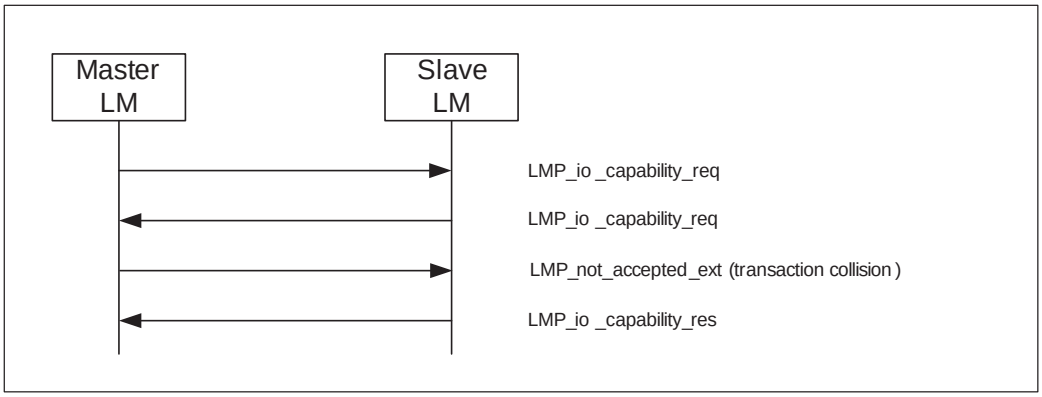
Sequence 45: IO Capability Exchange

4.2.7.1.1 IO Capability Exchange Transaction Collision and Resolution

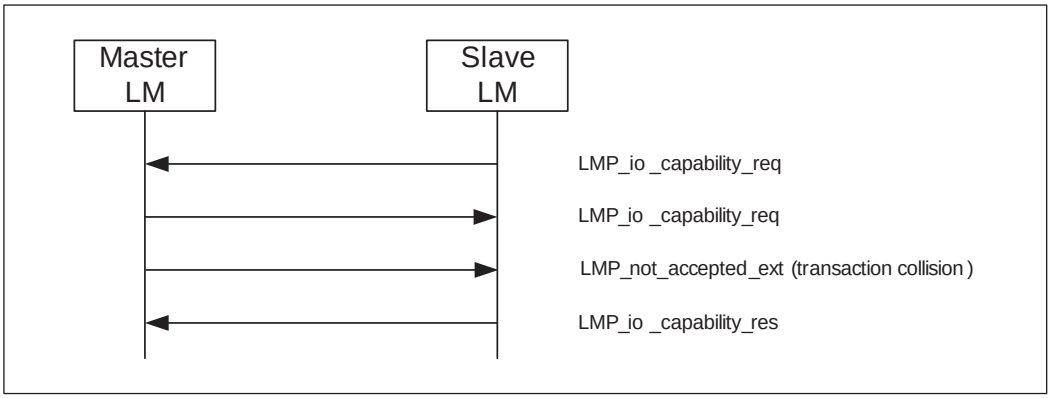
If both Link Managers attempt to become the initiator of the IO capability exchange, the master Link Manager shall send an LMP_not_accepted_ext

PDU with error code 'Transaction Collision'. The slave Link Manager shall respond with the LMP_io_capability_res PDU.

The master LM shall remain the initiating LM for the remainder of the Secure Simple Pairing sequences.



Sequence 46: IO Capability exchange with transaction Collision (master transmitting LMP_io_capability_req first) and resolution



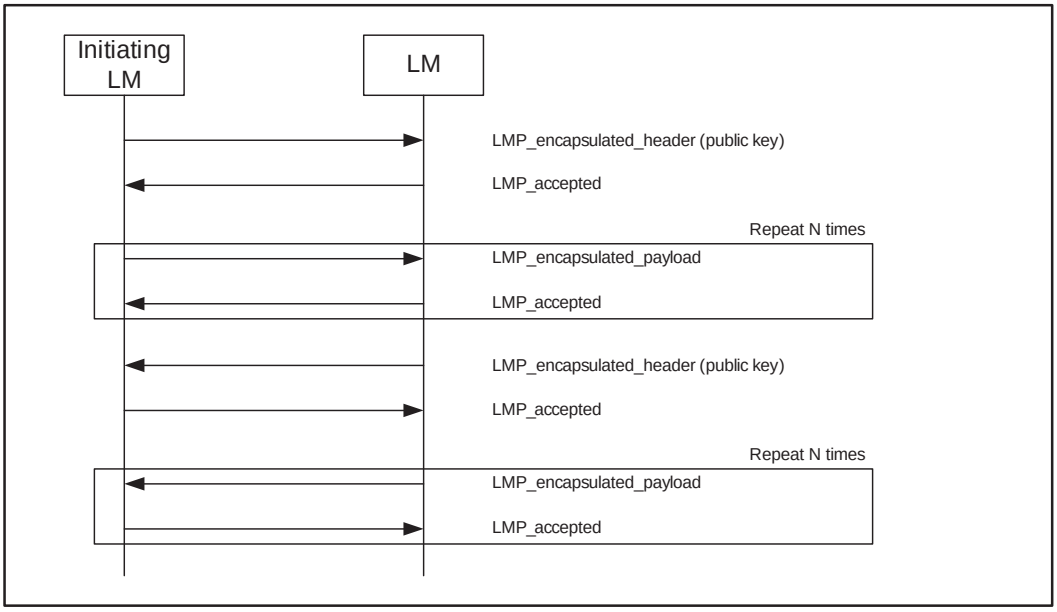
Sequence 47: IO Capability exchange with transaction Collision (slave transmitting LMP_io_capability_req first) and resolution

4.2.7.2 Public Key Exchange

Once the IO capabilities are exchanged, public keys shall be exchanged between the two devices.

Since the public key size is 48 bytes long, the exchange shall be done using the LMP LMP_encapsulated_header and LMP_encapsulated_payload PDUs as defined in Section 4.1.12.

The Initiator shall first send its public key, and the Responder shall reply with its public key.



Sequence 48: Public key exchange

A public key shall be considered as received when the last LMP_encapsulated_payload has been received and the associated LMP_accepted PDU has been sent.

The device can then start computing its Diffie Hellman Key.

4.2.7.3 Authentication Stage 1

One of the following procedures shall be used in Authentication Stage 1:

- If one or both devices have the OOB Authentication Data parameter set to Received, the Out-of-Band procedure shall be used.
- If both devices have the Authentication_Requirements parameter set to one of the man-in-the middle (MITM) Protection Not Required options, the Numeric Comparison procedure shall be used.
- If one or both devices have the Authentication_Requirements parameter set to one of the MITM Protection Required options, the Passkey Entry procedure shall be used if the either the local or remote IO Capability is set to KeyboardOnly and the other IO capability is not set to NoInputNoOutput. Otherwise, the Numeric Comparison authentication procedure shall be used.

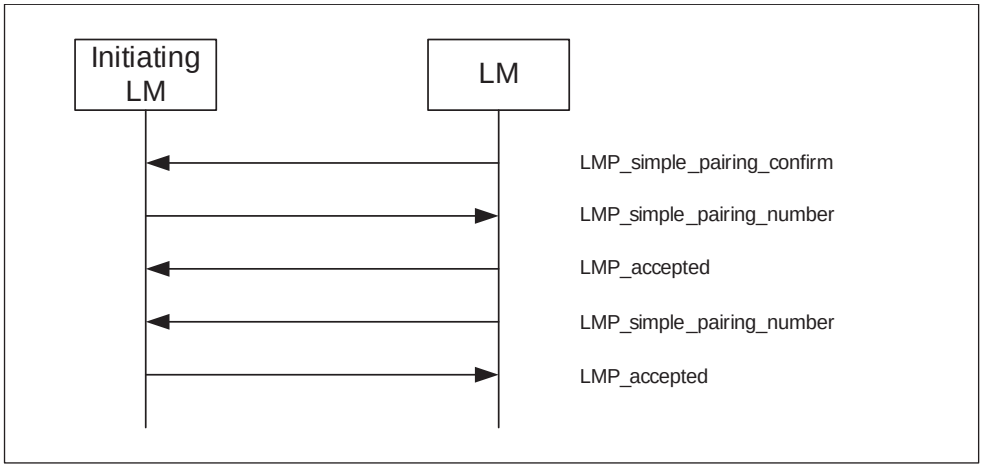
4.2.7.3.1 Authentication Stage 1: Numeric Comparison

Once public keys have been exchanged, both devices shall generate a random number.

The Responder shall compute its commitment as defined in Section 7.7.1 on page 1102, and shall send this value to the Initiator by using LMP_simple_pairing_confirm PDU.

The Initiator shall then send an LMP_simple_pairing_number PDU with its generated random number. The Responder shall acknowledge by sending an LMP_accepted PDU.

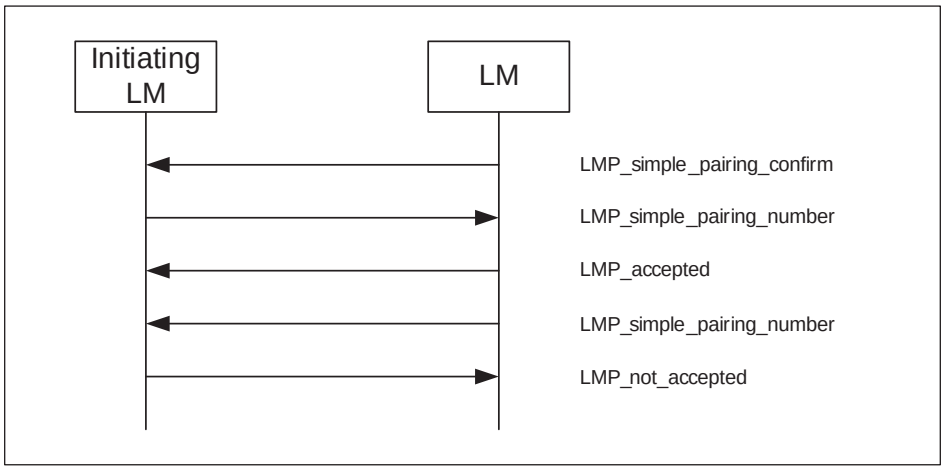
The Responder shall then send an LMP_simple_pairing_number containing its own generated random number. Upon reception, the Initiator shall calculate the commitment as defined in Section 7.7.1 on page 1102 and compare it to the one received previously with the LMP_simple_pairing_confirm PDU. If both values are equal, the Initiator shall respond with an LMP_accepted PDU.



Sequence 49: Numeric Comparison Authentication: Commitment check success

4.2.7.3.1.1 Commitment Check Failure

If the calculated commitment by the Initiator is not equal to the received commitment, the Initiator shall abort the Secure Simple Pairing process by sending an LMP_not_accepted PDU with reason "Authentication Failure."

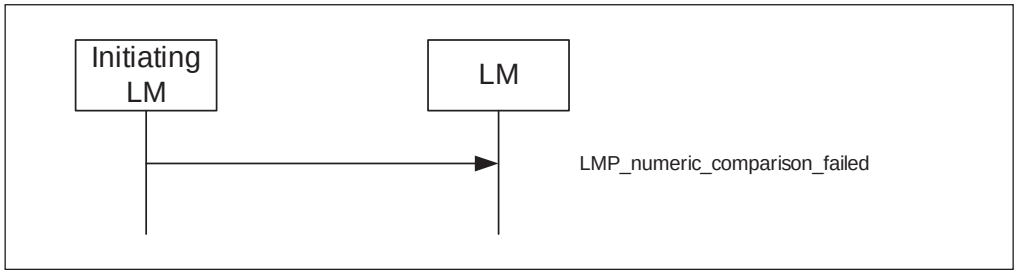


Sequence 50: Numeric Comparison Authentication: Commitment check failure

4.2.7.3.1.2 Numeric Comparison Failure on Initiator Side

If the user on the initiating side indicates that the confirm values do not match (as indicated by the HCI_User_Confirmation_Negative_Reply command) the initiating LM shall send an LMP_numeric_comparison_failure PDU.

Secure Simple Pairing process shall then be aborted. The Link Managers shall not disconnect the connection.

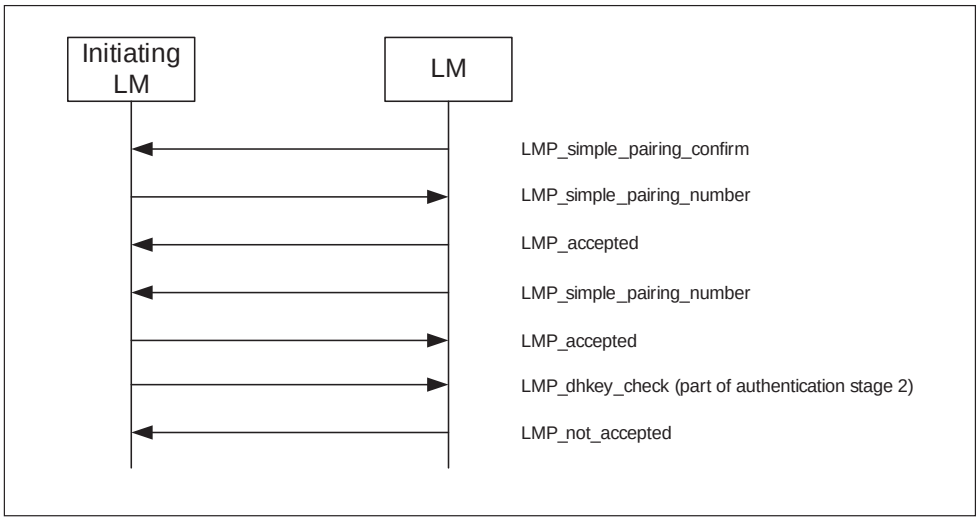


Sequence 51: Authentication stage 1 Numeric Comparison Failure on Initiator Side

4.2.7.3.1.3 Numeric Comparison failure on Responder Side

If the user on the responding side indicates that the confirm values do not match (as indicated by the HCI_User_Confirmation_Negative_Reply command) the responding LM shall send an LMP_not_accepted PDU in response to the LMP_dhkey_check PDU.

Simple Pairing process shall then be aborted. The Link Managers shall not disconnect the connection.

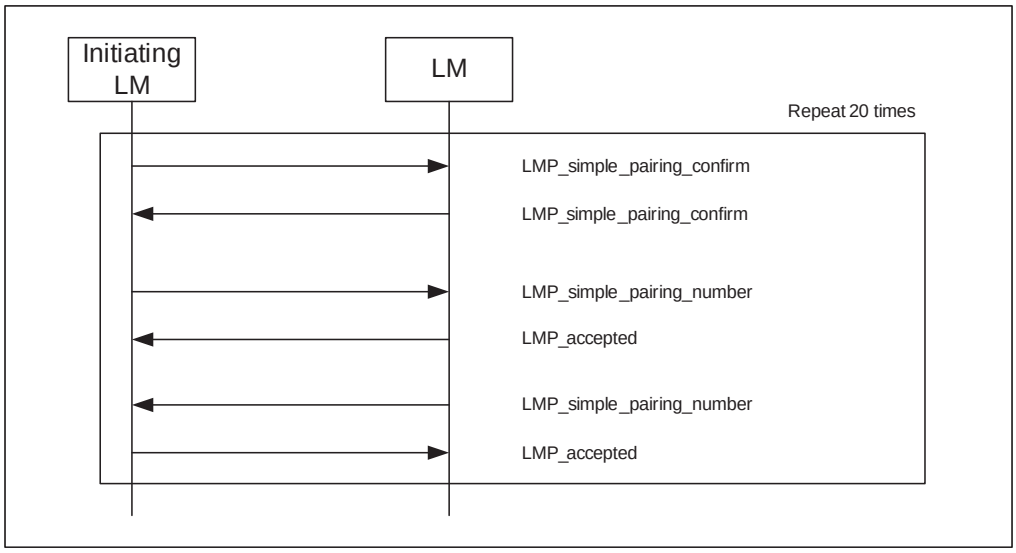


Sequence 52: Authentication stage 1: Numeric Comparison Failure on Responder Side

4.2.7.3.2 Authentication Stage 1: Passkey Entry Authentication

The initiating LM shall start the following procedure after receiving the Passkey Reply from the host. This procedure shall be repeated 20 times:

1. The Initiator and the Responder shall generate a new random number.
2. The Initiator and the Responder shall calculate the local commitment value using the exchanged public keys, the local random number, and the passkey from the local Host, according to Section 7.7.1 on page 1102.
3. The Initiator shall send an LMP_simple_pairing_confirm PDU with the commitment it calculated in step 2.
4. The Responder shall respond with an LMP_simple_pairing_confirm PDU with the commitment it calculated in step 2.
5. The Initiator shall then send an LMP_simple_pairing_number PDU with the random number it generated in step 1.
6. The Responder shall then calculate commitment from the exchanged public keys, the random number it received and the passkey from the local Host, according to Section 7.7.1 on page 1102. If the calculated commitment and the received commitment are equal, the Responder shall reply with an LMP_accepted PDU.
7. The Responder shall then send an LMP_simple_pairing_number PDU with the random value it generated in step 1.
8. The Initiator shall calculate the commitment using exchanged public keys, the received the random number it received and the passkey from the local Host, according to Section 7.7.1 on page 1102. If the calculated commitment is equal to the received commitment, the Initiator shall reply with an LMP_accepted PDU.



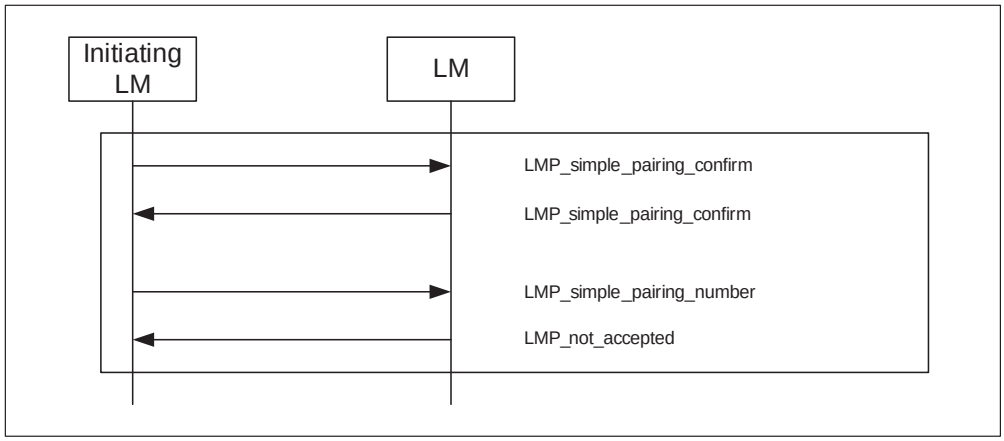
Sequence 53: Authentication Passkey entry

When this procedure has successfully passed 20 times, the Secure Simple Pairing process continues with the Authentication step 2 as described in Authentication stage 2: DHKey Check on page 274.

4.2.7.3.2.1 Commitment Check Failure on the Responder side

If during one of the 20 repetitions, the commitment calculated by the Responder is not equal to the one received from the Initiator (step 6), the Responder shall abort the Secure Simple Pairing process by sending an LMP_not_accepted PDU with reason "Authentication Failure."

Secure Simple Pairing procedures shall then be aborted. The Link Managers shall not disconnect the connection.

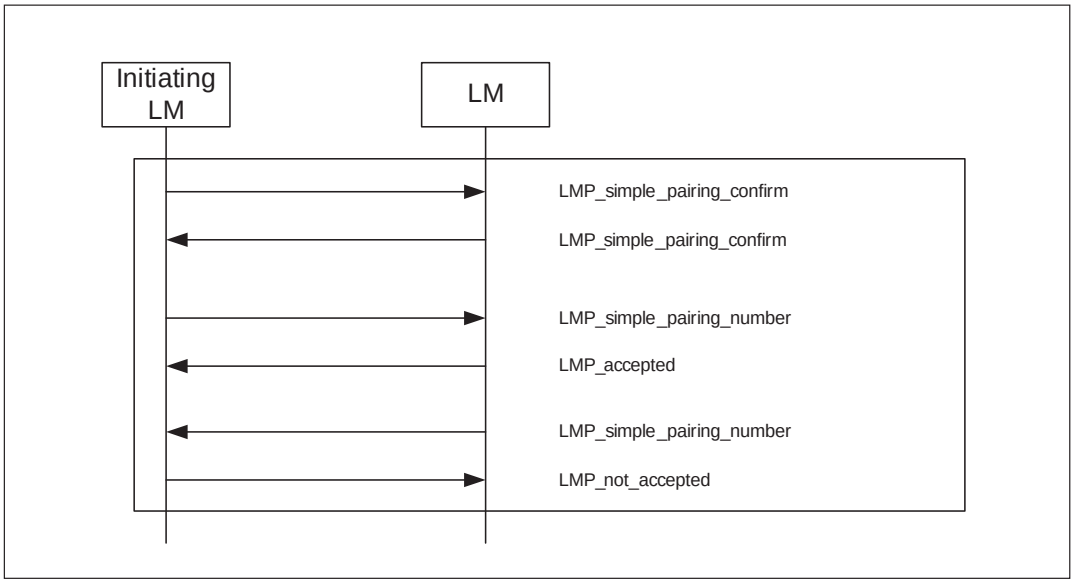


Sequence 54: Authentication Passkey Entry: Commitment check failure on the Responder side

4.2.7.3.2.2 Commitment check failure on the Initiator side

If during one of the 20 repetitions, the commitment calculated by the Initiator is not equal to the one received from the Responder (step 8), the Initiator shall abort the Secure Simple Pairing process by sending an LMP_not_accepted PDU with reason "Authentication Failure".

Secure Simple Pairing procedures shall then be aborted. The Link Managers shall not disconnect the connection.

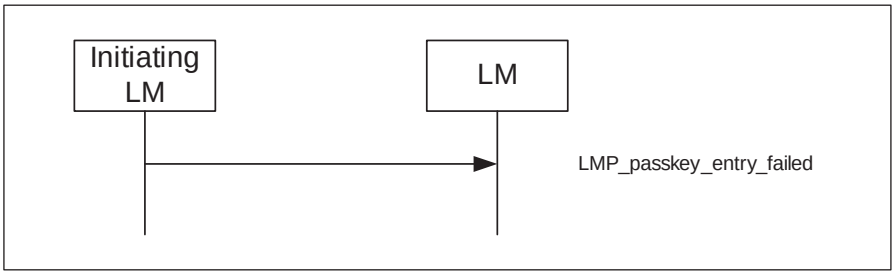


Sequence 55: Authentication Passkey Entry: Commitment check failure on the Initiator side

4.2.7.3.2.3 Passkey Entry Failure on Initiator Side

If the initiating side indicates that the passkey was not entered or canceled (as indicated by the HCI_Passkey_Request_Negative_Reply command) the initiating LM shall send an LMP_passkey_entry_failed PDU.

Secure Simple Pairing process shall then be aborted. The Link Managers shall not disconnect the connection.

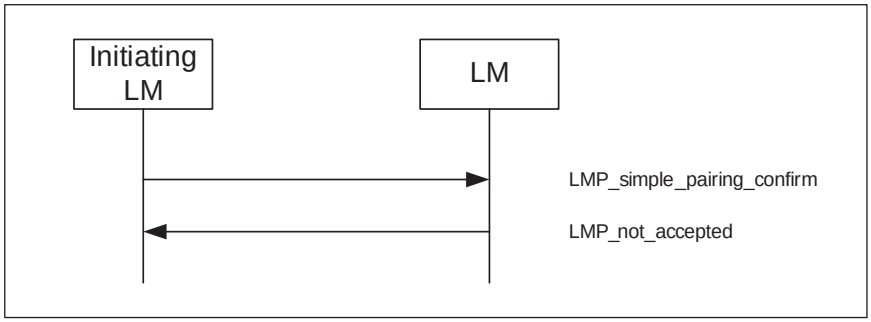


Sequence 56: Authentication stage 1 Passkey Entry Failure on Initiator Side

4.2.7.3.3 Passkey Entry Failure on Responding side

If the responding side indicates that the passkey was not entered or canceled (as indicated by the HCI_Passkey_Request_Negative_Reply command), the responding LM shall send an LMP_not_accepted PDU in response to the LMP_simple_pairing_config PDU.

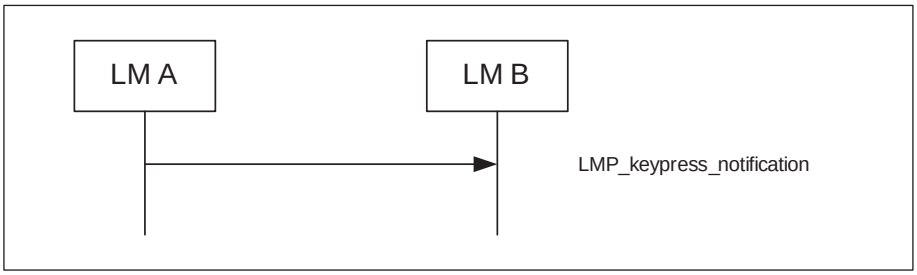
Simple Pairing process shall then be aborted. The Link Managers shall not disconnect the connection.



Sequence 57: Authentication Stage 1 Passkey Entry Failure on Responding Side

4.2.7.3.4 Keypress Notifications

A Controller that allows the Host to change its IO capabilities shall send notifications on key presses to the remote side using the LMP_keypress_notification PDU when the Host sets the IO capabilities to KeyboardOnly IO and when Secure Simple Pairing is supported on the Host and Controller.



Sequence 58: Keypress Notifications

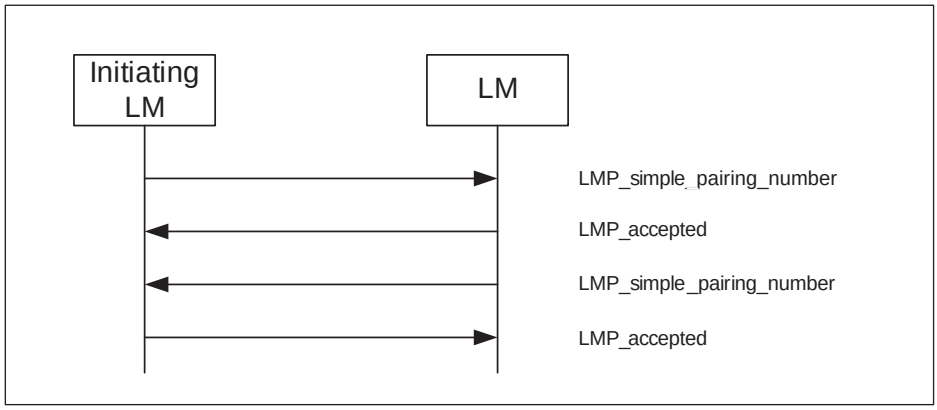
4.2.7.3.5 Authentication Stage 1: OOB

Upon reception of the OOB information (as defined in “Authentication Stage 1: Out of Band Protocol” on page 1096 in Vol. 2) from the host, the devices shall compare the received commitment from its host, with the one calculated using the secret number received from the Host and the public key received from the remote device. If the local Host has not set the OOB Authentication Data Present, the LM shall set the remote secret number to zero and base the subsequent calculations on this value.

If the commitment check on the initiator is valid, the Initiator shall then generate a random number (nonce) and send it to the Responder using an LMP_simple_pairing_number PDU. If the commitment succeeds, the Responder shall acknowledge by sending an LMP_accepted PDU otherwise it shall send an LMP_not_accepted PDU. The Responder shall then generate a random number (nonce) and send it to the Initiator using an LMP_simple_pairing_number PDU. If the commitment succeeds, the Initiator shall acknowledge by sending an LMP_accepted PDU otherwise it shall send an LMP_not_accepted PDU.

If the commitment values don't match in the Initiator, the procedure in Section 4.2.7.3.5.2, “Commitment Check Failure on the Initiator Side,” on page 273 shall apply.

If the commitment values don't match in the Responder, the procedure in Section 4.2.7.3.5.1, “Commitment Check Failure on the Responder Side,” on page 272 shall apply.



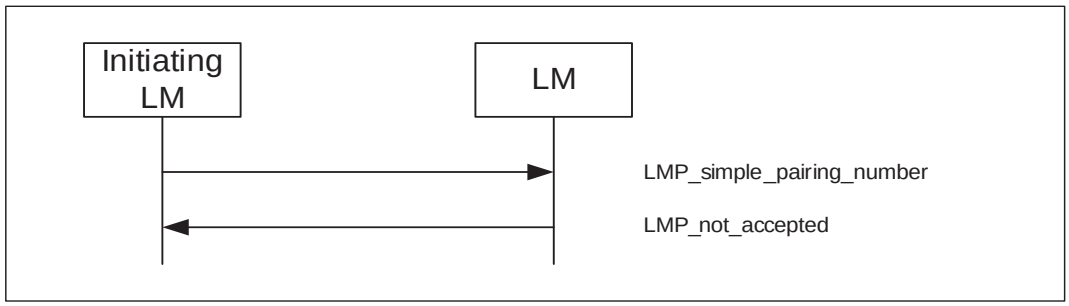
Sequence 59: Authentication OOB: Only one device is OOB-r capable

When these operations have been passed successfully, Secure Simple Pairing procedures continue with Authentication step 2 as described in Authentication stage 2: DHKey Check on page 274.

4.2.7.3.5.1 Commitment Check Failure on the Responder Side

If the commitment received OOB from the host is not equal to the calculated commitment, the Responder shall send an LMP_not_accepted PDU with reason "Authentication Failure" in response to LMP_simple_pairing_number PDU.

Secure Simple Pairing process shall then be aborted. The Link Managers shall not disconnect the connection.

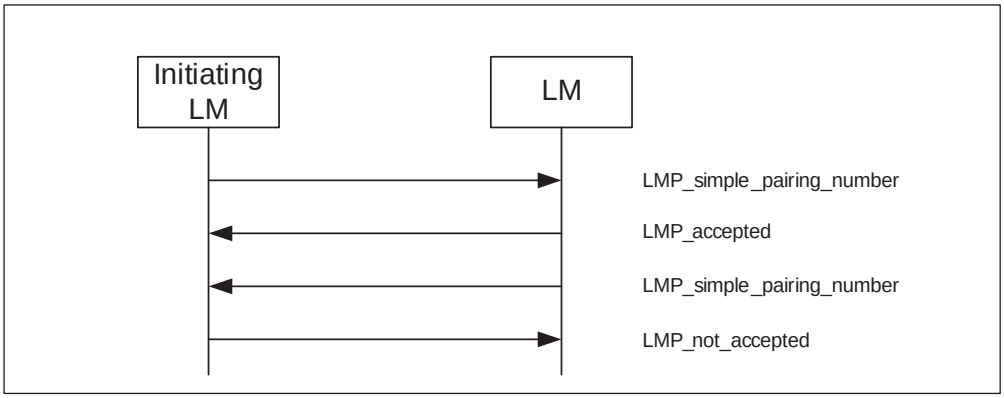


Sequence 60: Authentication stage 1 OOB: Commitment check failure on the Responder side

4.2.7.3.5.2 Commitment Check Failure on the Initiator Side

If the commitment received OOB from the host is not equal to the calculated commitment, the Initiator shall send an LMP_not_accepted PDU with reason "Authentication Failure" in response to LMP_simple_pairing_number PDU.

Secure Simple Pairing process shall then be aborted. The Link Managers shall not disconnect the connection.

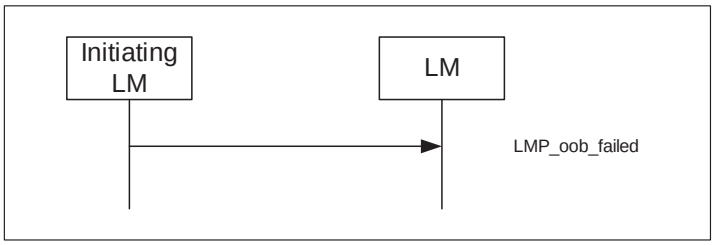


Sequence 61: Authentication stage 1 OOB: Commitment check failure on the Initiator side

4.2.7.3.6 Out-of-Band Information not Available on the Initiator Side

If the Host on the initiating side does not have out-of-band information the Initiator shall send an LMP_oob_failed PDU.

Simple Pairing process shall then be aborted. The Link Managers shall not disconnect the connection.



Sequence 62: Authentication stage 1 OOB: OOB information not available on the Initiator side

4.2.7.4 Authentication stage 2: DHKey Check

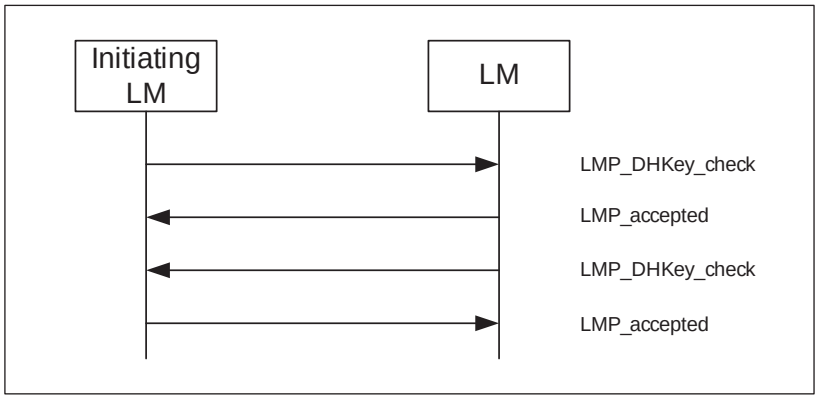
At this stage, both devices compute a new confirmation values based on Diffie-Hellman key and previously exchanged information according to Section 7.7.4, “The Simple Pairing Check Function f3,” on page 1105.

The Initiator shall send an LMP_DHKey_check PDU to the Responder including the confirmation value it has computed. Upon reception, the Responder shall reply with an LMP_accepted PDU if the received value is equal to the one it has calculated according to Section 7.7.4 on page 1105. If it fails, refer to Section 4.2.7.3.5.1 on page 272.

The Responder shall then send an LMP_DHKey_check PDU to the Initiator including its confirmation value it has computed. Upon reception, the Initiator shall reply with an LMP_accepted PDU if the received value is equal to the one it has calculated according to Section 7.7.4 on page 1105. If it fails, refer to Section 4.2.7.4.1.1, “Check Failure on the Initiator Side,” on page 275.

At this point, both devices shall compute the link key according to Section 7.7.3, “The Simple Pairing Key Derivation Function f2,” on page 1104.

The Initiator shall then start standard mutual authentication as described in Section 4.2.1.1

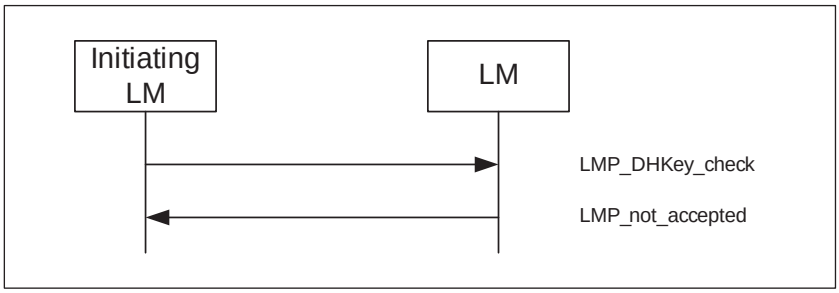


Sequence 63: DHKey check

4.2.7.4.1 Check Failure on the Responder Side

If the confirmation value received via LMP by the Responder is not equal to the one it has calculated according to Section 7.7.4 on page 1105, the Responder shall send an LMP_not_accepted PDU with reason "Authentication Failure".

Secure Simple Pairing procedures shall then be aborted. The Link Managers shall not disconnect the connection.

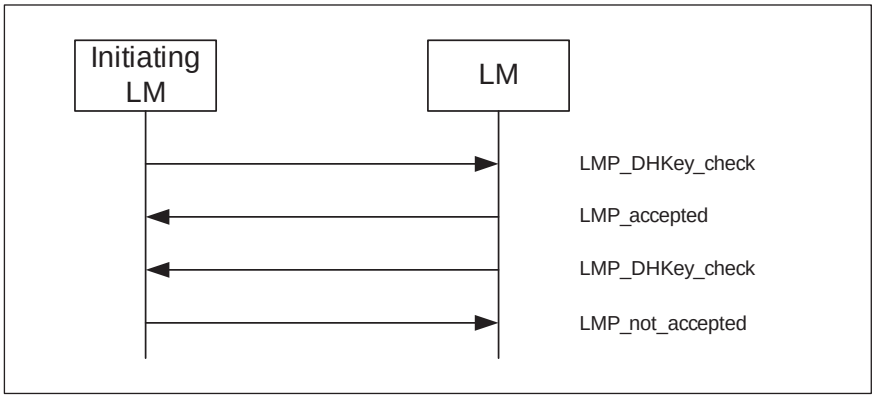


Sequence 64: DHKey check: Check failure on the Responder side

4.2.7.4.1.1 Check Failure on the Initiator Side

If the confirmation value received via LMP by the Initiator is not equal to the one it has calculated according to Section 7.7.4 on page 1105, the Initiator shall send an LMP_not_accepted PDU with reason "Authentication Failure".

Secure Simple Pairing procedures shall then be aborted. The Link Managers shall not disconnect the connection.



Sequence 65: DHKey check: check failure on the Initiator side

4.3 INFORMATIONAL REQUESTS

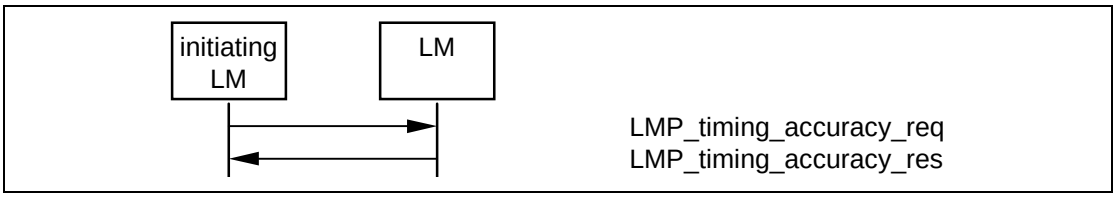
4.3.1 Timing Accuracy

LMP supports requests for the timing accuracy. This information can be used to minimize the scan window during piconet physical channel re-synchronization

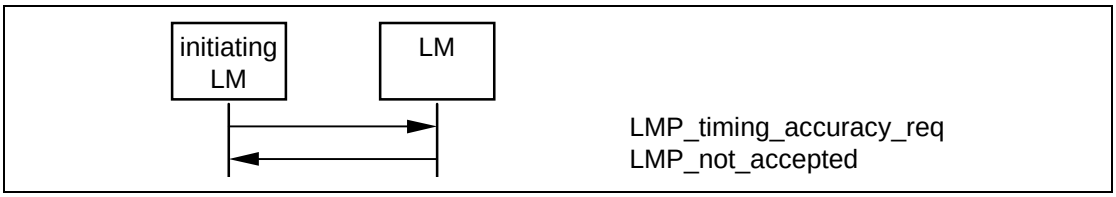
(see Baseband Specification, Section 2.2.5.2, on page 75). The timing accuracy parameters returned are the long term drift measured in ppm and the long term jitter measured in μs of the worst case clock used. These parameters are fixed for a certain device and shall be identical when requested several times. Otherwise, the requesting device shall assume worst case values (drift=250ppm and jitter=10μs).

M/O	PDU	Contents
O(4)	LMP_timing_accuracy_req	-
O(4)	LMP_timing_accuracy_res	drift jitter

Table 4.19: Request limited timing PDU



Sequence 66: The requested device supports timing accuracy information.



Sequence 67: The requested device does not support timing accuracy information.

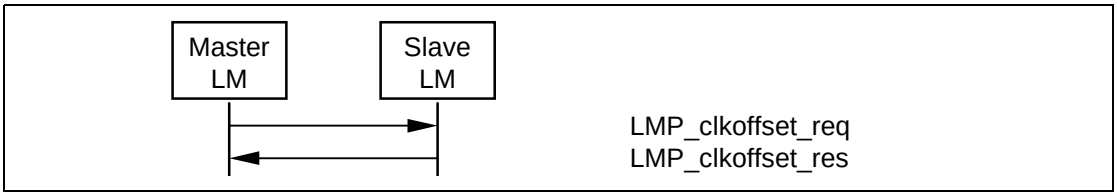
4.3.2 Clock Offset

The clock offset can be used to speed up the paging time the next time the same device is paged. The master can request the clock offset at anytime following a successful baseband paging procedure (i.e., before, during or after connection setup). The clock offset shall be defined by the following equation:

$(CLKN_{16\text{-}2\ slave} - CLKN_{16\text{-}2\ master})$ mod 2**15.

M/O	PDU	Contents
M	LMP_clkoffset_req	-
M	LMP_clkoffset_res	clock offset

Table 4.20: PDUs used for clock offset request.



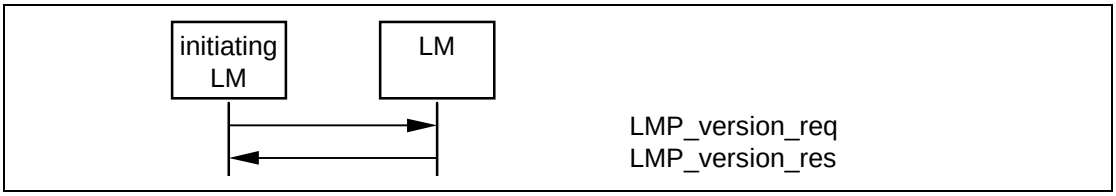
Sequence 68: Clock offset requested.

4.3.3 LMP version

LMP supports requests for the version of the LM protocol. The LMP_version_req and LMP_version_res PDUs contain three parameters: VersNr, CompId and SubVersNr. VersNr specifies the version of the Bluetooth LMP specification that the device supports. CompId is used to track possible problems with the lower Bluetooth layers. All companies that create a unique implementation of the LM shall have their own CompId. The same company is also responsible for the administration and maintenance of the SubVersNr. It is recommended that each company has a unique SubVersNr for each RF/BB/LM implementation. For a given VersNr and CompId, the values of the SubVersNr shall increase each time a new implementation is released. For both CompId and SubVersNr the value 0xFFFF means that no valid number applies. There is no ability to negotiate the version of the LMP. The sequence below is only used to exchange the parameters. LMP version can be requested at anytime following a successful baseband paging procedure.

M/O	PDU	Contents
M	LMP_version_req	VersNr CompId SubVersNr
M	LMP_version_res	VersNr CompId SubVersNr

Table 4.21: PDUs used for LMP version request.



Sequence 69: Request for LMP version.

4.3.4 Supported Features

The supported features may be requested at anytime following a successful baseband paging procedure by sending the LMP_features_req PDU. Upon reception of an LMP_features_req PDU, the receiving device shall return an LMP_features_res PDU.

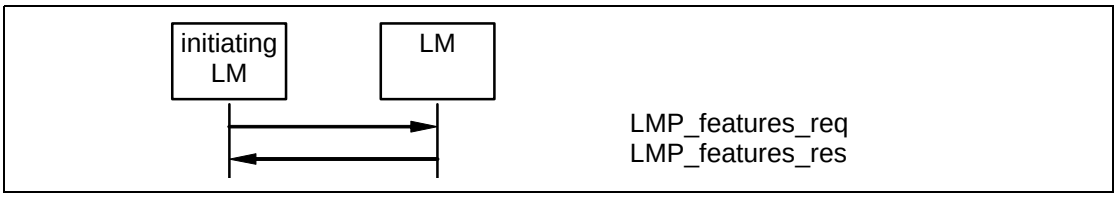
The number of features bits required will in the future exceed the size of a single page of features. An extended features mask is therefore provided to allow support for more than 64 features. Support for the extended features mask is indicated by the presence of the appropriate bit in the LMP features mask. The LMP_features_req_ext and LMP_features_res_ext PDUs operate in precisely the same way as the LMP_features_req and LMP_features_res PDUs except that they allow the various pages of the extended features mask to be requested. The LMP_features_req_ext may be sent at any time following the exchange of the LMP_features_req and LMP_features_rsp PDUs.

The LMP_features_req_ext PDU contains a feature page index that specifies which page is requested and the contents of that page for the requesting device. Pages are numbered from 0-255 with page 0 corresponding to the normal features mask. Each page consists of 64 bits. If a device does not support any page number it shall return a mask with every bit set to 0. It also contains the maximum features page number containing any non-zero bit for this device. The recipient of an LMP_features_req_ext PDU shall respond with an LMP_features_res_ext PDU containing the same page number and the appropriate features page along with its own maximum features page number.

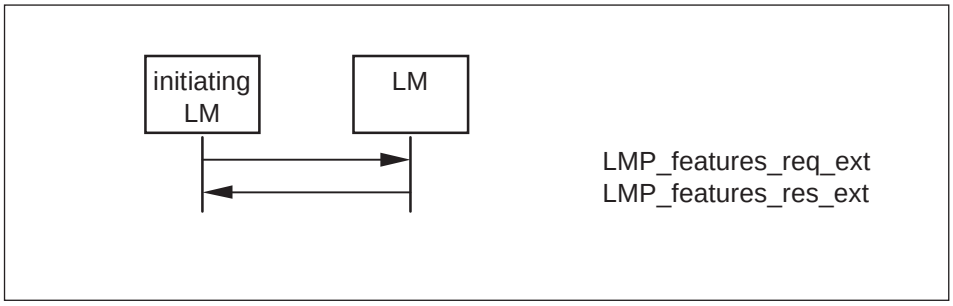
If the extended features request is not supported then all bits in all extended features pages for that device shall be assumed to be zero.

M/O	PDU	Contents
M	LMP_features_req	features
M	LMP_features_res	features
O(63)	LMP_features_req_ext	features page max supported page extended features
O(63)	LMP_features_res_ext	features page max supported page extended features

Table 4.22: PDUs used for features request.



Sequence 70: Request for supported features.



Sequence 71: Request for extended features.

4.3.5 Name Request

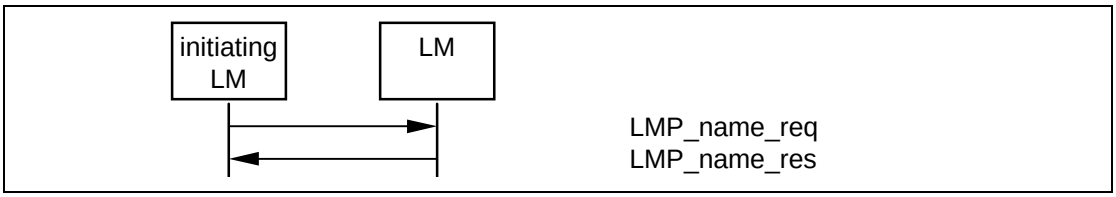
LMP supports name request to another device. The name is a user-friendly name associated with the device and consists of a maximum of 248 bytes coded according to the UTF-8 standard. The name is fragmented over one or more DM1 packets. When an LMP_name_req PDU is sent, a name offset indicates which fragment is expected. The corresponding LMP_name_res PDU carries the same name offset, the name length indicating the total number of bytes in the name of the device and the name fragment, where:

- name fragment(N) = name(N + name offset), if (N + name offset) < name length
- name fragment(N) = 0,otherwise.

Here $0 \leq N \leq 13$. In the first sent LMP_name_req PDU, name offset=0. Sequence 72 is then repeated until the initiator has collected all fragments of the name. The name request may be made at any time following a successful baseband paging procedure.

M/O	PDU	Contents
M	LMP_name_req	name offset
M	LMP_name_res	name offset name length name fragment

Table 4.23: Name request PDUs



Sequence 72: Device's name requested and it responses.

4.4 ROLE SWITCH

4.4.1 Slot Offset

With LMP_slot_offset the information about the difference between the slot boundaries in different piconets is transmitted. The LMP_slot_offset PDU may be sent anytime after the baseband paging procedure has completed. This PDU carries the parameters slot offset and BD_ADDR. The slot offset shall be the time in microseconds between the start of a master transmission in the current piconet to the start of the next following master transmission in the piconet where the BD_ADDR device (normally the slave) is master at the time that the request is interpreted by the BD_ADDR device.

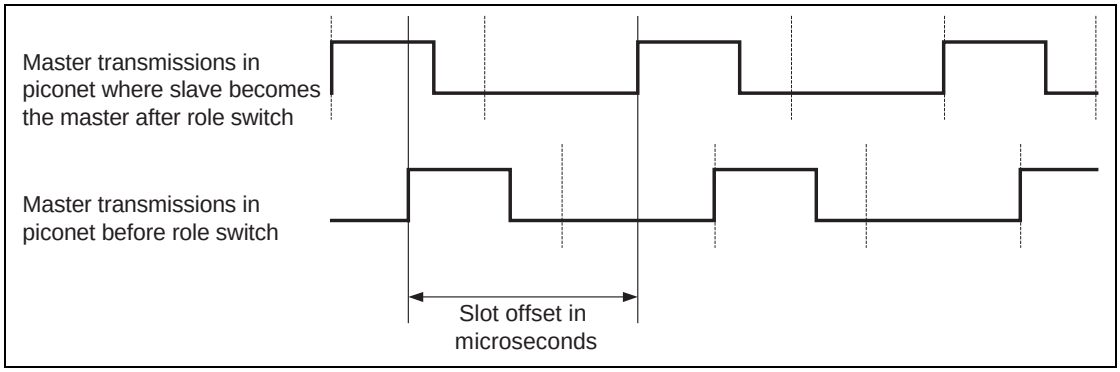
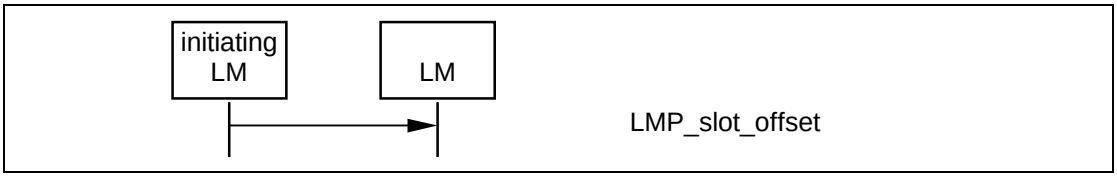


Figure 4.2: Slot offset for role switch

See Section 4.4 on page 281 for the use of LMP_slot_offset in the context of the role switch. In the case of role switch the BD_ADDR is that of the slave device.

M/O	PDU	Contents
O(3)	LMP_slot_offset	slot offset BD_ADDR

Table 4.24: Role switch PDU



Sequence 73: Slot offset information is sent

4.4.2 Role Switch

Since the paging device always becomes the master of the piconet, a switch of the master slave role is sometimes needed, see Baseband Specification, Section 8.6.5, on page 172. A role switch may be performed anytime after the baseband paging procedure has completed.

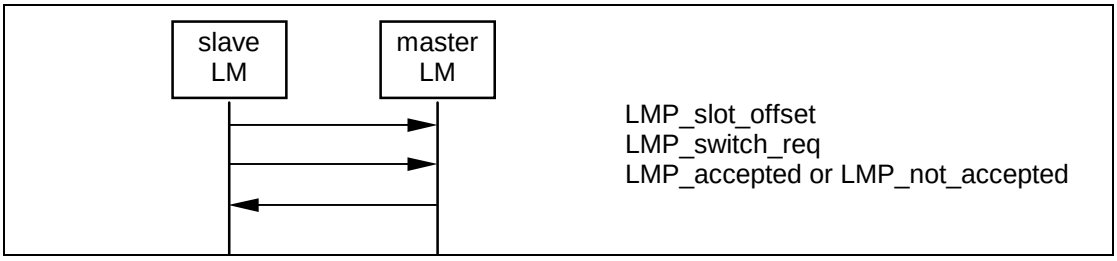
Support for LMP_slot_offset is mandatory if LMP_switch_req is supported.

The LMP_slot_offset shall be sent only if the ACL logical transport is in active mode. The LMP_switch_req shall be sent only if the ACL logical transport is in active mode, when encryption is disabled or paused, and all synchronous logical transports on the same physical link are disabled. Additionally, LMP_slot_offset or LMP_switch_req shall not be initiated or accepted while a synchronous logical transport is being negotiated by LM.

M/O	PDU	Contents
O(5)	LMP_switch_req	switch instant
O(5)	LMP_slot_offset	slot offset BD_ADDR

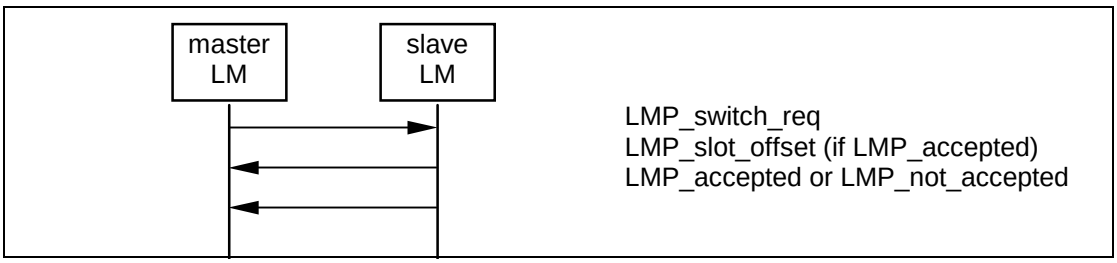
Table 4.25: Role switch PDU

The initiating LM shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107). If the encryption mode is set to "encryption" and both devices support pausing encryption, then the initiating device shall initiate the pause encryption sequence (see Section 4.2.5.5, “Pause Encryption,” on page 258.) It shall then send an LMP_slot_offset PDU immediately followed by an LMP_switch_req PDU. If the master accepts the role switch and encryption is not already paused, it shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107) and respond with an LMP_accepted PDU. When the role switch has been completed at the baseband level (successfully or not) if encryption was paused, the device that paused encryption shall initiate the resume encryption sequence (see Section 4.2.5.6, “Resume Encryption,” on page 260). If encryption was not paused, both devices shall re-enable transmission on the ACL-U logical link. If the master rejects the role switch it responds with an LMP_not_accepted PDU and the slave re-enables transmission on the ACL-U logical link. The transaction ID for the role switch PDUs in the sequence (LMP_slot_offset and LMP_switch_req along with the associated LMP_accepted or LMP_not_accepted) shall be set to 1.



Sequence 74: Role switch (slave initiated).

If the master initiates the role switch and if the encryption mode is set to "encryption" and both devices support pausing encryption, the master device shall initiate the pause encryption sequence (See [Part C] Section 4.2.5.5 on page 258), otherwise it shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107) and send an LMP_switch_req PDU. If the slave accepts the role switch and encryption is not already paused, it shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107) and responds with an LMP_slot_offset PDU immediately followed by an LMP_accepted PDU. When the role switch has been completed at the baseband (successfully or not) if encryption was paused, the device that paused encryption shall initiate the resume encryption sequence (see Section 4.2.5.6, “Resume Encryption,” on page 260). If encryption was not paused, both devices re-enable transmission on the ACL-U logical link. If the slave rejects the role switch it responds with an LMP_not_accepted PDU and the master re-enables transmission on the ACL-U logical link. The transaction ID for the role switch PDUs in the sequence (LMP_slot_offset and LMP_switch_req along with the associated LMP_accepted or LMP_not_accepted) shall be set to 0.



Sequence 75: Role switch (master initiated).

The LMP_switch_req PDU contains a parameter, switch instant, which specifies the instant at which the TDD switch is performed. This is specified as a Bluetooth clock value of the master's clock, that is available to both devices. This instant is chosen by the sender of the message and shall be at least $2*T_{poll}$ or 32 (whichever is greater) slots in the future. The switch instant shall be within 12 hours of the current clock value to avoid clock wrap.

The sender of the LMP_switch_req PDU selects the switch instant and queues the LMP_switch_req PDU to LC for transmission and starts a timer to expire at the switch instant. When the timer expires it initiates the mode switch. In the case of a master initiated switch if the LMP_slot_offset PDU has not been

received by the switch instant the role switch is carried out without an estimate of the slave's slot offset. If an LMP_not_accepted PDU is received before the timer expires then the timer is stopped and the role switch shall not be initiated.

When the LMP_switch_req is received the switch instant is compared with the current master clock value. If it is in the past then the instant has been passed and an LMP_not_accepted PDU with the error code *instant passed* shall be returned. If it is in the future then an LMP_accepted PDU shall be returned assuming the role switch is allowed and a timer is started to expire at the switch instant. When this timer expires the role switch shall be initiated.

After a successful role switch the supervision timeout and poll interval (T_{poll}) shall be set to their default values. The authentication state and the ACO shall remain unchanged. Adaptive Frequency Hopping shall follow the procedures described in Baseband Specification, Section 8.6.5, on page 172. The default value for max_slots shall be used.

4.5 MODES OF OPERATION

4.5.1 Hold Mode

The ACL logical transport of a connection between two Bluetooth devices can be placed in hold mode for a specified hold time. See Baseband Specification, Section 8.8, on page 183 for details.

M/O	PDU	Contents
O(6)	LMP_hold	hold time, hold instant
O(6)	LMP_hold_req	hold time, hold instant

Table 4.26: Hold mode PDUs

The LMP_hold and LMP_hold_req PDUs both contain a parameter, hold instant, that specifies the instant at which the hold becomes effective. This is specified as a Bluetooth clock value of the master's clock, that is available to both devices. The hold instant is chosen by the sender of the message and should be at least $6*T_{poll}$ slots in the future. The hold instant shall be within 12 hours of the current clock value to avoid clock wrap.

4.5.1.1 Master Forces Hold Mode

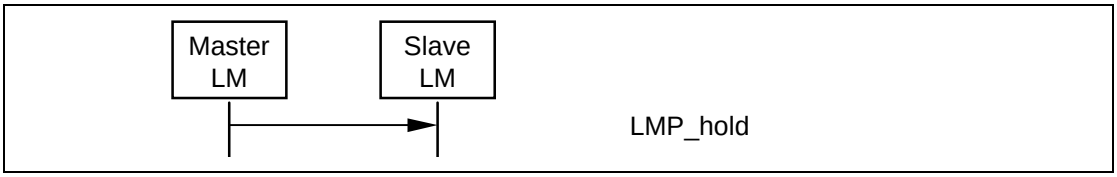
The master may force hold mode if there has previously been a request for hold mode that has been accepted by the slave. The hold time included in the PDU when the master forces hold mode shall not be longer than any hold time the slave has previously accepted when there was a request for hold mode.

The master LM shall first pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107). It shall select the hold instant and queue the LMP_hold PDU to its LC for transmission. It shall then start a timer

to wait until the hold instant occurs. When this timer expires then the connection shall enter hold mode. If the baseband acknowledgement for the LMP_hold PDU is not received then the master may enter hold mode, but it shall not use its low accuracy clock during the hold.

When the slave LM receives an LMP_hold PDU it compares the hold instant with the current master clock value. If it is in the future then it starts a timer to expire at this instant and enters hold mode when it expires.

When the master LM exits from Hold mode it re-enables transmission on the ACL-U logical link.



Sequence 76: Master forces slave into hold mode

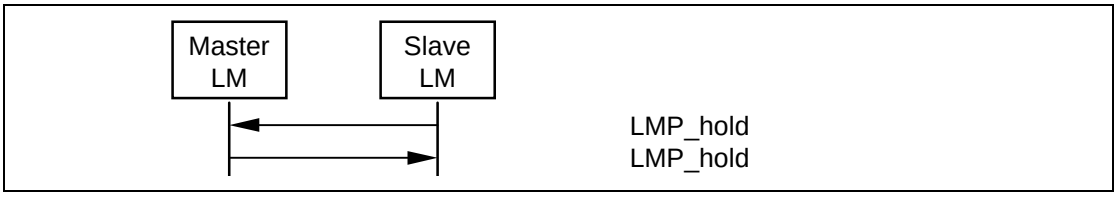
4.5.1.2 Slave Forces Hold Mode

The slave may force hold mode if there has previously been a request for hold mode that has been accepted by the master. The hold time included in the PDU when the slave forces hold mode shall not be longer than any hold time the master has previously accepted when there was a request for hold mode.

The slave LM shall first complete the transmission of the current packet on the ACL logical transport and then shall suspend transmission on the ACL-U logical link. It shall select the hold instant and queue the LMP_hold PDU to its LC for transmission. It shall then wait for an LMP_hold PDU from the master acting according to the procedure described in Section 4.5.1.1.

When the master LM receives an LMP_hold PDU it shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107). It shall then inspect the hold instant. If this is less than $6*T_{poll}$ slots in the future it shall modify the instant so that it is at least $6*T_{poll}$ slots in the future. It shall then send an LMP_hold PDU using the mechanism described in Section 4.5.1.1.

When the master and slave LMs exit from Hold mode they shall re-enable transmission on the ACL-U logical link.



Sequence 77: Slave forces master into hold mode

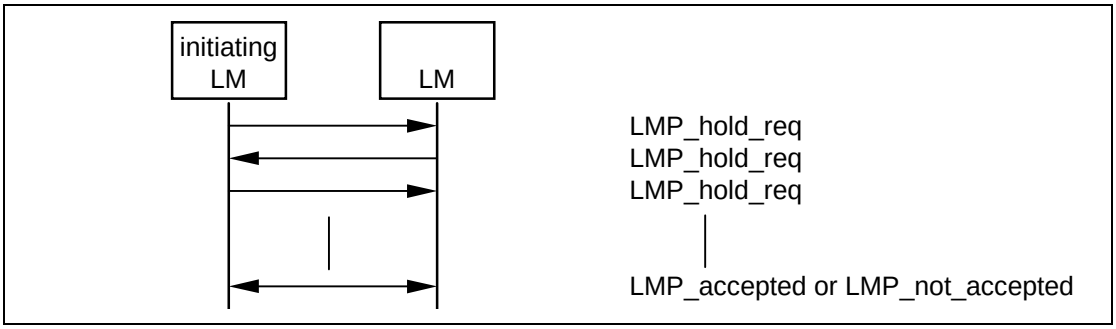
4.5.1.3 Master or Slave requests Hold Mode

The master or the slave may request to enter hold mode. Upon receipt of the request, the same request with modified parameters may be returned or the negotiation may be terminated. If an agreement is seen an LMP_accepted PDU terminates the negotiation and the ACL link is placed in hold mode. If no agreement is seen, an LMP_not_accepted PDU with the error code *unsupported parameter value* terminates the negotiation and hold mode is not entered.

The initiating LM shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107). On receiving an LMP_hold_req PDU the receiving LM shall complete the transmission of the current packet on the ACL logical transport and then shall suspend transmission on the ACL-U logical link.

The LM sending the LMP_hold_req PDU selects the hold instant, that shall be at least $9*T_{poll}$ slots in the future. If this is a response to a previous LMP_hold_req PDU and the contained hold instant is at least $9*T_{poll}$ slots in the future then this shall be used. The LMP_hold_req PDU shall then be queued to its LC for transmission and a timer shall be started to expire at this instant and the connection enters hold mode when it expires unless an LMP_not_accepted or LMP_hold_req PDU is received by its LM before that point. If the LM receiving LMP_hold_req PDU agrees to enter hold mode it shall return an LMP_accepted PDU and shall start a timer to expire at the hold instant. When this timer expires it enters hold mode.

When each LM exits from Hold mode it shall re-enable transmission on the ACL-U logical link.



Sequence 78: Negotiation for hold mode

4.5.2 Park State

If a slave does not need to participate in the channel, but should still remain synchronized to the master, it may be placed in park state. See Baseband Specification, Section 8.9, on page 184 for details.

Note: To keep a parked slave connected the master shall periodically unpark and repark the slave if the supervision timeout is not set to zero (see Baseband Specification, Section 3.1, on page 96).

All PDUs sent from the master to parked slaves are carried on the PSB-C logical link (LMP link of parked slave broadcast logical transport). These PDUs, LMP_set_broadcast_scan_window, LMP_modify_beacon, LMP_unpark_BD_addr_req and LMP_unpark_PM_addr_req, are the only PDUs that shall be sent to a slave in park state and the only PDUs that shall be broadcast. To increase reliability for broadcast, the packets are as short as possible. Therefore the format for these LMP PDUs are somewhat different. The parameters are not always byte-aligned and the length of the PDUs is variable.

The messages for controlling park state include parameters, defined in Baseband Specification, Section 8.9, on page 184. When a slave is placed in park state it is assigned a unique PM_ADDR, that can be used by the master to unpark that slave. The all-zero PM_ADDR has a special meaning; it is not a valid PM_ADDR. If a device is assigned this PM_ADDR, it shall be identified with its BD_ADDR when it is unparked by the master.

4.5.2.1 Master requests slave to enter park state

M/O	PDU	Contents
O(8)	LMP_park_req	timing control flags D_B T_B N_B Δ_B PM_ADDR AR_ADDR N_{Bsleep} D_{Bsleep} D_{access} T_{access} $N_{acc\text{-}slots}$ N_{poll} M_{access} access scheme
O(8)	LMP_set_broadcast_scan_window	timing control flags D_B (optional) broadcast scan window
O(8)	LMP_modify_beacon	timing control flags D_B (optional) T_B N_B Δ_B D_{access} T_{access} $N_{acc\text{-}slots}$ N_{poll} M_{access} access scheme

Table 4.27: PDUs used for park state

M/O	PDU	Contents
O(8)	LMP_unpark_PM_ADDR_req	timing control flags D_B (optional) LT_ADDR 1st unpark LT_ADDR 2nd unpark PM_ADDR 1st unpark PM_ADDR 2nd unpark LT_ADDR 3rd unpark LT_ADDR 4th unpark PM_ADDR 3rd unpark PM_ADDR 4th unpark LT_ADDR 5th unpark LT_ADDR 6th unpark PM_ADDR 5th unpark PM_ADDR 6th unpark LT_ADDR 7th unpark PM_ADDR 7th unpark
O(8)	LMP_unpark_BD_ADDR_req	timing control flags D_B (optional) LT_ADDR LT_ADDR (optional) BD_ADDR BD_ADDR (optional)

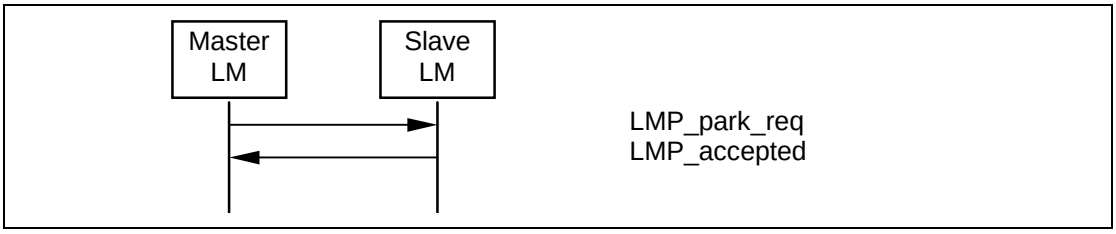
Table 4.27: PDUs used for park state

The master can request park state. The master LM shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107) and then send an LMP_park_req PDU. If the slave agrees to enter park state it shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107). and then respond with an LMP_accepted PDU.

When the slave queues an LMP_accepted PDU it shall start a timer for 6*Tpoll slots. If the baseband acknowledgement is received before this timer expires it shall enter park state immediately otherwise it shall enter park state when the timer expires.

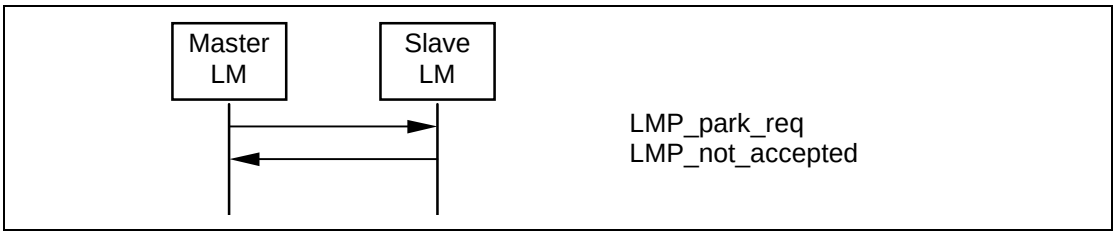
When the master receives an LMP_accepted PDU it shall start a timer for 6*Tpoll slots. When this timer expires the slave is in park state and the LT_ADDR may be re-used.

If the master never receives an LMP_accepted PDU then a link supervision timeout will occur.



Sequence 79: Slave accepts to enter park state.

If the slave rejects the attempt to enter park state it shall respond with an LMP_not_accepted PDU and the master shall re-enable transmission on the ACL-U logical link.



Sequence 80: Slave rejects to enter into park state

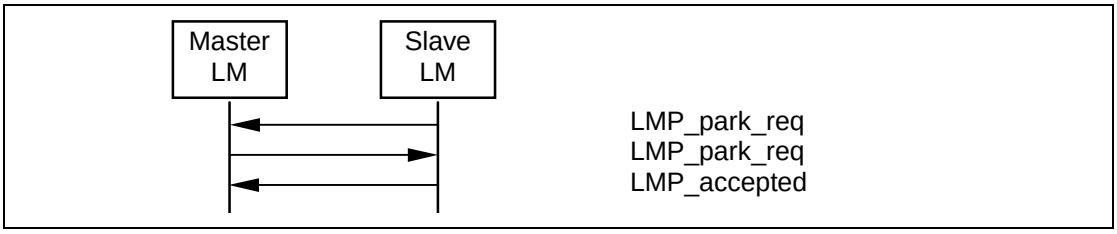
4.5.2.2 Slave requests to enter Park State

The slave can request park state. The slave LM shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107) and then send an LMP_park_req PDU. When sent by the slave, the parameters PM_ADDR and AR_ADDR are not valid and the other parameters represent suggested values. If the master accepts the slave's request to enter park state it shall pause traffic on the ACL-U logical link (see Baseband Specification, Section 5.3.1, on page 107) and then send an LMP_park_req PDU, where the parameter values may be different from the values in the PDU sent from the slave. If the slave can accept these parameter it shall respond with an LMP_accepted PDU.

When the slave queues an LMP_accepted PDU for transmission it shall start a timer for 6*Tpoll slots. If the baseband acknowledgement is received before this timer expires it shall enter park state immediately otherwise it shall enter park state when the timer expires.

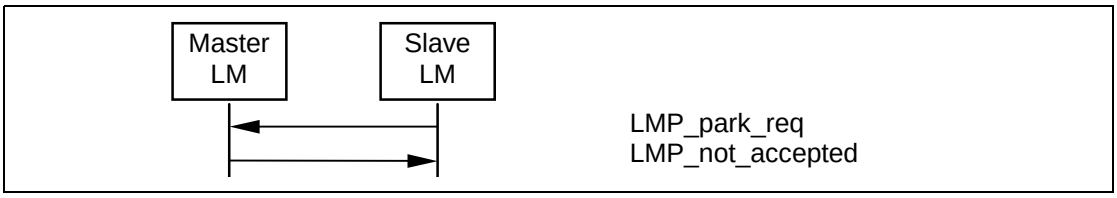
When the master receives an LMP_accepted PDU it shall start a timer for 6*Tpoll slots. When this timer expires the slave is in park state and the LT_ADDR may be re-used.

If the master never receives the LMP_accepted PDU then a link supervision timeout will occur.



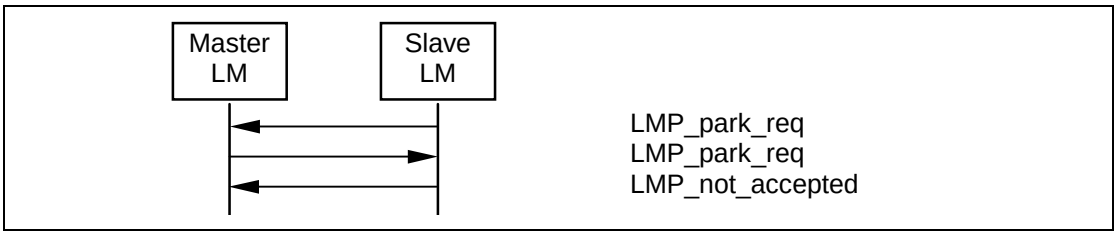
Sequence 81: Slave requests to enter park state and accepts master's beacon parameters

If the master does not agree that the slave enters park state it shall send an LMP_not_accepted PDU. The slave shall then re-enable transmission on the ACL-U logical link.



Sequence 82: Master rejects slave's request to enter park state

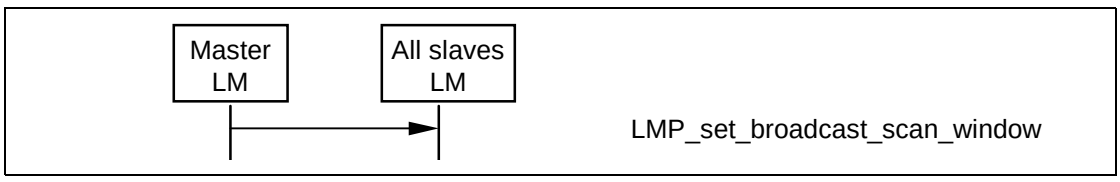
If the slave does not accept the parameters in the LMP_park_req PDU sent from the master it shall respond with an LMP_not_accepted PDU and both devices shall re-enable transmission on the ACL-U logical link.



Sequence 83: Slave requests to enter park state, but rejects master's beacon parameters

4.5.2.3 Master sets up Broadcast Scan Window

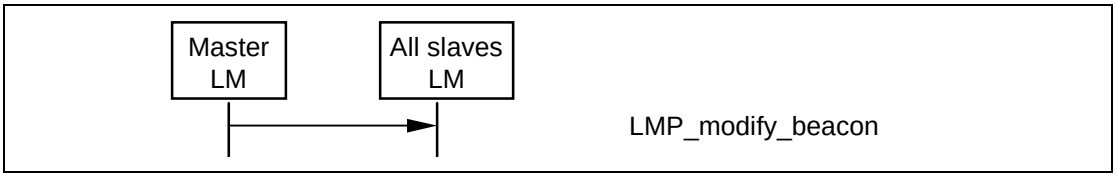
If more broadcast capacity is needed than the beacon train, the master may indicate to the slaves that more broadcast information will follow the beacon train by sending an LMP_set_broadcast_scan_window PDU. This message shall be sent in a broadcast packet at the beacon slot(s). The scan window shall start in the beacon instant and shall only be valid for the current beacon.



Sequence 84: Master notifies all slaves of increase in broadcast capacity

4.5.2.4 Master Modifies Beacon Parameters

When the beacon parameters change the master notifies the parked slaves of this by sending an LMP_modify_beacon PDU. This PDU shall be sent in a broadcast packet.



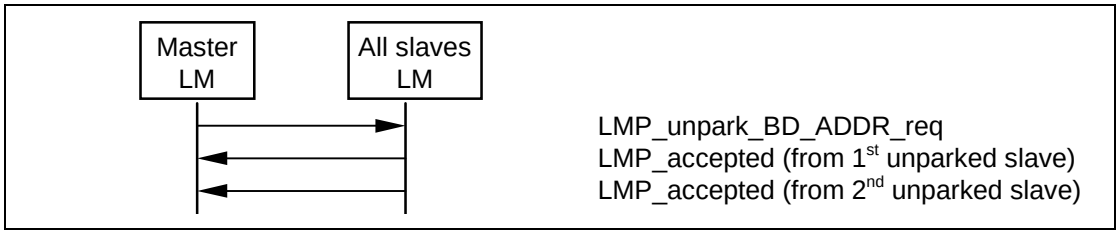
Sequence 85: Master modifies beacon parameters

4.5.2.5 Unparking Slaves

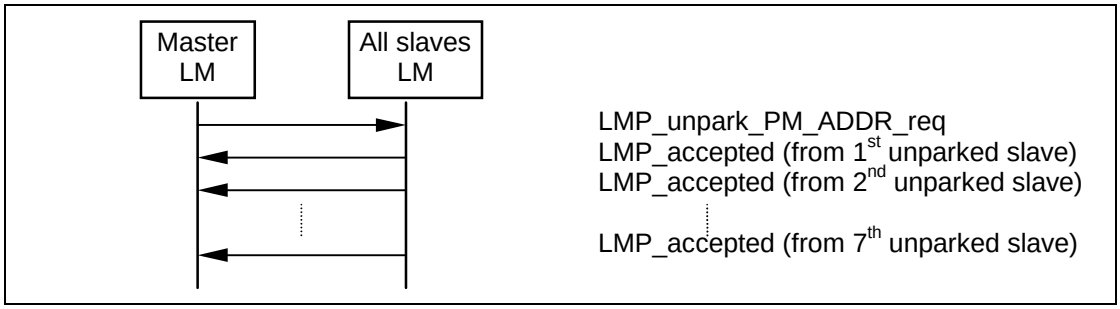
The master can unpark one or many slaves by sending a broadcast LMP message including the PM_ADDR or the BD_ADDR of the device(s) to be unparked. Broadcast LMP messages are carried on the PSB-C logical link. See Baseband Specification, Section 8.9.5, on page 190 for further details. This message also includes the LT_ADDR that the master assigns to the slave(s). After sending this message, the master shall check the success of the unpark by polling each unparked slave by sending POLL packets, so that the slave is granted access to the channel. The unparked slave shall then send a response with an LMP_accepted PDU. If this message is not received from the slave within a certain time after the master sent the unpark message, the unpark failed and the master shall consider the slave as still being in park state.

One PDU is used where the parked device is identified with the PM_ADDR, and another PDU is used where it is identified with the BD_ADDR. Both messages have variable length depending on the number of slaves the master unparks. For each slave the master wishes to unpark an LT_ADDR followed by the PM_ADDR or BD_ADDR of the device that is assigned this LT_ADDR is included in the payload. If the slaves are identified with the PM_ADDR a maximum of 7 slaves can be unparked with the same message. If they are identified with the BD_ADDR a maximum of 2 slaves can be unparked with the same message.

After a successful unparking, both devices re-enable transmission on the ACL-U logical link.



Sequence 86: Master unparks slaves addressed with their BD_ADDR



Sequence 87: Master unparks slaves addressed with their PM_ADDR

4.5.3 Sniff Mode

To enter sniff mode, master and slave negotiate a sniff interval T_{sniff} and a sniff offset, D_{sniff}, that specifies the timing of the sniff slots. The offset determines the time of the first sniff slot; after that the sniff slots follow periodically with the sniff interval T_{sniff}. To avoid clock wrap-around during the initialization, one of two options is chosen for the calculation of the first sniff slot. A timing control flag in the message from the master indicates this. Only bit1 of the timing control flag is valid.

When the ACL logical transport is in sniff mode the master shall only start a transmission in the sniff slots. Two parameters control the listening activity in the slave: the sniff attempt and the sniff timeout. The sniff attempt parameter determines for how many slots the slave shall listen when the slave is not treating this as a scatternet link, beginning at the sniff slot, even if it does not receive a packet with its own LT_ADDR. The sniff timeout parameter determines for how many additional slots the slave shall listen when the slave is not treating this as a scatternet link if it continues to receive only packets with its own LT_ADDR. It is not possible to modify the sniff parameters while the device is in sniff mode.

M/O	PDU	Contents
O(7)	LMP_sniff_req	timing control flags D_{sniff} T_{sniff} sniff attempt sniff timeout
O(7)	LMP_unsniff_req	-

Table 4.28: Sniff mode PDUs

4.5.3.1 Master or Slave requests Sniff Mode

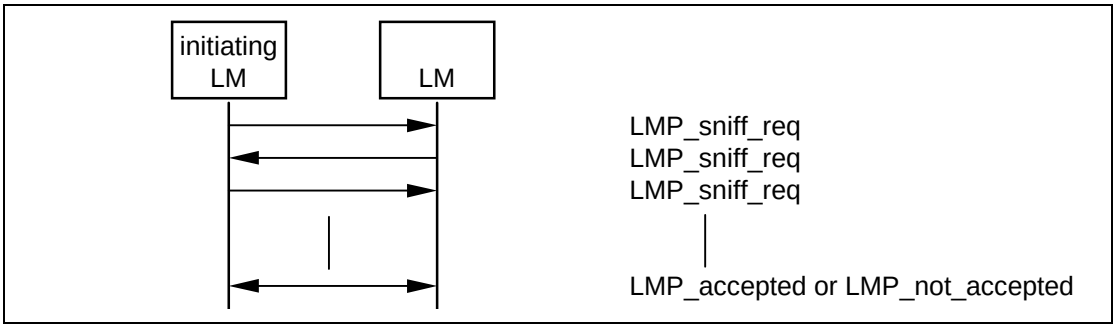
Either the master or the slave may request entry to sniff mode. The process is initiated by sending an LMP_sniff_req PDU containing a set of parameters. The receiving LM shall then decide whether to reject the attempt by sending an LMP_not_accepted PDU, to suggest different parameters by replying with an LMP_sniff_req PDU or to accept the request.

Before the first time that the master sends LMP_sniff_req it shall enter sniff transition mode. If the master receives or sends an LMP_not_accepted PDU it shall exit from sniff transition mode. If the master receives an LMP_sniff_req PDU it shall enter sniff transition mode.

If the master decides to accept the request it shall send an LMP_accepted PDU. When the master receives the baseband acknowledgement for this PDU it shall exit sniff transition mode and enter sniff mode.

If the master receives an LMP_accepted PDU the master shall exit from sniff transition mode and enter sniff mode.

If the slave receives an LMP_sniff_req PDU it must decide whether to accept the request. If the slave does not wish to enter sniff mode then it replies with an LMP_not_accepted PDU. If it is happy to enter sniff mode but requires a different set of parameters it shall respond with an LMP_sniff_req PDU containing the new parameters. If the slave decides that the parameters are acceptable then it shall send an LMP_accepted PDU and enter sniff mode. If the slave receives an LMP_not_accepted PDU it shall terminate the attempt to enter sniff mode.



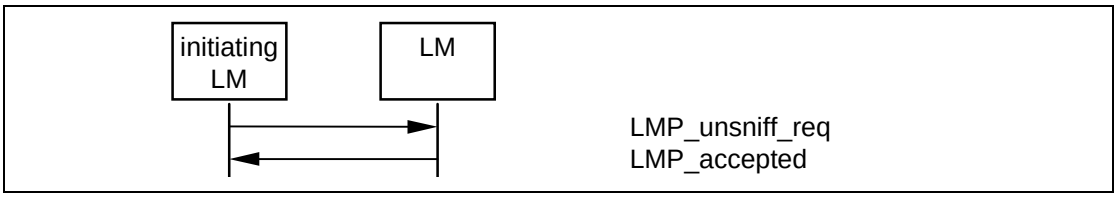
Sequence 88: Negotiation for sniff mode

4.5.3.2 Moving a Slave From Sniff Mode to Active Mode

Sniff mode may be exited by either the master or the slave sending an LMP_unsniff_req PDU. The requested device shall reply with an LMP_accepted PDU.

If the master requests an exit from sniff mode it shall enter sniff transition mode and then send an LMP_unsniff_req PDU. When the slave receives the LMP_unsniff_req it shall exit from sniff mode and reply with an LMP_accepted PDU. When the master receives the LMP_accepted PDU it shall exit from sniff transition mode and enter active mode.

If the slave requests an exit from sniff mode it shall send an LMP_unsniff_req PDU. When the master receives the LMP_unsniff_req PDU it shall enter sniff transition mode and then send an LMP_accepted PDU. When the slave receives the LMP_accepted PDU it shall exit from sniff mode and enter active mode. When the master receives the baseband acknowledgement for the LMP_accepted PDU it shall leave sniff transition mode and enter active mode.



Sequence 89: Slave moved from sniff mode to active mode

4.5.3.3 Sniff Subrating

Once sniff mode has been started, sniff subrating may be initiated by either Link Manager.

The LMP_sniff_subrating_req and LMP_sniff_subrating_res PDUs specify parameters that the peer and initiating device shall use respectively for sniff subrating.

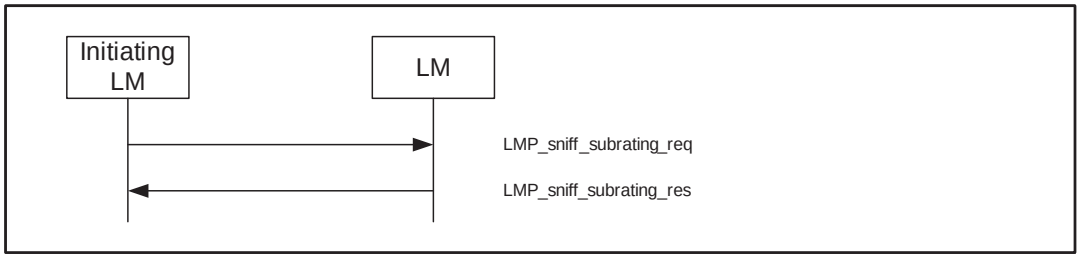
The sniff subrating instant value shall be used to calculate the first sniff subrating anchor point. The sniff subrating instant value shall be a maximum of 2^{16} time slots (40.9 seconds) from the current master clock and shall be a sniff anchor point. The sniff subrating instant value should indicate a clock value in the future with respect to the clock value when the LMP message is first sent.

If the LMP_sniff_subrating_req PDU is sent by the master, the sniff subrating instant value shall be used. The slave device shall reply with an LMP_sniff_subrating_res PDU using the same sniff subrating instant value given by the master.

When the LMP_sniff_subrating_req PDU is sent by the slave, the sniff subrating instant value shall be ignored. The master device shall reply with an

LMP_sniff_subrating_res PDU with the sniff subrating instant value that shall be used for sniff subrating.

The initiating device shall not transition to the new sniff subrating parameters until the sniff subrating instant has passed and the LMP_sniff_subrating_res PDU has been received. The non-initiating device shall remain in sniff mode and shall not transition to the new sniff subrating parameters until after the sniff subrating instant has passed and the baseband acknowledgement of the LMP_sniff_subrating_res PDU has been received.



Sequence 90: LM accepts sniff subrating request

A device shall not send a new LMP_sniff_subrating_req PDU until the previous sniff subrating transaction has completed and the sniff subrating instant has passed.

The maximum clock interval between two sniff subrating anchor points shall be less than the link supervision timeout. If the link supervision timeout needs to be updated to a shorter value than the clock interval between two sniff subrating anchor points, the master shall disable sniff subrating, shall send the LMP_Supervision_Timeout PDU with the new supervision timeout value, and shall start using the new supervision timeout value after receiving a baseband ACK for the LMP_Supervision_Timeout PDU. Upon reception of the LMP_Supervision_Timeout PDU the slave shall disable sniff subrating and shall start using the new supervision timeout value.

The master shall initiate sniff subrating with the max_sniff_subrate parameter less than the new supervision timeout. The slave shall respond with the LMP_sniff_subrating_res PDU with the max_sniff_subrate parameter less than the new supervision timeout.

Note: When changing the link supervision timeout while sniff subrating is enabled, refer to Section 4.5.3.3 on page 295.

4.6 LOGICAL TRANSPORTS

When a connection is first established between two devices the connection consists of the default ACL logical links: ACL-C (for LMP messages) and ACL-U (for L2CAP data.) One or more synchronous logical transports (SCO or eSCO) may then be added. A new logical transport shall not be created if it would cause all slots to be allocated to reserved slots on secondary LT_ADDRs.

4.6.1 SCO Logical Transport

The SCO logical transport reserves slots separated by the SCO interval, T_{sco}. The first slot reserved for the SCO logical transport is defined by T_{sco} and the SCO offset, D_{sco}. See Baseband Specification, Section 8.6.2, on page 167 for details. A device shall initiate a request for HV2 or HV3 packet type only if the other device supports it (bits 12, 13) in its features mask. A device shall initiate CVSD, μ-law or A-law coding or uncoded (transparent) data only if the other device supports the corresponding feature. To avoid problems with a wrap-around of the clock during initialization of the SCO logical transport, the timing control flags parameter is used to indicate how the first SCO slot shall be calculated. Only bit1 of the timing control flags parameter is valid. The SCO link is distinguished from all other SCO links by an SCO handle. The SCO handle zero shall not be used.

M/O	PDU	Contents
O(11)	LMP_SCO_link_req	SCO handle timing control flags D_{sco} T_{sco} SCO packet air mode
O(11)	LMP_remove_SCO_link_req	SCO handle error

Table 4.29: SCO link management PDUs

4.6.1.1 Master Initiates an SCO Link

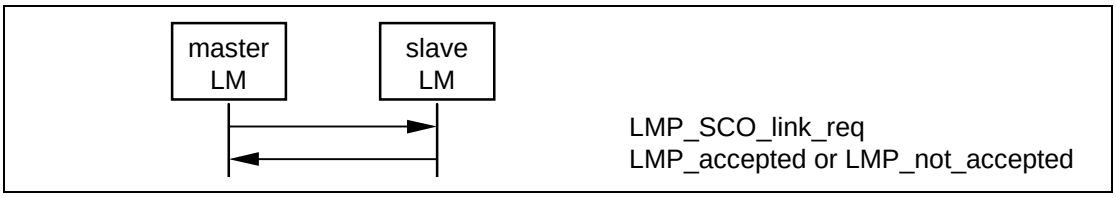
When establishing an SCO link the master sends a request, a LMP_SCO_link_req PDU, with parameters that specify the timing, packet type and coding that will be used on the SCO link. Each of the SCO packet types supports three different voice coding formats on the air-interface: μ-law log PCM, A-law log PCM and CVSD. The air coding by log PCM or CVSD may be deactivated to achieve a transparent synchronous data link at 64 kbits/s.

The slots used for the SCO links are determined by three parameters controlled by the master: T_{sco}, D_{sco} and a flag indicating how the first SCO slot is calculated. After the first slot, the SCO slots follow periodically at an interval of T_{sco}.

If the slave does not accept the SCO link, but is willing to consider another possible set of SCO parameters, it can indicate what it does not accept in the error code field of LMP_not_accepted PDU. The master may then issue a new request with modified parameters.

The SCO handle in the message shall be different from existing SCO link(s).

If the SCO packet type is HV1 the LMP_accepted shall be sent using the DM1 packet.

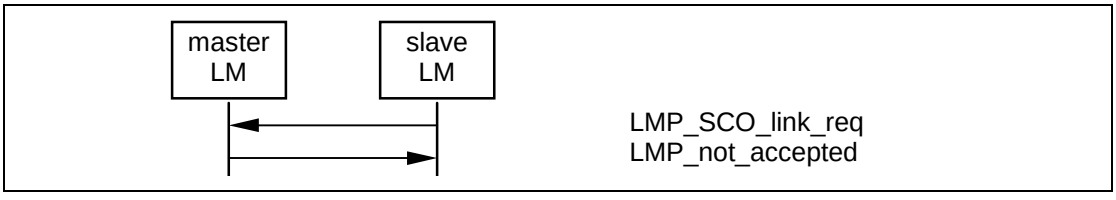


Sequence 91: Master requests an SCO link

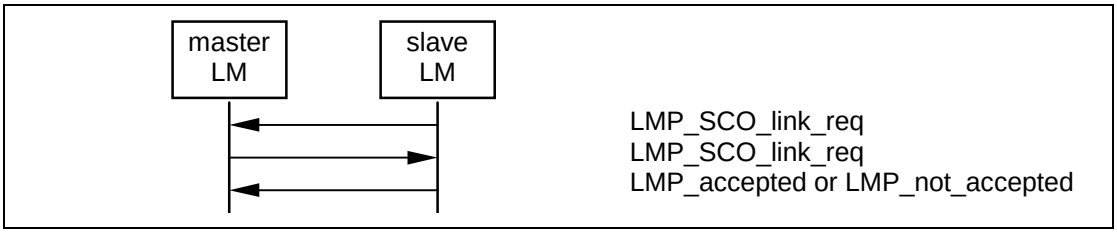
4.6.1.2 Slave Initiates an SCO Link

The slave may initiate the establishment of an SCO link. The slave sends an LMP_SCO_link_req PDU, but the parameters timing control flags and D_{sco} are invalid as well as the SCO handle, that shall be zero. If the master is not capable of establishing an SCO link, it replies with an LMP_not_accepted PDU. Otherwise it sends back an LMP_SCO_link_req PDU. This message includes the assigned SCO handle, D_{sco} and the timing control flags. The master should try to use the same parameters as in the slave request; if the master cannot meet that request, it is allowed to use other values. The slave shall then reply with LMP_accepted or LMP_not_accepted PDU.

If the SCO packet type is HV1 the LMP_accepted shall be sent using the DM1 packet.



Sequence 92: Master rejects slave's request for an SCO link



Sequence 93: Master accepts slave's request for an SCO link

4.6.1.3 Master Requests Change of SCO Parameters

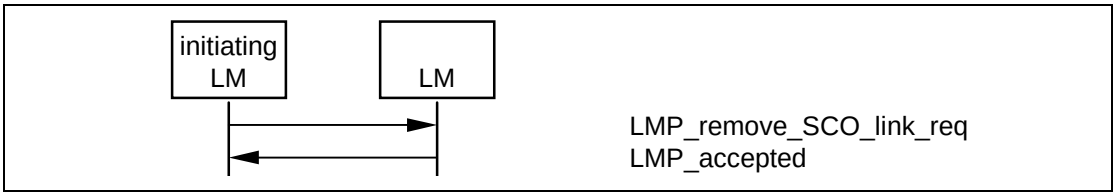
The master sends an LMP_SCO_link_req PDU, where the SCO handle is the handle of the SCO link the master wishes to change parameters for. If the slave accepts the new parameters, it replies with an LMP_accepted PDU and the SCO link will change to the new parameters. If the slave does not accept the new parameters, it shall reply with an LMP_not_accepted PDU and the SCO link is left unchanged. When the slave replies with an LMP_not_accepted PDU it shall indicate in the error code parameter what it does not accept. The master may then try to change the SCO link again with modified parameters. The sequence is the same as in Section 4.6.1.1 on page 297.

4.6.1.4 Slave Requests Change of SCO Parameters

The slave sends an LMP_SCO_link_req PDU, where the SCO handle is the handle of the SCO link to be changed. The parameters timing control flags and D_{sco} are not valid in this PDU. If the master does not accept the new parameters it shall reply with an LMP_not_accepted PDU and the SCO link is left unchanged. If the master accepts the new parameters it shall reply with an LMP_SCO_link_req PDU containing the same parameters as in the slave request. When receiving this message the slave replies with an LMP_not_accepted PDU if it does not accept the new parameters. The SCO link is then left unchanged. If the slave accepts the new parameters it replies with an LMP_accepted PDU and the SCO link will change to the new parameters. The sequence is the same as in Section 4.6.1.2 on page 298.

4.6.1.5 Remove a SCO link

Master or slave may remove the SCO link by sending a request including the SCO handle of the SCO link to be removed and an error code indicating why the SCO link is removed. The receiving side shall respond with an LMP_accepted PDU.



Sequence 94: SCO link removed

Note that the slave shall use the initialization flag appropriate to the master's Bluetooth clock. See section 8.6.3, “eSCO,” on page 169.

4.6.2 eSCO Logical Transport

After an ACL link has been established, one or more extended SCO (eSCO) links can be set up to the remote device. The eSCO links are similar to SCO links using timing control flags, an interval T_{eSCO} and an offset D_{eSCO}. Only bit1 of the timing control flags parameter is valid. As opposed to SCO links, eSCO links have a configurable data rate that may be asymmetric, and can be set up to provide limited retransmissions of lost or damaged packets inside a retransmission window of size W_{eSCO}. The D_{eSCO} shall be based on CLK.

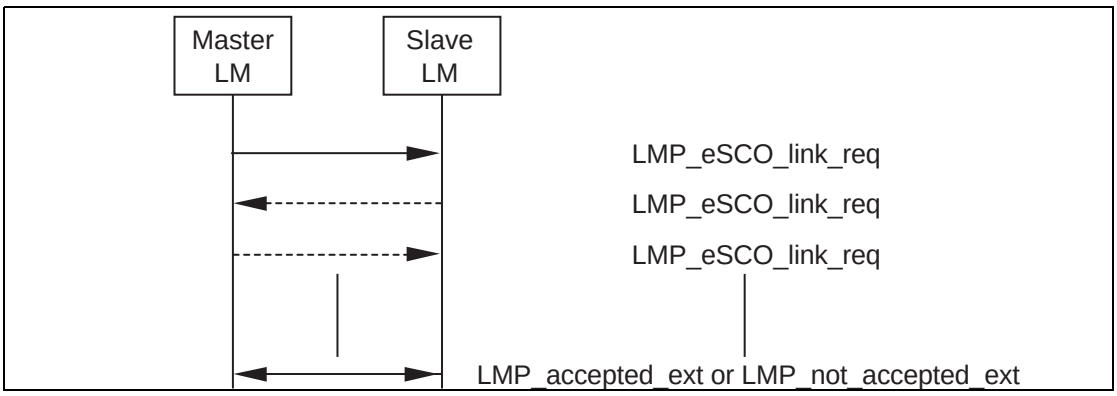
M/O	PDU	Contents
O(31)	LMP_eSCO_link_req	eSCO handle eSCO LT_ADDR timing control flags D_{eSCO} T_{eSCO} W_{eSCO} eSCO packet type M->S eSCO packet type S->M packet length M->S packet length S->M air mode negotiation state
O(31)	LMP_remove_eSCO_link_req	eSCO handle error

Table 4.30: PDUs used for managing the eSCO links

The parameters D_{eSCO}, T_{eSCO}, W_{eSCO}, eSCO packet type M->S, eSCO packet type S->M, packet length M->S, packet length S->M are henceforth referred to as the negotiable parameters.

4.6.2.1 Master initiates an eSCO link

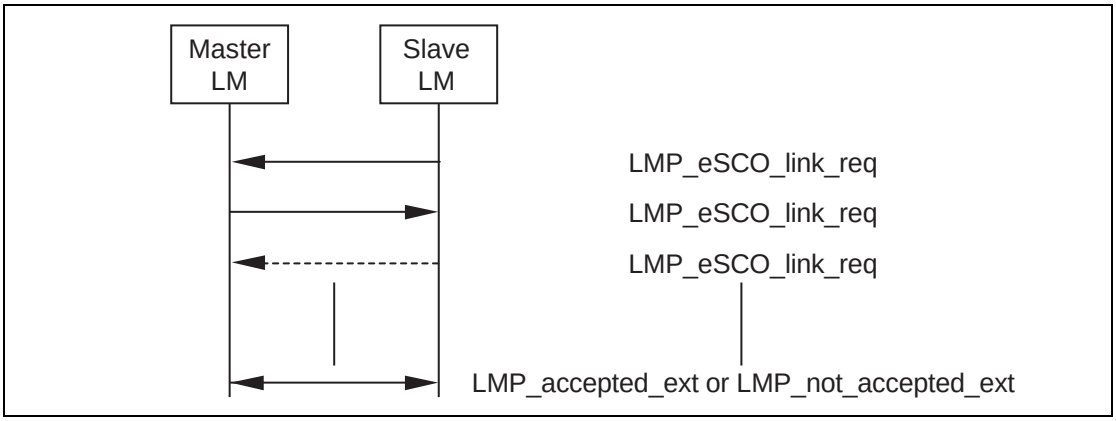
When establishing an eSCO link the master sends an LMP_eSCO_link_req PDU specifying all parameters. The slave may accept this with an LMP_accepted_ext PDU, reject it with an LMP_not_accepted_ext PDU, or respond with its own LMP_eSCO_link_req specifying alternatives for some or all parameters. The slave shall not negotiate the eSCO handle or eSCO LT_ADDR parameters. The negotiation of parameters continues until the master or slave either accepts the latest parameters with an LMP_accepted_ext PDU, or terminates the negotiation with an LMP_not_accepted_ext PDU. The negotiation shall use the procedures defined in Section 4.6.2.5 on page 303.



Sequence 95: Master requests an eSCO link

4.6.2.2 Slave Initiates an eSCO Link

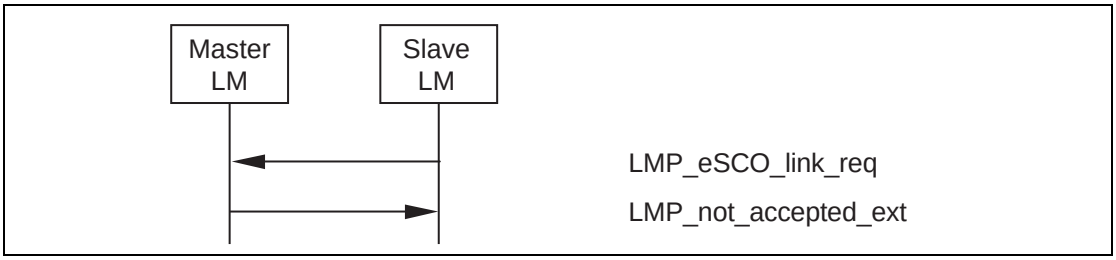
When attempting to establish an eSCO link the slave shall send an LMP_eSCO_link_req PDU specifying all parameters, with the exception of eSCO LT_ADDR and eSCO handle, which are invalid. The latter shall be set to zero. The master may respond to this with an LMP_eSCO_link_req PDU, filling in these missing parameters, and potentially changing the other requested parameters. The slave can accept this with an LMP_accepted_ext PDU, or respond with a further LMP_eSCO_link_req PDU specifying alternatives for some or all of the parameters. The negotiation of parameters continues until the master or slave either accepts the latest parameters with an LMP_accepted_ext PDU, or terminates the negotiation with an LMP_not_accepted_ext PDU.



Sequence 96: Slave requests an eSCO link.

Note that the slave should use the initialization flag appropriate to the master's Bluetooth clock. See Baseband Specification Section 8.6.3.

The master may reject the request immediately with an LMP_not_accepted_ext PDU. The negotiation shall use the procedures defined in Section 4.6.2.5 on page 303.



Sequence 97: Master rejects slave's request for an eSCO link.

4.6.2.3 Master or slave requests change of eSCO parameters

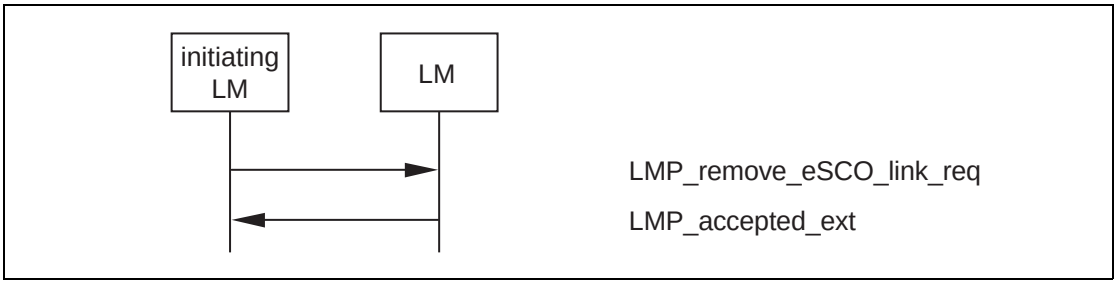
The master or slave may request a renegotiation of the eSCO parameters. The master or slave shall send an LMP_eSCO_link_req PDU with the eSCO handle of the eSCO link the device wishes to renegotiate. The remote device may accept the changed parameters immediately with LMP_accepted_ext PDU, or the negotiation may be continued with further LMP_eSCO_link_req PDUs until the master or slave accepts the latest parameters with an LMP_accepted_ext PDU or terminates the negotiation with an LMP_not_accepted_ext PDU. In the case of termination with an LMP_not_accepted_ext PDU, the eSCO link continues on the previously negotiated parameters.

The sequence is the same as in Section 4.6.2.2 on page 301.

During re-negotiation, the eSCO LT_ADDR and eSCO handle shall not be re-negotiated and shall be set to the originally negotiated values. The negotiation shall use the procedures defined in Section 4.6.2.5 on page 303.

4.6.2.4 Remove an eSCO Link

Either the master or slave may remove the eSCO link by sending a request including the eSCO handle of the eSCO link to be removed and a error code indicating why the eSCO link is removed. The receiving side shall respond with an LMP_accepted_ext PDU. The slave shall shut down its eSCO logical transport before sending its PDU (LMP_remove_eSCO_link_req for slave initiated removal, LMP_accepted_ext for master initiated). The master shall not reassign the eSCO LT_ADDR until the end of the removal procedure. If, for some reason, such as the slave being out of range, the procedure cannot be completed, the master shall not reassign the eSCO LT_ADDR until the primary LT_ADDR is available for reuse (supervision timeout).



Sequence 98: eSCO link removed

4.6.2.5 Rules for the LMP negotiation and renegotiation

Rule 1: the negotiation_state shall be set to 0 by the initiating LM. After the initial LMP_eSCO_link_req is sent the negotiation_state shall not be set to 0.

Rule 2: if the bandwidth (defined as 1600 times the packet length in bytes divided by T_{eSCO} in slots) for either RX or TX or the air_mode cannot be accepted the device shall send LMP_not_accepted_ext with the appropriate error code.

Rule 3: Bandwidth and air_mode are not negotiable and shall not be changed for the duration of the negotiation. Once one side has rejected the negotiation (with LMP_not_accepted_ext) a new negotiation may be started with different bandwidth and air_mode parameters.

Rule 4: if the parameters will cause a latency violation (T_{eSCO} + W_{eSCO} + reserved synchronous slots > allowed local latency) the device should propose new parameters that shall not cause a reserved slot violation or latency violation for the device that is sending the parameters. In this case the negotiation_state shall be set to 3. Otherwise the device shall send LMP_not_accepted_ext.

Rule 5: once a device has received an LMP_eSCO_link_req with the negotiation_state set to 3 (latency violation), the device shall not propose any combination of packet type, T_{eSCO}, and W_{eSCO} that will give an equal or larger

latency than the combination that caused the latency violation for the other device.

Rule 6: if the parameters cause both a reserved slot violation and a latency violation or if the parameters are not supported and a latency violation occurs, then the device shall set the negotiation_state to 3 (latency violation).

Rule 7: if the parameters will cause a reserved slot violation the device should propose new parameters that shall not cause a reserved slot violation. In this case the negotiation_state shall be set to 2. Otherwise the device shall send LMP_not_accepted_ext.

Rule 8: If the requested parameters are not supported the device should propose a setting that is supported, and set the negotiation_state to 4. If it is not possible to find such a parameter set, the device shall send LMP_not_accepted_ext.

Rule 9: when proposing new parameters for reasons other than a latency violation, reserved slot violation, or configuration not supported, the negotiation_state shall be set to 1.

4.6.2.6 Negotiation State Definitions

Reserved Slot Violation: a reserved slot violation is when the receiving LM cannot setup the requested eSCO logical transport because the eSCO reserved slots would overlap with other regularly scheduled slots (e.g. other synchronous reserved slots, sniff anchor points, or park beacons).

Latency Violation: a latency violation is when the receiving LM cannot setup the requested eSCO logical transport because the latency (W_{eSCO}+T_{eSCO} + reserved synchronous slots) is greater than the maximum allowed latency.

Configuration not supported: The combination of parameters requested is not inside the supported range for the device.

4.7 TEST MODE

LMP has PDUs to support different test modes used for certification and compliance testing of the Bluetooth radio and baseband. See "Test Methodology" on page 411 in Vol. 3 for a detailed description of these test modes.

4.7.1 Activation and Deactivation of Test Mode

The activation may be carried out locally (via a HW or SW interface), or using the air interface.

- For activation over the air interface, entering the test mode shall be locally enabled for security and type approval reasons. The implementation of this local enabling is not subject to standardization.

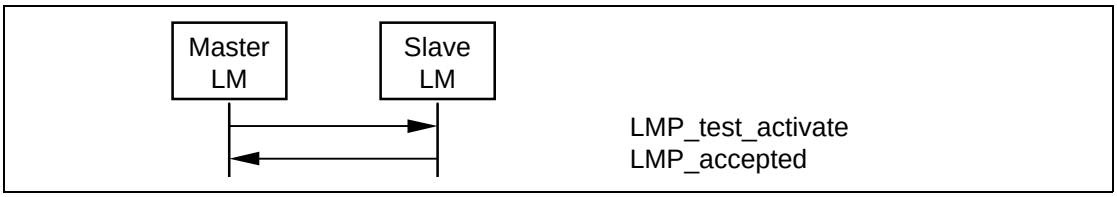
  The tester sends an LMP command that shall force the DUT to enter test mode. The DUT shall terminate all normal operation before entering the test mode.

  The DUT shall return an LMP_Accepted PDU on reception of an activation command. An LMP_Not_Accepted PDU shall be returned if the DUT is not locally enabled.

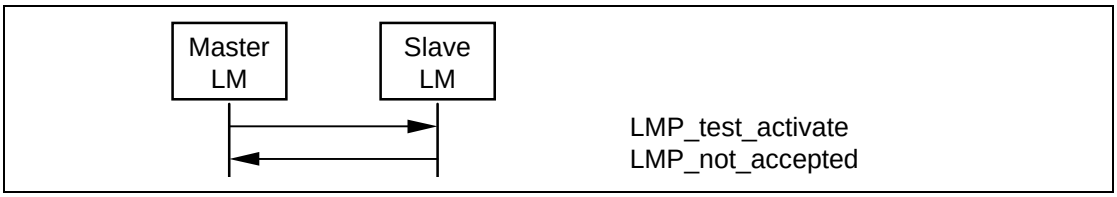
- If the activation is performed locally using a HW or SW interface, the DUT shall terminate all normal operation before entering the test mode.

  Until a connection to the tester exists, the device shall perform page scan and inquiry scan. Extended scan activity is recommended.

The test mode is activated by sending an LMP_test_activate PDU to the device under test (DUT). The DUT is always the slave. The lm shall be able to receive this message anytime. If entering test mode is locally enabled in the DUT it shall respond with an LMP_accepted PDU and test mode is entered. Otherwise the DUT responds with an LMP_not_accepted PDU and the DUT remains in normal operation. The error code in the LMP_not_accepted PDU shall be *PDU not allowed*.



Sequence 99: Activation of test mode successful.



Sequence 100: Activation of test mode fails. Slave is not allowed to enter test mode.

The test mode can be deactivated in two ways. Sending an LMP_test_control PDU with the test scenario set to "exit test mode" exits the test mode and the slave returns to normal operation still connected to the master. Sending an LMP_detach PDU to the DUT ends the test mode and the connection.

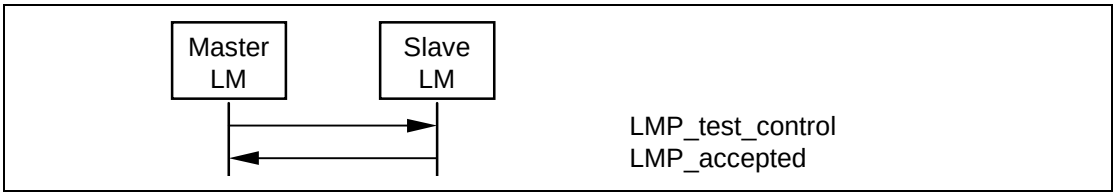
4.7.2 Control of Test Mode

Control and configuration is performed using special LMP commands (see Section 4.7.3 on page 306). These commands shall be rejected if the Bluetooth device is not in test mode. In this case, an LMP_not_accepted shall be returned. The DUT shall return an LMP_accepted on reception of a control command when in test mode.

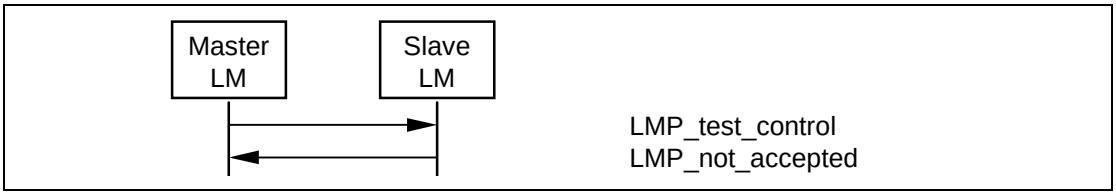
A Bluetooth device in test mode shall ignore all LMP commands not related to control of the test mode. LMP commands dealing with power control and the request for LMP features (LMP_features_req), and adaptive frequency hopping (LMP_set_AFH, LMP_channel_classification_req and LMP_channel_classification) are allowed in test mode; the normal procedures are also used to test the adaptive power control.

The DUT shall leave the test mode when an LMP_Detach command is received or an LMP_test_control command is received with test scenario set to 'exit test mode'.

When the DUT has entered test mode, the PDU LMP_test_control PDU can be sent to the DUT to start a specific test. This PDU is acknowledged with an LMP_accepted PDU. If a device that is not in test mode receives an LMP_test_control PDU it responds with an LMP_not_accepted PDU, where the error code shall be *PDU not allowed*.



Sequence 101: Control of test mode successful.



Sequence 102: Control of test mode rejected since slave is not in test mode.

4.7.3 Summary of Test Mode PDUs

Table 4.13 lists all LMP messages used for test mode. To ensure that the contents of LMP_test_control PDU are suitably whitened (important when sent in

transmitter mode), all parameters listed in Table 4.31 on page 307 are XORed with 0x55 before being sent.

LMP PDU	PDU number	Possible Direction	Contents	Position in Payload
LMP_test_activate	56	m → s		
LMP_test_control	57	m → s	test scenario hopping mode TX frequency RX frequency power control mode poll period packet type length of test data	2 3 4 5 6 7 8 9-10
LMP_detach	7	m → s		
LMP_accepted	3	m ← s		
LMP_not_accepted	4	m ← s		

Name	Length (bytes)	Type	Unit	Detailed
Test scenario	1	u_int8		0 Pause Test Mode 1 Transmitter test – 0 pattern 2 Transmitter test – 1 pattern 3 Transmitter test – 1010 pattern 4 Pseudorandom bit sequence 5 Closed Loop Back – ACL packets 6 Closed Loop Back – Synchro-nous packets 7 ACL Packets without whitening 8 Synchronous Packets without whitening 9 Transmitter test – 1111 0000 pat-tern 10–254 reserved 255 Exit Test Mode The value is XORed with 0x55.
Hopping mode	1	u_int8		0 RX/TX on single frequency 1 Normal hopping 2 Reserved 3 Reserved 4 Reserved 5–255 reserved The value is XORed with 0x55.
TX frequency (for DUT)	1	u_int8		f = [2402 + k] MHz The value is XORed with 0x55.

Table 4.31: Parameters used in LMP_Test_Control PDU

Name	Length (bytes)	Type	Unit	Detailed
RX frequency (for DUT)	1	u_int8		f = [2402 + k] MHz The value is XORed with 0x55.
Power control mode	1	u_int8		0 fixed TX output power 1 adaptive power control The value is XORed with 0x55.
Poll period	1	u_int8	1.25 ms	The value is XORed with 0x55.
Packet type	1	u_int8		Bits 3-0 numbering as in packet header, see Baseband Specification Bits 7-4 0: ACL/SCO 1: eSCO 2: Enhanced Data Rate ACL 3: Enhanced Data Rate eSCO 4-15: reserved Other values are reserved The value is XORed with 0x55.
length of test sequence (=length of user data in Baseband Specification)	2	u_int16	1 byte	unsigned binary number The value is XORed with 0x5555.

Table 4.31: Parameters used in LMP_Test_Control PDU

The control PDU is used for both transmitter and loop back tests. The following restrictions apply for the parameter settings:

Parameter	**Restrictions Transmitter Test**	**Restrictions Loopback Test**
TX frequency	$0 \leq k \leq 78$[#2541]	$0 \leq k \leq 78$
RX frequency	same as TX frequency	$0 \leq k \leq 78$
Poll period		not applicable (set to 0)
Length of test sequence	Depends on packet type. See table 6.9 and 6.10 in the Baseband specification	For ACL and SCO packets: not applicable (set to 0) For eSCO packets: [see table 6.10 in the Baseband specification]

5 SUMMARY

5.1 PDU SUMMARY

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
Escape 1	variable	124	DM1	m ↔ s	extended op code	2
					variable	3-?
Escape 2	variable	125	DM1	m ↔ s	extended op code	2
					variable	3-?
Escape 3	variable	126	DM1	m ↔ s	extended op code	2
					variable	3-?
Escape 4	variable	127	DM1	m ↔ s	extended op code	2
					variable	3-?
LMP_accepted	2	3	DM1/DV	m ↔ s	op code	2
LMP_accepted_ext	4	127/01	DM1	m ↔ s	escape op code	3
					extended op code	4
LMP_au_rand	17	11	DM1	m ↔ s	random number	2-17
LMP_auto_rate	1	35	DM1/DV	m ↔ s	-	
LMP_channel _classification_req	7	127/16	DM1	m → s	AFH_reporting _mode	3
					AFH_min_interval	4-5
					AFH_max_interval	6-7
LMP_channel _classification	12	127/17	DM1	m ← s	AFH_channel_ classification	3 – 12
LMP_clkoffset_req	1	5	DM1/DV	m → s	–	
LMP_clkoffset_res	3	6	DM1/DV	m ← s	clock offset	2–3
LMP_comb_key	17	9	DM1	m ↔ s	random number	2-17
LMP_decr_power _req	2	32	DM1/DV	m ↔ s	for future use	2
LMP_detach	2	7	DM1/DV	m ↔ s	error code	2
LMP_DHkey_ Check	17	65	DM1	m ↔ s	Confirmation Value	2-17

Table 5.1: Coding of the different LM PDUs

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_encapsulated_header	4	61	DM1	m ↔ s	encapsulated major type	2
					encapsulated minor type	3
					encapsulated pay-load length	4
LMP_encapsulated_payload	17	62	DM1	m ↔ s	encapsulated data	2-17
LMP_encryption_key_size_mask_req	1	58	DM1	m → s		
LMP_encryption_key_size_mask_res	3	59	DM1	m ← s	key size mask	2-3
LMP_encryption_key_size_req	2	16	DM1/DV	m ↔ s	key size	2
LMP_encryption_mode_req	2	15	DM1/DV	m ↔ s	encryption mode	2
LMP_eSCO_link_req	16	127/12	DM1	m↔s	eSCO handle	3
					eSCO LT_ADDR	4
					timing control flags	5
					D_{eSCO}	6
					T_{eSCO}	7
					W_{eSCO}	8
					eSCO packet type M->S	9
					eSCO packet type S->M	10
					packet length M->S	11-12
					packet length S->M	13-14
					air mode	15
					negotiation state	16
LMP_features_req	9	39	DM1/DV	m ↔ s	features	2-9

Table 5.1: Coding of the different LM PDUs

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_features_req_ext	12	127/03	DM1	m ↔ s	features page	3
					max supported page	4
					extended features	5-12
LMP_features_res	9	40	DM1/DV	m ↔ s	features	2-9
LMP_features_res_ext	12	127/04	DM1	m ↔ s	features page	3
					max supported page	4
					extended features	5-12
LMP_host_connection_req	1	51	DM1/DV	m ↔ s	–	
LMP_hold	7	20	DM1/DV	m ↔ s	hold time	2-3
					hold instant	4-7
LMP_hold_req	7	21	DM1/DV	m ↔ s	hold time	2-3
					hold instant	4-7
LMP_incr_power_req	2	31	DM1/DV	m ↔ s	for future use	2
LMP_in_rand	17	8	DM1	m ↔ s	random number	2-17
LMP_IO_Capability_req	5	127/25	DM1	m ↔ s	IO_capabilities	3
					OOB Authentication Data	4
					Authentication_Requirement	5
LMP_IO_Capability_res	5	127/26	DM1	m ↔ s	IO_capabilities	3
					OOB Authentication Data	4
					Authentication_Requirement	5
LMP_pause_encryption_req	2	127/23	DM1	m ↔ s	–	
LMP_resume_encryption_req	2	127/24	DM1	m ← s	–	
LMP_max_power	1	33	DM1/DV	m ↔ s	–	
LMP_max_slot	2	45	DM1/DV	m ↔ s	max slots	2

Table 5.1: Coding of the different LM PDUs

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_max_slot_req	2	46	DM1/DV	m ↔ s	max slots	2
LMP_min_power	1	34	DM1/DV	m ↔ s	–	
LMP_modify_beacon	11 or 13	28	DM1	m → s	timing control flags	2
					D_B	3-4
					T_B	5-6
					N_B	7
					Δ_B	8
					D_{access}	9
					T_{access}	10
					$N_{acc\text{-}slots}$	11
					N_{poll}	12
					M_{access}	13:0-3
					access scheme	13:4-7
LMP_name_req	2	1	DM1/DV	m ↔ s	name offset	2
LMP_name_res	17	2	DM1	m ↔ s	name offset	2
					name length	3
					name fragment	4-17
LMP_numeric_comparison_failed	2	127/27	DM1	m ↔ s	–	
LMP_not_accepted	3	4	DM1/DV	m ↔ s	op code	2
					error code	3
LMP_not_accepted_ext	5	127/02	DM1	m ↔ s	escape op code	3
					extended op code	4
					error code	5
LMP_packet_type_table_req	3	127/11	DM1	m ↔ s	packet type table	3
LMP_page_mode_req	3	53	DM1/DV	m ↔ s	paging scheme	2
					paging scheme settings	3

Table 5.1: Coding of the different LM PDUs

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_page_scan_ mode_req	3	54	DM1/DV	m ↔ s	paging scheme	2
					paging scheme set-tings	3
LMP_park_req	17	25	DM1	m ↔ s	timing control flags	2
					D_B	3-4
					T_B	5-6
					N_B	7
					Δ_B	8
					PM_ADDR	9
					AR_ADDR	10
					N_{Bsleep}	11
					D_{Bsleep}	12
					D_{access}	13
					T_{access}	14
					$N_{acc-slots}$	15
					N_{poll}	16
					M_{access}	170-3
					access scheme	v
LMP_passkey_ failed	2	127/28	DM1	m ↔ s	–	
LMP_oob _failed	2	127/29	DM1	m ↔ s		
LMP_keypress _notification	3	127/30	DM1	m ↔ s	Notification Type	2
LMP_preferred_rate	2	36	DM1/DV	m ↔ s	data rate	2
LMP_quality_of _service	4	41	DM1/DV	m → s	poll interval	2-3
					N_{BC}	4
LMP_quality_o f_service_req	4	42	DM1/DV	m ↔ s	poll interval	2-3
					N_{BC}	4

Table 5.1: Coding of the different LM PDUs

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_remove_eSCO_link_req see Note 4 on page 319	4	127/ 13	DM1	m↔s	eSCO handle	3
					error code	4
LMP_remove_SCO_link_req	3	44	DM1/DV	m ↔ s	SCO handle	2
					error code	3
LMP_SCO_link_req	7	43	DM1/DV	m ↔ s	SCO handle	2
					timing control flags	3
					D_{sco}	4
					T_{sco}	5
					SCO packet	6
					air mode	7
LMP_set_AFH	16	60	DM1	m → s	AFH_instant	2-5
					AFH_mode	6
					AFH_channel_map	7-16
LMP_set_broadcast_scan_window	4 or 6	27	DM1	m → s	timing control flags	2
					D_B	3-4
					broadcast scan window	5-6
LMP_setup_complete	1	49	DM1	m ↔ s	-	
LMP_Simple_Pairing_Confirm	17	63	DM1	m ↔ s	Commitment Value	2-17
LMP_Simple_Pairing_Number	17	64	DM1	m ↔ s	Nonce Value	2-17
LMP_slot_offset	9	52	DM1/DV	m ↔ s	slot offset	2-3
					BD_ADDR	4-9
LMP_sniff_req	10	23	DM1	m ↔ s	timing control flags	2
					D_{sniff}	3-4
					T_{sniff}	5-6
					sniff attempt	7-8
					sniff timeout	9-10

Table 5.1: Coding of the different LM PDUs

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_sres	5	12	DM1/DV	m ↔ s	authentication response	2-5
LMP_start _encryption_req	17	17	DM1	m → s	random number	2-17
LMP_stop _encryption_req	1	18	DM1/DV	m → s	–	
LMP_supervision _timeout	3	55	DM1/DV	m →s	supervision timeout	2-3
LMP_switch_req	5	19	DM1/DV	m ↔ s	switch instant	2-5
LMP_temp_rand	17	13	DM1	m → s	random number	2-17
LMP_temp_key	17	14	DM1	m → s	key	2-17
LMP_test_activate	1	56	DM1/DV	m → s	–	
LMP_test_control	10	57	DM1	m → s	test scenario	2
					hopping mode	3
					TX frequency	4
					RX frequency	5
					power control mode	6
					poll period	7
					packet type	8
					length of test data	9-10
LMP_timing _accuracy_req	1	47	DM1/DV	m ↔ s	–	
LMP_timing _accuracy_res	3	48	DM1/DV	m ↔ s	drift	2
					jitter	3
LMP_unit_key	17	10	DM1	m ↔ s	key	2-17

Table 5.1: Coding of the different LM PDUs

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_unpark_BD_ADDR_req	variable	29	DM1	m → s	timing control flags	2
					D_B	3-4
					LT_ADDR 1st unpark	5:0-2
					LT_ADDR 2nd unpark	5:4-6
					BD_ADDR 1st unpark	6-11
					BD_ADDR 2nd unpark	12-17

Table 5.1: Coding of the different LM PDUs

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_unpark_PM_ADDR_req	variable	30	DM1	m → s	timing control flags	2
					D_B	3-4
					LT_ADDR 1st unpark	5:0-3
					LT_ADDR 2nd unpark	5:4-7
					PM_ADDR 1st unpark	6
					PM_ADDR 2nd unpark	7
					LT_ADDR 3rd unpark	8:0-3
					LT_ADDR 4th unpark	8:4-7
					PM_ADDR 3rd unpark	9
					PM_ADDR 4th unpark	10
					LT_ADDR 5th unpark	11:0-3
					LT_ADDR 6th unpark	11:4-7
					PM_ADDR 6th unpark	12
					PM_ADDR 6th unpark	13
					LT_ADDR 7th unpark	14:0-3
					PM_ADDR 7th unpark	15
LMP_unsniff_req	1	24	DM1/DV	m ↔ s	-	
LMP_use_semi_permanent_key	1	50	DM1/DV	m → s	-	
LMP_version_req	6	37	DM1/DV	m ↔ s	VersNr	2
					CompId	3-4
					SubVersNr	5-6

Table 5.1: Coding of the different LM PDUs

LMP PDU	Length (bytes)	op code	Packet type	Possible direction	Contents	Position in payload
LMP_version_res	6	38	DM1/DV	m ↔ s	VersNr	2
					CompId	3-4
					SubVersNr	5-6
LMP_sniff_ subrating_req	9	127/21	DM1	m ↔ s	max_sniff_subrate	3
					min_sniff_mode_ timeout	4-5
					sniff_subrating_inst ant	6-9
LMP_sniff_ subrating_res	9	127/22	DM1	m ↔ s	max_sniff_subrate	3
					min_sniff_mode_ timeout	4-5
					sniff_subrating_ instant	6-9
LMP_power_ control_req	3	127/31	DM1/DV	m ↔ s	power_adjustment_ request	3
LMP_power_ control_res	3	127/32	DM1/DV	m ↔ s	power_adjustment_ response	3

Table 5.1: Coding of the different LM PDUs

Note1: For LMP_set_broadcast_scan_window, LMP_modify_beacon, LMP_unpark_BD_ADDR_req and LMP_unpark_PM_ADDR_req the parameter D_B is optional. This parameter is only present if bit0 of *timing control flags* is 1. If the parameter is not included, the position in payload for all parameters following D_B are decreased by 2.

Note2: For LMP_unpark_BD_ADDR the LT_ADDR and the BD_ADDR of the 2nd unparked slave are optional. If only one slave is unparked *LT_ADDR 2nd unpark* shall be zero and *BD_ADDR 2nd unpark* is left out.

Note3: For LMP_unpark_PM_ADDR the LT_ADDR and the PM_ADDR of the 2nd – 7th unparked slaves are optional. If N slaves are unparked, the fields up to and including the Nth unparked slave are present. If N is odd, the *LT_ADDR (N+1)th unpark* shall be zero. The length of the message is x + 3N/2 if N is even and x + 3(N+1)/2 -1 if N is odd, where x = 2 or 4 depending on if the D_B is included or Not (See Note1).

Note4: Parameters coincide with their namesakes in LMP_<remove>_SCO_link_req apart from the following:

1. eSCO_LT_ADDR - the eSCO connection will be active on an additional LT_ADDR that needs to be defined. The master is allowed to re-assign an active eSCO link to a different LT_ADDR.
2. D_{eSCO}, T_{eSCO} - as per LMP_SCO_link_req but with a greater flexibility in values (e.g. no longer fixed with respect to HV1, HV2, and HV3 packet choice).
3. W_{eSCO} - the eSCO retransmission window size (in slots)
4. packet type and packet length may be prescribed differently in Master-to-Slave or Slave-to-Master directions for asynchronous eSCO links
5. packet length (in bytes) - eSCO packet types no longer have fixed length
6. negotiation state – this is used to better enable the negotiation of the negotiable parameters: D_{eSCO}, T_{eSCO}, W_{eSCO}, eSCO packet type M->S, eSCO packet type S->M, packet length M->S, packet length S->M. When responding to an eSCO link request with a new suggestion for these parameters, this flag may be set to 1 to indicate that the last received negotiable parameters are possible, but the new parameters specified in the response eSCO link request would be preferable, to 2 to indicate that the last received negotiable parameters are not possible as they cause a reserved slot violation or to 3 to indicate that the last received negotiable parameters would cause a latency violation. The flag shall be set to zero in the initiating LMP_eSCO_link_req.

5.2 PARAMETER DEFINITIONS

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
access scheme	1	u_int4		0: polling technique 1-15: Reserved	

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
AFH_channel_classification	10	multiple bytes	-	This parameter contains 40 2-bit fields. The nth (numbering from 0) such field defines the classification of channels *2n* and *2n+1*, other than the 39th field which just contains the classification of channel 78. Each field interpreted as an integer whose values indicate: 0 = unknown 1 = good 2 = reserved 3 = bad	
AFH_channel_map	10	multiple bytes	-	If *AFH_mode* is *AFH_enabled*, this parameter contains 79 1-bit fields, otherwise the contents are reserved. The nth (numbering from 0) such field (in the range 0 to 78) contains the value for channel *n*. Bit 79 is reserved (set to 0 when transmitted and ignored when received) The 1-bit field is interpreted as follows: 0: channel *n* is unused 1: channel *n* is used	
AFH_instant	4	u_int32	slots	Bits 27:1 of the Bluetooth master clock value at the time of switching hop sequences. Shall be even.	
AFH_max_interval	2	u_int16	slots	Range is 0x0640 to 0xBB80 slots (1 to 30s)	
AFH_min_interval	2	u_int16	slots	Range is 0x0640 to 0xBB80 slots (1 to 30s)	
AFH_mode	1	u_int8	-	0: AFH_disabled 1: AFH_enabled 2-255: Reserved	

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
AFH_reporting_mode	1	u_int8	-	0: AFH_reporting_disabled 1: AFH_reporting_enabled 2-255: reserved	
air mode	1	u_int8		0: μ-law log 1: A-law log 2: CVSD 3: transparent data 4-255: Reserved	See Table 5.3 on page 331
AR_ADDR	1	u_int8			
Authentication_Requirements	1	u_int8		0x00:MITM Protection Not Required – No Bonding 0x01: MITM Protection Required – No Bonding 0x02: MITM Protection Not Required – Dedicated Bonding 0x03: MITM Protection Required – Dedicated Bonding 0x04: MITM Protection Not Required – General Bonding 0x05: MITM Protection Required – General Bonding 0x06 to 0xFF– Reserved	
authentication response	4	multiple bytes			
BD_ADDR	6	multiple bytes		BD_ADDR of the sending device	
broadcast scan window	2	u_int16	slots		
clock offset	2	u_int16	1.25ms	($CLKN_{16-2}$ slave - $CLKN_{16-2}$ master) mod 2^{15} MSbit of second byte not used.	
CompId	2	u_int16		see Bluetooth Assigned Numbers)	

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
Commitment value	16	Multiple bytes		Little Endian format	
Confirmation value	16	Multiple bytes		Little Endian format	
D_{access}	1	u_int8	slots		
data rate	1	u_int8		When in Basic Rate mode: bit0 = 0: use FEC bit0 = 1: do not use FEC bit1-2=0: No packet-size preference available bit1-2=1: use 1-slot packets bit1-2=2: use 3-slot packets bit1-2=3: use 5-slot packets When in Enhanced Data Rate mode: bit3-4=0: use DM1 packets bit3-4=1: use 2 Mbps packets bit3-4=2: use 3 Mbps packets bit3-4=3: reserved bit5-6=0: No packet-size preference available bit5-6=1: use 1-slot packets bit5-6=2: use 3-slot packets bit5-6=3: use 5-slot packets bit7: Reserved - shall be zero	
D_B	2	u_int16	slots		
Δ_B	1	u_int8	slots		
D_{Bsleep}	1	u_int8			
D_{eSCO}	1	u_int8	slots	Valid range is 0 - 254 slots	See Table 5.3 on page 331

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
drift	1	u_int8	ppm		
D_{sco}	1	u_int8	slots	Only even values are valid[1]	0 to T_{sco} -2
D_{sniff}	2	u_int16	slots	Only even values are valid[1]	0 to (T_{sniff} - 2)
encapsulated data	16	Multiple bytes		MSBs zero padded when data is less than 16 bytes. Little Endian format	
encapsulated major type	1	u_int8		See Table 5.3, “LMP encapsulated,” on page 331	
encapsulated minor type	1	u_int8		See Table 5.3, “LMP encapsulated,” on page 331	
encapsulated payload length	1	u_int8		See Table 5.3, “LMP encapsulated,” on page 331	
encryption mode	1	u_int8		0: no encryption 1: encryption 2: encryption 3 -255: Reserved	
error code	1	u_int8		See Section Part D on page 339	
escape op code	1	u_int8		Identifies which escape op code is being acknowl-edged: range 124-127	
eSCO handle	1	u_int8			
eSCO LT_ADDR	1	u_int8		Logical transport address for the eSCO logical trans-port. The range is extended to 8 bits com-pared with the normal LT_ADDR field: range 0-7.	0 - 7

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
eSCO packet type	1	u_int8		0x00: NULL/POLL 0x07: EV3 0x0C: EV4 0x0D: EV5 0x26: 2-EV3 0x2C: 2-EV5 0x37: 3-EV3 0x3D: 3-EV5 Other values are reserved	If the value is 0x00 the POLL packet shall be used by the master, the NULL packet shall be used by the slave. See Table 5.3 on page 331
extended features	8	multiple bytes		One page of extended features	
extended op code	1	u_int8		Which extended op code is being acknowledged	
features	8	multiple bytes		See Table 3.2 on page 225	
features page	1	u_int8		Identifies which page of extended features is being requested. 0 means standard features 1-255 other feature pages	
hold instant	4	u_int32	slots	Bits 27:1 of the master Bluetooth clock value	
hold time	2	u_int16	slots	Only even values are valid[1]	0x0014 to 0x8000; shall not exceed (supervisionTO * 0.999)
IO_ Capabilities	1	u_int8		0: Display only 1: Display YesNo 2: KeyboardOnly 3: NoInputNoOutput 4-255: reserved	
jitter	1	u_int8	µs		
key	16	multiple bytes			
key size	1	u_int8	byte		

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
key size mask	2	u_int16		Bit mask of supported broadcast encryption key sizes: least significant bit is support for length 1, and so on. The bit shall be one if the key size is supported.	
LT_ADDR	1	u_int4			
M_{access}	1	u_int4		number of access windows	
max slots	1	u_int8	slots		
max_sniff_subrate	1	u_int8	sub-rate		1-255
min_sniff_mode_timeout	2	u_int16	slots		0x0000 to 0xFFFE (even)
max supported page	1	u_int8		Highest page of extended features which contains a non-zero bit for the originating device. Range 0-255	
$N_{acc-slots}$	1	u_int8	slots		
name fragment	14	multiple bytes		UTF-8 characters.	
name length	1	u_int8	bytes		
name offset	1	u_int8	bytes		
N_B	1	u_int8			
N_{BC}	1	u_int8			
N_{Bsleep}	1	u_int8			

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
negotiation state	1	u_int8		0: Initiate negotiation 1: the latest received set of negotiable parameters were possible but these parameters are preferred. 2: the latest received set of negotiable parameters would cause a reserved slot violation. 3: the latest received set of negotiable parameters would cause a latency violation. 4: the latest received set of of negotiable parameters are not supported. Other values are reserved	
N_{poll}	1	u_int8			
Nonce Value	16	Multiple bytes		Little Endian Format	
Notification Value	1	U_int8		0=passkey entry started 1=passkey digit entered 2=passkey digit erased 3=passkey cleared 4=passkey entry completed 5-255: reserved	
OOB Authentication Data	1	u_int8		0: No OOB Authentication Data received 1: OOB Authentication Data received 2-255: reserved	
op code	1	u_int8			

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
packet length	2	uint16	bytes	Length of the eSCO payload 0 for POLL/NULL 1-30 for EV3 1-120 for EV4 1-180 for EV5 1-60 for 2-EV3 1-360 for 2-EV5 1-90 for 3-EV3 1-540 for 3-EV5 Other values are invalid	See Table 5.3 on page 331
packet type table	1	u_int8		0: 1Mbps only 1: 2/3 Mbps 2-255: reserved	0-1
paging scheme	1	u_int8		0: mandatory scheme 1-255: Reserved	
paging scheme settings	1	u_int8		For mandatory scheme: 0: R0 1: R1 2: R2 3-255: Reserved	
PM_ADDR	1	u_int8			
poll interval	2	u_int16	slots	Only even values are valid[1]	0x0006 to 0x1000
power_ adjustment_ request	1	u_int8		0: decrement power one step 1: increment power one step 2: increase to maximum power 3-255: reserved for future use	
power_ adjustment_ response	1	u_int8		bit0-1: GFSK bit2-3: DQPSK bit4-5: 8DPSK bit6-7: reserved Each 2-bit value is defined as follows 0: not supported 1: changed one step, (not min or max) 2: max power 3: min power	

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
random number	16	multiple bytes			
reserved(n)	n	u_int8		Reserved for future use – shall be 0 when transmitted, ignore value when received	
SCO handle	1	u_int8			
SCO packet	1	u_int8		0: HV1 1: HV2 2: HV3 3-255: Reserved	
slot offset	2	u_int16	µs	0 ≤ slot offset < 1250	
sniff attempt	2	u_int16	received slots	Number of receive slots	1 to $T_{sniff}/2$
sniff_subrating_instant	4	u_int32	slots	Bits 27:1 of the master Bluetooth clock value	
sniff timeout	2	u_int16	received slots	Number of receive slots	0 to 0x0028
SubVersNr	2	u_int16		Defined by each company	
supervision timeout	2	u_int16	slots	0 means an infinite timeout	0 and 0x0190 to 0xFFFF
switch instant	4	u_int32	slots	Bits 27:1 of the master Bluetooth clock value	
T_{access}	1	u_int8	slots		
T_B	2	u_int16	slots		
T_{eSCO}	1	u_int8	slots	Valid range is 4 – 254 slots	See Table 5.3 on page 331
timing control flags	1	u_int8		bit0 = 0: o timing change bit0 = 1: timing change bit1 = 0: use initialization 1 bit1 = 1: use initialization 2 bit2 = 0: access window bit2 = 1: no access window bit3-7: Reserved	
T_{sco}	1	u_int8	slots	Only even values are valid[1]	2 and 6

Table 5.2: Parameters in LM PDUs.

Name	Length (bytes)	Type	Unit	Detailed	Mandatory range
T_{sniff}	2	u_int16	slots	Only even values are valid[1]	0x0006 to 0x0540; shall not exceed (supervi-sionTO * 0.999)
VersNr	1	u_int8		See Bluetooth Assigned Numbers	
W_{eSCO}	1	u_int8	slots	Number of slots in the retransmission window Valid range is 0 – 254 slots	See Table 5.3 on page 331

Table 5.2: Parameters in LM PDUs.

1. If a device receives an LMP PDU with an odd value in this parameter field the PDU should be rejected with an error code of *invalid LMP parameters*.

5.3 LMP ENCAPSULATED

Name	Major Type	Minor Type	Payload Length	Detailed
P-192 Public Key	1	1	48	X, Y format Bytes 23-0: X co-ordinate Bytes 47-24: Y co-ordinate Little Endian Format

Table 5.3: LMP encapsulated

	Single Slot Packets	3-Slot Packets
D_{eSCO}	0 to **T_{eSCO}**-2 (even)	0 to **T_{eSCO}**-2 (even)
T_{eSCO}	EV3: 6 2-EV3: 6-12 (even) 3-EV3: 6-18 (even)	EV4: 16 EV5: 16 2-EV5: 16 3-EV5: 16
W_{eSCO}	0, 2, and 4	0 and 6
packet length M->S	10*T_{eSCO} /2	10*T_{eSCO} /2
packet length S->M	10*T_{eSCO} /2	10*T_{eSCO} /2
air mode	At least one of A-law, mu-law, CVSD, transparent	transparent

Table 5.4: Mandatory parameter ranges for eSCO packet types

5.4 DEFAULT VALUES

Devices shall use these values before anything else has been negotiated:

Parameter	Value
AFH_mode	AFH_disabled
AFH_reporting_mode	AFH_reporting_disabled
drift	250
jitter	10
max slots	1
poll interval	40

Table 5.5: Device default values

6 LIST OF FIGURES

Figure 1.1: Link Manager Protocol signaling layer. 211
Figure 2.1: Transmission of a message from master to slave. 213
Figure 2.2: Payload body when LMP PDUs are sent. 214
Figure 2.3: Symbols used in sequence diagrams. 216
Figure 4.1: Connection establishment. 228
Sequence 1: Connection closed by sending LMP_detach. 230
Sequence 2: A device requests a change of the other device's TX power. 231
Sequence 3: The TX power cannot be increased. 231
Sequence 4: The TX power cannot be decreased. 231
Sequence 5: A device responds to a power control request. 233
Sequence 6: Master Enables AFH. 235
Sequence 7: Master disables AFH. 235
Sequence 8: Master Updates AFH. 236
Sequence 9: Channel classification reporting. 238
Sequence 10: Setting the link supervision timeout. 239
Sequence 11: A notifies B to enable CQDDR 240
Sequence 12: B sends A a preferred packet type 240
Sequence 13: Master notifies slave of quality of service. 241
Sequence 14: Device accepts new quality of service 242
Sequence 15: Device rejects new quality of service. 242
Sequence 16: Negotiation for page mode. 243
Sequence 17: Negotiation for page scan mode 243
Sequence 18: Device allows Remote Device to use a maximum number of slots. 244
Sequence 19: Device requests a maximum number of slots. Remote Device accepts. 244
Sequence 20: Device requests a maximum number of slots. Remote Device rejects. 244
Sequence 21: Packet type table change is rejected. 245
Sequence 22: Packet type table change is accepted. 246
Sequence 23: Sending an encapsulated PDU 247
Sequence 24: Authentication. Claimant has link key. 248
Sequence 25: Authentication fails. Claimant has no link key. 248
Sequence 26: Pairing accepted. Responder has a variable PIN. Initiator has a variable or fixed PIN. 250
Sequence 27: Responder has a fixed PIN and initiator has a variable PIN. 250
Sequence 28: Both devices have a fixed PIN. 250
Sequence 29: Responder rejects pairing. 251
Sequence 30: Creation of the link key. 252

Sequence 31: Successful change of the link key 252
Sequence 32: Change of the link key not possible since the other device uses a unit key. 252
Sequence 33: Change to a temporary link key 254
Sequence 34: Link key changed to the semi-permanent link key 254
Sequence 35: Negotiation for encryption mode 256
Sequence 36: Encryption key size negotiation successful 257
Sequence 37: Encryption key size negotiation failed 257
Sequence 38: Start of encryption 257
Sequence 39: Stop of encryption 258
Sequence 40: Master-initiated pausing of encryption 259
Sequence 41: Slave-initiated pausing of encryption 259
Sequence 42: Initiating Master LM resumes encryption 260
Sequence 43: Initiating Slave LM resumes encryption 260
Sequence 44: Request for supported encryption key sizes. 262
Sequence 45: IO Capability Exchange 263
Sequence 46: IO Capability exchange with transaction Collision (master transmitting LMP_io_capability_req first) and resolution 264
Sequence 47: IO Capability exchange with transaction Collision (slave transmitting LMP_io_capability_req first) and resolution 264
Sequence 48: Public key exchange 265
Sequence 49: Numeric Comparison Authentication: Commitment check success 266
Sequence 50: Numeric Comparison Authentication: Commitment check failure 267
Sequence 51: Authentication stage 1 Numeric Comparison Failure on Initiator Side 267
Sequence 52: Authentication stage 1: Numeric Comparison Failure on Responder Side 268
Sequence 53: Authentication Passkey entry 269
Sequence 54: Authentication Passkey Entry: Commitment check failure on the Responder side 269
Sequence 55: Authentication Passkey Entry: Commitment check failure on the Initiator side 270
Sequence 56: Authentication stage 1 Passkey Entry Failure on Initiator Side 270
Sequence 57: Authentication Stage 1 Passkey Entry Failure on Responding Side 271
Sequence 58: Keypress Notifications 271
Sequence 59: Authentication OOB: Only one device is OOB-r capable ... 272
Sequence 60: Authentication stage 1 OOB: Commitment check failure on the Responder side 273
Sequence 61: Authentication stage 1 OOB: Commitment check failure on the Initiator side 273

Sequence 62: Authentication stage 1 OOB: OOB information not available on the Initiator side 274
Sequence 63: DHKey check 274
Sequence 64: DHKey check: Check failure on the Responder side 275
Sequence 65: DHKey check: check failure on the Initiator side 275
Sequence 66: The requested device supports timing accuracy information. 276
Sequence 67: The requested device does not support timing accuracy information. 276
Sequence 68: Clock offset requested. 277
Sequence 69: Request for LMP version. 278
Sequence 70: Request for supported features. 279
Sequence 71: Request for extended features. 279
Sequence 72: Device's name requested and it responses. 280
Figure 4.2: Slot offset for role switch 281
Sequence 73: Slot offset information is sent 281
Sequence 74: Role switch (slave initiated). 283
Sequence 75: Role switch (master initiated). 283
Sequence 76: Master forces slave into hold mode 285
Sequence 77: Slave forces master into hold mode 285
Sequence 78: Negotiation for hold mode 286
Sequence 79: Slave accepts to enter park state. 290
Sequence 80: Slave rejects to enter into park state 290
Sequence 81: Slave requests to enter park state and accepts master's beacon parameters 291
Sequence 82: Master rejects slave's request to enter park state 291
Sequence 83: Slave requests to enter park state, but rejects master's beacon parameters 291
Sequence 84: Master notifies all slaves of increase in broadcast capacity 291
Sequence 85: Master modifies beacon parameters 292
Sequence 86: Master unparks slaves addressed with their BD_ADDR 292
Sequence 87: Master unparks slaves addressed with their PM_ADDR 293
Sequence 88: Negotiation for sniff mode 294
Sequence 89: Slave moved from sniff mode to active mode 295
Sequence 90: LM accepts sniff subrating request 296
Sequence 91: Master requests an SCO link 298
Sequence 92: Master rejects slave's request for an SCO link 298
Sequence 93: Master accepts slave's request for an SCO link 299
Sequence 94: SCO link removed 300
Sequence 95: Master requests an eSCO link 301
Sequence 96: Slave requests an eSCO link. 302
Sequence 97: Master rejects slave's request for an eSCO link. 302

Sequence 98: eSCO link removed 303
Sequence 99: Activation of test mode successful. 305
Sequence 100: Activation of test mode fails. Slave is not allowed to enter test mode. 305
Sequence 101: Control of test mode successful. 306
Sequence 102: Control of test mode rejected since slave is not in test mode. 306

7 LIST OF TABLES

Table 2.1: General response messages. 217
Table 3.1: Feature Definitions 218
Table 3.2: Feature mask definitions (page 0) 225
Table 3.3: Extended feature mask definition (page 1) 227
Table 4.1: Connection establishment PDU 229
Table 4.2: Detach PDU 229
Table 4.3: Legacy Power control PDU 230
Table 4.4: Enhanced Power control PDU 230
Table 4.5: AFH PDU 234
Table 4.6: Set supervision timeout PDU 239
Table 4.7: Quality-driven change of the data rate PDU 240
Table 4.8: Quality of Service PDU 241
Table 4.9: Paging scheme request PDU 243
Table 4.10: Multi-slot packet control PDU 244
Table 4.11: Enhanced Data Rate PDUs 245
Table 4.12: Encapsulated LMP PDUs 246
Table 4.13: Authentication PDUs 248
Table 4.14: Pairing PDUs 249
Table 4.15: Change link key PDU 252
Table 4.16: Change current link key PDU 253
Table 4.17: Encryption handling PDU 255
Table 4.18: Encryption key size request PDU 262
Table 4.19: Request limited timing PDU 276
Table 4.20: PDUs used for clock offset request. 277
Table 4.21: PDUs used for LMP version request. 278
Table 4.22: PDUs used for features request. 279
Table 4.23: Name request PDUs 280
Table 4.24: Role switch PDU 281
Table 4.25: Role switch PDU 282
Table 4.26: Hold mode PDUs 284
Table 4.27: PDUs used for park state 288
Table 4.28: Sniff mode PDUs 293
Table 4.29: SCO link management PDUs 297
Table 4.30: PDUs used for managing the eSCO links 300
Table 4.31: Parameters used in LMP_Test_Control PDU 307
Table 5.1: Coding of the different LM PDUs 310
Table 5.2: Parameters in LM PDUs. 320
Table 5.3: LMP encapsulated 331
Table 5.4: Mandatory parameter ranges for eSCO packet types 331
Table 5.5: Device default values 332

ERROR CODES

Ford Ex. 1052

CONTENTS

1 Overview of Error Codes 343
1.1 Usage Descriptions 343
1.2 HCI Command Errors 343
1.3 List of Error Codes 344
2 Error Code Descriptions 347
2.1 Unknown HCI Command (0X01) 347
2.2 Unknown Connection Identifier (0X02) 347
2.3 Hardware Failure (0X03) 347
2.4 Page Timeout (0X04) 347
2.5 Authentication Failure (0X05) 347
2.6 PIN or key Missing (0X06) 347
2.7 Memory Capacity Exceeded (0X07) 347
2.8 Connection Timeout (0X08) 348
2.9 Connection Limit Exceeded (0X09) 348
2.10 Synchronous Connection Limit to a Device Exceeded (0X0A) 348
2.11 ACL Connection Already Exists (0X0B) 348
2.12 Command Disallowed (0X0C) 348
2.13 Connection Rejected due to Limited Resources (0X0D) 348
2.14 Connection Rejected due to Security Reasons (0X0E) 348
2.15 Connection Rejected due to Unacceptable BD_ADDR (0X0F) 349
2.16 Connection Accept Timeout Exceeded (0X10) 349
2.17 Unsupported Feature or Parameter Value (0X11) 349
2.18 Invalid HCI Command Parameters (0X12) 349
2.19 Remote User Terminated Connection (0X13) 349
2.20 Remote Device Terminated Connection due to Low Resources (0X14) 350
2.21 Remote Device Terminated Connection due to Power Off (0X15) 350
2.22 Connection Terminated by Local Host (0X16) 350
2.23 Repeated Attempts (0X17) 350
2.24 Pairing not Allowed (0X18) 350
2.25 Unknown LMP PDU (0X19) 350
2.26 Unsupported Remote Feature / Unsupported LMP Feature (0X1A) 350
2.27 SCO Offset Rejected (0X1B) 350
2.28 SCO Interval Rejected (0X1C) 351
2.29 SCO Air Mode Rejected (0X1D) 351
2.30 Invalid LMP Parameters (0X1E) 351
2.31 Unspecified Error (0X1F) 351
2.32 Unsupported LMP Parameter Value (0X20) 351

2.33 Role Change Not Allowed (0X21) 351
2.34 LMP Response Timeout / LL Response Timeout (0X22) 351
2.35 LMP Error Transaction Collision (0X23) 352
2.36 LMP PDU Not Allowed (0X24) 352
2.37 Encryption Mode Not Acceptable (0X25) 352
2.38 Link Key cannot be Changed (0X26) 352
2.39 Requested QoS Not Supported (0X27) 352
2.40 Instant Passed (0X28) 352
2.41 Pairing with Unit Key Not Supported (0X29) 352
2.42 Different Transaction Collision (0x2A) 352
2.43 QoS Unacceptable Parameter (0X2C) 352
2.44 QoS Rejected (0X2D) 353
2.45 Channel Assessment Not Supported (0X2E) 353
2.46 Insufficient Security (0X2F) 353
2.47 Parameter out of Mandatory Range (0X30) 353
2.48 Role Switch Pending (0X32) 353
2.49 Reserved Slot Violation (0X34) 353
2.50 Role Switch Failed (0X35) 353
2.51 Extended Inquiry Response Too Large (0x36) 353
2.52 Simple Pairing Not Supported By Host (0X37) 354
2.53 Host Busy–Pairing(0X38) 354
2.54 Connection Rejected Due To No Suitable Channel Found (0X39) 354
2.55 Controller Busy (0X3A) 354
2.56 Unacceptable Connection Interval (0X3B) 354
2.57 Directed Advertising Timeout (0X3C) 354
2.58 Connection Terminated Due To MIC Failure (0X3D) 354
2.59 Connection Failed To Be Established (0X3E) 355
2.60 MAC Connection Failed (0x3F) 355

1 OVERVIEW OF ERROR CODES

This document lists the various possible error codes. When a command fails, or an LMP, LL, or AMP message needs to indicate a failure, error codes are used to indicate the reason for the error. Error codes have a size of one octet.

1.1 USAGE DESCRIPTIONS

The purpose of this section is to give descriptions of how the error codes should be used. It is beyond the scope of this document to give detailed descriptions of all situations where error codes can be used, especially as this is implementation dependent.

1.2 HCI COMMAND ERRORS

If an HCI Command that should generate a Command Complete event generates an error then this error shall be reported in the Command Complete event.

If an HCI Command that sent a Command Status with the error code ‘Success’ to the Host before processing may find an error during execution then the error may be reported in the normal completion command for the original command or in a Command Status event.

Some HCI Commands may generate errors that need to be reported to the Host, but there is insufficient information to determine how the command would normally be processed. In this case, two events can be used to indicate this to the Host, the Command Complete event and Command Status events. Which of the two events is used is implementation-dependent.

1.3 LIST OF ERROR CODES

The error code of 0x00 means Success. The possible range of failure error codes is 0x01-0xFF. Section 2 provides an error code usage description for each failure error code.

Values marked as “Reserved for Future Use”, can be used in future versions of the specification. A Host shall consider any error code that it does not explicitly understand equivalent to the “Unspecified Error (0x1F).”

Error Code	Name
0x00	Success
0x01	Unknown HCI Command
0x02	Unknown Connection Identifier
0x03	Hardware Failure
0x04	Page Timeout
0x05	Authentication Failure
0x06	PIN or Key Missing
0x07	Memory Capacity Exceeded
0x08	Connection Timeout
0x09	Connection Limit Exceeded
0x0A	Synchronous Connection Limit To A Device Exceeded
0x0B	ACL Connection Already Exists
0x0C	Command Disallowed
0x0D	Connection Rejected due to Limited Resources
0x0E	Connection Rejected Due To Security Reasons
0x0F	Connection Rejected due to Unacceptable BD_ADDR
0x10	Connection Accept Timeout Exceeded
0x11	Unsupported Feature or Parameter Value
0x12	Invalid HCI Command Parameters
0x13	Remote User Terminated Connection
0x14	Remote Device Terminated Connection due to Low Resources
0x15	Remote Device Terminated Connection due to Power Off
0x16	Connection Terminated By Local Host
0x17	Repeated Attempts
0x18	Pairing Not Allowed
0x19	Unknown LMP PDU
0x1A	Unsupported Remote Feature / Unsupported LMP Feature
0x1B	SCO Offset Rejected

Table 1.1: List of Possible Error Codes

Error Code	Name
0x1C	SCO Interval Rejected
0x1D	SCO Air Mode Rejected
0x1E	Invalid LMP Parameters
0x1F	Unspecified Error
0x20	Unsupported LMP Parameter Value
0x21	Role Change Not Allowed
0x22	LMP Response Timeout / LL Response Timeout
0x23	LMP Error Transaction Collision
0x24	LMP PDU Not Allowed
0x25	Encryption Mode Not Acceptable
0x26	Link Key cannot be Changed
0x27	Requested QoS Not Supported
0x28	Instant Passed
0x29	Pairing With Unit Key Not Supported
0x2A	Different Transaction Collision
0x2B	Reserved
0x2C	QoS Unacceptable Parameter
0x2D	QoS Rejected
0x2E	Channel Classification Not Supported
0x2F	Insufficient Security
0x30	Parameter Out Of Mandatory Range
0x31	Reserved
0x32	Role Switch Pending
0x33	Reserved
0x34	Reserved Slot Violation
0x35	Role Switch Failed
0x36	Extended Inquiry Response Too Large
0x37	Secure Simple Pairing Not Supported By Host.
0x38	Host Busy - Pairing
0x39	Connection Rejected due to No Suitable Channel Found
0x3A	Controller Busy
0x3B	Unacceptable Connection Interval
0x3C	Directed Advertising Timeout
0x3D	Connection Terminated due to MIC Failure

Table 1.1: List of Possible Error Codes

Error Code	Name
0x3E	Connection Failed to be Established
0x3F	MAC Connection Failed

Table 1.1: List of Possible Error Codes

2 ERROR CODE DESCRIPTIONS

2.1 UNKNOWN HCI COMMAND (0X01)

The Unknown HCI Command error code indicates that the Controller does not understand the HCI Command Packet OpCode that the Host sent. The OpCode given might not correspond to any of the OpCodes specified in this document, or any vendor-specific OpCodes, or the command may not have been implemented.

2.2 UNKNOWN CONNECTION IDENTIFIER (0X02)

The Unknown Connection Identifier error code indicates that a command was sent from the Host that should identify a connection, but that connection does not exist.

2.3 HARDWARE FAILURE (0X03)

The Hardware Failure error code indicates to the Host that something in the Controller has failed in a manner that cannot be described with any other error code. The meaning implied with this error code is implementation dependent.

2.4 PAGE TIMEOUT (0X04)

The Page Timeout error code indicates that a page timed out because of the Page Timeout configuration parameter. This error code may occur only with the Remote_Name_Request and Create_Connection commands.

2.5 AUTHENTICATION FAILURE (0X05)

The Authentication Failure error code indicates that pairing or authentication failed due to incorrect results in the pairing or authentication procedure. This could be due to an incorrect PIN or Link Key.

2.6 PIN OR KEY MISSING (0X06)

The PIN or Key Missing error code is used when pairing failed because of a missing PIN, or authentication failed because of a missing Key.

2.7 MEMORY CAPACITY EXCEEDED (0X07)

The Memory Capacity Exceeded error code indicates to the Host that the Controller has run out of memory to store new parameters.

2.8 CONNECTION TIMEOUT (0X08)

The Connection Timeout error code indicates that the link supervision timeout has expired for a given connection.

2.9 CONNECTION LIMIT EXCEEDED (0X09)

The Connection Limit Exceeded error code indicates that an attempt to create another connection failed because the Controller is already at its limit of the number of connections it can support. The number of connections a device can support is implementation dependent.

2.10 SYNCHRONOUS CONNECTION LIMIT TO A DEVICE EXCEEDED (0X0A)

The Synchronous Connection Limit to a Device Exceeded error code indicates that the Controller has reached the limit to the number of synchronous connections that can be achieved to a device. The number of synchronous connections a device can support is implementation dependent.

2.11 ACL CONNECTION ALREADY EXISTS (0X0B)

The ACL Connection Already Exists error code indicates that an attempt to create a new ACL Connection to a device when there is already a connection to this device.

2.12 COMMAND DISALLOWED (0X0C)

The Command Disallowed error code indicates that the command requested cannot be executed because the Controller is in a state where it cannot process this command at this time. This error shall not be used for command OpCodes where the error code Unknown HCI Command is valid.

2.13 CONNECTION REJECTED DUE TO LIMITED RESOURCES (0X0D)

The Connection Rejected Due To Limited Resources error code indicates that an incoming connection was rejected due to limited resources.

2.14 CONNECTION REJECTED DUE TO SECURITY REASONS (0X0E)

The Connection Rejected Due To Security Reasons error code indicates that a connection was rejected due to security requirements not being fulfilled, like authentication or pairing.

2.15 CONNECTION REJECTED DUE TO UNACCEPTABLE BD_ADDR (0X0F)

The Connection Rejected due to Unacceptable BD_ADDR error code indicates that a connection was rejected because this device does not accept the BD_ADDR. This may be because the device will only accept connections from specific BD_ADDRs.

2.16 CONNECTION ACCEPT TIMEOUT EXCEEDED (0X10)

The Connection Accept Timeout Exceeded error code indicates that the Connection Accept Timeout has been exceeded for this connection attempt.

2.17 UNSUPPORTED FEATURE OR PARAMETER VALUE (0X11)

The Unsupported Feature Or Parameter Value error code indicates that a feature or parameter value in the HCI command is not supported. This error code shall not be used in an LMP PDU.

2.18 INVALID HCI COMMAND PARAMETERS (0X12)

The Invalid HCI Command Parameters error code indicates that at least one of the HCI command parameters is invalid.

This shall be used when:

- the parameter total length is invalid.
- a command parameter is an invalid type.
- a connection identifier does not match the corresponding event.
- a parameter value shall be even.
- a parameter is outside of the specified range.
- two or more parameter values have inconsistent values.

Note: An invalid type can be, for example, when an SCO connection handle is used where an ACL connection handle is required.

2.19 REMOTE USER TERMINATED CONNECTION (0X13)

The Remote User Terminated Connection error code indicates that the user on the remote device terminated the connection.

2.20 REMOTE DEVICE TERMINATED CONNECTION DUE TO LOW RESOURCES (0X14)

The Remote Device Terminated Connection due to Low Resources error code indicates that the remote device terminated the connection because of low resources.

2.21 REMOTE DEVICE TERMINATED CONNECTION DUE TO POWER OFF (0X15)

The Remote Device Terminated Connection due to Power Off error code indicates that the remote device terminated the connection because the device is about to power off.

2.22 CONNECTION TERMINATED BY LOCAL HOST (0X16)

The Connection Terminated By Local Host error code indicates that the local device terminated the connection.

2.23 REPEATED ATTEMPTS (0X17)

The Repeated Attempts error code indicates that the Controller is disallowing an authentication or pairing procedure because too little time has elapsed since the last authentication or pairing attempt failed.

2.24 PAIRING NOT ALLOWED (0X18)

The Pairing Not Allowed error code indicates that the device does not allow pairing. For example, when a device only allows pairing during a certain time window after some user input allows pairing.

2.25 UNKNOWN LMP PDU (0X19)

The Unknown LMP PDU error code indicates that the Controller has received an unknown LMP OpCode.

2.26 Unsupported Remote Feature / Unsupported LMP Feature (0X1A)

The Unsupported Remote Feature error code indicates that the remote device does not support the feature associated with the issued command or LMP PDU.

2.27 SCO OFFSET REJECTED (0X1B)

The SCO Offset Rejected error code indicates that the offset requested in the LMP_SCO_link_req PDU has been rejected.

2.28 SCO INTERVAL REJECTED (0X1C)

The SCO Interval Rejected error code indicates that the interval requested in the LMP_SCO_link_req PDU has been rejected.

2.29 SCO AIR MODE REJECTED (0X1D)

The SCO Air Mode Rejected error code indicates that the air mode requested in the LMP_SCO_link_req PDU has been rejected.

2.30 INVALID LMP PARAMETERS (0X1E)

The Invalid LMP Parameters error code indicates that some LMP PDU parameters were invalid. This shall be used when:

- the PDU length is invalid.
- a parameter value shall be even.
- a parameter is outside of the specified range.
- two or more parameters have inconsistent values.

2.31 UNSPECIFIED ERROR (0X1F)

The Unspecified Error error code indicates that no other error code specified is appropriate to use.

2.32 UNSUPPORTED LMP PARAMETER VALUE (0X20)

The Unsupported LMP Parameter Value error code indicates that an LMP PDU contains at least one parameter value that is not supported by the Controller at this time. This is normally used after a long negotiation procedure, for example during an LMP_hold_req, LMP_sniff_req and LMP_encryption_key_size_req PDU exchanges.

2.33 ROLE CHANGE NOT ALLOWED (0X21)

The Role Change Not Allowed error code indicates that a Controller will not allow a role change at this time.

2.34 LMP RESPONSE TIMEOUT / LL RESPONSE TIMEOUT (0X22)

The LMP Response Timeout / LL Response Timeout error code indicates that an LMP transaction failed to respond within the LMP response timeout or an LL transaction failed to respond within the LL response timeout.

2.35 LMP ERROR TRANSACTION COLLISION (0X23)

The LMP Error Transaction Collision error code indicates that an LMP transaction has collided with the same transaction that is already in progress.

2.36 LMP PDU NOT ALLOWED (0X24)

The LMP PDU Not Allowed error code indicates that a Controller sent an LMP PDU with an OpCode that was not allowed.

2.37 ENCRYPTION MODE NOT ACCEPTABLE (0X25)

The Encryption Mode Not Acceptable error code indicates that the requested encryption mode is not acceptable at this time.

2.38 LINK KEY CANNOT BE CHANGED (0X26)

The Link Key cannot be Changed error code indicates that a link key cannot be changed because a fixed unit key is being used.

2.39 REQUESTED QoS NOT SUPPORTED (0X27)

The Requested QoS Not Supported error code indicates that the requested Quality of Service is not supported.

2.40 INSTANT PASSED (0X28)

The Instant Passed error code indicates that an LMP PDU or LL PDU that includes an instant cannot be performed because the instant when this would have occurred has passed.

2.41 PAIRING WITH UNIT KEY NOT SUPPORTED (0X29)

The Pairing With Unit Key Not Supported error code indicates that it was not possible to pair as a unit key was requested and it is not supported.

2.42 DIFFERENT TRANSACTION COLLISION (0X2A)

The Different Transaction Collision error code indicates that an LMP transaction was started that collides with an ongoing transaction.

2.43 QoS UNACCEPTABLE PARAMETER (0X2C)

The QoS Unacceptable Parameter error code indicates that the specified quality of service parameters could not be accepted at this time, but other parameters may be acceptable.

2.44 QoS REJECTED (0X2D)

The QoS Rejected error code indicates that the specified quality of service parameters cannot be accepted and QoS negotiation should be terminated.

2.45 CHANNEL ASSESSMENT NOT SUPPORTED (0X2E)

The Channel Assessment Not Supported error code indicates that the Controller cannot perform channel assessment because it is not supported.

2.46 INSUFFICIENT SECURITY (0X2F)

The Insufficient Security error code indicates that the HCI command or LMP PDU sent is only possible on an encrypted link.

2.47 PARAMETER OUT OF MANDATORY RANGE (0X30)

The Parameter Out Of Mandatory Range error code indicates that a parameter value requested is outside the mandatory range of parameters for the given HCI command or LMP PDU.

2.48 ROLE SWITCH PENDING (0X32)

The Role Switch Pending error code indicates that a Role Switch is pending. This can be used when an HCI command or LMP PDU cannot be accepted because of a pending role switch. This can also be used to notify a peer device about a pending role switch.

2.49 RESERVED SLOT VIOLATION (0X34)

The Reserved Slot Violation error code indicates that the current Synchronous negotiation was terminated with the negotiation state set to Reserved Slot Violation.

2.50 ROLE SWITCH FAILED (0X35)

The Role Switch Failed error code indicates that a role switch was attempted but it failed and the original piconet structure is restored. The switch may have failed because the TDD switch or piconet switch failed.

2.51 EXTENDED INQUIRY RESPONSE TOO LARGE (0X36)

The Extended Inquiry Response Too Large error code indicates that the extended inquiry response, with the requested requirements for FEC, is too large to fit in any of the packet types supported by the Controller.

2.52 SIMPLE PAIRING NOT SUPPORTED BY HOST (0X37)

The Secure Simple Pairing Not Supported by Host error code indicates that the IO capabilities request or response was rejected because the sending Host does not support Secure Simple Pairing even though the receiving Link Manager does.

2.53 HOST BUSY–PAIRING(0X38)

The Host Busy - Pairing error code indicates that the Host is busy with another pairing operation and unable to support the requested pairing. The receiving device should retry pairing again later.

2.54 CONNECTION REJECTED DUE TO NO SUITABLE CHANNEL FOUND (0X39)

The Connection Rejected due to No Suitable Channel Found error code indicates that the Controller could not calculate an appropriate value for the Channel selection operation.

2.55 CONTROLLER BUSY (0X3A)

The Controller Busy error code indicates that the operation was rejected because the Controller was busy and unable to process the request.

2.56 UNACCEPTABLE CONNECTION INTERVAL (0X3B)

The Unacceptable Connection Interval error code indicates that the remote device terminated the connection because of an unacceptable connection interval.

2.57 DIRECTED ADVERTISING TIMEOUT (0X3C)

The Directed Advertising Timeout error code indicates that directed advertising completed without a connection being created.

2.58 CONNECTION TERMINATED DUE TO MIC FAILURE (0X3D)

The Connection Terminated Due to MIC Failure error code indicates that the connection was terminated because the Message Integrity Check (MIC) failed on a received packet.

2.59 CONNECTION FAILED TO BE ESTABLISHED (0X3E)

The Connection Failed to be Established error code indicates that the LL initiated a connection but the connection has failed to be established.

2.60 MAC CONNECTION FAILED (0X3F)

The MAC of the 802.11 AMP was requested to connect to a peer, but the connection failed.

HOST CONTROLLER INTERFACE FUNCTIONAL SPECIFICATION

This document describes the functional specification for the Host Controller Interface (HCI).
The HCI provides a uniform command interface to a Controller.

Ford Ex. 1052

CONTENTS

1 Introduction 369
1.1 Lower Layers of the Bluetooth Software Stack 370
2 Overview of Host Controller Transport Layer 372
2.1 Host Controller Transport Layer and AMPS 372
3 Overview of Commands and Events 373
3.1 Generic Events 374
3.2 Device Setup 374
3.3 Controller Flow Control 375
3.4 Controller Information 376
3.5 Controller Configuration 377
3.6 Device Discovery 380
3.7 Connection Setup 382
3.8 Remote Information 387
3.9 Synchronous Connections 388
3.10 Connection State 389
3.11 Piconet Structure 392
3.12 Quality of Service 393
3.13 Physical Links 395
3.14 Host Flow Control 397
3.15 Link Information 398
3.16 Authentication and Encryption 400
3.17 Testing 405
3.18 Alphabetical List of Commands and Events 408
3.19 LE Controller Requirements 415
4 HCI Flow Control 418
4.1 Host to Controller Data Flow Control 418
4.1.1 Packet-based Data Flow Control 418
4.1.2 Data-Block-Based Data Flow Control 420
4.2 Controller to Host Data Flow Control 421
4.3 Disconnection Behavior 421
4.4 Command Flow Control 422
4.5 Command Error Handling 422
5 HCI Data Formats 424
5.1 Introduction 424
5.2 Data and Parameter Formats 424
5.3 Handles 425
5.3.1 Primary Controller Handles 425
5.3.1.1 Broadcast Connection Handles 425

5.3.2 AMP Controller Handles....426
5.4 Exchange of HCI-Specific Information....427
5.4.1 HCI Command Packet....427
5.4.2 HCI ACL Data Packets....428
5.4.3 HCI Synchronous Data Packets....431
5.4.4 HCI Event Packet....434
6 HCI Configuration Parameters....435
6.1 Scan Enable....435
6.2 Inquiry Scan Interval....435
6.3 Inquiry Scan Window....436
6.4 Inquiry Scan Type....436
6.5 Inquiry Mode....436
6.6 Page Timeout....437
6.7 Connection Accept Timeout....437
6.8 Page Scan Interval....438
6.9 Page Scan Window....438
6.10 Page Scan Period Mode (Deprecated)....438
6.11 Page Scan Type....439
6.12 Voice Setting....439
6.13 PIN Type....440
6.14 Link Key....440
6.15 Failed Contact Counter....440
6.16 Authentication Enable....441
6.17 Hold Mode Activity....441
6.18 Link Policy Settings....443
6.19 Flush Timeout....444
6.20 Num Broadcast Retransmissions....444
6.21 Link Supervision Timeout....445
6.22 Synchronous Flow Control Enable....445
6.23 Local Name....446
6.24 Extended Inquiry Response....446
6.25 Erroneous Data Reporting....446
6.26 Class Of Device....447
6.27 Supported Commands....447
6.28 Logical Link Accept Timeout....454
6.29 Location Domain Aware....455
6.30 Location Domain....455
6.31 Location Domain Options....455
6.32 Location Options....456
6.33 Flow Control Mode....456
6.34 LE Supported Host....457
6.35 Simultaneous LE Host....457

7 HCI Commands and Events 458

7.1 Link Control Commands.....458

7.1.1 Inquiry Command 459

7.1.2 Inquiry Cancel Command.....461

7.1.3 Periodic Inquiry Mode Command 462

7.1.4 Exit Periodic Inquiry Mode Command 465

7.1.5 Create Connection Command.....466

7.1.6 Disconnect Command 469

7.1.7 Create Connection Cancel Command.....470

7.1.8 Accept Connection Request Command 472

7.1.9 Reject Connection Request Command 474

7.1.10 Link Key Request Reply Command 475

7.1.11 Link Key Request Negative Reply Command 477

7.1.12 PIN Code Request Reply Command.....478

7.1.13 PIN Code Request Negative Reply Command 480

7.1.14 Change Connection Packet Type Command 481

7.1.15 Authentication Requested Command.....484

7.1.16 Set Connection Encryption Command 486

7.1.17 Change Connection Link Key Command 487

7.1.18 Master Link Key Command.....488

7.1.19 Remote Name Request Command 489

7.1.20 Remote Name Request Cancel Command.....491

7.1.21 Read Remote Supported Features Command.....493

7.1.22 Read Remote Extended Features Command 494

7.1.23 Read Remote Version Information Command.....496

7.1.24 Read Clock Offset Command.....497

7.1.25 Read LMP Handle Command 498

7.1.26 Setup Synchronous Connection Command 500

7.1.27 Accept Synchronous Connection Request Command 504

7.1.28 Reject Synchronous Connection Request Command .508

7.1.29 IO Capability Request Reply Command.....509

7.1.30 User Confirmation Request Reply Command 512

7.1.31 User Confirmation Request Negative Reply Command 513

7.1.32 User Passkey Request Reply Command.....514

7.1.33 User Passkey Request Negative Reply Command.....515

7.1.34 Remote OOB Data Request Reply Command 516

7.1.35 Remote OOB Data Request Negative Reply Command 518
7.1.36 IO Capability Request Negative Reply Command 519
7.1.37 Create Physical Link Command 520
7.1.38 Accept Physical Link Command 523
7.1.39 Disconnect Physical Link Command 525
7.1.40 Create Logical Link Command 527
7.1.41 Accept Logical Link Command 529
7.1.42 Disconnect Logical Link Command 531
7.1.43 Logical Link Cancel Command 532
7.1.44 Flow Spec Modify Command 534
7.2 Link Policy Commands 535
7.2.1 Hold Mode Command 536
7.2.2 Sniff Mode Command 538
7.2.3 Exit Sniff Mode Command 541
7.2.4 Park State Command 542
7.2.5 Exit Park State Command 544
7.2.6 QoS Setup Command 545
7.2.7 Role Discovery Command 548
7.2.8 Switch Role Command 549
7.2.9 Read Link Policy Settings Command 551
7.2.10 Write Link Policy Settings Command 553
7.2.11 Read Default Link Policy Settings Command 555
7.2.12 Write Default Link Policy Settings Command 556
7.2.13 Flow Specification Command 557
7.2.14 Sniff Subrating Command 559
7.3 Controller & Baseband Commands 560
7.3.1 Set Event Mask Command 561
7.3.2 Reset Command 564
7.3.3 Set Event Filter Command 565
7.3.4 Flush Command 571
7.3.5 Read PIN Type Command 573
7.3.6 Write PIN Type Command 574
7.3.7 Create New Unit Key Command 575
7.3.8 Read Stored Link Key Command 576
7.3.9 Write Stored Link Key Command 578
7.3.10 Delete Stored Link Key Command 580
7.3.11 Write Local Name Command 582

7.3.12 Read Local Name Command 583
7.3.13 Read Connection Accept Timeout Command 584
7.3.14 Write Connection Accept Timeout Command 585
7.3.15 Read Page Timeout Command 586
7.3.16 Write Page Timeout Command 587
7.3.17 Read Scan Enable Command 588
7.3.18 Write Scan Enable Command 589
7.3.19 Read Page Scan Activity Command 590
7.3.20 Write Page Scan Activity Command 592
7.3.21 Read Inquiry Scan Activity Command 593
7.3.22 Write Inquiry Scan Activity Command 595
7.3.23 Read Authentication Enable Command 596
7.3.24 Write Authentication Enable Command 597
7.3.25 Read Class of Device Command 598
7.3.26 Write Class of Device Command 599
7.3.27 Read Voice Setting Command 600
7.3.28 Write Voice Setting Command 601
7.3.29 Read Automatic Flush Timeout Command 602
7.3.30 Write Automatic Flush Timeout Command 604
7.3.31 Read Num Broadcast Retransmissions Command 606
7.3.32 Write Num Broadcast Retransmissions Command 607
7.3.33 Read Hold Mode Activity Command 608
7.3.34 Write Hold Mode Activity Command 609
7.3.35 Read Transmit Power Level Command 610
7.3.36 Read Synchronous Flow Control Enable Command 612
7.3.37 Write Synchronous Flow Control Enable Command 613
7.3.38 Set Controller To Host Flow Control Command 614
7.3.39 Host Buffer Size Command 616
7.3.40 Host Number Of Completed Packets Command 618
7.3.41 Read Link Supervision Timeout Command 620
7.3.42 Write Link Supervision Timeout Command 622
7.3.43 Read Number Of Supported IAC Command 624
7.3.44 Read Current IAC LAP Command 625
7.3.45 Write Current IAC LAP Command 626
7.3.46 Set AFH Host Channel Classification Command 628
7.3.47 Read Inquiry Scan Type Command 629
7.3.48 Write Inquiry Scan Type Command 630
7.3.49 Read Inquiry Mode Command 631

7.3.50 Write Inquiry Mode Command 632
7.3.51 Read Page Scan Type Command 633
7.3.52 Write Page Scan Type Command 634
7.3.53 Read AFH Channel Assessment Mode Command 635
7.3.54 Write AFH Channel Assessment Mode Command 636
7.3.55 Read Extended Inquiry Response Command 638
7.3.56 Write Extended Inquiry Response Command 639
7.3.57 Refresh Encryption Key Command 640
7.3.58 Read Simple Pairing Mode Command 641
7.3.59 Write Simple Pairing Mode Command 642
7.3.60 Read Local OOB Data Command 644
7.3.61 Read Inquiry Response Transmit Power Level Command 646
7.3.62 Write Inquiry Transmit Power Level Command 647
7.3.63 Send Keypress Notification Command 648
7.3.64 Read Default Erroneous Data Reporting 650
7.3.65 Write Default Erroneous Data Reporting 651
7.3.66 Enhanced Flush Command 652
7.3.67 Read Logical Link Accept Timeout Command 654
7.3.68 Write Logical Link Accept Timeout Command 655
7.3.69 Set Event Mask Page 2 Command 656
7.3.70 Read Location Data Command 658
7.3.71 Write Location Data Command 659
7.3.72 Read Flow Control Mode Command 660
7.3.73 Write Flow Control Mode Command 661
7.3.74 Read Enhanced Transmit Power Level Command 662
7.3.75 Read Best Effort Flush Timeout Command 665
7.3.76 Write Best Effort Flush Timeout Command 666
7.3.77 Short Range Mode Command 667
7.3.78 Read LE Host Supported Command 668
7.3.79 Write LE Host Supported Command 669
7.4 Informational Parameters 670
7.4.1 Read Local Version Information Command 670
7.4.2 Read Local Supported Commands Command 672
7.4.3 Read Local Supported Features Command 673
7.4.4 Read Local Extended Features Command 674
7.4.5 Read Buffer Size Command 676
7.4.6 Read BD_ADDR Command 678

7.4.7 Read Data Block Size Command 679
7.5 Status Parameters 680
7.5.1 Read Failed Contact Counter Command 681
7.5.2 Reset Failed Contact Counter Command 683
7.5.3 Read Link Quality Command 685
7.5.4 Read RSSI Command 687
7.5.5 Read AFH Channel Map Command 689
7.5.6 Read Clock Command 691
7.5.7 Read Encryption Key Size Command 693
7.5.8 Read Local AMP Info Command 695
7.5.9 Read Local AMP ASSOC Command 700
7.5.10 Write Remote AMP ASSOC Command 703
7.6 Testing Commands 704
7.6.1 Read Loopback Mode Command 705
7.6.2 Write Loopback Mode Command 706
7.6.3 Enable Device Under Test Mode Command 709
7.6.4 Write Simple Pairing Debug Mode Command 710
7.6.5 Enable AMP Receiver Reports Command 712
7.6.6 AMP Test End Command 713
7.6.7 AMP Test Command 714
7.7 Events 715
7.7.1 Inquiry Complete Event 715
7.7.2 Inquiry Result Event 716
7.7.3 Connection Complete Event 718
7.7.4 Connection Request Event 719
7.7.5 Disconnection Complete Event 721
7.7.6 Authentication Complete Event 722
7.7.7 Remote Name Request Complete Event 723
7.7.8 Encryption Change Event 724
7.7.9 Change Connection Link Key Complete Event 725
7.7.10 Master Link Key Complete Event 726
7.7.11 Read Remote Supported Features Complete Event 727
7.7.12 Read Remote Version Information Complete Event 728
7.7.13 QoS Setup Complete Event 730
7.7.14 Command Complete Event 732
7.7.15 Command Status Event 733
7.7.16 Hardware Error Event 734
7.7.17 Flush Occurred Event 735

7.7.18 Role Change Event 736
7.7.19 Number Of Completed Packets Event 737
7.7.20 Mode Change Event 739
7.7.21 Return Link Keys Event 741
7.7.22 PIN Code Request Event 742
7.7.23 Link Key Request Event 743
7.7.24 Link Key Notification Event 744
7.7.25 Loopback Command Event 746
7.7.26 Data Buffer Overflow Event 747
7.7.27 Max Slots Change Event 748
7.7.28 Read Clock Offset Complete Event 749
7.7.29 Connection Packet Type Changed Event 750
7.7.30 QoS Violation Event 752
7.7.31 Page Scan Repetition Mode Change Event 753
7.7.32 Flow Specification Complete Event 754
7.7.33 Inquiry Result with RSSI Event 756
7.7.34 Read Remote Extended Features Complete Event 758
7.7.35 Synchronous Connection Complete Event 760
7.7.36 Synchronous Connection Changed Event 762
7.7.37 Sniff Subrating Event 764
7.7.38 Extended Inquiry Result Event 766
7.7.39 Encryption Key Refresh Complete Event 768
7.7.40 IO Capability Request Event 769
7.7.41 IO Capability Response Event 770
7.7.42 User Confirmation Request Event 772
7.7.43 User Passkey Request Event 773
7.7.44 Remote OOB Data Request Event 774
7.7.45 Simple Pairing Complete Event 775
7.7.46 Link Supervision Timeout Changed Event 776
7.7.47 Enhanced Flush Complete Event 777
7.7.48 User Passkey Notification Event 778
7.7.49 Keypress Notification Event 779
7.7.50 Remote Host Supported Features Notification Event 780
7.7.51 Physical Link Complete Event 781
7.7.52 Channel Selected Event 782
7.7.53 Disconnection Physical Link Complete Event 783
7.7.54 Physical Link Loss Early Warning Event 784
7.7.55 Physical Link Recovery Event 785

7.7.56 Logical Link Complete Event....786
7.7.57 Disconnection Logical Link Complete Event....787
7.7.58 Flow Spec Modify Complete Event....788
7.7.59 Number Of Completed Data Blocks Event....789
7.7.60 Short Range Mode Change Complete Event....791
7.7.61 AMP Status Change Event....792
7.7.62 AMP Start Test Event....794
7.7.63 AMP Test End Event....795
7.7.64 AMP Receiver Report Event....796
7.7.65 LE Meta Event....798
7.7.65.1 LE Connection Complete Event....798
7.7.65.2 LE Advertising Report Event....801
7.7.65.3 LE Connection Update Complete Event....802
7.7.65.4 LE Read Remote Used Features Complete Event....804
7.7.65.5 LE Long Term Key Request Event....805
7.8 LE Controller Commands....806
7.8.1 LE Set Event Mask Command....806
7.8.2 LE Read Buffer Size Command....808
7.8.3 LE Read Local Supported Features Command....810
7.8.4 LE Set Random Address Command....811
7.8.5 LE Set Advertising Parameters Command....812
7.8.6 LE Read Advertising Channel Tx Power Command....815
7.8.7 LE Set Advertising Data Command....816
7.8.8 LE Set Scan Response Data Command....817
7.8.9 LE Set Advertise Enable Command....818
7.8.10 LE Set Scan Parameters Command....820
7.8.11 LE Set Scan Enable Command....823
7.8.12 LE Create Connection Command....825
7.8.13 LE Create Connection Cancel Command....829
7.8.14 LE Read White List Size Command....830
7.8.15 LE Clear White List Command....831
7.8.16 LE Add Device To White List Command....832
7.8.17 LE Remove Device From White List Command....833
7.8.18 LE Connection Update Command....835
7.8.19 LE Set Host Channel Classification Command....838
7.8.20 LE Read Channel Map Command....839
7.8.21 LE Read Remote Used Features Command....841
7.8.22 LE Encrypt Command....842

7.8.23 LE Rand Command 844
7.8.24 LE Start Encryption Command 845
7.8.25 LE Long Term Key Request Reply Command 847
7.8.26 LE Long Term Key Request Negative Reply Command 849
7.8.27 LE Read Supported States Command 850
7.8.28 LE Receiver Test Command 853
7.8.29 LE Transmitter Test Command 854
7.8.30 LE Test End Command 856

8 List of Figures 857

9 List of Tables 858

10 Appendix A: Deprecated Commands, Events and Configuration Parameters 859

10.1 Read Page Scan Mode Command 860
10.2 Write Page Scan Mode Command 861
10.3 Read Page Scan Period Mode Command 862
10.4 Write Page Scan Period Mode Command 863
10.5 Add SCO Connection Command 864
10.6 Page Scan Mode Change Event 866
10.7 Read Country Code Command 867
10.8 Read Encryption Mode Command 868
10.9 Write Encryption Mode Command 868
10.10 Deprecated Parameters 869
10.10.1 Encryption Mode 869
10.10.2 Page Scan Mode 871

1 INTRODUCTION

This Part of the Bluetooth Core Specification describes the functional specifications for the Host Controller Interface (HCI). The HCI provides a uniform interface method of accessing a Bluetooth Controller's capabilities. The next three sections provide a brief overview of the lower layers of the Bluetooth software stack and of the Bluetooth hardware. Section 2 provides an overview of the Lower HCI Device Driver Interface on the host device. Section 4 describes the flow control used between the Host and the Controller. Section 7 describes each of the HCI Commands in detail, identifies parameters for each of the commands, and lists events associated with each command.

This specification is applicable to the following types of Controllers:

- BR/EDR Controller
- BR/EDR/LE Controller
- LE Controller
- AMP Controller

A Host may drive a Primary Controller and zero or more AMP Controllers. A Primary Controller shall support one of the following sets of functionality:

- BR/EDR only
- LE only
- BR/EDR/LE

In the following sections, the term "BR/EDR Controller" is used to describe the BR/EDR functionality of a Primary Controller which may be either a BR/EDR only Controller or a BR/EDR/LE Controller. Similarly, the term "LE Controller" is used to describe the LE functionality of a Primary Controller which may be either an LE only Controller or a BR/EDR/LE Controller.

A BR/EDR Controller supports BR/EDR functionality only. A BR/EDR/LE Controller supports BR/EDR functionality and LE functionality. An LE Controller supports LE functionality only. An AMP Controller supports an AMP PAL. A Primary Controller is either a BR/EDR Controller, a BR/EDR/LE Controller, or an LE Controller.

1.1 LOWER LAYERS OF THE BLUETOOTH SOFTWARE STACK

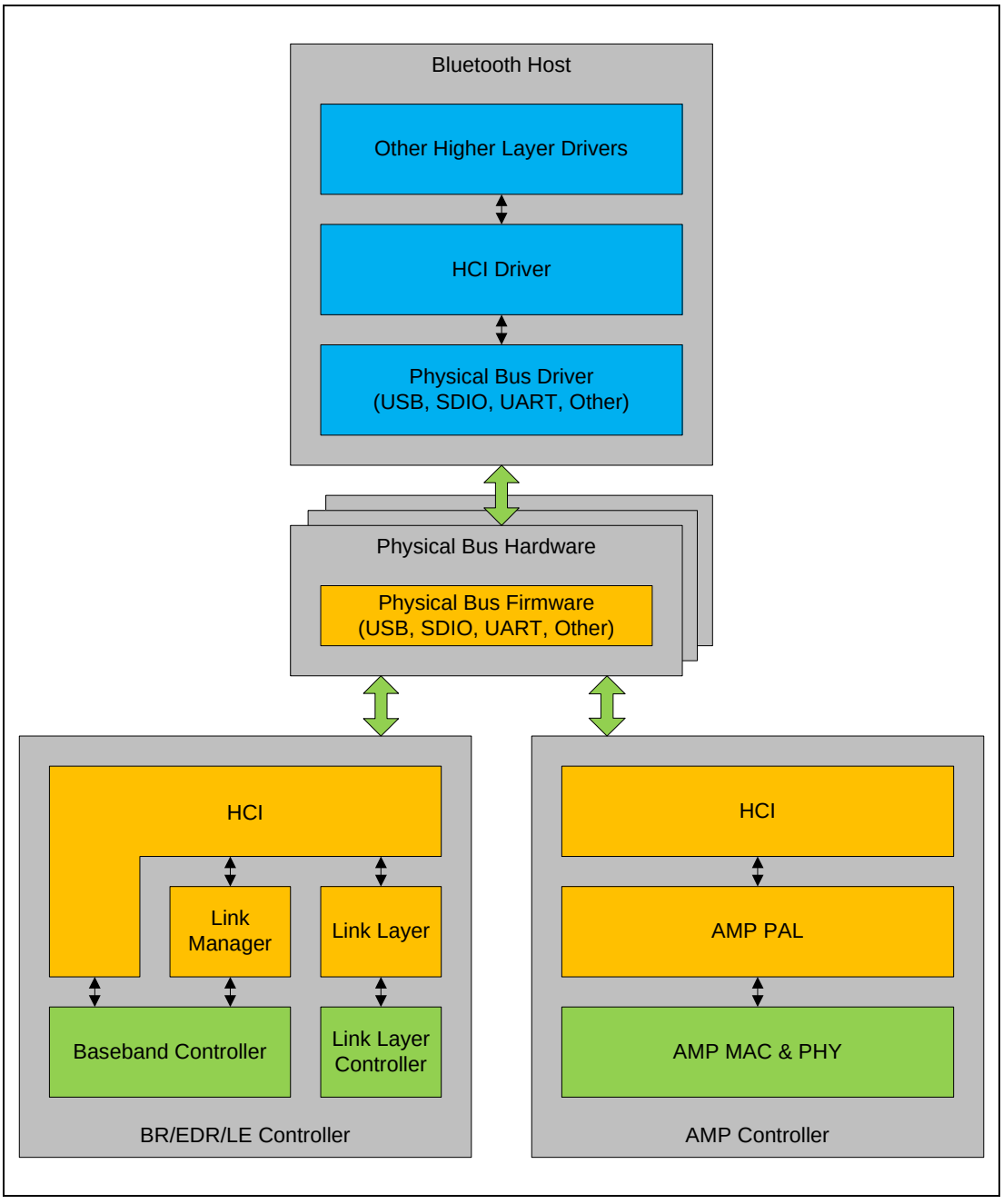


Figure 1.1: Overview of the Lower Software Layers

Figure 1.1 provides an overview of the lower software layers.

Several layers may exist between the HCI driver on the Host system and the HCI layer in the Controller(s). These intermediate layers, the Host Controller Transport Layer, provide the ability to transfer data without intimate knowledge of the data.

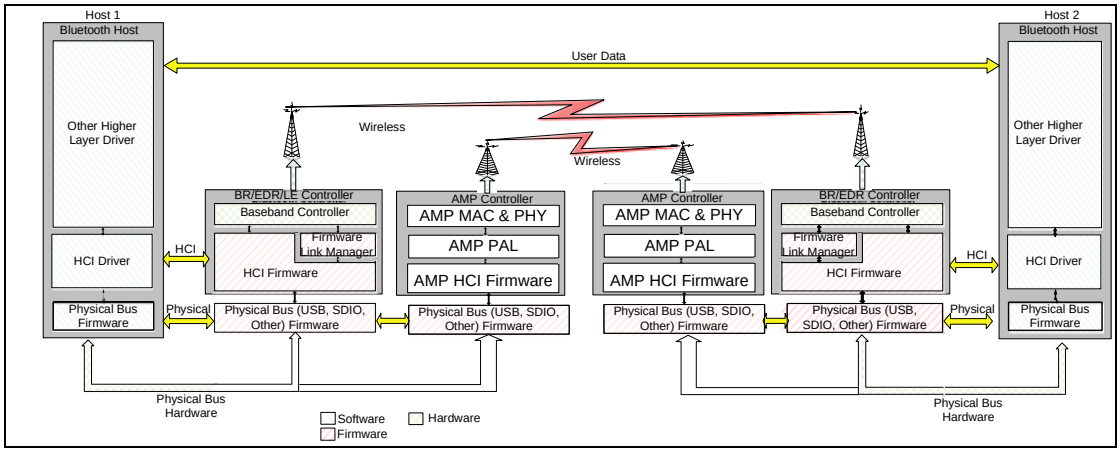


Figure 1.2: End to End Overview of Lower Software Layers to Transfer Data

Figure 1.2 illustrates the path of a data transfer from one device to another. The HCI driver on the Host exchanges data and commands with the HCI firmware on the Bluetooth hardware. The Host Control Transport Layer (i.e. physical bus) driver provides both HCI layers with the ability to exchange information with each other.

The Host will receive asynchronous notifications of HCI events independent of which Host Controller Transport Layer is used. HCI events are used for notifying the Host when something occurs. When the Host discovers that an event has occurred it will then parse the received event packet to determine which event occurred.

A BR/EDR/LE Controller uses one shared command buffer and flow control for BR/EDR and LE. Data buffers can be either shared between BR/EDR and LE or there may be separate data buffers for BR/EDR and LE. The configuration of a Controller is determined through HCI.

LE Controllers use a reduced set of HCI commands and events. Some commands and events are re-used for multiple Controller types.

2 OVERVIEW OF HOST CONTROLLER TRANSPORT LAYER

The Host driver stack has a transport layer between the Host Controller Interface driver and the Host.

The main goal of this transport layer is transparency. The Host Controller driver (which interfaces to the Controller) should be independent of the underlying transport technology. In addition, the transport should not require any understanding of the data that the Host Controller driver passes to the Controller. This allows the logical interface (HCI) or the Controller to be upgraded without affecting the transport layer.

The specified Host Controller Transport Layers are described in Volume 4, Host Controller Interface [Transport Layer].

2.1 HOST CONTROLLER TRANSPORT LAYER AND AMPS

The logical Host Controller Interface does not consider multiplexing/routing over the Host Controller Transport Layer(s). The Host designer must consider this in deciding which of the multiple controller configurations it will support. Each logical AMP HCI instantiation is particular to a single AMP Controller. The Host shall direct each HCI Command to a specific AMP, and be aware of the source of each HCI Event, and manage flow control separately for each HCI. The Host's Controller_ID concept used for AMP enumeration may assist with this.

The logical HCI definition does not consider the initial discovery of what AMPs are present in the system.

The upper layers discover the AMP capabilities (see Section 3.3). As seen by the AMP Manager, each local AMP has a Controller_ID that is used to route the data and commands to the correct AMP Controller. This identifier is not used in AMP HCI.

The separate logical HCIs within a system may be on the same physical HCI transport, on separate physical HCI transports, or any combination. This may require a layer of multiplexing, which is not defined within this AMP HCI logical specification.

The logical HCI is not affected by this topology. The logical HCI definition describes commands and events as seen by a single AMP.

3 OVERVIEW OF COMMANDS AND EVENTS

The commands and events are sent between the Host and the Controller(s). These are grouped into logical groups by function.

Generic Events	The generic events can occur due to multiple commands, or events that can occur at any time.
Device Setup	The device setup commands are used to place the Controller into a known state.
Controller Flow Control	The Controller flow control commands and events are used to control data flow from the Host to the Controller.
Controller Information	The Controller information commands allow the Host to discover local information about the device.
Controller Configuration	The Controller configuration commands and events allow the global configuration parameters to be configured.
Device Discovery	The device discovery commands and events allow a device to discover other devices in the surrounding area.
Connection Setup	The connection setup commands and events allow a device to make a connection to another device.
Remote Information	The remote information commands and events allow information about a remote device's configuration to be discovered.
Synchronous Connections	The synchronous connection commands and events allow synchronous connections to be created
Connection State	The connection state commands and events allow the configuration of a link, especially for low power operation.
Piconet Structure	The piconet structure commands and events allow the discovery and reconfiguration of piconet.
Quality of Service	The quality of service commands and events allow quality of service parameters to be specified.
Physical Links	The physical link commands and events allow the configuration of a physical link.
Host Flow Control	The Host flow control commands and events allow flow control to be used towards the Host.
Link Information	The link information commands and events allow information about a link to be read.
Authentication and Encryption	The authentication and encryption commands and events allow authentication of a remote device and then encryption of the link.
Testing	The testing commands and events allow a device to be placed into test mode.

Table 3.1: Overview of commands and events

The version information in this section denotes the version number of the specification in which this command or event is first specified. The Supported Con-

trollers column denotes the Controller types that support the command: All, BR/EDR, LE, or AMP, or a comma-separated list of Controller types (e.g. “BR/EDR, LE”).

3.1 GENERIC EVENTS

The generic events occur due to multiple commands, or events that can occur at any time.

Name	Vers.	Summary description	Supported Controllers
Command Complete Event	1.1	The Command Complete event is used by the Controller to pass the return status of a command and the other event parameters for each HCI Command.	All
Command Status Event	1.1	The Command Status event is used to indicate that the command described by the Command_Opcode parameter has been received and the Controller is currently performing the task for this command.	All
Hardware Error Event	1.1	The Hardware Error event is used to indicate some type of hardware failure for the Controller.	All

Table 3.2: Generic events

3.2 DEVICE SETUP

The device setup group of commands are used to place the Controller into a known state.

Name	Vers.	Summary description	Supported Controllers
Reset Command	1.1	For a BR/EDR Controller, the Reset command resets HCI, the Link Manager, and the Bluetooth radio. For an AMP Controller, the Reset command resets HCI and the AMP PAL. For an LE Controller, the Reset command resets HCI, the Link Layer, and LE PHY.	All

Table 3.3: Device setup

3.3 CONTROLLER FLOW CONTROL

The controller flow control group of commands and events are used to control data flow from the Host to the Controller.

Name	Vers.	Summary description	Supported Controllers
Read Buffer Size Command	1.1	The Read_Buffer_Size command returns the size of the HCI buffers. These buffers are used by the Controller to buffer data that is to be transmitted.	All
Number Of Completed Packets Event	1.1	The Number Of Completed Packets event is used by the Controller to indicate to the Host how many HCI Data Packets have been completed for each Connection_Handle since the previous Number Of Completed Packets event was sent.	All
Read Data Block Size Command	3.0 + HS	The Read Data Block Size command returns the maximum size of the HCI buffers. These buffers are used by the Controller to buffer data that is to be transmitted.	BR/EDR, AMP
Number Of Completed Data Blocks Event	3.0 + HS	The Number Of Completed Data Blocks event is used by the Controller to indicate to the Host how many HCI ACL Data Packets have been completed and how many data block buffers have been freed for each Handle since the previous Number of Completed Data Blocks event was sent.	BR/EDR, AMP
Read Flow Control Mode Command	3.0 + HS	The Read Flow Control Mode command returns the value of the Flow_Control_Mode configuration parameter supported by this Controller.	BR/EDR, AMP
Write Flow Control Mode Command	3.0 + HS	The Write Flow Control Mode command sets the value of the Flow_Control_Mode configuration parameter for this Controller.	BR/EDR, AMP
LE Read Buffer Size Command	4.0	The LE Read Buffer Size command returns the size of the HCI buffers. These buffers are used by the LE Controller to buffer data that is to be transmitted.	LE

Table 3.4: Controller flow control

3.4 CONTROLLER INFORMATION

The controller information group of commands allows the Host to discover local information about the device.

Name	Vers.	Summary description	Supported Controllers
Read Local Version Information Command	1.1	The Read Local Version Information command will read the version information for the local Controller.	All
Read Local Supported Commands Command	1.2	The Read Local Supported Commands command requests a list of the supported HCI commands for the local device.	All
Read Local Supported Features Command	1.1	The Read Local Supported Features command requests a list of the supported features for the local device.	All
Read Local Extended Features Command	1.2	The Read Local Extended Features command requests a list of the supported extended features for the local device	BR/EDR
Read BD_ADDR Command	1.1	The Read BD_ADDR command will read the value for the BD_ADDR parameter.	BR/EDR, LE
LE Read Local Supported Features Command	4.0	The LE Read Local Supported Features command will read the version information for the local LE Controller.	LE
LE Read Supported States Command	4.0	The LE Read Supported States Command will read the current supported state and role combinations for the local LE Controllers.	LE

Table 3.5: Controller information

3.5 CONTROLLER CONFIGURATION

The controller configuration group of commands and events allows the global configuration parameters to be configured.

Name	Vers.	Summary description	Supported Controllers
Read Local Name Command	1.1	The Read Local Name command provides the ability to read the stored user-friendly name for the BR/EDR Controller.	BR/EDR
Write Local Name Command	1.1	The Write Local Name command provides the ability to modify the user-friendly name for the BR/EDR Controller.	BR/EDR
Read Class of Device Command	1.1	The Read Class of Device command will read the value for the Class of Device configuration parameter, which is used to indicate its capabilities to other devices.	BR/EDR
Write Class of Device Command	1.1	The Write Class of Device command will write the value for the Class_of_Device configuration parameter, which is used to indicate its capabilities to other devices.	BR/EDR
Read Number Of Supported IAC Command	1.1	The Read Number of Supported IAC command will read the value for the number of Inquiry Access Codes (IAC) that the local BR/EDR Controller can simultaneously listen for during an Inquiry Scan.	BR/EDR
Read Current IAC LAP Command	1.1	The Read Current IAC LAP command will read the LAP(s) used to create the Inquiry Access Codes (IAC) that the local BR/EDR Controller is simultaneously scanning for during Inquiry Scans.	BR/EDR
Write Current IAC LAP Command	1.1	The Write Current IAC LAP command will write the LAP(s) used to create the Inquiry Access Codes (IAC) that the local BR/EDR Controller is simultaneously scanning for during Inquiry Scans.	BR/EDR

Table 3.6: Controller configuration

Name	Vers.	Summary description	Supported Controllers
Read Scan Enable Command	1.1	The Read Scan Enable command will read the value for the Scan Enable configuration parameter, which controls whether or not the BR/EDR Controller will periodically scan for page attempts and/or inquiry requests from other BR/EDR Controllers.	BR/EDR
Write Scan Enable Command	1.1	The Write Scan Enable command will write the value for the Scan Enable configuration parameter, which controls whether or not the BR/EDR Controller will periodically scan for page attempts and/or inquiry requests from other BR/EDR Controllers.	BR/EDR
Write Extended Inquiry Response Command	2.1 + EDR	The Write Extended Inquiry Response command will write the data that the BR/EDR Controller sends in the extended inquiry response packet during inquiry response.	BR/EDR
Read Extended Inquiry Response Command	2.1 + EDR	The Read Extended Inquiry Response command will read the data that the BR/EDR Controller sends in the extended inquiry response packet during inquiry response.	BR/EDR
Write Default Erroneous Data Reporting Command	2.1 + EDR	The Write Default Erroneous Data Reporting command will write the value for the Erroneous Data Reporting configuration parameter, which controls whether the Bluetooth controller will provide data for every (e)SCO interval, with the Packet_Status_Flag in HCI Synchronous Data Packets set according to HCI Synchronous Data Packets.	BR/EDR

Table 3.6: Controller configuration

Name	Vers.	Summary description	Supported Controllers
Read Default Erroneous Data Reporting Command	2.1 + EDR	The Read Default Erroneous Data Reporting command will read the value for the Erroneous Data Reporting configuration parameter, which controls whether the BR/EDR Controller will provide data for every (e)SCO interval, with the Packet_Status_Flag in HCI Synchronous Data Packets set according to HCI Synchronous Data Packets.	BR/EDR
Read Location Data Command	3.0 + HS	The Read Location Data command provides the ability to read the Location Data parameters from any stored knowledge of environment or regulations or currently in use in the AMP Controller	AMP
Write Location Data Command	3.0 + HS	The Write Location Data command is used to tell the Controller any knowledge of environment or regulations currently in force, which may affect the operation of the Controller.	AMP
LE Set Advertise Enable Command	4.0	The LE Set Advertise Enable Command will enable or disable advertising.	LE
LE Set Advertising Data Command	4.0	The LE Set Advertising Data Command will set the data transmitted when advertising.	LE
LE Set Advertising Parameters Command	4.0	The LE Set Advertising Parameters Ccommand will set the parameters used for advertising.	LE
LE Set Random Address Command	4.0	The LE Set Random Address Command will set the Random Device Address that may be used in an advertising packet.	LE
LE Set Scan Response Data Command	4.0	The LE Set Scan Response Data Command will set the data transmitted in a scan response.	LE
Read LE Host Support Command	4.0	This command reads the LE Supported Host and Simultaneous LE Host settings from the BR/EDR Controller.	BR/EDR

Table 3.6: Controller configuration

Name	Vers.	Summary description	Supported Controllers
Write LE Host Support Command	4.0	This command writes the LE Supported Host and Simultaneous LE Host settings to the BR/EDR Controller.	BR/EDR

Table 3.6: Controller configuration

3.6 DEVICE DISCOVERY

The device discovery group of commands and events allow a device to discovery other devices in the surrounding area.

Name	Vers.	Summary description	Supported Controllers
Inquiry Command	1.1	The Inquiry command will cause the BR/EDR Controller to enter Inquiry Mode. Inquiry Mode is used to discovery other nearby BR/EDR Controllers.	BR/EDR
Inquiry Result Event	1.1	The Inquiry Result event indicates that a BR/EDR Controller or multiple BR/EDR Controllers have responded so far during the current Inquiry process.	BR/EDR
Inquiry Result with RSSI Event	1.2	The Inquiry Result with RSSI event indicates that a BR/EDR Controller or multiple BR/EDR Controllers have responded so far during the current Inquiry process.	BR/EDR
Extended Inquiry Result Event	2.1 + EDR	The Extended Inquiry Result event indicates that a BR/EDR Controller has responded with an extended inquiry response during the current Inquiry process.	BR/EDR
Inquiry Cancel Command	1.1	The Inquiry Cancel command will cause the BR/EDR Controller to stop the current Inquiry if the BR/EDR Controller is in Inquiry Mode.	BR/EDR
Inquiry Complete Event	1.1	The Inquiry Complete event indicates that the Inquiry is finished.	BR/EDR
Periodic Inquiry Mode Command	1.1	The Periodic Inquiry Mode command is used to configure the BR/EDR Controller to perform an automatic Inquiry based on a specified period range.	BR/EDR

Table 3.7: Device discovery

Name	Vers.	Summary description	Supported Controllers
Exit Periodic Inquiry Mode Command	1.1	The Exit Periodic Inquiry Mode command is used to end the Periodic Inquiry mode when the local device is in Periodic Inquiry Mode.	BR/EDR
Read Inquiry Scan Activity Command	1.1	The Read Inquiry Scan Activity command will read the value for Inquiry Scan Interval and Inquiry Scan Window configuration parameters. Inquiry Scan Interval defines the amount of time between consecutive inquiry scans. Inquiry Scan Window defines the amount of time for the duration of the inquiry scan.	BR/EDR
Write Inquiry Scan Activity Command	1.1	The Write Inquiry Scan Activity command will write the value for Inquiry Scan Interval and Inquiry Scan Window configuration parameters. Inquiry Scan Interval defines the amount of time between consecutive inquiry scans. Inquiry Scan Window defines the amount of time for the duration of the inquiry scan.	BR/EDR
Read Inquiry Scan Type Command	1.2	The Read Inquiry Scan Type command is used to read the Inquiry Scan Type configuration parameter of the local BR/EDR Controller. The Inquiry Scan Type configuration parameter can set the inquiry scan to either normal or interlaced scan.	BR/EDR
Write Inquiry Scan Type Command	1.2	The Write Inquiry Scan Type command is used to write the Inquiry Scan Type configuration parameter of the local BR/EDR Controller. The Inquiry Scan Type configuration parameter can set the inquiry scan to either normal or interlaced scan.	BR/EDR
Read Inquiry Mode Command	1.2	The Read Inquiry Mode command is used to read the Inquiry Mode configuration parameter of the local BR/EDR Controller.	BR/EDR
Write Inquiry Mode Command	1.2	The Write Inquiry Mode command is used to write the Inquiry Mode configuration parameter of the local BR/EDR Controller.	BR/EDR

Table 3.7: Device discovery

Name	Vers.	Summary description	Supported Controllers
Read Inquiry Response Transmit Power Level Command	2.1 + EDR	This command will read the inquiry response Transmit Power level used to transmit the FHS and EIR data packets. This can be used directly in the Tx Power Level EIR data type.	BR/EDR
Write Inquiry Transmit Power Level Command	2.1 + EDR	This command is used to write the transmit power level used to transmit the inquiry (ID) data packets.	BR/EDR
LE Advertising Report Event	4.0	The LE Advertising Report event indicates that an advertising or scan response packet has been received.	LE
LE Set Scan Enable Command	4.0	The LE Set Scan Enable Command will enable or disable scanning.	LE
LE Set Scan Parameters Command	4.0	The LE Set Scan Parameters Command will set the parameters used for scanning.	LE

Table 3.7: Device discovery

3.7 CONNECTION SETUP

The connection setup group of commands and events are used to allow a device to make a connection to another device.

Name	Vers.	Summary description	Supported Controllers
Create Connection Command	1.1	The Create Connection command will cause the BR/EDR Link Manager to create an ACL connection to the BR/EDR Controller with the BD_ADDR specified by the command parameters.	BR/EDR
Connection Request Event	1.1	The Connection Request event is used to indicate that a new incoming BR/EDR connection is trying to be established.	BR/EDR
Accept Connection Request Command	1.1	The Accept Connection Request command is used to accept a new incoming BR/EDR connection request.	BR/EDR
Reject Connection Request Command	1.1	The Reject Connection Request command is used to decline a new incoming BR/EDR connection request.	BR/EDR

Table 3.8: Connection setup

Name	Vers.	Summary description	Supported Controllers
Create Connection Cancel Command	1.2	The Create Connection Cancel command is used to cancel an ongoing Create Connection.	BR/EDR
Connection Complete Event	1.1	The Connection Complete event indicates to both of the Hosts forming the connection that a new BR/EDR connection has been established.	BR/EDR
Disconnect Command	1.1	The Disconnect command is used to terminate an existing BR/EDR or LE connection.	BR/EDR, LE
Disconnection Complete Event	1.1	The Disconnection Complete event occurs when a connection has been terminated.	BR/EDR, LE
Read Page Timeout Command	1.1	The Read Page Timeout command will read the value for the Page Reply Timeout configuration parameter, which determines the time the BR/EDR Controller will wait for the remote device to respond to a connection request before the local device returns a connection failure.	BR/EDR
Write Page Timeout Command	1.1	The Write Page Timeout command will write the value for the Page Reply Timeout configuration parameter, which allows the BR/EDR Controller to define the amount of time a connection request will wait for the remote device to respond before the local device returns a connection failure.	BR/EDR
Read Page Scan Activity Command	1.1	The Read Page Scan Activity command will read the values for the Page Scan Interval and Page Scan Window configuration parameters. Page Scan Interval defines the amount of time between consecutive page scans. Page Scan Window defines the duration of the page scan.	BR/EDR

Table 3.8: Connection setup

Name	Vers.	Summary description	Supported Controllers
Write Page Scan Activity Command	1.1	The Write Page Scan Activity command will write the value for Page Scan Interval and Page Scan Window configuration parameters. Page Scan Interval defines the amount of time between consecutive page scans. Page Scan Window defines the duration of the page scan.	BR/EDR
Page Scan Repetition Mode Change Event	1.1	The Page Scan Repetition Mode Change event indicates that the connected remote BR/EDR Controller with the specified Connection_Handle has successfully changed the Page_Scan_Repetition_Mode (SR)."	BR/EDR
Read Page Scan Type Command	1.2	The Read Page Scan Type command is used to read the page scan type of the local BR/EDR Controller. The Page Scan Type configuration parameter can set the page scan to either normal or interlaced scan.	BR/EDR
Write Page Scan Type Command	1.2	The Write Page Scan Type command is used to write the page scan type of the local BR/EDR Controller. The Page Scan Type configuration parameter can set the page scan to either normal or interlaced scan.	BR/EDR
Read Connection Accept Timeout Command	1.1	The Read Connection Accept Timeout command will read the value for the Connection Accept Timeout configuration parameter, which allows the Controller to automatically deny a connection request after a specified period has occurred, and to refuse a new connection.	BR/EDR, AMP
Read Hold Mode Activity	1.1	The Read Hold Mode Activity command is used to read which activities should be suspended when the BR/EDR Controller is in Hold Mode.	BR/EDR

Table 3.8: Connection setup

Name	Vers.	Summary description	Supported Controllers
Write Hold Mode Activity	1.1	The Write Hold Mode Activity command is used to write which activities should be suspended when the BR/EDR Controller is in Hold Mode.	BR/EDR
Write Connection Accept Timeout Command	1.1	The Write Connection Accept Timeout command will write the value for the Connection Accept Timeout configuration parameter, which allows the Controller to automatically deny a connection request after a specified period has occurred, and to refuse a new connection.	BR/EDR, AMP
Accept Logical Link Command	3.0 + HS	The Accept Logical Command will cause the AMP Controller to create a logical link to the remote AMP Controller identified by the physical link handle.	AMP
Accept Physical Link Command	3.0 + HS	The Accept Physical Link command will cause the AMP Controller to accept a Physical Link request from the initiating BR/EDR Controller 's AMP Controller specified by the command parameters.	AMP
Create Logical Link Command	3.0 + HS	The Create Logical Command will cause the AMP Controller to create a logical link to the remote AMP Controller identified by the physical link handle.	AMP
Create Physical Link Command	3.0 + HS	The Create Physical Link command will cause the AMP Controller to initiate a Physical Link creation to the remote BR/EDR Controller's AMP Controller specified by the command parameters.	AMP
Disconnect Logical Link Command	3.0 + HS	The Disconnect Logical Link command is used to terminate the designated logical link.	AMP
Disconnection Logical Link Complete Event	3.0 + HS	The Disconnection Logical Link Complete event occurs when an AMP Logical Link has been terminated.	AMP

Table 3.8: Connection setup

Name	Vers.	Summary description	Supported Controllers
Disconnect Physical Link Command	3.0 + HS	The Disconnect Physical Link command is used to terminate the designated physical link (during or after creation).	AMP
Disconnection Physical Link Complete Event	3.0 + HS	The Disconnection Physical Link Complete event occurs when an AMP Physical Link has been terminated.	AMP
Logical Link Cancel Command	3.0 + HS	The Logical Link Cancel command is used to request cancellation of the ongoing logical link creation process identified by the Physical_Link_Handle and the Tx_Flow_Spec_ID	AMP
Logical Link Complete Event	3.0 + HS	The Logical Link Complete event indicates to both of the Hosts forming the connection that a new AMP Logical Link has been established.	AMP
Physical Link Complete Event	3.0 + HS	The Physical Link Complete event indicates to both of the Hosts forming the connection that a new AMP Logical Link has been established	AMP
Physical Link Loss Early Warning Event	3.0 + HS	The Physical Link Loss Early Warning event occurs when a physical link has indication that the link may be disrupted.	AMP
Physical Link Recovery Event	3.0 + HS	The Physical Link Recovery event indicates that whatever interruption caused an earlier Physical Link Loss Early Warning event has now been cleared.	AMP
Read Logical Link Accept Timeout Command	3.0 + HS	The Read Logical Link Accept Timeout command will read the value for the Logical Link Accept Timeout configuration parameter, which allows the AMP Controller to automatically deny a Logical Link request after a specified period has occurred, and to refuse a new connection.	AMP

Table 3.8: Connection setup

Name	Vers.	Summary description	Supported Controllers
Write Logical Link Accept Timeout Command	3.0 + HS	The Write Logical Link Accept Timeout command will write the value for the Logical Link Accept Timeout configuration parameter, which allows the AMP Controller to automatically deny a connection request after a specified period has occurred, and to refuse a new connection.	AMP
Channel Selected Event	3.0 + HS	The Channel Selected event indicates to the Host originating the physical link that the link information data is available to be read using the Read Local AMP Assoc command.	AMP
LE Connection Complete Event	4.0	The LE Connection Complete event indicates to the Host that a new connection has been created.	LE
LE Create Connection Cancel Command	4.0	The LE Create Connection Cancel Command is used to cancel an ongoing LE Create Connection Command.	LE
LE Create Connection Command	4.0	The LE Create Connection Command is used to create a new connection.	LE

Table 3.8: Connection setup

3.8 REMOTE INFORMATION

The remote information group of commands and events allows information about a remote devices configuration to be discovered.

Name	Vers.	Summary description	Supported Controllers
Remote Name Request Command	1.1	The Remote Name Request command is used to obtain the user-friendly name of another BR/EDR Controller.	BR/EDR
Remote Name Request Cancel Command	1.2	The Remote Name Request Cancel command is used to cancel an ongoing Remote Name Request.	BR/EDR
Remote Name Request Complete Event	1.1	The Remote Name Request complete event is used to indicate a remote name request has been completed.	BR/EDR

Table 3.9: Remote information

Name	Vers.	Summary description	Supported Controllers
Read Remote Supported Features Command	1.1	The Read Remote Supported Features command requests a list of the supported features of a remote device.	BR/EDR
Read Remote Supported Features Complete Event	1.1	The Read Remote Supported Features Complete event is used to indicate the completion of the process of the Link Manager obtaining the supported features of the remote BR/EDR Controller specified by the Connection_Handle event parameter.	BR/EDR
Read Remote Extended Features Command	1.2	The Read Remote Extended Features command requests a list of the supported extended features of a remote device	BR/EDR
Read Remote Extended Features Complete Event	1.2	The Read Remote Extended Features Complete event is used to indicate the completion of the process of the Link Manager obtaining the supported Extended features of the remote BR/EDR Controller specified by the Connection_Handle event parameter.	BR/EDR
Read Remote Version Information Command	1.1	The Read Remote Version Information command will read the values for the version information for the remote device associated with the Connection_Handle.	BR/EDR, LE
Read Remote Version Information Complete Event	1.1	The Read Remote Version Information Complete event is used to indicate the completion of the process of the Link Manager obtaining the version information of the remote device associated with the Connection_Handle event parameter.	BR/EDR, LE
LE Read Remote Used Features Command	4.0	The LE Read Remote Used Features Command is used to read the used features of a LE remote device.	LE
LE Read Remote Used Features Complete Event	4.0	The LE Read Remote Used Features Complete event indicates the completion of the process to read the remote used features on a remote LE device.	LE

Table 3.9: Remote information

3.9 SYNCHRONOUS CONNECTIONS

The synchronous connections group of commands and events allows synchronous connections to be created.

Name	Vers.	Summary description	Supported Controllers
Setup Synchronous Connection Command	1.2	The Setup Synchronous Connection command adds a new or modifies an existing synchronous logical transport (SCO or eSCO) on a physical link depending on the Connection_Handle parameter specified.	BR/EDR
Synchronous Connection Complete Event	1.2	The Synchronous Connection Complete event indicates to both the Hosts that a new Synchronous connection has been established.	BR/EDR
Synchronous Connection Changed Event	1.2	The Synchronous Connection Changed event indicates to the Host that an existing Synchronous connection has been reconfigured.	BR/EDR
Accept Synchronous Connection Request Command	1.2	The Accept Synchronous Connection Request command is used to accept an incoming request for a synchronous connection and to inform the local Link Manager about the acceptable parameter values for the synchronous connection.	BR/EDR
Reject Synchronous Connection Request Command	1.2	The Reject Synchronous Connection Request command is used to decline an incoming request for a synchronous link.	BR/EDR
Read Voice Setting Command	1.1	The Read Voice Setting command will read the values for the Voice Setting configuration parameter, which controls all the various settings for the voice connections.	BR/EDR
Write Voice Setting Command	1.1	The Write Voice Setting command will write the values for the Voice Setting configuration parameter, which controls all the various settings for the voice connections.	BR/EDR

Table 3.10: Synchronous connections

3.10 CONNECTION STATE

The connection state group of commands and events allows the configuration of a link, especially for low power operation.

Name	Vers.	Summary description	Supported Controllers
Mode Change Event	1.1	The Mode Change event is used to indicate that the current mode has changed.	BR/EDR
Max Slots Change Event	1.1	The Max Slots Change event it used to indicate a change in the max slots by the LM.	BR/EDR
Hold Mode Command	1.1	The Hold Mode command is used to initiate Hold Mode.	BR/EDR
Sniff Mode Command	1.1	The Sniff Mode command is used to alter the behavior of the LM and have the LM place the local or remote device into the sniff mode.	BR/EDR
Sniff Subrating Command	2.1 + EDR	The Sniff Subrating command is used to configure the sniff sub-rating parameters in the local device	BR/EDR
Sniff Subrating Event	2.1 + EDR	The Sniff Subrating event is used to inform the Host of the local and remote transmit and receive latencies	BR/EDR
Exit Sniff Mode Command	1.1	The Exit Sniff Mode command is used to end the sniff mode for a Connection_Handle which is currently in sniff mode.	BR/EDR
Park State Command	1.1	The Park State command is used to alter the behavior of the Link Manager and have the Link Manager place the local or remote device into the park state.	BR/EDR
Exit Park State Command	1.1	The Exit Park State command is used to switch the BR/EDR Controller from park state back to active mode.	BR/EDR
Read Link Policy Settings Command	1.1	The Read Link Policy Settings command will read the Link Policy configuration parameter for the specified Connection_Handle. The Link Policy settings allow the Host to specify which Link Modes the Link Manager can use for the specified Connection_Handle.	BR/EDR

Table 3.11: Connection state

Name	Vers.	Summary description	Supported Controllers
Write Link Policy Settings Command	1.1	The Write Link Policy Settings command will write the Link Policy configuration parameter for the specified Connection_Handle. The Link Policy settings allow the Host to specify which Link Modes the Link Manager can use for the specified Connection_Handle.	BR/EDR
Read Default Link Policy Settings Command	1.2	The Read Default Link Policy Settings command will read the Default Link Policy configuration parameter for all new connections.	BR/EDR
Write Default Link Policy Settings Command	1.2	The Write Default Link Policy Settings command will write the Default Link Policy configuration parameter for all new connections.	BR/EDR
LE Connection Update Command	3.14.0	The LE Connection Update Command will be used to change the connection parameters of an existing connection.	LE
LE Connection Update Complete Event	3.14.0	The LE Connection Update Complete event indicates the completion of the process to change the connection parameters.	LE

Table 3.11: Connection state

3.11 PICONET STRUCTURE

The piconet structure group of commands and events allows the discovery and reconfiguration of a piconet.

Name	Vers.	Summary description	Supported Controllers
Role Discovery Command	1.1	The Role Discovery command is used for a BR/EDR Controller to determine which role the device is performing for a particular Connection_Handle.	BR/EDR
Switch Role Command	1.1	The Switch Role command is used to switch master and slave roles of the devices on either side of a connection.	BR/EDR
Role Change Event	1.1	The Role Change event is used to indicate that the current BR/EDR Controller role related to the particular connection has been changed.	BR/EDR
Link Supervision Timeout Changed Event	2.1 + EDR	The Link Supervision Timeout Changed event is used to notify the slave's Host when the link supervision timeout value has changed	BR/EDR

Table 3.12: Piconet structure

3.12 QUALITY OF SERVICE

The quality of service group of commands and events allows the configuration of links to allow for quality of service parameters to be specified.

Name	Vers.	Summary description	Supported Controllers
Flow Specification Command	1.2	The Flow Specification command is used to specify the flow parameters for the traffic carried over the ACL connection identified by the Connection_Handle.	BR/EDR
Flow Specification Complete Event	1.2	The Flow Specification Complete event is used to inform the Host about the Quality of Service for the ACL connection the Controller is able to support.	BR/EDR
QoS Setup Command	1.1	The QoS Setup command is used to specify Quality of Service parameters for a Connection_Handle.	BR/EDR
QoS Setup Complete Event	1.1	The QoS Setup Complete event is used to indicate that QoS is setup.	BR/EDR
QoS Violation Event	1.1	The QoS Violation event is used to indicate the Controller's Link Manager or PAL is unable to provide the current QoS requirement for the Handle.	BR/EDR, AMP
Flush Command	1.1	The Flush command is used to discard all data that is currently pending for transmission in the Controller for the specified Connection_Handle.	BR/EDR
Flush Occurred Event	1.1	The Flush Occurred event is used to indicate that, for the specified Handle, the data to be transmitted has been discarded.	BR/EDR, AMP
Enhanced Flush Command	2.1 + EDR	The Enhanced Flush command is used to discard specific packets currently pending for transmission in the Controller for the specified Handle. This command takes a parameter specifying the type of packets to be flushed.	BR/EDR, AMP
Enhanced Flush Complete Event	2.1 + EDR	The Enhanced Flush Complete event is used to indicate that an Enhanced Flush is complete.	BR/EDR, AMP

Table 3.13: Quality of service

Name	Vers.	Summary description	Supported Controllers
Read Automatic Flush Timeout Command	1.1	The Read Automatic Flush Timeout command will read the value for the Flush Timeout configuration parameter for the specified Connection_Handle. The Flush Timeout parameter is only used for ACL connections.	BR/EDR
Write Automatic Flush Timeout Command	1.1	The Write Automatic Flush Timeout command will write the value for the Flush Timeout configuration parameter for the specified Connection_Handle. The Flush Timeout parameter is only used for ACL connections.	BR/EDR
Read Failed Contact Counter Command	1.1	The Read Failed Contact Counter command will read the value for the Failed Contact Counter configuration parameter for a particular connection to another device.	BR/EDR, AMP
Reset Failed Contact Counter Command	1.1	The Reset Failed Contact Counter command will reset the value for the Failed Contact Counter configuration parameter for a particular connection to another device.	BR/EDR, AMP
Read Num Broadcast Retransmissions Command	1.1	The Read Num Broadcast Retransmissions command will read the parameter value for the Number of Broadcast Retransmissions for the BR/EDR Controller.	BR/EDR
Write Num Broadcast Retransmissions Command	1.1	The Write Num Broadcast Retransmissions command will write the parameter value for the Number of Broadcast Retransmissions for the BR/EDR Controller.	BR/EDR
Flow Spec Modify Command	3.0 + HS	The Flow Spec Modify command modifies certain parameters of an existing Logical Link's Flow Spec on this AMP Controller	AMP
Flow Spec Modify Complete Event	3.0 + HS	The Flow Spec Modify Complete event indicates to the Host that a Flow Spec Modify command has been completed.	AMP
Read Best Effort Flush Timeout Command	3.0 + HS	The Read Best Effort Flush Timeout command returns the last value written with the Write Best Effort Flush Timeout command.	AMP

Table 3.13: Quality of service

Name	Vers.	Summary description	Supported Controllers
Write Best Effort Flush Timeout Command	3.0 + HS	The Write Best Effort Flush Timeout command configures the AMP device with a maximum time to attempt to transmit any given frame on the Best Effort logical link.	802.11 PAL only

Table 3.13: Quality of service

3.13 PHYSICAL LINKS

The physical links commands and events allows configuration of the physical link.

Name	Vers.	Summary description	Supported Controllers
Read Link Supervision Timeout Command	1.1	The Read Link Supervision Timeout command will read the value for the Link Supervision Timeout configuration parameter for the device. This parameter is used by the Controller to determine link loss.	BR/EDR, AMP
Write Link Supervision Timeout Command	1.1	The Write Link Supervision Timeout command will write the value for the Link Supervision Timeout configuration parameter for the device. This parameter is used by the Controller to determine link loss.	BR/EDR, AMP
Link Supervision Timeout Changed Event	2.1 + EDR	The Link Supervision Timeoutevent indicates that the remote device changed the Link Supervision Timeout.	BR/EDR
Read AFH Channel Assessment Mode Command	1.2	The Read AFH Channel Assessment Mode command will read the value for the AFH Channel Classification Mode parameter. This value is used to enable or disable the Controller's channel assessment scheme.	BR/EDR

Table 3.14: Physical links

Name	Vers.	Summary description	Supported Controllers
Write AFH Channel Assessment Mode Command	1.2	The Write AFH Channel Assessment Mode command will write the value for the Channel Classification Mode configuration parameter. This value is used to enable or disable the Controller's channel assessment scheme.	BR/EDR
Set AFH Host Channel Classification Command	1.2	The Set AFH Host Channel Classification command allows the Host to specify a channel classification based on its "local information".	BR/EDR
Change Connection Packet Type Command	1.1	The Change Connection Packet Type command is used to change which packet types can be used for a connection that is currently established.	BR/EDR
Connection Packet Type Changed Event	1.1	The Connection Packet Type Changed event is used to indicate the completion of the process of the Link Manager changing the packet type mask used for the specified Connection_Handle.	BR/EDR
Read Local AMP_ASSOC Command	3.0 + HS	The Read Local AMP_ASSOC command will return a fragment of the AMP_ASSOC structure, which contains AMP specific information for this AMP Controller.	AMP
Write Remote AMP_ASSOC Command	3.0 + HS	The Write Remote AMP_ASSOC command allows the Host to write a fragment of the AMP_ASSOC structure, which contains AMP specific information for the AMP Controller.	AMP
Short Range Mode Command	3.0 + HS	The Short Range Mode command is used to notify the AMP that the peer may be at close enough range to require modification of AMP transmission power level.	802.11 PAL only
Short Range Mode Change Complete Event	3.0 + HS	The Short Range Mode Change Complete event is used to indicate completion of any tasks relating to establishment of short range mode.	802.11 PAL only

Table 3.14: Physical links

Name	Vers.	Summary description	Supported Controllers
LE Set Host Channel Classification Command	4.0	The LE Set Host Channel Classification Command allows the Host to specify a channel classification based on its “local information.”.	LE

Table 3.14: Physical links

3.14 HOST FLOW CONTROL

The Host flow control group of commands and events allows flow control to be used towards the Host.

Name	Vers.	Summary description	Supported Controllers
Host Buffer Size Command	1.1	The Host Buffer Size command is used by the Host to notify the Controller about its buffer sizes for ACL and synchronous data. The Controller will segment the data to be transmitted from the Controller to the Host, so that data contained in HCI Data Packets will not exceed these sizes.	BR/EDR, LE
Set Event Mask Command	1.1	The Set Event Mask command is used to control which events are generated by the HCI for the Host.	All
Set Event Filter Command	1.1	The Set Event Filter command is used by the Host to specify different event filters. The Host may issue this command multiple times to request various conditions for the same type of event filter and for different types of event filters.	BR/EDR, AMP
Set Controller To Host Flow Control Command	1.1	The Set Controller To Host Flow Control command is used by the Host to turn flow control on or off in the direction from the Controller to the Host.	BR/EDR, LE
Host Number Of Completed Packets Command	1.1	The Host Number Of Completed Packets command is used by the Host to indicate to the Controller when the Host is ready to receive more HCI packets for any Connection_Handle.	All

Table 3.15: Host flow control

Name	Vers.	Summary description	Supported Controllers
Data Buffer Overflow Event	1.1	The Data Buffer Overflow event is used to indicate that the Controller's data buffers have overflowed, because the Host has sent more packets than allowed.	All
Read Synchronous Flow Control Enable Command	1.1	The Read Synchronous Flow Control Enable command provides the ability to read the Synchronous Flow Control Enable setting. By using this setting, the Host can decide if the Controller will send Number Of Completed Packets events for Synchronous Connection_Handles.	BR/EDR
Write Synchronous Flow Control Enable Command	1.1	The Write Synchronous Flow Control Enable command provides the ability to write the Synchronous Flow Control Enable setting. By using this setting, the Host can decide if the Controller will send Number Of Completed Packets events for Synchronous Connection_Handles.	BR/EDR
Set Event Mask Page 2 Command	3.0 + HS	The Set Event Mask Page 2 command is used to control which events are generated by the HCI for the Host.	BR/EDR, AMP
LE Add Device To White List Command	4.0	The LE Add Device To White List Command will add a device to the white list.	LE
LE Clear White List Command	4.0	The LE Clear White List Command will clear the white list.	LE
LE Read White List Size Command	4.0	The LE Read White List Size Command will read the maximum number of white list entries that this Controller supports.	LE
LE Remove Device From White List Command	4.0	The LE Remove Device From White List Command will remove a single device from the white list.	LE
LE Set Event Mask Command	4.0	The LE Set Event Mask Command is used to control which events are generated by the HCI for the Host.	LE

Table 3.15: Host flow control

3.15 LINK INFORMATION

The link information group of commands and events allows information about a link to be read.

Name	Vers.	Summary description	Supported Controllers
Read LMP Handle Command	1.2	The Read LMP Handle command will read the current LMP Handle associated with the Connection_Handle.	BR/EDR
Read Transmit Power Level Command	1.1	The Read Transmit Power Level command will read the values for the Transmit Power Level parameter for the specified Connection_Handle.	BR/EDR, LE
Read Link Quality Command	1.1	The Read Link Quality command will read the value for the Link Quality for the specified Connection_Handle.	BR/EDR, AMP
Read RSSI Command	1.1	The Read RSSI command will read the value for the Received Signal Strength Indication (RSSI) for a Connection_Handle to another Controller.	All
Read Clock Offset Command	1.1	The Read Clock Offset command allows the Host to read the clock offset of remote BR/EDR Controllers.	BR/EDR
Read Clock Offset Complete Event	1.1	The Read Clock Offset Complete event is used to indicate the completion of the process of the LM obtaining the Clock offset information.	BR/EDR
Read Clock Command	1.2	The Read Clock command will read an estimate of a piconet or the local Bluetooth Clock.	BR/EDR
Read AFH Channel Map Command	1.2	The Read AFH Channel Map command will read the current state of the channel map for a connection.	BR/EDR
Read Local AMP Info Command	3.0 + HS	The Read Local AMP Info command returns information about this AMP Controller.	AMP
AMP Status Change Event	3.0 + HS	The AMP Status Change event can occur at any time, after initial Read Local AMP_Info request, the Controller detects a change has occurred regarding status.	AMP
Read Enhanced Transmit Power Level Command	3.0 + HS	The Read Enhanced Transmit Power Level command will read the values for the GFSK, DQPSK and 8DPSK Transmit Power Level parameters for the specified Connection_Handle.	BR/EDR

Table 3.16: Link information

Name	Vers.	Summary description	Supported Controllers
LE Read Advertising Channel TX Power Command	4.0	The LE Read Advertising Channel TX Power Command will read the transmit power level that will be used for advertising.	LE
LE Read Channel Map Command	4.0	The LE Read Channel Map Command will read the current state of the channel map for a connection.	LE

Table 3.16: Link information

3.16 AUTHENTICATION AND ENCRYPTION

The authentication and encryption group of commands and events allows authentication of a remote device and then encryption of the link to one or more remote devices.

Name	Vers.	Summary description	Supported Controllers
Read Authentication Enable Command	1.1	The Read Authentication Enable command will read the value for the Authentication Enable parameter, which controls whether the Bluetooth device will require authentication for each connection with other Bluetooth devices.	BR/EDR
Write Authentication Enable Command	1.1	The Write Authentication Enable command will write the value for the Authentication Enable parameter, which controls whether the Bluetooth device will require authentication for each connection with other Bluetooth devices.	BR/EDR
Read Encryption Mode Command	1.1	The Read Encryption Mode command will read the value for the Encryption Mode parameter, which controls whether the Bluetooth device will require encryption for each connection with other Bluetooth devices.	BR/EDR
Write Encryption Mode Command	1.1	The Write Encryption Mode command will write the value for the Encryption Mode parameter, which controls whether the Bluetooth device will require encryption for each connection with other Bluetooth devices.	BR/EDR

Table 3.17: Authentication and encryption

Name	Vers.	Summary description	Supported Controllers
Link Key Request Event	1.1	The Link Key Request event is used to indicate that a Link Key is required for the connection with the device specified in BD_ADDR.	BR/EDR
Link Key Request Reply Command	1.1	The Link Key Request Reply command is used to reply to a Link Key Request event from the BR/EDR Controller, and specifies the Link Key stored on the Host to be used as the link key for the connection with the other BR/EDR Controller specified by BD_ADDR.	BR/EDR
Link Key Request Negative Reply Command	1.1	The Link Key Request Negative Reply command is used to reply to a Link Key Request event from the BR/EDR Controller if the Host does not have a stored Link Key for the connection with the other BR/EDR Controller specified by BD_ADDR.	BR/EDR
PIN Code Request Event	1.1	The PIN Code Request event is used to indicate that a PIN code is required to create a new link key for a connection.	BR/EDR
PIN Code Request Reply Command	1.1	The PIN Code Request Reply command is used to reply to a PIN Code Request event from the Controller and specifies the PIN code to use for a connection.	BR/EDR
PIN Code Request Negative Reply Command	1.1	The PIN Code Request Negative Reply command is used to reply to a PIN Code Request event from the Controller when the Host cannot specify a PIN code to use for a connection.	BR/EDR
Link Key Notification Event	1.1	The Link Key Notification event is used to indicate to the Host that a new Link Key has been created for the connection with the BR/EDR Controller specified in BD_ADDR.	BR/EDR
Authentication Requested Command	1.1	The Authentication Requested command is used to establish authentication between the two devices associated with the specified Connection_Handle.	BR/EDR

Table 3.17: Authentication and encryption

Name	Vers.	Summary description	Supported Controllers
Authentication Complete Event	1.1	The Authentication Complete event occurs when authentication has been completed for the specified connection.	BR/EDR
Set Connection Encryption Command	1.1	The Set Connection Encryption command is used to enable and disable the link level encryption.	BR/EDR
Encryption Change Event	1.1	The Encryption Change event is used to indicate that the change in the encryption has been completed for the Connection_Handle specified by the Connection_Handle event parameter.	BR/EDR, LE
Change Connection Link Key Command	1.1	The Change Connection Link Key command is used to force both devices of a connection associated to the Connection_Handle, to generate a new link key.	BR/EDR
Change Connection Link Key Complete Event	1.1	The Change Connection Link Key Complete event is used to indicate that the change in the Link Key for the Connection_Handle specified by the Connection_Handle event parameter had been completed.	BR/EDR
Master Link Key Command	1.1	The Master Link Key command is used to force both BR/EDR Controllers of a connection associated to the Connection_Handle to use the temporary link key of the Master device or the regular link keys.	BR/EDR
Master Link Key Complete Event	1.1	The Master Link Key Complete event is used to indicate that the change in the temporary Link Key or in the semi-permanent link keys on the Bluetooth master side has been completed.	BR/EDR
Read PIN Type Command	1.1	The Read PIN Type command is used for the Host to read the value that is specified to indicate whether the Host supports variable PIN or only fixed PINs.	BR/EDR
Write PIN Type Command	1.1	The Write PIN Type command is used for the Host to specify whether the Host supports variable PIN or only fixed PINs.	BR/EDR

Table 3.17: Authentication and encryption

Name	Vers.	Summary description	Supported Controllers
Read Stored Link Key Command	1.1	The Read Stored Link Key command provides the ability to read whether one or more link keys are stored in the Controller.	BR/EDR
Return Link Keys Event	1.1	The Return Link Keys event is used to return stored link keys after a Read Stored Link Key command is used.	BR/EDR
Write Stored Link Key Command	1.1	The Write Stored Link Key command provides the ability to write one or more link keys to be stored in the Controller.	BR/EDR
Delete Stored Link Key Command	1.1	The Delete Stored Link Key command provides the ability to remove one or more of the link keys stored in the Controller.	BR/EDR
Create New Unit Key Command	1.1	The Create New Unit Key command is used to create a new unit key.	BR/EDR
Refresh Encryption Key Command	2.1 + EDR	The Refresh Encryption Key command s used by the Host to cause the Controller to refresh the encryption key by pausing and resuming encryption	BR/EDR
Encryption Key Refresh Complete Event	2.1 + EDR	The Encryption Key Refresh Complete event is used to indicate to the Host that the encryption key was refreshed on the given Connection_Handle any time encryption is paused and then resumed.	BR/EDR, LE
IO Capability Request Reply	2.1 + EDR	The IO Capability_Request_Reply command is used to reply to an IO Capability Request event from the controller, and specifies the current I/O capabilities of the Host.	BR/EDR
User Confirmation Request Reply Command	2.1 + EDR	The User Confirmation Request Reply command is used to reply to a User Confirmation Request event and indicates that the user selected “yes”. It is also used when the Host has no input and no output capabilities.	BR/EDR

Table 3.17: Authentication and encryption

Name	Vers.	Summary description	Supported Controllers
User Confirmation Request Negative Reply Command	2.1 + EDR	The User Confirmation Request Negative Reply command is used to reply to a User Confirmation Request event and indicates that the user selected “no”. This command will terminate Secure Simple Pairing.	BR/EDR
User Passkey Request Reply Command	2.1 + EDR	The User Passkey Request Reply command is used to reply to a User Passkey Request event and specifies the Numeric_Value (passkey) entered by the user to be used in the Secure Simple Pairing process.	BR/EDR
User Passkey Request Negative Reply Command	2.1 + EDR	The User Passkey Request Negative Reply command is used to reply to a User Passkey Request event and indicates the host could not provide a passkey. This command will terminate Simple Pairing.	BR/EDR
Remote OOB Data Request Reply Command	2.1 + EDR	The Remote OOB Data Request Reply command is used to reply to a Remote OOB Data Request event with the C and R values received via an OOB transfer from a remote BR/EDR Controller identified by BD ADDR.	BR/EDR
Remote OOB Data Request Negative Reply Command	2.1 + EDR	The Remote OOB Data Request Negative Reply command is used to reply to a Remote OOB Data Request event that the Host does not have the C and R	BR/EDR
Read Simple Pairing Mode Command	2.1 + EDR	This command reads the Simple Pairing mode setting in the BR/EDR Controller.	BR/EDR
Write Simple Pairing Mode Command	2.1 + EDR	This command writes the Simple Pairing mode setting in the BR/EDR Controller.	BR/EDR
Read Local OOB Data Command	2.1 + EDR	This command is used to obtain a Simple Pairing Hash C and Simple Pairing Randomizer R which are intended to be transferred to a remote device using an OOB mechanism.	BR/EDR

Table 3.17: Authentication and encryption

Name	Vers.	Summary description	Supported Controllers
Send Keypress Notification Command	2.1 + EDR	This command is used during the Passkey Entry protocol by a device with KeyboardOnly IO capabilities. It is used by a Host to inform the remote device when keys have been entered or erased.	BR/EDR
Read Encryption Key Size Command	v3.0 + HS	The Read Encryption Key Size command is used to read the encryption key size on a given Connection_Handle.	BR/EDR
LE Encrypt Command	4.0	The LE Encrypt Command will encrypt a block of unencrypted data against a key and generate a block of encrypted data.	LE
LE Long Term Key Requested Event	4.0	The LE Long Term Key Requested event indicates that a Long Term Key is required for a connection.	LE
LE Long Term Key Requested Reply Command	4.0	The LE Long Term Key Requested Reply Command is used to reply to an LE Long Term Key Requested event and includes the Long Term Key stored in the Host for that connection.	LE
LE Long Term Key Requested Negative Reply Command	4.0	The LE Long Term Key Requested Negative Reply Command is used to reply to an LE Long Term Key Requested event and indicates that the Host does not have a Long Term Key for that connection.	LE
LE Rand Command	4.0	The LE Rand Command will generate a random number.	LE
LE Start Encryption Command	4.0	The LE Start Encryption Command is used to enable link level encryption.	LE

Table 3.17: Authentication and encryption

3.17 TESTING

The testing group of commands and events allows a device to be placed into a special testing mode to allow for testing to be performed.

Name	Vers.	Summary description	Supported Controllers
Read Loopback Mode Command	1.1	The Read Loopback Mode command will read the value for the setting of the BR/EDR Controllers' Loopback Mode. The setting of the Loopback Mode will determine the path of information.	BR/EDR
Write Loopback Mode Command	1.1	The Write Loopback Mode command will write the value for the setting of the BR/EDR Controllers Loopback Mode. The setting of the Loopback Mode will determine the path of information.	BR/EDR
Loopback Command Event	1.1	The Loopback Command event is used to loop back all commands that the Host sends to the BR/EDR Controller with some exceptions.	BR/EDR
Enable Device Under Test Mode Command	1.1	The Enable Device Under Test Mode command will allow the local Controller to enter test mode via LMP test commands. The Host issues this command when it wants the local device to be the DUT for the Testing scenarios as described in the Bluetooth Test Mode document.	BR/EDR, AMP
Write Simple Pairing Debug Mode Command	2.1 + EDR	This command configures the BR/EDR Controller to use a predefined Diffie Hellman private key for Simple Pairing to enable debug equipment to monitor the encrypted connection.	BR/EDR
AMP Receiver Report Event	3.0 + HS	The AMP Receiver Reportevent shall be sent from the AMP to the tester via the AMP Test Manager at the interval configured by the EnableAMPReceiverReports command.	AMP
AMP Start Test Event	3.0 + HS	The AMP Start Test event shall be generated when the AMP Test command has completed and the first data is ready to be sent or received .	AMP

Table 3.18: Testing

Name	Vers.	Summary description	Supported Controllers
AMP Test Command	3.0 + HS	The AMP Test command is used to configure and start a test. This command shall be only valid in AMP Test Mode.	AMP
AMP Test End Command	3.0 + HS	The AMP Test End command is used to stop any test scenario which is in progress. This command shall only be used in AMP Test Mode when a test is in progress	AMP
AMP Test End Event	3.0 + HS	The AMP Test End event shall be generated to indicate that the AMP has transmitted or received the number of frames/bursts configured.	AMP
Enable AMP Receiver Reports Command	4.0	The Enable AMP Receiver Reports command is used to enable and disable the reporting of frames received.	AMP
LE Receiver Test Command	4.0	The LE Receiver Test Command will run the LE receiver test.	LE
LE Transmitter Test Command	4.0	The LE Transmitter Test Command will run the LE transmitter test.	LE
LE Test End Command	4.0	The LE Test End Command will end the current the receiver or transmitter test.	LE

Table 3.18: Testing

3.18 ALPHABETICAL LIST OF COMMANDS AND EVENTS

Commands/Events	Group
AMP Receiver Report Event	Testing
AMP Start Test Event	Testing
AMP Status Change Event	Link Information
AMP Test Command	Testing
AMP Test End Command	Testing
AMP Test End Event	Testing
Accept Connection Request Command	Connection Setup
Accept Logical Link Command	Connection Setup
Accept Physical Link Command	Connection Setup
Accept Synchronous Connection Request Command	Connection Setup
Authentication Complete Event	Authentication and Encryption
Authentication Requested Command	Authentication and Encryption
Change Connection Link Key Command	Authentication and Encryption
Change Connection Link Key Complete Event	Authentication and Encryption
Change Connection Packet Type Command	Physical Links
Channel Selected Event	Connection Setup
Command Complete Event	Generic Events
Command Status Event	Generic Events
Connection Complete Event	Connection Setup
Connection Packet Type Changed Event	Physical Links
Connection Request Event	Connection Setup
Create Connection Cancel Command	Connection Setup
Create Connection Command	Connection Setup
Create New Unit Key Command	Authentication and Encryption
Create Logical Link Command	Connection Setup
Create Physical Link Command	Connection Setup
Data Buffer Overflow Event	Host Flow Control
Delete Stored Link Key Command	Authentication and Encryption
Disconnect Command	Connection Setup
Disconnection Complete Event	Connection Setup

Table 3.19: Alphabetical list of commands and events.

Commands/Events	Group
Disconnect Logical Link Command	Connection Setup
Disconnection Logical Link Complete Event	Connection Setup
Disconnect Physical Link Command	Connection Setup
Disconnection Physical Link Complete Event	Connection Setup
Enable AMP Receiver Reports Command	Testing
Enable Device Under Test Mode Command	Testing
Encryption Change Event	Authentication and Encryption
Encryption Key Refresh Complete Event	Authentication and Encryption
Enhanced Flush Command	Quality of Service
Enhanced Flush Complete Event	Quality of Service
Exit Park State Command	Connection State
Exit Periodic Inquiry Mode Command	Device Discovery
Exit Sniff Mode Command	Connection State
Extended Inquiry Result Event	Device Discovery
Flow Spec Modify Command	Quality of Service
Flow Spec Modify Complete Event	Quality of Service
Flow Specification Command	Quality of Service
Flow Specification Complete Event	Quality of Service
Flush Command	Quality of Service
Flush Occurred Event	Quality of Service
Hardware Error Event	Generic Events
Hold Mode Command	Connection State
Host Buffer Size Command	Host Flow Control
Host Number Of Completed Packets Command	Host Flow Control
Inquiry Cancel Command	Device Discovery
Inquiry Command	Device Discovery
Inquiry Complete Event	Device Discovery
Inquiry Result Event	Device Discovery
Inquiry Result with RSSI Event	Device Discovery
LE Read Buffer Size Command	Controller Flow Control
LE Read Local Supported Features Command	Controller Information
LE Read Supported States Command	Controller Information

Table 3.19: Alphabetical list of commands and events.

Commands/Events	Group
LE Set Advertise Enable Command	Controller Configuration
LE Set Advertising Data Command	Controller Configuration
LE Set Advertising Parameters Command	Controller Configuration
LE Set Random Address Command	Controller Configuration
LE Set Scan Response Data Command	Controller Configuration
LE Advertising Report Event	Device Discovery
LE Set Scan Enable Command	Device Discovery
LE Set Scan Parameters Command	Device Discovery
LE Connection Complete Event	Connection Setup
LE Create Connection Cancel Command	Connection Setup
LE Create Connection Command	Connection Setup
LE Read Remote Used Features Command	Remote Information
LE Read Remote Used Features Complete Event	Remote Information
LE Connection Update Command	Connection State
LE Connection Update Complete Event	Connection State
LE Set Host Channel Classification Command	Physical Links
LE Add Device To White List Command	Host Flow Control
LE Clear White List Command	Host Flow Control
LE Read White List Size Command	Host Flow Control
LE Remove Device From White List Command	Host Flow Control
LE Set Event Mask Command	Host Flow Control
LE Read Advertising Channel TX Power Command	Link Information
LE Read Channel Map Command	Link Information
-LE Encrypt Command	Authentication and Encryption
LE Long Term Key Requested Event	Authentication and Encryption
LE Long Term Key Requested Reply Command	Authentication and Encryption
LE Long Term Key Requested Negative Reply Command	Authentication and Encryption
LE Rand Command	Authentication and Encryption
LE Start Encryption Command	Authentication and Encryption
LE Receiver Test Command	Testing
LE Transmitter Test Command	Authentication and Encryption
LE Test End Command	Authentication and Encryption

Table 3.19: Alphabetical list of commands and events.

Commands/Events	Group
Link Key Notification Event	Authentication and Encryption
Link Key Request Event	Authentication and Encryption
Link Key Request Negative Reply Command	Authentication and Encryption
Link Key Request Reply Command	Authentication and Encryption
Logical Link Cancel Command	Connection State
Logical Link Complete Event	Connection State
Loopback Command Event	Testing
Master Link Key Command	Authentication and Encryption
Master Link Key Complete Event	Authentication and Encryption
Max Slots Change Event	Connection State
Mode Change Event	Connection State
Number Of Completed Data Blocks Event	Controller Flow Control
Number Of Completed Packets Event	Controller Flow Control
Page Scan Repetition Mode Change Event	Connection Setup
Park State Command	Connection State
Periodic Inquiry Mode Command	Device Discovery
Physical Link Complete Event	Connection State
Physical Link Loss Early Warning Event	Connection State
Physical Link Recovery Event	Connection State
PIN Code Request Event	Authentication and Encryption
PIN Code Request Negative Reply Command	Authentication and Encryption
PIN Code Request Reply Command	Authentication and Encryption
QoS Setup Command	Quality of Service
QoS Setup Complete Event	Quality of Service
QoS Violation Event	Quality of Service
Read AFH Channel Assessment Mode Command	Physical Links
Read AFH Channel Map Command	Link Information
Read Authentication Enable Command	Authentication and Encryption
Read Automatic Flush Timeout Command	Quality of Service
Read BD_ADDR Command	Controller Information
Read Best Effort Flush Timeout Command	Quality of Service
Read Buffer Size Command	Controller Flow Control

Table 3.19: Alphabetical list of commands and events.

Commands/Events	Group
Read Class of Device Command	Controller Information
Read Clock Command	Link Information
Read Clock Offset Command	Link Information
Read Clock Offset Complete Event	Link Information
Read Connection Accept Timeout Command	Connection Setup
Read Current IAC LAP Command	Controller Information
Read Data Block Size Command	Controller Flow Control
Read Default Link Policy Settings Command	Connection State
Read Default Erroneous Data Reporting	Controller Configuration
Read Encryption Key Size Command	Authentication and Encryption
Read Encryption Mode Command	Authentication and Encryption
Read Enhanced Transmit Power Level Command	Link Information
Read Extended Inquiry Response Command	Controller Configuration
Read Failed Contact Counter Command	Quality of Service
Read Flow Control Mode Command	Controller Flow Control
Read Hold Mode Activity Command	Connection State
Read Inquiry Mode Command	Device Discovery
Read Inquiry Scan Activity Command	Device Discovery
Read Inquiry Scan Type Command	Device Discovery
Read Link Policy Settings Command	Connection State
Read Link Quality Command	Link Information
Read Link Supervision Timeout Command	Physical Links
Read LMP Handle Command	Link Information
Read Local AMP ASSOC Command	Physical Links
Read Local AMP Info Command	Link Information
Read Local Extended Features Command	Controller Information
Read Local Name Command	Controller Configuration
Read Local Supported Commands Command	Controller Information
Read Local Supported Features Command	Controller Information
Read Local Version Information Command	Controller Information
Read Logical Link Accept Timeout Command	Connection Setup
Read Loopback Mode Command	Testing

Table 3.19: Alphabetical list of commands and events.

Commands/Events	Group
Read Num Broadcast Retransmissions Command	Quality of Service
Read Number Of Supported IAC Command	Controller Information
Read Page Scan Activity Command	Connection Setup
Read Page Scan Type Command	Connection Setup
Read Page Timeout Command	Connection Setup
Read PIN Type Command	Authentication and Encryption
Read Location Data Command	Controller Information
Read Remote Extended Features Command	Remote Information
Read Remote Extended Features Complete Event	Remote Information
Read Remote Supported Features Command	Remote Information
Read Remote Supported Features Complete Event	Remote Information
Read Remote Version Information Command	Remote Information
Read Remote Version Information Complete Event	Remote Information
Read RSSI Command	Link Information
Read Scan Enable Command	Controller Information
Read Stored Link Key Command	Authentication and Encryption
Read Synchronous Flow Control Enable Command	Host Flow Control
Read Transmit Power Level Command	Link Information
Read Voice Setting Command	Synchronous Connections
Refresh Encryption Key Command	Authentication and Encryption
Reject Connection Request Command	Connection Setup
Reject Synchronous Connection Request Command	Connection Setup
Remote Name Request Cancel Command	Remote Information
Remote Name Request Command	Remote Information
Remote Name Request Complete Event	Remote Information
Reset Command	Device Setup
Reset Failed Contact Counter Command	Quality of Service
Return Link Keys Event	Authentication and Encryption
Role Change Event	Piconet Structure
Role Discovery Command	Piconet Structure
Set AFH Host Channel Classification Command	Physical Links
Set Connection Encryption Command	Authentication and Encryption

Table 3.19: Alphabetical list of commands and events.

Commands/Events	Group
Set Controller To Host Flow Control Command	Host Flow Control
Set Event Filter Command	Host Flow Control
Set Event Mask Command	Host Flow Control
Set Event Mask Page 2 Command	Host Flow Control
Setup Synchronous Connection Command	Synchronous Connections
Short Range Mode Command	Connection Setup
Short Range Mode Change Complete Event	Connection Setup
Sniff Mode Command	Connection State
Sniff Subrating Command	Connection State
Sniff Subrating Event	Connection State
Switch Role Command	Piconet Structure
Synchronous Connection Changed Event	Synchronous Connections
Synchronous Connection Complete Event	Synchronous Connections
Write AFH Channel Assessment Mode Command	Physical Links
Write Authentication Enable Command	Authentication and Encryption
Write Automatic Flush Timeout Command	Quality of Service
Write Best Effort Flush Timeout Command	Quality of Service
Write Class of Device Command	Controller Information
Write Connection Accept Timeout Command	Connection Setup
Write Current IAC LAP Command	Controller Information
Write Default Link Policy Settings Command	Connection State
Write Default Erroneous Data Reporting	Controller Configuration
Write Extended Inquiry Response Command	Controller Configuration
Write Flow Control Mode Command	Controller Flow Control
Write Hold Mode Activity Command	Connection State
Write Inquiry Mode Command	Device Discovery
Write Inquiry Scan Activity Command	Device Discovery
Write Inquiry Scan Type Command	Device Discovery
Write Link Policy Settings Command	Connection State
Write Link Supervision Timeout Command	Physical Links
Write Local Name Command	Controller Information
Write Location Data Command	Controller Information

Table 3.19: Alphabetical list of commands and events.

Commands/Events	Group
Write Logical Link Accept Timeout Command	Connection Setup
Write Loopback Mode Command	Testing
Write Num Broadcast Retransmissions Command	Quality of Service
Write Page Scan Activity Command	Connection Setup
Write Page Scan Type Command	Connection Setup
Write Page Timeout Command	Connection Setup
Write PIN Type Command	Authentication and Encryption
Write Remote AMP ASSOC Command	Physical Links
Write Scan Enable Command	Controller Configuration
Write Stored Link Key Command	Authentication and Encryption
Write Synchronous Flow Control Enable Command	Host Flow Control
Write Voice Setting Command	Synchronous Connections

Table 3.19: Alphabetical list of commands and events.

3.19 LE CONTROLLER REQUIREMENTS

Table 3.20 lists the commands and events that a Controller supporting LE shall implement if HCI is claimed. The Link Layer Features are defined in Part B, Section 4.6, Active Slave Broadcast Transport.

Name	LE Feature Requirements
Command Complete Event	Mandatory
Command Status Event	Mandatory
LE Add Device To White List Command	Mandatory
LE Clear White List Command	Mandatory
LE Read Buffer Size Command	Mandatory
LE Read Local Supported Features Command	Mandatory
LE Read Supported States Command	Mandatory
LE Read White List Size Command	Mandatory
LE Remove Device From White List Command	Mandatory
LE Set Event Mask Command	Mandatory
LE Test End Command	Mandatory
Read BD_ADDR Command	Mandatory
Read Local Supported Features Command	Mandatory

Table 3.20: Bluetooth Controller supporting LE requirements

Read Local Version Information Command	Mandatory
Reset Command	Mandatory
Set Event Mask Command	Mandatory
LE Read Advertising Channel TX Power Command	C1
LE Transmitter Test Command	C1
LE Set Advertise Enable Command	C1
LE Set Advertising Data Command	C1
LE Set Advertising Parameters Command	C1
LE Set Random Address Command	C1
LE Advertising Report Event	C2
LE Receiver Test Command	C2
LE Set Scan Enable Command	C2
LE Set Scan Parameters Command	C2
Disconnect Command	C3
Disconnection Complete Event	C3
LE Connection Complete Event	C3
LE Connection Update Command	C3
LE Connection Update Complete Event	C3
LE Create Connection Cancel Command	C3
LE Create Connection Command	C3
LE Read Channel Map Command	C3
LE Read Remote Used Features Command	C3
LE Read Remote Used Features Complete Event	C3
LE Set Host Channel Classification Command	C3
LE Set Scan Response Data Command	C3
Number Of Completed Packets Event	C3
Read Transmit Power Level Command	C3
Read Remote Version Information Command	C3
Read Remote Version Information Complete Event	C3
Read RSSI Command	C3
Encryption Change Event	C4
Encryption Key Refresh Complete Event	C4
LE Encrypt Command	C4

Table 3.20: Bluetooth Controller supporting LE requirements

LE Long Term Key Request Event	C4
LE Long Term Key Request Reply Command	C4
LE Long Term Key Request Negative Reply Command	C4
LE Rand Command	C4
LE Start Encryption Command	C4
Read Buffer Size Command	C5
Read LE Host Support	C5
Write LE Host Support	C5
Data Buffer Overflow Event	Optional
Hardware Error Event	Optional
Host Buffer Size Command	Optional
Host Number Of Completed Packets Command	Optional
Set Controller To Host Flow Control Command	Optional

Table 3.20: Bluetooth Controller supporting LE requirements

C1: Mandatory if Controller supports transmitting packets, otherwise optional.C2: Mandatory if Controller supports receiving packets, otherwise optional.

C3: Mandatory if Controller supports transmitting and receiving packets, otherwise optional.

C4: Mandatory if LE Feature (LL Encryption) is supported otherwise optional.

C5: Mandatory if BR/EDR is supported otherwise optional

4 HCI FLOW CONTROL

Flow control for data shall be used in the direction from the Host to the Controller to avoid overflowing the Controller data buffers with ACL data destined for a remote device (using a Connection_Handle) that is not responding. The Host manages the data buffers of the Controller. For Primary Controller, packet based flow control is the default. For AMP Controllers, data block based data flow control is the default for ACL traffic. Flow control for data moving from the Controller to the Host may be used in the Primary Controller in accordance with Section 4.2.

Command flow control is covered in Section 4.4 and Section 4.5.

4.1 HOST TO CONTROLLER DATA FLOW CONTROL

Two methods of data flow control are defined: 'packet-based' flow control and 'data-block-based' flow control, known as buffer management. Selection of the data flow control mechanism is performed with the Write Flow Control Mode command (see Section 7.3.73).

If a BR/EDR/LE Controller implements separate buffers for ACL Data:

1. The Host shall use the LE Read Buffer Size command to determine the buffers that are used for ACL Data on an LE-U logical link.
2. The Host shall use separate packet based flow control for each set of buffers.
3. The Connection_Handle contained in the ACL Data packet shall be used by the Controller to determine which set of buffers to use and the logical link (ACL-U or LE-U) over which the data is to be sent.

If a BR/EDR/LE Controller does not implement separate buffers, then all ACL Data shall use the BR/EDR buffer management as described below, and only the logical link (ACL-U or LE-U) shall be determined by the Connection_Handle.

4.1.1 Packet-based Data Flow Control

When the packet based flow control mechanism is enabled, on initialization, a Host that supports LE shall issue the LE Read Buffer Size command. Two of the return parameters of this command determine the maximum size of HCI ACL (excluding header) Data Packets that can be used to transmit ACL data for an LE transport sent from the Host to the Controller. There is an additional return parameter that specifies the total number of HCI ACL Data Packets that the Controller may have waiting for transmission in those buffers. A Primary Controller may return zero for the total number of HCI ACL Packets used to transmit ACL data for an LE transport. In this case the Host shall then send all

BR/EDR and LE data using the HCI ACL Data Packets into the buffers identified using the Read Buffer Size command.

In a BR/EDR Controller, when the packet based flow control mechanism is enabled, on initialization, the Host shall issue the Read Buffer Size command. Two of the return parameters of this command determine the maximum size of HCI ACL and synchronous Data Packets (excluding header) sent from the Host to the Controller. There are also two additional return parameters that specify the total number of HCI ACL and synchronous Data Packets that the Controller may have waiting for transmission in its buffers.

When there is at least one connection to another device, or when in local loopback mode on a BR/EDR Controller, the Controller shall use the Number Of Completed Packets event to control the flow of data from the Host. This event contains a list of Connection_Handles and a corresponding number of HCI Data Packets that have been completed (transmitted, flushed, or looped back to the Host) since the previous time the event was returned (or since the connection was established, if the event has not been returned before for a particular Connection_Handle).

The Host chooses the Connection Handles for the following HC Date Packets based on the information returned in this event, and/or the LE Read Buffer Size commands.

Every time it has sent an HCI Data Packet, the Host shall assume that the free buffer space for the corresponding link type (ACL, SCO or eSCO) in the Controller has decreased by one HCI Data Packet.

Each Number Of Completed Packets event received by the Host provides information about how many HCI Data Packets have been completed (transmitted or flushed) for each Connection_Handle since the previous Number Of Completed Packets event was sent to the Host. It can then calculate the actual current buffer usage.

When the Controller has completed one or more HCI Data Packet(s) it shall send a Number Of Completed Packets event to the Host, until it finally reports that all the pending HCI Data Packets have been completed. The frequency at which this event is sent is manufacturer specific.

Note: The Number Of Completed Packets events will not report on synchronous Connection_Handles if Synchronous Flow Control is disabled. (See Read Synchronous Flow Control Enable Command, and Write Synchronous Flow Control Enable Command.)

For each individual Handle, the data shall be sent to the Controller in HCI Data Packets in the order in which it was created in the Host. The Controller shall also transmit data on the air that is received from the Host for a given Handle in the same order as it is received from the Host.

Data that is received on the air from another device shall, for the corresponding Connection_Handle, be sent in HCI Data Packets to the Host in the same order as it is received. This means the scheduling shall be decided separately for each Handle basis. For each individual Handle, the order of the data shall not be changed from the order in which the data has been created.

HCI ACL Data Packets on an LE-U logical link shall be 27 octets or larger. A Host shall not fragment HCI ACL Data Packets on an LE-U logical link that are 27 octets or less in length.

4.1.2 Data-Block-Based Data Flow Control

When the data-block-based flow control mechanism is enabled, on initialization the Host shall issue the Read Data Block Size Command. Two of the return parameters of this command determine the maximum size of HCI ACL Data Packets (excluding header) sent from the Host to the Controller. Two additional return parameters specify the total number of HCI ACL Data Packets that the Controller may have waiting for transmission in its buffers.

When there is at least one Logical Link to another AMP the Controller shall use the Number Of Completed Data Blocks event to control the flow of data from the Host. This event contains a list of Handles and a corresponding number of HCI Data Packets that have been completed (transmitted or flushed) since the previous time the event was returned (or since the link was established, if the event has not been returned before for a particular Handle).

Based on the information returned in this event, and the return parameters of the Read Data Block Size command that specify the total number of HCI ACL Data Packets that can be stored in the Controller, the Host decides for which Handles the following HCI Data Packets should be sent.

Every time it has sent an HCI Data Packet, the Host shall assume that the free buffer space for the corresponding ACL link type in the Controller has decreased by one HCI Data Packet.

Each Number Of Completed Data Blocks event received by the Host provides information about how many HCI Data Packets have been completed (transmitted or flushed) for each Handle since the previous Number Of Completed Data Blocks event was sent to the Host. It can then calculate the actual current buffer usage.

When the Controller has completed one or more HCI Data Packet(s) it shall send a Number Of Completed Data Blocks event to the Host until it finally reports that all the pending HCI Data Packets have been completed. The frequency at which this event is sent is manufacturer specific.

For each individual Handle, the data must be sent to the Controller in HCI Data Packets in the order in which it was created in the Host. The Controller shall

also transmit data for a given Handle in the same order as it is received from the Host.

Data that is received shall be sent in HCI Data Packets to the Host in the same order as it is received. This means the scheduling shall be decided separately for each Handle basis.

4.2 CONTROLLER TO HOST DATA FLOW CONTROL

In some implementations, flow control may also be necessary in the direction from the Controller to the Host. The Set Host Controller To Host Flow Control command can be used to turn flow control on or off in that direction. For AMP Controllers, Controller to Host data flow control is provided by the HCI physical transport or software equivalent.

On initialization, the Host uses the Host Buffer Size command to notify the Controller about the maximum size of HCI ACL and synchronous Data Packets sent from the Controller to the Host. The command also contains two additional command parameters to notify the Primary Controller about the total number of ACL and synchronous Data Packets that can be stored in the data buffers of the Host.

The Host uses the Host Number Of Completed Packets command in exactly the same way as the Primary Controller uses the Number Of Completed Packets event as was previously described in this section.

The Host Number Of Completed Packets command is a special command for which no command flow control is used, and which can be sent anytime there is a connection or when in local loopback mode. The command also has no event after the command has completed. This makes it possible for the flow control to work in exactly the same way in both directions, and the flow of normal commands will not be disturbed.

4.3 DISCONNECTION BEHAVIOR

When the Host receives a Disconnection Complete, Disconnection Physical Link Complete or Disconnection Logical Link Complete event, the Host shall assume that all unacknowledged HCI Data Packets that have been sent to the Controller for the returned Handle have been flushed, and that the corresponding data buffers have been freed. A Primary Controller does not have to notify the Host about this in a Number Of Completed Packets event, nor does the AMP Controller have to notify the Host about this in a Number Of Completed Data Blocks event.

If flow control is also enabled in the direction from the Controller to the Host, the Controller may, after it has sent a Disconnection Complete event, assume that the Host will flush its data buffers for the sent Handle when it receives the Disconnection Complete event. The Host does not have to notify the Primary Controller about this in a Host Number Of Completed Packets command.

4.4 COMMAND FLOW CONTROL

On initial power-on, and after a reset, the Host shall send a maximum of one outstanding HCI Command Packet until a Command Complete or Command Status event has been received.

The Command Complete and Command Status events contain a parameter called Num HCI Command Packets, which indicates the number of HCI Command Packets the Host is currently allowed to send to the Controller. The Controller may buffer one or more HCI command packets, but the Controller shall start performing the commands in the order in which they are received. The Controller can start performing a command before it completes previous commands. Therefore, the commands do not always complete in the order they are started.

To indicate to the Host that the Controller is ready to receive HCI command packets, the Controller may generate a Command Complete event with the Command Opcode 0x0000, and the Num HCI Command Packets event parameter is set to 1 or more. Command Opcode 0x0000 is a NOP (No OPeration), and can be used to change the number of outstanding HCI command packets that the Host can send. The Controller may generate a Command Complete event with the Num HCI Command Packets event parameter set to zero to inform Host it must stop sending commands.

For most commands, a Command Complete event shall be sent to the Host when the Controller completes the command. Some commands are executed in the background and do not return a Command Complete event when they have been completed. Instead, the Controller shall send a Command Status event back to the Host when it has begun to execute the command. When the actions associated with the command have finished, an event that is associated with the command shall be sent by the Controller to the Host.

If the command does not begin to execute (for example, if there was a parameter error or the command is currently not allowed), the Command Status event shall be returned with the appropriate error code in the Status parameter, and the event associated with the sent command shall not be returned.

4.5 COMMAND ERROR HANDLING

If an error occurs for a command for which a Command Complete event is returned, the Return Parameters field may not contain all the return parameters specified for the command. The Status parameter, which explains the error reason and which is the first return parameter, shall always be returned. If there is a Handle parameter or a BD_ADDR parameter right after the Status parameter, this parameter shall also be returned so that the Host can identify to which instance of a command the Command Complete event belongs. In this case, the Handle or BD_ADDR parameter shall have exactly the same value as that in the corresponding command parameter. It is implementation specific whether more parameters will be returned in case of an error.

The above also applies to commands that have associated command specific completion events with a status parameter in their completion event, with three exceptions. The first two exceptions are the Connection Complete and the Synchronous Connection Complete events. On failure, for these two events only, the second parameter, Connection_Handle, is not valid and the third parameter, BD_ADDR, is valid for identification purposes. The third exception is the Logical Link Complete event. On failure, for this event, the second parameter (Logical_Link_Handle) is not valid but the third parameter (Physical_Link_Handle) is valid for identification purposes. The validity of other parameters is likewise implementation specific for failed commands in this group.

Note: The BD_ADDR return parameter of the command Read BD_ADDR is not used to identify to which instance of the Read BD ADDR command the Command Complete event belongs. It is optional for the Controller to return this parameter in case of an error.

5 HCI DATA FORMATS

5.1 INTRODUCTION

The HCI provides a uniform command method of accessing Controller capabilities. The HCI Link commands provide the Host with the ability to control connections to other BR/EDR Controllers. For the BR/EDR Controller, these commands typically involve the Link Manager (LM) to exchange LMP commands or the Link Layer (LL) to exchange LL Control packets with remote Bluetooth devices. For details see "Part C, Link Manager Protocol Specification" and see [Vol 6] Part B, Link Layer Specification. For AMP Controllers, these commands typically involve the AMP PAL. For details see appropriate PAL specification (see Part A, Section 2, Frequency Bands and Channel Arrangement).

The HCI Policy commands are used to affect the behavior of the local and remote LM or LL. These Policy commands provide the Host with methods of influencing how the LM or LL manages the piconet. The Controller & Baseband, Informational, and Status commands provide the Host access to various registers in the Controller.

HCI commands may take different amounts of time to be completed. Therefore, the results of commands will be reported back to the Host in the form of an event. For example, for most HCI commands the Controller will generate the Command Complete event when a command is completed. This event contains the return parameters for the completed HCI command. For enabling the Host to detect errors on the HCI-Transport Layer, there needs to be a timeout between the transmission of the Host's command and the reception of the Controller's response (e.g. a Command Complete or Command Status event). Since the maximum response timeout is strongly dependent on the HCI-Transport Layer used, it is recommended to use a default value of one second for this timer. This amount of time is also dependent on the number of commands unprocessed in the command queue.

5.2 DATA AND PARAMETER FORMATS

- All values are in Binary and Hexadecimal Little Endian formats unless otherwise noted
- In addition, all parameters which can have negative values must use 2's complement when specifying values
- Arrayed parameters are specified using the following notation: ParameterA[i]. If more than one set of arrayed parameters are specified (e.g. ParameterA[i], ParameterB[i]), then the order of the parameters are as follows: ParameterA[0], ParameterB[0], ParameterA[1], ParameterB[1], ParameterA[2], ParameterB[2], … ParameterA[n], ParameterB[n]

- Unless noted otherwise, all parameter values are sent and received in Little Endian format (i.e. for multi-octet parameters the rightmost (Least Signification Octet) is transmitted first)
- All command and event parameters that are not-arrayed and all elements in an arrayed parameter have fixed sizes (an integer number of octets). The parameters and the size of each not arrayed parameter (or of each element in an arrayed parameter) contained in a command or an event is specified for each command or event. The number of elements in an arrayed parameter is not fixed.
- Where bit strings are specified, the low order bit is the right hand bit, e.g. 0 is the low order bit in '10'.
- Values or parameters marked as Reserved for Future Use, shall be set to 0 unless explicitly stated otherwise on transmission, and shall be ignored on reception. Parameter values or opcodes that an implementation does not know how to interpret shall be ignored, and the operation that is being attempted shall be completed with the correct signaling. The host or controller shall not stop functioning because of receiving a reserved value.

5.3 HANDLES

Three types of handles are used to identify logical channels between the Host and a Controller: Connection Handles, Logical Link Handles, and Physical Link Handles.

5.3.1 Primary Controller Handles

Connection Handles are used to identify logical channels between the Host and the Primary Controller. Connection Handles are assigned by the Primary Controller when a new logical link is created, using the Connection Complete, Synchronous Connection Complete, or LE Connection Complete events. Broadcast Connection Handles are handled differently, and are described in Section 5.3.1.1.

Note: There are no Broadcast or Synchronous Connection Handles in a Controller that only supports LE.

5.3.1.1 Broadcast Connection Handles

The first time the Host sends an HCI Data Packet with Broadcast_Flag set to 01b (active slave broadcast) or 10b (parked slave broadcast) after a power-on or a reset, the value of the Connection Handle parameter must be a value which is not currently assigned by the Host Controller. The Host must use different Connection Handles for active broadcast and piconet broadcast.

The BR/EDR Controller must then continue to use the same connection Handles for each type of broadcast until a reset is made. Note: The Host Controller

must not send a Connection Complete event containing a new Connection_Handle that it knows is used for broadcast.

Note: In some situations, it may happen that the Host Controller sends a Connection Complete event before having interpreted a Broadcast packet received from the Host, and that the Connection_Handles of both Connection Complete event and HCI Data packet are the same. This conflict has to be avoided as follows:

If a Connection Complete event is received containing one of the Connection Handles used for broadcast, the Host has to wait before sending any packets for the new connection until it receives a Number Of Completed Packets event indicating that there are no pending broadcast packets belonging to the Connection Handle. In addition, the Host must change the Connection_Handle used for the corresponding type of broadcast to a Connection_Handle which is currently not assigned by the Host Controller.

This Connection_Handle must then be used for all the following broadcasts of that type until a reset is performed or the same conflict situation happens again. However, this will occur very rarely.

The Host Controller must, in the above conflict case, be able to distinguish between the Broadcast message sent by the Host and the new connection made (this could be even a new synchronous link) even though the Connection Handles are the same.

For an HCI Data Packet sent from the Host Controller to the Host where the Broadcast_Flag is 01 or 10, the Connection_Handle parameter should contain the Connection Handle for the ACL connection to the master that sent the broadcast.

5.3.2 AMP Controller Handles

AMP Controllers have two types of handles: Physical Link Handles and Logical Link Handles. For data, command and event operations between the Host and an AMP Controller, a Logical Link Handle is used where a Connection Handle is specified, unless a Physical Link Handle is explicitly specified.

A Physical Link is an association between the local Controller and a remote Controller. For any AMP Controller, there can be at most one Physical Link. The Host is responsible for allocating Physical Link Handles. A value of 0 shall not be used for a Physical Link Handle. Physical Links are created and removed explicitly, and encryption keys are established on a per Physical Link basis. The AMP may support one or more Physical Links (representing different remote devices) at any one time.

For each Physical Link, an AMP Controller provides zero or more Logical Links.

A Logical Link is an allocation of bandwidth or other resources for application level data transfer between the local AMP Controller and a remote AMP Controller. A Logical Link exists with respect to a specific Physical Link and the Controller ensures this Handle's uniqueness. The AMP shall support one or more Logical Links (representing distinct resource allocations) over any one Physical Link.

5.4 EXCHANGE OF HCI-SPECIFIC INFORMATION

The Host Controller Transport Layer provides transparent exchange of HCI specific information. These transporting mechanisms provide the ability for the Host to send HCI commands, ACL data and synchronous data to the BR/EDR Controller, and HCI commands and ACL data to the LE Controller or AMP Controller. These transport mechanisms also provide the ability for the Host to receive HCI events, ACL data and synchronous data from the BR/EDR Controller, and HCI events and ACL data from the LE Controller or AMP Controller. Since the Host Controller Transport Layer provides transparent exchange of HCI-specific information, the HCI specification specifies the format of the commands, events, and data exchange between the Host and the Controller(s). The next sections specify the HCI packet formats.

5.4.1 HCI Command Packet

The HCI Command Packet is used to send commands to the Controller from the Host. The format of the HCI Command Packet is shown in Figure 5.1, and the definition of each field is explained below.

Controllers shall be able to accept HCI Command Packets with up to 255 bytes of data excluding the HCI Command Packet header.

Each command is assigned a 2 byte Opcode used to uniquely identify different types of commands. The Opcode parameter is divided into two fields, called the OpCode Group Field (OGF) and OpCode Command Field (OCF). The OGF occupies the upper 6 bits of the Opcode, while the OCF occupies the remaining 10 bits. The OGF of 0x3F is reserved for vendor-specific debug commands. The organization of the opcodes allows additional information to be inferred without fully decoding the entire Opcode.

Note: The OGF composed of all ‘ones’ has been reserved for vendor-specific debug commands. These commands are vendor-specific and are used during manufacturing, for a possible method for updating firmware, and for debugging.

1. A Command Status Event. If the status indicates success (Section 7.7.15) then this event shall be followed by an HCI Event with Event Code field of 0xFF (Section 5.4.4).
2. A Command Complete Event specifying the corresponding Vendor Specific Debug command opcode.

The host shall assume that sending of a Vendor Specific Debug command will consume an HCI Command credit.

Note: The OGF composed of all ‘zeros’ and an OCF or all ‘zeros’ is the NOP command. This command Opcode may be used in Command Flow Control. (See Section 4.4 on page 422.)

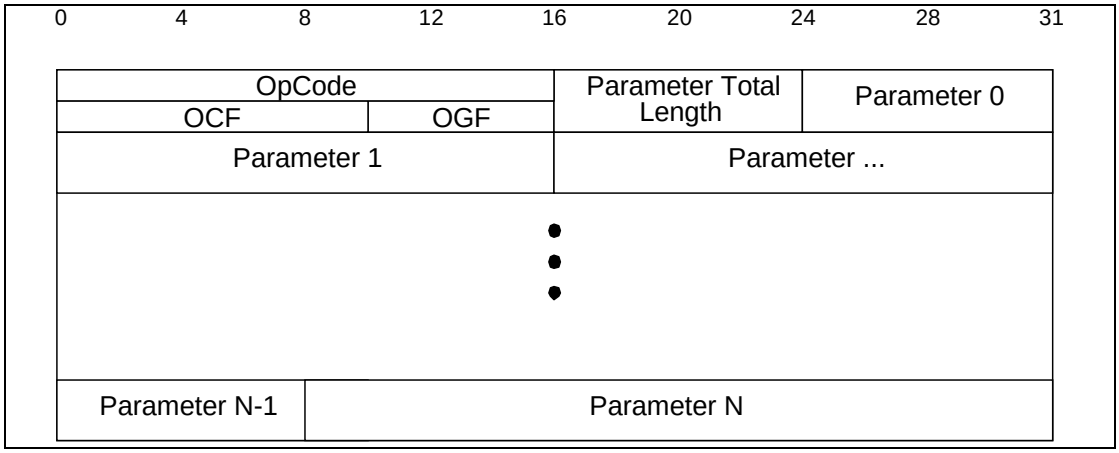


Figure 5.1: HCI Command Packet

Op_Code: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	OGF Range (6 bits): 0x00-0x3F (0x3F reserved for vendor-specific debug commands) OCF Range (10 bits): 0x0000-0x03FF

Parameter_Total_Length: *Size: 1 Octet*

Value	Parameter Description
0xXX	Lengths of all of the parameters contained in this packet measured in octets. (N.B.: total length of parameters, not number of parameters)

Parameter 0 - N: *Size: Parameter Total Length*

Value	Parameter Description
0xXX	Each command has a specific number of parameters associated with it. These parameters and the size of each of the parameters are defined for each command. Each parameter is an integer number of octets in size.

5.4.2 HCI ACL Data Packets

HCI ACL Data Packets are used to exchange data between the Host and Controller. There are two types of HCI ACL Data Packets:

- Automatically-Flushable

- Non-Automatically-Flushable

Automatically-Flushable HCI Data Packets are flushed based on the setting of an automatic flush timer (see Read Automatic Flush Timeout Command). Non-Automatically-Flushable HCI Data Packets are not controlled by the automatic flush timeout and shall not be automatically flushed. The format of the HCI ACL Data Packet is shown in Figure 5.2. The definition for each of the fields in the data packets is explained below.

Note: HCI ACL Data Packets with a Connection_Handle associated with an LE-U logical link will not be affected by the automatic flush timer because only non-flushable packet boundary flags are allowed.

All packets on an AMP logical link are affected by the automatic flush timer.

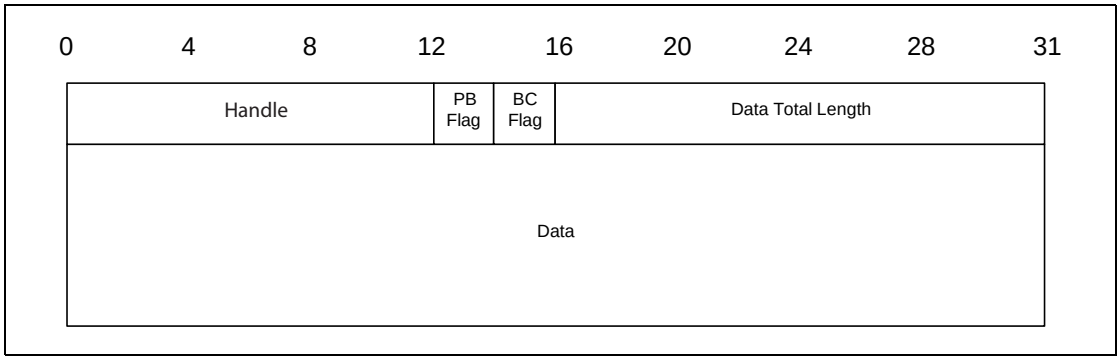


Figure 5.2: HCI ACL Data Packet

Handle: *Size: 12 Bits*

Value	Parameter Description
0xXXX	Connection_Handle to be used for transmitting a data packet or segment over a Primary Controller. Logical_Link_Handle to be used for transmitting a data packet over an AMP Controller. On the receiving side (AMP Controller to Host), the least significant 8 bits of the Handle is the Physical_Link_Handle and the most significant 4 bits are reserved. (This is because certain AMP Controllers might not be able to provide the logical link context on the receiving side, and the Host does not require it.) Range: 0x000-0xEFF (0xF00 - 0xFFF Reserved for future use)

Flags: *Size: 2 Bits*

The Flag Bits consist of the Packet_Boundary_Flag and Broadcast_Flag. The Packet_Boundary_Flag is located in bit 4 and bit 5, and the Broadcast_Flag is located in bit 6 and bit 7 in the second octet of the HCI ACL Data packet.

Packet_Boundary_Flag: *Size: 2 Bits*

Value	Parameter Description		ACL-U	AMP-U	LE-U
00	First non-automatically-flushable packet of Higher Layer Message (start of a non-automatically-flushable L2CAP PDU) from Host to Controller.	Host to Controller	Allowed	Not allowed	Allowed
		Controller to Host	Not allowed (except during loopback)	Not allowed	Not allowed
01	Continuing fragment of Higher Layer Message	Host to Controller	Allowed	Not allowed	Allowed
		Controller to Host	Allowed	Not allowed	Allowed
10	First automatically flushable packet of Higher Layer Message (start of an automatically-flushable L2CAP PDU).	Host to Controller	Allowed	Not allowed	Not Allowed
		Controller to Host	Allowed	Not allowed	Allowed
11	A complete L2CAP PDU. Automatically flushable.	Host to Controller	Allowed	Allowed	Not Allowed
		Controller to Host	Not Allowed	Allowed	Not Allowed

The start of a non-flushable packet of a Higher Layer Message (start of a non-automatically-flushable L2CAP PDU) with the PBF of 00 shall be transmitted with an LLID of 10. All continuing fragment packets of a Higher Layer Message shall be transmitted with an LLID of 01.

Broadcast_Flag (in packet from Host to Controller): *Size: 2 Bits*

Value	Parameter Description
00	No broadcast. Only point-to-point (this is the only valid option for AMPs).
01	Active Slave Broadcast: packet is sent to all active slaves (i.e. packet is usually not sent during park beacon slots), and it may be received by slaves in sniff mode or park state.
10	Parked Slave Broadcast: packet is sent to all slaves and all slaves in park state (i.e. packet is sent during park beacon slots if there are parked slaves), and it may be received by slaves in sniff mode.
11	Reserved for future use.

Broadcast_Flag (in packet from Controller to Host): *Size: 2 Bits*

Value	Parameter Description

00	Point-to-point
01	BR/EDR Packet received as a slave not in park state (either Active Slave Broadcast or Parked Slave Broadcast)
10	BR/EDR Packet received as a slave in park state (Parked Slave Broadcast)
11	Reserved for future use.

Note: Active slave broadcast packets may be sent in park beacon slots.

Note: Slaves in sniff mode may or may not receive a broadcast packet depending on whether they happen to be listening at sniff slots, when the packet is sent.

Data_Total_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Length of data measured in octets.

5.4.3 HCI Synchronous Data Packets

HCI synchronous (SCO and eSCO) Data Packets are used to exchange synchronous data between the Host and Controller. The format of the synchronous Data Packet is shown in Figure 5.3. The definition for each of the fields in the data packets is explained below.

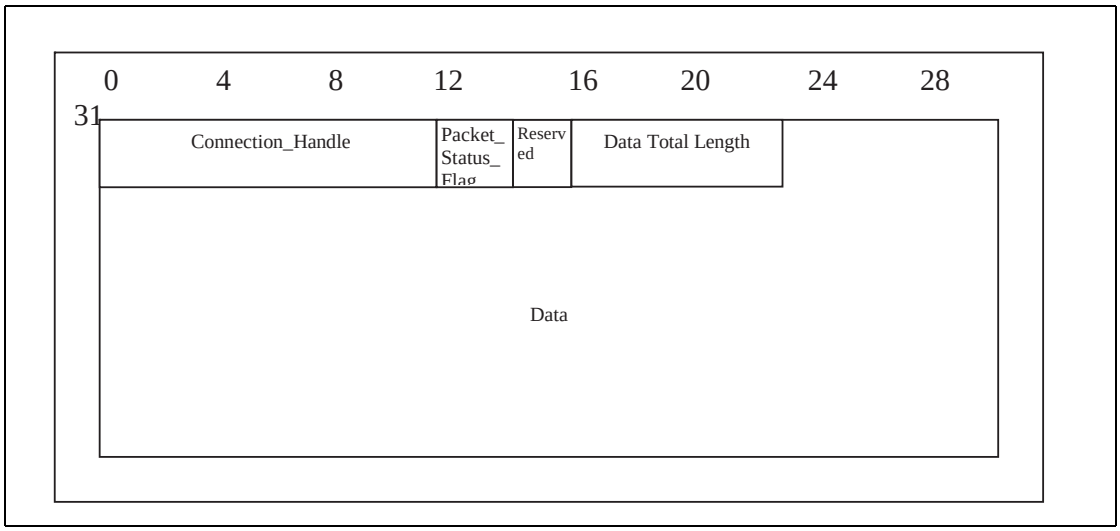


Figure 5.3: HCI Synchronous Data Packet

Connection_Handle: *Size: 12 Bits*

Value	Parameter Description
0xXXX	Connection_Handle to be used to for transmitting a synchronous data packet or segment. Range: 0x0000-0x0EFF (0x0F00- 0x0FFF Reserved for future use)

The Packet_Status_Flag bits consist of two bits, which are located from bit 4 to 5 in the second octet of the HCI Synchronous Data packet.

The Host shall set the Packet_Status_Flag bits to 00.

If the Erroneous_Data_Reporting parameter was set to disabled when the synchronous connection was created, the Controller shall set the Packet_Status_Flag bits to 00 and whether or not data is provided for cases when a valid (e)SCO packet was not received is unspecified.

If the Erroneous_Data_Reporting parameter was set to enabled when the synchronous connection was created, the controller shall set the Packet_Status_Flag according to the following table.

Packet_Status_Flag (in packets sent by the Controller) *Size: 2 Bits*

Value	Parameter Description
00	Correctly received data. The payload data belongs to received eSCO or SCO packets that the baseband marked as “good data”.
01	Possibly invalid data. At least one eSCO packet has been marked by the baseband as “data with possible errors” and all others have been marked as “good data” in the eSCO interval(s) corresponding to the HCI Synchronous Data Packet.
10	No data received. All data from the baseband received during the (e)SCO interval(s) corresponding to the HCI Synchronous Data Packet have been marked as "lost data" by the baseband. The Payload data octets shall be set to 0.
11	Data partially lost. Not all, but at least one (e)SCO packet has been marked as “lost data” by the baseband in the (e)SCO intervals corresponding to the HCI Synchronous Data Packet. The payload data octets corresponding to the missing (e)SCO packets shall be set to 0.

Note: Some HCI transports and/or controller implementations will align the HCI Synchronous Data Packets with the (e)SCO baseband packets such that data integrity can be explicitly marked in the Packet_Status_Flag. For HCI transports or Controller implementations that do not preserve this alignment, information in the Packet_Status_Flag may be ambiguous.

The Reserved Bits consist of two bits which are located from bit 6 to bit 7 in the second octet of the HCI Synchronous Data packet.

Reserved: *Size: 2 Bits*

Value	Parameter Description
XXXX	Reserved for future use.

Data_Total_Length: *Size: 1 Octet*

Value	Parameter Description
0xXX	Length of synchronous data measured in octets

5.4.4 HCI Event Packet

The HCI Event Packet is used by the Controller to notify the Host when events occur. The Host must be able to accept HCI Event Packets with up to 255 octets of data excluding the HCI Event Packet header. The format of the HCI Event Packet is shown in Figure 5.4, and the definition of each field is explained in Table 7.1.

An LE Controller uses sub-event codes to transmit LE specific events from the Controller to the Host. Note: The sub-event code will always be the first Event Parameter (See Section 7.7.65).

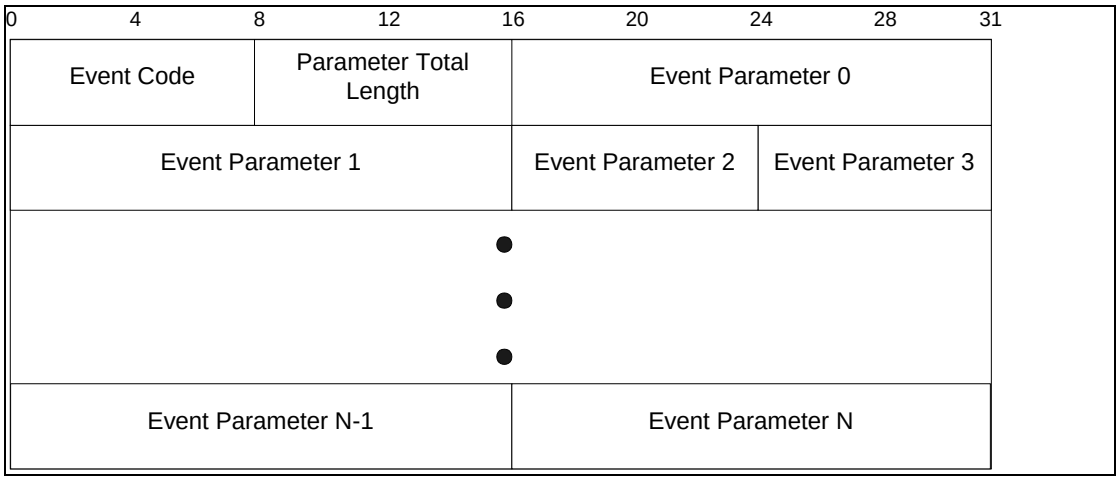


Figure 5.4: HCI Event Packet

Event_Code: *Size: 1 Octet*

Value	Parameter Description
0xXX	Each event is assigned a 1-Octet event code used to uniquely identify different types of events. Range: 0x00-0xFF (The event code 0xFF is reserved for the event code used for vendor-specific debug events.)

Parameter_Total_Length: *Size: 1 Octet*

Value	Parameter Description
0xXX	Length of all of the parameters contained in this packet, measured in octets

Event_Parameter 0 - N: *Size: Parameter Total Length*

Value	Parameter Description
0xXX	Each event has a specific number of parameters associated with it. These parameters and the size of each of the parameters are defined for each event. Each parameter is an integer number of octets in size.

6 HCI CONFIGURATION PARAMETERS

6.1 SCAN ENABLE

The Scan_Enable parameter controls whether or not the BR/EDR Controller will periodically scan for page attempts and/or inquiry requests from other BR/EDR Controllers. If Page_Scan is enabled, then the device will enter page scan mode based on the value of the Page_Scan_Interval and Page_Scan_Window parameters. If Inquiry_Scan is enabled, then the BR/EDR Controller will enter Inquiry Scan mode based on the value of the Inquiry_Scan_Interval and Inquiry_Scan_Window parameters.

Value	Parameter Description
0x00	No Scans enabled.
0x01	Inquiry Scan enabled. Page Scan always disabled.
0x02	Inquiry Scan disabled. Page Scan enabled.
0x03	Inquiry Scan enabled. Page Scan enabled.
0x04-0xFF	Reserved [0x04 limited page scan removed]

6.2 INQUIRY SCAN INTERVAL

The Inquiry_Scan_Interval configuration parameter defines the amount of time between consecutive inquiry scans. This is defined as the time interval from when the BR/EDR Controller started its last inquiry scan until it begins the next inquiry scan.

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0012 to 0x1000; only even values are valid Default: 0x1000 Mandatory Range: 0x0012 to 0x1000 Time = N * 0.625 msec Time Range: 11.25 to 2560 msec Time Default: 2.56 sec

6.3 INQUIRY SCAN WINDOW

The Inquiry_Scan_Window configuration parameter defines the amount of time for the duration of the inquiry scan. The Inquiry_Scan_Window can only be less than or equal to the Inquiry_Scan_Interval.

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0011 to 0x1000 Default: 0x0012 Mandatory Range: 0x0011 to Inquiry Scan Interval Time = N * 0.625 msec Time Range: 10.625 msec to 2560 msec Time Default: 11.25 msec

6.4 INQUIRY SCAN TYPE

The Inquiry_Scan_Type configuration parameter indicates whether inquiry scanning will be done using non-interlaced scan or interlaced scan. Currently, one mandatory inquiry scan type and one optional inquiry scan type are defined. For details, see the Baseband Specification, Part B, Section 8.4.1, Inquiry scan substate.

Value	Parameter Description
0x00	Mandatory: Standard Scan (default)
0x01	Optional: Interlaced Scan
0x02-0xFF	Reserved

6.5 INQUIRY MODE

The Inquiry_Mode configuration parameter indicates whether inquiry returns Inquiry Result events in the standard format, with RSSI, or with RSSI and extended inquiry response information.

Value	Parameter Description
0x00	Standard Inquiry Result event format
0x01	Inquiry Result format with RSSI
0x02	Inquiry Result with RSSI format or Extended Inquiry Result format
0x03-0xFF	Reserved

6.6 PAGE TIMEOUT

The Page_Timeout configuration parameter defines the maximum time the local Link Manager will wait for a baseband page response from the remote device at a locally initiated connection attempt. If this time expires and the remote BR/EDR Controller has not responded to the page at baseband level, the connection attempt will be considered to have failed.

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0001 to 0xFFFF Default: 0x2000 Mandatory Range: 0x0016 to 0xFFFF Time = N * 0.625 msec Time Range: 0.625 msec to 40.9 sec Time Default: 5.12 sec

6.7 CONNECTION ACCEPT TIMEOUT

The Connection_Accept_Timeout configuration parameter allows the BR/EDR Controller to automatically deny a connection request after a specified time period has occurred and the new connection is not accepted. The parameter defines the time duration from when the BR/EDR Controller sends a Connection Request event until the BR/EDR Controller will automatically reject an incoming connection.

The Connection_Accept_Timeout configuration parameter allows each AMP Controller to limit the duration of the Create_Physical_Link command or Accept_Physical_Link command it is locally serving. The parameter defines the maximum time duration for all physical connection setup activities that occur between the Create_Physical_Link command or Accept_Physical_Link command and the corresponding Physical Link Compete event (including all authentication activity) before the AMP Controller will automatically reject the physical link setup.

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0001 to 0xB540 Default: 0x1F40 Mandatory Range: 0x00A0 to 0xB540 Time = N * 0.625 msec Time Range: 0.625 msec to 29 sec Time Default: BR/EDR 5 sec AMP type dependent

6.8 PAGE SCAN INTERVAL

The Page_Scan_Interval configuration parameter defines the amount of time between consecutive page scans. This time interval is defined from when the Controller started its last page scan until it begins the next page scan.

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0012 to 0x1000; only even values are valid Default: 0x0800 Mandatory Range: 0x0012 to 0x1000 Time = N * 0.625 msec Time Range: 11.25 msec to 2560 msec Time Default: 1.28 sec

6.9 PAGE SCAN WINDOW

The Page_Scan_Window configuration parameter defines the amount of time for the duration of the page scan. The Page_Scan_Window can only be less than or equal to the Page_Scan_Interval.

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0011 to 0x1000 Default: 0x0012 Mandatory Range: 0x0011 to Page Scan Interval Time = N * 0.625 msec Time Range: 10.625 msec to Page Scan Interval Time Default: 11.25 msec

6.10 PAGE SCAN PERIOD MODE (DEPRECATED)

Every time an inquiry response message is sent, the BR/EDR Controller will start a timer (T_mandatory_pscan), the value of which is dependent on the Page_Scan_Period_Mode. As long as this timer has not expired, the BR/EDR Controller will use the mandatory page scan mode for all following page scans.

Note: The timer T_mandatory_pscan will be reset at each new inquiry response. For details see Part B, Baseband Specification.

Value	Parameter Description
0x00	P0
0x01	P1
0x02	P2

0x03-0xFF	Reserved.

6.11 PAGE SCAN TYPE

The Page_Scan_Type parameter indicates whether inquiry scanning will be done using non-interlaced scan or interlaced scan. For details, see the Baseband Specification, Baseband Specification, Section 8.3.1, on page 151.

Value	Parameter Description
0x00	Mandatory: Standard Scan (default)
0x01	Optional: Interlaced Scan
0x02-0xFF	Reserved

6.12 VOICE SETTING

The Voice_Setting parameter controls all the various settings for voice connections. The input settings apply to all voice connections, and **cannot** be set for individual voice connections. The Voice_Setting parameter controls the configuration for voice connections: Input Coding, Air coding format, input data format, Input sample size, and linear PCM parameter. The air coding format bits in the Voice_Setting command parameter specify which air coding format the local device requests. The air coding format bits do not specify which air coding format(s) the local device accepts when a remote device requests an air coding format. This is determined by the hardware capabilities of the local BR/EDR Controller.

Value	Parameter Description
00XXXXXXXX	Input Coding: Linear
01XXXXXXXX	Input Coding: μ-law Input Coding
10XXXXXXXX	Input Coding: A-law Input Coding
11XXXXXXXX	Reserved for future use
XX00XXXXXX	Input Data Format: 1's complement
XX01XXXXXX	Input Data Format: 2's complement
XX10XXXXXX	Input Data Format: Sign-Magnitude
XX11XXXXXX	Input Data Format: Unsigned
XXXX0XXXXX	Input Sample Size: 8-bit (only for linear PCM)
XXXX1XXXXX	Input Sample Size: 16-bit (only for linear PCM)
XXXXXnnnXX	Linear_PCM_Bit_Pos: # bit positions that MSB of sample is away from starting at MSB (only for Linear PCM).
XXXXXXXX00	Air Coding Format: CVSD

Value	Parameter Description
XXXXXXXX01	Air Coding Format: μ-law
XXXXXXXX10	Air Coding Format: A-law
XXXXXXXX11	Air Coding Format: Transparent Data

6.13 PIN TYPE

The PIN Type configuration parameter determines whether the Link Manager assumes that the Host supports variable PIN codes or a fixed PIN code. The host controller uses the PIN-type information during pairing.

Value	Parameter Description
0x00	Variable PIN.
0x01	Fixed PIN.

6.14 LINK KEY

The Controller can store a limited number of link keys for other BR/EDR Controllers. Link keys are shared between two BR/EDR Controllers, and are used for all security transactions between the two BR/EDR Controllers. A Host device may have additional storage capabilities, which can be used to save additional link keys to be reloaded to the BR/EDR Controller when needed. A Link Key is associated with a BD_ADDR.

Value	Parameter Description
0xXXXXXXXXXX XXXXXXXXXX XXXXXXXXXX	Link Key for an associated BD_ADDR.

6.15 FAILED CONTACT COUNTER

The Failed_Contact_Counter records the number of consecutive incidents in which either the local or remote device did not respond after the flush timeout had expired, and the L2CAP PDU that was currently being transmitted was automatically ‘flushed’. When this occurs, the Failed_Contact_Counter is incremented by 1.

The Failed Contact Counter is maintained for each Connection_Handle and Logical_Link_Handle.

The Failed_Contact_Counter for a connection is reset to zero on the following conditions:

1. When a new connection is established

2. When the Failed_Contact_Counter is > zero and an L2CAP PDU is acknowledged for that connection
3. When the Reset_Failed_Contact_Counter command has been issued.

Value	Parameter Description
0xXXXX	Number of consecutive failed contacts for a connection corresponding to the Connection or Logical Link Handle.

6.16 AUTHENTICATION ENABLE

The Authentication_Enable parameter controls if the local device requires to authenticate the remote device at connection setup (between the Create_Connection command or acceptance of an incoming ACL connection and the corresponding Connection Complete event). At connection setup, only the device(s) with the Authentication_Enable parameter enabled will try to authenticate the other device.

Note: Changing this parameter does not affect existing connections.

Value	Parameter Description
0x00	Authentication not required.
0x01	Authentication required for all connections.
0x02-0xFF	Reserved

6.17 HOLD MODE ACTIVITY

The Hold_Mode_Activity value is used to determine what activities should be suspended when the BR/EDR Controller is in hold mode. After the hold period has expired, the device will return to the previous mode of operation. Multiple hold mode activities may be specified for the Hold_Mode_Activity parameter by performing a bitwise OR operation of the different activity types. If no activities are suspended, then all of the current Periodic Inquiry, Inquiry Scan, and Page Scan settings remain valid during the Hold Mode. If the Hold_Mode_Activity parameter is set to Suspend Page Scan, Suspend Inquiry Scan, and Suspend Periodic Inquiries, then the device can enter a low-power state during the Hold Mode period, and all activities are suspended. Suspending multiple activities can be specified for the Hold_Mode_Activity parameter by performing a bitwise OR operation of the different activity types. The Hold Mode Activity is only valid if all connections are in Hold Mode.

Value	Parameter Description
0x00	Maintain current Power State.
0x01	Suspend Page Scan.

0x02	Suspend Inquiry Scan.
0x04	Suspend Periodic Inquiries.
0x08-0xFF	Reserved for future use.

6.18 LINK POLICY SETTINGS

The Link_Policy_Settings parameter determines the behavior of the local Link Manager when it receives a request from a remote Link Manager or it determines itself to change the master-slave role or to enter park state, hold, or sniff mode. The local Link Manager will automatically accept or reject such a request from the remote device, and may even autonomously request itself, depending on the value of the Link_Policy_Settings parameter for the corresponding Connection_Handle. When the value of the Link_Policy_Settings parameter is changed for a certain Connection_Handle, the new value will only be used for requests from a remote device or from the local Link Manager itself made after this command has been completed. By enabling each mode individually, the Host can choose any combination needed to support various modes of operation. Multiple LM policies may be specified for the Link_Policy_Settings parameter by performing a bitwise OR operation of the different activity types.

Note: The local BR/EDR Controller may be forced into hold mode (regardless of whether the local device is master or slave) by the remote device regardless of the value of the Link_Policy_Settings parameter. The forcing of hold mode can however only be done once the connection has already been placed into hold mode through an LMP request (the Link_Policy_Settings determine if requests from a remote device should be accepted or rejected). The forcing of hold mode can after that be done as long as the connection lasts regardless of the setting for hold mode in the Link_Policy_Settings parameter.

Note that the previous description implies that if the implementation in the remote device is a “polite” implementation that does not force another device into hold mode via LMP PDUs, then the Link_Policy_Settings will never be overruled.

Value	Parameter Description
0x0000	Disable All LM Modes Default
0x0001	Enable Role switch.
0x0002	Enable Hold Mode.
0x0004	Enable Sniff Mode.
0x0008	Enable Park State.
0x0008 – 0x8000	Reserved for future use.

6.19 FLUSH TIMEOUT

The Flush_Timeout configuration parameter is used for ACL connections on a BR/EDR Controller only. The Flush Timeout is defined in the Baseband specification Baseband Specification, Section 7.6.3, on page 147. This parameter allows automatically-flushable ACL packets to be automatically flushed without the Host device issuing a Flush command. This provides support for isochronous data, such as audio. Non-automatically-flushable ACL packets shall not be affected by this parameter (see HCI ACL Data Packets) providing support for both asynchronous and isochronous data on the same ACL connection. When the L2CAP packet that is currently being transmitted is automatically 'flushed', the Failed Contact Counter is incremented by one.

Note: Since for AMP Controllers a Flush Timeout value is maintained as part of the Extended Flow Specification for each logical link, the separate Flush Timeout configuration parameter is not used.

Value	Parameter Description
0	Timeout = ∞; No Automatic Flush
N = 0xXXXX	Size: 2 Octets Range: 0x0001 to 0x07FF Mandatory Range: 0x0002 to 0x07FF Time = N * 0.625 msec Time Range: 0.625 msec to 1279.375 msec

6.20 NUM BROADCAST RETRANSMISSIONS

Broadcast packets are not acknowledged and are unreliable. The Number of Broadcast Retransmissions parameter, N, is used to increase the reliability of a broadcast message by retransmitting the broadcast message multiple times. This sets the value N_{BC} in the baseband to one greater than the Num Broadcast Retransmissions value. (See Baseband Specification, Section 7.6.5, on page 148) This parameter should be adjusted as the link quality measurement changes.

Value	Parameter Description
N = 0xXX	N_{BC} = N + 1 Range 0x00-0xFE

6.21 LINK SUPERVISION TIMEOUT

The Link_Supervision_Timeout parameter is used by the Controller to monitor link loss. If, for any reason, no packets are received from that Handle (a Connection_Handle for the BR/EDR Controller or the Physical_Link_Handle for an AMP Controller) for a duration longer than the Link_Supervision_Timeout, the connection is disconnected. For the BR/EDR Controller, the same timeout value is used for both synchronous and ACL connections for the device specified by the Connection_Handle. For an AMP Controller, the timeout value is maintained for each physical link. Note: Some AMP Controllers may also have an early notification event (See Physical Link Loss Early Warning Event).

Note: Setting the Link_Supervision_Timeout to No Link_Supervision_Timeout (0x0000) on a BR/EDR Controller will disable the Link_Supervision_Timeout check for the specified Connection_Handle. This makes it unnecessary for the master of the piconet to unpark and then park each BR/EDR Controller every ~40 seconds. By using the No Link_Supervision_Timeout setting, the scalability of the Park state is not limited.

Value	Parameter Description
0x0000	No Link_Supervision_Timeout. Shall not be used with AMP Controllers.
N = 0xXXXX	Size: 2 Octets Range: 0x0001 to 0xFFFF Default: 0x7D00 Mandatory Range: 0x0190 to 0xFFFF Time = N * 0.625 msec Time Range: 0.625 msec to 40.9 sec Time Default: BR/EDR 20 sec AMP 10 sec

6.22 SYNCHRONOUS FLOW CONTROL ENABLE

The Synchronous Flow Control Enable configuration parameter allows the Host to decide if the BR/EDR Controller will send Number Of Completed Packets events for synchronous Connection_Handles. This setting allows the Host to enable and disable synchronous flow control.

Value	Parameter Description
0x00	Synchronous Flow Control is disabled. No Number of Completed Packets events will be sent from the BR/EDR Controller for synchronous Connection_Handles.
0x01	Synchronous Flow Control is enabled. Number of Completed Packets events will be sent from the BR/EDR Controller for synchronous Connection_Handles.

6.23 LOCAL NAME

The user-friendly Local Name provides the user the ability to distinguish one BR/EDR Controller from another. The Local Name configuration parameter is a UTF-8 encoded string with up to 248 octets in length. The Local Name configuration parameter will be null terminated (0x00) if the UTF-8 encoded string is less than 248 octets.

Note: The Local Name configuration parameter is a string parameter. Endianess does therefore not apply to the Local Name configuration parameter. The first octet of the name is received first.

Value	Parameter Description
	A UTF-8 encoded User Friendly Descriptive Name for the device. If the name contained in the parameter is shorter than 248 octets, the end of the name is indicated by a NULL octet (0x00), and the following octets (to fill up 248 octets, which is the length of the parameter) do not have valid values.

6.24 EXTENDED INQUIRY RESPONSE

The Extended Inquiry Response provides information about the local device in response to inquiry from remote devices. The configuration parameter has two parts, a significant part followed by a non-significant part. The non-significant part contains only zero octets. The length of the extended inquiry response configuration parameter is 240 octets. The data format of the significant part is defined in [Vol 3] Part C, Section 8, Extended Inquiry Response Data Format.

Value	Parameter Description
	Information about the local device that will be sent in an extended inquiry response packet to remote devices during inquiry response.

6.25 ERRONEOUS DATA REPORTING

The Erroneous_Data_Reporting configuration parameter shall be used for SCO and eSCO connections only. This parameter determines if the BR/EDR Controller is required to provide data to the Host for every (e)SCO interval, with the Packet_Status_Flag in HCI Synchronous Data Packets set according to section 5.4.3, “HCI Synchronous Data Packets,” on page 431.

Value	Parameter Description
0x00	Erroneous data reporting disabled
0x01	Erroneous data reporting enabled.
0x02 - 0xFF	Reserved

6.26 CLASS OF DEVICE

The Class_of_Device parameter is used to indicate the capabilities of the local device to other devices.

Value	Parameter Description
0xXXXXXX	Class of Device for the device.

6.27 SUPPORTED COMMANDS

The Supported Commands configuration parameter lists which HCI commands the local controller supports. It is implied that if a command is listed as supported, the feature underlying that command is also supported.

The Supported Commands is a 64 octet bit field. If a bit is set to 1, then this command is supported.

Octet	Bit	Command Supported
0	0	Inquiry
	1	Inquiry Cancel
	2	Periodic Inquiry Mode
	3	Exit Periodic Inquiry Mode
	4	Create Connection
	5	Disconnect
	6	Add SCO Connection (deprecated)
	7	Cancel Create Connection
1	0	Accept Connection Request
	1	Reject Connection Request
	2	Link Key Request Reply
	3	Link Key Request Negative Reply
	4	PIN Code Request Reply
	5	PIN Code Request Negative Reply
	6	Change Connection Packet Type
	7	Authentication Request

Octet	Bit	Command Supported
2	0	Set Connection Encryption
	1	Change Connection Link Key
	2	Master Link Key
	3	Remote Name Request
	4	Cancel Remote Name Request
	5	Read Remote Supported Features
	6	Read Remote Extended Features
	7	Read Remote Version Information
3	0	Read Clock Offset
	1	Read LMP Handle
	2	Reserved
	3	Reserved
	4	Reserved
	5	Reserved
	6	Reserved
	7	Reserved
4	0	Reserved
	1	Hold Mode
	2	Sniff Mode
	3	Exit Sniff Mode
	4	Park State
	5	Exit Park State
	6	QoS Setup
	7	Role Discovery
5	0	Switch Role
	1	Read Link Policy Settings
	2	Write Link Policy Settings
	3	Read Default Link Policy Settings
	4	Write Default Link Policy Settings
	5	Flow Specification
	6	Set Event Mark
	7	Reset

Octet	Bit	Command Supported
6	0	Set Event Filter
	1	Flush
	2	Read PIN Type
	3	Write PIN Type
	4	Create New Unit Key
	5	Read Stored Link Key
	6	Write Stored Link Key
	7	Delete Stored Link Key
7	0	Write Local Name
	1	Read Local Name
	2	Read Connection Accept Timeout
	3	Write Connection Accept Timeout
	4	Read Page Timeout
	5	Write Page Timeout
	6	Read Scan Enable
	7	Write Scan Enable
8	0	Read Page Scan Activity
	1	Write Page Scan Activity
	2	Read Inquiry Scan Activity
	3	Write Inquiry Scan Activity
	4	Reserved
	5	Reserved
	6	Read Encryption Mode (deprecated)
	7	Write Encryption Mode (deprecated)
9	0	Read Class Of Device
	1	Write Class Of Device
	2	Read Voice Setting
	3	Write Voice Setting
	4	Read Automatic Flush Timeout
	5	Write Automatic Flush Timeout
	6	Read Num Broadcast Retransmissions
	7	Write Num Broadcast Retransmissions

Octet	Bit	Command Supported
10	0	Read Hold Mode Activity
	1	Write Hold Mode Activity
	2	Read Transmit Power Level
	3	Read Synchronous Flow Control Enable
	4	Write Synchronous Flow Control Enable
	5	Set Host Controller To Host Flow Control
	6	Host Buffer Size
	7	Host Number Of Completed Packets
11	0	Read Link Supervision Timeout
	1	Write Link Supervision Timeout
	2	Read Number of Supported IAC
	3	Read Current IAC LAP
	4	Write Current IAC LAP
	5	Read Page Scan Mode Period (deprecated)
	6	Write Page Scan Mode Period (deprecated)
	7	Read Page Scan Mode (deprecated)
12	0	Write Page Scan Mode (deprecated)
	1	Set AFH Channel Classification
	2	Reserved
	3	Reserved
	4	Read Inquiry Scan Type
	5	Write Inquiry Scan Type
	6	Read Inquiry Mode
	7	Write Inquiry Mode
13	0	Read Page Scan Type
	1	Write Page Scan Type
	2	Read AFH Channel Assessment Mode
	3	Write AFH Channel Assessment Mode
	4	Reserved
	5	Reserved
	6	Reserved
	7	Reserved

Octet	Bit	Command Supported
14	0	Reserved
	1	Reserved
	2	Reserved
	3	Read Local Version Information
	4	Reserved
	5	Read Local Supported Features
	6	Read Local Extended Features
	7	Read Buffer Size
15	0	Read Country Code [Deprecated]
	1	Read BD ADDR
	2	Read Failed Contact Count
	3	Reset Failed Contact Count
	4	Get Link Quality
	5	Read RSSI
	6	Read AFH Channel Map
	7	Read BD Clock
16	0	Read Loopback Mode
	1	Write Loopback Mode
	2	Enable Device Under Test Mode
	3	Setup Synchronous Connection
	4	Accept Synchronous Connection
	5	Reject Synchronous Connection
	6	Reserved
	7	Reserved
17	0	Read Extended Inquiry Response
	1	Write Extended Inquiry Response
	2	Refresh Encryption Key
	3	Reserved
	4	Sniff Subrating
	5	Read Simple Pairing Mode
	6	Write Simple Pairing Mode
	7	Read Local OOB Data

Octet	Bit	Command Supported
18	0	Read Inquiry Response Transmit Power
	1	Write Inquiry Transmit Power Level
	2	Read Default Erroneous Data Reporting
	3	Write Default Erroneous Data Reporting
	4	Reserved
	5	Reserved
	6	Reserved
	7	IO Capability Request Reply
19	0	User Confirmation Request Reply
	1	User Confirmation Request Negative Reply
	2	User Passkey Request Reply
	3	User Passkey Request Negative Reply
	4	Remote OOB Data Request Reply
	5	Write Simple Pairing Debug Mode
	6	Enhanced Flush
	7	Remote OOB Data Request Negative Reply
20	0	Reserved
	1	Reserved
	2	Send Keypress Notification
	3	IO Capability Request Negative Reply
	4	Read Encryption Key Size
	5	Reserved
	6	Reserved
	7	Reserved
21	0	Create Physical Link
	1	Accept Physical Link
	2	Disconnect Physical Link
	3	Create Logical Link
	4	Accept Logical Link
	5	Disconnect Logical Link
	6	Logical Link Cancel
	7	Flow Spec Modify

Octet	Bit	Command Supported
22	0	Read Logical Link Accept Timeout
	1	Write Logical Link Accept Timeout
	2	Set Event Mask Page 2
	3	Read Location Data
	4	Write Location Data
	5	Read Local AMP Info
	6	Read Local AMP_ASSOC
	7	Write Remote AMP_ASSOC
23	0	Read Flow Control Mode
	1	Write Flow Control Mode
	2	Read Data Block Size
	3	Reserved
	4	Reserved
	5	Enable AMP Receiver Reports
	6	AMP Test End
	7	AMP Test Command
24	0	Read Enhanced Transmit Power Level
	1	Reserved
	2	Read Best Effort Flush Timeout
	3	Write Best Effort Flush Timeout
	4	Short Range Mode
	5	Read LE Host Support
	6	Write LE Host Support
	7	Reserved
25	0	LE Set Event Mask
	1	LE Read Buffer Size
	2	LE Read Local Supported Features
	3	Reserved
	4	LE Set Random Address
	5	LE Set Advertising Parameters
	6	LE Read Advertising Channel TX Power
	7	LE Set Advertising Data

Octet	Bit	Command Supported
26	0	LE Set Scan Response Data
	1	LE Set Advertise Enable
	2	LE Set Scan Parameters
	3	LE Set Scan Enable
	4	LE Create Connection
	5	LE Create Connection Cancel
	6	LE Read White List Size
	7	LE Clear White List
27	0	LE Add Device To White List
	1	LE Remove Device From White List
	2	LE Connection Update
	3	LE Set Host Channel Classification
	4	LE Read Channel Map
	5	LE Read Remote Used Features
	6	LE Encrypt
	7	LE Rand
28	0	LE Start Encryption
	1	LE Long Term Key Request Reply
	2	LE Long Term Key Requested Negative Reply
	3	LE Read Supported States
	4	LE Receiver Test
	5	LE Transmitter Test
	6	LE Test End
	7	Reserved

6.28 LOGICAL LINK ACCEPT TIMEOUT

The Logical_Link_Accept_Timeout configuration parameter allows the AMP Controller to automatically deny a logical link request after a specified time period has elapsed and the new logical link has not completed. The parameter defines the timeout duration, which includes all logical link setup activities that occur between the Create_Logical_Link or Accept_Logical_Link commands and the Logical Link Compete event. If these activities are not completed within the specified time, the AMP Controller will automatically reject the logical link set up.

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0001 to 0xB540 Default: 0x1F40 Mandatory Range: 0x00A0 to 0xB540 Time = N * 0.625 msec Time Range: 0.625 msec to 29 sec Time Default: AMP Type dependent

6.29 LOCATION DOMAIN AWARE

The Location_Domain_Aware parameter indicates if the transmitter has information about the current regulatory domain.

Location_Domain_Aware *Size 1 Octet*

Value	Parameter Description
0x00	Regulatory domain unknown
0x01	Regulatory domain known
0x02-0xFF	Reserved for future use

6.30 LOCATION DOMAIN

The Location_Domain parameter specifies the country or other location code if known.

Location_Domain *Size 1 Octet*

Value	Parameter Description
0xXXXX	Shall be set to "XX" (0x5858) for a non-country entity or when Location_Domain_Aware is set to regulatory domain unknown. When the regulatory domain is known, it shall be set according to the two Octet country code as defined in ISO 3166-1.[1]

1. ISO 3166-1-alpha-2 code elements (English), http://www.iso.org/iso/english_country_names_and_code_elements

6.31 LOCATION DOMAIN OPTIONS

The Location_Domain_Options parameter may specify additional information specific to that country, or that the location is not a single country.

Location_Domain_Options *Size: 1 Octet*

Values	Parameter Descriptions
	Single octet suffix:
0x20	"space" indicates that the code applies to the entire country
0x4F	"O" indicates for use outdoors only
0x49	"I" indicates for use indoors only
0x58	"X" indicates a non-country entity
Other values	Reserved for future use

The format of the Location_Domain_Options parameter is derived from IEEE 802.11-2007 Annex D "ASN.1 encoding of the MAC and PHY MIB."

6.32 LOCATION OPTIONS

The Location_Options parameter indicates if the transmitter has information about any options that may affect operation in different locations.

Location_Options: *Size: 1 Octet*

Bit	Value	Parameter Description
0	0	Not mains-powered
0	1	Mains powered
1-7		Reserved for future use

6.33 FLOW CONTROL MODE

The Flow_Control_Mode configuration parameter allows the Host to select the HCI Data flow control mode used by the Controller for ACL Data traffic.

Flow_Control_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Packet based data flow control mode (**default** for a Primary Controller)
0x01	Data block based data flow control mode (**default** for an AMP Controller)
0x02-0xFF	Reserved for future use

6.34 LE SUPPORTED HOST

The LE_Supported_Host parameter allows the Host to read and set the Link Manager Protocol feature bit LE Supported (Host). See Part C, Section 3.2, Feature Definitions.

Value	Parameter Description
0x00	LE Supported (Host) disabled (**default**)
0x01	LE Supported (Host) enabled
0x02 – 0xFF	Reserved

6.35 SIMULTANEOUS LE HOST

The Simultaneous_LE_Host parameter allows the Host to read and set the Link Manager Protocol feature bit Simultaneous LE and BR/EDR to Same Device Capable (Host). See Part C, Section 3.2, Feature Definitions.

Value	Parameter Description
0x00	Simultaneous LE and BR/EDR to Same Device Capable (Host) disabled (**default**)
0x01	Simultaneous LE and BR/EDR to Same Device Capable (Host) enabled
0x02 – 0xFF	Reserved

7 HCI COMMANDS AND EVENTS

7.1 LINK CONTROL COMMANDS

The Link Control commands allow a Controller to control connections to other BR/EDR Controllers. Some Link Control commands are used only with a BR/EDR Controller whereas other Link Control commands are used only with an AMP Controller.

In the BR/EDR Controller, when the Link Control commands are used, the Link Manager (LM) controls how the Bluetooth piconets and scatternets are established and maintained. These commands instruct the LM to create and modify link layer connections with Bluetooth remote devices, perform Inquiries of other BR/EDR Controllers in range, and other LMP commands.

In the AMP Controller, Link Control commands are used to create, modify and disconnect physical and logical links.

In the LE Controller, Link Control commands are used to disconnect physical links.

For the Link Control commands, the OGF is defined as 0x01.

7.1.1 Inquiry Command

Command	OCF	Command Parameters	Return Parameters
HCI_Inquiry	0x0001	LAP, Inquiry_Length, Num_Responses	

Description:

This command causes the BR/EDR Controller to enter Inquiry Mode. Inquiry Mode is used to discover other nearby BR/EDR Controllers. The LAP input parameter contains the LAP from which the inquiry access code shall be derived when the inquiry procedure is made. The Inquiry_Length parameter specifies the total duration of the Inquiry Mode and, when this time expires, Inquiry will be halted. The Num_Responses parameter specifies the number of responses that can be received before the Inquiry is halted. The event parameters of Inquiry Complete event will have a summary of the result from the Inquiry process, which reports the number of nearby BR/EDR Controllers that responded.

A device which responds during an inquiry or inquiry period should always be reported to the Host in an Inquiry Result event if the device has not been reported earlier during the current inquiry or inquiry period and the device has not been filtered out using the command Set_Event_Filter. If the device has been reported earlier during the current inquiry or inquiry period, it may or may not be reported depending on the implementation (depending on if earlier results have been saved in the BR/EDR Controller and in that case how many responses that have been saved). It is recommended that the BR/EDR Controller tries to report a particular device only once during an inquiry or inquiry period.

Command Parameters:

LAP: *Size: 3 Octets*

Value	Parameter Description
0x9E8B00–0X9E8B3F	This is the LAP from which the inquiry access code should be derived when the inquiry procedure is made; see Bluetooth Assigned Numbers.

Inquiry_Length: *Size: 1 Octet*

Value	Parameter Description
N = 0xXX	Maximum amount of time specified before the Inquiry is halted. Size: 1 octet Range: 0x01 – 0x30 Time = N * 1.28 sec Range: 1.28 – 61.44 Sec

Num_Responses: *Size: 1 Octet*

Value	Parameter Description
0x00	Unlimited number of responses.
0xXX	Maximum number of responses from the Inquiry before the Inquiry is halted. Range: 0x01 – 0xFF

Return Parameters:

None.

Event(s) generated (unless masked away):

A Command Status event shall be sent from the BR/EDR Controller to the Host when the BR/EDR Controller has started the Inquiry process. Unless filtered, an Inquiry Result event shall be created for each BR/EDR Controller which responds to the Inquiry message. In addition, multiple BR/EDR Controllers which respond to the Inquire message may be combined into the same event. An Inquiry Complete event shall be generated when the Inquiry process has completed.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Inquiry Complete event will indicate that this command has been completed. No Inquiry Complete event will be generated for the canceled Inquiry process.

7.1.2 Inquiry Cancel Command

Command	OCF	Command Parameters	Return Parameters
HCI_Inquiry_Cancel	0x0002		Status

Description:

This command shall cause the BR/EDR Controller to stop the current Inquiry if the BR/EDR Controller is in Inquiry Mode. This command allows the Host to interrupt the BR/EDR Controller and request the BR/EDR Controller to perform a different task. The command should only be issued after the Inquiry command has been issued, a Command Status event has been received for the Inquiry command, and before the Inquiry Complete event occurs.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Inquiry_Cancel command succeeded.
0x01-0xFF	Inquiry_Cancel command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Inquiry_Cancel command has completed, a Command Complete event shall be generated. No Inquiry Complete event will be generated for the canceled Inquiry process.

7.1.3 Periodic Inquiry Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Periodic_ Inquiry_Mode	0x0003	Max_Period_Length, Min_Period_Length, LAP, Inquiry_Length, Num_Responses	Status

Description:

The Periodic_Inquiry_Mode command is used to configure the BR/EDR Controller to enter the Periodic Inquiry Mode that performs an automatic Inquiry. Max_Period_Length and Min_Period_Length define the time range between two consecutive inquiries, from the beginning of an inquiry until the start of the next inquiry. The BR/EDR Controller shall use this range to determine a new random time between two consecutive inquiries for each Inquiry. The LAP input parameter contains the LAP from which the inquiry access code shall be derived when the inquiry procedure is made. The Inquiry_Length parameter specifies the total duration of the Inquiry Mode and, when time expires, Inquiry will be halted. The Num_Responses parameter specifies the number of responses that can be received before the Inquiry is halted. This command is completed when the Inquiry process has been started by the BR/EDR Controller, and a Command Complete event is sent from the Controller to the Host. When each of the periodic Inquiry processes are completed, the Controller will send an Inquiry Complete event to the Host indicating that the latest periodic Inquiry process has finished. When a BR/EDR Controller responds to the Inquiry message an Inquiry Result event will occur to notify the Host of the discovery.

Note: Max_Period_Length > Min_Period_Length > Inquiry_Length

A device which responds during an inquiry or inquiry period should always be reported to the Host in an Inquiry Result event if the device has not been reported earlier during the current inquiry or inquiry period and the device has not been filtered out using the command Set_Event_Filter. If the device has been reported earlier during the current inquiry or inquiry period, it may or may not be reported depending on the implementation (depending on if earlier results have been saved in the BR/EDR Controller and in that case how many responses that have been saved). It is recommended that the BR/EDR Controller tries to report a particular device only once during an inquiry or inquiry period.

Command Parameters:

Max_Period_Length: *Size: 2 Octets*

Value	Parameter Description

N = 0xXXXX	Maximum amount of time specified between consecutive inquiries. Size: 2 octets Range: 0x03 – 0xFFFF Time = N * 1.28 sec Range: 3.84 – 83884.8 Sec 0.0 – 23.3 hours

Min_Period_Length: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Minimum amount of time specified between consecutive inquiries. Size: 2 octets Range: 0x02 – 0xFFFE Time = N * 1.28 sec Range: 2.56 – 83883.52 Sec 0.0 – 23.3 hours

LAP: *Size: 3 Octets*

Value	Parameter Description
0x9E8B00– 0X9E8B3F	This is the LAP from which the inquiry access code should be derived when the inquiry procedure is made; see “Bluetooth Assigned Numbers”

Inquiry_Length: *Size: 1 Octet*

Value	Parameter Description
N = 0xXX	Maximum amount of time specified before the Inquiry is halted. Size: 1 octet Range: 0x01 – 0x30 Time = N * 1.28 sec Range: 1.28 – 61.44 Sec

Num_Responses: *Size: 1 Octet*

Value	Parameter Description
0x00	Unlimited number of responses.
0xXX	Maximum number of responses from the Inquiry before the Inquiry is halted. Range: 0x01 – 0xFF

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Periodic Inquiry Mode command succeeded.
0x01-0xFF	Periodic Inquiry Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

The Periodic Inquiry Mode begins when the BR/EDR Controller sends the Command Complete event for this command to the Host. Unless filtered, an Inquiry Result event shall be created for each remote device that have responded to the Inquiry message. In addition, multiple BR/EDR Controllers which response to the Inquiry message may be combined into the same event. An Inquiry Complete event shall be generated when each of the periodic Inquiry processes has completed. No Inquiry Complete event will be generated for the canceled Inquiry process.

7.1.4 Exit Periodic Inquiry Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Exit_Periodic_Inquiry_Mode	0x0004		Status

Description:

The Exit_Periodic_Inquiry_Mode command is used to end the Periodic Inquiry mode when the local device is in Periodic Inquiry Mode. If the BR/EDR Controller is currently in an Inquiry process, the Inquiry process shall be stopped directly and the BR/EDR Controller shall no longer perform periodic inquiries until the Periodic Inquiry Mode command is reissued.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Exit Periodic Inquiry Mode command succeeded.
0x01-0xFF	Exit Periodic Inquiry Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

A Command Complete event for this command shall occur when the local device is no longer in Periodic Inquiry Mode. No Inquiry Complete event will be generated for the canceled Inquiry process.

7.1.5 Create Connection Command

Command	OCF	Command Parameters	Return Parameters
HCI_Create_Connection	0x0005	BD_ADDR, Packet_Type, Page_Scan_Repetition_Mode, Reserved, Clock_Offset, Allow_Role_Switch	

Description:

This command causes the Link Manager to create a connection to the remote device with the BD_ADDR specified by the command parameters. This command causes the local BR/EDR Controller to begin the Page process to create a link level connection. The Link Manager will determine how the new ACL connection is established. This ACL connection is determined by the current state of the device, its piconet, and the state of the device to be connected. The Packet_Type command parameter specifies which packet types the Link Manager shall use for the ACL connection. When sending HCI ACL Data Packets the Link Manager shall only use the packet type(s) specified by the Packet_Type command parameter or the always-allowed DM1 packet type. Multiple packet types may be specified for the Packet Type parameter by performing a bit-wise OR operation of the different packet types. The Link Manager may choose which packet type to be used from the list of acceptable packet types. The Page_Scan_Repetition_Mode parameter specifies the page scan repetition mode supported by the remote device with the BD_ADDR. This is the information that was acquired during the inquiry process. The Clock_Offset parameter is the difference between its own clock and the clock of the remote device with BD_ADDR. Only bits 2 through 16 of the difference are used, and they are mapped to this parameter as bits 0 through 14 respectively. A Clock_Offset_Valid_Flag, located in bit 15 of the Clock_Offset parameter, is used to indicate if the Clock Offset is valid or not. A Connection_Handle for this connection is returned in the Connection Complete event (see below). The Allow_Role_Switch parameter specifies if the local device accepts or rejects the request of a master-slave role switch when the remote device requests it at the connection setup (in the Role parameter of the Accept_Connection_Request command) (before the local Controller returns a Connection Complete event). For a definition of the different packet types see the Part B, Baseband Specification on page 59.

Note: The Host should enable as many packet types as possible for the Link Manager to perform efficiently. However, the Host must not enable packet types that the local device does not support.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device to be connected.

Packet_Type: *Size: 2 Octets*

Value	Parameter Description
0x0001	Reserved for future use.
0x0002	2-DH1 may not be used.
0x0004	3-DH1 may not be used.
0x0008[1]	DM1 may be used.
0x0010	DH1 may be used.
0x0020	Reserved for future use.
0x0040	Reserved for future use.
0x0080	Reserved for future use.
0x0100	2-DH3 may not be used.
0x0200	3-DH3 may not be used.
0x0400	DM3 may be used.
0x0800	DH3 may be used.
0x1000	2-DH5 may not be used.
0x2000	3-DH5 may not be used.
0x4000	DM5 may be used.
0x8000	DH5 may be used.

1. This bit will be interpreted as set to 1 by Bluetooth V1.2 or later controllers.

Page_Scan_Repetition_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	R0
0x01	R1
0x02	R2
0x03 – 0xFF	Reserved.

Reserved: *Size: 1 Octet*

Value	Parameter Description
0x00	Reserved, must be set to 0x00. See Page Scan Mode on page 871.

Clock_Offset: *Size: 2 Octets*

Bit format	Parameter Description
Bit 14-0	Bit 16-2 of CLKslave-CLKmaster.
Bit 15	Clock_Offset_Valid_Flag Invalid Clock Offset = 0 Valid Clock Offset = 1

Allow_Role_Switch: *Size: 1 Octet*

Value	Parameter Description
0x00	The local device will be a master, and will not accept a role switch requested by the remote device at the connection setup.
0x01	The local device may be a master, or may become a slave after accepting a role switch requested by the remote device at the connection setup.
0x02-0xFF	Reserved for future use.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Create_Connection command, the BR/EDR Controller shall send the Command Status event to the Host. In addition, when the Link Manager determines the connection is established, the BR/EDR Controller, on both BR/EDR Controllers that form the connection, shall send a Connection Complete event to each Host. The Connection Complete event contains the Connection_Handle if this command is successful.

Note: No Command Complete event will be sent by the Controller to indicate that this command has been completed. Instead, the Connection Complete event will indicate that this command has been completed.

7.1.6 Disconnect Command

Command	OCF	Command Parameters	Return Parameters
HCI_Disconnect	0x0006	Connection_Handle, Reason	

Description:

The Disconnection command is used to terminate an existing connection. The Connection_Handle command parameter indicates which connection is to be disconnected. The Reason command parameter indicates the reason for ending the connection. The remote Controller will receive the Reason command parameter in the Disconnection Complete event. All synchronous connections on a physical link should be disconnected before the ACL connection on the same physical connection is disconnected.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle for the connection being disconnected. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Reason: *Size: 1 Octet*

Value	Parameter Description
0x05, 0x13-0x15, 0x1A, 0x29	Authentication Failure error code (0x05), Other End Terminated Connection error codes (0x13-0x15), Unsupported Remote Feature error code (0x1A) and Pairing with Unit Key Not Supported error code (0x29), see Part D, Error Codes on page 339 for a list of error codes and descriptions.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Controller receives the Disconnect command, it shall send the Command Status event to the Host. The Disconnection Complete event will occur at each Host when the termination of the connection has completed, and indicates that this command has been completed.

Note: No Command Complete event will be sent by the Controller to indicate that this command has been completed. Instead, the Disconnection Complete event will indicate that this command has been completed.

7.1.7 Create Connection Cancel Command

Command	OCF	Command Parameters	Return Parameters
HCI_Create_Connection_Cancel	0x0008	BD_ADDR	Status, BD_ADDR

Description:

This command is used to request cancellation of the ongoing connection creation process, which was started by a Create_Connection command of the local BR/EDR Controller.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Create Connection command request that was issued before and is subject of this cancellation request

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Create Connection Cancel command succeeded
0x01-0xff	Create Connection Cancel command failed. See Part D, Error Codes on page 339 for list of error codes

BD_ADDR: *Size: 6 Octet*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Create Connection command that was issued before and is the subject of this cancellation request.

Event(s) generated (unless masked away):

When the Create_Connection_Cancel command has completed, a Command Complete event shall be generated.

If the connection is already established by the baseband, but the BR/EDR Controller has not yet sent the Connection Complete event, then the local device shall detach the link and return a Command Complete event with the status “Success”.

If the connection is already established, and the Connection Complete event has been sent, then the Controller shall return a Command Complete event with the error code *ACL Connection already exists* (0x0B).

If the Create_Connection_Cancel command is sent to the Controller without a preceding Create Connection command to the same device, the BR/EDR Controller shall return a Command Complete event with the error code *Unknown Connection Identifier* (0x02).

The Connection Complete event for the corresponding Create_Connection command shall always be sent. The Connection Complete event shall be sent after the Command Complete event for the Create_Connection_Cancel command. If the cancellation was successful, the Connection Complete event will be generated with the error code *Unknown Connection Identifier* (0x02).

7.1.8 Accept Connection Request Command

Command	OCF	Command Parameters	Return Parameters
HCI_Accept_Connection_Request	0x0009	BD_ADDR, Role	

Description:

The Accept_Connection_Request command is used to accept a new incoming connection request. The Accept_Connection_Request command shall only be issued after a Connection Request event has occurred. The Connection Request event will return the BD_ADDR of the device which is requesting the connection. This command will cause the Link Manager to create a connection to the BR/EDR Controller, with the BD_ADDR specified by the command parameters. The Link Manager will determine how the new connection will be established. This will be determined by the current state of the device, its piconet, and the state of the device to be connected. The Role command parameter allows the Host to specify if the Link Manager shall request a role switch and become the Master for this connection. This is a preference and not a requirement. If the Role Switch fails then the connection will still be accepted, and the Role Discovery Command will reflect the current role.

Note: The Link Manager may terminate the connection if it would be low on resources if the role switch fails. The decision to accept a connection must be completed before the connection accept timeout expires on the local Bluetooth Module.

Note: When accepting synchronous connection request, the Role parameter is not used and will be ignored by the BR/EDR Controller.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device to be connected

Role: *Size: 1 Octet*

Value	Parameter Description
0x00	Become the Master for this connection. The LM will perform the role switch.
0x01	Remain the Slave for this connection. The LM will **NOT** perform the role switch.

Return Parameters:

None.

Event(s) generated (unless masked away):

The Accept_Connection_Request command shall cause the Command Status event to be sent from the BR/EDR Controller when the BR/EDR Controller begins setting up the connection. In addition, when the Link Manager determines the connection is established, the local BR/EDR Controller shall send a Connection Complete event to its Host, and the remote Controller will send a Connection Complete event or a Synchronous Connection Complete event to the Host. The Connection Complete event contains the Connection_Handle if this command is successful.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Connection Complete event will indicate that this command has been completed.

7.1.9 Reject Connection Request Command

Command	OCF	Command Parameters	Return Parameters
HCI_Reject_Connection_Request	0x000A	BD_ADDR, Reason	

Description:

The Reject_Connection_Request command is used to decline a new incoming connection request. The Reject_Connection_Request command shall only be called after a Connection Request event has occurred. The Connection Request event will return the BD_ADDR of the device that is requesting the connection. The Reason command parameter will be returned to the connecting device in the Status parameter of the Connection Complete event returned to the Host of the connection device, to indicate why the connection was declined.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device to reject the connection from.

Reason: *Size: 1 Octet*

Value	Parameter Description
0x0D-0x0F	Host Reject Error Code. See Part D, Error Codes on page 339 for list of Error Codes and descriptions.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Controller receives the Reject_Connection_Request command, the Controller shall send the Command Status event to the Host. Then, the local BR/EDR Controller will send a Connection Complete event to its host, and the remote device shall send a Connection Complete event or a Synchronous Connection Complete event to the host. The Status parameter of the Connection Complete event, which is sent to the Host of the device attempting to make the connection, will contain the Reason command parameter from this command.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Connection Complete event will indicate that this command has been completed.

7.1.10 Link Key Request Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_Link_Key_Request_Reply	0x000B	BD_ADDR, Link_Key	Status, BD_ADDR

Description:

The Link_Key_Request_Reply command is used to reply to a Link Key Request event from the Controller, and specifies the Link Key stored on the Host to be used as the link key for the connection with the other BR/EDR Controller specified by BD_ADDR. The Link Key Request event will be generated when the BR/EDR Controller needs a Link Key for a connection.

When the BR/EDR Controller generates a Link Key Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host must respond with either a Link_Key_Request_Reply or Link_Key_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout. (See Part C, Link Manager Protocol Specification on page 207.)

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device of which the Link Key is for.

Link_Key: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXX XXXXXXXXXX XXXXXXXXXX	Link Key for the associated BD_ADDR.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Link_Key_Request_Reply command succeeded.
0x01-0xFF	Link_Key_Request_Reply command failed. See Part D, Error Codes on page 339 for error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device of which the Link Key request reply has completed.

Event(s) generated (unless masked away):

When the Link_Key_Request_Reply command has completed, a Command Complete event shall be generated.

7.1.11 Link Key Request Negative Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_Link_Key_Request_Negative_Reply	0x000C	BD_ADDR	Status, BD_ADDR

Description:

The Link_Key_Request_Negative_Reply command is used to reply to a Link Key Request event from the BR/EDR Controller if the Host does not have a stored Link Key for the connection with the other BR/EDR Controller specified by BD_ADDR. The Link Key Request event will be generated when the BR/EDR Controller needs a Link Key for a connection.

When the Controller generates a Link Key Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host must respond with either a Link_Key_Request_Reply or Link_Key_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout. (See Part C, Link Manager Protocol Specification on page 207.)

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device which the Link Key is for.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Link_Key_Request_Negative_Reply command succeeded.
0x01-0xFF	Link_Key_Request_Negative_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device which the Link Key request negative reply has completed.

Event(s) generated (unless masked away):

When the Link_Key_Request_Negative_Reply command has completed, a Command Complete event shall be generated.

7.1.12 PIN Code Request Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_PIN_Code_Request_Reply	0x000D	BD_ADDR, PIN_Code_Length, PIN_Code	Status, BD_ADDR

Description:

The PIN_Code_Request_Reply command is used to reply to a PIN Code request event from the BR/EDR Controller, and specifies the PIN code to use for a connection. The PIN Code Request event will be generated when a connection with remote initiating device has requested pairing.

When the BR/EDR Controller generates a PIN Code Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host must respond with either a PIN_Code_Request_Reply or PIN_Code_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout. (See Part C, Link Manager Protocol Specification on page 207.)

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device which the PIN code is for.

PIN_Code_Length: *Size: 1 Octet*

Value	Parameter Description
0xXX	The PIN code length specifics the length, in octets, of the PIN code to be used. Range: 0x01-0x10

PIN_Code: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXX XXXXXXXXXX XXXXXXXXXX	PIN code for the device that is to be connected. The Host should ensure that strong PIN Codes are used. PIN Codes can be up to a maximum of 128 bits. Note: The PIN_Code Parameter is a string parameter. Endianess does therefore not apply to the PIN_Code Parameter. The first octet of the PIN code should be transmitted first.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	PIN_Code_Request_Reply command succeeded.
0x01-0xFF	PIN_Code_Request_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXX XXXX	BD_ADDR of the Device which the PIN Code request reply has completed.

Event(s) generated (unless masked away):

When the PIN_Code_Request_Reply command has completed, a Command Complete event shall be generated.

7.1.13 PIN Code Request Negative Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_PIN_Code_ Request_Negative_Reply	0x000E	BD_ADDR	Status, BD_ADDR

Description:

The PIN_Code_Request_Negative_Reply command is used to reply to a PIN Code request event from the BR/EDR Controller when the Host cannot specify a PIN code to use for a connection. This command will cause the pair request with remote device to fail.

When the BR/EDR Controller generates a PIN Code Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host must respond with either a PIN_Code_Request_Reply or PIN_Code_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout. (See Part C, Link Manager Protocol Specification on page 207.)

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device which this command is responding to.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	PIN_Code_Request_Negative_Reply command succeeded.
0x01-0xFF	PIN_Code_Request_Negative_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device which the PIN Code request negative reply has completed.

Event(s) generated (unless masked away):

When the PIN_Code_Request_Negative_Reply command has completed, a Command Complete event shall be generated.

7.1.14 Change Connection Packet Type Command

Command	OCF	Command Parameters	Return Parameters
HCI_Change_Connection_ Packet_Type	0x000F	Connection_Handle, Packet_Type	

Description:

The Change_Connection_Packet_Type command is used to change which packet types can be used for a connection that is currently established. This allows current connections to be dynamically modified to support different types of user data. The Packet_Type command parameter specifies which packet types the Link Manager can use for the connection. When sending HCI ACL Data Packets the Link Manager shall only use the packet type(s) specified by the Packet_Type command parameter or the always-allowed DM1 packet type. The interpretation of the value for the Packet_Type command parameter will depend on the Link_Type command parameter returned in the Connection Complete event at the connection setup. Multiple packet types may be specified for the Packet_Type command parameter by bitwise OR operation of the different packet types. For a definition of the different packet types see the Part B, Baseband Specification on page 59.

Note: The Host should enable as many packet types as possible for the Link Manager to perform efficiently. However, the Host must not enable packet types that the local device does not support.

Note: To change an eSCO connection, use the Setup Synchronous Connection command.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to for transmitting and receiving voice or data. Returned from creating a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Packet_Type: *Size: 2 Octets*

For ACL Link_Type

Value	Parameter Description
0x0001	Reserved for future use.
0x0002	2-DH1 may not be used.
0x0004	3-DH1 may not be used.

Value	Parameter Description
0x0008[1]	DM1 may be used.
0x0010	DH1 may be used.
0x0020	Reserved for future use.
0x0040	Reserved for future use.
0x0080	Reserved for future use.
0x0100	2-DH3 may not be used.
0x0200	3-DH3 may not be used.
0x0400	DM3 may be used.
0x0800	DH3 may be used.
0x1000	2-DH5 may not be used.
0x2000	3-DH5 may not be used.
0x4000	DM5 may be used.
0x8000	DH5 may be used.

1. This bit will be interpreted as set to 1 by Bluetooth V1.2 or later controllers.

For SCO Link Type

Value	Parameter Description
0x0001	Reserved for future use.
0x0002	Reserved for future use.
0x0004	Reserved for future use.
0x0008	Reserved for future use.
0x0010	Reserved for future use.
0x0020	HV1
0x0040	HV2
0x0080	HV3
0x0100	Reserved for future use.
0x0200	Reserved for future use.
0x0400	Reserved for future use.
0x0800	Reserved for future use.
0x1000	Reserved for future use.
0x2000	Reserved for future use.
0x4000	Reserved for future use.
0x8000	Reserved for future use.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Change Connection Packet Type command, BR/EDR the Controller shall send the Command Status event to the Host. In addition, when the Link Manager determines the packet type has been changed for the connection, the Controller on the local device will send a Connection Packet Type Changed event to the Host. This will be done at the local side only.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Connection Packet Type Changed event will indicate that this command has been completed.

7.1.15 Authentication Requested Command

Command	OCF	Command Parameters	Return Parameters
HCI_Authentication_ Requested	0x0011	Connection_Handle	

Description:

The Authentication_Requested command is used to try to authenticate the remote device associated with the specified Connection_Handle. On an authentication failure, the BR/EDR Controller or Link Manager shall not automatically detach the link. The Host is responsible for issuing a Disconnect command to terminate the link if the action is appropriate.

Note: The Connection_Handle command parameter is used to identify the other BR/EDR Controller, which forms the connection. The Connection_Handle should be a Connection_Handle for an ACL connection.

Command Parameters:

Connection_Handle: *Size 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to set up authentication for all Connection Handles with the same BR/EDR Controller end-point as the specified Connection_Handle. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Controller receives the Authentication_Requested command, it shall send the Command Status event to the Host. The Link Key Request event shall be generated when Simple Pairing Mode is enabled. If authentication fails and Simple Pairing Mode is not enabled, the PIN Code Request event may be generated. The Authentication Complete event will occur when the authentication has been completed for the connection and is indication that this command has been completed.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Authentication Complete event will indicate that this command has been completed.

On receipt of an Authentication Requested Command, a local BR/EDR Controller that is not in Simple Pairing Mode may use an existing stored link key and respond immediately to the Host with an Authentication Complete event.

This behavior is not recommended for controllers as it offers the Host no method for enhancing the security of an existing link (e.g., in the case where a profile mandating a minimum passkey length is started over a link that is already authenticated with shorter passkey than the new service requires).

7.1.16 Set Connection Encryption Command

Command	OCF	Command Parameters	Return Parameters
HCI_Set_Connection_ Encryption	0x0013	Connection_Handle, Encryption_Enable	

Description:

The Set_Connection_Encryption command is used to enable and disable the link level encryption. Note**:** the Connection_Handle command parameter is used to identify the other BR/EDR Controller which forms the connection. The Connection_Handle should be a Connection_Handle for an ACL connection. While the encryption is being changed, all ACL traffic must be turned off for all Connection_Handles associated with the remote device.

Command Parameters:

Connection_Handle: *Size 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to enable/disable the link layer encryption for all Connection_Handles with the same BR/EDR Controller end-point as the specified Connection_Handle. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Encryption_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	Turn Link Level Encryption OFF.
0x01	Turn Link Level Encryption ON.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Set_Connection_Encryption command, the BR/EDR Controller shall send the Command Status event to the Host. When the Link Manager has completed enabling/disabling encryption for the connection, the local BR/EDR Controller shall send an Encryption Change event to the Host, and the BR/EDR Controller on the remote device will also generate an Encryption Change event.

Note: No Command Complete event will be sent by the Controller to indicate that this command has been completed. Instead, the Encryption Change event will indicate that this command has been completed.

7.1.17 Change Connection Link Key Command

Command	OCF	Command Parameters	Return Parameters
HCI_Change_Connection_Link_Key	0x0015	Connection_Handle	

Description:

The Change_Connection_Link_Key command is used to force both devices of a connection associated with the Connection_Handle to generate a new link key. The link key is used for authentication and encryption of connections.

Note: The Connection_Handle command parameter is used to identify the other BR/EDR Controller forming the connection. The Connection_Handle should be a Connection_Handle for an ACL connection.

Note: The resulting link key, generated as a result of Change_Connection_Link_Key command, will be of equal link key strength to the previously used link key.

Command Parameters:

Connection_Handle: *Size 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Controller receives the Change_Connection_Link_Key command, the BR/EDR Controller shall send the Command Status event to the Host. When the Link Manager has changed the Link Key for the connection, the local BR/EDR Controller shall send a Link Key Notification event and a Change Connection Link Key Complete event to the Host, and the remote BR/EDR Controller will also generate a Link Key Notification event. The Link Key Notification event indicates that a new connection link key is valid for the connection.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Change Connection Link Key Complete event will indicate that this command has been completed.

7.1.18 Master Link Key Command

Command	OCF	Command Parameters	Return Parameters
HCI_Master_Link_Key	0x0017	Key_Flag	

Description:

The Master_Link_Key command is used to force the device that is master of the piconet to use the temporary link key of the master device, or the semi-permanent link keys. The temporary link key is used for encryption of broadcast messages within a piconet, and the semi-permanent link keys are used for private encrypted point-to-point communication. The Key_Flag command parameter is used to indicate which Link Key (temporary link key of the Master, or the semi-permanent link keys) shall be used.

Command Parameters:

Key_Flag: *Size: 1 Octet*

Value	Parameter Description
0x00	Use semi-permanent Link Keys.
0x01	Use Temporary Link Key.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Master_Link_Key command, the BR/EDR Controller shall send the Command Status event to the Host. When the Link Manager has changed link key, the BR/EDR Controller on both the local and the remote device shall send a Master Link Key Complete event to the Host. The Connection_Handle on the master side shall be a Connection_Handle for one of the existing connections to a slave. On the slave side, the Connection_Handle shall be a Connection_Handle to the initiating master.

The Master Link Key Complete event contains the status of this command.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Master Link Key Complete event will indicate that this command has been completed.

7.1.19 Remote Name Request Command

Command	OCF	Command Parameters	Return Parameters
HCI_Remote_Name_Request	0x0019	BD_ADDR, Page_Scan_Repetition_Mode, Reserved, Clock_Offset	

Description:

The Remote_Name_Request command is used to obtain the user-friendly name of another BR/EDR Controller. The user-friendly name is used to enable the user to distinguish one BR/EDR Controller from another. The BD_ADDR command parameter is used to identify the device for which the user-friendly name is to be obtained. The Page_Scan_Repetition_Mode parameter specifies the page scan repetition mode supported by the remote device with the BD_ADDR. This is the information that was acquired during the inquiry process. The Clock_Offset parameter is the difference between its own clock and the clock of the remote device with BD_ADDR. Only bits 2 through 16 of the difference are used and they are mapped to this parameter as bits 0 through 14 respectively. A Clock_Offset_Valid_Flag, located in bit 15 of the Clock_Offset command parameter, is used to indicate if the Clock Offset is valid or not.

When the Remote Supported Host Features Notification event is unmasked and when the Remote_Name_Request command initiates a connection, the Link Manager shall read the remote LMP features mask pages 0 and 1.

Note: If no connection exists between the local device and the device corresponding to the BD_ADDR, a temporary link layer connection will be established to obtain the LMP features and name of the remote device.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for the device whose name is requested.

Page_Scan_Repetition_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	R0
0x01	R1
0x02	R2
0x03 – 0xFF	Reserved.

Reserved: *Size: 1 Octet*

Value	Parameter Description
0x00	Reserved, must be set to 0x00. See Page Scan Mode on page 871.

Clock_Offset: *Size: 2 Octets*

Bit format	Parameter Description
Bit 14.0	Bit 16.2 of CLKslave-CLKmaster.
Bit 15	Clock_Offset_Valid_Flag Invalid Clock Offset = 0 Valid Clock Offset = 1

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Remote_Name_Request command, the BR/EDR Controller shall send the Command Status event to the Host. When the Link Manager has completed the LMP messages to obtain the remote host supported features, if present, the BR/EDR Controller on the local BR/EDR Controller shall send a Remote Host Supported Features Notification event. When the Link Manager has completed the LMP messages to obtain the remote name, the BR/EDR Controller on the local BR/EDR Controller shall send a Remote Name Request Complete event to the Host. If the remote host supported features page is present, the Remote Host Supported Features Notification event shall be sent before the Remote Name Request Complete event. If not, only the Remote Name Request Complete event shall be sent.

Note: No Command Complete event shall be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, only the Remote Name Request Complete event will indicate that this command has been completed.

7.1.20 Remote Name Request Cancel Command

Command	OCF	Command Parameters	Return Parameters
HCI_Remote_Name_Request_Cancel	0x001A	BD_ADDR	Status, BD_ADDR

Description:

This command is used to request cancellation of the ongoing remote name request process, which was started by the Remote Name Request command.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Remote Name Request command that was issued before and that is subject of this cancellation request

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Remote Name Request Cancel command succeeded
0x01-0xff	Remote Name Request Cancel command failed. See Part D, Error Codes on page 339 for list of error codes

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Remote Name Request Cancel command that was issued before and that was subject of this cancellation request

Event(s) generated (unless masked away):

When the Remote Name Request Cancel command has completed, a Command Complete event shall be generated.

If the Remote Name Request Cancel command is sent to the BR/EDR Controller without a preceding Remote Name Request command to the same device, the Controller shall return a Command Complete event with the error code *Invalid HCI Command Parameters* (0x12).

The Remote Name Request Complete event for the corresponding Remote Name Request Command shall always be sent. The Remote Name Request Complete event shall be sent after the Command Complete event for the

Remote Name Request Cancel command. If the cancellation was successful, the Remote Name Request Complete event shall be generated with the error code *Unknown Connection Identifier (0x02)*.

7.1.21 Read Remote Supported Features Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Remote_Supported_Features	0x001B	Connection_Handle	

Description:

This command requests a list of the supported features for the remote device identified by the Connection_Handle parameter. The Connection_Handle must be a Connection_Handle for an ACL connection. The Read Remote Supported Features Complete event will return a list of the LMP features. For details see Part C, Link Manager Protocol Specification on page 207.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's LMP-supported features list to get. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Read_Remote_Supported_Features command, the BR/EDR Controller shall send the Command Status event to the Host. When the Link Manager has completed the LMP messages to determine the remote features, the BR/EDR Controller on the local BR/EDR Controller shall send a Read Remote Supported Features Complete event to the Host. The Read Remote Supported Features Complete event contains the status of this command, and parameters describing the supported features of the remote device.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Read Remote Supported Features Complete event will indicate that this command has been completed.

7.1.22 Read Remote Extended Features Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Remote_Extended_Features	0x001C	Connection_Handle , Page Number	

Description:

The Read_Remote_Extended_Features command returns the requested page of the extended LMP features for the remote device identified by the specified Connection_Handle. The Connection_Handle must be the Connection_Handle for an ACL connection. This command is only available if the extended features feature is implemented by the remote device. The Read Remote Extended Features Complete event will return the requested information. For details see Part C, Link Manager Protocol Specification on page 207.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection_Handle identifying the remote device for which extended feature information is required. Range: 0x0000-0x0EFF (0x0F00-0x0FFF Reserved for future use)

Page Number: *Size: 1 Octet*

Value	Parameter Description
0x00	Requests the normal LMP features as returned by the Read_Remote_Supported_Features command
0x01-0xFF	Return the corresponding page of features

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Read_Remote_Extended_Features command the BR/EDR Controller shall send the Command Status event to the Host. When the Link Manager has completed the LMP sequence to determine the remote extended features the controller on the local device shall generate a Read Remote Extended Features Complete event to the host. The Read Remote Extended Features Complete event contains the page number and the remote features returned by the remote device.

Note: No Command Complete event will ever be sent by the BR/EDR Controller to indicate that this command has been completed. Instead the Read Remote Extended Features Complete event will indicate that this command has been completed.

7.1.23 Read Remote Version Information Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Remote_Version_ Information	0x001D	Connection_Handle	

Description:

This command will obtain the values for the version information for the remote device identified by the Connection_Handle parameter. The Connection_Handle must be a Connection_Handle for an ACL or LE connection.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's version information to get. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Controller receives the Read_Remote_Version_Information command, the Controller shall send the Command Status event to the Host. When the Link Manager or Link Layer has completed the sequence to determine the remote version information, the local Controller shall send a Read Remote Version Information Complete event to the Host. The Read Remote Version Information Complete event contains the status of this command, and parameters describing the version and subversion of the LMP or Link Layer used by the remote device.

Note: No Command Complete event will be sent by the Controller to indicate that this command has been completed. Instead, the Read Remote Version Information Complete event will indicate that this command has been completed.

7.1.24 Read Clock Offset Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Clock_Offset	0x001F	Connection_Handle	

Description:

Both the System Clock and the clock offset to a remote device are used to determine what hopping frequency is used by a remote device for page scan. This command allows the Host to read clock offset to remote devices. The clock offset can be used to speed up the paging procedure when the local device tries to establish a connection to a remote device, for example, when the local Host has issued Create_Connection or Remote_Name_Request. The Connection_Handle must be a Connection_Handle for an ACL connection.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's Clock Offset parameter is returned. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Read_Clock_Offset command, the BR/EDR Controller shall send the Command Status event to the Host. If this command was requested at the master and the Link Manager has completed the LMP messages to obtain the Clock Offset information, the BR/EDR Controller on the local BR/EDR Controller shall send a Read Clock Offset Complete event to the Host.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, only the Read Clock Offset Complete event will indicate that this command has been completed. If the command is requested at the slave, the LM will immediately send a Command Status event and a Read Clock Offset Complete event to the Host, without an exchange of LMP PDU.

7.1.25 Read LMP Handle Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_LMP_Handle	0x0020	Connection_Handle	Status, Connection_Handle, LMP_Handle, Reserved

Description:

This command reads the current LMP Handle associated with the Connection_Handle. The Connection_Handle shall be a SCO or eSCO Handle. If the Connection_Handle is a SCO Connection_Handle, then this command shall read the LMP SCO Handle for this connection. If the Connection_Handle is an eSCO Connection_Handle, then this command shall read the LMP eSCO Handle for this connection.

Command Parameters:

Connection_Handle: *Size 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify which connection to be used for reading the LMP Handle. This must be a synchronous handle. Range: 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_LMP_Handle command succeeded.
0x01 – 0xFF	Read_LMP_Handle command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection_Handle for the Connection for which the LMP_Handle has been read. Range: 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

LMP_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	The LMP Handle is the LMP Handle that is associated with this Connection_Handle. For a synchronous handle, this would be the LMP Synchronous Handle used when negotiating the synchronous connection in the link manager.

Reserved: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	This parameter is reserved, must to set to zero.

Events(s) generated (unless masked away):

When the Read_LMP_Handle command has completed, a Command Complete event shall be generated.

7.1.26 Setup Synchronous Connection Command

Command	OCF	Command Parameters	Return Parameters
HCI_Setup_Synchronous_ Connection	0x0028	Connection_Handle Transmit_Bandwidth Receive_Bandwidth Max_Latency Voice_Setting Retransmission_Effort Packet Type	

Description:

The Setup_Synchronous_Connection command adds a new or modifies an existing synchronous logical transport (SCO or eSCO) on a physical link depending on the Connection_Handle parameter specified. If the Connection_Handle refers to an ACL link a new synchronous logical transport will be added. If the Connection_Handle refers to an already existing synchronous logical transport (eSCO only) this link will be modified. The parameters are specified per connection. This synchronous connection can be used to transfer synchronous voice at 64kbps or transparent synchronous data.

When used to setup a new synchronous logical transport, the Connection_Handle parameter shall specify an ACL connection with which the new synchronous connection will be associated. The other parameters relate to the negotiation of the link, and may be reconfigured during the lifetime of the link. The transmit and receive bandwidth specify how much bandwidth shall be available for transmitting and for receiving data. While in many cases the receive and transmit bandwidth parameters may be equal, they may be different. The latency specifies an upper limit to the time in milliseconds between the eSCO (or SCO) instants, plus the size of the retransmission window, plus the length of the reserved synchronous slots for this logical transport. The content format specifies the settings for voice or transparent data on this connection. The retransmission effort specifies the extra resources that are allocated to this connection if a packet may need to be retransmitted. The Retransmission_Effort parameter shall be set to indicate the required behavior, or to don't care.

When used to modify an existing synchronous logical transport, the Transmit_Bandwidth, Receive_Bandwidth and Voice_Settings shall be set to the same values as were used during the initial setup. The Packet_Type, Retransmission_Effort and Max_Latency parameters may be modified.

The Packet_Type field is a bitmap specifying which packet types the LM shall accept in the negotiation of the link parameters. Multiple packet types are specified by bitwise OR of the packet type codes in the table. At least one packet type must be specified for each negotiation. It is recommended to enable as many packet types as possible. Note that it is allowed to enable packet types that are not supported by the local device.

A Connection_Handle for the new synchronous connection will be returned in the synchronous connection complete event.

Note: The link manager may choose any combination of packet types, timing, and retransmission window sizes that satisfy the parameters given. This may be achieved by using more frequent transmissions of smaller packets. The link manager may choose to set up either a SCO or an eSCO connection, if the parameters allow, using the corresponding LMP sequences.

Note: To modify a SCO connection, use the Change Connection Packet Type command.

Note: If the lower layers cannot achieve the exact transmit and receive bandwidth requested subject to the other parameters, then the link shall be rejected.

A synchronous connection may only be created when an ACL connection already exists and when it is not in Park state.

Command Parameters:

Connection_Handle: *2 octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle for the ACL connection being used to create a synchronous Connection or for the existing Connection that shall be modified. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Transmit_Bandwidth: *4 octets*

Value	Parameter Description
0xXXXXXXXX	Transmit bandwidth in octets per second.

Receive_Bandwidth: *4 octets*

Value	Parameter Description
0xXXXXXXXX	Receive bandwidth in octets per second.

Max_Latency: *2 octets*

Value	Parameter Description
0x0000-0x0003	Reserved
0x0004-0xFFFE	This is a value in milliseconds representing the upper limit of the sum of the synchronous interval, and the size of the eSCO window, where the eSCO window is the reserved slots plus the retransmission window. (See Figure 8.7 in the Baseband specification)
0xFFFF	Don't care.

Voice_Setting: *2 octets (10 bits meaningful)*

Value	Parameter Description
See Section 6.12 on page 439.	

Retransmission_Effort: *1 octet*

Value	Parameter Description
0x00	No retransmissions
0x01	At least one retransmission, optimize for power consumption.
0x02	At least one retransmission, optimize for link quality
0xFF	Don't care
0x03 – 0xFE	Reserved

Packet_Type: *2 octets*

Value	Parameter Description
0x0001	HV1 may be used.
0x0002	HV2 may be used.
0x0004	HV3 may be used.
0x0008	EV3 may be used.
0x0010	EV4 may be used.
0x0020	EV5 may be used.
0x0040	2-EV3 may not be used.
0x0080	3-EV3 may not be used.
0x0100	2-EV5 may not be used.
0x0200	3-EV5 may not be used.
0x0400	Reserved for future use
0x0800	Reserved for future use
0x1000	Reserved for future use
0x2000	Reserved for future use
0x4000	Reserved for future use
0x8000	Reserved for future use

Return Parameters:

None.

Event(s) generated (unless masked away)

When the BR/EDR Controller receives the Setup_Synchronous_Connection command, it shall send the Command Status event to the Host. In addition, when the LM determines the connection is established, the local BR/EDR Controller shall send a Synchronous Connection Complete event to the local Host, and the remote Controller will send a Synchronous Connection Complete event or a Connection Complete event to the remote Host. The synchronous Connection Complete event contains the Connection_Handle if this command is successful.

If this command is used to change the parameters of an existing eSCO link, the Synchronous Connection Changed Event is sent to both hosts. In this case no Connection Setup Complete Event or Connection Request Event will be sent to either host. This command cannot be used to change the parameters of an SCO link.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Synchronous Connection Complete event will indicate that this command has been completed.

7.1.27 Accept Synchronous Connection Request Command

Command	OCF	Command Parameters	Return Parameters
HCI_Accept_ Synchronous_ Connection_Request	0x0029	BD_ADDR Transmit_Bandwidth Receive_Bandwidth Max_Latency Content_Format Retransmission_Effort Packet_Type	

Description:

The Accept_Synchronous_Connection_Request command is used to accept an incoming request for a synchronous connection and to inform the local Link Manager about the acceptable parameter values for the synchronous connection. The Command shall only be issued after a Connection_Request event with link type SCO or eSCO has occurred. Connection_Request event contains the BD_ADDR of the device requesting the connection. The decision to accept a connection must be taken before the connection accept timeout expires on the local device.

The parameter set of the Accept_Synchronous_Connection_Request command is the same as for the Setup_Synchronous_Connection command. The Transmit_Bandwidth and Receive_Bandwidth values are required values for the new link and shall be met. The Max_Latency is an upper bound to the acceptable latency for the Link, as defined in Section 7.1.26 on page 500 Setup_Synchronous_Connection and shall not be exceeded. Content_Format specifies the encoding in the same way as in the Setup_Synchronous_Connection command and shall be met. The Retransmission_Effort parameter shall be set to indicate the required behavior, or to don't care. The Packet_Type parameter is a bit mask specifying the synchronous packet types that are allowed on the link and shall be met. The reserved bits in the Packet_Type field shall be set to one. If all bits are set in the packet type field then all packets types shall be allowed.

If the Link Type of the incoming request is SCO, then only the Transmit_Bandwidth, Max_Latency, Content_Format, and Packet_Type fields are valid.

If the Connection_Request event is masked away, and the BR/EDR Controller is not set to auto-accept this connection attempt, the BR/EDR Controller will automatically reject it. If the controller is set to automatically accept the connection attempt, the LM should assume default parameters. In that case the Synchronous Connection Complete Event shall be generated, unless masked away.

Command Parameters:

BD_ADDR: *6 octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the device requesting the connection

Transmit_Bandwidth: *4 octets*

Value	Parameter Description
0x00000000-0xFFFFFFFE	Maximum possible transmit bandwidth in octets per second.
0xFFFFFFFF	Don't care

Default: Don't care

Receive_Bandwidth: *4 octets*

Value	Parameter Description
0x00000000-0xFFFFFFFE	Maximum possible receive bandwidth in octets per second.
0xFFFFFFFF	Don't care

Default: Don't care

Max_Latency: *2 octets*

Value	Parameter Description
0x0000-0x0003	Reserved
0x0004-0xFFFE	This is a value in milliseconds representing the upper limit of the sum of the synchronous interval and the size of the eSCO window, where the eSCO window is the reserved slots plus the retransmission window.
0xFFFF	Don't care.

Default: Don't care

Content_Format: *2 octets (10 bits meaningful)*

Value	Parameter Description
00XXXXXXXX	Input Coding: Linear
01XXXXXXXX	Input Coding: u-law
10XXXXXXXX	Input Coding: A-law
11XXXXXXXX	Reserved for future use.
XX00XXXXXX	Input Data Format: 1's complement
XX01XXXXXX	Input Data Format: 2's complement

Value	Parameter Description
XX10XXXXXX	Input Data Format: Sign-Magnitude
XX11XXXXXX	Input Data Format: Unsigned
XXXX0XXXXX	Input Sample Size: 8 bit (only for Linear PCM)
XXXX1XXXXX	Input Sample Size: 16 bit (only for Linear PCM)
XXXXXnnnXX	Linear PCM Bit Position: number of bit positions that MSB of sample is away from starting at MSB (only for Linear PCM)
XXXXXXXX00	Air Coding Format: CVSD
XXXXXXXX01	Air Coding Format: u-law
XXXXXXXX10	Air Coding Format: A-law
XXXXXXXX11	Air Coding Format: Transparent Data

Default: When links are auto-accepted, the values written by the Write_Voice_Settings command are used.

Retransmission_Effort: *1 octet*

Value	Parameter Description
0x00	No retransmissions
0x01	At least one retransmission, optimize for power consumption.
0x02	At least one retransmission, optimize for link quality.
0x03-0xFE	Reserved
0xFF	Don’t care

Default: Don't care

Packet_Type: *2 octets*

Value	Parameter Description
0x0001	HV1 may be used.
0x0002	HV2 may be used.
0x0004	HV3 may be used.
0x0008	EV3 may be used.
0x0010	EV4 may be used.
0x0020	EV5 may be used.
0x0040	2-EV3 may not be used.
0x0080	3-EV3 may not be used
0x0100	2-EV5 may not be used.

Value	Parameter Description
0x0200	3-EV5 may not be used.
0x0400	Reserved for future use
0x0800	Reserved for future use
0x1000	Reserved for future use
0x2000	Reserved for future use
0x4000	Reserved for future use
0x8000	Reserved for future use

Default: 0xFFFF - means all packet types may be used.

Return Parameters:

None.

Event(s) generated (unless masked away):

The Accept_Synchronous_Request command shall cause the Command Status event to be sent from the BR/EDR Controller when the BR/EDR Controller starts setting up the connection. When the link setup is complete, the local BR/EDR Controller shall send a Synchronous Connection Complete event to its Host, and the remote BR/EDR Controller will send a Connection Complete event or a Synchronous Connection Complete event to the Host. The Synchronous Connection Complete will contain the Connection_Handle and the link parameters if the setup is successful.

Note: No Command Complete event will be sent by the host controller as the result of this command. Instead, the Synchronous Connection Complete or Connection Complete event will indicate that this command has been completed.

7.1.28 Reject Synchronous Connection Request Command

Command	OCF	Command Parameters	Return Parameters
HCI_Reject_Synchronous_ Connection_Request	0x002A	BD_ADDR Reason	

Description:

The Reject_Synchronous_Connection_Request command is used to decline an incoming request for a synchronous link. It shall only be issued after a Connection Request Event with Link Type equal to SCO or eSCO has occurred. The Connection Request Event contains the BD_ADDR of the device requesting the connection. The Reason parameter will be returned to the initiating Host in the Status parameter of the Synchronous connection complete event on the remote side.

Command Parameters:

BD_ADDR: *6 octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the device requesting the connection

Reason: *1 octet*

Value	Parameter Description
0x0D-0x0F	Host Reject Error Code. See Part D, Error Codes on page 339 for error codes and descriptions.

Return Parameters:

None.

Event(s) Generated (unless masked away):

When the Host Controller receives the Reject_Synchronous_Connection_Request, it shall send a Command Status event to the Host. When the setup is terminated, the local Controller shall send a Synchronous Connection Complete event to its Host, and the remote Controller will send a Connection Complete event or a Synchronous Connection Complete event to the Host with the Reason code from this command.

Note: No Command complete event will be sent by the Host Controller to indicate that this command has been completed. Instead, the Synchronous Connection Complete or Connection Complete event will indicate that this command has been completed.

7.1.29 IO Capability Request Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_IO_Capability_ Request_Reply	0x002B	BD_ADDR, IO_Capability, OOB_Data_Present, Authentication_ Requirements	Status, BD_ADDR

Description:

The IO_Capability_Request_Reply command is used to reply to an IO Capability Request event from the controller, and specifies the current I/O capabilities of the host. This includes the Host input, output and out-of-band (OOB) capabilities.

The Authentication_Requirements parameter shall be set to man-in-the middle (MITM) Protection Required Single Profile, MITM Protection Required – General Bonding, or MITM Protection Required – Dedicated Bonding if an authenticated link key is required by the Host. The Authentication_Requirements parameter may be set to MITM Protection Not Required - Single Profile, MITM Protection Not Required – General Bonding, or MITM Protection Not Required – Dedicated Bonding if an authenticated link key is not required. If one or both Hosts set the Authentication_Requirements parameter to MITM Protection Required - Single Profile, MITM Protection Required – General Bonding, or MITM Protection Required – Dedicated Bonding, the Link Managers shall use the IO_Capability parameter to determine the authentication procedure. A Host that sets the Authentication_Requirements parameter to MITM Protection Required - Single Profile, MITM Protection Required – General Bonding, or MITM Protection Required – All Profiles shall verify that the resulting Link Key type meets the security requirements.

If both Hosts set the Authentication_Requirements parameter to MITM Protection Not Required - Single Profile, MITM Protection Not Required – General Bonding, or MITM Protection Not Required – Dedicated Bonding, the Link Managers shall use the numeric comparison authentication procedure and the Hosts shall use the Just Works Association Model.

The OOB_Data_Present parameter shall be set to "OOB authentication data from remote device present" if the host has received OOB data from a device with the same BD_ADDR sent in the IO Capability Request event. Otherwise OOB_Data_Present shall be set to "OOB authentication data not present".

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

IO_Capability: *Size: 1 Octet*

Value	Parameter Description
0x00	DisplayOnly
0x01	DisplayYesNo
0x02	KeyboardOnly
0x03	NoInputNoOutput
0x04 - 0xFF	Reserved for future use

OOB_Data_Present: *Size: 1 Octet*

Value	Parameter Description
0x00	OOB authentication data not present
0x01	OOB authentication data from remote device present
0x02 - 0xFF	Reserved for future use

Authentication_Requirements: *Size: 1 Octet*

Value	Parameter Description
0x00	MITM Protection Not Required – No Bonding. Numeric comparison with automatic accept allowed.
0x01	MITM Protection Required – No Bonding. Use IO Capabilities to determine authentication procedure
0x02	MITM Protection Not Required – Dedicated Bonding. Numeric comparison with automatic accept allowed.
0x03	MITM Protection Required – Dedicated Bonding. Use IO Capabilities to determine authentication procedure
0x04	MITM Protection Not Required – General Bonding. Numeric Comparison with automatic accept allowed.
0x05	MITM Protection Required – General Bonding. Use IO capabilities to determine authentication procedure.
0x06 - 0xFF	Reserved for future use

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	IO_Capability_Request_Reply command succeeded
0x01 - 0xFF	IO_Capability_Request_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Event(s) generated (unless masked away):

When the IO_Capability_Request_Reply command has completed, a Command Complete event shall be generated. When the device is the initiator of Secure Simple Pairing, an IO Capability Response event shall be generated. Additionally, when the OOB_Data_Present parameter indicates that OOB authentication data from the remote device is present, the Remote OOB Data Request event shall be generated.

7.1.30 User Confirmation Request Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_User_Confirmation_ Request_Reply	0x002C	BD_ADDR	Status, BD_ADDR

Description:

The User_Confirmation_Request_Reply command is used to reply to a User Confirmation Request event and indicates that the user selected "yes". It is also used when the host has no input and no output capabilities.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	User_Confirmation_Request_Reply command succeeded
0x01 - 0xFF	User_Confirmation_Request_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Event(s) generated (unless masked away):

When the User_Confirmation_Request_Reply command has completed, a Command Complete event shall be generated.

7.1.31 User Confirmation Request Negative Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_User_Confirmation_ Request_Negative_Reply	0x002D	BD_ADDR	Status BD_ADDR

Description:

The User_Confirmation_Request_Negative_Reply command is used to reply to a User Confirmation Request event and indicates that the user selected "no". This command shall cause the initiating Link Manager to transmit an LMP_Numeric_Comparison_Failed PDU and terminate Secure Simple Pairing.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	User_Confirmation_Request_Negative_Reply command succeeded
0x01 - 0xFF	User_Confirmation_Request_Negative_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Event(s) generated (unless masked away):

When the User_Confirmation_Request_Negative_Reply command has completed, a Command Complete event shall be generated.

7.1.32 User Passkey Request Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_User_Passkey_Request_Reply	0x002E	BD_ADDR, Numeric_Value	Status, BD_ADDR

Description:

The User_Passkey_Request_Reply command is used to reply to a User Passkey Request event and specifies the Numeric_Value (passkey) entered by the user to be used in the Secure Simple Pairing process.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Numeric_Value: *Size: 4 Octets*

Value	Parameter Description
0x00000000 - 0x000F423F	Numeric value (passkey) entered by user. Valid values are decimal 000000 - 999999.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	User_Passkey_Request_Reply command succeeded
0x01 - 0xFF	User_Passkey_Request_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Event(s) generated (unless masked away):

When the User_Passkey_Request_Reply command has completed, a Command Complete event shall be generated.

7.1.33 User Passkey Request Negative Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_User_Passkey_Request Negative_Reply	0x002F	BD_ADDR	Status, BD_ADDR

Description:

The User_Passkey_Request_Negative_Reply command is used to reply to a User Passkey Request event and indicates the host could not provide a pass-key. This command shall cause the initiating Link Manager to transmit an LMP_Passkey_Entry_Failed PDU and terminate Simple Pairing.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXX X	BD_ADDR of remote device involved in simple pairing process

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	User_Passkey_Request_Negative_Reply command succeeded
0x01 - 0xFF	User_Passkey_Request_Negative_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXX X	BD_ADDR of remote device involved in simple pairing process

Event(s) generated (unless masked away):

When the User_Passkey_Negative_Request_Reply command has completed, a Command Complete event shall be generated.

7.1.34 Remote OOB Data Request Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_Remote_OOB_Data_Request_Reply	0x0030	BD_ADDR, C, R	Status, BD_ADDR

Description:

The Remote_OOB_Data_Request_Reply command is used to reply to a Remote OOB Data Request event with the C and R values received via an OOB transfer from a remote device identified by BD_ADDR. If the R value is not present in the received OOB data from the remote device, the Host shall set R to zeros.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXX	BD_ADDR of remote device from which the C and R values were received

C: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Simple Pairing Hash C

R: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Simple Pairing Randomizer R

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Remote_OOB_Data_Request_Reply command succeeded
0x01 - 0xFF	Remote_OOB_Data_Request_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXX X	BD_ADDR of remote device involved in simple pairing process

Event(s) generated (unless masked away):

When the Remote_OOB_Data_Request_Reply command has completed, a Command Complete event shall be generated.

7.1.35 Remote OOB Data Request Negative Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_Remote_OOB_Data_ Request_Negative_Reply	0x0033	BD_ADDR	Status, BD_ADDR

Description:

The Remote_OOB_Data_Request_Negative_Reply command is used to reply to a Remote OOB Data Request event that the Host does not have the C and R values associated with the remote device identified by BD_ADDR.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Remote_OOB_Data_Request_Negative_Reply command succeeded
0x01 - 0xFF	Remote_OOB_Data_Request_Negative_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Event(s) generated (unless masked away):

When the Remote_OOB_Data_Request_Negative_Reply command has completed, a Command Complete event shall be generated.

7.1.36 IO Capability Request Negative Reply Command

Command	OCF	Command Parameters	Return Parameters
HCI_IO_Capability_ Request_Negative_Reply	0x0034	BD_ADDR, Reason	Status, BD_ADDR

Description:

The IO_Capability_Request_Negative_Reply command shall be used to reject a pairing attempt after an HCI IO Capability Request event has been received by the Host. The reason for the rejection is given in the Reason parameter. Error code 0x37 (Simple Pairing not Supported by Host) shall not be used in the Reason parameter.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in the Simple Pairing process

Reason: *Size: 1 Octet*

Value	Parameter Description
0xXX	Reason that Simple Pairing rejected. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	IO_Capability_Request_Negative_Reply command succeeded.
0x01 - 0xFF	IO_Capability_Request_Negative_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in Simple Pairing process

Event(s) generated (unless masked away):

When the IO_Capability_Request_Negative_Reply command has completed, a Command Complete event shall be generated.

7.1.37 Create Physical Link Command

Command	OCF	Command Parameters	Return Parameters
HCI_Create_Physical_Link	0x0035	Physical_Link_Handle, Dedicated_AMP_Key_Length, Dedicated_AMP_Key_Type, Dedicated_AMP_Key	

Description:

When this command is received by an AMP Controller, the AMP Controller shall initiate a connection to another AMP Controller. The AMP_ASSOC used to create the physical link shall be written to the AMP Controller using the Write Remote AMP_ASSOC command following the Create Physical Link command.

The Dedicated_AMP_Key_Length parameter is used by the Host to indicate the number of valid octets of the Dedicated_AMP_Key. The AMP shall use the supplied key to generate a session key in order to encrypt all data on the indicated physical link. The method by which to do this is AMP type specific. The Dedicated_AMP_Key_Type parameter indicates whether the key is an authenticated combination key, an unauthenticated combination key, or a debug combination key.

Command Parameters:

Physical _Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical Link Handle identifying the physical link to be created (see Section 5.3.2).
0x00	Reserved

Dedicated_AMP_Key_Length: *Size: 1 Octet*

Value	Parameter Description
0	Reserved
0xXX	See Assigned Numbers.
33-255	Reserved

Dedicated_AMP_Key_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Reserved
0x01	Reserved
0x02	Reserved
0x03	Debug combination keys
0x04	Unauthenticated combination key
0x05	Authenticated combination key
0x06	Reserved
0x07-0xFF	Reserved

Dedicated_AMP_Key: *Size: AMP_Key_Length Octets*

Value	Parameter Description
0xXX...XX	Dedicated AMP Key for the associated AMP. The Dedicated AMP Key is a multi-octet integer.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the AMP Controller receives the Create Physical Link command, the AMP Controller shall send the Command Status event to the Host. When the AMP Controller has determined underlying connection details to populate the AMP_ASSOC the AMP Controller shall generate a Channel Selected event to the Host. After the Channel Selected event is received the Host uses the Read Local AMP_ASSOC command to obtain the capabilities and status of the local AMP. The AMP_ASSOC is sent to the remote AMP to be used to perform a matching Accept Physical Link command. When the connection has been completed the AMP Controller on both devices that form the connection shall send a Physical Link Complete event to its Host. The Physical Link Complete event indicates to the Host whether the physical link creation was successful. This sequence must be completed before any Logical Links using this Controller can be created.

If the AMP physical link security establishment process fails (for instance, if the two AMP Controllers do not have matching keys) then the physical link shall be disconnected and the Physical Link Complete event shall be sent to the Host with the Authentication Failure error code see Part D, Error Codes on page 339.

Note: No Command Complete event will be sent by the AMP Controller to indicate that this command has been completed. Instead, the Physical Link Complete event will indicate that this command has been completed.

7.1.38 Accept Physical Link Command

Command	OCF	Command Parameters	Return Parameters
HCI_Accept_Physical_Link	0x0036	Physical_Link_Handle, Dedicated_AMP_Key_Length, Dedicated_AMP_Key_Type, Dedicated_AMP_Key	

Description:

When this command is received by an AMP Controller, the AMP Controller shall attempt to accept a connection request from the initiating BR/EDR Controller's AMP Controller. The AMP_ASSOC used to create the physical link shall be written to the AMP Controller using the Write_Remote_AMP_ASSOC command after issuing the Accept_Physical_Link command.

The Dedicated_AMP_Key_Length parameter is used by the Host to indicate the number of valid octets of the Dedicated_AMP_Key. The AMP shall use the Dedicated_AMP_Key for AMP type-specific security functions. The Dedicated_AMP_Key_Type parameter indicates whether the key is an authenticated combination key, an unauthenticated combination key, or a debug combination key.

Command Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical Link Handle identifying the physical link to be created.

Dedicated_AMP_Key_Length: *Size: 1 Octet*

Value	Parameter Description
0x00	Reserved
0xXX	See Assigned Numbers.
33-255	Reserved

Dedicated_AMP_Key_Type *Size: 1 Octet*

Value	Parameter Description
0x00	Reserved
0x01	Reserved
0x02	Reserved
0x03	Debug combination key

Value	Parameter Description
0x04	Authenticated combination key
0x05	Unauthenticated combination key
0x06	Reserved
0x07-0xFF	Reserved

Dedicated_AMP_Key *Size: Dedicated_AMP_Key_Length Octets*

Value	Parameter Description
0xXX..XX	Dedicated AMP Key for the associated AMP. The Dedicated AMP Key is a multi-octet integer.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the AMP Controller receives the Accept Physical Link command, the AMP Controller shall send the Command Status event to the Host. When the connection has been completed, the AMP Controller shall send a Physical Link Complete event to the Host. The Physical Link Complete event indicates to the Host whether the link creation was successful. This sequence must be completed before any Logical Links can be created.

If communication is not established within the Connection Accept Timeout then a Physical Link Complete event shall be returned with a Status value of Connection Accept Timeout Exceeded (0X10).

If the AMP-specific security establishment process fails (for instance, if the two AMP Controllers do not have matching keys) then the physical link shall be deleted and the Physical Link Complete event sent to the Host with the Authentication Failure error code.

No Physical Link Complete event shall be sent before the security is established.

Note: No Command Complete event will be sent by the AMP Controller to indicate that this command has been completed. Instead, the Physical Link Complete event will indicate that this command has been completed.

7.1.39 Disconnect Physical Link Command

Command	OCF	Command Parameters	Return Parameters
HCI_Disconnect_Physical_Link	0x0037	Physical_Link_Handle, Reason	

Description:

When the Disconnect_Physical_Link command is received by an AMP Controller, the AMP Controller shall terminate the AMP physical link indicated by the Physical_Link_Handle parameter. The Reason parameter indicates the reason for ending the connection. The remote BR/EDR Controller will receive the Reason parameter in the Disconnection Physical Link Complete event.

All logical links on a physical link should be disconnected before the physical link is disconnected.

The Disconnect_Physical_Link command may be used to cancel the creation of a physical link, before the Physical Link Complete event for this link has been generated.

Command Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle identifying the physical link which has been created.

Reason: *Size: 1 Octet*

Value	Parameter Description
0x05, 0x13-0x16, 0x1A	Authentication Failure error code (0x05), Other End Terminated Connection error codes (0x13-0x16), Unsupported Remote Feature error code (0x1A), see Part D, Error Codes on page 339 for a list of error codes and descriptions.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the AMP Controller receives the Disconnect Physical Link command, a Command Status event shall be sent to the Host. The Disconnection Physical Link Complete event occurs at both Hosts when the termination of the connection has completed, and indicates that this command has been completed.

The Disconnect Physical Link command may be used to cancel the creation of a physical link. In such a case, the Physical Link Complete event for the corresponding Create Physical Link or Accept Physical Link command shall always be sent. The Physical Link Complete event shall be sent after the Disconnection Physical Link Complete event, in the case the Controller has already issued a Command Status event for the Disconnect Physical Link command, thus treating it as a creation cancel. If the cancellation was successful, the Physical Link Complete event shall be generated with the error code Unknown Connection Identifier (0x02).

Note: No Command Complete event will be sent by the AMP Controller to indicate that this command has been completed. Instead, the Disconnection Physical Link Complete event will indicate that this command has been completed.

7.1.40 Create Logical Link Command

Command	OCF	Command Parameters	Return Parameters
HCI_Create_Logical_Link	0x0038	Physical_Link_Handle, Tx_Flow_Spec, Rx_Flow_Spec	

Description:

When the Create_Logical_Link command is received by an AMP Controller, the AMP Controller shall create a logical connection to the remote AMP Controller identified by the physical link handle.

The Flow_Spec structures define the traffic requirements of the link. The Flow Spec ID values in the Tx/Rx parameters identify the logical link. Note: The Tx_Flow_Spec and Rx_Flow_Spec parameter only use 16 out of the 18 octets of the L2CAP Extended Flow Spec. The Type and Length fields from the L2CAP format are not included in the Tx_Flow_Spec and Rx_Flow_Spec parameters. A logical link can be created between two devices that are already joined by a physical link. In order to create a logical link the Create Logical Link command is called on one device (the initiating device) and the Accept Logical Link command is called on the other device (the responding device). The Flow_Spec information must match for the two calls (see [Vol 3] Part A, Section 5.6 and Section 9.1).

The AMP Controller may refuse to create the logical link if it has insufficient resources. If the AMP Controller is already in the process of creating one or more logical links, it may reject a Create Logical Link command with the error code "Busy" (see Part D, Section 2.55, Controller Busy (0X3A). When the Host receives this error code, it shall wait until a previous logical link creation completes before requesting another logical link.

The logical link must be completed before the Logical Link Accept Timeout expires on the local AMP.

Command Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical Link Handle over which the logical link is to be established.

Tx_Flow_Spec: *Size: 16 Octets*

Parameter Description
Extended Flow Specification value defining transmitted traffic. See [Vol 3] Part A, Section 5.6.

Rx_Flow_Spec: *Size: 16 Octets*

Parameter Description
Extended Flow Specification value defining received traffic. (See [Vol 3] Part A, Section 5.6).

Return Parameters:

None.

Event(s) generated (unless masked away):

When the AMP Controller receives the Create Logical Link command, the AMP Controller shall send the Command Status event to the Host. In addition, when the AMP Controller determines the logical link is established, the AMP Controller shall send a Logical Link Complete event to each Host. The Logical Link Complete event contains the Logical Link Handle, Physical Link Handle, and the corresponding flow spec ID if this command is successful.

Note: No Command Complete event will be sent by the AMP Controller to indicate that this command has been completed. Instead, the Logical Link Complete event will indicate that this command has been completed.

7.1.41 Accept Logical Link Command

Command	OCF	Command Parameters	Return Parameters
HCI_Accept_Logical_Link	0x0039	Physical_Link_Handle, Tx_Flow_Spec, Rx_Flow_Spec	

Description:

When the Accept_Logical_Link command is received by an AMP Controller, the AMP Controller shall attempt to accept a logical link with the remote AMP Controller identified by the Physical_Link_Handle.

The Flow_Spec structures define the traffic requirements of the link. The Flow Spec ID in the Tx/Rx parameter values identify the logical link.

A logical link may be created between two devices which are already joined by a physical link. In order to create a logical link the Create Logical Link command is called on one device (the initiating device) and the Accept Logical Link command is called on the other device (the responding device). The Flow_Spec information must match for the two calls. See [Vol 3] Part A, Section 5.6 and Section 9.1 for how this is achieved.

The AMP Controller may refuse to create the logical link if it has insufficient resources.

The logical link must be completed before the Logical Link Accept Timeout expires on the local AMP.

Command Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical Link Handle over which the logical link is to be established.

Tx_Flow_Spec: *Size: 16 Octets*

Parameter Description
Extended Flow Specification value defining transmitted traffic (see [Vol 3] Part A, Section 5.6, Extended Flow Specification Option)).

Rx_Flow_Spec: *Size: 16 Octets*

Parameter Description
Extended Flow Specification value defining received traffic (see [Vol 3] Part A, Section 5.6, Extended Flow Specification Option).

Return Parameters:

None.

Event(s) generated (unless masked away):

When the AMP Controller receives the Accept Logical Link command, the AMP Controller shall send the Command Status event to the Host. In addition, when the AMP Controller determines the logical link is established, it shall send a Logical Link Complete event to each Host. This action shall occur on both controllers that form the connection. The Logical Link Complete event contains the Logical Link Handle if this command is successful.

Note: No Command Complete event will be sent by the AMP Controller to indicate that this command has been completed. Instead, the Logical Link Complete event will indicate that this command has been completed.

7.1.42 Disconnect Logical Link Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Disconnect_Logical_Link	0x003A	Logical_Link_Handle	

Description:

When the Disconnect_Logical_Link command is received by an AMP Controller it shall terminate the designated logical link on the local Controller. The Logical_Link_Handle command parameter indicates which logical link is to be disconnected.

Command Parameters:

Logical_Link_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Logical_Link_Handle for the logical link being disconnected. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

None.

Event(s) generated (unless masked away):

When the AMP Controller receives the Disconnect Logical Link command it shall send the Command Status event to the Host. The Disconnection Logical Link Complete event shall be sent to the Host when the termination of the logical link has completed, and indicates that this command has been completed.

Note: No Command Complete event will be sent by the AMP Controller to indicate that this command has been completed. Instead, the Disconnection Logical Link Complete event will indicate that this command has been completed.

7.1.43 Logical Link Cancel Command

Command	OCF	Command Parameters	Return Parameters
HCI_Logical_Link_Cancel	0x003B	Physical_Link_Handle, Tx_Flow_Spec_ID	Status, Physical_Link_Handle, Tx_Flow_Spec_ID

Description:

When the Logical_Link_Cancel command is received by an AMP Controller, the Host shall request cancellation of the ongoing logical link creation process identified by the Physical_Link_Handle and the Tx_Flow_Spec_ID.

Command Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle over which the logical link is to be established.

Tx_Flow_Spec_ID: *Size: 1 Octet*

Value	Parameter Description
0xXX	Flow Spec ID identifying the logical link whose creation is being cancelled. This matches the ID field of the Tx_Flow_Spec in the matching Create or Accept Logical Link command.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Logical Link Cancel command succeeded
0x01-0xFF	Logical Link Cancel command failed. See Part D, Error Codes on page 339 for list of error codes.

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical Link Handle identifying the logical link creation being cancelled.

Tx_Flow_Spec_ID: *Size: 1 Octet*

Value	Parameter Description
0xXX	Flow Spec ID identifying the logical link creation being cancelled. This matches the ID field of the Tx_Flow_Spec in the matching Create or Accept Logical Link command.

Event(s) generated (unless masked away):

When the Logical Link Cancel command has completed, a Command Complete event shall be generated.

If the logical link is already established by the Controller, but the AMP Controller has not yet sent the Logical Link Complete event, then the local Controller shall detach the logical link and return a Command Complete event with the status "Success".

If the logical link is already established, and the Logical Link Complete event has been sent, then the AMP Controller shall return a Command Complete event with the error code ACL Connection already exists (0x0B).

If the Logical Link Cancel command is sent to the AMP Controller without a preceding Create or Accept Logical Link command to the same device, the Controller shall return a Command Complete event with the error code Unknown Connection Identifier (0x02).

The Logical Link Complete event for the corresponding Create Logical Link Command shall always be sent. The Logical Link Complete event shall be sent after the Command Complete event for the Logical Link Cancel command. If the cancellation was successful, the Logical Link Complete event shall be generated with the error code Unknown Connection Identifier (0x02).

7.1.44 Flow Spec Modify Command

Command	OCF	Command Parameters	Return Parameters
HCI_Flow_Spec_Modify	0x003C	Handle, Tx_Flow_Spec, Rx_Flow_Spec	

Description:

When the Flow_Spec_Modify command is received by an AMP Controller, the AMP Controller shall modify the Flow Spec of the logical connection identified by the Handle.

Matching commands will be issued on both devices. This ensures that both Controllers have full knowledge of the traffic requirements on this logical link, in both directions. Each Controller is responsible for meeting its own Tx Flow specification.

The Flow_Spec_Modify command cannot be used to change the Identifier or the Service Type fields of an existing Handle's Flow Spec.

Command Parameters:

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle if the receiving Controller is a BR/EDR Controller. Logical_Link_Handle, if the receiving Controller is an AMP Controller, identifying the logical link to be changed. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use

Tx_Flow_Spec: *Size: 16 Octets*

Parameter Description
Extended Flow Specification value defining transmitted traffic (see [Vol 3] Part A, Section 5.6, Extended Flow Specification Option).

Rx_Flow_Spec: *Size: 16 Octets*

Parameter Description
Extended Flow Specification value defining received traffic (see [Vol 3] Part A, Section 5.6).

Return Parameters:

None.

Event(s) generated (unless masked away):

When the AMP Controller receives the Flow Spec Modify command, the AMP Controller shall send the Command Status event to the Host. When each AMP Controller has completed the change a Flow Spec Modify Complete event shall be sent to the Host. If the modify fails, the controllers shall revert to previously specified values.

Note: No Command Complete event will be sent by the AMP Controller to indicate that this command has been completed. Instead, the Flow Spec Modify Complete event will indicate that this command has been completed.

7.2 LINK POLICY COMMANDS

The Link Policy Commands provide methods for the Host to affect how the Link Manager manages the piconet. When Link Policy Commands are used, the LM still controls how Bluetooth piconets and scatternets are established and maintained, depending on adjustable policy parameters. These policy commands modify the Link Manager behavior that can result in changes to the link layer connections with Bluetooth remote devices.

Note: Only one ACL connection can exist between two BR/EDR Controllers, and therefore there can only be one ACL HCI Connection_Handle for each physical link layer Connection. The BR/EDR Controller provides policy adjustment mechanisms to provide support for a number of different policies. This capability allows one Bluetooth module to be used to support many different usage models, and the same Bluetooth module can be incorporated in many different types of BR/EDR Controllers.

For the Link Policy Commands, the OGF is defined as 0x02.

7.2.1 Hold Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Hold_Mode	0x0001	Connection_Handle, Hold_Mode_Max_Interval, Hold_Mode_Min_Interval	

Description:

The Hold_Mode command is used to alter the behavior of the Link Manager, and have it place the ACL baseband connection associated by the specified Connection_Handle into the hold mode. The Hold_Mode_Max_Interval and Hold_Mode_Min_Interval command parameters specify the length of time the Host wants to put the connection into the hold mode. The local and remote devices will negotiate the length in the hold mode. The Hold_Mode_Max_ Interval parameter is used to specify the maximum length of the Hold interval for which the Host may actually enter into the hold mode after negotiation with the remote device. The Hold interval defines the amount of time between when the Hold Mode begins and when the Hold Mode is completed. The Hold_ Mode_Min_Interval parameter is used to specify the minimum length of the Hold interval for which the Host may actually enter into the hold mode after the negotiation with the remote device. Therefore the Hold_Mode_Min_Interval cannot be greater than the Hold_Mode_Max_Interval. The BR/EDR Controller will return the actual Hold interval in the Interval parameter of the Mode Change event, if the command is successful. This command enables the Host to support a low-power policy for itself or several other BR/EDR Controllers, and allows the devices to enter Inquiry Scan, Page Scan, and a number of other possible actions.

Note: The Connection_Handle cannot be of the SCO or eSCO link type
If the Host sends data to the BR/EDR Controller with a Connection_Handle corresponding to a connection in hold mode, the BR/EDR Controller will keep the data in its buffers until either the data can be transmitted (the hold mode has ended) or a flush, a flush timeout or a disconnection occurs. This is valid even if the Host has not yet been notified of the hold mode through a Mode Change event when it sends the data.

Note: The above is not valid for an HCI Data Packet sent from the Host to the BR/EDR Controller on the master side where the Connection_Handle is a Connection_Handle used for broadcast and the Broadcast_Flag is set to Active Broadcast or Piconet Broadcast. The broadcast data will then never be received by slaves in hold mode.

The Hold_Mode_Max_Interval shall be less than the Link Supervision Timeout configuration parameter.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Hold_Mode_Max_Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Maximum acceptable number of Baseband slots to wait in Hold Mode. Time Length of the Hold = N * 0.625 msec (1 Baseband slot) Range for N: 0x0002-0xFFFE; only even values are valid. Time Range: 1.25ms - 40.9 sec Mandatory Range: 0x0014 to 0x8000

Hold_Mode_Min_Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Minimum acceptable number of Baseband slots to wait in Hold Mode. Time Length of the Hold = N * 0.625 msec (1 Baseband slot) Range for N: 0x0002-0xFF00; only even values are valid Time Range: 1.25 msec - 40.9 sec Mandatory Range: 0x0014 to 0x8000

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Hold_Mode command, the BR/EDR Controller shall send the Command Status event to the Host. The Mode Change event shall occur when the Hold Mode has started and the Mode Change event shall occur again when the Hold Mode has completed for the specified Connection_Handle. The Mode Change event signaling the end of the Hold Mode is an estimation of the hold mode ending if the event is for a remote BR/EDR Controller.

Note: No Command Complete event will be sent by the Controller to indicate that this command has been completed. Instead, only the Mode Change event will indicate that this command has been completed. If an error occurs after the Command Status event has occurred, then the status in the Mode Change event shall indicate the error.

7.2.2 Sniff Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Sniff_Mode	0x0003	Connection_Handle, Sniff_Max_Interval, Sniff_Min_Interval, Sniff_Attempt, Sniff_Timeout	

Description:

The Sniff_Mode command is used to alter the behavior of the Link Manager and have it place the ACL baseband connection associated with the specified Connection_Handle into the sniff mode. The Connection_Handle command parameter is used to identify which ACL link connection is to be placed in sniff mode. The Sniff_Max_Interval and Sniff_Min_Interval command parameters are used to specify the requested acceptable maximum and minimum periods in the Sniff Mode. The Sniff_Min_Interval shall not be greater than the Sniff_Max_Interval. The sniff interval defines the amount of time between each consecutive sniff period. The BR/EDR Controller will return the actual sniff interval in the Interval parameter of the Mode Change event, if the command is successful. For a description of the meaning of the Sniff_Attempt and Sniff_Timeout parameters, see Baseband Specification, Section 8.7, on page 180. Sniff_Attempt is there called $N_{sniff\ attempt}$ and Sniff_Timeout is called $N_{sniff\ timeout}$. This command enables the Host to support a low-power policy for itself or several other BR/EDR Controllers, and allows the devices to enter Inquiry Scan, Page Scan, and a number of other possible actions.

Note: In addition, the Connection_Handle cannot be one of the synchronous link types. If the Host sends data to the BR/EDR Controller with a Connection_Handle corresponding to a connection in sniff mode, the BR/EDR Controller will keep the data in its buffers until either the data can be transmitted or a flush, a flush timeout or a disconnection occurs. This is valid even if the Host has not yet been notified of the sniff mode through a Mode Change event when it sends the data. Note that it is possible for the master to transmit data to a slave without exiting sniff mode (see description in Baseband Specification, Section 8.7, on page 180).

Note: The above is not valid for an HCI Data Packet sent from the Host to the BR/EDR Controller on the master side where the Connection_Handle is a Connection_Handle used for broadcast and the Broadcast_Flag is set to Active Broadcast or Piconet Broadcast. In that case, the broadcast data will only be received by a slave in sniff mode if that slave happens to listen to the master when the broadcast is made.

The Sniff_Max_Interval shall be less than the Link Supervision Timeout configuration parameter.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Sniff_Max_Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Range: 0x0002 to 0xFFFE; only even values are valid Mandatory Range: 0x0006 to 0x0540 Time = N * 0.625 msec Time Range: 1.25 msec to 40.9 sec

Sniff_Min_Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Range: 0x0002 to 0xFFFE; only even values are valid Mandatory Range: 0x0006 to 0x0540 Time = N * 0.625 msec Time Range: 1.25 msec to 40.9 sec

Sniff_Attempt: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Number of Baseband receive slots for sniff attempt. Length = N* 1.25 msec Range for N: 0x0001 – 0x7FFF Time Range: 1.25msec - 40.9 sec Mandatory Range for Controller: 1 to $T_{sniff}/2$

Sniff_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Number of Baseband receive slots for sniff timeout. Length = N * 1.25 msec Range for N: 0x0000 – 0x7FFF Time Range: 0 msec - 40.9 sec Mandatory Range for Controller: 0 to 0x0028

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Sniff_Mode command, the BR/EDR Controller shall send the Command Status event to the Host. The Mode Change event shall occur when the Sniff Mode has started for the specified Connection_Handle.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead only the Mode Change event will indicate that this command has been completed. If an error occurs after the Command Status event has occurred, then the status in the Mode Change event shall indicate the error.

7.2.3 Exit Sniff Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Exit_Sniff_Mode	0x0004	Connection_Handle	

Description:

The Exit_Sniff_Mode command is used to end the sniff mode for a Connection_Handle, which is currently in sniff mode. The Link Manager will determine and issue the appropriate LMP commands to remove the sniff mode for the associated Connection_Handle.

Note: The Connection_Handle cannot be one of the synchronous link types.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

None.

Event(s) generated (unless masked away):

When BR/EDR Controller receives the Exit_Sniff_Mode command, the BR/EDR Controller shall send the Command Status event to the Host. The Mode Change event shall occur when the Sniff Mode has ended for the specified Connection_Handle.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, only the Mode Change event will indicate that this command has been completed.

7.2.4 Park State Command

Command	OCF	Command Parameters	Return Parameters
HCI_Park_State	0x0005	Connection_Handle, Beacon_Max_Interval, Beacon_Min_Interval	

Description:

The Park_State command is used to alter the behavior of the Link Manager, and have the LM place the baseband connection associated by the specified Connection_Handle into Park state. The Connection_Handle command parameter is used to identify which connection is to be placed in Park state. The Connection_Handle must be a Connection_Handle for an ACL connection. The Beacon Interval command parameters specify the acceptable length of the interval between beacons. However, the remote device may request shorter interval. The Beacon_Max_Interval parameter specifies the acceptable longest length of the interval between beacons. The Beacon_Min_Interval parameter specifies the acceptable shortest length of the interval between beacons. Therefore, the Beacon Min Interval cannot be greater than the Beacon Max Interval. The BR/EDR Controller will return the actual Beacon interval in the Interval parameter of the Mode Change event, if the command is successful. This command enables the Host to support a low-power policy for itself or several other BR/EDR Controllers, allows the devices to enter Inquiry Scan, Page Scan, provides support for large number of BR/EDR Controllers in a single piconet, and a number of other possible activities.

Note: When the Host issues the Park State command, no Connection_Handles for synchronous connections are allowed to exist to the remote device that is identified by the Connection_Handle parameter. If one or more Connection_Handles for synchronous connections exist to that device, depending on the implementation, a Command Status event or a Mode Change event (following a Command Status event where Status=0x00) will be returned with the error code 0x0C *Command Disallowed*.

If the Host sends data to the BR/EDR Controller with a Connection_Handle corresponding to a parked connection, the BR/EDR Controller will keep the data in its buffers until either the data can be transmitted (after unpark) or a flush, a flush timeout or a disconnection occurs. This is valid even if the Host has not yet been notified of park state through a Mode Change event when it sends the data.

Note: The above is not valid for an HCI Data Packet sent from the Host to the BR/EDR Controller on the master side where the Connection_Handle is a Connection_Handle used for Piconet Broadcast and the Broadcast_Flag is set to Piconet Broadcast. In that case, slaves in park state will also receive the broadcast data. (If the Broadcast_Flag is set to Active Broadcast, the broadcast data will usually not be received by slaves in park state.)
It is possible for the Controller to do an automatic unpark to transmit data and then park the connection again depending on the value of the Link_Policy_Settings

parameter (see Write_Link_Policy_Settings) and depending on whether the implementation supports this or not (optional feature). The optional feature of automatic unpark/park can also be used for link supervision. Whether Mode Change events are returned or not at automatic unpark/park if this is implemented, is vendor specific. This could be controlled by a vendor specific HCI command.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Beacon_Max_Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Range: 0x000E to 0xFFFE; only even values are valid Mandatory Range: 0x000E to 0x1000 Time = N * 0.625 msec Time Range: 8.75 msec to 40.9 sec

Beacon_Min_Interval *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Range: 0x000E to 0xFFFE; only even values are valid Mandatory Range: 0x000E to 0x1000 Time = N * 0.625 msec Time Range: 8.75 msec to 40.9 sec

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Park_State command, the BR/EDR Controller shall send the Command Status event to the Host . The Mode Change event shall occur when the Park State has started for the specified Connection_Handle.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, only the Mode Change event will indicate that this command has been completed. If an error occurs after the Command Status event has occurred, then the status in the Mode Change event shall indicate the error.

7.2.5 Exit Park State Command

Command	OCF	Command Parameters	Return Parameters
HCI_Exit_Park_State	0x0006	Connection_Handle	

Description:

The Exit_Park_State command is used to switch the BR/EDR Controller from park state back to the active mode. This command may only be issued when the device associated with the specified Connection_Handle is in park state. The Connection_Handle must be a Connection_Handle for an ACL connection. This function does not complete immediately.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Exit_Park_State command, the BR/EDR Controller shall send the Command Status event to the Host. The Mode Change event shall occur when park state has ended for the specified Connection_Handle.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, only the Mode Change event will indicate that this command has been completed.

7.2.6 QoS Setup Command

Command	OCF	Command Parameters	Return Parameters
HCI_QoS_Setup	0x0007	Connection_Handle, Flags, Service_Type, Token_Rate, Peak_Bandwidth, Latency, Delay_Variation	

Description:

The QoS_Setup command is used to specify Quality of Service parameters for a Connection_Handle. The Connection_Handle must be a Connection_Handle for an ACL connection. These QoS parameter are the same parameters as L2CAP QoS. For more detail see [Vol 3] Part A, Logical Link Control and Adaptation Protocol Specification. This allows the Link Manager to have all of the information about what the Host is requesting for each connection. The LM will determine if the QoS parameters can be met. BR/EDR Controllers that are both slaves and masters can use this command. When a device is a slave, this command will trigger an LMP request to the master to provide the slave with the specified QoS as determined by the LM. When a device is a master, this command is used to request a slave device to accept the specified QoS as determined by the LM of the master. The Connection_Handle command parameter is used to identify for which connection the QoS request is requested.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify which connection for the QoS Setup. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Flags: *Size: 1 Octet*

Value	Parameter Description
0x00 – 0xFF	Reserved for future use.

Service_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	No Traffic.
0x01	Best Effort.
0x02	Guaranteed.
0x03-0xFF	Reserved for future use.

Token_Rate: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Token Rate in octets per second.

Peak_Bandwidth: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Peak Bandwidth in octets per second.

Latency: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Latency in microseconds.

Delay_Variation: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Delay Variation in microseconds.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the QoS_Setup command, the BR/EDR Controller shall send the Command Status event to the Host. When the Link Manager has completed the LMP messages to establish the requested QoS parameters, the BR/EDR Controller shall send a QoS Setup Complete event to the Host, and the event may also be generated on the remote side if there was LMP negotiation. The values of the parameters of the QoS Setup Complete event may, however, be different on the initiating and the remote side. The QoS Setup Complete event returned by the BR/EDR Controller on the local side contains the status of this command, and returned QoS parameters describing the supported QoS for the connection.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the QoS Setup Complete event will indicate that this command has been completed.

7.2.7 Role Discovery Command

Command	OCF	Command Parameters	Return Parameters
HCI_Role_Discovery	0x0009	Connection_Handle	Status, Connection_Handle, Current_Role

Description:

The Role_Discovery command is used for a Host to determine which role the device is performing for a particular Connection_Handle. The Connection_Handle must be a Connection_Handle for an ACL connection.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Role_Discovery command succeeded
0x01-0xFF	Role_Discovery command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify which connection the Role_Discovery command was issued on. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Current_Role: *Size: 1 Octet*

Value	Parameter Description
0x00	Current Role is Master for this Connection_Handle.
0x01	Current Role is Slave for this Connection_Handle.

Event(s) generated (unless masked away):

When the Role_Discovery command has completed, a Command Complete event shall be generated.

7.2.8 Switch Role Command

Command	OCF	Command Parameters	Return Parameters
HCI_Switch_Role	0x000B	BD_ADDR, Role	

Description:

The Switch_Role command is used to switch the current BR/EDR role the device is performing for a particular connection with another specified BR/EDR Controller. The BD_ADDR command parameter indicates for which connection the role switch is to be performed. The Role parameter indicates the requested new role that the local device performs.

Note: The BD_ADDR command parameter must specify a BR/EDR Controller for which a connection already exists.

Note: If there is an SCO connection between the local device and the device identified by the BD_ADDR parameter, an attempt to perform a role switch shall be rejected by the local device.

Note: If the connection between the local device and the device identified by the BD_ADDR parameter is placed in Sniff Mode, an attempt to perform a role switch will be rejected by the local device.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for the connected device with which a role switch is to be performed.

Role: *Size: 1 Octet*

Value	Parameter Description
0x00	Change own Role to Master for this BD_ADDR.
0x01	Change own Role to Slave for this BD_ADDR.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the BR/EDR Controller receives the Switch_Role command, the BR/EDR Controller shall send the Command Status event to the Host. When the role switch is performed, a Role Change event shall occur to indicate that the roles have been changed, and will be communicated to both Hosts.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, only the Role Change event will indicate that this command has been completed.

7.2.9 Read Link Policy Settings Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Link_Policy_Settings	0x000C	Connection_Handle	Status, Connection_Handle, Link_Policy_Settings

Description:

This command will read the Link Policy setting for the specified Connection_Handle. The Connection_Handle shall be a Connection_Handle for an ACL connection. Section 6.18 on page 443.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Link_Policy_Settings command succeeded.
0x01-0xFF	Read_Link_Policy_Settings command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Link_Policy_Settings *Size: 2 Octets*

Value	Parameter Description
0x0000	Disable All LM Modes Default.
0x0001	Enable Role Switch.
0x0002	Enable Hold Mode.
0x0004	Enable Sniff Mode.
0x0008	Enable Park State.

0x0010 – 0x8000	Reserved for future use.

Event(s) generated (unless masked away):

When the Read_Link_Policy_Settings command has completed, a Command Complete event shall be generated.

7.2.10 Write Link Policy Settings Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Link_Policy_Settings	0x000D	Connection_Handle, Link_Policy_Settings	Status, Connection_Handle

Description:

This command writes the Link Policy setting for the specified Connection_Handle. The Connection_Handle shall be a Connection_Handle for an ACL connection. See Section 6.18 on page 443.

The default value is the value set by the Write Default Link Policy Settings Command.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Link_Policy_Settings *Size: 2 Octets*

Value	Parameter Description
0x0000	Disable All LM Modes.
0x0001	Enable Role Switch.
0x0002	Enable Hold Mode.
0x0004	Enable Sniff Mode.
0x0008	Enable Park State.
0x0010 – 0x8000	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Link_Policy_Settings command succeeded.
0x01-0xFF	Write_Link_Policy_Settings command failed. See Part D, Error Codes on page 339 for error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Event(s) generated (unless masked away):

When the Write_Link_Policy_Settings command has completed, a Command Complete event shall be generated.

7.2.11 Read Default Link Policy Settings Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Default_Link_Policy_Settings	0x000E		Status, Default_Link_Policy_Settings

Description:

This command reads the Default Link Policy setting for all new BR/EDR connections.

Note: See the Link Policy Settings configuration parameter for more information. See Section 6.18 on page 443.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Link_Policy_Settings command succeeded
0x01-0xFF	Read_Link_Policy_Settings command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Default_Link_Policy_Settings *Size: 2 Octets*

Value	Parameter Description
0x0000	Disable All LM Modes Default
0x0001	Enable Role Switch
0x0002	Enable Hold Mode
0x0004	Enable Sniff Mode
0x0008	Enable Park State
0x0010 – 0x8000	Reserved for future use.

Event(s) generated (unless masked away):

When the Read_Default_Link_Policy_Settings command has completed, a Command Complete event shall be generated.

7.2.12 Write Default Link Policy Settings Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Default_Link_Policy_Settings	0x000F	Default_Link_Policy_Settings	Status

Description:

This command writes the Default Link Policy configuration value. The Default_Link_Policy_Settings parameter determines the initial value of the Link_Policy_Settings for all new BR/EDR connections.

Note: See the Link Policy Settings configuration parameter for more information. See Section 6.18 on page 443.

Command Parameters:

Default_Link_Policy_Settings *Size: 2 Octets*

Value	Parameter Description
0x0000	Disable All LM Modes Default
0x0001	Enable Role Switch
0x0002	Enable Hold Mode
0x0004	Enable Sniff Mode
0x0008	Enable Park State
0x0010 – 0x8000	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Link_Policy_Settings command succeeded
0x01-0xFF	Write_Link_Policy_Settings command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Default_Link_Policy_Settings command has completed, a Command Complete event will be generated.

7.2.13 Flow Specification Command

Command	OCF	Command Parameters	Return Parameters
HCI_Flow_Specification	0x0010	Connection_Handle, Flags, Flow_direction, Service_Type, Token_Rate, Token_Bucket_Size, Peak_Bandwidth, Access Latency	

Description:

The Flow_Specification command is used to specify the flow parameters for the traffic carried over the ACL connection identified by the Connection_Handle. The Connection_Handle must be a Connection_Handle for an ACL connection. The Connection_Handle command parameter is used to identify for which connection the Flow Specification is requested. The flow parameters refer to the outgoing or incoming traffic of the ACL link, as indicated by the Flow_direction field. The Flow Specification command allows the Link Manager to have the parameters of the outgoing as well as the incoming flow for the ACL connection. The flow parameters are defined in the L2CAP specification [Vol 3] Part A, Section 5.3, Quality of Service (QoS) Option. The Link Manager will determine if the flow parameters can be supported. BR/EDR Controllers that are both master and slave can use this command.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle used to identify for which ACL connection the Flow is specified. Range: 0x0000 - 0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Flags: *Size: 1 Octet*

Value	Parameter Description
0x00 – 0xFF	Reserved for future use.

Flow_direction: *Size: 1 Octet*

Value	Parameter Description
0x00	Outgoing Flow i.e. traffic send over the ACL connection
0x01	Incoming Flow i.e. traffic received over the ACL connection
0x02 – 0xFF	Reserved for future use.

Service_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	No Traffic
0x01	Best Effort
0x02	Guaranteed
0x03 – 0xFF	Reserved for future use

Token Rate: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Token Rate in octets per second

Token Bucket Size: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Token Bucket Size in octets

Peak_Bandwidth: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Peak Bandwidth in octets per second

Access Latency: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Latency in microseconds

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Controller receives the Flow_Specification command, the BR/EDR Controller shall send the Command Status event to the Host. When the Link Manager has determined if the Flow specification can be supported, the BR/EDR Controller on the local BR/EDR Controller shall send a Flow Specification Complete event to the Host. The Flow Specification Complete event returned by the Controller on the local side contains the status of this command, and returned Flow parameters describing the supported QoS for the ACL connection.

Note: No Command Complete event will be sent by the BR/EDR Controller to indicate that this command has been completed. Instead, the Flow Specification Complete event will indicate that this command has been completed.

7.2.14 Sniff Subrating Command

Command	OCF	Command Parameters	Return Parameters
HCI_Sniff_ Subrating	0x0011	Connection Handle, Maximum_Latency, Minimum_Remote_Timeout, Minimum_Local_Timeout	Status, Connection_Handle

Description:

The Sniff_Subrating command specifies the parameters for sniff subrating for a given link. The interval shall be determined from the sniff interval and the maximum subrate latency parameters from the command. The link may have smaller subrates and therefore lower latencies and longer timeouts than those specified. When the sniff subrate has been exchanged a Sniff Subrating event shall be generated. If this command is used on a link in sniff mode this shall cause sniff subrating to be negotiated at the Link Manager, otherwise sniff subrating shall be negotiated only after the device has entered the sniff mode.

The Connection_Handle shall be the primary Connection_Handle between the two devices.

The Maximum Latency parameter shall define the maximum allowed sniff subrate of the remote device.

Note: If the Host does not write the sniff subrating parameters prior to sniff subrating being initiated by the Link Manager the default values shall be used.

Note: Setting both subrate values to zero is equivalent to sniff mode without subrating enabled.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000 – 0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Maximum_Latency: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	The Maximum Latency parameter shall be used to calculate the maximum_sniff subrate that the remote device may use. Default: 0x0000 Latency = N * 0.625 msec (1 Baseband slot) Range for N: 0x0000 – 0xFFFE Time Range: 0 sec – 40.9 sec

Minimum_Remote_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Minimum base sniff subrate timeout that the remote device may use Default: 0x0000 Latency = N * 0.625 msec (1 Baseband slot) Range for N: 0x0000 – 0xFFFE Time Range: 0 sec - 40.9 sec

Minimum_Local_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Minimum base sniff subrate timeout that the local device may use. Default: 0x0000 Latency = N * 0.625 msec (1 Baseband slot) Range for N: 0x0000 – 0xFFFE Time Range: 0 sec - 40.9 sec

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	The Sniff Subrating command succeeded.
0x01 - 0xFF	Sniff Subrating command failed. See Part D, Error Codes for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000 - 0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Event(s) generated (unless masked away)

When the Sniff_Subrating command has been received by the BR/EDR Controller, a Command Complete event shall be generated.

A Sniff_Subrating event shall occur when the sniff subrating has been negotiated for the specified Connection_Handle.

7.3 CONTROLLER & BASEBAND COMMANDS

The Controller & Baseband Commands provide access and control to various capabilities of the Bluetooth hardware. These parameters provide control of BR/EDR Controllers and of the capabilities of the Link Manager and Baseband

in the BR/EDR Controller, the PAL in an AMP Controller, and the Link Layer in an LE Controller. The Host can use these commands to modify the behavior of the local Controller.

For the HCI Control and Baseband Commands, the OGF is defined as 0x03.

7.3.1 Set Event Mask Command

Command	OCF	Command Parameters	Return Parameters
HCI_Set_Event_Mask	0x0001	Event_Mask	Status

Description:

The Set_Event_Mask command is used to control which events are generated by the HCI for the Host. If the bit in the Event_Mask is set to a one, then the event associated with that bit will be enabled. For an LE Controller, the “LE Meta Event” bit in the Event_Mask shall enable or disable all LE events in the LE Meta Event (see Section 7.7.65). The Host has to deal with each event that occurs. The event mask allows the Host to control how much it is interrupted.

Command Parameters:

Event_Mask: *Size: 8 Octets*

Value	Parameter Description
0x0000000000000000	No events specified
0x0000000000000001	Inquiry Complete Event
0x0000000000000002	Inquiry Result Event
0x0000000000000004	Connection Complete Event
0x0000000000000008	Connection Request Event
0x0000000000000010	Disconnection Complete Event
0x0000000000000020	Authentication Complete Event
0x0000000000000040	Remote Name Request Complete Event
0x0000000000000080	Encryption Change Event
0x0000000000000100	Change Connection Link Key Complete Event
0x0000000000000200	Master Link Key Complete Event
0x0000000000000400	Read Remote Supported Features Complete Event
0x0000000000000800	Read Remote Version Information Complete Event
0x0000000000001000	QoS Setup Complete Event
0x0000000000002000	Reserved
0x0000000000004000	Reserved

Value	Parameter Description
0x0000000000008000	Hardware Error Event
0x0000000000010000	Flush Occurred Event
0x0000000000020000	Role Change Event
0x0000000000040000	Reserved
0x0000000000080000	Mode Change Event
0x0000000000100000	Return Link Keys Event
0x0000000000200000	PIN Code Request Event
0x0000000000400000	Link Key Request Event
0x0000000000800000	Link Key Notification Event
0x0000000001000000	Loopback Command Event
0x0000000002000000	Data Buffer Overflow Event
0x0000000004000000	Max Slots Change Event
0x0000000008000000	Read Clock Offset Complete Event
0x0000000010000000	Connection Packet Type Changed Event
0x0000000020000000	QoS Violation Event
0x0000000040000000	Page Scan Mode Change Event [deprecated]
0x0000000080000000	Page Scan Repetition Mode Change Event
0x0000000100000000	Flow Specification Complete Event
0x0000000200000000	Inquiry Result with RSSI Event
0x0000000400000000	Read Remote Extended Features Complete Event
0x0000000800000000	Reserved
0x0000001000000000	Reserved
0x0000002000000000	Reserved
0x0000004000000000	Reserved
0x0000008000000000	Reserved
0x0000010000000000	Reserved
0x0000020000000000	Reserved
0x0000040000000000	Reserved
0x0000080000000000	Synchronous Connection Complete Event
0x0000100000000000	Synchronous Connection Changed Event
0x0000200000000000	Sniff Subrating Event
0x0000400000000000	Extended Inquiry Result Event

Value	Parameter Description
0x0000800000000000	Encryption Key Refresh Complete Event
0x0001000000000000	IO Capability Request Event
0x0002000000000000	IO Capability Request Reply Event
0x0004000000000000	User Confirmation Request Event
0x0008000000000000	User Passkey Request Event
0x0010000000000000	Remote OOB Data Request Event
0x0020000000000000	Simple Pairing Complete Event
0x0040000000000000	Reserved
0x0080000000000000	Link Supervision Timeout Changed Event
0x0100000000000000	Enhanced Flush Complete Event
0x0200000000000000	Reserved
0x0400000000000000	User Passkey Notification Event
0x0800000000000000	Keypress Notification Event
0x1000000000000000	Remote Host Supported Features Notification Event
0x2000000000000000	LE Meta-Event
0xC000000000000000	Reserved for future use
0x00 00 1F FF FF FF FF FF	Default

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Set_Event_Mask command succeeded.
0x01-0xFF	Set_Event_Mask command failed. See Part D, Error Codes for error codes and descriptions.

Event(s) generated (unless masked away):

When the Set_Event_Mask command has completed, a Command Complete event shall be generated.

7.3.2 Reset Command

Command	OCF	Command Parameters	Return Parameters
HCI_Reset	0x0003		Status

Description:

The Reset command will reset the Controller and the Link Manager on the BR/EDR Controller, the PAL on an AMP Controller, or the Link Layer on an LE Controller. If the Controller supports both BR/EDR and LE then the Reset command shall reset the Link Manager, Baseband and Link Layer. The Reset command shall not affect the used HCI transport layer since the HCI transport layers may have reset mechanisms of their own. After the reset is completed, the current operational state will be lost, the Controller will enter standby mode and the Controller will automatically revert to the default values for the parameters for which default values are defined in the specification.

Note: The Reset command will not necessarily perform a hardware reset. This is implementation defined. On an AMP Controller, the Reset command shall reset the service provided at the logical HCI to its initial state , but beyond this the exact effect on the Controller device is implementation defined and should not interrupt the service provided to other protocol stacks.

The Host shall not send additional HCI commands before the Command Complete event related to the Reset command has been received.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Reset command succeeded, was received and will be executed.
0x01-0xFF	Reset command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the reset has been performed, a Command Complete event shall be generated.

7.3.3 Set Event Filter Command

Command	OCF	Command Parameters	Return Parameters
HCI_Set_Event_Filter	0x0005	Filter_Type, Filter_Condition_Type, Condition	Status

Description:

The Set_Event_Filter command is used by the Host to specify different event filters. The Host may issue this command multiple times to request various conditions for the same type of event filter and for different types of event filters. The event filters are used by the Host to specify items of interest, which allow the BR/EDR Controller to send only events which interest the Host. Only some of the events have event filters. By default (before this command has been issued after power-on or Reset) no filters are set, and the Auto_Accept_Flag is off (incoming connections are not automatically accepted). An event filter is added each time this command is sent from the Host and the Filter_Condition_Type is not equal to 0x00. (The old event filters will not be overwritten). To clear all event filters, the Filter_Type = 0x00 is used. The Auto_Accept_Flag will then be set to off. To clear event filters for only a certain Filter_Type, the Filter_Condition_Type = 0x00 is used.

The Inquiry Result filter allows the BR/EDR Controller to filter out Inquiry Result events. The Inquiry Result filter allows the Host to specify that the BR/EDR Controller only sends Inquiry Results to the Host if the Inquiry Result event meets one of the specified conditions set by the Host. For the Inquiry Result filter, the Host can specify one or more of the following Filter Condition Types:

1. Return responses from all devices during the Inquiry process
2. A device with a specific Class of Device responded to the Inquiry process
3. A device with a specific BD_ADDR responded to the Inquiry process

The Inquiry Result filter is used in conjunction with the Inquiry and Periodic Inquiry command.

The Connection Setup filter allows the Host to specify that the Controller only sends a Connection Complete or Connection Request event to the Host if the event meets one of the specified conditions set by the Host. For the Connection Setup filter, the Host can specify one or more of the following Filter Condition Types:

1. Allow Connections from all devices
2. Allow Connections from a device with a specific Class of Device
3. Allow Connections from a device with a specific BD_ADDR

For each of these conditions, an Auto_Accept_Flag parameter allows the Host to specify what action should be done when the condition is met. The Auto_ Accept_Flag allows the Host to specify if the incoming connection should be auto accepted (in which case the BR/EDR Controller will send the Connection Complete event to the Host when the connection is completed) or if the Host should make the decision (in which case the BR/EDR Controller will send the Connection Request event to the Host, to elicit a decision on the connection).

The Connection Setup filter is used in conjunction with the Read/Write_ Scan_Enable commands. If the local device is in the process of a page scan, and is paged by another device which meets one on the conditions set by the Host, and the Auto_Accept_Flag is off for this device, then a Connection Request event will be sent to the Host by the BR/EDR Controller. A Connection Complete event will be sent later on after the Host has responded to the incoming connection attempt. In this same example, if the Auto_Accept_Flag is on, then a Connection Complete event will be sent to the Host by the Controller. (No Connection Request event will be sent in that case.)

The BR/EDR Controller will store these filters in volatile memory until the Host clears the event filters using the Set_Event_Filter command or until the Reset command is issued. The number of event filters the BR/EDR Controller can store is implementation dependent. If the Host tries to set more filters than the BR/EDR Controller can store, the BR/EDR Controller will return the *Memory Full* error code and the filter will not be installed.

Note: The Clear All Filters has no Filter Condition Types or Conditions.

Note: In the condition that a connection is auto accepted, a Link Key Request event and possibly also a PIN Code Request event and a Link Key Notification event could be sent to the Host by the Controller before the Connection Complete event is sent.

If there is a contradiction between event filters, the latest set event filter will override older ones. An example is an incoming connection attempt where more than one Connection Setup filter matches the incoming connection attempt, but the Auto-Accept_Flag has different values in the different filters.

Command Parameters:

Filter_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Clear All Filters (Note: In this case, the Filter_Condition_Type and Condition parameters should not be given, they should have a length of 0 octets. Filter_Type should be the only parameter.)
0x01	Inquiry Result.
0x02	Connection Setup.
0x03-0xFF	Reserved for future use.

Filter Condition Types: For each Filter Type one or more Filter Condition types exists.

Inquiry_Result_Filter_Condition_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Return responses from all devices during the Inquiry process. (Note: A device may be reported to the Host in an Inquiry Result event more than once during an inquiry or inquiry period depending on the implementation, see description in Section 7.1.1 on page 459 and Section 7.1.3 on page 462)
0x01	A device with a specific Class of Device responded to the Inquiry process.
0x02	A device with a specific BD_ADDR responded to the Inquiry process.
0x03-0xFF	Reserved for future use

Connection_Setup_Filter_Condition_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Allow Connections from all devices.
0x01	Allow Connections from a device with a specific Class of Device.
0x02	Allow Connections from a device with a specific BD_ADDR.
0x03-0xFF	Reserved for future use.

Condition: For each Filter Condition Type defined for the Inquiry Result Filter and the Connection Setup Filter, zero or more Condition parameters are required – depending on the filter condition type and filter type.

Condition for Inquiry_Result_Filter_Condition_Type = 0x00

Condition: *Size: 0 Octet*

Value	Parameter Description
	The Condition parameter is not used.

Condition for Inquiry_Result_Filter_Condition_Type = 0x01

Condition:

Size: 6 Octets

Class_of_Device: *Size: 3 Octets*

Value	Parameter Description
0x000000	Default, Return All Devices.
0xXXXXXX	*Class of Device* of Interest.

Class_of_Device_Mask: *Size: 3 Octets*

Value	Parameter Description
0xXXXXXX	Bit Mask used to determine which bits of the Class of Device parameter are 'don't care'. Zero-value bits in the mask indicate the 'don't care' bits of the Class of Device.

Condition for Inquiry_Result_Filter_Condition_Type = 0x02

Condition:

Size: 6 Octets

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device of Interest

Condition for Connection_Setup_Filter_Condition_Type = 0x00

Condition:

Size: 1 Octet

Auto_Accept_Flag: *Size:1 Octet*

Value	Parameter Description
0x01	Do NOT Auto accept the connection. (Auto accept is off)
0x02	Do Auto accept the connection with role switch disabled. (Auto accept is on).
0x03	Do Auto accept the connection with role switch enabled. (Auto accept is on). Note: When auto accepting an incoming synchronous connection, no role switch will be performed. The value 0x03 of the Auto_Accept_Flag will then get the same effect as if the value had been 0x02.
0x04 – 0xFF	Reserved for future use.

Condition for Connection_Setup_Filter_Condition_Type = 0x01

Condition:

Size: 7 Octets

Class_of_Device: *Size: 3 Octets*

Value	Parameter Description
0x000000	Default, Return All Devices.

0xXXXXXX	*Class of Device* of Interest.

Class_of_Device_Mask: *Size: 3 Octets*

Value	Parameter Description
0xXXXXXX	Bit Mask used to determine which bits of the Class of Device parameter are 'don't care'. Zero-value bits in the mask indicate the 'don't care' bits of the Class of Device. Note: For an incoming SCO connection, if the class of device is unknown then the connection will be accepted.

Auto_Accept_Flag: *Size: 1 Octet*

Value	Parameter Description
0x01	Do NOT Auto accept the connection. (Auto accept is off)
0x02	Do Auto accept the connection with role switch disabled. (Auto accept is on).
0x03	Do Auto accept the connection with role switch enabled. (Auto accept is on). Note: When auto accepting an incoming synchronous connection, no role switch will be performed. The value 0x03 of the Auto_Accept_Flag will then get the same effect as if the value had been 0x02.
0x04 – 0xFF	Reserved for future use.

Condition for Connection_Setup_Filter_Condition_Type = 0x02

Condition:

Size: 7 Octets

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device of Interest.

Auto_Accept_Flag: *Size: 1 Octet*

Value	Parameter Description
0x01	Do NOT Auto accept the connection. (Auto accept is off)
0x02	Do Auto accept the connection with role switch disabled. (Auto accept is on).
0x03	Do Auto accept the connection with role switch enabled. (Auto accept is on). Note: When auto accepting an incoming synchronous connection, no role switch will be performed. The value 0x03 of the Auto_Accept_Flag will then get the same effect as if the value had been 0x02.

0x04 – 0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Set_Event_Filter command succeeded.
0x01-0xFF	Set_Event_Filter command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

A Command Complete event for this command shall occur when the Controller has enabled the filtering of events. When one of the conditions are met, a specific event shall occur.

7.3.4 Flush Command

Command	OCF	Command Parameters	Return Parameters
HCI_Flush	0x0008	Connection_Handle	Status, Connection_Handle

Description:

The Flush command is used to discard all data that is currently pending for transmission in the Controller for the specified Connection_Handle, even if there currently are chunks of data that belong to more than one L2CAP packet in the Controller. Both automatically-flushable and non-automatically-flushable packets shall be discarded (see Section 5.4.2 on page 428). After this, all data that is sent to the Controller for the same Connection_Handle will be discarded by the Controller until an HCI Data Packet with one of the start Packet_Boundary_Flag values (0x00 or 0x02) is received. When this happens, a new transmission attempt can be made.

This command, when used on a BR/EDR Controller or an AMP Controller, will allow higher-level software to control how long the baseband should try to retransmit a baseband packet for a Connection_Handle before all data that is currently pending for transmission in the Controller should be flushed.

Note: The Flush command is used for ACL connections only. In addition to the Flush command, the automatic flush timers (see Section 7.3.29 on page 602) can be used to automatically flush an automatically-flushable L2CAP packet that is currently being transmitted after the specified flush timer has expired.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify which connection to flush. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Flush command succeeded.
0x01-0xFF	Flush command failed. See Part D, Error Codes for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description

0xXXXX	Connection_Handle to be used to identify which connection the flush command was issued on. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Event(s) generated (unless masked away):

The Flush Occurred event shall occur once the flush is completed. A Flush Occurred event could be from an automatic Flush or could be caused by the Host issuing the Flush command. When the Flush command has completed, a Command Complete event shall be generated, to indicate that the Host caused the Flush.

7.3.5 Read PIN Type Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_PIN_Type	0x0009		Status, PIN_Type

Description:

The Read_PIN_Type command is used to read the PIN_Type configuration parameter. See Section 6.13 on page 440.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_PIN_Type command succeeded.
0x01-0xFF	Read_PIN_Type command failed. See Part D, Error Codes for a list of error codes and descriptions.

PIN_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Variable PIN.
0x01	Fixed PIN.

Event(s) generated (unless masked away):

When the Read_PIN_Type command has completed, a Command Complete event will be generated.

7.3.6 Write PIN Type Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_PIN_Type	0x000A	PIN_Type	Status

Description:

The Write_PIN_Type command is used to write the PIN Type configuration parameter. See Section 6.13 on page 440.

Command Parameters:

PIN_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Variable PIN.
0x01	Fixed PIN.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_PIN_Type command succeeded.
0x01-0xFF	Write_PIN_Type command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_PIN_Type command has completed, a Command Complete event shall be generated.

7.3.7 Create New Unit Key Command

Command	OCF	Command Parameters	Return Parameters
HCI_Create_New_Unit_Key	0x000B		Status

Description:

The Create_New_Unit_Key command is used to create a new unit key. The Controller shall generate a random seed that shall be used to generate the new unit key. All new connection shall use the new unit key, but the old unit key shall still be used for all current connections.

Note: This command will not have any effect for a device which doesn't use unit keys (i.e. a device which uses only combination keys).

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Create_New_Unit_Key command succeeded.
0x01-0xFF	Create_New_Unit_Key command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Create_New_Unit_Key command has completed, a Command Complete event shall be generated.

7.3.8 Read Stored Link Key Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Stored_ Link_Key	0x000D	BD_ADDR, Read_All_Flag	Status, Max_Num_Keys, Num_Keys_Read

Description:

The Read_Stored_Link_Key command provides the ability to read whether one or more link keys are stored in the BR/EDR Controller. The BR/EDR Controller can store a limited number of link keys for other BR/EDR Controllers. Link keys are shared between two BR/EDR Controllers, and are used for all security transactions between the two devices. The read link key command shall not return the link keys value. A Host device may have additional storage capabilities, which can be used to save additional link keys to be reloaded to the BR/EDR Controller when needed. The Read_All_Flag parameter is used to indicate if all of the stored Link Keys should be returned. If Read_All_Flag indicates that all Link Keys are to be returned, then the BD_ADDR command parameter must be ignored. The BD_ADDR command parameter is used to identify which link key to read. The stored Link Keys are returned by one or more Return Link Keys events. See Section 6.14 on page 440.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for the stored link key to be read.

Read_All_Flag: *Size: 1 Octet*

Value	Parameter Description
0x00	Return Link Key for specified BD_ADDR.
0x01	Return all stored Link Keys.
0x02-0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Stored_Link_Key command succeeded.
0x01-0xFF	Read_Stored_Link_Key command failed. See Part D, Error Codes for error codes and descriptions.

Max_Num_Keys: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Total Number of Link Keys that the Controller can store. Range: 0x0000 – 0xFFFF

Num_Keys_Read: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Number of Link Keys Read. Range: 0x0000 – 0xFFFF

Event(s) generated (unless masked away):

Zero or more instances of the Return Link Keys event shall occur after the command is issued. When there are no link keys stored, no Return Link Keys events will be returned. When there are link keys stored, the number of link keys returned in each Return Link Keys event is implementation specific. When the Read_Stored_Link_Key command has completed a Command Complete event shall be generated.

7.3.9 Write Stored Link Key Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Stored_Link_Key	0x0011	Num_Keys_To_Write, BD_ADDR[i], Link_Key[i]	Status, Num_Keys_Written

Description:

The Write_Stored_Link_Key command provides the ability to write one or more link keys to be stored in the BR/EDR Controller. The BR/EDR Controller can store a limited number of link keys for other BR/EDR Controllers. If no additional space is available in the BR/EDR Controller then no additional link keys will be stored. If space is limited and if all the link keys to be stored will not fit in the limited space, then the order of the list of link keys without any error will determine which link keys are stored. Link keys at the beginning of the list will be stored first. The Num_Keys_Written parameter will return the number of link keys that were successfully stored. If no additional space is available, then the Host must delete one or more stored link keys before any additional link keys are stored. The link key replacement algorithm is implemented by the Host and not the BR/EDR Controller. Link keys are shared between two BR/EDR Controllers and are used for all security transactions between the two devices. A Host device may have additional storage capabilities, which can be used to save additional link keys to be reloaded to the BR/EDR Controller when needed. See Section 6.14 on page 440.

Note: Link Keys are only stored by issuing this command.

Command Parameters:

Num_Keys_To_Write: *Size: 1 Octet*

Value	Parameter Description
0xXX	Number of Link Keys to Write. Range: 0x01 - 0x0B

BD_ADDR [i]: *Size: 6 Octets * Num_Keys_To_Write*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for the associated Link Key.

Link_Key[i]: *Size: 16 Octets* Num_Keys_To_Write*

Value	Parameter Description
0xXXXXXXXXXX XXXXXXXXXX XXXXXXXXXX	Link Key for an associated BD_ADDR.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Stored_Link_Key command succeeded.
0x01-0xFF	Write_Stored_Link_Key command failed. See Part D, Error Codes for a list of error codes and descriptions.

Num_Keys_Written: *Size: 1 Octet*

Value	Parameter Description
0xXX	Number of Link Keys successfully written. Range: 0x00 – 0x0B

Event(s) generated (unless masked away):

When the Write_Stored_Link_Key command has completed, a Command Complete event shall be generated.

7.3.10 Delete Stored Link Key Command

Command	OCF	Command Parameters	Return Parameters
HCI_Delete_Stored_ Link_Key	0x0012	BD_ADDR, Delete_All_Flag	Status, Num_Keys_Deleted

Description:

The Delete_Stored_Link_Key command provides the ability to remove one or more of the link keys stored in the BR/EDR Controller. The BR/EDR Controller can store a limited number of link keys for other BR/EDR devices. Link keys are shared between two BR/EDR devices and are used for all security transactions between the two devices. The Delete_All_Flag parameter is used to indicate if all of the stored Link Keys should be deleted. If the Delete_All_Flag indicates that all Link Keys are to be deleted, then the BD_ADDR command parameter must be ignored This command provides the ability to negate all security agreements between two devices. The BD_ADDR command parameter is used to identify which link key to delete. If a link key is currently in use for a connection, then the link key will be deleted when all of the connections are disconnected. See Section 6.14 on page 440.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for the link key to be deleted.

Delete_All_Flag: *Size: 1 Octet*

Value	Parameter Description
0x00	Delete only the Link Key for specified BD_ADDR.
0x01	Delete all stored Link Keys.
0x02-0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Delete_Stored_Link_Key command succeeded.
0x01-0xFF	Delete_Stored_Link_Key command failed. See Part D, Error Codes for error codes and descriptions.

Num_Keys_Deleted: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Number of Link Keys Deleted

Event(s) generated (unless masked away):

When the Delete_Stored_Link_Key command has completed, a Command Complete event shall be generated.

7.3.11 Write Local Name Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Local_Name	0x0013	Local Name	Status

Description:

The Write_Local_Name command provides the ability to modify the user-friendly name for the BR/EDR Controller. See Section 6.23 on page 446.

Command Parameters:

Local Name: *Size: 248 Octets*

Value	Parameter Description
	A UTF-8 encoded User-Friendly Descriptive Name for the device. If the name contained in the parameter is shorter than 248 octets, the end of the name is indicated by a NULL octet (0x00), and the following octets (to fill up 248 octets, which is the length of the parameter) do not have valid values.
	Null terminated Zero length String. **Default.**

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Local_Name command succeeded.
0x01-0xFF	Write_Local_Name command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Local_Name command has completed, a Command Complete event shall be generated.

7.3.12 Read Local Name Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Local_Name	0x0014		Status, Local_Name

Description:

The Read_Local_Name command provides the ability to read the stored user-friendly name for the BR/EDR Controller. See Section 6.23 on page 446.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Local_Name command succeeded
0x01-0xFF	Read_Local_Name command failed (see Part D, Error Codes for list of Error Codes).

Local_Name: *Size: 248 Octets*

Value	Parameter Description
	A UTF-8 encoded User Friendly Descriptive Name for the device. If the name contained in the parameter is shorter than 248 octets, the end of the name is indicated by a NULL octet (0x00), and the following octets (to fill up 248 octets, which is the length of the parameter) do not have valid values.

Event(s) generated (unless masked away):

When the Read_Local_Name command has completed a Command Complete event shall be generated.

7.3.13 Read Connection Accept Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Connection_ Accept_Timeout	0x0015		Status, Conn_Accept_Timeout

Description:

This command reads the value for the Connection_Accept_Timeout configuration parameter. See Section 6.7 on page 437.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Connection_Accept_Timeout command succeeded.
0x01-0xFF	Read_Connection_Accept_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Conn_Accept_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Connection Accept Timeout measured in Number of BR/EDR Baseband slots. Interval Length = N * 0.625 msec (1 Baseband slot) Range for N: 0x0001 – 0xB540 Time Range: 0.625 msec -29 seconds

Event(s) generated (unless masked away):

When the Read_Connection_Timeout command has completed, a Command Complete event shall be generated.

7.3.14 Write Connection Accept Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Connection_Accept_Timeout	0x0016	Conn_Accept_Timeout	Status

Description:

This command writes the value for the Connection_Accept_Timeout configuration parameter. See Section 6.7 on page 437.

Command Parameters:

Conn_Accept_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Connection Accept Timeout measured in Number of BR/EDR Baseband slots. Interval Length = N * 0.625 msec (1 Baseband slot) Range for N: 0x0001 – 0xB540 Time Range: 0.625 msec - 29 seconds Default: N = 0x1FA0 Time = 5.06 Sec Mandatory Range for Controller: 0x00A0 to 0xB540

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Connection_Accept_Timeout command succeeded.
0x01-0xFF	Write_Connection_Accept_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Connection_Accept_Timeout command has completed, a Command Complete event shall be generated.

7.3.15 Read Page Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Page_Timeout	0x0017		Status, Page_Timeout

Description:

This command reads the value for the Page_Timeout configuration parameter. See Section 6.6 on page 437.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Page_Timeout command succeeded.
0x01-0xFF	Read_Page_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Page_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Page Timeout measured in Number of Baseband slots. Interval Length = N * 0.625 msec (1 Baseband slot) Range for N: 0x0001 – 0xFFFF Time Range: 0.625 msec -40.9 Seconds

Event(s) generated (unless masked away):

When the Read_Page_Timeout command has completed, a Command Complete event shall be generated.

7.3.16 Write Page Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Page_Timeout	0x0018	Page_Timeout	Status

Description:

This command writes the value for the Page_Timeout configuration parameter. The Page_Timeout configuration parameter defines the maximum time the local Link Manager shall wait for a baseband page response from the remote device at a locally initiated connection attempt. If this time expires and the remote device has not responded to the page at baseband level, the connection attempt will be considered to have failed.

Command Parameters:

Page_Timeout: *Size: 2 Octets*

Value	Parameter Description
0	Illegal Page Timeout. Must be larger than 0.
N = 0xXXXX	Page Timeout measured in Number of Baseband slots. Interval Length = N * 0.625 msec (1 Baseband slot) Range for N: 0x0001 – 0xFFFF Time Range: 0.625 msec -40.9 Seconds Default: N = 0x2000 Time = 5.12 Sec Mandatory Range for Controller: 0x0016 to 0xFFFF

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Page_Timeout command succeeded.
0x01-0xFF	Write_Page_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Page_Timeout command has completed, a Command Complete event shall be generated.

7.3.17 Read Scan Enable Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Scan_Enable	0x0019		Status, Scan_Enable

Description:

This command reads the value for the Scan_Enable parameter configuration parameter. See Section 6.1 on page 435.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Scan_Enable command succeeded.
0x01-0xFF	Read_Scan_Enable command failed. See Part D, Error Codes for a list of error codes and descriptions.

Scan_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	No Scans enabled.
0x01	Inquiry Scan enabled. Page Scan disabled.
0x02	Inquiry Scan disabled. Page Scan enabled.
0x03	Inquiry Scan enabled. Page Scan enabled.
0x04-0xFF	Reserved

Event(s) generated (unless masked away):

When the Read_Scan_Enable command has completed, a Command Complete event shall be generated.

7.3.18 Write Scan Enable Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Scan_Enable	0x001A	Scan_Enable	Status

Description:

This command writes the value for the Scan_Enable configuration parameter. See Section 6.1 on page 435.

Command Parameters:

Scan_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	No Scans enabled. **Default.**
0x01	Inquiry Scan enabled. Page Scan disabled.
0x02	Inquiry Scan disabled. Page Scan enabled.
0x03	Inquiry Scan enabled. Page Scan enabled.
0x04-0xFF	Reserved

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Scan_Enable command succeeded.
0x01-0xFF	Write_Scan_Enable command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Scan_Enable command has completed, a Command Complete event shall be generated.

7.3.19 Read Page Scan Activity Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Page_Scan_ Activity	0x001B		Status, Page_Scan_Interval, Page_Scan_Window

Description:

This command reads the value for Page_Scan_Interval and Page_Scan_Window configuration parameters. See Section 6.8 on page 438 and Section 6.9 on page 438.

Note: Page Scan is only performed when Page_Scan is enabled (see 6.1, 7.3.17 and 7.3.18). A changed Page_Scan_Interval could change the local Page_Scan_Repetition_Mode (see Part B, Baseband Specification).

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Page_Scan_Activity command succeeded.
0x01-0xFF	Read_Page_Scan_Activity command failed. See Part D, Error Codes for a list of error codes and descriptions.

Page_Scan_Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0012 – 0x1000 Time = N * 0.625 msec Range: 11.25 msec - 2560 msec; only even values are valid

Page_Scan_Window: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0011 - 0x1000 Time = N * 0.625 msec Range: 10.625 msec - 2560 msec

Event(s) generated (unless masked away):

When the Read_Page_Scan_Activity command has completed, a Command Complete event shall be generated.

7.3.20 Write Page Scan Activity Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Page_Scan_Activity	0x001C	Page_Scan_Interval, Page_Scan_Window	Status

Description:

This command writes the values for the Page_Scan_Interval and Page_Scan_Window configuration parameters. The Page_Scan_Window shall be less than or equal to the Page_Scan_Interval. See Section 6.8 on page 438 and Section 6.9 on page 438.

Note: Page Scan is only performed when Page_Scan is enabled (see 6.1, 7.3.17 and 7.3.18). A changed Page_Scan_Interval could change the local Page_Scan_Repetition_Mode (see Part B, Baseband Specification).

Command Parameters:

Page_Scan_Interval: *Size: 2 Octets*

Value	Parameter Description
See Section 6.8 on page 438	

Page_Scan_Window: *Size: 2 Octets*

Value	Parameter Description
See Section 6.9 on page 438	

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Page_Scan_Activity command succeeded.
0x01-0xFF	Write_Page_Scan_Activity command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Page_Scan_Activity command has completed, a Command Complete event shall be generated.

7.3.21 Read Inquiry Scan Activity Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_ Inquiry_Scan_Activity	0x001D		Status, Inquiry_Scan_Interval, Inquiry_Scan_Window

Description:

This command reads the value for Inquiry_Scan_Interval and Inquiry_Scan_Window configuration parameter. See Section 6.2 on page 435 and Section 6.3 on page 436.

Note: Inquiry Scan is only performed when Inquiry_Scan is enabled see 6.1, 7.3.17 and 7.3.18).

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Inquiry_Scan_Activity command succeeded.
0x01-0xFF	Read_Inquiry_Scan_Activity command failed. See Part D, Error Codes for a list of error codes and descriptions.

Inquiry_Scan_Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0012 – 0x1000 Time = N * 0.625 msec Range: 11.25 - 2560 msec; only even values are valid

Inquiry_Scan_Window: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Size: 2 Octets Range: 0x0011 - 0x1000 Time = N * 0.625 msec Range: 10.625 msec - 2560 msec

Event(s) generated (unless masked away):

When the Read_Inquiry_Scan_Activity command has completed, a Command Complete event shall be generated.

7.3.22 Write Inquiry Scan Activity Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Inquiry_Scan_Activity	0x001E	Inquiry_Scan_Interval, Inquiry_Scan_Window	Status

Description:

This command writes the values for the Inquiry_Scan_Interval and Inquiry_Scan_Window configuration parameters. The Inquiry_Scan_Window shall be less than or equal to the Inquiry_Scan_Interval. See Section 6.2 on page 435 and Section 6.3 on page 436.

Note: Inquiry Scan is only performed when Inquiry_Scan is enabled (see 6.1, 7.3.17 and 7.3.18).

Command Parameters:

Inquiry_Scan_Interval: *Size: 2 Octets*

Value	Parameter Description
See Section 6.2 on page 435.	

Inquiry_Scan_Window: *Size: 2 Octets*

Value	Parameter Description
See Section 6.3 on page 436.	

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Inquiry_Scan_Activity command succeeded.
0x01-0xFF	Write_Inquiry_Scan_Activity command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Inquiry_Scan_Activity command has completed, a Command Complete event shall be generated.

7.3.23 Read Authentication Enable Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_ Authentication_Enable	0x001F		Status, Authentication_Enable

Description:

This command reads the value for the Authentication_Enable configuration parameter. See Section 10.10.1 on page 869.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Authentication_Enable command succeeded.
0x01-0xFF	Read_Authentication_Enable command failed. See Part D, Error Codes for a list of error codes and descriptions.

Authentication_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	Authentication not required.
0x01	Authentication required for all connections.
0x02-0xFF	Reserved

Event(s) generated (unless masked away):

When the Read_Authentication_Enable command has completed, a Command Complete event shall be generated.

7.3.24 Write Authentication Enable Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_ Authentication_Enable	0x0020	Authentication_Enable	Status

Description:

This command writes the value for the Authentication_Enable configuration parameter. See Authentication Enable on page 441.

The Authentication_Enable configuration parameter shall only apply to connections (e.g. send an LMP_in_rand or LMP_au_rand) when the remote device's Host or BR/EDR Controller does not support Secure Simple Pairing or when the local Host does not support Secure Simple Pairing.

Note: Requires LM to read host features during connection setup.

Command Parameters:

Authentication_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	Authentication not required. **Default.**
0x01	Authentication required for all connections.
0x02-0xFF	Reserved

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Authentication_Enable command succeeded.
0x01-0xFF	Write_Authentication_Enable command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Authentication_Enable command has completed, a Command Complete event shall be generated.

7.3.25 Read Class of Device Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Class_of_Device	0x0023		Status, Class_of_Device

Description:

This command reads the value for the Class_of_Device parameter. See Section 6.26 on page 447.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Class_of_Device command succeeded.
0x01-0xFF	Read_Class_of_Device command failed. See Part D, Error Codes for a list of error codes and descriptions.

Class_of_Device: *Size: 3 Octets*

Value	Parameter Description
0xXXXXXX	Class of Device for the device.

Event(s) generated (unless masked away):

When the Read_Class_of_Device command has completed, a Command Complete event shall be generated.

7.3.26 Write Class of Device Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Class_of_Device	0x0024	Class_of_Device	Status

Description:

This command writes the value for the Class_of_Device parameter.
See Section 6.26 on page 447.

Command Parameters:

Class_of_Device: *Size: 3 Octets*

Value	Parameter Description
0xXXXXXX	Class of Device for the device.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Class_of_Device command succeeded.
0x01-0xFF	Write_Class_of_Device command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Class_of_Device command has completed, a Command Complete event shall be generated.

7.3.27 Read Voice Setting Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Voice_Setting	0x0025		Status, Voice_Setting

Description:

This command reads the values for the Voice_Setting configuration parameter. See Section 6.12 on page 439.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Voice_Setting command succeeded.
0x01-0xFF	Read_Voice_Setting command failed. See Part D, Error Codes for a list of error codes and descriptions.

Voice_Setting: *Size: 2 Octets (10 Bits meaningful)*

Value	Parameter Description
See Section 6.12 on page 439.	

Event(s) generated (unless masked away):

When the Read_Voice_Setting command has completed, a Command Complete event shall be generated.

7.3.28 Write Voice Setting Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Voice_Setting	0x0026	Voice_Setting	Status

Description:

This command writes the values for the Voice_Setting configuration parameter. See Section 6.12 on page 439.

Command Parameters:

Voice_Setting: *Size: 2 Octets (10 Bits meaningful)*

Value	Parameter Description
See Section 6.12 on page 439.	

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Voice_Setting command succeeded.
0x01-0xFF	Write_Voice_Setting command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Voice_Setting command has completed, a Command Complete event shall be generated.

7.3.29 Read Automatic Flush Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Automatic_Flush_Timeout	0x0027	Connection_Handle	Status, Connection_Handle, Flush_Timeout

Description:

This command reads the value for the Flush_Timeout parameter for the specified Connection_Handle. See Section 6.19 on page 444.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's Flush Timeout to read. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Automatic_Flush_Timeout command succeeded.
0x01-0xFF	Read_Automatic_Flush_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's Flush Timeout has been read. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Flush_Timeout: *Size: 2 Octets*

Value	Parameter Description
0	Timeout = ∞; No Automatic Flush
N = 0xXXXX	Flush Timeout = N * 0.625 msec Size: 11 bits Range: 0x0001 – 0x07FF

Event(s) generated (unless masked away):

When the Read_Automatic_Flush_Timeout command has completed, a Command Complete event shall be generated.

7.3.30 Write Automatic Flush Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Automatic_Flush_Timeout	0x0028	Connection_Handle, Flush_Timeout	Status, Connection_Handle

Description:

This command writes the value for the Flush_Timeout parameter for the specified Connection_Handle. See Section 6.19 on page 444.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's Flush Timeout to write to. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Flush_Timeout: *Size: 2 Octets*

Value	Parameter Description
0	Timeout = ∞; No Automatic Flush. **Default.**
N = 0xXXXX	Flush Timeout = N * 0.625 msec Size: 11 bits Range: 0x0001 – 0x07FF Mandatory Range for Controller: 0x0002 to 0x07FF

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Automatic_Flush_Timeout command succeeded.
0x01-0xFF	Write_Automatic_Flush_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's Flush Timeout has been written. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Event(s) generated (unless masked away):

When the Write_Automatic_Flush_Timeout command has completed, a Command Complete event shall be generated.

7.3.31 Read Num Broadcast Retransmissions Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Num_Broadcast_ Retransmissions	0x0029		Status, Num_Broadcast_Retransmissions

Description:

This command reads the device's parameter value for the Number of Broadcast Retransmissions. See Section 6.20 on page 444

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Num_Broadcast_Retransmissions command succeeded.
0x01-0xFF	Read_Num_Broadcast_Retransmissions command failed. See Part D, Error Codes for a list of error codes and descriptions.

Num_Broadcast_Retransmissions: *Size: 1 Octet*

Value	Parameter Description
See Section 6.20 on page 444.	

Event(s) generated (unless masked away):

When the Read_Num_Broadcast_Retransmission command has completed, a Command Complete event shall be generated.

7.3.32 Write Num Broadcast Retransmissions Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Num_Broadcast_ Retransmissions	0x002A	Num_Broadcast_ Retransmissions	Status

Description:

This command writes the device's parameter value for the Number of Broadcast Retransmissions. See Section 6.20 on page 444.

Command Parameters:

Num_Broadcast_Retransmissions: *Size: 1 Octet*

Value	Parameter Description
See Section 6.20 on page 444.	

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Num_Broadcast_Retransmissions command succeeded.
0x01-0xFF	Write_Num_Broadcast_Retransmissions command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Num_Broadcast_Retransmissions command has completed, a Command Complete event shall be generated.

7.3.33 Read Hold Mode Activity Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Hold_Mode_Activity	0x002B		Status, Hold_Mode_Activity

Description:

This command reads the value for the Hold_Mode_Activity parameter.
See Section 6.16 on page 441.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Hold_Mode_Activity command succeeded.
0x01-0xFF	Read_Hold_Mode_Activity command failed. See Part D, Error Codes for a list of error codes and descriptions.

Hold_Mode_Activity: *Size: 1 Octet*

Value	Parameter Description
0x00	Maintain current Power State.
0x01	Suspend Page Scan.
0x02	Suspend Inquiry Scan.
0x04	Suspend Periodic Inquiries.
0x08-0xFF	Reserved for future use.

Event(s) generated (unless masked away):

When the Read_Hold_Mode_Activity command has completed, a Command Complete event shall be generated.

7.3.34 Write Hold Mode Activity Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Hold_Mode_Activity	0x002C	Hold_Mode_Activity	Status

Description:

This command writes the value for the Hold_Mode_Activity parameter. See Section 6.16 on page 441.

Command Parameters:

Hold_Mode_Activity: *Size: 1 Octet*

Value	Parameter Description
0x00	Maintain current Power State. **Default.**
0x01	Suspend Page Scan.
0x02	Suspend Inquiry Scan.
0x04	Suspend Periodic Inquiries.
0x08-0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Hold_Mode_Activity command succeeded.
0x01-0xFF	Write_Hold_Mode_Activity command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Hold_Mode_Activity command has completed, a Command Complete event shall be generated.

7.3.35 Read Transmit Power Level Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Transmit_Power_Level	0x002D	Connection_Handle, Type	Status, Connection_Handle, Transmit_Power_Level

Description:

This command reads the values for the Transmit_Power_Level parameter for the specified Connection_Handle. The Connection_Handle shall be a Connection_Handle for an ACL connection.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's Transmit Power Level setting to read. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Read Current Transmit Power Level.
0x01	Read Maximum Transmit Power Level.
0x02-0xFF	Reserved

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Transmit_Power_Level command succeeded.
0x01-0xFF	Read_Transmit_Power_Level command failed. See Part D, Error Codes for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's Transmit Power Level setting is returned. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Transmit_Power_Level: *Size: 1 Octet*

Value	Parameter Description
N = 0xXX	Size: 1 Octet (signed integer) Range: -30 ≤ N ≤ 20 Units: dBm

Event(s) generated (unless masked away):

When the Read_Transmit_Power_Level command has completed, a Command Complete event shall be generated.

7.3.36 Read Synchronous Flow Control Enable Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_ Synchronous_Flow_ Control_Enable	0x002E		Status, Synchronous_Flow_ Control_Enable

Description:

The Read_Synchronous_Flow_Control_Enable command provides the ability to read the Synchronous_Flow_Control_Enable parameter. See Section 6.22 on page 445.

The Synchronous_Flow_Control_Enable parameter shall only be changed if no connection exists.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Synchronous_Flow_Control_Enable command succeeded
0x01-0xFF	Read_Synchronous_Flow_Control_Enable command failed (see Part D, Error Codes for list of Error Codes).

Synchronous_Flow_Control_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	Synchronous Flow Control is disabled. No Number of Completed Packets events will be sent from the Controller for Synchronous Connection_Handles.
0x01	Synchronous Flow Control is enabled. Number of Completed Packets events will be sent from the Controller for Synchronous Connection_Handles.

Event(s) generated (unless masked away):

When the Read_Synchronous_Flow_Control_Enable command has completed a Command Complete event shall be generated.

7.3.37 Write Synchronous Flow Control Enable Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Synchronous_ Flow_Control_Enable	0x002F	Synchronous_Flow_ Control_Enable	Status

Description:

The Write_Synchronous_Flow_Control_Enable command provides the ability to write the Synchronous_Flow_Control_Enable parameter. See Section 6.22 on page 445.

The Synchronous_Flow_Control_Enable parameter can only be changed if no connections exist.

Command Parameters:

Synchronous_Flow_Control_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	Synchronous Flow Control is disabled. No Number of Completed Packets events shall be sent from the Controller for synchronous Connection Handles. **Default**
0x01	Synchronous Flow Control is enabled. Number of Completed Packets events shall be sent from the Controller for synchronous Connection Handles.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Synchronous_Flow_Control_Enable command succeeded
0x01-0xFF	Write_Synchronous_Flow_Control_Enable command failed (see Part D, Error Codes for list of Error Codes.)

Event(s) generated (unless masked away):

When the Write_Synchronous_Flow_Control_Enable command has completed a Command Complete event shall be generated.

7.3.38 Set Controller To Host Flow Control Command

Command	OCF	Command Parameters	Return Parameters
HCI_Set_Controller_To_ Host_Flow_Control	0x0031	Flow_Control_Enable	Status

Description:

This command is used by the Host to turn flow control on or off for data and/or voice sent in the direction from the Controller to the Host. If flow control is turned off, the Host should not send the Host_Number_Of_Completed_Packets command. That command will be ignored by the Controller if it is sent by the Host and flow control is off. If flow control is turned on for HCI ACL Data Packets and off for HCI synchronous Data Packets, Host_Number_Of_Completed_Packets commands sent by the Host should only contain Connection Handles for ACL connections. If flow control is turned off for HCI ACL Data Packets and on for HCI synchronous Data Packets, Host_Number_Of_Completed_Packets commands sent by the Host should only contain Connection Handles for synchronous connections. If flow control is turned on for HCI ACL Data Packets and HCI synchronous Data Packets, the Host will send Host_Number_Of_Completed_Packets commands both for ACL connections and synchronous connections.

The Flow_Control_Enable parameter shall only be changed if no connections exist.

Command Parameters:

Flow_Control_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	Flow control **off** in direction from Controller to Host. **Default.**
0x01	Flow control **on** for HCI ACL Data Packets and **off** for HCI synchronous Data Packets in direction from Controller to Host.
0x02	Flow control **off** for HCI ACL Data Packets and **on** for HCI synchronous Data Packets in direction from Controller to Host.
0x03	Flow control **on** both for HCI ACL Data Packets and HCI synchronous Data Packets in direction from Controller to Host.
0x04-0xFF	Reserved

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Set_Controller_To_Host_Flow_Control command succeeded.
0x01-0xFF	Set_Controller_To_Host_Flow_Control command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Set_Controller_To_Host_Flow_Control command has completed, a Command Complete event shall be generated.

7.3.39 Host Buffer Size Command

Command	OCF	Command Parameters	Return Parameters
HCI_Host_Buffer_Size	0x0033	Host_ACL_Data_Packet_Length, Host_Synchronous_Data_Packet_Length, Host_Total_Num_ACL_Data_Packets, Host_Total_Num_Synchronous_Data_Packets	Status

Description:

The Host_Buffer_Size command is used by the Host to notify the Controller about the maximum size of the data portion of HCI ACL and synchronous Data Packets sent from the Controller to the Host. The Controller shall segment the data to be transmitted from the Controller to the Host according to these sizes, so that the HCI Data Packets will contain data with up to these sizes. The Host_Buffer_Size command also notifies the Controller about the total number of HCI ACL and synchronous Data Packets that can be stored in the data buffers of the Host. If flow control from the Controller to the Host is turned off, and the Host_Buffer_Size command has not been issued by the Host, this means that the Controller will send HCI Data Packets to the Host with any lengths the Controller wants to use, and it is assumed that the data buffer sizes of the Host are unlimited. If flow control from the Controller to the Host is turned on, the Host_Buffer_Size command shall after a power-on or a reset always be sent by the Host before the first Host_Number_Of_Completed_Packets command is sent.

The Set Controller To Host Flow Control Command is used to turn flow control on or off. The Host_ACL_Data_Packet_Length command parameter will be used to determine the size of the L2CAP segments contained in ACL Data Packets, which are transferred from the Controller to the Host. The Host_Synchronous_Data_Packet_Length command parameter is used to determine the maximum size of HCI synchronous Data Packets. Both the Host and the Controller shall support command and event packets, where the data portion (excluding header) contained in the packets is 255 octets in size.

The Host_Total_Num_ACL_Data_Packets command parameter contains the total number of HCI ACL Data Packets that can be stored in the data buffers of the Host. The Controller will determine how the buffers are to be divided between different Connection Handles. The Host_Total_Num_Synchronous_Data_Packets command parameter gives the same information for HCI synchronous Data Packets.

Note: The Host_ACL_Data_Packet_Length and Host_Synchronous_Data_Packet_Length command parameters do not include the length of the HCI Data Packet header.

Command Parameters:

Host_ACL_Data_Packet_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Maximum length (in octets) of the data portion of each HCI ACL Data Packet that the Host is able to accept.

Host_Synchronous_Data_Packet_Length: *Size: 1 Octet*

Value	Parameter Description
0xXX	Maximum length (in octets) of the data portion of each HCI synchronous Data Packet that the Host is able to accept.

Host_Total_Num_ACL_Data_Packets: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Total number of HCI ACL Data Packets that can be stored in the data buffers of the Host.

Host_Total_Num_Synchronous_Data_Packets: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Total number of HCI synchronous Data Packets that can be stored in the data buffers of the Host.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Host_Buffer_Size command succeeded.
0x01-0xFF	Host_Buffer_Size command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Host_Buffer_Size command has completed, a Command Complete event shall be generated.

7.3.40 Host Number Of Completed Packets Command

Command	OCF	Command Parameters	Return Parameters
HCI_Host_Number_Of_Completed_Packets	0x0035	Number_Of_Handles, Connection_Handle[i], Host_Num_Of_Completed_Packets [i]	

Description:

The Host_Number_Of_Completed_Packets command is used by the Host to indicate to the Controller the number of HCI Data Packets that have been completed for each Connection Handle since the previous Host_Number_Of_Completed_Packets command was sent to the Controller. This means that the corresponding buffer space has been freed in the Host. Based on this information, and the Host_Total_Num_ACL_Data_Packets and Host_Total_Num_Synchronous_Data_Packets command parameters of the Host_Buffer_Size command, the Controller can determine for which Connection Handles the following HCI Data Packets should be sent to the Host. The command should only be issued by the Host if flow control in the direction from the Controller to the Host is on and there is at least one connection, or if the Controller is in local loopback mode. Otherwise, the command will be ignored by the Controller. When the Host has completed one or more HCI Data Packet(s) it shall send a Host_Number_Of_Completed_Packets command to the Controller, until it finally reports that all pending HCI Data Packets have been completed. The frequency at which this command is sent is manufacturer specific.

The Set Controller To Host Flow Control Command is used to turn flow control on or off. If flow control from the Controller to the Host is turned on, the Host_Buffer_Size command shall always be sent by the Host after a power-on or a reset before the first Host_Number_Of_Completed_Packets command is sent.

Note: The Host_Number_Of_Completed_Packets command is a special command in the sense that no event is normally generated after the command has completed. The command may be sent at any time by the Host when there is at least one connection, or if the Controller is in local loopback mode independent of other commands. The normal flow control for commands is not used for the Host_Number_Of_Completed_Packets command.

Command Parameters:

Number_Of_Handles: *Size: 1 Octet*

Value	Parameter Description

0xXX	The number of Connection Handles and Host_Num_Of_Completed_Packets parameters pairs contained in this command. Range: 0-255

*Connection_Handle[i]: Size: Number_Of_Handles*2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection Handle Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

*Host_Num_Of_Completed_Packets [i]: Size: Number_Of_Handles * 2 Octets*

Value	Parameter Description
N = 0xXXXX	The number of HCI Data Packets that have been completed for the associated Connection Handle since the previous time the event was returned. Range for N: 0x0000-0xFFFF

Return Parameters:

None.

Event(s) generated (unless masked away):

Normally, no event is generated after the Host_Number_Of_Completed_Packets command has completed. However, if the Host_Number_Of_Completed_Packets command contains one or more invalid parameters, the Controller shall return a Command Complete event with a failure status indicating the *Invalid HCI Command Parameters error code*. The Host may send the Host_Number_Of_Completed_Packets command at any time when there is at least one connection, or if the Controller is in local loopback mode. The normal flow control for commands is not used for this command.

7.3.41 Read Link Supervision Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Link_Supervision_Timeout	0x0036	Handle	Status, Handle, Link_Supervision_Timeout

Description:

This command reads the value for the Link_Supervision_Timeout parameter for the Controller.

The Handle used for this command shall be the ACL connection to the appropriate device. See Section 6.21 on page 445.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection Handle's Link Supervision Timeout value is to be read. On an AMP, a Physical Link Handle is used as the lower 8 bits of the Handle. The upper 4 bits are reserved and shall be set to 0. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Link_Supervision_Timeout command succeeded.
0x01-0xFF	Read_Link_Supervision_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection Handle's Link Supervision Timeout value was read. The Handle is a Connection Handle for a BR/EDR Controller. On an AMP a Physical Link Handle is used as the lower 8 bits of the Handle. The upper 4 bits are reserved and shall be set to 0. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Link_Supervision_Timeout: *Size: 2 Octets*

Value	Parameter Description
0x0000	No Link_Supervision_Timeout.
N = 0xXXXX	Measured in Number of BR/EDR Baseband slots Link_Supervision_Timeout = N * 0.625 msec (1 Baseband slot) Range for N: 0x0001 – 0xFFFF Time Range: 0.625ms - 40.9 sec

Event(s) generated (unless masked away):

When the Read_Link_Supervision_Timeout command has completed, a Command Complete event shall be generated.

7.3.42 Write Link Supervision Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Link_Supervision_ Timeout	0x0037	Handle, Link_Supervision_ Timeout	Status, Handle

Description:

This command writes the value for the Link_Supervision_Timeout parameter for a BR/EDR or AMP Controller. For BR/EDR Controllers, this command shall only be issued on the master for the given Connection Handle. If this command is issued on a slave, the command shall be rejected by the BR/EDR controller with the error code Command Disallowed. The command may be issued to any AMP Controller without restriction to role.

The Handle used for this command shall be the ACL connection to the appropriate device. This command will set the Link_Supervision_
Timeout values for other Synchronous Handles to that device.
See Section 6.21 on page 445.

Command Parameters:

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Handle's Link Supervision Timeout value is to be written. The Handle is a Connection Handle for a BR/EDR Controller. On an AMP a Physical Link Handle is used as the lower 8 bits of the Handle. The upper 4 bits are reserved and shall be set to 0. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Link_Supervision_Timeout: *Size: 2 Octets*

Value	Parameter Description
0x0000	No Link_Supervision_Timeout.
N = 0xXXXX	Measured in Number of BR/EDR Baseband slots Link_Supervision_Timeout = N*0.625 msec (1 Baseband slot) Range for N: 0x0001 – 0xFFFF Time Range: 0.625ms – 40.9 sec **Default**: N = 0x7D00 Link_Supervision_Timeout = 20 sec Mandatory Range for Controller: 0x0190 to 0xFFFF; plus 0 for infinite timeout

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Link_Supervision_Timeout command succeeded.
0x01-0xFF	Write_Link_Supervision_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Handle's Link Supervision Timeout value was written. The Handle is a Connection Handle for a BR/EDR Controller. On an AMP a Physical Link Handle is used as the lower 8 bits of the Handle. The upper 4 bits (of the 12 meaningful bits) are reserved and shall be set to 0. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Event(s) generated (unless masked away):

When the Write_Link_Supervision_Timeout command has completed, a Command Complete event shall be generated.

7.3.43 Read Number Of Supported IAC Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Number_Of_Supported_IAC	0x0038		Status, Num_Support_IAC

Description:

This command reads the value for the number of Inquiry Access Codes (IAC) that the local BR/EDR Controller can simultaneous listen for during an Inquiry Scan. All BR/EDR Controllers are required to support at least one IAC, the General Inquiry Access Code (the GIAC). Some BR/EDR Controllers support additional IACs.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Number_Of_Supported_IAC command succeeded.
0x01-0xFF	Read_Number_Of_Supported_IAC command failed. See Part D, Error Codes for a list of error codes and descriptions.

Num_Support_IAC *Size: 1 Octet*

Value	Parameter Description
0xXX	Specifies the number of Supported IAC that the local BR/EDR Controller can simultaneous listen for during an Inquiry Scan. Range: 0x01-0x40

Event(s) generated (unless masked away):

When the Read_Number_Of_Supported_IAC command has completed, a Command Complete event shall be generated.

7.3.44 Read Current IAC LAP Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Current_IAC_LAP	0x0039		Status, Num_Current_IAC, IAC_LAP[i]

Description:

This command reads the LAP(s) used to create the Inquiry Access Codes (IAC) that the local BR/EDR Controller is simultaneously scanning for during Inquiry Scans. All BR/EDR Controllers shall support at least one IAC, the General Inquiry Access Code (the GIAC). Some BR/EDR Controllers support additional IACs.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Current_IAC_LAP command succeeded.
0x01-0xFF	Read_Current_IAC_LAP command failed. See Part D, Error Codes for a list of error codes and descriptions.

Num_Current_IAC *Size: 1 Octet*

Value	Parameter Description
0xXX	Specifies the number of IACs which are currently in use by the local BR/EDR Controller to simultaneously listen for during an Inquiry Scan. Range: 0x01-0x40

IAC_LAP[i] *Size: 3 Octets * Num_Current_IAC*

Value	Parameter Description
0xXXXXXX	LAPs used to create the IAC which is currently in use by the local BR/EDR Controller to simultaneously listen for during an Inquiry Scan. Range: 0x9E8B00-0x9E8B3F

Event(s) generated (unless masked away):

When the Read_Current_IAC_LAP command has completed, a Command Complete event shall be generated.

7.3.45 Write Current IAC LAP Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Current_IAC_LAP	0x003A	Num_Current_IAC, IAC_LAP[i]	Status

Description:

This command writes the LAP(s) used to create the Inquiry Access Codes (IAC) that the local BR/EDR Controller is simultaneously scanning for during Inquiry Scans. All BR/EDR Controller shall support at least one IAC, the General Inquiry Access Code (the GIAC). Some BR/EDR Controllers support additional IACs.

This command shall clear any existing IACs and stores Num_Current_IAC and the IAC_LAPs in to the controller. If Num_Current_IAC is greater than Num_Support_IAC then only the first Num_Support_IAC shall be stored in the controller, and a Command Complete event with error code *Success (0x00)* shall be generated.

Command Parameters:

Num_Current_IAC *Size: 1 Octet*

Value	Parameter Description
0xXX	Specifies the number of IACs which are currently in use by the local BR/EDR Controller to simultaneously listen for during an Inquiry Scan. Range: 0x01-0x40

IAC_LAP[i] *Size: 3 Octets * Num_Current_IAC*

Value	Parameter Description
0xXXXXXX	LAP(s) used to create IAC which is currently in use by the local BR/EDR Controller to simultaneously listen for during an Inquiry Scan. Range: 0x9E8B00-0x9E8B3F. The GIAC is the default IAC to be used. If additional IACs are supported, additional default IAC will be determined by the manufacturer.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Current_IAC_LAP command succeeded.
0x01-0xFF	Write_Current_IAC_LAP command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Current_IAC_LAP command has completed, a Command Complete event shall be generated.

7.3.46 Set AFH Host Channel Classification Command

Command	OCF	Command Parameters	Return Parameters
Set_AFH_Host_Channel_ Classification	0x003F	AFH_Host_Channel_ Classification	Status

Description:

The Set_AFH_Host_Channel_Classification command allows the Host to specify a channel classification based on its "local information". This classification persists until overwritten with a subsequent Set_AFH_Host_Channel_Classification command or until the BR/EDR Controller is reset.

This command shall be supported by a device that declares support for any of the AFH_capable_master, AFH_classification_slave or AFH_classification_master features.

If this command is used, updates should be sent within 10 seconds, of the Host knowing that the channel classification has changed. The interval between two successive commands sent shall be at least 1 second.

Command Parameters:

AFH_Host_Channel_Classification: *Size: 10 Octets (79 Bits meaningful)*

Value	Parameter Description
0xXXXXXXXX XXXXXXXXXX XX	This parameter contains 79 1-bit field. The nth such field (in the range 0 to 78) contains the value for channel *n*: Channel *n* is bad = 0 Channel *n* is unknown = 1 The most significant bit is reserved and shall be set to 0 At least N_{min} channels shall be marked as unknown. (See Baseband Specification, Section 2.3.1, on page 76)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Set_AFH_Host_Channel_Classification command succeeded.
0x01-0xFF	Set_AFH_Host_Channel_Classification command failed. Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Set_AFH_Host_Channel_Classification command has completed, a Command Complete event shall be generated.

7.3.47 Read Inquiry Scan Type Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Inquiry_Scan_Type	0x0042		Status, Inquiry_Scan_Type

Description:

This command reads the Inquiry_Scan_Type configuration parameter from the local BR/EDR Controller. See Section 6.4 on page 436.For details, see the Baseband Specification, [Vol 2] Part B, Section 8.4.1, Inquiry scan substate.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Inquiry_Scan_Type command succeeded
0x01-0xFF	Read_Inquiry_Scan_Type command failed. See Part D, Error Codes for a list of error codes and descriptions.

Inquiry_Scan_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Standard Scan (**default**)
0x01	Interlaced Scan
0x02-0xFF	Reserved

Event(s) generated (unless masked away):

When the Read_Inquiry_Scan_Type command has completed, a Command Complete event shall be generated.

7.3.48 Write Inquiry Scan Type Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Inquiry_Scan_Type	0x0043	Scan_Type	Status

Description:

This command writes the Inquiry Scan Type configuration parameter of the local BR/EDR Controller. See Section 6.4 on page 436. For details, see the Baseband Specification, Part B, Inquiry scan substate.

Command Parameters:

Scan_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Standard Scan (**default**)
0x01	Interlaced Scan
0x02-0xFF	Reserved

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Inquiry_Scan_Type command succeeded
0x01-0xFF	Write_Inquiry_Scan_Type command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Inquiry_Scan_Type command has completed, a Command Complete event shall be generated.

7.3.49 Read Inquiry Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Inquiry_Mode	0x0044		Status, Inquiry_Mode

Description:

This command reads the Inquiry_Mode configuration parameter of the local BR/EDR Controller. See Section 6.5 on page 436.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Inquiry_Mode command succeeded.
0x01-0xFF	Read_Inquiry_Mode command failed. See Part D, Error Codes for list of Error Codes.

Inquiry_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Standard Inquiry Result event format
0x01	Inquiry Result format with RSSI
0x02	Inquiry Result with RSSI format or Extended Inquiry Result format
0x03-0xFF	Reserved

Event(s) generated (unless masked away):

When the Read_Inquiry_Mode command has completed, a Command Complete event shall be generated.

7.3.50 Write Inquiry Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Inquiry_Mode	0x0045	Inquiry_Mode	Status

Description:

This command writes the Inquiry_Mode configuration parameter of the local BR/EDR Controller. See Section 6.5 on page 436.

Command Parameters:

Inquiry_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Standard Inquiry Result event format (default)
0x01	Inquiry Result format with RSSI
0x02[	Inquiry Result with RSSI format or Extended Inquiry Result format
0x03-0xFF	Reserved

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Inquiry_Mode command succeeded.
0x01-0xFF	Write_Inquiry_Mode command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Inquiry_Mode command has completed, a Command Complete event shall be generated.

7.3.51 Read Page Scan Type Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Page_Scan_Type	0x0046		Status, Page_Scan_Type

Description:

This command reads the Page Scan Type configuration parameter of the local BR/EDR Controller. See Section 6.11 on page 439. For details, see the Baseband Specification, "Part B, Page Scan Substate" .

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Page_Scan_Type command succeeded.
0x01-0xFF	Read_Page_Scan_Type command failed. See Part D, Error Codes for a list of error codes and descriptions.

Page_Scan_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Mandatory: Standard Scan (default)
0x01	Optional: Interlaced Scan
0x02-0xFF	Reserved

Event(s) generated (unless masked away):

When the Read_Page_Scan_Type command has completed, a Command Complete event shall be generated.

7.3.52 Write Page Scan Type Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Page_Scan_Type	0x0047	Page_Scan_Type	Status

Description:

This command writes the Page Scan Type configuration parameter of the local BR/EDR Controller. See Section 6.11 on page 439. For details, see the Baseband Specification, “Part B, Page Scan Substate” .

Command Parameters:

Page_Scan_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Mandatory: Standard Scan (default)
0x01	Optional: Interlaced Scan
0x02-0xFF	Reserved

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Page_Scan_Type command succeeded.
0x01-0xFF	Write_Page_Scan_Type command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Page_Scan_Type command has completed, a Command Complete event shall be generated.

7.3.53 Read AFH Channel Assessment Mode Command

Command	OCF	Command Parameters	Return Parameters
Read_AFH_Channel_Assessment_Mode	0x0048		Status, AFH_Channel_Assessment_Mode

Description:

The Read_AFH_Channel_Assessment_Mode command reads the value for the AFH_Channel_Assessment_Mode parameter. The AFH_Channel_Assessment_Mode parameter controls whether the controller's channel assessment scheme is enabled or disabled.

This command shall be supported by a device that declares support for any of the AFH_capable_master, AFH_classification_slave or AFH_classification_master features.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_AFH_Channel_Assessment_Mode command succeeded.
0x01-0xFF	Read_AFH_Channel_Assessment_Mode command failed. See Part D, Error Codes for a list of error codes and descriptions.

AFH_Channel_Assessment_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Controller channel assessment disabled.
0x01	Controller channel assessment enabled.
0x02-0xFF	Reserved for future use.

Event(s) generated (unless masked away):

When the Read_AFH_Channel_Assessment_Mode command has completed, a Command Complete event shall be generated.

7.3.54 Write AFH Channel Assessment Mode Command

Command	OCF	Command Parameters	Return Parameters
Write_AFH_Channel _Assessment_Mode	0x0049	AFH_Channel _Assessment_Mode	Status

Description:

The Write_AFH_Channel_Assessment_Mode command writes the value for the AFH_Channel_Assessment_Mode parameter. The AFH_Channel_Assessment_Mode parameter controls whether the Controller's channel assessment scheme is enabled or disabled.

Disabling channel assessment forces all channels to be unknown in the local classification, but does not affect the AFH_reporting_mode or support for the Set_AFH_Host_Channel_Classification command. A slave in the AFH_reporting_enabled state shall continue to send LMP channel classification messages for any changes to the channel classification caused by either this command (altering the AFH_Channel_Assessment_Mode) or HCI Set_AFH_Host_Channel_Classification command (providing a new channel classification from the Host).

This command shall be supported by a device that declares support for any of the AFH_capable_master, AFH_classification_slave or AFH_classification_master features.

If the AFH_Channel_Assessment_Mode parameter is enabled and the BR/EDR Controller does not support a channel assessment scheme, other than via the Set_AFH_Host_Channel_Classification command, then a Status parameter of 'Channel Assessment Not Supported' should be returned. See "Part D, Error Codes" for a list of error codes and descriptions.

If the BR/EDR Controller supports a channel assessment scheme then the default AFH_Channel_Assessment_Mode is enabled, otherwise the default is disabled.

Command Parameters:

AFH_Channel_Assessment_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Controller channel assessment disabled.
0x01	Controller channel assessment enabled.
0x02-0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_AFH_Channel_Assessment_Mode command succeeded.
0x01-0xFF	Write_AFH_Channel_Assessment_Mode command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_AFH_Channel_Assessment_Mode command has completed, a Command Complete event shall be generated.

7.3.55 Read Extended Inquiry Response Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Extended_Inquiry_Response	0x0051		Status, FEC_Required, Extended_Inquiry_Response

Description:

The Read_Extended_Inquiry_Response command reads the extended inquiry response to be sent during the extended inquiry response procedure. The FEC_Required parameter states if FEC encoding is required.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Extended_Inquiry_Response command succeeded.
0x01 - 0xFF	Read_Extended_Inquiry_Response command failed. See Part D, Error Codes for a list of error codes and descriptions.

FEC_Required: *Size: 1 Octet*

Value	Parameter Description
0x00	FEC is not required
0x01	FEC is required
0x02-0xFF	Reserved

Extended_Inquiry_Response: *Size: 240 Octets*

Value	Parameter Description
	Extended inquiry response data as defined in [Vol 3] Part C, Section 8, Extended Inquiry Response Data Format.

Event(s) generated (unless masked away):

When the Read_Extended_Inquiry_Response command has completed, a Command Complete event shall be generated.

7.3.56 Write Extended Inquiry Response Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Extended_Inquiry_Response	0x0052	FEC_Required, Extended_Inquiry_Response	Status

Description:

The Write_Extended_Inquiry_Response command writes the extended inquiry response to be sent during the extended inquiry response procedure. The FEC_Required command parameter states if FEC encoding is required. The extended inquiry response data is not preserved over a reset. The initial value of the inquiry response data is all zero octets. The controller shall not interpret the extended inquiry response data.

Command Parameters:

FEC_Required: *Size: 1 Octet*

Value	Parameter Description
0x00	FEC is not required
0x01	FEC is required
0x02-0xFF	Reserved

Extended Inquiry Response: *Size: 240 Octets*

Value	Parameter Description
	Extended inquiry response data as defined in [Vol 3] Part C, Section 8, Extended Inquiry Response Data Format.
	All octets zero (**default**).

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Extended_Inquiry_Response command succeeded
0x01-0xFF	Write_Extended_Inquiry_Response command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Extended_Inquiry_Response command has completed, a Command Complete event shall be generated.

7.3.57 Refresh Encryption Key Command

Command	OCF	Command Parameters	Return Parameters
HCI_Refresh_ Encryption_Key	0x0053	Connection_Handle	

Description:

This command is used by the Host to cause the BR/EDR Controller to refresh the encryption key by pausing and resuming encryption.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection Handle for the ACL connection to have the encryption key refreshed on.

Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Event(s) generated (unless masked away):

A Command_Status event is sent from the BR/EDR Controller to the Host when the Controller has started the Refresh Encryption Key procedure. An Encryption Key Refresh Complete event shall be generated when the Refresh Encryption Key procedure has completed.

7.3.58 Read Simple Pairing Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Simple_Pairing_Mode	0x0055		Status, Simple_Pairing_Mode

Description:

This command reads the Simple_Pairing_Mode parameter in the BR/EDR Controller.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Simple_Pairing_Mode command succeeded.
0x01 - 0xFF	Read_Simple_Pairing_Mode command failed. See Part D, Error Codes for a list of error codes and descriptions.

Simple_Pairing_Mode *Size: 1Octet*

Value	Parameter Description
0x00	Simple Pairing not set (default)
0x01	Simple pairing enabled
0x02 - 0xFF	Reserved for future use

Event(s) generated (unless masked away):

When the Read_Simple_Pairing_Mode command has completed, a Command Complete event shall be generated.

7.3.59 Write Simple Pairing Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Simple_Pairing_Mode	0x0056	Simple_Pairing_Mode	Status

Description:

This command enables Simple Pairing mode in the BR/EDR Controller. When Simple Pairing Mode is set to 'enabled' the Link Manager shall respond to an LMP_io_capability_req PDU with an LMP_io_capability_res PDU and continue with the subsequent pairing procedure. When Simple Pairing mode is set to 'disabled', the Link Manager shall reject an IO capability request. A Host shall not set the Simple Pairing Mode to ‘disabled.’

Until Write_Simple_Pairing_Mode is received by the BR/EDR Controller, it shall not support any Simple Pairing sequences, and shall respond with the error code Simple Pairing not Supported by Host. This command shall be written before initiating page scan or paging procedures.

The Link Manager Secure Simple Pairing (Host Support) feature bit shall be set to the Simple_Pairing_Mode parameter. The default value for Simple_Pairing_Mode shall be 'disabled.' When Simple_Pairing_Mode is set to 'enabled,' the bit in the LMP features mask indicating support for Secure Simple Pairing (Host Support) shall be set to enabled in subsequent responses to an LMP_feature_req from a remote device.

Command Parameters:

Simple_Pairing_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Simple Pairing disabled (**default**)
0x01	Simple pairing enabled
0x02 - 0xFF	Reserved for future use

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Simple_Pairing_Mode command succeeded.
0x01 - 0xFF	Write_Simple_Pairing_Mode command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Simple_Pairing_Mode command has completed, a Command Complete event shall be generated.

7.3.60 Read Local OOB Data Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Local_OOB_Data	0x0057		Status, C, R

Description:

This command obtains a Simple Pairing Hash C and Simple Pairing Randomizer R which are intended to be transferred to a remote device using an OOB mechanism. The BR/EDR Controller shall create new values for C and R for each invocation of this command.

Note: Each OOB transfer will have unique C and R values so after each OOB transfer this command shall be used to obtain a new set of values for the next OOB transfer.

Note: The controller keeps information used to generate these values for later use in the simple pairing process. If the BR/EDR Controller is powered off or reset then this information is lost and the values obtained before the power off or reset are invalid.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Local_OOB_Data command succeeded.
0x01 - 0xFF	Read_Local_OOB_Data command failed. See Part D, Error Codes for a list of error codes and descriptions.

C: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXX XXXXXXXXXXX XXXXXXXXXX	Simple Pairing Hash C

R: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXX XXXXXXXXXXX XXXXXXXXXX	Simple Pairing Randomizer R

Event(s) generated (unless masked away):

When the Read_Local_OOB_Data command has completed, a Command Complete event shall be generated.

7.3.61 Read Inquiry Response Transmit Power Level Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Inquiry _Response_Transmit _Power_Level	0x0058		Status, TX_Power

Description:

This command reads the inquiry Transmit Power level used to transmit the FHS and EIR data packets. This can be used directly in the Tx Power Level EIR data type.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Inquiry_Response_Transmit_Power_Level command succeeded.
0x01 - 0xFF	Read_Inquiry_Response_Transmit_Power_Level command failed. See Part D, Error Codes for a list of error codes and descriptions.

TX Power: *Size: 1 Octet*

Value	Parameter Description
0xXX	Size: 1 Octet (signed integer) Range: $-70 \leq N \leq 20$ Units: dBm

Event(s) generated (unless masked away):

When the Read_Inquiry_Response_Transmit_Power_Level command has completed, a Command Complete event shall be generated.

7.3.62 Write Inquiry Transmit Power Level Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Inquiry_Transmit_Power_Level	0x0059	TX_Power	Status

Description:

This command writes the inquiry transmit power level used to transmit the inquiry (ID) data packets. The Controller should use the supported TX power level closest to the Tx_Power parameter.

Command Parameters:

None.

TX Power: Size: 1 Octet

Value	Parameter Description
0xXX	Size: 1 Octet (signed integer) Range: $-70 \leq N \leq 20$ Units: dBm

When the Write_Inquiry_Transmit_Power_Level command has completed, a Command Complete event shall be generated.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Inquiry_Transmit_Power_Level command succeeded
0x01 - 0xFF	Write_Inquiry_Transmit_Power_Level command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Inquiry_Transmit_Power_Level command has completed, a Command Complete event shall be generated.

7.3.63 Send Keypress Notification Command

Command	OCF	Command Parameters	Return Parameters
HCI_Send_Keypress_Notification	0x0060	BD_ADDR, Notification_Type	Status, BD_ADDR

Description:

This command is used during the Passkey Entry protocol by a device with KeyboardOnly IO capabilities. It is used by a Host to inform the remote device when keys have been entered or erased.

Command Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Notification_Type: *Size: 1 Octet*

Value	Parameter Description
0	Passkey entry started
1	Passkey digit entered
2	Passkey digit erased
3	Passkey cleared
4	Passkey entry completed
5-255	Reserved for future use

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Send_Keypress_Notification command succeeded
0x01 - 0xFF	Send_Keypress_Notification command failed

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Event(s) generated (unless masked away):

When the Send_Keypress_Notification command has completed, a Command Complete event shall be generated.

7.3.64 Read Default Erroneous Data Reporting

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Default_ Erroneous_Data_Reporting	0x005A		Status, Erroneous_Data_Reporting

Description:

This command reads the Erroneous_Data_Reporting parameter.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Default_Erroneous_Data_Reporting command succeeded.
0x01 - 0xFF	Read_Default_Erroneous_Data_Reporting command failed. See Part D, Error Codes for a list of error codes and descriptions.

Erroneous_Data_Reporting: *Size: 1 Octet*

Value	Parameter Description
0x00	Erroneous data reporting disabled.
0x01	Erroneous data reporting enabled.
0x02 - 0xFF	Reserved.

Event(s) generated (unless masked away):

When the Read_Default_Erroneous_Data_Reporting command has completed, a Command Complete event shall be generated.

7.3.65 Write Default Erroneous Data Reporting

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Default_ Erroneous_Data_Reporting	0x005B	Erroneous_Data_Reporting	Status

Description:

This command writes the Erroneous_Data_Reporting parameter. The BR/EDR Controller shall set the Packet_Status_Flag as defined in Section 5.4.3, “HCI Synchronous Data Packets,” on page 431, depending on the value of this parameter. The new value for the Erroneous_Data_Reporting parameter shall not apply to existing synchronous connections.

Command Parameters:

Erroneous_Data_Reporting: *Size: 1 Octet*

Value	Parameter Description
0x00	Erroneous Data reporting disabled(**default**).
0x01	Erroneous data reporting enabled.
0x02 - 0xFF	Reserved.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Default_Erroneous_Data_Reporting command succeeded.
0x01 - 0xFF	Write_Default_Erroneous_Data_Reporting command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Default_Erroneous_Data_Reporting command has completed, a Command Complete event shall be generated.

7.3.66 Enhanced Flush Command

Command	OCF	Command Parameters	Return Parameters
HCI_Enhanced_Flush	0x005F	Handle, Packet_Type	

Description:

The Enhanced_Flush command is used to discard all L2CAP packets identified by Packet_Type that are currently pending for transmission in the Controller for the specified Handle, even if there currently are chunks of data that belong to more than one L2CAP packet of the same type in the Controller. The only packet type defined is automatically-flushable. Packets not identified by Packet_Type will not be flushed and will be processed normally by the Controller.

After flushing the packets, all data that is sent to the BR/EDR Controller for the same Handle and packet type shall be discarded by the Controller until an HCI Data Packet with the start Packet_Boundary_Flag (0x00 or 0x02) is received. This command allows higher-level software to control how long the baseband should try to retransmit a baseband packet of a specific type for a Handle before all data of that type currently pending for transmission in the Controller should be flushed. Note that the Enhanced Flush command is used for ACL-U connections only. On the BR/EDR Controller, the Flush command can be used to flush all packets (see Section 7.3.4 on page 571). In addition to the Enhanced Flush and Flush commands, the automatic flush timers (see Section 7.3.29 on page 602) can be used to automatically flush an automatically-flushable L2CAP packet that is currently being transmitted after the specified flush timer has expired.

This command shall be supported by a device that declares support for the Non-Flushable Packet Boundary Flag feature.

On an AMP Controller, in addition to the Enhanced_Flush command, the automatic flush timer specified in the Flow Specification for this logical link may be used to automatically flush packets on this logical link.

Command Parameters:

Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Handle to be used to identify a connection. The Handle is a Connection Handle for a BR/EDR Controller and a Logical Link Handle for an AMP Controller. Range: 0x0000 - 0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Packet_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Automatically flushable only.
0x01 - 0xFF	Reserved for future use.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Controller receives the Enhanced_Flush command, the Controller shall send the Command Status event to the host. In addition, when all the packets identified by Packet_Type have been flushed for the specified Handle, the Controller shall send an Enhanced Flush Complete event to the host. The controller may send the Enhanced Flush Complete event immediately after flushing all the packets of type Packet_Type for the specified Handle, or it may wait until all packets for the specified Handle, independent of Packet_Type, buffered in the controller at the time of the receipt of the Enhanced Flush Command, have been either flushed or transmitted.

Note: No Command Complete event will be sent by the Controller to indicate that this command has been completed. Instead, the Enhanced Flush Complete event will indicate that this command has been completed.

7.3.67 Read Logical Link Accept Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Read_Logical_Link_Accept_Timeout	0x0061		Status, Logical_Link_Accept_Timeout

Description:

This command reads the value for the Logical_Link_Accept_Timeout configuration parameter.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Logical_Link_Accept_Timeout command succeeded.
0x01-0xFF	Read_Logical_Link_Accept_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Logical_Link_Accept_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Logical_Link Accept Timeout measured in Number of BR/EDR Baseband slots. Interval Length = N * 0.625 msec (1 BR/EDR Baseband slot) Range for N: 0x0001 - 0xB540 Time Range: 0.625 msec -29 seconds

Event(s) generated (unless masked away):

When the Read_Logical_Link_Timeout command has completed, a Command Complete event shall be generated.

7.3.68 Write Logical Link Accept Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Logical_Link _Accept_Timeout	0x0062	Logical_Link_Accept_ Timeout	Status

Description:

This command writes the value for the Logical_Link_Accept_Timeout configuration parameter. See Logical Link Accept Timeout.

Command Parameters:

Logical_Link_Accept_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Logical_Link Accept Timeout measured in Number of BR/EDR Baseband slots. Interval Length = N * 0.625 msec (1 BR/EDR Baseband slot) Range for N: 0x0001 - 0xB540 Time Range: 0.625 msec - 29 seconds Default: N = 0x1FA0 Time = 5.06 Sec Mandatory Range for Controller: 0x00A0 to 0xB540

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Logical_Link_Accept_Timeout command succeeded.
0x01-0xFF	Write_Logical_Link_Accept_Timeout command failed. See Part D, Error Codesor a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Logical_Link_Accept_Timeout command has completed, a Command Complete event shall be generated.

7.3.69 Set Event Mask Page 2 Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Set_Event_Mask_Page_2	0x0063	Event_Mask_Page_2	Status

Description:

The Set_Event_Mask_Page_2 command is used to control which events are generated by the HCI for the Host. The Event_Mask_Page_2 is a logical extension to the Event_Mask parameter of the Set_Event_Mask command. If the bit in the Event_Mask_Page_2 is set to a one, then the event associated with that bit shall be enabled. The Host has to deal with each event that occurs by the Controllers. The event mask allows the Host to control how much it is interrupted.

Command Parameters:

Event_Mask_Page_2: *Size: 8 Octets*

Value	Parameter Description
0x0000000000000000	No events specified (**default**)
0x0000000000000001	Physical Link Complete Event
0x0000000000000002	Channel Selected Event
0x0000000000000004	Disconnection Physical Link Event
0x0000000000000008	Physical Link Loss Early Warning Event
0x0000000000000010	Physical Link Recovery Event
0x0000000000000020	Logical Link Complete Event
0x0000000000000040	Disconnection Logical Link Complete Event
0x0000000000000080	Flow Spec Modify Complete Event
0x0000000000000100	Number of Completed Data Blocks Event
0x0000000000000200	AMP Start Test Event
0x0000000000000400	AMP Test End Event
0x0000000000000800	AMP Receiver Report Event
0x0000000000001000	Short Range Mode Change Complete Event
0x0000000000002000	AMP Status Change Event
0xFFFFFFFFFFFFC000	Reserved for future use

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Set_Event_Mask_Page_2 command succeeded.
0x01-0xFF	Set_Event_Mask_Page_2 command failed. See Part D, Error Codesfor error codes and descriptions.

Event(s) generated (unless masked away):

When the Set_Event_Mask_Page_2 command has completed, a Command Complete event shall be generated.

7.3.70 Read Location Data Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Read_Location_Data	0x0064		Status, Location_Domain_Aware, Location_Domain, Location_Domain Options, Location_Options

Description:

The Read_Location_Data command provides the ability to read any stored knowledge of environment or regulations or currently in use in the AMP Controller. See Section 6.29 to Section 6.32 on page 455.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Location_Data command succeeded.
0x01-0xFF	Read_Location_Data command failed. See " Part D, Error Codes for error codes and descriptions.

See Section 6.29 to Section 6.32 for Location_Domain_Aware, Location_Domain, Location_Domain_Options and Location_Options respectively.

Event(s) generated (unless masked away):

When the Read_Location_Data command has completed, a Command Complete event shall be generated.

7.3.71 Write Location Data Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Write_Location_Data	0x0065	Location_Domain_Aware, Location_Domain, Location_Domain_Options, Location_Options	Status

Description:

The Write_Location_Data command writes information about the environment or regulations currently in force, which may affect the operation of the Controller. The Controller may use this information to optimize its operation and to ensure its conformance to currently applicable regulations. The exact way in which this affects AMP operation is AMP type specific.

Command Parameters:

See Section 6.29 to Section 6.32 for Location_Domain_Aware, Location_Domain, Location_Domain_Options, Location_Options.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Location_Data command succeeded.
0x01-0xFF	Write_Location_Data command failed. See Part D, Error Codes for error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Location_Data command has completed, a Command Complete event shall be generated.

7.3.72 Read Flow Control Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Read_Flow_Control_Mode	0x0066		Status, Flow_Control_Mode

Description:

This command reads the value for the Flow_Control_Mode configuration parameter. See Section 6.33 on page 456.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Flow_Control_Mode command succeeded.
0x01-0xFF	Read_Flow_Control_Mode command failed. See Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Read_Flow_Control_Mode command has completed, a Command Complete event shall be generated.

7.3.73 Write Flow Control Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Flow_Control_Mode	0x0067	Flow_Control_Mode	Status

Description:

This command writes the value for the Flow_Control_Mode configuration parameter. See Section 6.33 on page 456.

Command Parameters:

Flow_Control_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Packet based data flow control mode (**default for a BR/EDR Controller**)
0x01	Data block based data flow control mode (**default for an AMP Controller**)
0x02-0xFF	Reserved for future use

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Flow_Control_Mode command succeeded.
0x01-0xFF	Write_Flow_Control_Mode command failed. See Part D, Error Codesfor a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Flow_Control_Mode command has completed, a Command Complete event shall be generated. If the set fails then the Controller continues using its current mode.

7.3.74 Read Enhanced Transmit Power Level Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Enhance_Transmit_Power_Level	0x0068	Connection_Handle, Type	Status, Connection_Handle, Transmit_Power_Level_GFSK, Transmit_Power_Level_DQPSK, Transmit_Power_Level_8DPSK

Description:

This command reads the values for the Enhanced_Transmit_Power_Level parameters for the specified Connection Handle. The Connection_Handle must be a Connection_Handle for an ACL connection.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000 - 0x0EFF (0x0F00 - 0x0FFF Reserved for future use).

Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Read Current Transmit Power Level
0x01	Read Maximum Transmit Power Level
0x01-0xFF	Reserved.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Enhanced_Transmit_Power_Level command succeeded.
0x01-0xFF	Read_Enhanced_Transmit_Power_Level command failed. See Part D, Error Codes for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range: 0x0000 - 0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Transmit_Power_Level_GFSK: *Size: 1 Octet*

Value	Parameter Description
0xXX	Size: 1 Octet (signed integer) Range: -100 ≤ N ≤ 20 Units: dBm

Transmit_Power_Level_DQPSK: *Size: 1 Octet*

Value	Parameter Description
0xXX	Size: 1 Octet (signed integer) Range: -100 ≤ N ≤ 20 Units: dBm

Transmit_Power_Level_8DPSK: *Size: 1 Octet*

Value	Parameter Description
0x00	Size: 1 Octet (signed integer) Range: -100 ≤ N ≤ 20 Units: dBm

Event(s) generated (unless masked away):

When the Read_Enhanced_Transmit_Power_Level command has completed, a Command Complete event shall be generated.

7.3.75 Read Best Effort Flush Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Read_Best_Effort_ Flush_Timeout	0x0069	Logical_Link_Handle	Status, Best_Effort_Flush_Timeout

Description:

This command reads the value of the Best Effort Flush Timeout from the AMP Controller.

Command Parameters:

Logical_Link_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Logical_Link_Handle to which the Write_Best_Effort_Flush_Timeout command applies. Range: 0x0000-0xEFF (0x0F00 - 0x0FF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Best_Effort_Flush_Timeout command succeeded.
0x01-0xFF	Read_Best_Effort_Flush_Timeout command failed. See Part D, Error Codes for a list of error codes and descriptions.

Best_Effort_Flush_Timeout: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	0x00000000 - 0xFFFFFFFE: Best Effort Flush Timeout value in microseconds 0xFFFFFFFF: No Best Effort Flush Timeout used (default)

Event(s) generated (unless masked away):

When the Read_Best_Effort_Flush_Timeout command has completed, a Command Complete event shall be generated.

7.3.76 Write Best Effort Flush Timeout Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Write_Best_Effort_Flush_ Timeout	0x006A	Logical_Link_Handle, Best_Effort_Flush_Timeout	Status

Description:

This command writes the value of the Best_Effort_Flush_Timeout into the AMP Controller. The Best_Effort_Flush_Timeout shall be associated with the given logical link, and that Logical_Link_Handle must specify a Best Effort logical link.

Command Parameters:

Logical_Link_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Logica_Link_Handle to which the Write_Best_Effort_Flush_Timeout command applies. Range: 0x0000-0xEFF (0x0F00 - 0x0FF Reserved for future use)

Best_Effort_Flush_Timeout: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	0x00000000 - 0xFFFFFFFE: Best_Effort_Flush_Timeout value in microseconds 0xFFFFFFFF: No Best Effort Flush Timeout used (default)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Best_Effort_Flush_Timeout command succeeded.
0x01-0xFF	Write_Best_Effort_Flush_Timeout command failed. See " Part D, Error Codes for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Best_Effort_Flush_Timeout command has completed, a Command Complete event shall be generated.

7.3.77 Short Range Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Short_Range_Mode	0x006B	Physical_Link_Handle, Short_Range_Mode	Status

Description:

This command will configure the value of Short_Range_Mode to the AMP Controller and is AMP type specific (see Volume 5,).

Command Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle to which the Short Range Mode command applies.

Short_Range_Mode: *Size: 1 Octet*

Value	Parameter Description
0xXX	Configuration setting of Short Range Mode 0x00 - Short Range Mode disabled (**default**) 0x01 - Short Range Mode enabled 0x02 - 0xFF - Reserved

Return Parameters:

None.

Event(s) generated (unless masked away):

A Command Status event is sent from the AMP Controller to the Host when the AMP Controller has received the Short_Range_Mode command. When the Short_Range_Mode command has completed, a Short Range Mode Change Complete event shall be generated.

7.3.78 Read LE Host Supported Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_LE_Host_ Support	0x006C		Status, LE_Supported_Host, Simultaneous_LE_Host

Description:

The Read_LE_Host_Support command is used to read the LE Supported (Host) and Simultaneous LE and BR/EDR to Same Device Capable (Host) Link Manager Protocol feature bits. These Link Manager Protocol feature bits are used by the local host and a remote device (Controller and Host). See Part C, Section 3.2, Feature Definitions.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_LE_Host_Support command succeeded.
0x01 - 0xFF	Read_LE_Host_Support command failed. See "Part D, Error Codes on page 339 for a list of error codes and descriptions.

LE_Supported_Host: *Size: 1 Octet*

Value	Parameter Description
0xXX	LE_Supported_Host parameter, see Section 6.33

Simultaneous_LE_Host: *Size: 1 Octet*

Value	Parameter Description
0xXX	Simultaneous_LE_Host parameter, see Section 6.35

Event(s) Generated (unless masked away):

When the Read_LE_Host_Support command has completed, a Command Complete event shall be generated.

7.3.79 Write LE Host Supported Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_LE_Host_Support	0x006D	LE_Supported_Host, Simultaneous_LE_Host	Status

Description:

The Write_LE_Host_Support command is used to set the LE Supported (Host) and Simultaneous LE and BR/EDR to Same Device Capable (Host) Link Manager Protocol feature bits. These Link Manager Protocol feature bits are used by a remote Host. See Part C, Section 3.2, Feature Definitions.

The default value for these feature bits shall be disabled. When LE_Supported_Host is set to enabled the bit in LMP features mask indicating support for LE Support (Host) shall be set. When the Simultaneous_LE_Host parameter is set to enabled the bit in LMP features mask indicating support for Simultaneous LE and BR/EDR to Same Device Capable (Host) shall be set. A Host shall not set the LE_Supported_Host or Simultaneous_LE_Host parameters to disabled if they have been set to enabled.

Command Parameters:

LE_Supported_Host: *Size: 1 Octet*

Value	Parameter Description
0xXX	LE_Supported_Host parameter. See Section 6.33

Simultaneous_LE_Host: *Size: 1 Octet*

Value	Parameter Description
0xXX	Simultaneous_LE_Host parameter. See Section 6.35

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_LE_Host_Support command succeeded.
0x01 - 0xFF	Write_LE_Host_Support command failed. See "Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the Write_LE_Host_Support command has completed, a Command Complete event shall be generated.

7.4 INFORMATIONAL PARAMETERS

The Informational Parameters are fixed by the manufacturer of the Bluetooth hardware. These parameters provide information about the BR/EDR Controller and the capabilities of the Link Manager and Baseband in the BR/EDR Controller and PAL in the AMP Controller. The host device cannot modify any of these parameters.

For Informational Parameters Commands, the OGF is defined as 0x04.

7.4.1 Read Local Version Information Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Local_Version_ Information	0x0001		Status, HCI Version, HCI Revision, LMP Version, Manufacturer_Name, LMP Subversion

Description:

This command reads the values for the version information for the local Controller.

The HCI Version information defines the version information of the HCI layer. The LMP/PAL Version information defines the version of the LMP or PAL. The Manufacturer_Name information indicates the manufacturer of the local device.

The HCI Revision and LMP/PAL Subversion are implementation dependent.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Local_Version_Information command succeeded.

0x01-0xFF	Read_Local_Version_Information command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

HCI_Version: *Size: 1 Octet*

Value	Parameter Description
See Bluetooth Assigned Numbers	

HCI_Revision: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Revision of the Current HCI in the BR/EDR Controller.

LMP/PAL_Version: *Size: 1 Octet*

Value	Parameter Description
0xXX	Version of the Current LMP or PAL in the Controller. See Bluetooth Assigned Numbers

Manufacturer_Name: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Manufacturer Name of the BR/EDR Controller. See Bluetooth Assigned Numbers

LMP/PAL_Subversion: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Subversion of the Current LMP or PAL in the Controller. This value is implementation dependent.

Event(s) generated (unless masked away):

When the Read_Local_Version_Information command has completed, a Command Complete event shall be generated.

7.4.2 Read Local Supported Commands Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Read_Local_Supported_Commands	0x0002		Status, Supported_Commands

Description:

This command reads the list of HCI commands supported for the local Controller.

This command shall return the Supported_Commands configuration parameter. It is implied that if a command is listed as supported, the feature underlying that command is also supported.

See Section 6.27 on page 447 for more information.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0	Read_Local_Supported_Commands command succeeded
0x01-0xff	Read_Local_Supported_Commands command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Supported Commands: *Size: 64 Octets*

Value	Parameter Description
	Bit mask for each HCI Command. If a bit is 1, the Controller supports the corresponding command and the features required for the command. Unsupported or undefined commands shall be set to 0. See section 6.27, “Supported Commands,” on page 447.

Event(s) generated (unless masked away):

When the Read_Local_Supported_Commands command has completed, a Command Complete event shall be generated.

7.4.3 Read Local Supported Features Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Local_Supported_Features	0x0003		Status, LMP_Features

Description:

This command requests a list of the supported features for the local BR/EDR Controller. This command will return a list of the LMP features. For details see Part C, Link Manager Protocol Specification on page 207.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Local_Supported_Features command succeeded.
0x01-0xFF	Read_Local_Supported_Features command failed. See Part D, Error Codes on page 339.

LMP_Features: *Size: 8 Octets*

Value	Parameter Description
0xXXXXXXXX XXXXXXXX	Bit Mask List of LMP features. For details see Part C, Link Manager Protocol Specification on page 207.

Event(s) generated (unless masked away):

When the Read_Local_Supported_Features command has completed, a Command Complete event shall be generated.

7.4.4 Read Local Extended Features Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Local_ Extended_Features	0x0004	Page number	Status, Page number, Maximum Page Number, Extended_LMP_Features

Description:

The Read_Local_Extended_Features command returns the requested page of the extended LMP features.

Command Parameters:

Page Number: *Size: 1 Octet*

Value	Parameter Description
0x00	Requests the normal LMP features as returned by Read_Local_Supported_Features.
0x01-0xFF	Return the corresponding page of features.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Local_Extended_Features command succeeded
0x01-0xFF	Read_Local_Extended_Features command failed. See Part D, Error Codes on page 339 for list of error codes.

Page Number: *Size: 1 Octet*

Value	Parameter Description
0x00	The normal LMP features as returned by Read_Local_Supported_Features.
0x01-0xFF	The page number of the features returned.

Maximum Page Number: *Size: 1 Octet*

Value	Parameter Description
0x00-0xFF	The highest features page number which contains non-zero bits for the local device.

Extended_LMP_Features: *Size: 8 Octets*

Value	Parameter Description

0xFFFFFFFFFFFFFFFF	Bit map of requested page of LMP features. See LMP specification for details.

Event(s) generated (unless masked away):

When the Read_Local_Extended_Features command has completed, a Command Complete event shall be generated.

7.4.5 Read Buffer Size Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Buffer_Size	0x0005		Status, HC_ACL_Data_Packet_Length, HC_Synchronous_Data_Packet_Length, HC_Total_Num_ACL_Data_Packets, HC_Total_Num_Synchronous_Data_Packets

Description:

The Read_Buffer_Size command is used to read the maximum size of the data portion of HCI ACL and synchronous Data Packets sent from the Host to the Controller. The Host will segment the data to be transmitted from the Host to the Controller according to these sizes, so that the HCI Data Packets will contain data with up to these sizes. The Read_Buffer_Size command also returns the total number of HCI ACL and synchronous Data Packets that can be stored in the data buffers of the Controller. The Read_Buffer_Size command must be issued by the Host before it sends any data to the Controller.

For a device supporting BR/EDR and LE, if the LE_Read_Buffer_Size command returned zero for the number of buffers, then buffers returned by Read_Buffer_Size are shared between BR/EDR and LE.

On an Primary Controller that supports LE only, the Read_Buffer_Size command shall not be supported (the LE_Read_Buffer_Size command is to be used in this case).

The HC_ACL_Data_Packet_Length return parameter will be used to determine the size of the L2CAP segments contained in ACL Data Packets, which are transferred from the Host to the Controller to be broken up into baseband packets by the Link Manager. The HC_Synchronous_Data_Packet_Length return parameter is used to determine the maximum size of HCI synchronous Data Packets. Both the Host and the Controller must support command and event packets, where the data portion (excluding header) contained in the packets is 255 octets in size. The HC_Total_Num_ACL_Data_Packets return parameter contains the total number of HCI ACL Data Packets that can be stored in the data buffers of the Controller. The Host will determine how the buffers are to be divided between different Connection Handles. The HC_Total_Num_Synchronous_Data_Packets return parameter gives the same information but for HCI synchronous Data Packets.

Note: The HC_ACL_Data_Packet_Length and HC_Synchronous_Data_Packet_Length return parameters do not include the length of the HCI Data Packet header.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Buffer_Size command succeeded.
0x01-0xFF	Read_Buffer_Size command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

HC_ACL_Data_Packet_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Maximum length (in octets) of the data portion of each HCI ACL Data Packet that the Controller is able to accept.

HC_Synchronous_Data_Packet_Length: *Size: 1 Octet*

Value	Parameter Description
0xXX	Maximum length (in octets) of the data portion of each HCI Synchronous Data Packet that the Controller is able to accept.

HC_Total_Num_ACL_Data_Packets: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Total number of HCI ACL Data Packets that can be stored in the data buffers of the Controller.

HC_Total_Num_Synchronous_Data_Packets: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Total number of HCI Synchronous Data Packets that can be stored in the data buffers of the Controller.

Event(s) generated (unless masked away):

When the Read_Buffer_Size command has completed, a Command Complete event shall be generated.

7.4.6 Read BD_ADDR Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_BD_ADDR	0x0009		Status, BD_ADDR

Description:

On a BR/EDR Controller, this command reads the Bluetooth Controller address (BD_ADDR). See the Part B, Baseband Specification on page 59 for details of how BD_ADDR is used (see [Vol 3] Part C, Section 3.2.1).

On an LE Controller, this command shall read the Public Device Address as defined in Part B, Section 4.1, General. If this Controller does not have a Public Device Address, the value 0x000000000000 shall be returned.

On a BR/EDR/LE Controller, the public address shall be the same as the BD_ADDR.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_BD_ADDR command succeeded.
0x01-0xFF	Read_BD_ADDR command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device

Event(s) generated (unless masked away):

When the Read_BD_ADDR command has completed, a Command Complete event shall be generated.

7.4.7 Read Data Block Size Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Read_Data_ Block_Size	0x000A		Status, Max_ACL_Data_Packet_Length, Data_Block_Length, Total_Num_Data_Blocks

Description:

The Read_Data_Block_Size command is used to read values regarding the maximum permitted data transfers over the Controller and the data buffering available in the Controller.

The Host uses this information when fragmenting data for transmission, and when performing block-based flow control, based on the Number Of Completed Data Blocks event. The Read_Data_Block_Size command shall be issued by the Host before it sends any data to the Controller.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Data_Block_Size command succeeded.
0x01-0xFF	Read_Data_Block_Size command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Max_ACL_Data_Packet_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Maximum length (in octets) of the data portion of an HCI ACL Data Packet that the Controller is able to accept for transmission. For AMP Controllers this always equals to Max_PDU_Size.

Data_Block_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Maximum length (in octets) of the data portion of each HCI ACL Data Packet that the Controller is able to hold in each of its data block buffers.

Total_Num_Data_Blocks: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Total number of data block buffers available in the Controller for the storage of data packets scheduled for transmission.

Event(s) generated (unless masked away):

When the Read_Data_Block_Size command has completed, a Command Complete event shall be generated.

7.5 STATUS PARAMETERS

The Controller modifies all status parameters. These parameters provide information about the current state of the Link Manager and Baseband in the BR/EDR Controller and the PAL in an AMP Controller. The host device cannot modify any of these parameters other than to reset certain specific parameters.

For the Status Parameters Commands, the OGF is defined as 0x05.

7.5.1 Read Failed Contact Counter Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Failed_ Contact_Counter	0x0001	Handle	Status, Handle, Failed_Contact_Counter

Description:

This command reads the value for the Failed_Contact_Counter parameter for a particular connection to another device. The Handle shall be a Handle for an ACL connection. See Section 6.15 on page 440.

Command Parameters:

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	TheHandle for the Connection for which the Failed Contact Counter should be read. The Handle is a Connection_Handle for a BR/EDR Controller and a Logical_Link_Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Failed_Contact_Counter command succeeded.
0x01-0xFF	Read_Failed_Contact_Counter command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Handle for the connection for which the Failed Contact Counter has been read. The Handle is a Connection_Handle for a BR/EDR Controller and a Logical_Link_Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Failed_Contact_Counter: *Size: 2 Octets*

Value	Parameter Description

0xXXXX	Number of consecutive failed contacts for a connection corresponding to the Handle.

Event(s) generated (unless masked away):

When the Read_Failed_Contact_Counter command has completed, a Command Complete event shall be generated.

7.5.2 Reset Failed Contact Counter Command

Command	OCF	Command Parameters	Return Parameters
HCI_Reset_Failed_ Contact_Counter	0x0002	Handle	Status, Handle

Description:

This command resets the value for the Failed_Contact_Counter parameter for a particular connection to another device. The Handle shall be a Handle for an ACL connection. See Section 6.15 on page 440.

Command Parameters:

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Handle for the connection for which the Failed Contact Counter should be reset. The Handle is a Connection Handle for a BR/EDR Controller and a Logical Link Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Reset_Failed_Contact_Counter command succeeded.
0x01-0xFF	Reset_Failed_Contact_Counter command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Handle for the connection for which the Failed Contact Counter has been reset. The Handle is a Connection_Handle for a BR/EDR Controller and a Logical_Link_Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Event(s) generated (unless masked away):

When the Reset_Failed_Contact_Counter command has completed, a Command Complete event shall be generated.

7.5.3 Read Link Quality Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Link_Quality	0x0003	Handle	Status, Handle, Link_Quality

Description:

This command returns the value for the Link_Quality for the specified Handle. The Handle shall be a Handle for an ACL connection. This command shall return a Link_Quality value from 0-255, which represents the quality of the link between two BR/EDR Controllers. The higher the value, the better the link quality is. Each Bluetooth module vendor will determine how to measure the link quality.

The Read_Link_Quality command is provided by AMPs. The meaning of the link quality metric is AMP type specific and defined in the AMP PALs (see Volume 5, Core System Package [AMP Controller volume]).

Command Parameters:

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Handle for the connection for which link quality parameters are to be read. The Handle is a Connection_Handle for a BR/EDR Controller and a Physical_Link_Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Link_Quality command succeeded.
0x01-0xFF	Read_Link_Quality command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description

0xXXXX	The Handle for the connection for which the link quality parameter has been read. The Handle is a Connection_Handle for a BR/EDR Controller and a Physical_Link_Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Link_Quality: *Size: 1 Octet*

Value	Parameter Description
0xXX	The current quality of the Link connection between the local device and the remote device specified by the Handle. Range: 0x00 – 0xFF The higher the value, the better the link quality is.

Event(s) generated (unless masked away):

When the Read_Link_Quality command has completed, a Command Complete event shall be generated.

7.5.4 Read RSSI Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_RSSI	0x0005	Handle	Status, Handle, RSSI

Description:

This command reads the Received Signal Strength Indication (RSSI) value from a Controller.

For a BR/EDR Controller, a Connection_Handle is used as the Handle command parameter and return parameter. The RSSI parameter returns the difference between the measured Received Signal Strength Indication (RSSI) and the limits of the Golden Receive Power Range for a Connection Handle to another BR/EDR Controller. The Connection_Handle must be a Connection_Handle for an ACL connection. Any positive RSSI value returned by the Controller indicates how many dB the RSSI is above the upper limit, any negative value indicates how many dB the RSSI is below the lower limit. The value zero indicates that the RSSI is inside the Golden Receive Power Range.

Note: How accurate the dB values will be depends on the Bluetooth hardware. The only requirements for the hardware are that the BR/EDR Controller is able to tell whether the RSSI is inside, above or below the Golden Device Power Range.

The RSSI measurement compares the received signal power with two threshold levels, which define the Golden Receive Power Range. The lower threshold level corresponds to a received power between -56 dBm and 6 dB above the actual sensitivity of the receiver. The upper threshold level is 20 dB above the lower threshold level to an accuracy of +/- 6 dB.

For an AMP Controller, a Physical_Link_Handle is used for the Handle command parameter and return parameter. The meaning of the RSSI metric is AMP type specific and defined in the AMP PALs (see Volume 5, Core System Package [AMP Controller volume]).

For an LE transport, a Connection_Handle is used as the Handle command parameter and return parameter. The meaning of the RSSI metric is an absolute receiver signal strength value in dBm to ± 6 dBm accuracy. If the RSSI cannot be read, the RSSI metric shall be set to 127.

Command Parameters:

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description

0xXXXX	The Handle for the connection for which the RSSI is to be read. The Handle is a Connection_Handle for a BR/EDR Controller and a Physical_Link_Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_RSSI command succeeded.
0x01-0xFF	Read_RSSI command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Handle for the connection for which the RSSI has been read. The Handle is a Connection_Handle for a BR/EDR Controller and a Physical_Link_Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

RSSI: *Size: 1 Octet*

Value	Parameter Description
	BR/EDR Range: -128 ≤ N ≤ 127 (signed integer) Units: dB AMP: Range: AMP type specific (signed integer) Units: dBm LE: Range: -127 to 20, 127 (signed integer) Units: dBm

Event(s) generated (unless masked away):

When the Read_RSSI command has completed, a Command Complete event shall be generated.

7.5.5 Read AFH Channel Map Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_AFH _Channel_Map	0x0006	Connection_Handle	Status, Connection_Handle, AFH_Mode, AFH_Channel_Map

Description:

This command returns the values for the AFH_Mode and AFH_Channel_Map for the specified Connection_Handle. The Connection_Handle shall be a Connection_Handle for an ACL connection.

The returned values indicate the state of the hop sequence specified by the most recent LMP_Set_AFH message for the specified Connection_Handle, regardless of whether the master has received the baseband ACK or whether the AFH_Instant has passed.

This command shall be supported by a device that declares support for either the AFH_capable_slave or AFH_capable_master feature.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection_Handle for the connection for which the Channel Map is to be read. Range: 0x0000-0x0EFF (0x0F00-0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_AFH_Channel_Map command succeeded.
0x01-0xFF	Read_AFH_Channel_Map command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection_Handle for the connection for which the Channel Map is to be read. Range: 0x0000-0x0EFF (0x0F00-0x0FFF Reserved for future use)

AFH_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	AFH disabled.
0x01	AFH enabled.
0x02-0xFF	Reserved for future use.

AFH_Channel_Map: *Size: 10 Octets (79 Bits meaningful)*

Value	Parameter Description
0xXXXXXXXX XXXXXXXX XXX	If AFH_Mode is AFH enabled then this parameter contains 79 1-bit fields, otherwise the contents are reserved. The nth such field (in the range 0 to 78) contains the value for channel *n*: Channel *n* is unused = 0 Channel *n* is used = 1 Range: 0x00000000000000000000-0x7FFFFFFFFFFFFFFFFFFF (0x80000000000000000000-0xFFFFFFFFFFFFFFFFFFFF Reserved for future use)

Event(s) generated (unless masked away):

When the Read_AFH_Channel_Map command has completed, a Command Complete event shall be generated.

7.5.6 Read Clock Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Clock	0x0007	Connection_Handle Which_Clock	Status, Connection_Handle, Clock, Accuracy

Description:

This command reads the estimate of the value of the Bluetooth Clock from the BR/EDR Controller.

If the Which_Clock value is 0, then the Connection_Handle shall be ignored, the local Bluetooth Clock value shall be returned and the accuracy parameter shall be set to 0.

If the Which_Clock value is 1, then the Connection_Handle shall be a valid ACL Connection_Handle. If the current role of this ACL connection is Master, then the Bluetooth Clock of this device shall be returned. If the current role is Slave, then an estimate of the Bluetooth Clock of the remote master and the accuracy of this value shall be returned.

The accuracy reflects the clock drift that might have occurred since the slave last received a valid transmission from the master.

Note: The Bluetooth Clock has a minimum accuracy of 250ppm, or about 22 seconds drift in one day.

Note: See Part B, Section 1.1, Bluetooth Clock for more information about the Bluetooth Clock.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify which connection to be used for reading the masters Bluetooth Clock. Range: 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Which_Clock: *Size 1 Octet)*

Value	Parameter Description
0xXX	0x00 = Local Clock (Connection_Handle does not have to be valid) 0x01 = Piconet Clock (Connection_Handle shall be valid) 0x02 to 0xFF = Reserved

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_BD_CLOCK command succeeded.
0x01 – 0xFF	Read_BD_CLOCK command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection_Handle for the connection for which the master clock has been read. If the Which_Clock was 0, then the Connection_Handle shall be set to 0 and ignored upon receipt. Range: 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Clock: *Size: 4 Octets (28 bits meaningful)*

Value	Parameter Description
0xXXXXXXXX	Bluetooth Clock of the device requested.

Accuracy: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	+/- maximum Bluetooth Clock error. Value of 0xFFFF means Unknown. Accuracy = +/- N * 0.3125 msec (1 Bluetooth Clock) Range for N: 0x0000 - 0xFFFE Time Range for N: 0 - 20479.375 msec

Event(s) generated (unless masked away):

When the Read_Clock command has completed, a Command Complete event shall be generated.

7.5.7 Read Encryption Key Size Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Read_Encryption_ Key_Size	0x0008	Connection_Handle	Status, Connection_Handle, Key_Size

Description:

This command reads the current encryption key size associated with the Connection_Handle. The Connection_Handle shall be a Connection_Handle for an active ACL connection.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle for the encryption key size to be read from. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use).

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Encryption_Key_Size succeeded
0x01 - 0xFF	Read_Encryption_Key_Size failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle for the encryption key size. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use).

Key_Size: *Size: 1 Octet*

Value	Parameter Description
0xXX	Encryption key size. See Part C, Section 5.2, Parameter Definitions.

Event(s) generated (unless masked away):

When the Read_Encryption_Key_Size command has completed, a Command_Complete event shall be generated.

If the ACL connection associated with the Connection_Handle is not encrypted, the Controller shall return a Command Complete event with the error code Insufficient Security (0x2F).

Ford Ex. 1052

7.5.8 Read Local AMP Info Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Local_AMP_Info	0x0009		Status, AMP_Status, Total_Bandwidth, Max_Guaranteed_Bandwidth, Min_Latency, Max_PDU_Size, Controller_Type, PAL_Capabilities, Max_AMP_ASSOC_Length, Max_Flush_Timeout, Best_Effort_Flush_Timeout

Description:

This command returns information about the AMP Controller.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Local_AMP_Info command succeeded
0x01-0xFF	Read_Local_AMP_Info command failed. See Part D, Error Codes on page 339

AMP_Status: *Size: 1 Octet*

Value	Parameter Description
0x00	The Controller is available but is currently physically powered down. This value shall be used if the AMP Controller is present and can be powered up by the AMP Manager. This value indicates that there may be a cost of time and power to use this AMP Controller (i.e., the time taken and power required to power up the AMP Controller). These costs are AMP type and AMP implementation dependent.

Value	Parameter Description
0x01	This value indicates that the AMP Controller is only used by Bluetooth technology and will not be shared with other non-Bluetooth technologies. This value shall only be used if the AMP Controller is powered up. This value does not indicate how much bandwidth is currently free on the AMP Controller.
0x02	The AMP Controller has no capacity available for Bluetooth operation This value indicates that all of the AMP Controller's bandwidth is currently allocated to servicing a non Bluetooth technology. A device is permitted to create a Physical Link to an AMP Controller that has this status. This value shall only be used if the AMP Controller is powered up.
0x03	The AMP Controller has low capacity available for Bluetooth operation. This value indicates that the majority of the AMP Controller's bandwidth is currently allocated to servicing a non Bluetooth technology. An AMP Controller with capacity in the approximate range of 0% to 30% should indicate this value. This value does not indicate how much of the capacity available for Bluetooth operation is currently available. This value shall only be used if the AMP Controller is powered up.
0x04	The AMP Controller has medium capacity available for Bluetooth operation. An AMP Controller with capacity in the approximate range of 30% to 70% should indicate this value. This value does not indicate how much of the capacity available for Bluetooth operation is currently available. This value shall only be used if the AMP Controller is powered up.
0x05	The AMP Controller has high capacity available for Bluetooth operation. This value indicates that the majority of the AMP Controller's bandwidth is currently allocated to servicing the Bluetooth technology. An AMP Controller with capacity in the approximate range of 70% to 100% should indicate this value. This value does not indicate how much of the capacity available for Bluetooth operation is currently available. This value shall only be used if the AMP Controller is powered up.

Value	Parameter Description
0x06	The AMP Controller has full capacity available for Bluetooth operation. This value indicates that while currently the AMP is only being used by Bluetooth the device allows a different technology to share the radio. This value shall be used by devices that are not capable of determining the current available capacity of an AMP that is shared by a different technology. This value does not indicate how much of the capacity available for Bluetooth operation is currently available. This value shall only be used if the AMP Controller is powered up.
0x07 - 0xFF	Reserved

The AMP Controller to generate AMP_Status_Change event if this parameter changes after initial Read_Local_AMP_Info request. See AMP Manager Protocol Specification, Part E Section 3.4.

Total_Bandwidth: *Size: 4 Octets*

Value	Parameter Description
	An upper bound on the total data rate that can be achieved by the AMP over HCI. It accounts for the total bandwidth provided by the HCI transport. No sustained combination of transmit and receive operations shall exceed this value. This can be used to help in AMP selection and admission control. Expressed in kbps.

Max_Guaranteed_Bandwidth: *Size: 4 Octets*

Value	Parameter Description
	An upper bound on the maximum data the AMP can guarantee for a single logical link over HCI. Any request made by an application above this threshold would be rejected. It accounts for any bandwidth limitations of the HCI transport. No sustained combination of transmit and receive operations shall exceed this value. This can be used to help in AMP selection and admission control. Expressed in kbps.

Min_Latency: *Size: 4 Octets*

Value	Parameter Description
	A lower bound on the latency in microseconds that the AMP can guarantee for a logical channel. It accounts for the minimum latency of the HCI transport. This can be used to help in AMP selection and admission control.

Max_PDU_Size: *Size: 4 Octets*

Value	Parameter Description
	An upper bound on the size of L2CAP PDU which may be provided for transmission or reception on this AMP. The Host shall not require the AMP to transport L2CAP PDUs larger than this value. Expressed in octets. The Maximum PDU Size parameter described in [Vol 3] Part A, Section 5.4 for any connection over this AMP should not exceed this value.

Controller_Type: *Size: 1 Octet*

Value	Parameter Description
See Bluetooth Assigned Numbers.	

PAL_Capabilities: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Bit 0: "Service Type = Guaranteed" is supported by PAL = 1 Bits 15-1: Reserved (shall be set to 0) (See [Vol 3] Part A, Section 5.6, Extended Flow Specification Option.)

Max_AMP_ASSOC_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	AMP_ASSOC maximum length for this local AMP Controller. For use with Read and Write AMP_ASSOC commands [Section 7.5.8 and Section 7.5.9]. This value is sent to the remote AMP Manager in the AMP Get Info Response (see "[Vol 3] Part E, Section 3.8, AMP Get Info Response (Code 0x07)).

Max_Flush_Timeout: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Maximum time period, in microseconds, which the AMP device uses to attempt to transmit a frame on a Guaranteed Logical Link. If the Controller is configured to retry frames for an unbounded time (i.e. there is no flushing at all), then it shall set this value to 0xFFFFFFFF.

Best_Effort_Flush_Timeout *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	The typical time period, in microseconds, which the AMP device may use to attempt to transmit a frame on a Best Effort Logical Link. The value shall not exceed the value given in Max_Flush_Timeout. If the Controller is configured to retry frames for an unbounded time (i.e. there is no flushing at all), then it shall set this value to 0xFFFFFFFF.

Event(s) generated (unless masked away):

When the Read_Local_AMP_Info command has completed, a Command Complete event shall be generated.

7.5.9 Read Local AMP ASSOC Command

Command	OCF	Command Parameters	Return Parameters
HCI_ Read_Local_AMP_ASSOC	0x000A	Physical_Link_Handle, Length_So_Far, Max_Remote_ AMP_ASSOC_Length	Status, Physical_Link_Handle, AMP_ASSOC_ Remaining_Length, AMP_ASSOC_fragment

Description:

This command returns a fragment of the AMP_ASSOC structure. The AMP_ASSOC contains information in an AMP type specific format and is defined in the AMP PALs (see Volume 5,). This may include a combination of address, capabilities and status information. This command can be used when creating an AMP physical link as given by the Physical_Link_Handle parameter, see the Create Physical Link command for more details. If the command is being executed outside of the context of any physical link creation, then the Physical_Link_Handle shall be 0x00.

The Controller shall limit the returned AMP_ASSOC to Max_AMP_ASSOC_Length octets. The Host shall set this parameter to the "AMP_ASSOC_Size" value returned from the remote device in the AMP Get Info Response (see AMP Manager Protocol Specification, [Vol 3] Part E, Section 3.8).

The AMP_ASSOC length may be larger that can fit in a single HCI event. Each time the Read_Local_AMP_ASSOC command is called, the remaining length of the AMP_ASSOC (including the data returned in the Command Complete event) is included in the AMP_ASSOC_Remaining_Length parameter and data from the AMP_ASSOC is contained in the AMP_ASSOC_fragment parameter. In order to obtain the entire AMP_ASSOC, a Host shall continue calling Read_Local_AMP_ASSOC until the AMP_ASSOC_Remaining_Length parameter returned is equal to the size of the AMP_ASSOC_fragment in that event.

The Length_So_Far parameter shall be set to 0 to obtain the first AMP_ASSOC_fragment. Subsequent fragments are obtained by incrementing the Length_So_Far parameter by the length of the previous fragment.

Once the Host begins reading an AMP_ASSOC, all fragments returned in a sequence of Read_Local_AMP_ASSOC commands shall be from one, static AMP_ASSOC.

Command Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Length_So_Far: *Size: 2 Octets*

Value	Parameter Description
0x00	Used to indicate invalid Physical_Link_Handle
0xXX	Physical_Link_Handle which describes the link to which the AMP_ASSOC belongs.

Value
Set Length_So_Far to 0 for the first fragment. Increment parameter by the length of the previous fragment, for each subsequent call

AMP_ASSOC_Length : *Size: 2 Octets*

Value	Parameter Description
0xXXXX	The maximum length, in octets, allowed by the Host for the AMP_ASSOC maximum length for the remote returned by the AMP Controller. For use with Read command to prevent interoperability issues.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Local_AMP_ASSOC command succeeded
0x01-0xFF	Read_Local_AMP_ASSOC command failed. See Part D, Error Codes on page 339 for list of Error Codes.

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle identifying the physical link to be created with the associated AMP_ASSOC.

AMP_ASSOC_Remaining_Length: *Size: 2 Octets*

Value	Parameter Description
1-size of variable	Length, in octets, of the remainder of the AMP_ASSOC including this fragment.
0	Reserved

AMP_ASSOC_fragment: *Size: 1 to 248 Octets*

Value
A fragment of the local AMP_ASSOC structure created by a remote AMP of the same Controller type. A fragment, other than the final one, may not be less than 248 bytes in length.

Event(s) generated (unless masked away):

When the Read_Local_AMP_ASSOC command has completed, a Command Complete event shall be generated.

7.5.10 Write Remote AMP ASSOC Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Remote_ AMP_ASSOC	0x000B	Physical_Link_Handle, Length_So_Far, AMP_ASSOC_ Remaining_Length, AMP_ASSOC_fragment	Status, Physical_Link_Handle

Description:

This command writes an AMP_ASSOC fragment to the AMP Controller. The AMP_ASSOC contains information in an AMP type specific format and is defined in the AMP PALs (see Volume 5, Core System Package [AMP Controller volume]). This may include a combination of address, capabilities and status information. This command may be used when creating an AMP physical link. See the Create_Physical_Link command for more details. See Section 7.1.37.

The AMP_ASSOC length may be larger that can fit in a single HCI command. Each time the Write_Remote_AMP_ASSOC command is called, the remaining length of the AMP_ASSOC (including the data provided in the command) is included in the AMP_ASSOC_Remaining_Length parameter and data from the AMP_ASSOC is contained in the AMP_ASSOC_fragment parameter. In order to write the entire AMP_ASSOC to the Controller, a Host shall continue calling Write_Remote_AMP_ASSOC until the AMP_ASSOC_Remaining_Length parameter returned is equal to the size of the AMP_ASSOC_fragment in that command.

The Write_Local_AMP_ASSOC command shall be called immediately after the Command Status event (with result of success) for either the Create_Physical_Link or Accept_Physical_Link commands.

The Length_So_Far parameter shall be set to 0 to write the first AMP_ASSOC_fragment. Subsequent fragments are written by incrementing the Length_So_Far parameter by the length of the previous fragment.

Command Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle identifying the physical link to be created with the associated AMP_ASSOC.

Length_So_Far: *Size: 2 Octets*

AMP_ASSOC_Remaining_Length: *Size: 2 Octets*

Value
Set Length_So_Far to 0 for the first fragment. Increment it by the length of the previous fragment, for each subsequent call.

Value	Parameter Description
0x0001-Max_AMP_ASSOC_Length	Length, in octets, of the remainder of the AMP_ASSOC, including this fragment. Max_AMP_ASSOC_Length as reported in Read_Local_AMP_Info command.
0x0000	Reserved

AMP_ASSOC_fragment: *Size: 1 to 248 Octets*

Value
A fragment of the AMP_ASSOC structure created by a remote AMP of the same Controller type.
The fragment length shall be 248 bytes for all but the last fragment.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Remote_AMP_ASSOC command succeeded
0x01-0xFF	Write_Remote_AMP_ASSOC command failed. See "Part D, Error Codes on page 339 for list of Error Codes.

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle identifying the physical link to be created with the associated AMP_ASSOC.

Event(s) generated (unless masked away):

When the Write_Remote_AMP_ASSOC command has completed, a Command Complete event shall be generated.

7.6 TESTING COMMANDS

The Testing commands are used to provide the ability to test various functional capabilities of the Bluetooth hardware. These commands provide the ability to arrange various conditions for testing.

For the Testing Commands, the OGF is defined as 0x06.

7.6.1 Read Loopback Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Loopback_Mode	0x0001		Status, Loopback_Mode

Description:

This command reads the value for the setting of the Controller's Loopback_Mode. The setting of the Loopback_Mode parameter shall determine the path of information. In Non-testing Mode operation, the Loopback_Mode parameter is set to Non-testing Mode and the path of the information is as specified by the Bluetooth specifications. In Local Loopback Mode, every Data Packet (ACL, SCO and eSCO) and Command Packet that is sent from the Host to the Controller is sent back with no modifications by the Controller, as shown in Figure 7.1 on page 707. For details of loopback modes see Section 7.6.2 on page 706.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Loopback_Mode command succeeded.
0x01-0xFF	Read_Loopback_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Loopback_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	No Loopback mode enabled (**default**).
0x01	Enable Local Loopback.
0x02	Enable Remote Loopback. (Not supported by AMPs)
0x03-0xFF	Reserved for future use.

Event(s) generated (unless masked away):

When the Read_Loopback_Mode command has completed, a Command Complete event shall be generated.

7.6.2 Write Loopback Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Loopback_Mode	0x0002	Loopback_Mode	Status

Description:

This command writes the value for the setting of the BR/EDR Controller's Loopback Mode. The setting of the Loopback_Mode parameter shall determine the path of information. In Non-testing Mode operation, the Loopback_Mode parameter is set to Non-testing Mode and the path of the information as specified by the Bluetooth specifications. In Local Loopback Mode, every Data Packet (ACL, SCO and eSCO) and Command Packet that is sent from the Host to the BR/EDR Controller is sent back with no modifications by the BR/EDR Controller, as shown in Figure 7.1 on page 707.

When the BR/EDR Controller enters Local Loopback Mode, it shall respond with one to four Connection_Handles, one for an ACL connection and zero to three for synchronous connections. The Host should use these Connection_Handles when sending data in Local Loopback Mode. The number of Connection_Handles returned for synchronous connections (between zero and three) is implementation specific. When in Local Loopback Mode, the BR/EDR Controller loops back commands and data to the Host. The Loopback Command event is used to loop back commands that the Host sends to the Controller.

When an AMP enters Local Loopback Mode, all data transmission will be looped back with the received Logical_Link_Handle value, without checking the validity of the Logical Link Handle. This allows local loopback of data without any link creation, and without any data transmission over the air.

'There are some commands that are not looped back in Local Loopback Mode: Reset, Set_Controller_To_Host_Flow_Control, Host_Buffer_Size, Host_Number_Of_Completed_Packets, Read_Buffer_Size, Read_Loopback_Mode and Write_Loopback_Mode. These commands should be executed in the way they are normally executed. The commands Reset and Write_Loopback_Mode can be used to exit local loopback mode.

If Write_Loopback_Mode is used to exit Local Loopback Mode on a BR/EDR Controller, Disconnection Complete events corresponding to the Connection Complete events that were sent when entering Local Loopback Mode should be sent to the Host. Furthermore, no connections are allowed in Local Loopback mode. If there is a connection, and there is an attempt to set the device to Local Loopback Mode, the attempt will be refused. When the device is in Local Loopback Mode, the Controller will refuse incoming connection attempts. This allows the Host BR/EDR Controller Transport Layer to be tested without any other variables.

If a BR/EDR Controller is set to Remote Loopback Mode, it will send back all data (ACL, SCO and eSCO) that comes over the air. It will only allow a maximum of one ACL connection and three synchronous connections, and these shall all be to the same remote device. If there are existing connections to a remote device and there is an attempt to set the local device to Remote Loopback Mode, the attempt shall be refused.

If a Host sets Loopback Mode to “Remote Loopback” on an AMP controller, the Controller shall reject the command with the error code Invalid HCI command parameter (0x12).

See Figure 7.2 on page 708, where the rightmost device is set to Remote Loopback Mode and the leftmost device is set to Non-testing Mode. This allows the BR/EDR Air link to be tested without any other variables.

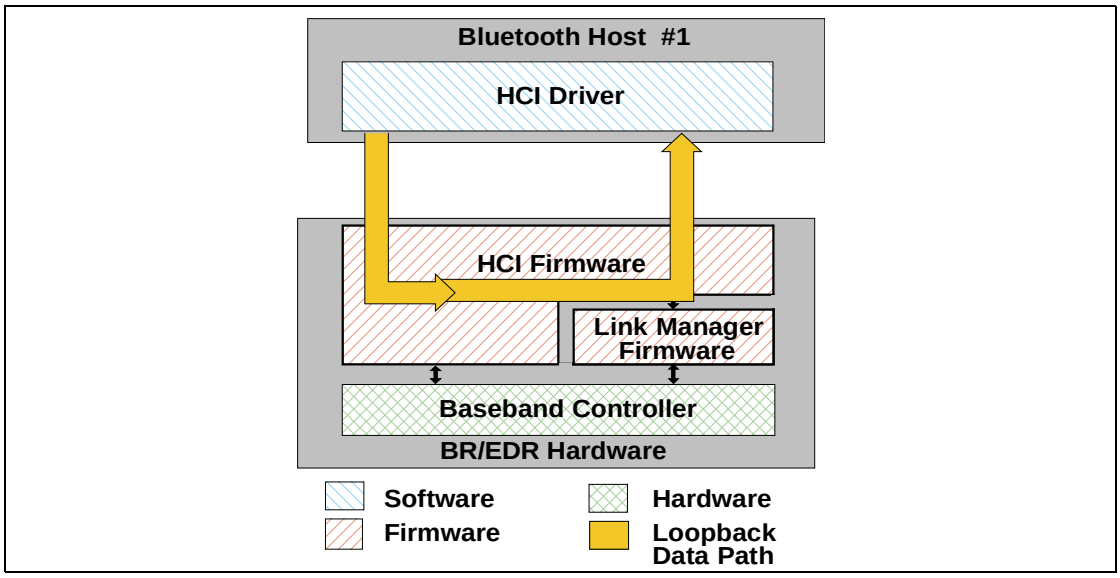


Figure 7.1: Local Loopback Mode

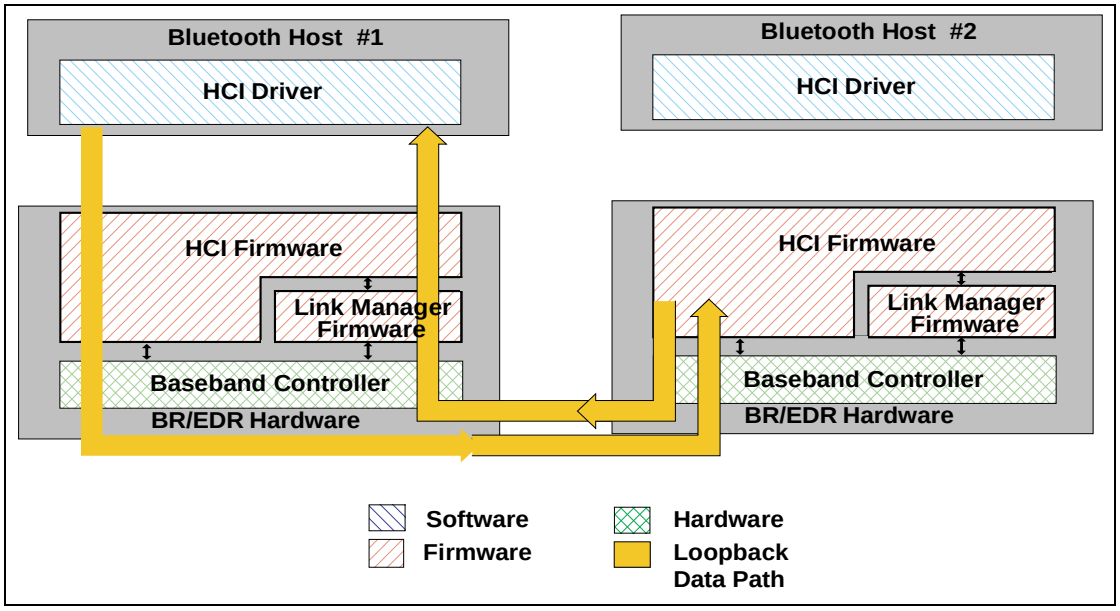


Figure 7.2: Remote Loopback Mode

Command Parameters:

Loopback_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	No Loopback mode enabled (**default**).
0x01	Enable Local Loopback.
0x02	Enable Remote Loopback. (Not Supported by AMPs)
0x03-0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Loopback_Mode command succeeded.
0x01-0xFF	Write_Loopback_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Loopback_Mode command has completed, a Command Complete event shall be generated.

7.6.3 Enable Device Under Test Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Enable_Device_Under_Test_Mode	0x0003		Status

Description:

The Enable_Device_Under_Test_Mode command allows the local BR/EDR Controller to enter test mode via LMP test commands for BR/EDR Controllers or via the AMP Test Command for AMP Controllers. For details see Part C, Link Manager Protocol Specification on page 207 for BR/EDR Controllers or Section 7.6.7 for AMP controllers. The Host issues this command when it wants the local device to be the DUT for the Testing scenarios as described in the [Vol 3] Part D, Test Methodology. When the BR/EDR Controller receives this command, it shall complete the command with a Command Complete event. The BR/EDR Controller functions as normal until the remote tester issues the LMP test command or for AMPs the AMP Test Command to place the local device into Device Under Test mode. To disable and exit the Device Under Test Mode, the Host may issue the Reset command. This command prevents remote BR/EDR Controllers from causing the local BR/EDR Controller to enter test mode without first issuing this command.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Enter_Device_Under_Test_Mode command succeeded.
0x01-0xFF	Enter_Device_Under_Test_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Enter_Device_Under_Test_Mode command has completed, a Command Complete event shall be generated.

7.6.4 Write Simple Pairing Debug Mode Command

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Simple_Pairing_Debug_Mode	0x0004	Simple_Pairing_Debug_Mode	Status

Description:

This command configures the BR/EDR Controller to use a predefined Diffie Hellman private key for Simple Pairing to enable debug equipment to monitor the encrypted connection.

Note: Only one side (initiator or responder) needs to set simple pairing debug mode in order for debug equipment to be able to determine the link key and, therefore, be able to monitor the encrypted connection.

When the Simple_Pairing_Debug_Mode parameter is set to enabled the BR/EDR Controller shall use the predefined Diffie Hellman private key. The BR/EDR Controller shall also set the resulting Link_Key type to "Debug Combination Key."

When in Simple Pairing debug mode, the Link Manager shall use the following Diffie Hellman private / public key pair:

- Private key: 07915f86918ddc27005df1d6cf0c142b625ed2eff4a518ff
- Public key (X): 15207009984421a6586f9fc3fe7e4329d2809ea51125f8ed
- Public key (Y): b09d42b81bc5bd009f79e4b59dbbaa857fca856fb9f7ea25

Command Parameters:

Simple_Pairing_Debug_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Simple Pairing debug mode disabled (**default**)
0x01	Simple pairing debug mode enabled
0x02 - 0xFF	Reserved for future use

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Simple_Pairing_Mode command succeeded.
0x01 - 0xFF	Write_Simple_Pairing_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Simple_Pairing_Debug_Mode command has completed, a Command Complete event shall be generated.

7.6.5 Enable AMP Receiver Reports Command

Command	OCF	Command Parameters	Return Parameters
HCI_Enable_AMP_Receiver_Reports	0x0007	Enable, Interval	Status

Description:

This command enables and disables the reporting of frames received. This command shall only be valid in AMP Test Mode. After multiples of the specified Interval, a report of frames that have been received/not received by the DUT shall be sent to the tester. Optionally, the sums of bits in error are also reported. The values returned are the cumulative totals since the mode was last initiated or reset. The totals are reset by sending the Enable_AMP_Receiver_Reports, the HCI_AMP_Test command or the AMP_Test_End command. This event is also generated after an AMP Test End event.

Command Parameters:

Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	Disable
0x01	Enable
0x02-0xFF	Reserved

Interval: *Size: 1 Octet*

Value	Parameter Description
0x00	Reserved
0x01-0xFF	Number of seconds between event reporting

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Enable_AMP_Receiver_Reports command succeeded
0x01-0xFF	Enable_AMP_Receiver_Reports command failed. Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the Enable_AMP_Receiver_Reports command has completed, a Command Complete event shall be generated.

7.6.6 AMP Test End Command

Command	OCF	Command Parameters	Return Parameters
HCI_AMP_Test_End	0x0008		Status

Description:

This command is used to stop any test scenario in progress. This command shall only be used in AMP Test Mode when a test is in progress. This command shall not exit an AMP Controller from AMP Test mode.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	AMP_Test_End command succeeded
0x01-0xFF	AMP_Test_End command failed. Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the AMP_Test_End command has completed an AMP Test End event shall be generated and the AMP Receiver Report event shall be generated if the AMP receiver reports are enabled.

7.6.7 AMP Test Command

Command	OCF	Command Parameters	Return Parameters
HCI_AMP_Test	0x0009	Test_Parameters	Status

Description:

This command is used to configure and start a test. This command shall be only valid in AMP Test Mode.

When a test scenario has completed or on receiving a AMP_Test_End command the AMP shall send a AMP Test End event and the AMP returns to an idle state with the RX and TX off.

The details of the Test_Parameters for each Controller Type are detailed in the AMP PAL specification.

Command Parameters:

Test_Parameters: *Size: 1 Octet*

Value	Parameter Description
0xXX	Controller Type. See Assigned Numbers

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	AMP_Test command succeeded
0x01-0xFF	AMP_Test command failed. See Part D, Error Codes on page 339

Event(s) generated (unless masked away):

When the AMP Controller receives the AMP_Test command, the AMP Controller shall send the Command Status event to the AMP Test Manager which shall be routed to the tester.

The AMP Start Test event shall be generated when the AMP_Test command has completed and the first data is ready to be sent or received.

Command Complete event shall not be sent by the AMP to indicate that this command has been completed. Instead, the AMP Start Test event shall indicate that this command has been completed.

When in a transmitter test scenario and the frames/bursts count have been transmitted the AMP Test End event shall be sent.

7.7 EVENTS

7.7.1 Inquiry Complete Event

Event	Event Code	Event Parameters
Inquiry Complete	0x01	Status

Description:

The Inquiry Complete event indicates that the Inquiry is finished. This event contains a Status parameter, which is used to indicate if the Inquiry completed successfully or if the Inquiry was not completed.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Inquiry command completed successfully.
0x01-0xFF	Inquiry command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

7.7.2 Inquiry Result Event

Event	Event Code	Event Parameters
Inquiry Result	0x02	Num_Responses, BD_ADDR[i], Page_Scan_Repetition_Mode[i], Reserved[i], Reserved[i], Class_of_Device[i] Clock_Offset[i]

Description:

The Inquiry Result event indicates that a BR/EDR Controller or multiple BR/EDR Controllers have responded so far during the current Inquiry process. This event will be sent from the BR/EDR Controller to the Host as soon as an Inquiry Response from a remote device is received if the remote device supports only mandatory paging scheme. The BR/EDR Controller may queue these Inquiry Responses and send multiple BR/EDR Controllers information in one Inquiry Result event. The event can be used to return one or more Inquiry responses in one event.

Event Parameters:

Num_Responses: *Size: 1 Octet*

Value	Parameter Description
0xXX	Number of responses from the Inquiry.

BD_ADDR[i]: *Size: 6 Octets * Num_Responses*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for each device which responded.

Page_Scan_Repetition_Mode[i]: *Size: 1 Octet * Num_Responses*

Value	Parameter Description
0x00	R0
0x01	R1
0x02	R2
0x03 – 0xFF	Reserved

Reserved[i]: [1] *Size: 1 Octet * Num_Responses*

Value	Parameter Description
0xXX	Reserved.

Reserved[i]: [2] *Size: 1 Octet * Num_Responses*

Value	Parameter Description
0xXX	Reserved, shall be set to 0x00.

Class_of_Device[i]: *Size: 3 Octets * Num_Responses*

Value	Parameter Description
0xXXXXXX	Class of Device for the device

Clock_Offset[i]: *Size: 2 Octets * Num_Responses*

Bit format	Parameter Description
Bit 14-0	Bit 16-2 of CLKslave-CLKmaster.
Bit 15	Reserved

1. This was the Page_Scan_Period_Mode parameter in the v1.1 specification. This parameter has no meaning in v1.2 or later and no default value.
2. This was the Page_Scan_Mode parameter in the v1.1 specification.

7.7.3 Connection Complete Event

Event	Event Code	Event Parameters
Connection Complete	0x03	Status, Connection_Handle, BD_ADDR, Link_Type, Encryption_Enabled

Description:

The Connection Complete event indicates to both of the Hosts forming the connection that a new connection has been established. This event also indicates to the Host, which issued the Create_Connection, or Accept_Connection_Request or Reject_Connection_Request command and then received a Command Status event, if the issued command failed or was successful.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Connection successfully completed.
0x01-0xFF	Connection failed to Complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection between two BR/EDR Controllers. The Connection_Handle is used as an identifier for transmitting and receiving voice or data. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the other connected Device forming the connection.

Link_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	SCO connection.
0x01	ACL connection (Data Channels).
0x02-0xFF	Reserved for future use.

Encryption_Enabled: *Size: 1 Octet*

Value	Parameter Description
0x00	Link level encryption disabled.
0x01	Link level encryption enabled.
0x02-0xFF	Reserved for future use.

7.7.4 Connection Request Event

Event	Event Code	Event Parameters
Connection Request	0x04	BD_ADDR, Class_of_Device, Link_Type

Description:

The Connection Request event is used to indicate that a new incoming connection is trying to be established. The connection may either be accepted or rejected. If this event is masked away and there is an incoming connection attempt and the BR/EDR Controller is not set to auto-accept this connection attempt, the BR/EDR Controller shall automatically refuse the connection attempt. When the Host receives this event and the link type parameter is ACL, it should respond with either an Accept_Connection_Request or Reject_Connection_Request command before the timer Conn_Accept_Timeout expires. If the link type is SCO or eSCO, the Host should reply with the Accept_Synchronous_Connection_Request or the Reject_Synchronous_Connection_Request command. If the link type is SCO, the Host may respond with Accept_Connection_Request command. If the event is responded to with Accept_Connection_Request command, then the default parameter settings of the Accept_Synchronous_Connection_Request command (see Section 7.1.27 on page 504) should be used by the local Link Manager when negotiating the SCO link parameters. In that case, the Connection Complete event and not the Synchronous Connection Complete event, shall be returned on completion of the connection.

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the device that requests the connection.

Class_of_Device: *Size: 3 Octets*

Value	Parameter Description

0xXXXXXX	Class of Device for the device, which requests the connection.
0x000000	Unknown Class of Device

Link_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	SCO Connection requested
0x01	ACL Connection requested
0x02	eSCO Connection requested
0x03-0xFF	Reserved for future use.

7.7.5 Disconnection Complete Event

Event	Event Code	Event Parameters
Disconnection Complete	0x05	Status, Connection_Handle, Reason

Description:

The Disconnection Complete event occurs when a connection is terminated. The status parameter indicates if the disconnection was successful or not. The reason parameter indicates the reason for the disconnection if the disconnection was successful. If the disconnection was not successful, the value of the reason parameter can be ignored by the Host. For example, this can be the case if the Host has issued the Disconnect command and there was a parameter error, or the command was not presently allowed, or a Connection_Handle that didn't correspond to a connection was given.

Note: When a physical link fails, one Disconnection Complete event will be returned for each logical channel on the physical link with the corresponding Connection_Handle as a parameter.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Disconnection has occurred.
0x01-0xFF	Disconnection failed to complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle which was disconnected. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Reason: *Size: 1 Octet*

Value	Parameter Description
0xXX	Reason for disconnection. See Part D, Error Codes on page 339 for error codes and descriptions.

7.7.6 Authentication Complete Event

Event	Event Code	Event Parameters
Authentication Complete	0x06	Status, Connection_Handle

Description:

The Authentication Complete event occurs when authentication has been completed for the specified connection. The Connection_Handle must be a Connection_Handle for an ACL connection.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Authentication Request successfully completed.
0x01-0xFF	Authentication Request failed to complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle for which Authentication has been performed. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

7.7.7 Remote Name Request Complete Event

Event	Event Code	Event Parameters
Remote Name Request Complete	0x07	Status, BD_ADDR, Remote_Name

Description:

The Remote Name Request Complete event is used to indicate that a remote name request has been completed. The Remote_Name event parameter is a UTF-8 encoded string with up to 248 octets in length. The Remote_Name event parameter will be null-terminated (0x00) if the UTF-8 encoded string is less than 248 octets. The BD_ADDR event parameter is used to identify which device the user-friendly name was obtained from.

Note: The Remote_Name parameter is a string parameter. Endian-ness does therefore not apply to the Remote_Name parameter. The first octet of the name is received first.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Remote_Name_Request command succeeded.
0x01-0xFF	Remote_Name_Request command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for the device whose name was requested.

Remote_Name: *Size: 248 Octets*

Value	Parameter Description
Name[248]	A UTF-8 encoded user-friendly descriptive name for the remote device. If the name contained in the parameter is shorter than 248 octets, the end of the name is indicated by a NULL octet (0x00), and the following octets (to fill up 248 octets, which is the length of the parameter) do not have valid values.

7.7.8 Encryption Change Event

Event	Event Code	Event Parameters
Encryption Change	0x08	Status, Connection_Handle, Encryption_Enabled

Description:

The Encryption Change event is used to indicate that the change of the encryption mode has been completed. The Connection_Handle will be a Connection_Handle for an ACL connection. The Encryption_Enabled event parameter specifies the new Encryption_Enabled parameter for the Connection_Handle specified by the Connection_Handle event parameter. This event will occur on both devices to notify the Hosts when Encryption has changed for the specified Connection_Handle between two devices. Note: This event shall not be generated if encryption is paused or resumed; during a role switch, for example.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Encryption Change has occurred.
0x01-0xFF	Encryption Change failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle for which the link layer encryption has been enabled/disabled for all Connection_Handles with the same BR/EDR Controller endpoint as the specified Connection_Handle. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Encryption_Enabled: *Size: 1 Octet*

Value	Parameter Description
0x00	Link Level Encryption is OFF.
0x01	Link Level Encryption is ON.
0x02-0xFF	Reserved.

7.7.9 Change Connection Link Key Complete Event

Event	Event Code	Event Parameters
Change Connection Link Key Complete	0x09	Status, Connection_Handle

Description:

The Change Connection Link Key Complete event is used to indicate that the change in the Link Key for the Connection_Handle specified by the Connection_Handle event parameter has been completed.

The Connection_Handle will be a Connection_Handle for an ACL connection. The Change Connection Link Key Complete event is sent only to the Host which issued the Change_Connection_Link_Key command.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Change_Connection_Link_Key command succeeded.
0x01-0xFF	Change_Connection_Link_Key command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle which the Link Key has been change for all Connection_Handles with the same BR/EDR Controller end point as the specified Connection_Handle. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

7.7.10 Master Link Key Complete Event

Event	Event Code	Event Parameters
Master Link Key Complete	0x0A	Status, Connection_Handle, Key_Flag

Description:

The Master Link Key Complete event is used to indicate that the Link Key managed by the master of the piconet has been changed. The Connection_Handle will be a Connection_Handle for an ACL connection. The link key used for the connection will be the temporary link key of the master device or the semi-permanent link key indicated by the Key_Flag. The Key_Flag event parameter is used to indicate which Link Key (temporary link key of the Master, or the semi-permanent link keys) is now being used in the piconet.

Note: For a master, the change from a semi-permanent Link Key to temporary Link Key will affect all Connection_Handles related to the piconet. For a slave, this change affects only this particular Connection_Handle. A temporary link key must be used when both broadcast and point-to-point traffic shall be encrypted.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Master_Link_Key command succeeded.
0x01-0xFF	Master_Link_Key command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle for which the Link Key has been changed for all devices in the same piconet. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Key_Flag: *Size: 1 Octet*

Value	Parameter Description
0x00	Using Semi-permanent Link Key.
0x01	Using Temporary Link Key.

7.7.11 Read Remote Supported Features Complete Event

Event	Event Code	Event Parameters
Read Remote Supported Features Complete	0x0B	Status, Connection_Handle, LMP_Features

Description:

The Read Remote Supported Features Complete event is used to indicate the completion of the process of the Link Manager obtaining the supported features of the remote BR/EDR Controller specified by the Connection_Handle event parameter. The Connection_Handle will be a Connection_Handle for an ACL connection. The event parameters include a list of LMP features. For details see Part C, Link Manager Protocol Specification on page 207.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Remote_Supported_Features command succeeded.
0x01-0xFF	Read_Remote_Supported_Features command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle which is used for the Read_Remote_Supported_Features command. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

LMP_Features: *Size: 8 Octets*

Value	Parameter Description
0xXXXXXXXX XXXXXXXX	Bit Mask List of LMP features. See Part C, Link Manager Protocol Specification on page 207.

7.7.12 Read Remote Version Information Complete Event

Event	Event Code	Event Parameters
Read Remote Version Information Complete	0x0C	Status, Connection_Handle, Version, Manufacturer_Name, Subversion

Description:

The Read Remote Version Information Complete event is used to indicate the completion of the process obtaining the version information of the remote Controller specified by the Connection_Handle event parameter. The Connection_Handle shall be for an ACL connection.

The Version event parameter defines the specification version of the BR/EDR or LE Controller. The Manufacturer_Name event parameter indicates the manufacturer of the remote Controller. The Subversion event parameter is controlled by the manufacturer and is implementation dependent. The Subversion event parameter defines the various revisions that each version of the Bluetooth hardware will go through as design processes change and errors are fixed. This allows the software to determine what Bluetooth hardware is being used and, if necessary, to work around various bugs in the hardware.

When the Connection_Handle is associated with a BR/EDR ACL-U logical link, the Version event parameter shall be LMP VersNr parameter, the Manufacturer_Name event parameter shall be the CompId parameter, and the Subversion event parameter shall be the LMP SubVersNr parameter.

When the Connection_Handle is associated with an LE-U logical link, the Version event parameter shall be Link Layer VersNr parameter, the Manufacturer_Name event parameter shall be the CompId parameter, and the Subversion event parameter shall be the SubVersNr parameter.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Remote_Version_Information command succeeded.
0x01-0xFF	Read_Remote_Version_Information command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description

0xXXXX	Connection Handle which is used for the Read_Remote_Version_Information command. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Version: *Size: 1 Octet*

Value	Parameter Description
0xXX	Version of the Current LMP in the remote Controller. See LMP VersNr and Link LayerVersNr in the Bluetooth Assigned Numbers.

Manufacturer_Name: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Manufacturer Name of the remote Controller. See CompId in the Bluetooth Assigned Numbers.

Subversion: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Subversion of the LMP in the remote Controller. See Part C, Table 5.2, page 320 and [Vol 6] Part B, Section 2.4.2.13(SubVersNr).

7.7.13 QoS Setup Complete Event

Event	Event Code	Event Parameters
QoS Setup Complete	0x0D	Status, Connection_Handle, Flags, Service_Type, Token_Rate, Peak_Bandwidth, Latency, Delay_Variation

Description:

The QoS Setup Complete event is used to indicate the completion of the process of the Link Manager setting up QoS with the remote BR/EDR Controller specified by the Connection_Handle event parameter. The Connection_Handle will be a Connection_Handle for an ACL connection. For more detail see
Part A, Logical Link Control and Adaptation Protocol Specification on page 23.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	QoS_Setup command succeeded.
0x01-0xFF	QoS_Setup command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle which is used for the QoS_Setup command. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Flags: *Size: 1 Octet*

Value	Parameter Description
0x00 – 0xFF	Reserved for future use.

Service_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	No Traffic Available.

0x01	Best Effort Available.
0x02	Guaranteed Available.
0x03-0xFF	Reserved for future use.

Token_Rate: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Available Token Rate, in octets per second.

Peak_Bandwidth: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Available Peak Bandwidth, in octets per second.

Latency: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Available Latency, in microseconds.

Delay_Variation: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Available Delay Variation, in microseconds.

7.7.14 Command Complete Event

Event	Event Code	Event Parameters
Command Complete	0x0E	Num_HCI_Command_Packets, Command_Opcode, Return_Parameters

Description:

The Command Complete event is used by the Controller for most commands to transmit return status of a command and the other event parameters that are specified for the issued HCI command.

The Num_HCI_Command_Packets event parameter allows the Controller to indicate the number of HCI command packets the Host can send to the Controller. If the Controller requires the Host to stop sending commands, the Num_HCI_Command_Packets event parameter will be set to zero. To indicate to the Host that the Controller is ready to receive HCI command packets, the Controller generates a Command Complete event with the Command_Opcode 0x0000, and the Num_HCI_Command_Packets event parameter is set to 1 or more. Command_Opcode, 0x0000 is a NOP (No OPeration), and can be used to change the number of outstanding HCI command packets that the Host can send before waiting. See each command for the parameters that are returned by this event.

Event Parameters:

Num_HCI_Command_Packets: *Size: 1 Octet*

Value	Parameter Description
N = 0xXX	The Number of HCI command packets which are allowed to be sent to the Controller from the Host. Range for N: 0 – 255

Command_Opcode: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Opcode of the command which caused this event.

Return_Parameter(s): *Size: Depends on Command*

Value	Parameter Description
0xXX	This is the return parameter(s) for the command specified in the Command_Opcode event parameter. See each command's definition for the list of return parameters associated with that command.

7.7.15 Command Status Event

Event	Event Code	Event Parameters
Command Status	0x0F	Status, Num_HCI_Command_Packets, Command_Opcode

Description:

The Command Status event is used to indicate that the command described by the Command_Opcode parameter has been received, and that the Controller is currently performing the task for this command. This event is needed to provide mechanisms for asynchronous operation, which makes it possible to prevent the Host from waiting for a command to finish. If the command cannot begin to execute (a parameter error may have occurred, or the command may currently not be allowed), the Status event parameter will contain the corresponding error code, and no complete event will follow since the command was not started. The Num_HCI_Command_Packets event parameter allows the Controller to indicate the number of HCI command packets the Host can send to the Controller. If the Controller requires the Host to stop sending commands, the Num_HCI_Command_Packets event parameter will be set to zero. To indicate to the Host that the Controller is ready to receive HCI command packets, the Controller generates a Command Status event with Status 0x00 and Command_Opcode 0x0000, and the Num_HCI_Command_Packets event parameter is set to 1 or more. Command_Opcode, 0x0000 is a NOP (No OPeration) and can be used to change the number of outstanding HCI command packets that the Host can send before waiting.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Command currently in pending.
0x01-0xFF	Command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Num_HCI_Command_Packets: *Size: 1 Octet*

Value	Parameter Description
N = 0xXX	The Number of HCI command packets which are allowed to be sent to the Controller from the Host. Range for N: 0 – 255

Command_Opcode: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Opcode of the command which caused this event and is pending completion.

7.7.16 Hardware Error Event

Event	Event Code	Event Parameters
Hardware Error	0x10	Hardware_Code

Description:

The Hardware Error event is used to indicate some type of hardware failure for the BR/EDR Controller. This event is used to notify the Host that a hardware failure has occurred in the Controller.

Event Parameters:

Hardware_Code: *Size: 1 Octet*

Value	Parameter Description
0x00-0xFF	These Hardware_Codes will be implementation-specific, and can be assigned to indicate various hardware problems.

7.7.17 Flush Occurred Event

Event	Event Code	Event Parameters
Flush Occurred	0x11	Handle

Description:

The Flush Occurred event is used to indicate that, for the specified Handle, the current user data to be transmitted has been removed. The Handle shall be a Connection_Handle for a BR/EDR LE ACL connection or a Logical_Link_Handle for an AMP ACL connection. This could result from the Flush command, or be due to the automatic flush. Multiple blocks of an L2CAP packet could have been pending in the Controller. If one baseband packet part of an L2CAP PDU is flushed, then the rest of the HCI data packets for the L2CAP PDU must also be flushed.

Event Parameters:

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Handle that was flushed. The Handle is a Connection_Handle for a Primary Controller or a Logical_Link_Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

7.7.18 Role Change Event

Event	Event Code	Event Parameters
Role Change	0x12	Status, BD_ADDR, New_Role

Description:

The Role Change event is used to indicate that the current role of the BR/EDR Controller related to the particular connection has changed. This event only occurs when both the remote and local BR/EDR Controllers have completed their role change for the BR/EDR Controller associated with the BD_ADDR event parameter. This event allows both affected Hosts to be notified when the Role has been changed.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Role change has occurred.
0x01-0xFF	Role change failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device for which a role change has completed.

New_Role: *Size: 1 Octet*

Value	Parameter Description
0x00	Currently the Master for specified BD_ADDR.
0x01	Currently the Slave for specified BD_ADDR.

7.7.19 Number Of Completed Packets Event

Event	Event Code	Event Parameters
Number Of Completed Packets	0x13	Number_of_Handles, Connection_Handle[i], HC_Num_Of_Completed_Packets[i]

Description:

The Number Of Completed Packets event is used by the Controller to indicate to the Host how many HCI Data Packets have been completed (transmitted or flushed) for each Connection_Handle since the previous Number Of Completed Packets event was sent to the Host. This means that the corresponding buffer space has been freed in the Controller. Based on this information, and the HC_Total_Num_ACL_Data_Packets and HC_Total_Num_Synchronous_Data_Packets return parameter of the Read_Buffer_Size command, the Host can determine for which Connection_Handles the following HCI Data Packets should be sent to the Controller. The Number Of Completed Packets event must not be sent before the corresponding Connection Complete event. While the Controller has HCI data packets in its buffer, it must keep sending the Number Of Completed Packets event to the Host at least periodically, until it finally reports that all the pending ACL Data Packets have been transmitted or flushed. The rate with which this event is sent is manufacturer specific.

Note: Number Of Completed Packets events will not report on synchronous Connection Handles if synchronous Flow Control is disabled. (See Read/Write_Synchronous_Flow_Control_Enable on page 612 and page 613.)

Event Parameters:

Number_of_Handles: *Size: 1 Octet*

Value	Parameter Description
0xXX	The number of Connection_Handles and Num_HCI_Data_Packets parameters pairs contained in this event. Range: 0-255

*Connection_Handle[i]: Size: Number_of_Handles * 2 Octets(12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

*HC_Num_Of_Completed_Packets [i]: Size: Number_of_Handles * 2 Octets*

Value	Parameter Description
N = 0xXXXX	The number of HCI Data Packets that have been completed (transmitted or flushed) for the associated Connection_Handle since the previous time the event was returned. Range for N: 0x0000-0xFFFF

7.7.20 Mode Change Event

Event	Event Code	Event Parameters
Mode Change	0x14	Status, Connection_Handle, Current_Mode, Interval

Description:

The Mode Change event is used to indicate when the device associated with the Connection_Handle changes between Active mode, Hold mode, and Sniff mode, and Park state. The Connection_Handle will be a Connection_Handle for an ACL connection. The Connection_Handle event parameter is used to indicate which connection the Mode Change event is for. The Current_Mode event parameter is used to indicate which state the connection is currently in. The Interval parameter is used to specify a time amount specific to each state. Each Controller that is associated with the Connection_Handle which has changed Modes shall send the Mode Change event to its Host.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	A Mode Change has occurred.
0x01-0xFF	Hold_Mode, Sniff_Mode, Exit_Sniff_Mode, Park_State, or Exit_Park_State command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets(12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Current_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Active Mode.
0x01	Hold Mode.
0x02	Sniff Mode.
0x03	Park State.
0x04-0xFF	Reserved for future use.

Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Hold: Number of Baseband slots to wait in Hold Mode. Hold Interval = N * 0.625 msec (1 Baseband slot) Range for N: 0x0002-0xFFFE Time Range: 1.25 msec-40.9 sec Sniff: Number of Baseband slots between sniff anchor points. Time between sniff anchor points = N * 0.625 msec (1 Baseband slot)Range for N: 0x0002-0xFFFE Time Range: 1.25 msec-40.9 sec Park: Number of Baseband slots between consecutive beacons. Interval Length = N * 0.625 msec (1 Baseband slot) Range for N: 0x0002-0xFFFE Time Range: 1.25 msec-40.9 Seconds

7.7.21 Return Link Keys Event

Event	Event Code	Event Parameters
Return Link Keys	0x15	Num_Keys, BD_ADDR [i], Link_Key[i]

Description:

The Return Link Keys event is used by the BR/EDR Controller to send the Host one or more stored Link Keys. Zero or more instances of this event will occur after the Read_Stored_Link_Key command. When there are no link keys stored, no Return Link Keys events shall be returned. When there are link keys stored, the number of link keys returned in each Return Link Keys event is implementation specific. This event shall never return the value of the link keys. The link keys value parameter shall always contain the value of zero.

Event Parameters:

Num_Keys: *Size: 1 Octet*

Value	Parameter Description
0xXX	Number of Link Keys contained in this event. Range: 0x01 – 0x0B

BD_ADDR [i]: *Size: 6 Octets * Num_Keys*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for the associated Link Key.

Link_Key[i]: *Size:16 Octets * Num_Keys*

Value	Parameter Description
0xXXXXXXXXXX XXXXXXXXXX XXXXXXXXXX	Link Key for the associated BD_ADDR.

7.7.22 PIN Code Request Event

Event	Event Code	Event Parameters
PIN Code Request	0x16	BD_ADDR

Description:

The PIN Code Request event is used to indicate that a PIN code is required to create a new link key. The Host shall respond using either the PIN_Code_Request_Reply or the PIN_Code_Request_Negative_Reply command, depending on whether the Host can provide the Controller with a PIN code or not.

Note: If the PIN Code Request event is masked away, then the BR/EDR Controller will assume that the Host has no PIN Code.

When the BR/EDR Controller generates a PIN Code Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host must respond with either a PIN_Code_Request_Reply or PIN_Code_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout. (See Part C, Link Manager Protocol Specification on page 207.)

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device which a new link key is being created for.

7.7.23 Link Key Request Event

Event	Event Code	Event Parameters
Link Key Request	0x17	BD_ADDR

Description:

The Link Key Request event is used to indicate that a Link Key is required for the connection with the device specified in BD_ADDR. If the Host has the requested stored Link Key, then the Host shall pass the requested Key to the Controller using the Link_Key_Request_Reply command. If the Host does not have the requested stored Link Key, or the stored Link Key does not meet the security requirements for the requested service, then the Host shall use the Link_Key_Request_Negative_Reply command to indicate to the Controller that the Host does not have the requested key.

Note: If the Link Key Request event is masked away, then the BR/EDR Controller will assume that the Host has no additional link keys.

If the Host uses the Link_Key_Request_Negative_Reply command when the requested service requires an authenticated Link Key and the current Link Key is unauthenticated, the Host should set the Authentication_Requirements parameter one of the MITM Protection Required options.

When the Controller generates a Link Key Request event in order for the local Link Manager to respond to the request from the remote Link Manager (as a result of a Create_Connection or Authentication_Requested command from the remote Host), the local Host must respond with either a Link_Key_Request_Reply or Link_Key_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout. (See Part C, Link Manager Protocol Specification on page 207.)

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device which a stored link key is being requested.

7.7.24 Link Key Notification Event

Event	Event Code	Event Parameters
Link Key Notification	0x18	BD_ADDR, Link_Key, Key_Type

Description:

The Link Key Notification event is used to indicate to the Host that a new Link Key has been created for the connection with the device specified in BD_ADDR. The Host can save this new Link Key in its own storage for future use. Also, the Host can decided to store the Link Key in the BR/EDR Controller's Link Key Storage by using the Write_Stored_Link_Key command. The Key_Type event parameter informs the Host about which key type (combination key, local unit key, or remote unit key, debug combination key, unauthenticated combination key, authenticated combination key or changed combination key) that was used during pairing. If pairing with unit key is not supported, the Host can for instance discard the key or disconnect the link.

The combination key Key_Type is used when standard pairing was used. The debug combination key Key_Type is used when Simple Pairing was used and the debug public key is sent or received. The unauthenticated combination key Key_Type is used when the Just Works Simple Pairing association model was used. The authenticated combination key Key_Type is used when Simple Pairing was used and the Just Works association mode was not used. The changed combination key Key_Type is used when the link key has been changed using the Change Connection Link Key procedure and Simple Pairing Mode is set to enabled. Note: It is the responsibility of the Host to remember the Key_Type (combination, debug combination, unauthenticated combination, or authenticated combination) prior to changing the link key.

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the Device for which the new link key has been generated.

Link_Key: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXXXXXXXXXXXXXX	Link Key for the associated BD_ADDR.

Key_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Combination Key
0x01	Local Unit Key
0x02	Remote Unit Key
0x03	Debug Combination Key
0x04	Unauthenticated Combination Key
0x05	Authenticated Combination Key
0x06	Changed Combination Key
0x07-0xFF	Reserved

7.7.25 Loopback Command Event

Event	Event Code	Event Parameters
Loopback Command	0x19	HCI_Command_Packet

Description:

When in Local Loopback mode, the BR/EDR Controller loops back commands and data to the Host. The Loopback Command event is used to loop back all commands that the Host sends to the Controller with some exceptions. See section 7.6.1, “Read Loopback Mode Command,” on page 705 for a description of which commands that are not looped back. The HCI_Command_Packet event parameter contains the entire HCI Command Packet including the header.

Note: The event packet is limited to a maximum of 255 octets in the payload; since an HCI Command Packet has 3 octets of header data, only the first 252 octets of the command parameters will be returned.

Event Parameters:

HCI_Command_Packet: *Size: Depends on Command*

Value	Parameter Description
0xXXXXXX	HCI Command Packet, including header.

7.7.26 Data Buffer Overflow Event

Event	Event Code	Event Parameters
Data Buffer Overflow	0x1A	Link_Type

Description:

This event is used to indicate that the Controller's data buffers have been overflowed. This can occur if the Host has sent more packets than allowed. The Link_Type parameter is used to indicate that the overflow was caused by ACL or synchronous data.

Event Parameters:

Link_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Synchronous Buffer Overflow (Voice Channels).
0x01	ACL Buffer Overflow (Data Channels).
0x02-0xFF	Reserved for future use.

7.7.27 Max Slots Change Event

Event	Event Code	Event Parameters
Max Slots Change	0x1B	Connection_Handle, LMP_Max_Slots

Description:

This event is used to notify the Host about the LMP_Max_Slots parameter when the value of this parameter changes. It shall be sent each time the maximum allowed length, in number of slots, for baseband packets transmitted by the local device, changes. The Connection_Handle will be a Connection_Handle for an ACL connection.

Event Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

LMP_Max_Slots: *Size: 1 octet*

Value	Parameter Description
0x01, 0x03, 0x05	Maximum number of slots allowed to use for baseband packets, see Link Manager Protocol Specification, Section 4.1.10, on page 244 and Section 5.2 on page 320.

7.7.28 Read Clock Offset Complete Event

Event	Event Code	Event Parameters
Read Clock Offset Complete	0x1C	Status, Connection_Handle, Clock_Offset

Description:

The Read Clock Offset Complete event is used to indicate the completion of the process of the Link Manager obtaining the Clock Offset information of the BR/EDR Controller specified by the Connection_Handle event parameter. The Connection_Handle will be a Connection_Handle for an ACL connection.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Clock_Offset command succeeded.
0x01-0xFF	Read_Clock_Offset command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Specifies which Connection_Handle's Clock Offset parameter is returned. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Clock_Offset: *Size: 2 Octets*

Bit format	Parameter Description
Bit 14-0	Bit 16-2 of CLKslave-CLKmaster.
Bit 15	Reserved.

7.7.29 Connection Packet Type Changed Event

Event	Event Code	Event Parameters
Connection Packet Type Changed	0x1D	Status, Connection_Handle, Packet_Type

Description:

The Connection Packet Type Changed event is used to indicate that the process has completed of the Link Manager changing which packet types can be used for the connection. This allows current connections to be dynamically modified to support different types of user data. The Packet_Type event parameter specifies which packet types the Link Manager can use for the connection identified by the Connection_Handle event parameter for sending L2CAP data or voice. The Packet_Type event parameter does not decide which packet types the LM is allowed to use for sending LMP PDUs.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Connection Packet Type changed successfully.
0x01-0xFF	Connection Packet Type Changed failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Packet_Type: *Size: 2 Octets*

For ACL_Link_Type

Value	Parameter Description
0x0001	Reserved for future use.
0x0002	2-DH1 may not be used.
0x0004	3-DH1 may not be used.
0x0008[1]	DM1 may be used.
0x0010	DH1 may be used.
0x0020	Reserved for future use.
0x0040	Reserved for future use.
0x0080	Reserved for future use.

0x0100	2-DH3 may not be used.
0x0200	3-DH3 may not be used.
0x0400	DM3 may be used.
0x0800	DH3 may be used.
0x1000	2-DH5 may not be used.
0x2000	3-DH5 may not be used.
0x4000	DM5 may be used.
0x8000	DH5 may be used.

1. This bit will be interpreted as set to 1 by Bluetooth V1.2 or later controllers.

For SCO_Link_Type

Value	Parameter Description
0x0001	Reserved for future use.
0x0002	Reserved for future use.
0x0004	Reserved for future use.
0x0008	Reserved for future use.
0x0010	Reserved for future use.
0x0020	HV1
0x0040	HV2
0x0080	HV3
0x0100	Reserved for future use.
0x0200	Reserved for future use.
0x0400	Reserved for future use.
0x0800	Reserved for future use.
0x1000	Reserved for future use.
0x2000	Reserved for future use.
0x4000	Reserved for future use.
0x8000	Reserved for future use.

7.7.30 QoS Violation Event

Event	Event Code	Event Parameters
QoS Violation	0x1E	Handle

Description:

The QoS Violation event is used to indicate the Controller is unable to provide the current QoS requirement for the Connection or Logical Link Handle. This event indicates that the Controller is unable to provide one or more of the agreed QoS parameters.

The Host chooses what action should be done. With a BR/EDR Controller the Host can reissue the QoS_Setup command to renegotiate the QoS setting for Connection_Handle. With an AMP Controller the Host can reissue the Flow_Spec_Modify command.

Event Parameters:

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Handle for the link that the Controller cannot provide the current QoS requested. The Handle is a Connection_Handle for a BR/EDR Controller and a Logical_Link_Handle for an AMP Controller. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

7.7.31 Page Scan Repetition Mode Change Event

Event	Event Code	Event Parameters
Page Scan Repetition Mode Change	0x20	BD_ADDR, Page_Scan_Repetition_Mode

Description:

The Page Scan Repetition Mode Change event indicates that the remote BR/EDR Controller with the specified BD_ADDR has successfully changed the Page_Scan_Repetition_Mode (SR).

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the remote device.

Page_Scan_Repetition_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	R0
0x01	R1
0x02	R2
0x03 – 0xFF	Reserved.

7.7.32 Flow Specification Complete Event

Event	Event Code	Event Parameters
Flow Specification Complete	0x21	Status, Connection_Handle, Flags, Flow_direction, Service_Type, Token_Rate, Token_Bucket_Size, Peak_Bandwidth, Access Latency

Description:

The Flow Specification Complete event is used to inform the Host about the Quality of Service for the ACL connection the Controller is able to support. The Connection_Handle will be a Connection_Handle for an ACL connection. The flow parameters refer to the outgoing or incoming traffic of the ACL link, as indicated by the Flow_direction field. The flow parameters are defined in the L2CAP specification [Vol 3] Part A, Section 5.3, Quality of Service (QoS) Option. When the Status parameter indicates a successful completion, the flow parameters specify the agreed values by the Controller. When the Status parameter indicates a failed completion with the Error Code *QoS Unacceptable Parameters* (0x2C), the flow parameters specify the acceptable values of the Controller. This enables the Host to continue the 'QoS negotiation' with a new HCI Flow_Specification command with flow parameter values that are acceptable for the Controller. When the Status parameter indicates a failed completion with the Error Code *QoS Rejected* (0x2D), this indicates a request of the Controller to discontinue the 'QoS negotiation'. When the Status parameter indicates a failed completion, the flow parameter values of the most recently successful completion must be assumed (or the default values when there was no success completion).

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Flow_Specification command succeeded
0x01 – 0xFF	Flow_Specification command failed. See Part D, Error Codes on page 339 for list of Error Codes

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle used to identify for which ACL connection the Flow is specified. Range: 0x0000 - 0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Flags: *Size: 1 Octet)*

Value	Parameter Description
0x00 – 0xFF	Reserved for future use.

Flow_direction: *Size: 1 Octet*

Value	Parameter Description
0x00	Outgoing Flow i.e. traffic send over the ACL connection
0x01	Incoming Flow i.e. traffic received over the ACL connection
0x02 – 0xFF	Reserved for future use.

Service_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	No Traffic
0x01	Best Effort
0x02	Guaranteed
0x03 – 0xFF	Reserved for future use

Token Rate: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Token Rate in octets per second

Token Bucket Size 4 Octets

Value	Parameter Description
0xXXXXXXXX	Token Bucket Size in octets

Peak_Bandwidth: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Peak Bandwidth in octets per second

Access Latency: *Size: 4 Octets*

Value	Parameter Description
0xXXXXXXXX	Access Latency in microseconds

7.7.33 Inquiry Result with RSSI Event

Event	Event Code	Event Parameters
Inquiry Result with RSSI	0x22	Num_responses, BD_ADDR[i], Page_Scan_Repetition_Mode[i], Reserved[i], Class_of_Device[i], Clock_Offset[i], RSSI[i]

Description:

The Inquiry Result with RSSI event indicates that a BR/EDR Controller or multiple BR/EDR Controllers have responded so far during the current Inquiry process. This event will be sent from the BR/EDR Controller to the Host as soon as an Inquiry Response from a remote device is received if the remote device supports only mandatory paging scheme. This BR/EDR Controller may queue these Inquiry Responses and send multiple BR/EDR Controllers information in one Inquiry Result event. The event can be used to return one or more Inquiry responses in one event. The RSSI parameter is measured during the FHS packet returned by each responding slave.

This event shall only be generated if the Inquiry Mode parameter of the last Write_Inquiry_Mode command was set to 0x01 (Inquiry Result format with RSSI).

Event Parameters:

Num_Responses: *Size: 1 Octet*

Value	Parameter Description
0xXX	Number of responses from the Inquiry.

BD_ADDR[i]: *Size: 6 Octets * Num_Responses*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for each device which responded.

Page_Scan_Repetition_Mode[i]: *Size: 1 Octet* Num_Responses*

Value	Parameter Description
0x00	R0
0x01	R1
0x02	R2

0x03 – 0xFF	Reserved

Reserved[i]:[1] *Size: 1 Octet* Num_Responses*

Value	Parameter Description
0xXX	Reserved.

Class_of_Device[i]: *Size: 3 Octets * Num_Responses*

Value	Parameter Description
0xXXXXXX	Class of Device for the device

Clock_Offset[i]: *Size: 2 Octets * Num_Responses*

Bit format	Parameter Description
Bit 14-0	Bit 16-2 of CLKslave-CLKmaster.
Bit 15	Reserved

RSSI[i]: *Size: 1 Octet * Num_Responses*

Value	Parameter Description
0xXX	Range: -127 to +20 Units: dBm

1. This was the Page_Scan_Period_Mode parameter in the v1.1 specification. This parameter has no meaning in v1.2 or later and no default value.

7.7.34 Read Remote Extended Features Complete Event

Event	Event Code	Event Parameters
Read Remote Extended Features Complete	0x23	Status, Connection_Handle, Page_Number, Maximum page number, Extended_LMP_Features

Description:

The Read Remote Extended Features Complete event is used to indicate the completion of the process of the Link Manager obtaining the remote extended LMP features of the remote device specified by the Connection_Handle event parameter. The Connection_Handle will be a Connection_Handle for an ACL connection. The event parameters include a page of the remote devices extended LMP features. For details see Part C, Link Manager Protocol Specification on page 207.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Request for remote extended features succeeded
0x01-0xFF	Request for remote extended features failed. See Part D, Error Codes on page 339 for error codes.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection_Handle identifying the device to which the remote features apply. Range: 0x0000-0x0EFF (0x0F00-0x0FFF Reserved for future use)

Page Number: *Size: 1Octet*

Value	Parameter Description
0x00	The normal LMP features as returned by Read_Remote_Supported_Features command
0x01-0xFF	The page number of the features returned

Maximum Page Number: *Size: 1Octet*

Value	Parameter Description
0x00-0xFF	The highest features page number which contains non-zero bits for the local device

Extended_LMP_Features: *Size: 8 Octets*

Value	Parameter Description
0xFFFFFFFFFFFFFFFF	Bit map of requested page of LMP features. See LMP Part C, Section 3.3, Feature Mask Definition for details.

7.7.35 Synchronous Connection Complete Event

Event	Event Code	Event Parameters
Synchronous Connection Complete	0x2C	Status, Connection_Handle, BD_ADDR, Link Type, Transmission_Interval, Retransmission Window, Rx_Packet_Length, Tx_Packet_Length Air Mode

Description:

The Synchronous Connection Complete event indicates to both the Hosts that a new synchronous connection has been established. This event also indicates to the Host, which issued the Setup_Synchronous_Connection, or Accept_Synchronous_Connection_Request or Reject_Synchronous_Connection_Request command and then received a Command Status event, if the issued command failed or was successful.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Connection successfully completed.
0x01 –0xFF	Connection failed to complete. See Part D, Error Codes on page 339 for error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection between two BR/EDR Controllers. Range: 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the other connected device forming the connection.

Link_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	SCO Connection

Value	Parameter Description
0x01	Reserved
0x02	eSCO Connection
0x03 – 0xFF	Reserved

Transmission_Interval: *Size: 1 Octet*

Value	Parameter Description
0xXX	Time between two consecutive eSCO instants measured in slots. Must be zero for SCO links.

Retransmission window: *Size: 1 Octet*

Value	Parameter Description
0xXX	The size of the retransmission window measured in slots. Must be zero for SCO links.

Rx_Packet_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Length in bytes of the eSCO payload in the receive direction. Must be zero for SCO links.

Tx_Packet_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Length in bytes of the eSCO payload in the transmit direction. Must be zero for SCO links.

Air Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	μ-law log
0x01	A-law log
0x02	CVSD
0x03	Transparent Data
0x04 – 0xFF	Reserved

7.7.36 Synchronous Connection Changed Event

Event	Event Code	Event Parameters
Synchronous Connection Changed	0x2D	Status, Connection_Handle, Transmission_Interval, Retransmission Window, Rx_Packet_Length, Tx_Packet_Length

Description:

The Synchronous Connection Changed event indicates to the Host that an existing synchronous connection has been reconfigured. This event also indicates to the initiating Host (if the change was host initiated) if the issued command failed or was successful.

Command Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Connection successfully reconfigured.
0x01 –0xFF	Reconfiguration failed to complete. See Part D, Error Codes on page 339 for error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection between two BR/EDR Controllers. Range: 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Transmission_Interval: *Size: 1 Octet*

Value	Parameter Description
0xXX	Time between two consecutive SCO/eSCO instants measured in slots.

Retransmission window: *Size: 1 Octet*

Value	Parameter Description
0xXX	The size of the retransmission window measured in slots. Must be zero for SCO links.

Rx_Packet_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Length in bytes of the SCO/eSCO payload in the receive direction.

Tx_Packet_Length: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Length in bytes of the SCO/eSCO payload in the transmit direction.

7.7.37 Sniff Subrating Event

Event	Event Code	Event Parameters
Sniff Subrating	0x2E	Status Connection_Handle Maximum_Transmit_Latency Maximum_Receive_Latency Minimum_Remote_Timeout Minimum_Local_Timeout

Description:

The Sniff Subrating event indicates that the device associated with the Connection_Handle has either enabled sniff subrating or the sniff subrating parameters have been renegotiated by the link manager. The Connection_Handle will be a Connection_Handle for an ACL connection. The Connection_Handle event parameter indicates which connection the Sniff Subrating event is for.

Each BR/EDR Controller that is associated with the Connection_Handle that has changed its subrating parameters will send the Sniff Subrating event to its host.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Sniff_Subrating command succeeded
0x01 – 0xFF	Subrating command failed to complete. See Part D, Error Codes on page 339 for error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle. Range: 0x0000 – 0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Maximum_Transmit_Latency: *Size: 2 Octet*

Value	Parameter Description

N = 0xXXXX	Maximum latency for data being transmitted from the local device to the remote device. Latency = N * 0.625 msec (1 Baseband slot) Range for N: 0x0000 – 0xFFFE Time Range: 0 sec - 40.9 sec

Maximum_Receive_Latency: *Size: 2 Octet*

Value	Parameter Description
N = 0xXXXX	Maximum latency for data being received by the local device from the remote device. Latency = N * 0.625 msec (1 Baseband slot) Range for N: 0x0000 – 0xFFFE Time Range: 0 sec - 40.9 sec

Minimum_Remote_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	The base sniff subrate timeout in baseband slots that the remote device shall use. Timeout = N * 0.625 msec (1 Baseband slot) Range for N: 0x0000 – 0x8000 Time Range: 0 sec – 40.9 sec

Minimum_Local_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	The base sniff subrate timeout in baseband slots that the local device will use. Timeout = N * 0.625 msec (1 Baseband slot) Range for N: 0x0000 – 0x8000 Time Range: 0 sec – 40.9 sec

7.7.38 Extended Inquiry Result Event

Event	Event Code	Event Parameters
Extended Inquiry Result	0x2F	Num_Responses, BD_ADDR, Page_Scan_Repetition_Mode, Reserved, Class_of_Device, Clock_Offset, RSSI, Extended_Inquiry_Response

Description:

The Extended Inquiry Result event indicates that a BR/EDR Controller has responded during the current inquiry process with extended inquiry response data. This event will be sent from the BR/EDR Controller to the Host upon reception of an Extended Inquiry Response from a remote device. One single Extended Inquiry Response is returned per event. This event contains RSSI and inquiry response data for the BR/EDR Controller that responded to the latest inquiry. The RSSI parameter is measured during the FHS packet returned by each responding slave. The Num_Responses parameter shall be set to one.

This event is only generated if the Inquiry_Mode parameter of the last Write_Inquiry_Mode command was set to 0x02 (Inquiry Result with RSSI format or Extended Inquiry Result format).

Note: This ensures that a Host that does not support Extended Inquiry Results will never receive the Extended Inquiry Result event.

If an inquiry response packet with the EIR field set to zero is received, the Inquiry Result with RSSI event format shall be used. If the EIR bit is set to one the Extended Inquiry Result event format shall be used. If the EIR bit is set to one but the Controller failed to receive the extended inquiry response packet, the Extended_Inquiry_Response parameter is set to zeros. If an extended inquiry response packet from the same device is correctly received in a later response, another event shall be generated.

Note: The only difference between the Extended Inquiry Result event and the Inquiry Result with RSSI event is the additional Extended_Inquiry_Response parameter.

Note: The Extended_Inquiry_Response parameter is not interpreted by the Controller. The tagged data set by the other Host should be passed unaltered if it has been correctly received.

Event Parameters:

Num_Responses: *Size: 1 Octet*

Value	Parameter Description
0x01	Number of responses from the inquiry. The Extended Inquiry Result event always contains a single response.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR for the device that responded.

Page_Scan_Repetition_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	R0
0x01	R1
0x02	R2
0x03 - 0xFF	Reserved

Reserved: *Size: 1 Octet*

Value	Parameter Description
0xXX	Reserved.

Class_of_Device: *Size: 3 Octets*

Value	Parameter Description
0xXXXXXX	Class of Device for the device that responded.

Clock_Offset *Size: 2 Octets*

Value	Parameter Description
Bit 14-0	Bit 16-2 of CLKslave-CLKmaster.
Bit 15	Reserved.

RSSI: *Size: 1 Octet*

Value	Parameter Description
0xXX	Range: -127 to +20
	Units: dBm

Extended_Inquiry_Response: *Size: 240 Octets*

Value	Parameter Description
	Extended Inquiry Response data as defined in [Vol 3] Part C, Section 8

7.7.39 Encryption Key Refresh Complete Event

Command	Event Code	Event Parameters
Encryption Key Refresh Complete	0x30	Status Connection_Handle

Description:

The Encryption Key Refresh Complete event is used to indicate to the Host that the encryption key was refreshed on the given Connection_Handle any time encryption is paused and then resumed. The BR/EDR Controller shall send this event when the encryption key has been refreshed due to encryption being started or resumed.

If the Encryption Key Refresh Complete event was generated due to an encryption pause and resume operation embedded within a change connection link key procedure, the Encryption Key Refresh Complete event shall be sent prior to the Change Connection Link Key Complete event.

If the Encryption Key Refresh Complete event was generated due to an encryption pause and resume operation embedded within a role switch procedure, the Encryption Key Refresh Complete event shall be sent prior to the Role Change event.

Event parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Encryption Key Refresh completed successfully
0x01 - 0xFF	Encryption Key Refresh failed. See Part D, Error Codes on page 339 for list of Error Codes

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection Handle for the ACL connection to have the encryption key refreshed on. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use).

7.7.40 IO Capability Request Event

Event	Event Code	Event Parameters
IO Capability Request	0x31	BD_ADDR

Description:

The IO Capability Request event is used to indicate that the IO capabilities of the Host are required for a simple pairing process. The Host shall respond with an IO_Capability_Request_Reply command. This event shall only be generated if simple pairing has been enabled with the Write_Simple_Pairing_Mode command.

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

7.7.41 IO Capability Response Event

Event	Event Code	Event Parameters
IO Capability Response	0x32	BD_ADDR, IO_Capability, OOB_Data_Present, Authentication_Requirements

Description:

The IO Capability Response event is used to indicate to the host that IO capabilities from a remote device specified by BD_ADDR have been received during a simple pairing process. This event will only be generated if simple pairing has been enabled with the Write_Simple_Pairing_Mode command.

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR identifying the device to which the IO capabilities apply.

IO_Capability: *Size: 1 Octet*

Value	Parameter Description
0x00	DisplayOnly
0x01	DisplayYesNo
0x02	KeyboardOnly
0x03	NoInputNoOutput
0x04 – 0xFF	Reserved for future use

OOB_Data_Present: *Size: 1 Octet*

Value	Parameter Description
0x00	OOB authentication data not present
0x01	OOB authentication data from remote device present
0x02 – 0xFF	Reserved for future use

Authentication_Requirements: *Size: 1 Octet*

Value	Parameter Description
0x00	MITM Protection Not Required –No Bonding. Numeric comparison with automatic accept allowed.

0x01	MITM Protection Required – No Bonding. Use IO Capabilities to determine authentication procedure
0x02	MITM Protection Not Required – Dedicated Bonding. Numeric comparison with automatic accept allowed.
0x03	MITM Protection Required – Dedicated Bonding. Use IO Capabilities to determine authentication procedure
0x04	MITM Protection Not Required – General Bonding. Numeric Comparison with automatic accept allowed.
0x05	MITM Protection Required – General Bonding. Use IO capabilities to determine authentication procedure.
0x06 - 0xFF	Reserved for future use

7.7.42 User Confirmation Request Event

Event	Event Code	Event Parameters
User Confirmation Request	0x33	BD_ADDR, Numeric_Value

Description:

The User Confirmation Request event is used to indicate that user confirmation of a numeric value is required. The Host shall reply with either the User_Confirmation_Request_Reply or the User_Confirmation_Request_Negative_Reply command. If the Host has output capability (DisplayYesNo or KeyboardOnly), it shall display the Numeric_Value until the Simple Pairing Complete event is received. It shall reply based on the yes/no response from the user. If the Host has no input and no output it shall reply with the User Confirmation Request Reply command. When the Controller generates a User Confirmation Request event, in order for the local Link Manager to respond to the request from the remote Link Manager, the local Host must respond with either a User_Confirmation_Request_Reply or a User_Confirmation_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout. (See Part C, Link Manager Protocol Specification on page 207.)

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of device involved in Simple Pairing process.

Numeric_Value: *Size: 4 Octets*

Value	Parameter Description
0x00000000 – 0x000F423F	Numeric value to be displayed. Valid values are decimal 000000 – 999999.

7.7.43 User Passkey Request Event

Event	Event Code	Event Parameters
User Passkey Request	0x34	BD_ADDR

Description:

The User Passkey Request event is used to indicate that a passkey is required as part of a Simple Pairing process. The Host shall respond with either a User_Passkey_Request_Reply or User_Passkey_Request_Negative_Reply command. This event will only be generated if simple pairing has been enabled with the Write_Simple_Pairing_Mode command. When the Controller generates a User Passkey Request event, in order for the local Link Manager to respond to the request from the remote Link Manager, the local Host must respond with either a User_Passkey_Request_Reply or User_Passkey_Request_Negative_Reply command before the remote Link Manager detects LMP response timeout. (See Part C, Link Manager Protocol Specification on page 207.)

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the device involved in simple pairing process.

7.7.44 Remote OOB Data Request Event

Event	Event Code	Event Parameters
Remote OOB Data Request	0x35	BD_ADDR

Description:

The Remote OOB Data Request event is used to indicate that the Simple Pairing Hash C and the Simple Pairing Randomizer R is required for the Secure Simple Pairing process involving the device identified by BD_ADDR. The C and R values were transferred to the Host from the remote device via an OOB mechanism. This event is sent by the Controller because the Host previously set the OOB Data Present parameter to "OOB authentication data from remote device present" in an IO Capability Request Reply command. The Host shall respond with the Remote OOB Data Request Reply command.

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the device from which the C and R values were received.

7.7.45 Simple Pairing Complete Event

Event	Event Code	Event Parameters
Simple Pairing Complete	0x36	Status, BD_ADDR

Description:

The Simple Pairing Complete event is used to indicate that the simple pairing process has completed. A Host that is displaying a numeric value can use this event to change its UI.

When the LMP simple pairing sequences fail for any reason, the Simple Pairing Complete event shall be sent to the Host. When Simple Pairing Complete event is sent in response to the IO capability exchange failing, the Status parameter shall be set to the error code received from the remote device. Otherwise, the Status shall be set to the error code “Authentication Failure.”

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Simple Pairing succeeded
0x01 - 0xFF	Simple Pairing failed. See Part D, Error Codes on page 339 for a list of Error Codes.

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the device involved in simple pairing process.

7.7.46 Link Supervision Timeout Changed Event

Event	Event Code	Event Parameters
Link Supervision Timeout Changed	0x38	Connection_Handle, Link_Supervision_Timeout

Description:

The Link_Supervision_Timeout_Changed event is used to notify the slave's Host when the Link_Supervision_Timeout parameter is changed in the slave Controller. This event shall only be sent to the Host by the slave controller upon receiving an LMP_supervision_timeout PDU from the master.

Note: The Connection_Handle used for this command shall be the ACL connection of the appropriate device.

Event Parameters:

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection Handle Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Link_Supervision_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Measured in Number of Baseband slots Link_Supervision_Timeout = N*0.625 msec (1 Baseband slot) Range for N: 0x0001 - 0xFFFF Time Range: 0.625ms - 40.9 sec (0 means infinite timeout)

7.7.47 Enhanced Flush Complete Event

Event	Event Code	Event parameter
Enhanced Flush Complete	0x39	Handle

Description:

The Enhanced Flush Complete event is used to indicate that an Enhanced Flush is complete for the specified Handle.

Event Parameters:

Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Handle of the link for which Controller cannot provide the requested QoS. The Handle is a Connection_Handle for a BR/EDR Controller and a Logical_Link_Handle for an AMP Controller. Range: 0x0000 - 0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

7.7.48 User Passkey Notification Event

Event	Event Code	Event Parameters
User Passkey Notification	0x3B	BD_ADDR, Passkey

Description:

The User Passkey Notification event is used to provide a passkey for the Host to display to the user as required as part of a Simple Pairing process. The Passkey parameter shall be a randomly generated number (see [Vol. 2], Part H, Section 2 on page 1061) generated by the Controller modulo 1,000,000.

This event will be generated if the IO capabilities of the local device are DisplayOnly or DisplayYesNo and the IO capabilities of the remote device are KeyboardOnly.

This event shall only be generated if simple pairing has been enabled with the Write_Simple_Pairing_Mode command.

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the device involved in simple pairing process.

Passkey: *Size: 4 Octets*

Value	Parameter Description
0x00000000 - 0x000F423F	Passkey to be displayed. Valid values are decimal 000000 - 999999.

7.7.49 Keypress Notification Event

Command	Event Code	Event Parameters
Keypress Notification	0x3C	BD_ADDR, Notification_Type

Description:

The Keypress Notification event is sent to the Host after a passkey notification has been received by the Link Manager on the given BD_ADDR. The Notification_Type parameter may be used by the Host's user interface.

Event parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device involved in simple pairing process

Notification_Type: *Size: 1 Octet*

Value	Parameter Description
0	Passkey entry started
1	Passkey digit entered
2	Passkey digit erased
3	Passkey cleared
4	Passkey entry completed
5-255	Reserved for future use

7.7.50 Remote Host Supported Features Notification Event

Command	Event Code	Event Parameters
Remote Host Supported Features Notification	0x3D	BD_ADDR Host_Supported_Features

Description:

The Remote Host Supported Features Notification event is used to return the LMP extended features page containing the Host features. The BD_ADDR shall be the address of the remote device.

This event shall only be generated after the LMP extended features are read from the remote device during a connection initiated by a Remote Name Request command.

Note: This event is not generated during a connection initiated by the Create_Connection command.

Event parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of remote device.

Host_Supported_Features: *Size: 8 Octets*

Value	Parameter Description
0xFFFFFFFFFFFFFFFF	Bit map of Host Supported Features page of LMP extended features. For more information, see "Part C, Link Manager Protocol Specification" .

7.7.51 Physical Link Complete Event

Event	Event Code	Event Parameters
Physical Link Complete	0x40	Status, Physical_Link_Handle

Description:

The Physical Link Complete event indicates to the Host that a new AMP physical link has been established. This event is used as a response for either Create_Physical_Link or Accept_Physical_Link commands and is preceded by a Command Status event.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Connection successfully completed.
0x01-0xFF	Connection failed to Complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle identifying the physical link which has been created.

7.7.52 Channel Selected Event

Event	Event Code	Event Parameters
Channel Selected	0x41	Physical_Link_Handle

Description:

The Channel Selected event indicates to the Host originating the physical link that the link information data is available to be read using the Read Local AMP Assoc command. This allows the originating device to determine link implementation parameters such as an underlying physical channel choice. This structure shall be sent in the AMP Create Physical Link Request (see [Vol 3] Part E, Section 3.11]) and be used at the remote end in the Accept_Physical_Link command.

Event Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle to the established physical link.

7.7.53 Disconnection Physical Link Complete Event

Event	Event Code	Event Parameters
Disconnection Physical Link Complete	0x42	Status, Physical_Link_Handle, Reason

Description:

The Disconnection Physical Link Complete event occurs when a physical link is terminated. The status parameter indicates if the disconnection was successful or not.

When a physical link fails, one Disconnection Logical Link Complete event will be returned for each logical channel on the physical link with the corresponding Logical_Link_Handle as a parameter.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Disconnection has occurred.
0x01-0xFF	Disconnection failed to complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle which was disconnected.

Reason: *Size: 1 Octet*

Value	Parameter Description
0xXX	Reason for disconnection. See Part D, Error Codes on page 339 for error codes and descriptions.

7.7.54 Physical Link Loss Early Warning Event

Event	Event Code	Event Parameters
Physical Link Loss Early Warning	0x43	Physical_Link_Handle, Link_Loss_Reason

Description:

The Physical Link Loss Early Warning event occurs when a physical link has early indication that the link may be disrupted. Data transmission and reception may suffer impairment on this link. The Host may use this to indicate to the user that traffic may be disrupted.

The exact low level meaning is dependent on the AMP type. Some AMP types may never generate this event.

Event Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle to which the warning refers.

Link_Loss_Reason: *Size: 1 Octet*

Value	Parameter Description
0	Unknown
1	Range related
2	Bandwidth related
3	Resolving conflict
4	Interference
5-255	Reserved

7.7.55 Physical Link Recovery Event

Event	Event Code	Event Parameters
Physical Link Recovery	0x44	Physical_Link_Handle

Description:

The Physical Link Recovery event indicates that whatever interruption caused an earlier Physical Link Loss Early Warning event has now been cleared. The normal data transmission and reception service can be expected. The exact low level meaning is dependent on the AMP type. The Host may use this event to cancel any indication to the user caused by the Physical Link Loss Early Warning event.

Event Parameters:

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle to which the warning referred.

7.7.56 Logical Link Complete Event

Event	Event Code	Event Parameters
Logical Link Complete	0x45	Status, Logical_Link_Handle, Physical_Link_Handle, TX_Flow_Spec_ID

Description:

The Logical Link Complete event indicates to both of the Hosts forming the connection whether a new connection has been established successfully.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Connection successfully completed.
0x01-0xFF	Connection failed to Complete. See Part D, Error Codes on page 339for a list of error codes and descriptions.

Logical_Link_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Logical_Link_Handle to be used to identify a connection between two BR/EDR Controllers with communicating AMPs. The Logical_Link_Handle is used as an identifier for transmitting and receiving data. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link identifying the physical link over which the logical link has been created.

TX_Flow_Spec_ID: *Size: 1 Octet*

Value	Parameter Description
0xXX	Flow Spec ID identifying the logical link creation that completed. This matches the ID field of the TX_Flow_Spec

7.7.57 Disconnection Logical Link Complete Event

Event	Event Code	Event Parameters
Disconnection Logical Link Complete	0x46	Status, Logical_Link_Handle, Reason

Description:

The Disconnection Logical Link Complete event occurs when a logical link is terminated on the local Controller. The status parameter indicates if the disconnection was successful or not.

Note: When the local Controller disconnects a logical link, the remote Controller will not be notified.

Note: When a physical link fails, one Disconnection Logical Link Complete event will be returned for each logical channel on the physical link with the corresponding Logical_Link_Handle as a parameter.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Disconnection has occurred.
0x01-0xFF	Disconnection failed to complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Logical_Link_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Logical_Link_Handle which was disconnected. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Reason: *Size: 1 Octet*

Value	Parameter Description
0xXX	Reason for disconnection. See Part D, Error Codes on page 339 for error codes and descriptions.

7.7.58 Flow Spec Modify Complete Event

Event	Event Code	Event Parameters
Flow Spec Modify Complete	0x47	Status, Handle

Description:

The Flow Spec Modify Complete event indicates to the Host that a Flow Spec Modify command has been completed. This event also indicates to the Host which issued the Flow_Spec_Modify command and then received a Command Status event if the issued command failed or was successful.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Flow_Spec_Modify command successfully completed.
0x01-0xFF	Flow_Spec_Modify command failed to Complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle if the receiving Controller is a BR/EDR Controller. Logical_Link_Handle, if the receiving Controller is an AMP Controller, identifying the logical link to be changed. Logical_Link_Handle to be used to identify a connection between two BR/EDR Controllers with communicating AMPs. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

7.7.59 Number Of Completed Data Blocks Event

Event	Event Code	Event Parameters
Number Of Completed Data Blocks	0x48	Total_Num_Data_Blocks, Number_of_Handles, Handle[i], Num_Of_Completed_Packets[i], Num_Of_Completed_Blocks[i]

Description:

The Number Of Completed Data Blocks event is used by the Controller to indicate to the Host how many HCI ACL Data Packets have been completed (transmitted or flushed), and how many data block buffers have been freed, for each Handle (a Connection_Handle if the Controller is a BR/EDR Controller or a Logical_Link_Handle if the Controller is an AMP Controller) since the previous Number Of Completed Data Blocks event was sent to the Host. This means that the corresponding buffer space has been freed in the Controller. Based on this information, and the Total_Num_Data_Blocks parameter, the Host can determine for which Handles the following HCI ACL Data Packets should be sent to the Controller.

The Host should determine the number of blocks occupied by each ACL data packet by dividing the ACL data packet size by the Data_Block_Length parameter of the Read_Data_Block_Size command.

The Total_Num_Data_Blocks parameter indicates the total number of buffer blocks available in the Controller. Before any Number of Complete Data Blocks event is received, the value of Total_Num_Data_Blocks from the Read_Data_Block_Size command is used. This allows the value to be updated at any time, which provides the Controller with some flexibility on its buffer allocation.

If the Controller were permitted to reduce its buffer pool in an arbitrary way then there is a potential race condition, in the case where the Host has just started to transmit a new packet. In order to prevent this race condition, the Total_Num_Data_Blocks parameter shall not indicate a reduction in the pool of blocks greater than the sum of the Num_Of_Completed_Blocks values in this event. If a greater reduction in the block pool is required then the value 0 shall be indicated here. The value 0 has a special meaning and indicates that the Host shall re-issue the Read Data Block Size command in order to find the new buffer pool size. The Host shall wait for any outstanding TX to complete and shall defer further TX until the Read_Data_Block_Size command has been issued and completed. The Controller shall reduce its allocation only after the Read_Data_Block_Size command has been issued and completed. This ensures that the race condition described above is avoided.

Event Parameters:

Total_Num_Data_Blocks: *Size: 2 Octets*

Value	Parameter Description
0x0000	The size of the buffer pool may have changed. The Host is requested to issue a Read Data Block Size command in order to determine the new value of Total_Num_Data_Blocks.
0xXXXX	Total number of data block buffers available in the Controller for the storage of data packets scheduled for transmission. This indicates the existing value is unchanged, or increased, or reduced by up to the sum of the Num_Of_Completed_Blocks values in this command.

Number_of_Handles: *Size: 1 Octet*

Value	Parameter Description
0xXX	The number of Handles and Num_Of_Completed_Packets and Num_Of_Completed_Blocks parameter triples contained in this event.
Range: 0-255	

Handle[i]: *Size: Number_of_Handles * 2 Octets(12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Handle (Connection_Handle for a BR/EDR Controller or a Logical_Link_Handle for an AMP Controller). Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Num_Of_Completed_Packets [i]: *Size: Number_of_Handles * 2 Octets*

Value	Parameter Description
N = 0xXXXX	The number of HCI ACL Data Packets that have been completed (transmitted or flushed) for the associated Handle since the previous time that a Number Of Completed Data Blocks event provided information about this Handle. Range for N: 0x0000-0xFFFF

Num_Of_Completed_Blocks [i]: *Size: Number_of_Handles * 2 Octets*

Value	Parameter Description
N = 0xXXXX	The number of data blocks that have been freed for the associated Handle since the previous time that a Number Of Completed Data Blocks event provided information about this Handle. Range for N: 0x0000-0xFFFF

7.7.60 Short Range Mode Change Complete Event

Event	Event Code	Event Parameters
Short_ Range_ Mode_ Change_Complete	0x4C	Status, Physical_Link_Handle, Short_Range_Mode_State

Description:

The Short_Range_Mode_Change_Complete event occurs after the AMP Controller has been notified to enable or disable Short Range Mode for a given physical link. The status parameter indicates if the configuration change was successful or not.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Short Range Mode configuration has occurred.
0x01-0xFF	Short Range Mode configuration failed to complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Physical_Link_Handle: *Size: 1 Octet*

Value	Parameter Description
0xXX	Physical_Link_Handle to which the configuration applies.

Short_Range_Mode_State: *Size: 1 Octet*

Value	Parameter Description
0xXX	State of Short Range Mode in controller. 0 - Short Range Mode disabled 1 - Short Range Mode enabled 0x02 - 0xFF - reserved

7.7.61 AMP Status Change Event

Event	Event Code	Event Parameters
AMP_ Status_ Change	0x4D	Status, AMP_Status

Description:

The AMP Status Change event can occur at any time, after initial Read_Local_AMP_Info_Request command, and when the AMP Controller detects a change has occurred regarding status and is based on availability thresholds.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	AMP_Status_Change has occurred.
0x01-0xFF	AMP_ Status_Change failed to complete. See Part D, Error Codes on page 339for a list of error codes and descriptions.

AMP_Status: *Size: 1 Octet*

Value	Parameter Description
0x00	The Controller radio is available but is currently physically powered down. This value should be used if the AMP Controller is present and can be powered up by the AMP Manager. This value indicates that there may be a cost of time and power to use this AMP Controller (i.e., the time taken and power required to power up the AMP Controller). These costs are AMP type and AMP implementation dependent.
0x01	This value indicates that the AMP Controller is only used by Bluetooth technology and will not be shared with other non-Bluetooth technologies. This value shall only be used if the AMP Controller is powered up. This value does not indicate how much bandwidth is currently free on the AMP Controller.
0x02	The AMP Controller has no capacity available for Bluetooth operation This value indicates that all of the AMP Controllers bandwidth is currently allocated to servicing a non Bluetooth technology. A device is permitted to create a Physical Link to an AMP Controller that has this status. This value shall only be used if the AMP Controller is powered up.

Value	Parameter Description
0x03	The AMP Controller has low capacity available for Bluetooth operation This value indicates that the majority of the AMP Controllers bandwidth is currently allocated to servicing a non Bluetooth technology. An AMP Controller that with capacity in the approximate range of 0 < capacity < 30% should indicate this value. This value does not indicate how much of the capacity available for Bluetooth operation is currently available. This value shall only be used if the AMP Controller is powered up.
0x04	The AMP Controller has medium capacity available for Bluetooth operation An AMP Controller that with capacity in the approximate range of 30% < capacity < 70% should indicate this value. This value does not indicate how much of the capacity available for Bluetooth operation is currently available. This value shall only be used if the AMP Controller is powered up
0x05	The AMP Controller has high capacity available for Bluetooth operation. This value indicates that the majority of the AMP Controllers bandwidth is currently allocated to servicing the Bluetooth technology. An AMP Controller that with capacity in the approximate range of 70% < capacity < 100% should indicate this value. This value does not indicate how much of the capacity available for Bluetooth operation is currently available. This value shall only be used if the AMP Controller is powered up
0x06	The AMP Controller has full capacity available for Bluetooth operation. This value indicates that while currently the AMP is only being used by Bluetooth the device allows a different technology to share the radio. This value shall be used by devices that are not capable of determining the current available capacity of an AMP that is shared by a different technology. This value does not indicate how much of the capacity available for Bluetooth operation is currently available. This value shall only be used if the AMP Controller is powered up.
0x07 - 0xFF	Reserved

7.7.62 AMP Start Test Event

Event	Event Code	Event Parameters
AMP Start Test	0x49	Status, Test Scenario

Description:

The AMP Start Test event shall be generated when the AMP_Test command has completed and the first data is ready to be sent or received .

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Test command succeeded
0x01-0xFF	Test command failed. See Part D, Error Codes on page 339

Test Scenario: *Size: 1 Octet*

Value	Parameter Description
0xXX	See the Test Commands section of the Protocol Adaptation Layer Functional Specification for the Controller type

7.7.63 AMP Test End Event

Event	Event Code	Event Parameters
AMP Test End	0x4A	Status, Test Scenario

Description:

The AMP Test End event shall be generated to indicate that the AMP has transmitted or received the number of frames/bursts configured.

If the Receiver reports are enabled an AMP Receiver Report Event shall be generated.

Event Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	AMP_Test_End command succeeded
0x01-0xFF	AMP_Test_End command failed. See Part D, Error Codes on page 339

Test Scenario: *Size: 1 Octet*

Value	Parameter Description
0xXX	See the Test Commands section of the Protocol Adaptation Layer Functional Specification for the Controller type

7.7.64 AMP Receiver Report Event

Event	Event Code	Event Parameters
AMP Receiver Report	0x4B	Controller_Type, Reason, Event_Type, Number_Of_Frames, Number_Of_Error_Frames, Number_Of_Bits, Number_Of_Error_Bits

Description:

The AMP Receiver Report event shall be sent from the AMP to the tester via the AMP Test Manager at the interval configured by the Enable_AMP_Receiver_Reports command.

The AMP Receiver Report event is generated to indicate the number of frames received and optionally the number of bits in error detected.

Event Parameters:

Controller_Type: *Size: 1 Octet*

Value	Parameter Description
0xXX	See Bluetooth Assigned Numbers

Reason: *Size: 1 Octet*

Value	Parameter Description
0xXX	0x00 Configured Interval report 0x01 Test Ended report 0x02 – 0xFF Reserved

Event_type: *Size: 4 Bytes*

Value	Parameter Description
0xXX	0x00 Frames received report 0x01 Frames received and bits in error report (optional) 0x02 to 0xFF Reserved

Number_Of_Frames: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	Number of frames received so far

Number_Of_Error_Frames: *Size: 2 Bytes*

Value	Parameter Description
0xXXXX	Number of frames with errors received so far

Number_Of_Bits: *Size: 4 Bytes*

Value	Parameter Description
0xXXXXXXXX	Number of bits received so far. Shall be set to 0x00000000 if the Event_type is not equal to 0x01.

Number_Of_Error_Bits: *Size: 4 Bytes*

Value	Parameter Description
0xXXXXXXXX	Number of bit errors received so far. Shall be set to 0x00000000 if the Event_type is not equal to 0x01.

7.7.65 LE Meta Event

Description:

The LE Meta Event is used to encapsulate all LE Controller specific events. The Event Code of all LE Meta Events shall be 0x3E. The Subevent_Code is the first octet of the event parameters. The Subevent_Code shall be set to one of the valid Subevent_Codes from an LE specific event. All other Subevent_Parameters are defined in the LE Controller specific events.

7.7.65.1 LE Connection Complete Event

Event	Event Code	Event parameters
LE Connection Complete	0x3E	Subevent_Code, Status, Connection_Handle, Role, Peer_Address_Type, Peer_Address, Conn_Interval, Conn_Latency, Supervision_Timeout, Master_Clock_Accuracy

Description:

The LE Connection Complete event indicates to both of the Hosts forming the connection that a new connection has been created. Upon the creation of the connection a Connection_Handle shall be assigned by the Controller, and passed to the Host in this event. If the connection establishment fails this event shall be provided to the Host that had issued the LE_Create_Connection command.

This event indicates to the Host which issued a LE_Create_Connection command and received a Command Status event if the connection establishment failed or was successful.

The Master_Clock_Accuracy parameter is only valid for a slave. On a master, this parameter shall be set to 0x00.

Event Parameters:

Subevent_Code: *Size: 1 Octet*

Value	Parameter Description
0x01	Subevent code for LE Connection Complete event

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Connection successfully completed.
0x01 – 0xFF	Connection failed to complete. Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection between two Bluetooth devices. The Connection_Handle is used as an identifier for transmitting and receiving data. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Role: *Size: 1 Octet*

Value	Parameter Description
0x00	Connection is master
0x01	Connection is slave
0x02-0xFF	Reserved for future use

Peer_Address_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Peer is using a Public Device Address
0x01	Peer is using a Random Device Address
0x02-0xFF	Reserved for future use

Peer_Address: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	Public Device Address or Random Device Address of the peer device

Conn_Interval: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Connection interval used on this connection. Range: 0x0006 to 0x0C80 Time = N * 1.25 msec Time Range: 7.5 msec to 4000 msec.

Conn_Latency: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Connection latency for this connection. Range: 0x0006 to 0x0C80 Time = N * 1.25 msec Time Range: 7.5 msec to 4000 msec.

Supervision_Timeout: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Connection supervision timeout. Range: 0x000A to 0x0C80 Time = N * 10 msec Time Range: 100 msec to 32 seconds

Master_Clock_Accuracy: *Size: 1 Octet*

Value	Parameter Description
0x00	500 ppm
0x01	250 ppm
0x02	150 ppm
0x03	100 ppm
0x04	75 ppm
0x05	50 ppm
0x06	30 ppm
0x07	20 ppm
0x08-0xFF	Reserved for future use

7.7.65.2 LE Advertising Report Event

Event	Event Code	Event parameters
LE Advertising Report	0x3E	Subevent_Code, Num_Reports, Event_Type[i], Address_Type[i], Address[i], Length[i], Data[i], RSSI[i]

Description:

The LE Advertising Report event indicates that a Bluetooth device or multiple Bluetooth devices have responded to an active scan or received some information during a passive scan. The Controller may queue these advertising reports and send information from multiple devices in one LE Advertising Report event.

Event Parameters:

Subevent_Code: *Size: 1 Octet*

Value	Parameter Description
0x02	Subevent code for LE Advertising Report event

Num_Reports: *Size: 1 Octet*

Value	Parameter Description
0x01 - 0x19	Number of responses in event.

Event_Type[i]: *Size: 1 Octet * Num_Reports*

Value	Parameter Description
0x00	Connectable undirected advertising (ADV_IND).
0x01	Connectable directed advertising (ADV_DIRECT_IND)
0x02	Scannable undirected advertising (ADV_SCAN_IND)
0x03	Non connectable undirected advertising (ADV_NONCONN_IND)
0x04	Scan Response (SCAN_RSP)
0x05-0xFF	Reserved for future use

Address_Type[i]: *Size: 1 Octet * Num_Reports*

Value	Parameter Description
0x00	Public Device Address
0x01	Random Device Address
0x02-0xFF	Reserved for future use

Address[i]: *Size: 6 Octets * Num_Reports*

Value	Parameter Description
0xXXXXXXXXXXXX	Public Device Address or Random Device Address for each device which responded.

Length_Data[i]: *Size: 1 Octets * Num_Reports*

Value	Parameter Description
0x00 - 0x1F	Length of the Data[i] field for each device which responded.
0x20 - 0xFF	Reserved for future use.

Data[i]: *Size: Length_Data[i] Octets * Num_Reports*

Value	Parameter Description
	Length_Data[i] octets of advertising or scan response data formatted as defined in [Vol 3] Part C, Section 8.

RSSI[i]: *Size: 1 Octet * Num_Reports*

Value	Parameter Description
N	Size: 1 Octet (signed integer) Range: -127 ≤ N ≤ +20 Units: dBm
127	RSSI is not available
21 to 126	Reserved for future use

7.7.65.3 LE Connection Update Complete Event

Event	Event Code	Event parameters
LE Connection Update Complete	0x3E	Subevent_Code, Status, Connection_Handle, Conn_Interval, Conn_Latency, Supervision_Timeout

Description:

The LE Connection Update Complete event is used to indicate that the Controller process to update the connection has completed.

On a slave, if no connection parameters are updated, then this event shall not be issued.

On a master, this event shall be issued if the Connection_Update command was sent.

Note: This event can be issued autonomously by the Master's Controller if it decides to change the connection interval based on the range of allowable connection intervals for that connection.

Event Parameters:

Subevent_Code: *Size: 1 Octet*

Value	Parameter Description
0x03	Subevent code for LE Connection Update Complete event

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Connection_Update command successfully completed.
0x01 – 0xFF	Connection_Update command failed to complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection between two Bluetooth devices. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Conn_Interval: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Connection interval used on this connection. Range: 0x0006 to 0x0C80 Time = N * 1.25 msec Time Range: 7.5 msec to 4000 msec.

Conn_Latency: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Connection latency for this connection. Range: 0x0006 to 0x0C80 Time = N * 1.25 msec Time Range: 7.5 msec to 4000 msec.

Supervision_Timeout: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Supervision timeout for this connection. Range: 0x0006 to 0x000A Time = N * 10 msec Time Range: 100 msec to 32 msec.

7.7.65.4 LE Read Remote Used Features Complete Event

Event	Event Code	Event parameters
LE Read Remote Used Features Complete	0x3E	Subevent_Code, Status, Connection_Handle, LE_Features

Description:

The LE Read Remote Used Features Complete event is used to indicate the completion of the process of the Controller obtaining the used features of the remote Bluetooth device specified by the Connection_Handle event parameter.

Event Parameters:

Subevent_Code: *Size: 1 Octet*

Value	Parameter Description
0x04	Subevent code for LE Read Remote Used Features Complete event

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Read_Remote_Used_Features command successfully completed.
0x01 – 0xFF	LE_Read_Remote_Used_Features command failed to complete. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection between two Bluetooth devices. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

LE_Features: *Size: 8 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXX	Bit Mask List of used LE features. For details see LE Link Layer specification.

7.7.65.5 LE Long Term Key Request Event

Event	Event Code	Event parameters
LE Long Term Key Request	0x3E	Subevent_Code, Connection_Handle, Random_Number, Encryption_Diversifier

Description:

The LE Long Term Key Request event indicates that the master device is attempting to encrypt or re-encrypt the link and is requesting the Long Term Key from the Host. (See [Vol 6] Part B, Section 5.1.3).

Event Parameters:

Subevent_Code: *Size: 1 Octet*

Value	Parameter Description
0x05	Subevent code for LE Long Term Key Request event

Connection Handle: *Size: 2 Octets (12 bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection between two Bluetooth devices. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Random_Number: *Size: 8 Octets*

Encrypted_Diversifier: *Size: 2 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXX	64 bit random number.

Value	Parameter Description
0xXXXX	16 bit encrypted diversifier.

7.8 LE CONTROLLER COMMANDS

The LE Controller Commands provide access and control to various capabilities of the Bluetooth hardware, as well as methods for the Host to affect how the Link Layer manages the piconet, and controls connections.

For the LE Controller Commands, the OGF code is defined as 0x08.

7.8.1 LE Set Event Mask Command

Command	OCF	Command Parameters	Return Parameters
HCI_LE_Set_Event_Mask	0x0001	LE_Event_Mask	Status

Description:

The LE_Set_Event_Mask command is used to control which LE events are generated by the HCI for the Host. If the bit in the LE_Event_Mask is set to a one, then the event associated with that bit will be enabled. The Host has to deal with each event that is generated by an LE Controller. The event mask allows the Host to control which events will interrupt it.

For LE events to be generated, the LE Meta-Event bit in the Event_Mask shall also be set. If that bit is not set, then LE events not shall be generated, regardless of how the LE_Event_Mask is set.

Command Parameters:

LE_Event_Mask: *Size: 8 Octets*

Value	LE Subevent Types
0x0000000000000000	No LE events specified
0x0000000000000001	LE Connection Complete Event
0x0000000000000002	LE Advertising Report Event
0x0000000000000004	LE Connection Update Complete Event
0x0000000000000008	LE Read Remote Used Features Complete Event

Value	LE Subevent Types
0x0000000000000010	LE Long Term Key Request Event
0x000000000000001F	Default
0xFFFFFFFFFFFFFE0	Reserved for future use

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Set_Event_Mask command succeeded.
0x01 – 0xFF	LE_Set_Event_Mask command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Set_Event_Mask command has completed, a Command Complete event shall be generated.

7.8.2 LE Read Buffer Size Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Read _Buffer_Size	0x0002		Status, HC_LE_ACL_Data_Packet_Length, HC_Total_Num_LE_ACL_Data_Packets

Description:

The LE_Read_Buffer_Size command is used to read the maximum size of the data portion of HCI LE ACL Data Packets sent from the Host to the Controller. The Host will segment the data transmitted to the Controller according to these values, so that the HCI Data Packets will contain data with up to this size. The LE_Read_Buffer_Size command also returns the total number of HCI LE ACL Data Packets that can be stored in the data buffers of the Controller. The LE_Read_Buffer_Size command must be issued by the Host before it sends any data to an LE Controller (see Section 4.1.1).

If the Controller returns a length value of zero, the Host shall use the Read_Buffer_Size command to determine the size of the data buffers (shared between BR/EDR and LE transports).

Note: Both the Read_Buffer_Size and LE_Read_Buffer_Size commands may return buffer length parameter values that are non-zero. This allows a Controller to offer different sized buffers for BR/EDR data packets and LE data packets.

The HC_LE_ACL_Data_Packet_Length return parameter shall be used to determine the size of the L2CAP PDU segments contained in ACL Data Packets, which are transferred from the Host to the Controller to be broken up into packets by the Link Layer. Both the Host and the Controller shall support command and event packets, where the data portion (excluding header) contained in the packets is 255 octets in size. The HC_Total_Num_LE_ACL_Data_Packets return parameter contains the total number of HCI ACL Data Packets that can be stored in the data buffers of the Controller. The Host determines how the buffers are to be divided between different Connection Handles.

Note: The HC_LE_ACL_Data_Packet_Length return parameter does not include the length of the HCI Data Packet header.

Command Parameters:

None.

Return Parameters:

Status *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Read_Buffer_Size command succeeded.
0x01 – 0xFF	LE_Read_Buffer_Size command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

HC_LE_Data_Packet_Length: *Size: 2 Octets*

Value	Parameter Description
0x0000	No dedicated LE Buffer – use Read_Buffer_Size command.
0x0001 – 0xFFFF	Maximum length (in octets) of the data portion of each HCI ACL Data Packet that the Controller is able to accept.

HC_Total_Num_LE_Data_Packets: *Size: 1 Octet*

Value	Parameter Description
0x00	No dedicated LE Buffer – use Read_Buffer_Size command.
0x01 – 0xFF	Total number of HCI ACL Data Packets that can be stored in the data buffers of the Controller.

Event(s) Generated (unless masked away):

When the LE_Read_Buffer_Size command has completed, a Command Complete event shall be generated.

7.8.3 LE Read Local Supported Features Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Read_Local_Supported_Features	0x0003		Status, LE_Features

Description:

This command requests the list of the supported LE features for the Controller.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Read_Local_Supported_Features command succeeded.
0x01 – 0xFF	LE_Read_Local_Supported_Features command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

LE_Features: *Size: 8 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXX	Bit Mask List of LE features. See [Vol 6] Part B, Section 4.6

Event(s) Generated (unless masked away):

When the LE_Read_Local_Supported_Features command has completed, a Command Complete event shall be generated.

7.8.4 LE Set Random Address Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Set_Random_Address	0x0005	Random_Address	Status

Description:

The LE_Set_Random_Address command is used by the Host to set the LE Random Device Address in the Controller (see [Vol 6] Part B, Section 1.3).

Command Parameters:

Random_Addess: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	Random Device Address as defined by [Vol 6] Part B, Section 1.3.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Set_Random_Address command succeeded.
0x01 – 0xFF	LE_Set_Random_Address command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Set_Random_Address command has completed, a Command Complete event shall be generated.

7.8.5 LE Set Advertising Parameters Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Set_ Advertising_ Parameters	0x0006	Advertising_Interval_Min, Advertising_Interval_Max, Advertising_Type, Own_Address_Type, Direct_Address_Type, Direct_Address, Advertising_Channel_Map, Advertising_Filter_Policy	Status

Description:

The LE_Set_Advertising_Parameters command is used by the Host to set the advertising parameters.

The Advertising_Interval_Min shall be less than or equal to the Advertising_Interval_Max. The Advertising_Interval_Min and Advertising_Interval_Max should not be the same value to enable the Controller to determine the best advertising interval given other activities.

For directed advertising, when Advertising_Type is 0x01 (ADV_DIRECT_IND), the Advertising_Interval_Min and Advertising_Interval_Max parameters are not used and shall be ignored.

The Advertising_Type is used to determine the packet type that is used for advertising when advertising is enabled.

The Advertising_Interval_Min and Advertising_Interval_Max shall not be set to less than 0x00A0 (100 ms) if the Advertising_Type is set to 0x02 (ADV_SCAN_IND) or 0x03 (ADV_NONCONN_IND). The Own_Address_Type determines if the advertising packets are identified with the Public Device Address of the device, or a Random Device Address as written by the LE_Set_Random_Address command.

If directed advertising is performed, then the Direct_Address_Type and Direct_Address shall be valid, otherwise they shall be ignored by the Controller and not used.

The Advertising_Channel_Map is a bit field that indicates the advertising channels that shall be used when transmitting advertising packets. At least one channel bit shall be set in the Advertising_Channel_Map parameter.

The Advertising_Filter_Policy parameter shall be ignored when directed advertising is enabled.

The Host shall not issue this command when advertising is enabled in the Controller; if it is the Command Disallowed error code shall be used.

Command Parameters:

Advertising_Interval_Min *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Minimum advertising interval for non-directed advertising. Range: 0x0020 to 0x4000 Default: N = 0x0800 (1.28 second) Time = N * 0.625 msec Time Range: 20 ms to 10.24 sec

Advertising_Interval_Max: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Maximum advertising interval for non-directed advertising. Range: 0x0020 to 0x4000 Default: N = 0x0800 (1.28 seconds) Time = N * 0.625 msec Time Range: 20 ms to 10.24 sec

Advertising_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Connectable undirected advertising (ADV_IND) (**default**)
0x01	Connectable directed advertising (ADV_DIRECT_IND)
0x02	Scannable undirected advertising (ADV_SCAN_IND)
0x03	Non connectable undirected advertising (ADV_NONCONN_IND)
0x04 – 0xFF	Reserved for future use

Own_Address_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Public Device Address (**default**)
0x01	Random Device Address
0x02 – 0xFF	Reserved for future use

Direct_Address_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Public Device Address (**default**)
0x01	Random Device Address

Value	Parameter Description
0x02 – 0xFF	Reserved for future use

Direct_Address: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	Public Device Address or Random Device Address of the device to be connected

Advertising_Channel_Map: *Size: 1 Octet*

Value	Parameter Description
00000000b	Reserved for future use
xxxxxxx1b	Enable channel 37 use
xxxxxx1xb	Enable channel 38 use
xxxxx1xxb	Enable channel 39 use
00000111b	Default (all channels enabled)

Advertising_Filter_Policy: *Size: 1 Octet*

Value	Parameter Description
0x00	Allow Scan Request from Any, Allow Connect Request from Any (**default**).
0x01	Allow Scan Request from White List Only, Allow Connect Request from Any.
0x02	Allow Scan Request from Any, Allow Connect Request from White List Only.
0x03	Allow Scan Request from White List Only, Allow Connect Request from White List Only.
0x04 – 0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Set_Advertising_Parameters command succeeded.
0x01 – 0xFF	LE_Set_Advertising_Parameters command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Set_Advertising_Parameters command has completed, a Command Complete event shall be generated.

7.8.6 LE Read Advertising Channel Tx Power Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Read_Advertising_Channel_Tx_Power	0x0007		Status, Transmit_Power_Level

Description:

The LE_Read_Advertising_Channel_Tx_Power command is used by the Host to read the transmit power level used for LE advertising channel packets.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Read_Advertising_Channel_Tx_Power command succeeded.
0x01 – 0xFF	LE_Read_Advertising_Channel_Tx_Power failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Transmit_Power_Level: *Size: 1 Octet*

Value	Parameter Description
N = 0xXX	Size: 1 Octet (signed integer) Range: -20 ≤ N ≤ 10 Units: dBm Accuracy: +/- 4 dBm

Event(s) Generated (unless masked away):

When the LE_Read_Advertising_Channel_Tx_Power command has completed, a Command Complete event shall be generated.

7.8.7 LE Set Advertising Data Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Set_Advertising_Data	0x0008	Advertising_Data_Length, Advertising_Data	Status

Description:

The LE_Set_Advertising_Data command is used to set the data used in advertising packets that have a data field.

Only the significant part of the Advertising_Data is transmitted in the advertising packets, as defined in [Vol 3] Part C, Section 11.

Command Parameters:

Advertising_Data_Length: *Size: 1 Octet*

Value	Parameter Description
0x00 – 0x1F	The number of significant octets in the Advertising_Data.

Advertising_Data: *Size: 31 Octets*

Value	Parameter Description
	31 octets of advertising data formatted as defined in [Vol 3] Part C, Section 11.
	All octets zero (**default**).

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Set_Advertising_Data command succeeded.
0x01 – 0xFF	LE_Set_Advertising_Data command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Set_Advertising_Data command has completed, a Command Complete event shall be generated.

7.8.8 LE Set Scan Response Data Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Set_Scan_Response_Data	0x0009	Scan_Response_Data_Length, Scan_Response_Data	Status

Description:

This command is used to provide data used in Scanning Packets that have a data field.

Only the significant part of the Scan_Response_Data is transmitted in the Scanning Packets, as defined in [Vol 3] Part C, Section 11.

Command Parameters:

Scan_Response_Data_Length: *Size: 1 Octet*

Value	Parameter Description
0x00 – 0x1F	The number of significant octets in the Scan_Response_Data.

Scan_Response_Data: *Size: 31 Octets*

Value	Parameter Description
	31 octets of Scan_Response_Data formatted as defined in [Vol 3] Part C, Section 11.
	All octets zero (**default**).

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Set_Scan_Response_Data command succeeded.
0x01 – 0xFF	LE_Set_Scan_Response_Data command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Scan_Response_Data command has completed, a Command Complete event shall be generated.

7.8.9 LE Set Advertise Enable Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Set_ Advertise_Enable	0x000A	Advertising_Enable	Status

Description:

The LE_Set_Advertise_Enable command is used to request the Controller to start or stop advertising. The Controller manages the timing of advertisements as per the advertising parameters given in the LE_Set_Advertising_Parameters command.

The Controller shall continue advertising until the Host issues an LE_Set_Advertise_Enable command with Advertising_Enable set to 0x00 (Advertising is disabled) or until a connection is created or until the Advertising is timed out due to Directed Advertising. In these cases, advertising is then disabled.

Command Parameters:

Advertising_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	Advertising is disabled (**default**)
0x01	Advertising is enabled.
0x02 – 0xFF	Reserved for future use

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Set_Advertise_Enable command succeeded.
0x01 – 0xFF	LE_Set_Advertise_Enable command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Set_Advertise_Enable command has completed, a Command Complete event shall be generated.

If the Advertising_Type parameter is 0x01 (ADV_DIRECT_IND) and the directed advertising fails to create a connection, an LE Connection Complete event shall be generated with the Status code set to Directed Advertising Timeout (0x3C).

If the Advertising_Type parameter is 0x00 (ADV_IND) or 0x01 (ADV_DIRECT_IND) and a connection is established, an LE Connection Complete event shall be generated.

Note: There is a possible race condition if the Advertising_Enable parameter is set to 0x00 (Disable) and the Advertising_Type parameter is 0x00 or 0x01. The advertisements might not be stopped before a connection is created, and therefore both the Command Complete event and an LE Connection Complete event could be generated. This can also occur when directed advertising is timed out and this command disables advertising.

7.8.10 LE Set Scan Parameters Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Set_Scan_Parameters	0x000B	LE_Scan_Type, LE_Scan_Interval, LE_Scan_Window, Own_Address_Type, Scanning_Filter_Policy	Status

Description:

The LE_Set_Scan_Parameters command is used to set the scan parameters.

The LE_Scan_Type parameter controls the type of scan to perform.

The LE_Scan_Interval and LE_Scan_Window parameters are recommendations from the Host on how long (LE_Scan_Window) and how frequently (LE_Scan_Interval) the Controller should scan (See [Vol 6] Part B, Section 4.5.3). The LE_Scan_Window parameter shall always be set to a value smaller or equal to the value set for the LE_Scan_Interval parameter. If they are set to the same value scanning should be run continuously.

The Own_Address_Type parameter determines the address used (Public or Random Device Address) when performing active scan.

The Host shall not issue this command when scanning is enabled in the Controller; if it is the Command Disallowed error code shall be used.

Command Parameters:

LE_Scan_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Passive Scanning. No SCAN_REQ packets shall be sent. (**default**)
0x01	Active scanning. SCAN_REQ packets may be sent.
0x02 – 0xFF	Reserved for future use

LE_Scan_Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	This is defined as the time interval from when the Controller started its last LE scan until it begins the subsequent LE scan. Range: 0x0004 to 0x4000 Default: 0x0010 (10 ms) Time = N * 0.625 msec Time Range: 2.5 msec to 10.24 seconds

LE_Scan_Window: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	The duration of the LE scan. LE_Scan_Window shall be less than or equal to LE_Scan_Interval Range: 0x0004 to 0x4000 Default: 0x0010 (10 ms) Time = N * 0.625 msec Time Range: 2.5 msec to 10240 msec

Own_Address_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Public Device Address (**default**)
0x01	Random Device Address
0x02-0xFF	Reserved for future use.

Scanning_Filter_Policy: *Size: 1 Octet*

Value	Parameter Description
0x00	Accept all advertisement packets (**default**). Directed advertising packets which are not addressed for this device shall be ignored.
0x01	Ignore advertisement packets from devices not in the White List Only. Directed advertising packets which are not addressed for this device shall be ignored.
0x02 – 0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Scan_Activity command succeeded.

Value	Parameter Description
0x01 – 0xFF	LE_Scan_Activity command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Set_Scan_Parameters command has completed, a Command Complete event shall be generated.

7.8.11 LE Set Scan Enable Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Set_Scan_Enable	0x000C	LE_Scan_Enable Filter_Duplicates	Status

Description:

The LE_Set_Scan_Enable command is used to start scanning. Scanning is used to discover advertising devices nearby.

The Filter_Duplicates parameter controls whether the Link Layer shall filter duplicate advertising reports to the Host, or if it shall generate advertising reports for each packet received.

Command Parameters:

LE_Scan_Enable: *Size: 1 Octet*

Value	Parameter Description
0x00	Scanning disabled.
0x01	Scanning enabled.
0x02 – 0xFF	Reserved for future use.

Filter_Duplicates: *Size: 1 Octet*

Value	Parameter Description
0x00	Duplicate filtering disabled.
0x01	Duplicate filtering enabled.
0x02 – 0xFF	Reserved for future use.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Set_Scan_Enable command succeeded.
0x01 – 0xFF	LE_Set_Scan_Enable command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Set_Scan_Enable command has completed, a Command Complete event shall be generated.

Zero or more LE Advertising Reports are generated by the Controller based on advertising packets received and the duplicate filtering. More than one advertising packet may be reported in each LE Advertising Report event.

7.8.12 LE Create Connection Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Create_Connection	0x000D	LE_Scan_Interval, LE_Scan_Window, Initiator_Filter_Policy, Peer_Address_Type, Peer_Address, Own_Address_Type, Conn_Interval_Min, Conn_Interval_Max, Conn_Latency, Supervision_Timeout, Minimum_CE_Length, Maximum_CE_Length	

Description:

The LE_Create_Connection command is used to create a Link Layer connection to a connectable advertiser.

The LE_Scan_Interval and LE_Scan_Window parameters are recommendations from the Host on how long (LE_Scan_Window) and how frequently (LE_Scan_Interval) the Controller should scan. The LE_Scan_Window parameter shall be set to a value smaller or equal to the value set for the LE_Scan_Interval parameter. If both are set to the same value, scanning should run continuously.

The Initiator_Filter_Policy is used to determine whether the White List is used. If the White List is not used, the Peer_Address_Type and the Peer_Address parameters specify the address type and address of the advertising device to connect to.

The Link Layer shall set the address in the CONNECT_REQ packets to either the Public Device Address or the Random Device Addressed based on the Own_Address_Type parameter.

The Conn_Interval_Min and Conn_Interval_Max parameters define the minimum and maximum allowed connection interval. The Conn_Interval_Min parameter shall not be greater than the Conn_Interval_Max parameter.

The Conn_Latency parameter defines the maximum allowed connection latency (see [Vol 6] Part B, Section 4.5.1).

The Supervision_Timeout parameter defines the link supervision timeout for the connection. The Supervision_Timeout in milliseconds shall be larger than the Conn_Interval_Max in milliseconds. (See [Vol 6] Part B, Section 4.5.2).

The Minimum_CE_Length and Maximum_CE_Length parameters are informative parameters providing the Controller with the expected minimum and maximum length of the connection events. The Minimum_CE_Length parameter shall be less than or equal to the Maximum_CE_Length parameter.

The Host shall not issue this command when another LE_Create_Connection is pending in the Controller; if this does occur the Controller shall return the Command Disallowed error code shall be used.

Command Parameters:

LE_Scan_Interval: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	This is defined as the time interval from when the Controller started its last LE scan until it begins the subsequent LE scan. Range: 0x0004 to 0x4000 Time = N * 0.625 msec Time Range: 2.5 msec to 10240 msec

LE_Scan_Window: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Amount of time for the duration of the LE scan. LE_Scan_Window shall be less than or equal to LE_Scan_Interval Range: 0x0004 to 0x4000 Time = N * 0.625 msec Time Range: 2.5 msec to 10.24 seconds

Initiator_Filter_Policy: *Size: 1 Octet*

Value	Parameter Description
0x00	White list is not used to determine which advertiser to connect to. Peer_Address_Type and Peer_Address shall be used.
0x01	White list is used to determine which advertiser to connect to. Peer_Address_Type and Peer_Address shall be ignored.
0x02 – 0xFF	Reserved for future use.

Peer_Address_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Public Device Address
0x01	Random Device Address
0x02 – 0xFF	Reserved for future use

Peer_Address: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	Public Device Address or Random Device Address of the device to be connected

Own_Address_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Public Device Address
0x01	Random Device Address
0x02 – 0xFF	Reserved for future use

Conn_Interval_Min: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Minimum value for the connection event interval. This shall be less than or equal to Conn_Interval_Max. Range: 0x0006 to 0x0C80 Time = N * 1.25 msec Time Range: 7.5 msec to 4 seconds.
0x0000 – 0x0005 and 0x0C81 – 0xFFFF	Reserved for future use

Conn_Interval_Max: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Maximum value for the connection event interval. This shall be greater than or equal to Conn_Interval_Min. Range: 0x0006 to 0x0C80 Time = N * 1.25 msec Time Range: 7.5 msec to 4 seconds.
0x0000 – 0x0005 and 0x0C81 – 0xFFFF	Reserved for future use

Conn_Latency: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Slave latency for the connection in number of connection events. Range: 0x0000 to 0x01F4

Supervision_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Supervision timeout for the LE Link. (See [Vol 6] Part B, Section 4.5.2) Range: 0x000A to 0x0C80 Time = N * 10 msec Time Range: 100 msec to 32 seconds
0x0000 - 0x0009 and 0x0C81 - 0xFFFF	Reserved for future use

Minimum_CE_Length: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Information parameter about the minimum length of connection needed for this LE connection. Range: 0x0000 – 0xFFFF Time = N * 0.625 msec.

Maximum_CE_Length: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Information parameter about the maximum length of connection needed for this LE connection. Range: 0x0000 – 0xFFFF Time = N * 0.625 msec.

Return Parameters:

None.

Event(s) Generated (unless masked away):

When the Controller receives the LE_Create_Connection command, the Controller sends the Command Status event to the Host. An LE Connection Complete event shall be generated when a connection is created or the connection creation procedure is cancelled.

Note: No Command Complete event is sent by the Controller to indicate that this command has been completed. Instead, the LE Connection Complete event indicates that this command has been completed.

7.8.13 LE Create Connection Cancel Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Create _Connection_Cancel	0x000E		Status

Description:

The LE_Create_Connection_Cancel command is used to cancel the LE_Create_Connection command. This command shall only be issued after the LE_Create_Connection command has been issued, a Command Status event has been received for the LE Create Connection command and before the LE Connection Complete event.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Create_Connection_Cancel command succeeded.
0x01 – 0xFF	LE_Create_Connection_Cancel command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Create_Connection_Cancel command has completed, a Command Complete event shall be generated.

If the LE_Create_Connection_Cancel command is sent to the Controller without a preceding LE_Create_Connection command, the Controller shall return a Command Complete event with the error code Command Disallowed (0x0C).

The LE Connection Complete event with the error code Unknown Connection Identifier (0x02) shall be sent after the Command Complete event for the LE_Create_Connection_Cancel command if the cancellation was successful.

7.8.14 LE Read White List Size Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Read_ White_List_Size	0x000F		Status, White_List_Size

Description:

The LE_Read_White_List_Size command is used to read the total number of white list entries that can be stored in the Controller.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Read_White_List_Size command succeeded.
0x01 – 0xFF	LE_Read_White_List_Size command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

White_List_Size: *Size: 1 Octet*

Value	Parameter Description
0x01 – 0xFF	Total number of white list entries that can be stored in the Controller.
0x00	Reserved for future use

Event(s) Generated (unless masked away):

When the LE_Read_White_List_Size command has completed, a Command Complete event shall be generated.

7.8.15 LE Clear White List Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Clear_ White_List	0x0010		Status

Description:

The LE_Clear_White_List command is used to clear the white list stored in the Controller.

This command can be used at any time except when:

- the advertising filter policy uses the white list and advertising is enabled.
- the scanning filter policy uses the white list and scanning is enabled.
- the initiator filter policy uses the white list and an LE_Create_Connection command is outstanding.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Clear_White_List command succeeded.
0x01 – 0xFF	LE_Clear_White_List command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE Clear White List command has completed, a Command Complete event shall be generated.

7.8.16 LE Add Device To White List Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Add _Device_To_White_ List	0x0011	Address_Type, Address	Status

Description:

The LE_Add_Device_To_White_List command is used to add a single device to the white list stored in the Controller.

This command can be used at any time except when:

- the advertising filter policy uses the white list and advertising is enabled.
- the scanning filter policy uses the white list and scanning is enabled.
- the initiator filter policy uses the white list and a create connection command is outstanding.

Command Parameters:

Address_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Public Device Address
0x01	Random Device Address
0x02-0xFF	Reserved for future use.

Address: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	Public Device Address or Random Device Address of the device to be added to the white list.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Add_Device_To_White_List command succeeded.
0x01 – 0xFF	LE_Add_Device_To_White_List command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away)::

When the LE_Add_Device_To_White_List command has completed, a Command Complete event shall be generated.

7.8.17 LE Remove Device From White List Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Remove_Device_From_White_List	0x0012	Address_Type, Address	Status

Description:

The LE_Remove_Device_From_White_List command is used to remove a single device from the white list stored in the Controller.

This command can be used at any time except when:

- the advertising filter policy uses the white list and advertising is enabled.
- the scanning filter policy uses the white list and scanning is enabled.
- the initiator filter policy uses the white list and a create connection command is outstanding.

Command Parameters:

Address_Type: *Size: 1 Octet*

Value	Parameter Description
0x00	Public Device Address
0x01	Random Device Address
0x02-0xFF	Reserved for future use.

Address: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	Public Device Address or Random Device Address of the device to be removed from the white list.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Remove_Device_From_White_List command succeeded.
0x01 – 0xFF	LE_Remove_Device_From_White_List command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Remove_Device_From_White_List command has completed, a Command Complete event shall be generated.

7.8.18 LE Connection Update Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Connection_Update	0x0013	Connection_Handle, Conn_Interval_Min, Conn_Interval_Max, Conn_Latency, Supervision_Timeout, Minimum_CE_Length, Maximum_CE_Length	

Description:

The LE_Connection_Update command is used to change the Link Layer connection parameters of a connection. This command shall only be used when the local device's role is Master.

The Conn_Interval_Min and Conn_Interval_Max parameters are used to define the minimum and maximum allowed connection interval. The Conn_Interval_Min parameter shall not be greater than the Conn_Interval_Max parameter.

The Conn_Latency parameter shall define the maximum allowed connection latency.

The Supervision_Timeout parameter shall define the link supervision timeout for the LE link. The Supervision_Timeout shall be larger than the Conn_Interval_Max.

The Minimum_CE_Length and Maximum_CE_Length are information parameters providing the Controller with a hint about the expected minimum and maximum length of the connection events. The Minimum_CE_Length shall be less than or equal to the Maximum_CE_Length.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Conn_Interval_Min: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Minimum value for the connection event interval. This shall be less than or equal to Conn_Interval_Max. Range: 0x0006 to 0x0C80 Time = N * 1.25 msec Time Range: 7.5 msec to 4 seconds.
0x0000-0x0005 and 0x0C81-0xFFFF	Reserved for future use

Conn_Interval_Max: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Maximum value for the connection event interval. This shall be greater than or equal to Conn_Interval_Min. Range: 0x0006 to 0x0C80 Time = N * 1.25 msec Time Range: 7.5 msec to 4 seconds.
0x0000-0x0005 and 0x0C81-0xFFFF	Reserved for future use

Conn_Latency: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Slave latency for the connection in number of connection events. Range: 0x0000 to 0x03E8
0x03E9 – 0xFFFF	Reserved for future use.

Supervision_Timeout: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Supervision timeout for the LE Link. Range: 0x000A to 0x0C80 Mandatory Range: 0x000A to 0x0C80 Time = N * 10 msec Time Range: 100 msec to 32 seconds
0x0001-0x0009 and 0x0C81-0xFFFF	Reserved for future use

Minimum_CE_Length: *Size: 2 Octets*

Maximum_CE_Length: *Size: 2 Octets*

Value	Parameter Description
N = 0xXXXX	Information parameter about the minimum length of connection needed for this LE connection. How this value is used is outside the scope of this specification. Range: 0x0000 – 0xFFFF Time = N * 0.625 msec.

Value	Parameter Description
N = 0xXXXX	Information parameter about the maximum length of connection needed for this LE connection. How this value is used is outside the scope of this specification. Range: 0x0000 – 0xFFFF Time = N * 0.625 msec.

Return Parameters:

None.

Event(s) Generated (unless masked away):

When the Controller receives the LE_Connection_Update command, the Controller sends the Command Status event to the Host. The LE Connection Update Complete event shall be generated after the connection parameters have been applied by the Controller.

Note: a Command Complete event is not sent by the Controller to indicate that this command has been completed. Instead, the LE Connection Update Complete event indicates that this command has been completed.

7.8.19 LE Set Host Channel Classification Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Set_Host_Channel_Classification	0x0014	Channel_Map	Status

Description:

The LE_Set_Host_Channel_Classification command allows the Host to specify a channel classification for data channels based on its “local information”. This classification persists until overwritten with a subsequent LE_Set_Host_Channel_Classification command or until the Controller is reset using the Reset command (see [Vol 6] Part B, Section 4.5.8.1).

If this command is used, the Host should send it within 10 seconds of knowing that the channel classification has changed. The interval between two successive commands sent shall be at least one second.

This command shall only be used when the local device supports the Master role.

Command Parameters:

LE_Channel_Map: *Size: 5 Octet (37 Bits meaningful)*

Value	Parameter Description
0xXXXXXXXXXX	This parameter contains 37 1-bit fields. The nth such field (in the range 0 to 36) contains the value for the link layer channel index n. Channel n is bad = 0. Channel n is unknown = 1. The most significant bits are reserved and shall be set to 0. At least one channel shall be marked as unknown.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Set_Host_Channel_Classification command succeeded.
0x01 – 0xFF	LE_Set_Host_Channel_Classification command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Set_Host_Channel_Classification command has completed, a Command Complete event shall be generated.

7.8.20 LE Read Channel Map Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Read_Channel_Map	0x0015	Connection_Handle	Status, Connection_Handle, Channel_Map

Description:

The LE_Read_Channel_Map command returns the current Channel_Map for the specified Connection_Handle. The returned value indicates the state of the Channel_Map specified by the last transmitted or received Channel_Map (in a CONNECT_REQ or LL_CHANNEL_MAP_REQ message) for the specified Connection_Handle, regardless of whether the Master has received an acknowledgement.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	The Connection_Handle for the Connection for which the Channel_Map is to be read. Range 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Read_Channel_Map command succeeded.
0x01 – 0xFF	LE_Read_Channel_Map command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

LE_Channel_Map: *Size: 5 Octets*

Value	Parameter Description
0xXXXXXXXXXX	This parameter contains 37 1-bit fields. The nth such field (in the range 0 to 36) contains the value for the link layer channel index n. Channel n is unused = 0. Channel n is used = 1. The most significant bits are reserved and shall be set to 0.

Event(s) Generated (unless masked away):

When the LE_Read_Channel_Map command has completed, a Command Complete event shall be generated.

7.8.21 LE Read Remote Used Features Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Read_ Remote_Used_ Features	0x0016	Connection_Handle	

Description:

This command requests a list of the used LE features from the remote device. This command shall return a list of the used LE features. For details see [Vol 6] Part B, Section 4.6.

This command shall only be used when the local device's role is Master.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Return Parameters:

None.

Event(s) Generated (unless masked away):

When the Controller receives the LE_Read_Remote_Used_Features command, the Controller shall send the Command Status event to the Host. When the Controller has completed the procedure to determine the remote features, the Controller shall send a LE Read Remote Used Features Complete event to the Host.

The LE Read Remote Used Features Complete event contains the status of this command, and the parameter describing the used features of the remote device.

Note: A Command Complete event is not sent by the Controller to indicate that this command has been completed. Instead, the LE Read Remote Used Features Complete event indicates that this command has been completed.

7.8.22 LE Encrypt Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Encrypt	0x0017	Key, Plaintext_Data	Status, Encrypted_Data

Description:

The LE_Encrypt command is used to request the Controller to encrypt the Plaintext_Data in the command using the Key given in the command and returns the Encrypted_Data to the Host. The AES-128 bit block cypher is defined in NIST Publication FIPS-197 (http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf).

Command Parameters:

Key: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	128 bit key for the encryption of the data given in the command. The most significant octet of the key corresponds to key[0] using the notation specified in FIPS 197.

Plaintext_Data: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	128 bit data block that is requested to be encrypted. The most significant octet of the PlainText_Data corresponds to in[0] using the notation specified in FIPS 197.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Encrypt command succeeded.
0x01 – 0xFF	LE_Encrypt command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Encrypted_Data: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	128 bit encrypted data block. The most significant octet of the Encrypted_Data corresponds to out[0] using the notation specified in FIPS 197.

Event(s) Generated (unless masked away):

When the LE_Encrypt command has completed, a Command Complete event shall be generated.

7.8.23 LE Rand Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Rand	0x0018		Status, Random_Number

Description:

The LE_Rand command is used to request the Controller to generate 8 octets of random data to be sent to the Host. The Random_Number shall be generated according to [Vol 2] Part H, Section 2 if the LE Feature (LL Encryption) is supported.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Rand command succeeded.
0x01 – 0xFF	LE_Rand command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Random_Number: *Size: 8 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXX	Random Number

Event(s) Generated (unless masked away):

When the LE_Rand command has completed, a Command Complete event shall be generated.

7.8.24 LE Start Encryption Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Start_ Encryption	0x0019	Connection_Handle, Random_Number, Encrypted_Diversifier, Long_Term_Key	

Description:

The LE_Start_Encryption command is used to authenticate the given encryption key associated with the remote device specified by the connection handle, and once authenticated will encrypt the connection. The parameters are as defined in [Vol 3] Part H, Section 2.4.4.

If the connection is already encrypted then the Controller shall pause connection encryption before attempting to authenticate the given encryption key, and then re-encrypt the connection. While encryption is paused no user data shall be transmitted.

On an authentication failure, the connection shall be automatically disconnected by the Link Layer. If this command succeeds, then the connection shall be encrypted.

This command shall only be used when the local device's role is Master.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Random_Number: *Size: 8 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXX	64 bit random number.

Encrypted_Diversifier: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	16 bit encrypted diversifier.

Long_Term_Key: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	128 bit long term key.

Return Parameters:

None.

Event(s) Generated (unless masked away):

When the Controller receives the LE_Start_Encryption command it shall send the Command Status event to the Host. If the connection is not encrypted when this command is issued, an Encryption Change event shall occur when encryption has been started for the connection. If the connection is encrypted when this command is issued, an Encryption Key Refresh Complete event shall occur when encryption has been resumed.

Note: A Command Complete event is not sent by the Controller to indicate that this command has been completed. Instead, the Encryption Change or Encryption Key Refresh Complete events indicate that this command has been completed.

7.8.25 LE Long Term Key Request Reply Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Long_ Term_Key _Request_Reply	0x001A	Connection_Handle, Long_Term_Key	Status, Connection_ Handle

Description:

The LE_Long_Term_Key_Request Reply command is used to reply to an LE Long Term Key Request event from the Controller, and specifies the Long_Term_Key parameter that shall be used for this Connection_Handle. The Long_Term_Key is used as defined in [Vol 6] Part B, Section 5.1.3.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Long Term Key: *Size: 16 Octets*

Value	Parameter Description
0xXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	128 bit long term key for the given connection.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Long_Term_Key_Request_Reply command succeeded.
0x01 – 0xFF	LE_Long_Term_Key_Request_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Event(s) Generated (unless masked away):

When the LE_Long_Term_Key_Request_Reply command has completed, a Command Complete event shall be generated.

7.8.26 LE Long Term Key Request Negative Reply Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Long _Term_Key _Requested _Negative_Reply	0x001B	Connection_Handle	Status, Connection_Handle

Description:

The LE_Long_Term_Key_Request_Negative_Reply command is used to reply to an LE Long Term Key Request event from the Controller if the Host cannot provide a Long Term Key for this Connection_Handle.

Command Parameters:

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Long_Term_Key_Request_Negative_Reply command succeeded.
0x01 – 0xFF	LE_Long_Term_Key_Request_Negative_Reply command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Connection_Handle: *Size: 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection_Handle to be used to identify a connection. Range 0x0000-0x0EFF (0x0F00 – 0x0FFF Reserved for future use)

Event(s) Generated (unless masked away):

When the LE_Long_Term_Key_Request_Negative_Reply command has completed, a Command Complete event shall be generated.

7.8.27 LE Read Supported States Command

Command	OCF	Command parameters	Return Parameters
HCI_LE_Read_ Supported_States	0x001C		Status, LE_States

Description:

The LE_Read_Supported_States command reads the states and state combinations that the link layer supports. See [Vol 6] Part B, Section 1.1.1.

LE_States is an 8-octet bit field. If a bit is set to 1 then this state or state combination is supported by the Controller. Multiple bits in LE_States may be set to 1 to indicate support for multiple state and state combinations.

All the Advertising type with the Initiate State combinations shall be set only if the corresponding Advertising types and Master Role combination are set.

All the Scanning types and the Initiate State combinations shall be set only if the corresponding Scanning types and Master Role combination are set.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Read_Supported_States command succeeded.
0x01 – 0xFF	LE_Read_Supported_States command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

LE_States: *Size: 8 Octets*

Value	Parameter Description
0x0000000000000000	Reserved for future use (No state combinations supported).
0x0000000000000001	Non-connectable Advertising State supported.
0x0000000000000002	Scannable Advertising State supported.
0x0000000000000004	Connectable Advertising State supported.

Value	Parameter Description
0x0000000000000008	Directed Advertising State supported.
0x0000000000000010	Passive Scanning State supported.
0x0000000000000020	Active Scanning State supported.
0x0000000000000040	Initiating State supported. Connection State in the Master Role supported is also supported.
0x0000000000000080	Connection State in the Slave Role supported.
0x0000000000000100	Non-connectable Advertising State and Passive Scanning State combination supported.
0x0000000000000200	Scannable Advertising State and Passive Scanning State combination supported.
0x0000000000000400	Connectable Advertising State and Passive Scanning State combination supported.
0x0000000000000800	Directed Advertising State and Passive Scanning State combination supported.
0x0000000000001000	Non-connectable Advertising State and Active Scanning State combination supported.
0x0000000000002000	Scannable Advertising State and Active Scanning State combination supported.
0x0000000000004000	Connectable Advertising State and Active Scanning State combination supported.
0x0000000000008000	Directed Advertising State and Active Scanning State combination supported.
0x0000000000010000	Non-connectable Advertising State and Initiating State combination supported.
0x0000000000020000	Scannable Advertising State and Initiating State combination supported
0x0000000000040000	Non-connectable Advertising State and Master Role combination supported.
0x0000000000080000	Scannable Advertising State and Master Role combination supported.
0x0000000000100000	Non-connectable Advertising State and Slave Role combination supported.
0x0000000000200000	Scannable Advertising State and Slave Role combination supported.
0x0000000000400000	Passive Scanning State and Initiating State combination supported.

Value	Parameter Description
0x0000000000800000	Active Scanning State and Initiating State combination supported.
0x0000000001000000	Passive Scanning State and Master Role combination supported.
0x0000000002000000	Active Scanning State and Master Role combination supported.
0x0000000004000000	Passive Scanning state and Slave Role combination supported.
0x0000000008000000	Active Scanning state and Slave Role combination supported.
0x0000000010000000	Initiating State and Master Role combination supported. Master Role and Master Role combination is also supported.
0xFFFFFFFFE0000000	Reserved for future use

Event(s) Generated (unless masked away):

When the LE_Read_Supported_States command has completed, a Command Complete event will be generated.

7.8.28 LE Receiver Test Command

Command	OCF	Command Parameters	Return Parameters
HCI_LE_Receiver_Test	0x001D	RX_Frequency	Status

Description:

This command is used to start a test where the DUT receives test reference packets at a fixed interval. The tester generates the test reference packets.

Command Parameters:

RX_Frequency *Size: 1 Octet*

Value	Parameter Description
N = 0xXX	N = (F – 2402) / 2
	Range: 0x00 – 0x27. Frequency Range : 2402 MHz to 2480 MHz

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Receiver_Test command succeeded.
0x01 – 0xFF	LE_Receiver_Test command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) Generated (unless masked away):

When the LE_Receiver_Test command has completed, a Command Complete event shall be generated.

7.8.29 LE Transmitter Test Command

Command	OCF	Command Parameters	Return Parameters
HCI_LE_Transmitter_Test	0x001E	TX_Frequency, Length_Of_Test_Data, Packet_Payload	Status

Description:

This command is used to start a test where the DUT generates test reference packets at a fixed interval. The Controller shall transmit at maximum power.

An LE Controller supporting the LE_Transmitter_Test command shall support Packet_Payload values 0x00, 0x01 and 0x02. An LE Controller may support other values of Packet_Payload.

Command Parameters:

TX_Frequency: *Size: 1 Octet*

Value	Parameter Description
N = 0xXX	N = (F – 2402) / 2
	Range: 0x00 – 0x27. Frequency Range : 2402 MHz to 2480 MHz

Length_Of_Test_Data: *Size: 1 Octet*

Value	Parameter Description
0x00-0x25	Length in bytes of payload data in each packet
0x26-0xFF	Reserved for future use

Packet_Payload: *Size: 1 Octet*

Value	Parameter Description
0x00	Pseudo-Random bit sequence 9
0x01	Pattern of alternating bits ‘11110000
0x02	Pattern of alternating bits ‘10101010’
0x03	Pseudo-Random bit sequence 15
0x04	Pattern of All ‘1’ bits
0x05	Pattern of All ‘0’ bits
0x06	Pattern of alternating bits ‘00001111
0x07	Pattern of alternating bits ‘0101’
0x08-0xFF	Reserved for future use

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Transmitter_Test command succeeded.
0x01 – 0xFF	LE_Transmitter_Test command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

When the LE_Transmitter_Test command has completed, a Command Complete event shall be generated.

7.8.30 LE Test End Command

Command	OCF	Command Parameters	Return Parameters
HCI_LE_Test_End	0x001F		Status, Number_Of_Packets

Description:

This command is used to stop any test which is in progress. The Number_Of_Packets for a transmitter test shall be reported as 0x0000. The Number_Of_Packets is an unsigned number and contains the number of received packets.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	LE_Test_End command succeeded.
0x01-0xFF	LE_Test_End command failed See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Number_Of_Packets: *Size: 2 Octets*

Value	Parameter Description
0xXXXX	Number of packets received

Event(s) Generated (unless masked away):

When the LE_Test_End command has completed, a Command Complete event shall be generated.

8 LIST OF FIGURES

Figure 1.1: Overview of the Lower Software Layers 370
Figure 1.2: End to End Overview of Lower Software Layers to Transfer Data 371
Figure 5.1: HCI Command Packet 428
Figure 5.2: HCI ACL Data Packet 429
Figure 5.3: HCI Synchronous Data Packet 431
Figure 5.4: HCI Event Packet 434
Figure 7.1: Local Loopback Mode 706
Figure 7.2: Remote Loopback Mode 707

9 LIST OF TABLES

Table 3.1: Overview of commands and events 373
Table 3.2: Generic events 374
Table 3.3: Device setup 374
Table 3.4: Controller flow control 375
Table 3.5: Controller information 376
Table 3.6: Controller configuration 377
Table 3.7: Device discovery 380
Table 3.8: Connection setup 382
Table 3.9: Remote information 387
Table 3.10: Synchronous connections 389
Table 3.11: Connection state 390
Table 3.12: Piconet structure 392
Table 3.13: Quality of service 393
Table 3.14: Physical links 395
Table 3.15: Host flow control 397
Table 3.16: Link information 399
Table 3.17: Authentication and encryption 400
Table 3.18: Testing 406
Table 3.19: Alphabetical list of commands and events 408
Table 3.20: Bluetooth Controller supporting LE requirements 415

10 APPENDIX A: DEPRECATED COMMANDS, EVENTS AND CONFIGURATION PARAMETERS

Commands, events and configuration parameters in this section were in prior versions of the specification, but have been determined not to be required.

They may be implemented by a controller to allow for backwards compatibility with a host utilizing a prior version of the specification.

A Host should not use these commands.

10.1 Read Page Scan Mode Command 861
10.2 Write Page Scan Mode Command 862
10.3 Read Page Scan Period Mode Command 863
10.4 Write Page Scan Period Mode Command 864
10.5 Add SCO Connection Command 865
10.6 Page Scan Mode Change Event 867
10.7 Read Country Code Command 868
10.8 Read Encryption Mode Command 869
10.9 Write Encryption Mode Command 869
10.10 Deprecated Parameters 870
10.10.1 Encryption Mode 870
10.10.2 Page Scan Mode 872

10.1 READ PAGE SCAN MODE COMMAND

Command	OGF	OCF	Command Parameters	Return Parameters
HCI_Read_Page_Scan_Mode	0x03	0x003D		Status, Page_Scan_Mode

Description:

This command is used to read the default Page Scan Mode configuration parameter of the local BR/EDR Controller. See “Page Scan Mode” on page 871.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Page_Scan_Mode command succeeded.
0x01-0xFF	Read_Page_Scan_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Page_Scan_Mode: *Size: 1 Octet*

Value	Parameter Description
See Appendix A, Page Scan Mode on page 871.	

Event(s) generated (unless masked away):

When the Read_Page_Scan_Mode command has completed, a Command Complete event will be generated.

10.2 WRITE PAGE SCAN MODE COMMAND

OGF: 0x03 (Controller and baseband commands)

Command	OGF	OCF	Command Parameters	Return Parameters
HCI_Write_Page_Scan_Mode	0x03	0x003E	Page_Scan_Mode	Status

Description:

This command is used to write the default Page Scan Mode configuration parameter of the local BR/EDR Controller. See Page Scan Mode on page 871

Command Parameters:

Page_Scan_Mode: *Size: 1 Octet*

Value	Parameter Description
See Appendix A, See “Page Scan Mode” on page 871.	

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Page_Scan_Mode command succeeded.
0x01-0xFF	Write_Page_Scan_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Page_Scan_Mode command has completed, a Command Complete event will be generated.

10.3 READ PAGE SCAN PERIOD MODE COMMAND

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Page_Scan_Period_Mode	0x003B		Status, Page_Scan_Period_Mode

Description:

This command is used to read the mandatory Page_Scan_Period_Mode configuration parameter of the local BR/EDR Controller. See Section 6.10 on page 438.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Page_Scan_Period_Mode command succeeded.
0x01-0xFF	Read_Page_Scan_Period_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Page_Scan_Period_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	P0
0x01	P1
0x02	P2
0x03-0xFF	Reserved.

Event(s) generated (unless masked away):

When the Read_Page_Scan_Period_Mode command has completed, a Command Complete event will be generated.

10.4 WRITE PAGE SCAN PERIOD MODE COMMAND

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Page_Scan_Period_Mode	0x003C	Page_Scan_Period_Mode	Status

Description:

This command is used to write the mandatory Page_Scan_Period_Mode configuration parameter of the local BR/EDR Controller. See Section 6.10 on page 438.

Command Parameters:

Page_Scan_Period_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	P0
0x01	P1
0x02	P2. **Default.**
0x03-0xFF	Reserved.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Page_Scan_Period_Mode command succeeded.
0x01-0xFF	Write_Page_Scan_Period_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Page_Scan_Period_Mode command has completed, a Command Complete event will be generated.

10.5 ADD SCO CONNECTION COMMAND

Command	OGF	OCF	Command Parameters	Return Parameters
HCI_Add_SCO_Connection	0x01	0x0007	Connection_Handle, Packet_Type	

Description:

This command will cause the Link Manager to create a SCO connection using the ACL connection specified by the Connection_Handle command parameter. This command causes the local BR/EDR Controller to create a SCO connection. The Link Manager will determine how the new connection is established. This connection is determined by the current state of the device, its piconet, and the state of the device to be connected. The Packet_Type command parameter specifies which packet types the Link Manager should use for the connection. The Link Manager must only use the packet type(s) specified by the Packet_Type command parameter for sending HCI SCO Data Packets. Multiple packet types may be specified for the Packet_Type command parameter by performing a bitwise OR operation of the different packet types. The Link Manager may choose which packet type is to be used from the list of acceptable packet types. A Connection Handle for this connection is returned in the Connection Complete event (see below).

Note: An SCO connection can only be created when an ACL connection already exists and when it is not put in park. For a definition of the different packet types, see the Part B, Baseband Specification on page 59.

Note: At least one packet type must be specified. The Host should enable as many packet types as possible for the Link Manager to perform efficiently. However, the Host must not enable packet types that the local device does not support.

Command Parameters:

Connection_Handle *Size 2 Octets (12 Bits meaningful)*

Value	Parameter Description
0xXXXX	Connection Handle for the ACL connection being used to create an SCO connection. Range: 0x0000-0x0EFF (0x0F00 - 0x0FFF Reserved for future use)

Packet_Type: *Size: 2 Octets*

Value	Parameter Description
0x0001	Reserved for future use.
0x0002	Reserved for future use.

0x0004	Reserved for future use.
0x0008	Reserved for future use.
0x0010	Reserved for future use.
0x0020	HV1
0x0040	HV2
0x0080	HV3
0x0100	Reserved for future use.
0x0200	Reserved for future use.
0x0400	Reserved for future use.
0x0800	Reserved for future use.
0x1000	Reserved for future use.
0x2000	Reserved for future use.
0x4000	Reserved for future use.
0x8000	Reserved for future use.

Return Parameters:

None.

Event(s) generated (unless masked away):

When the Controller receives the Add_SCO_Connection command, it sends the Command Status event to the Host. In addition, when the LM determines the connection is established, the local Controller will send a Connection Complete event to its Host, and the remote Controller will send a Connection Complete event or a Synchronous Connection Complete event to the Host. The Connection Complete event contains the Connection Handle if this command is successful.

Note: no Command Complete event will be sent by the Controller to indicate that this command has been completed. Instead, the Connection Complete event will indicate that this command has been completed.

10.6 PAGE SCAN MODE CHANGE EVENT

Event	Event Code	Event Parameters
Page Scan Mode Change	0x1F	BD_ADDR, Page_Scan_Mode

Description:

The Page Scan Mode Change event indicates that the connected remote BR/EDR Controller with the specified BD_ADDR has successfully changed the Page_Scan_Mode.

Event Parameters:

BD_ADDR: *Size: 6 Octets*

Value	Parameter Description
0xXXXXXXXXXXXX	BD_ADDR of the remote device.

Page_Scan_Mode: *Size: 1 Octet*

Value	Parameter Description
0x00	Mandatory Page Scan Mode.
0x01	Optional Page Scan Mode I.
0x02	Optional Page Scan Mode II.
0x03	Optional Page Scan Mode III.
0x04 – 0xFF	Reserved.

10.7 READ COUNTRY CODE COMMAND

Command	OGF	OCF	Command Parameters	Return Parameters
HCI_Read_Country_Code	0x04	0x0007		Status, Country_Code

Description:

This command will read the value for the Country_Code return parameter. The Country_Code defines which range of frequency band of the ISM 2.4 GHz band will be used by the device. Each country has local regulatory bodies regulating which ISM 2.4 GHz frequency ranges can be used.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Country_Code command succeeded.
0x01-0xFF	Read_Country_Code command failed. See Part D, Error Codes on page 339 for error codes and descriptions.

Country_Code: *Size: 1 Octet*

Value	Parameter Description
0x00	North America & Europe* and Japan
0x01	France
0x04-FF	Reserved for future use.

*. Except France

Event(s) generated (unless masked away):

When the Read_Country_Code command has completed, a Command Complete event will be generated.

10.8 READ ENCRYPTION MODE COMMAND

Command	OCF	Command Parameters	Return Parameters
HCI_Read_Encryption_Mode	0x0021		Status, Encryption_Mode

Description:

This command will read the value for the Encryption_Mode configuration parameter. See Section 10.10.1 on page 869.

Command Parameters:

None.

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Read_Encryption_Mode command succeeded.
0x01-0xFF	Read_Encryption_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Encryption_Mode: *Size: 1 Octet*

Value	Parameter Description
See Section 10.10.1 on page 869	

Event(s) generated (unless masked away):

When the Read_Encryption_Mode command has completed, a Command Complete event will be generated.

10.9 WRITE ENCRYPTION MODE COMMAND

Command	OCF	Command Parameters	Return Parameters
HCI_Write_Encryption_Mode	0x0022	Encryption_Mode	Status

Description:

This command will write the value for the Encryption_Mode configuration parameter. See Section 10.10.1 on page 869.

Command Parameters:

Encryption_Mode: *Size: 1 Octet*

Value	Parameter Description
See Section 10.10.1 on page 869.	

Return Parameters:

Status: *Size: 1 Octet*

Value	Parameter Description
0x00	Write_Encryption_Mode command succeeded.
0x01-0xFF	Write_Encryption_Mode command failed. See Part D, Error Codes on page 339 for a list of error codes and descriptions.

Event(s) generated (unless masked away):

When the Write_Encryption_Mode command has completed, a Command Complete event will be generated.

10.10 DEPRECATED PARAMETERS

10.10.1 Encryption Mode

The Encryption_Mode parameter controls if the local device requires encryption to the remote device at connection setup (between the Create_Connection command or acceptance of an incoming ACL connection and the corresponding Connection Complete event). At connection setup, only devices with the Authentication_Enabled configuration parameter set to required and the Encryption_Mode configuration parameter set to required will try to encrypt the physical link to the other device.

Note: Changing this parameter does not affect existing connections.

A temporary link key is used when both broadcast and point-to-point traffic are encrypted.

The Host must not specify the Encryption_Mode parameter with more encryption capability than its local device currently supports, although the parameter is used to request the encryption capability to the remote device. Note that the Host must not request the command with the Encryption_Mode parameter set to 0x01, when the local device does not support encryption.

Note: for encryption to be used, both devices must support and enable encryption.

Value	Parameter Description

0x00	Encryption not required.
0x01	Encryption required for all connections.
0x02-0xFF	Reserved.

4. Note: in the Connection Complete event the Encryption_Mode parameter will show whether encryption was successfully turned on. The remote device may not support encryption or may have set Encryption_Mode to 0x01 when the local device has not, so the encryption mode returned in the Connection Complete event may not equal the encryption mode set in the Write_Encryption_Mode command.

10.10.2 Page Scan Mode

The Page_Scan_Mode parameter indicates the page scan mode that is used for default page scan. Currently one mandatory page scan mode and three optional page scan modes are defined. Following an inquiry response, if the Baseband timer T_mandatory_pscan has not expired, the mandatory page scan mode must be applied. For details see the Part B, Baseband Specification on page 59 (Keyword: Page-Scan-Mode, FHS-Packet, T_mandatory_pscan)

Value	Parameter Description
0x00	Mandatory Page Scan Mode
0x01	Optional Page Scan Mode I
0x02	Optional Page Scan Mode II
0x03	Optional Page Scan Mode III
0x04-0xFF	Reserved

MESSAGE SEQUENCE CHARTS

Examples of interactions between Host Controller Interface Commands and Events and Link Manager Protocol Data Units are represented in the form of message sequence charts. The message sequence charts also present interaction examples of HCI with AMP (Alternate MAC and PHY) Controllers. These charts show typical interactions and do not indicate all possible protocol behavior.

Ford Ex. 1052

CONTENTS

1 Introduction 877
1.1 Notation 877
1.2 Flow of Control 878
1.3 Example MSC 878
2 Services Without Connection Request 879
2.1 Remote Name Request 879
2.2 One-time Inquiry 881
2.3 Periodic Inquiry 883
3 ACL Connection Establishment and Detachment 885
3.1 Connection Setup 886
4 Optional Activities After ACL Connection Establishment 893
4.1 Authentication Requested 893
4.2 Simple Pairing Message Sequence Charts 894
4.2.1 Optional OOB Information Collection 894
4.2.2 Enable Simple Pairing 895
4.2.3 Connection Establishment 895
4.2.4 L2CAP Connection Request for a Secure Service 895
4.2.5 Optional OOB Information Transfer 896
4.2.6 Start Simple Pairing 896
4.2.7 IO Capability Exchange 897
4.2.8 Public Key Exchange 897
4.2.9 Authentication 898
4.2.10 Numeric Comparison 898
4.2.11 Numeric Comparison Failure on Initiating Side 899
4.2.12 Numeric Comparison Failure on Responding Side 900
4.2.13 Passkey Entry 901
4.2.14 Passkey Entry Failure on Responding Side 902
4.2.15 Passkey Entry Failure on Initiator Side 903
4.2.16 Out of Band 903
4.2.17 OOB Failure on Initiator Side 904
4.2.18 DHKey Checks 905
4.2.19 Calculate Link Key 905
4.2.20 Enable Encryption 906
4.2.21 L2CAP Connection Response 907
4.3 Link Supervision Timeout Changed Event 907
4.4 Set Connection Encryption 908

4.5 Change Connection Link Key 909
4.6 Change Connection Link Key with Encryption Pause and Resume 909
4.7 Master Link Key 911
4.8 Read Remote Supported Features 913
4.9 Read Remote Extended Features 913
4.10 Read Clock Offset 914
4.11 Role Switch on an Encrypted Link using Encryption Pause and Resume 914
4.12 Refreshing Encryption Keys 915
4.13 Read Remote Version Information 916
4.14 QOS Setup 916
4.15 Switch Role 917
4.16 AMP Physical Link Creation and Disconnect 918
4.16.1 Physical Link Establishment 919
4.16.2 Logical Link Creation 922
4.17 AMP Test Mode Sequence Charts 925
4.17.1 Discover the AMP Present and Running Transmitter and Receiver Tests 925

5 Synchronous Connection Establishment and Detachment 929
5.1 Synchronous Connection Setup 929

6 Sniff, Hold and Park 934
6.1 Sniff Mode 934
6.2 Hold Mode 935
6.3 Park State 937

7 Buffer Management, Flow Control 940

8 Loopback Mode 942
8.1 Local Loopback Mode 942
8.2 Remote Loopback Mode 944

9 List of Figures 946

1 INTRODUCTION

This section shows typical interactions between Host Controller Interface (HCI) Commands and Events and Link Manager (LM) Protocol Data Units (PDU) on the BR/EDR Controller. It focuses on the message sequence charts (MSCs) for the procedures specified in [3] “Host Controller Interface Functional Specification” with regard to LM Procedures from [2] “Link Manager Protocol” and PALs found in Volume 5.

This section illustrates only the most useful scenarios, it does not cover all possible alternatives. Furthermore, the message sequence charts do not consider errors over the air interface or host interface. In all message sequence charts it is assumed that all events are not masked, so the Host Controller will not filter out any events.

The sequence of messages in these message sequence charts is for illustrative purposes. The messages may be sent in a different order where allowed by the Link Manager, PAL, or HCI sections. If any of these charts differ with text in the Baseband, Link Manager, PAL, or HCI sections, the text in those sections shall be considered normative. This section is informative.

1.1 NOTATION

The notation used in the message sequence charts (MSCs) consists of ovals, elongated hexagons, boxes, lines, and arrows. The vertical lines terminated on the top by a shadow box and at the bottom by solid oval indicate a protocol entity that resides in a device. MSCs describe interactions between these entities and states those entities may be in.

The following symbols represent interactions and states:

Oval	Defines the context for the message sequence chart.
Hexagon	Indicates a condition needed to start the transactions below this hexagon. The location and width of the Hexagon indicates which entity or entities make this decision.
Box	Replaces a group of transactions. May indicate a user action, or a procedure in the baseband.
Dashed Box	Optional group of transactions.
Solid Arrow	Represents a message, signal or transaction. Can be used to show LMP and HCI traffic. Some baseband packet traffic is also shown. These are prefixed by BB followed by either the type of packet, or an indication that there is an ACK signal in a packet.
Dashed Arrow	Represents a optional message, signal or transaction. Can be used to show LMP and HCI traffic.

1.2 FLOW OF CONTROL

Some message sequences are split into several charts. These charts are marked in sequence with different step numbers with multiple paths through with optional letters after the step numbers. Numbers indicate normal or required ordering. The letters represent alternative paths. For example, Step 4 is after Step 3, and Step 5a could be executed instead of Step 5b.

1.3 EXAMPLE MSC

The protocol entities represented in the example shown in Figure 1.1 on page 878 illustrate the interactions of two devices named A and B. Note that each device includes a Host and a LM entity in this example. Other MSCs in this section may show the interactions of more than two devices.

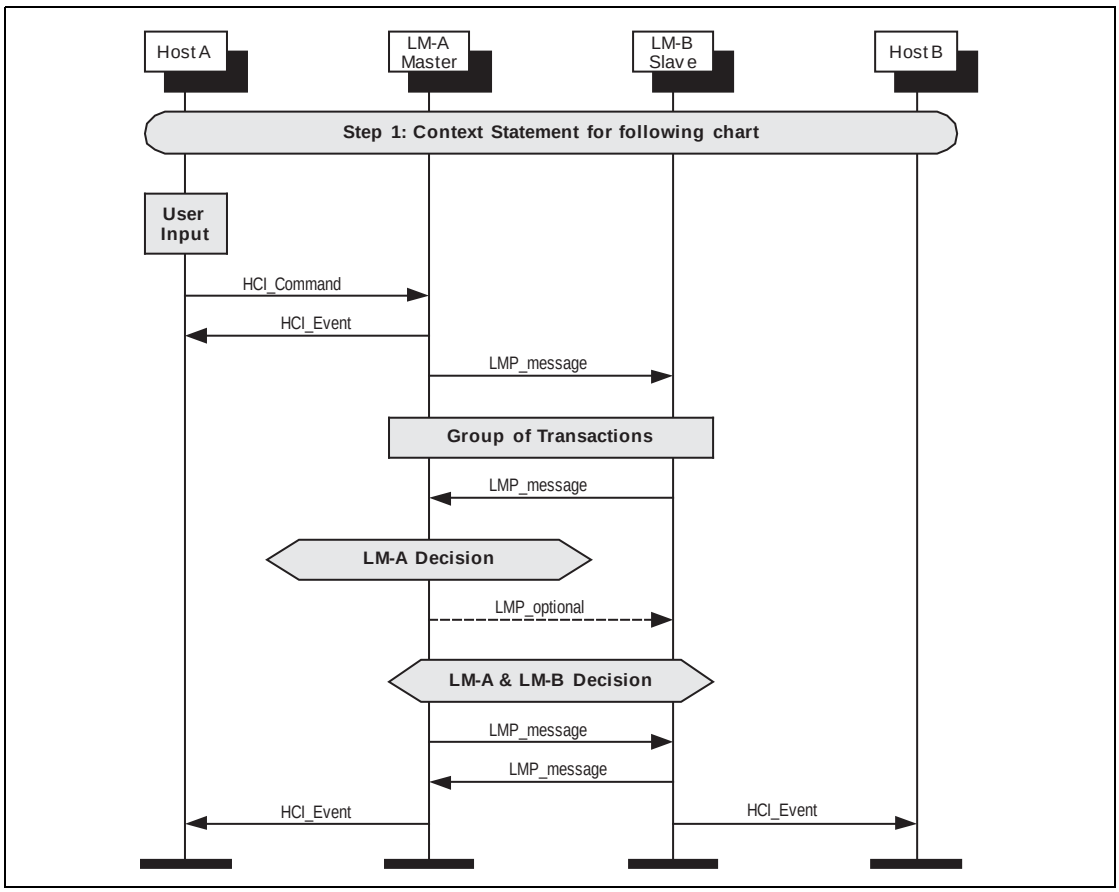


Figure 1.1: Example MSC.

2 SERVICES WITHOUT CONNECTION REQUEST

2.1 REMOTE NAME REQUEST

The service Remote Name Request is used to find out the name of the remote device without requiring an explicit ACL Connection.

Step 1: The host sends an HCI_Set_Event_Mask with the bit of Remote Host Supported Features Notification Event (0x1000000000000000) set and an HCI_Remote_Name_Request command expecting that its local device will automatically try to connect to the remote device. (See Figure 2.1 on page 879.)

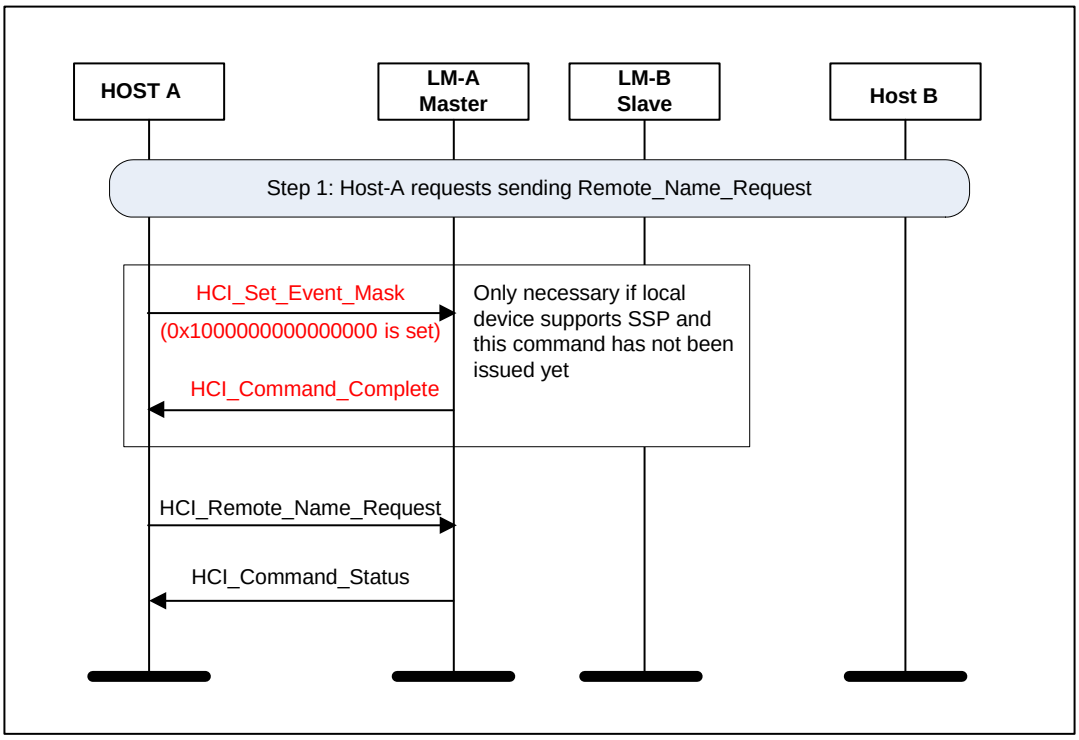


Figure 2.1: Remote name request.

Step 2a: If an ACL Connection does not exist device A pages device B. After the Baseband paging procedure, the local device attempts to get the remote device's extended features, send an HCI_Remote_Device_Supported_Features_Notification event, get the remote name, disconnect, and return the name of the remote device to the Host.(See Figure 2.2 on page 880.)

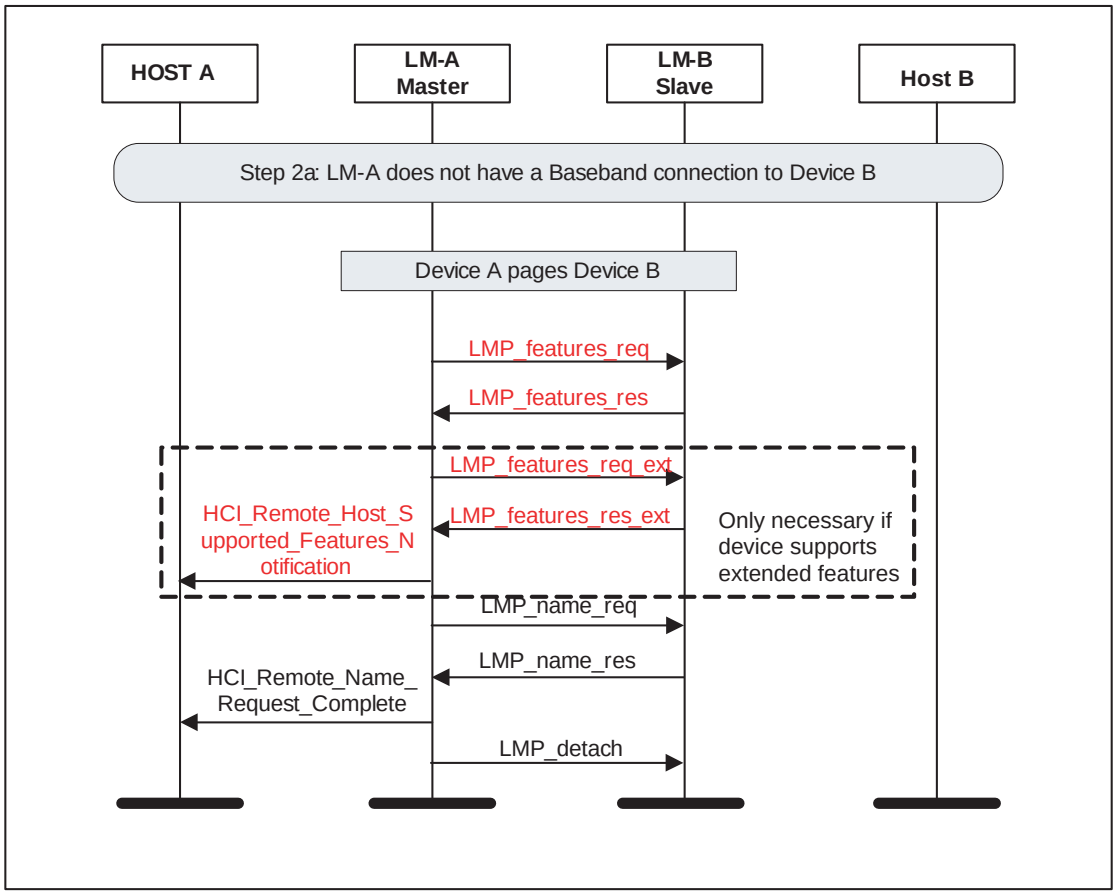


Figure 2.2: Remote name request if no current baseband connection.

Step 2b: If an ACL Connection exists when the request is made, then the Remote Name Request procedure will be executed like an optional service. No Paging and no ACL disconnect is done. (See Figure 2.3 on page 881.)

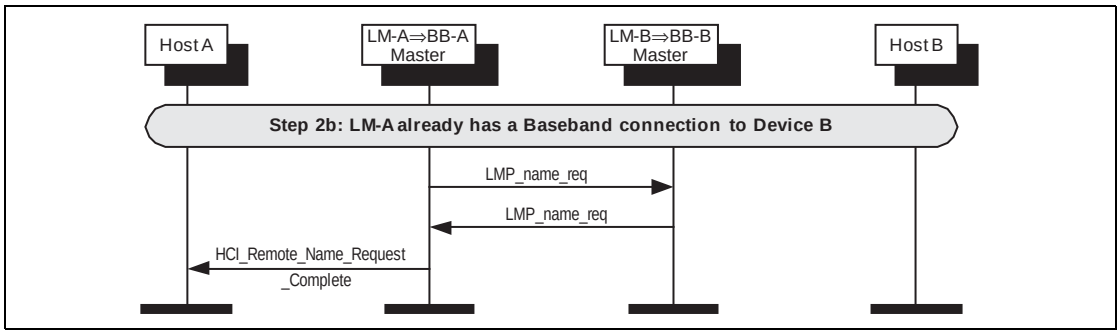


Figure 2.3: Remote name request with baseband connection.

2.2 ONE-TIME INQUIRY

Inquiry is used to detect and collect nearby devices.

Step 1: The host sends an HCI_Inquiry command. (See Figure 2.4 on page 881.)

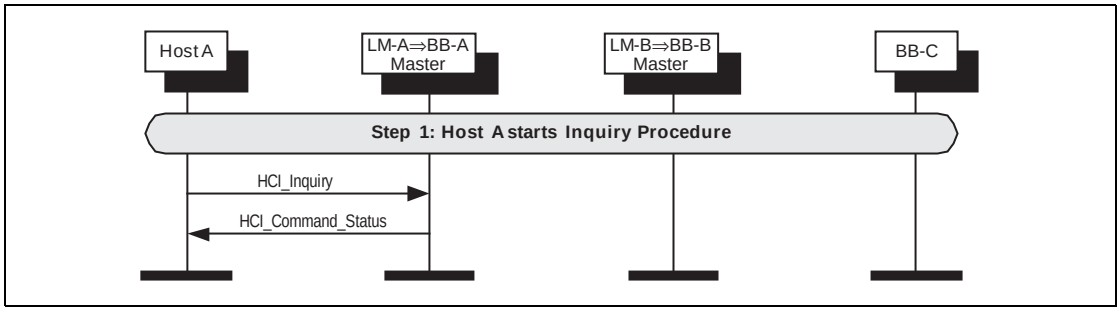


Figure 2.4: Host A starts inquiry procedure.

Step 2: The Controller will start the Baseband inquiry procedure with the specified Inquiry Access Code and Inquiry Length. When Inquiry Responses are received, the Controller extracts the required information and returns the information related to the found devices using one or more Inquiry Result events to the Host. (See Figure 2.5 on page 882.)

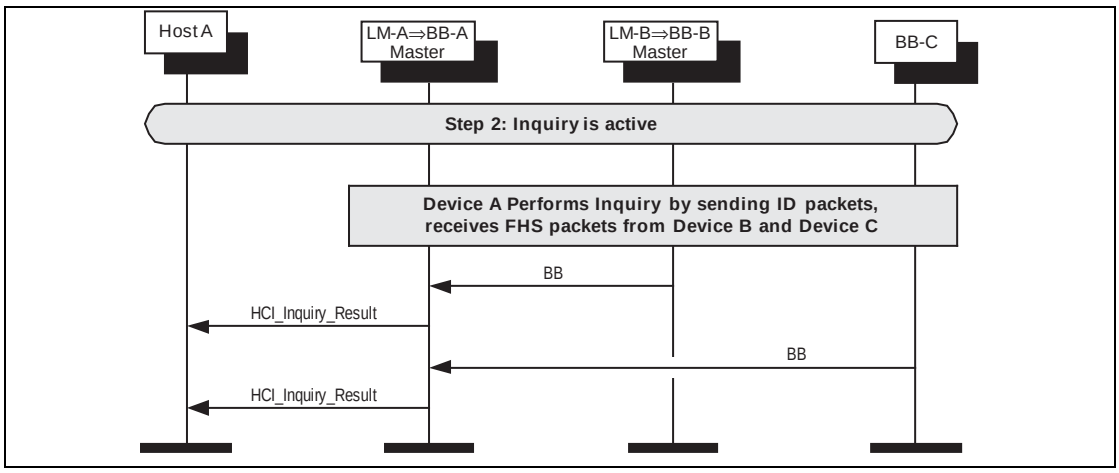


Figure 2.5: LM-A performs inquiry and reports result.

Step 3a: If the host wishes to terminate an Inquiry, the HCI_Inquiry_Cancel command is used to immediately stop the inquiry procedure.
(See Figure 2.6 on page 882.)

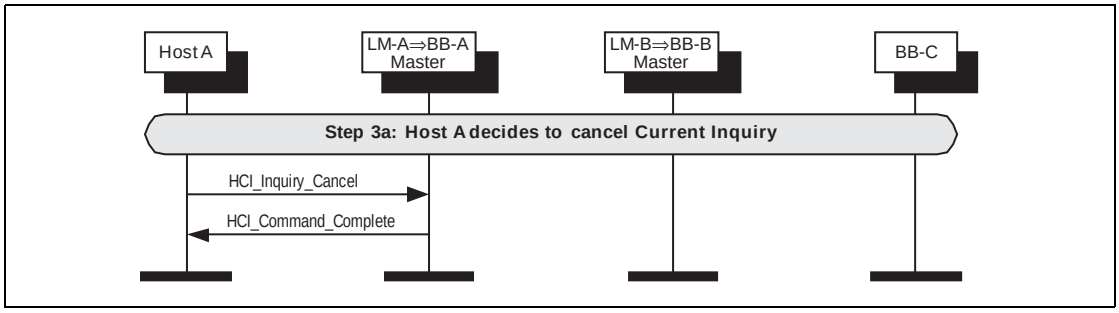


Figure 2.6: Host A cancels inquiry.

Step 3b: If the Inquiry procedure is completed due to the number of results obtained, or the Inquiry Length has expired, an Inquiry Complete event is returned to the Host. (See Figure 2.7 on page 883.)

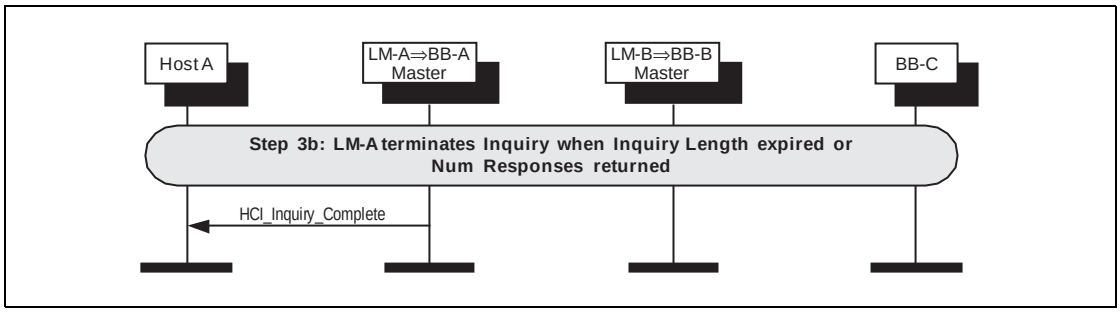


Figure 2.7: LM-A terminates current inquiry.

2.3 PERIODIC INQUIRY

Periodic inquiry is used when the inquiry procedure is to be repeated periodically.

Step 1: The hosts sends an HCI_Periodic_Inquiry_Mode command. (See Figure 2.8 on page 883.)

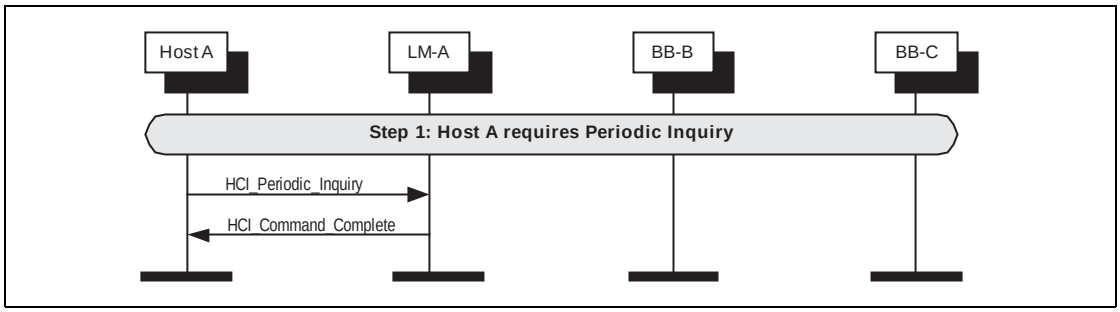


Figure 2.8: Host A starts periodic inquiry.

Step 2: The Controller will start a periodic Inquiry. In the inquiry cycle, one or several Inquiry Result events will be returned. (See Figure 2.9 on page 884.)

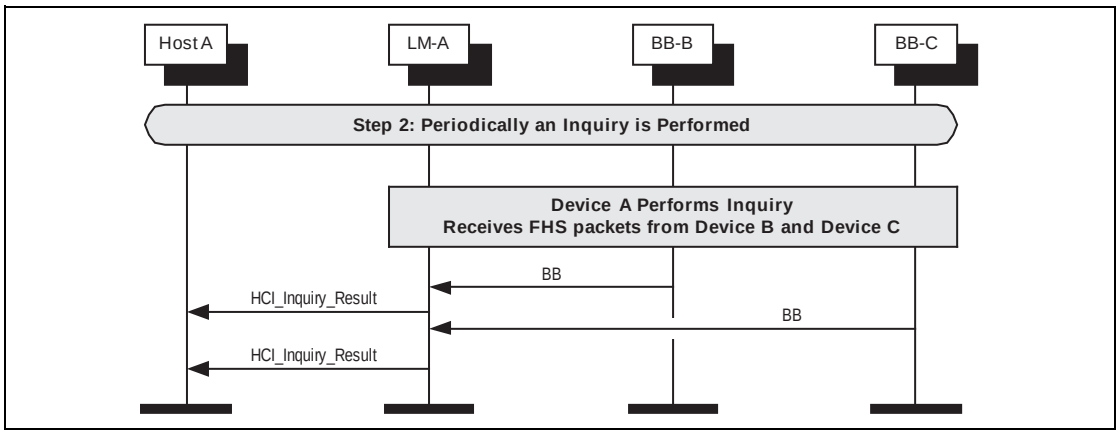


Figure 2.9: LM-A periodically performs an inquiry and reports result.

Step 3: An Inquiry Complete event will be returned to the Host when the current periodic inquiry has finished. (See Figure 2.10 on page 884.)

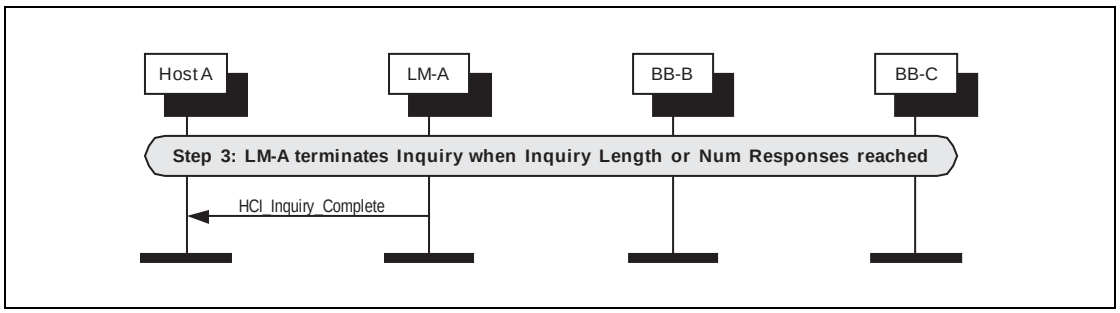


Figure 2.10: LM-A terminates current inquiry.

Step 4: The periodic Inquiry can be stopped using the HCI_Exit_Periodic_Inquiry_Mode command. (See Figure 2.11 on page 884.)

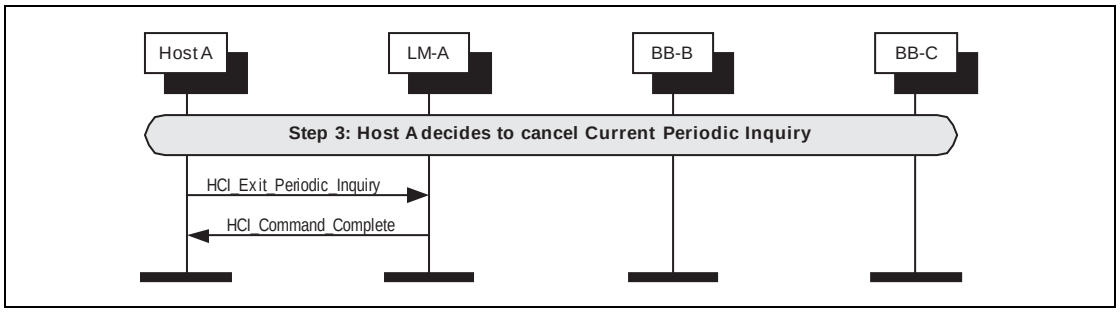


Figure 2.11: Host A decides to exit periodic inquiry.

3 ACL CONNECTION ESTABLISHMENT AND DETACHMENT

A flow diagram of the establishment and detachment of a connection between two devices is shown in Figure 3.1 on page 885. The process is illustrated in 9 distinct steps. A number of these steps may be optionally performed, such as authentication and encryption. Some steps are required, such as the Connection Request and Setup Complete steps. The steps in the overview diagram directly relate to the steps in the following message sequence charts.

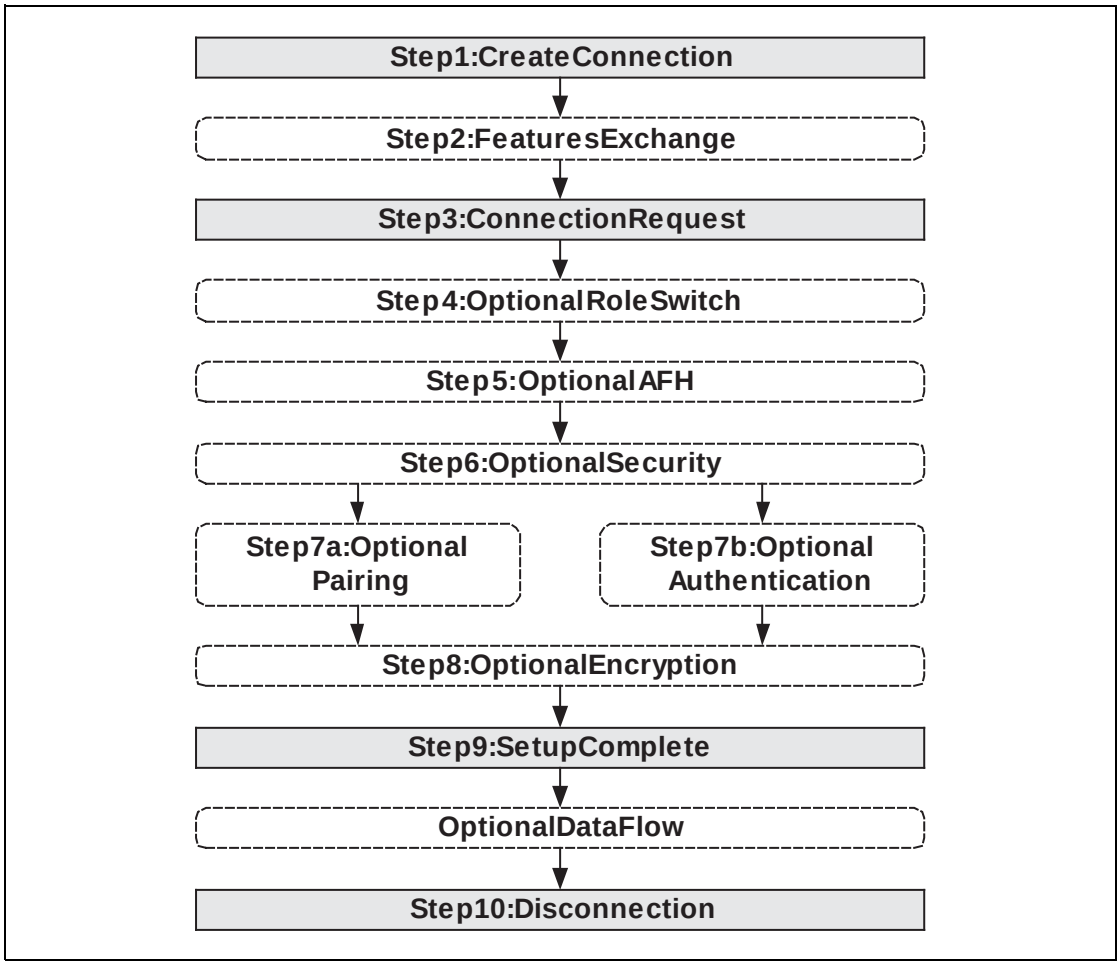


Figure 3.1: Overview diagram for connection setup.

3.1 CONNECTION SETUP

Step 1: The host sends an HCI_Create_Connection command to the Controller. The Controller then performs a Baseband paging procedure with the specified BD_ADDR. (See Figure 3.2 on page 886.)

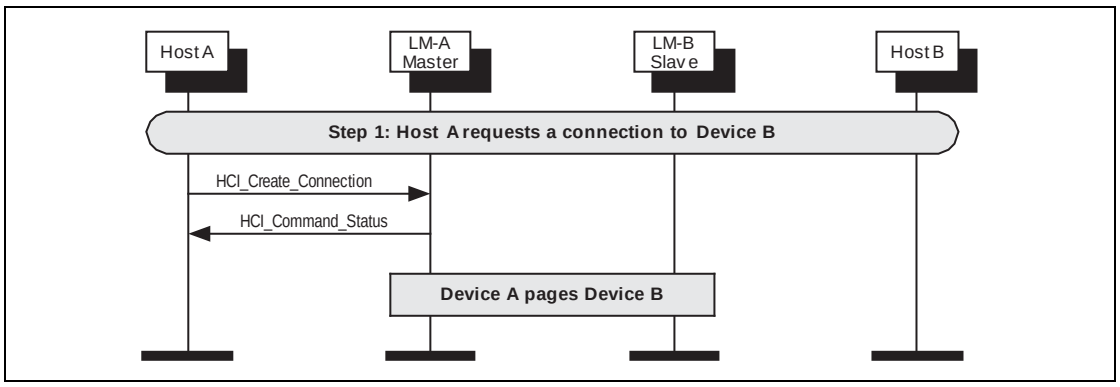


Figure 3.2: Host A requests connection with device B.

Step 2: Optionally, the LM may decide to exchange features. (See Figure 3.3 on page 886.)

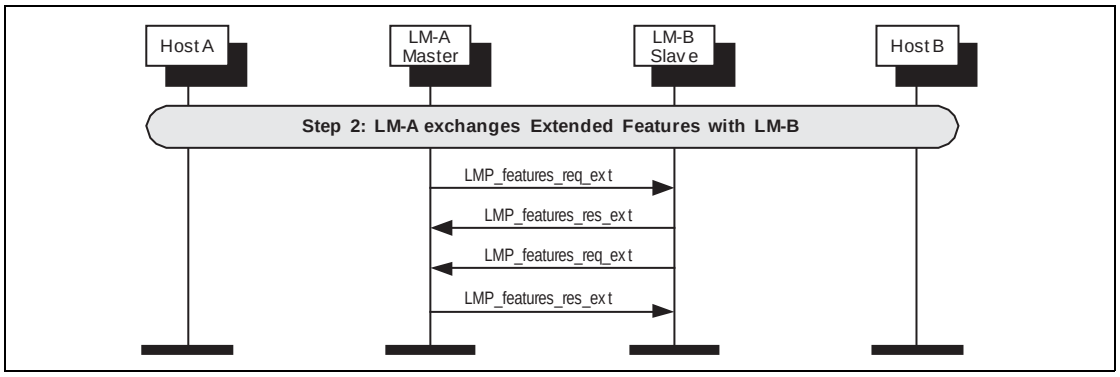


Figure 3.3: LM-A and LM-B exchange features.

Step 3: The LM on the master will request an LMP_host_connection_req PDU. The LM on the slave will then confirm that a connection is OK, and if so, what role is preferred. (See Figure 3.4 on page 886)

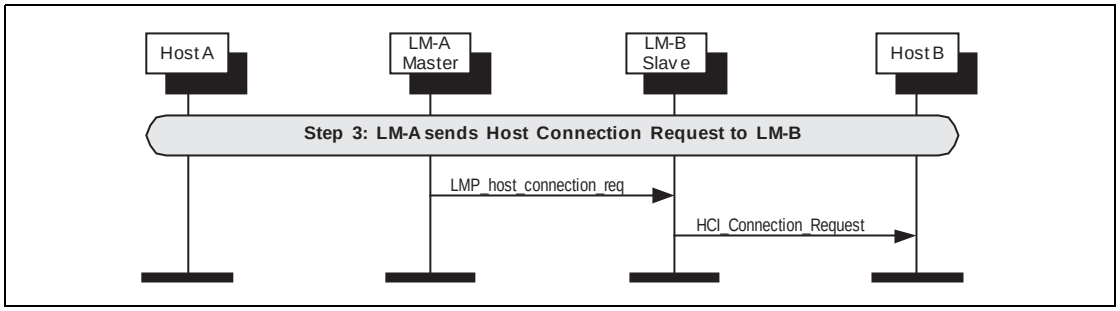


Figure 3.4: LM-A requests host connection.

Step 4a: The remote host rejects this connection, and the link is terminated. (See Figure 3.5 on page 887.)

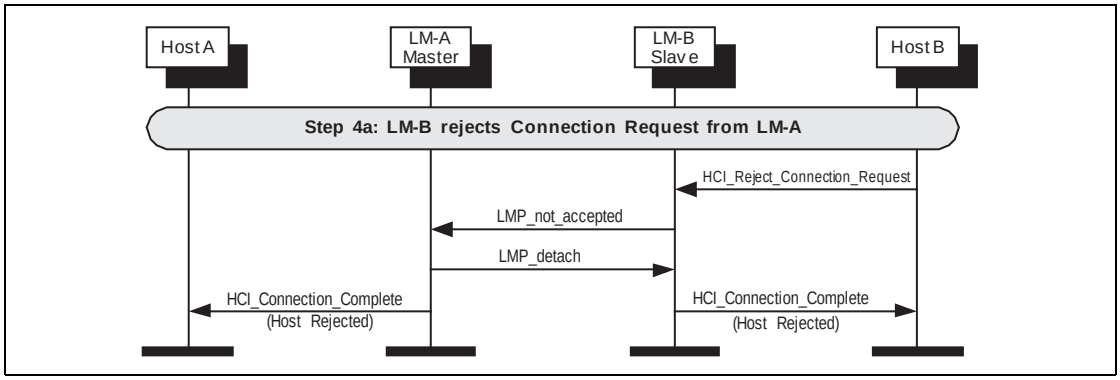


Figure 3.5: Device B rejects connection request.

Step 4b: The remote host accepts this connection. (See Figure 3.6 on page 887.)

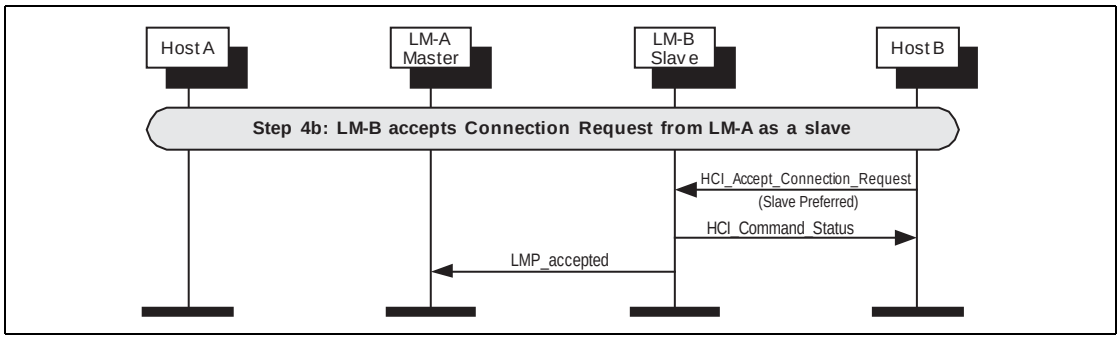


Figure 3.6: Device B accepts connection request.

Step 4c: The remote host accepts this connection but with the preference of being a master. This will cause a role switch to occur before the LMP_accepted for the LMP_host_connection_req PDU is sent. (See Figure 3.7 on page 888.)

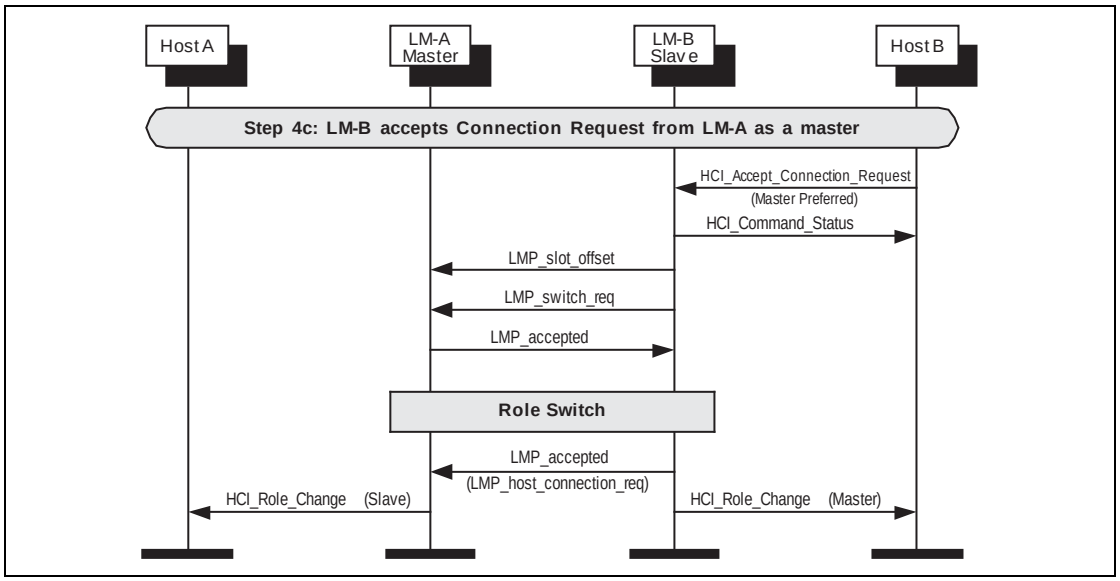


Figure 3.7: Device B accepts connection requests as master.

Step 5: After the features have been exchanged and AFH support is determined to be available, the master may at any time send an LMP_set_AFH and LMP_channel_classification_req PDU. (See Figure 3.8 on page 888.)

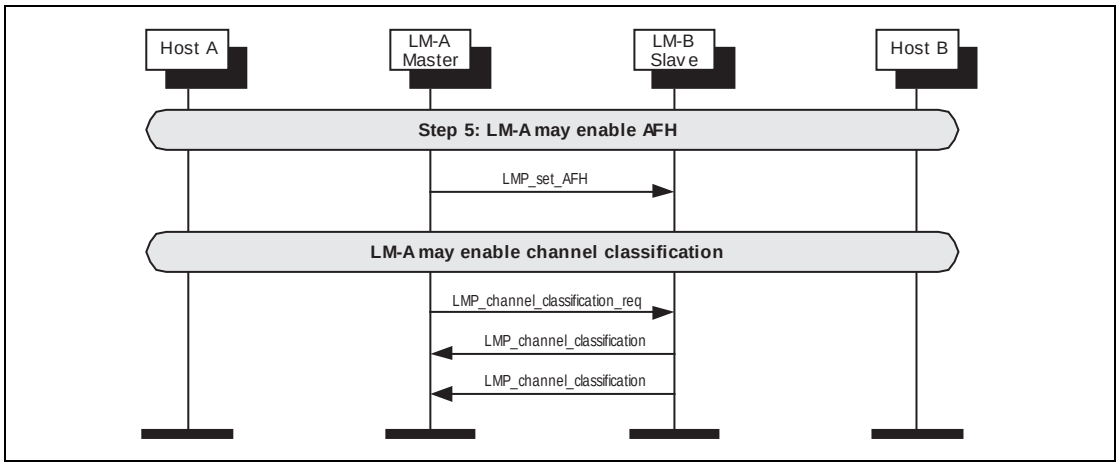


Figure 3.8: LM-A starts adaptive frequency hopping.

Step 6: The LM will request if authentication is required. It does this by requesting the Link Key for this connection from the Host. (See Figure 3.9 on page 889.)

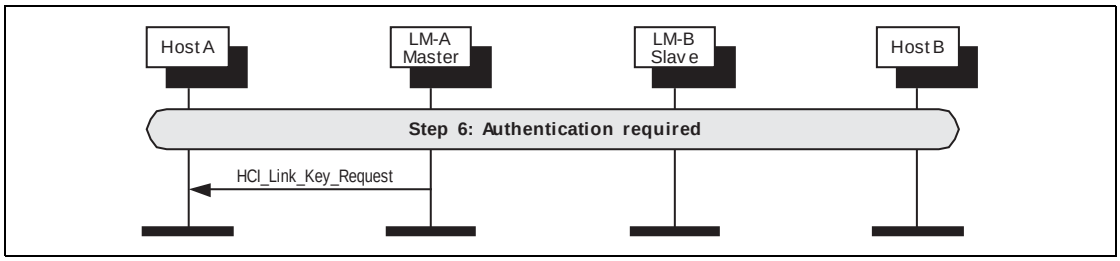


Figure 3.9: Authentication initiated.

Step 7a: If authentication is required by the higher layers and the devices to be connected do not have a common link key, a pairing procedure will be used. The LM will have requested a link key from the host for this connection. If there is a negative reply, then a PIN code will be requested. This PIN code will be requested on both sides of the connection, and authentication performed based on this PIN code. The last step is for the new link key for this connection to be passed to the host so that it may store it for future connections.
(See Figure 3.10 on page 890.)

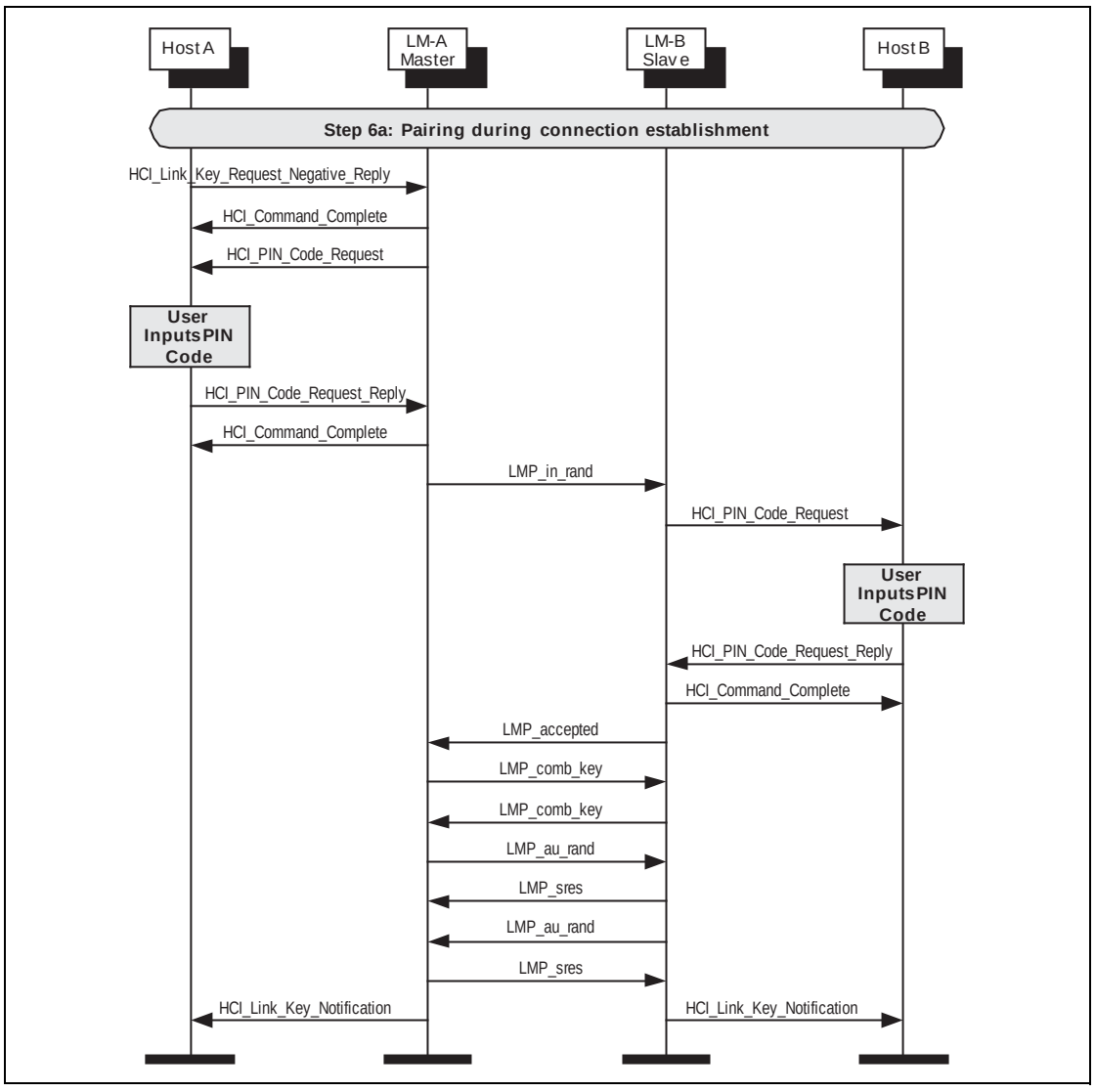


Figure 3.10: Pairing during connection setup.

Step 7b: If a common link key exists between the devices, then pairing is not needed. The LM will have asked for a link key from the host for this connection. If this is a positive reply, then the link key is used for authentication. If the configuration parameter Authentication_Enable is set, then the authentication procedure must be executed. This MSC only shows the case when Authentication_Enable is set on both sides. (See Figure 3.11.)

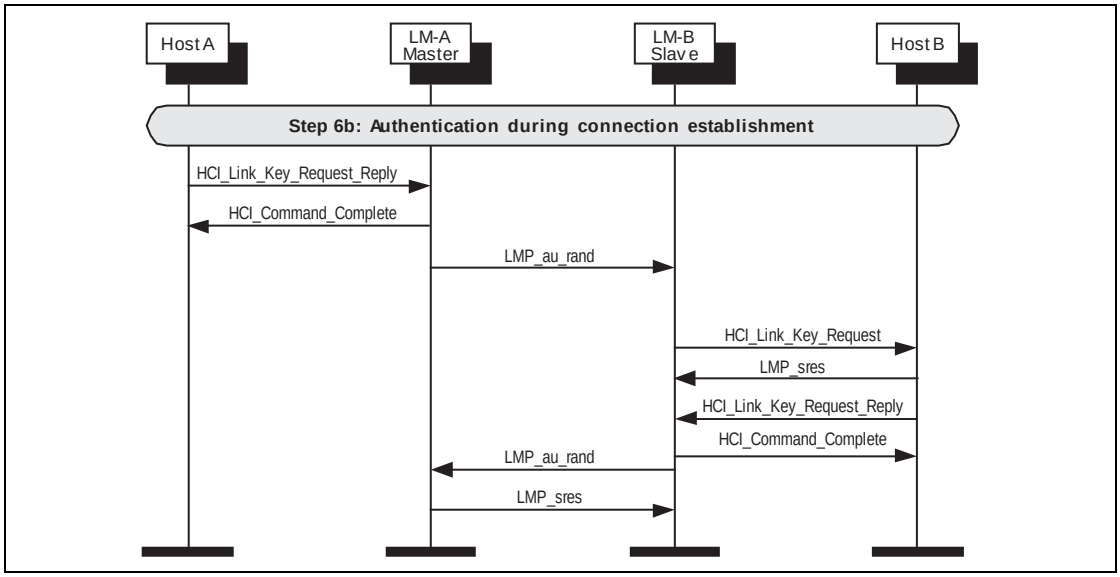


Figure 3.11: Authentication during connection setup.

Step 8: Once the pairing or authentication procedure is successful, the encryption procedure may be started. This MSC only shows the set up of an encrypted point-to-point connection. (See Figure 3.12.)

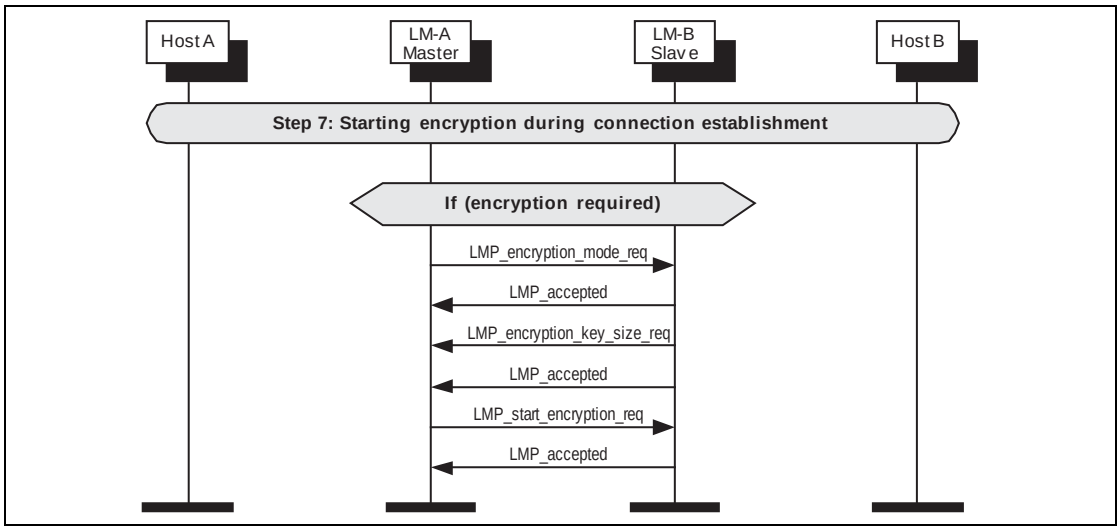


Figure 3.12: Starting encryption during connection setup.

Step 9: The LMs indicate that the connection is setup by sending LMP_setup_complete PDU. This will cause the Host to be notified of the new connection handle, and this connection may be used to send higher layer data such as L2CAP information. (See Figure 3.13 on page 892.)

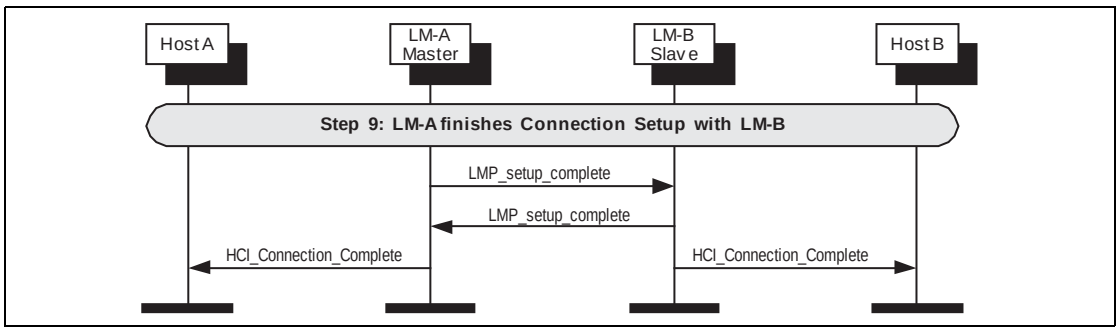


Figure 3.13: LM-A and LM-B finishes connection setup.

Step 10: Once the connection is no longer needed, either device may terminate the connection using the HCI_Disconnect command and LMP_detach message PDU. The disconnection procedure is one-sided and does not need an explicit acknowledgment from the remote LM. The use of ARQ Acknowledgment from the Baseband is needed to ensure that the remote LM has received the LMP_detach PDU. (See Figure 3.13 on page 892.)

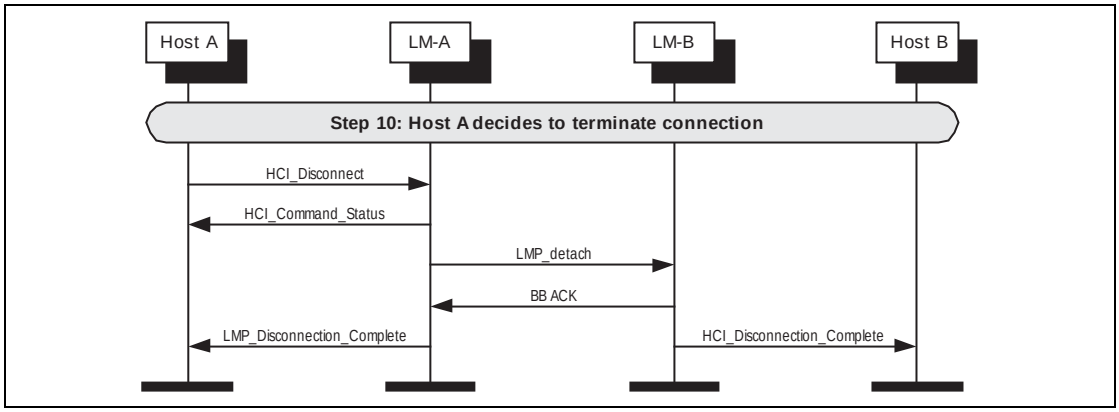


Figure 3.14: Host A decides to disconnect.

4 OPTIONAL ACTIVITIES AFTER ACL CONNECTION ESTABLISHMENT

4.1 AUTHENTICATION REQUESTED

Step 1: Authentication can be explicitly executed at any time after a connection has been established. If no Link Key is available then the Link Key is required from the Host. (See Figure 4.1 on page 893.)

Note: If the Controller or LM and the Host do not have the Link Key a PIN Code Request event will be sent to the Host to request a PIN Code for pairing. A procedure identical to that used during Connection Setup (Section 3.1, Step 7a:) will be used. (See Figure 3.9 on page 889.)

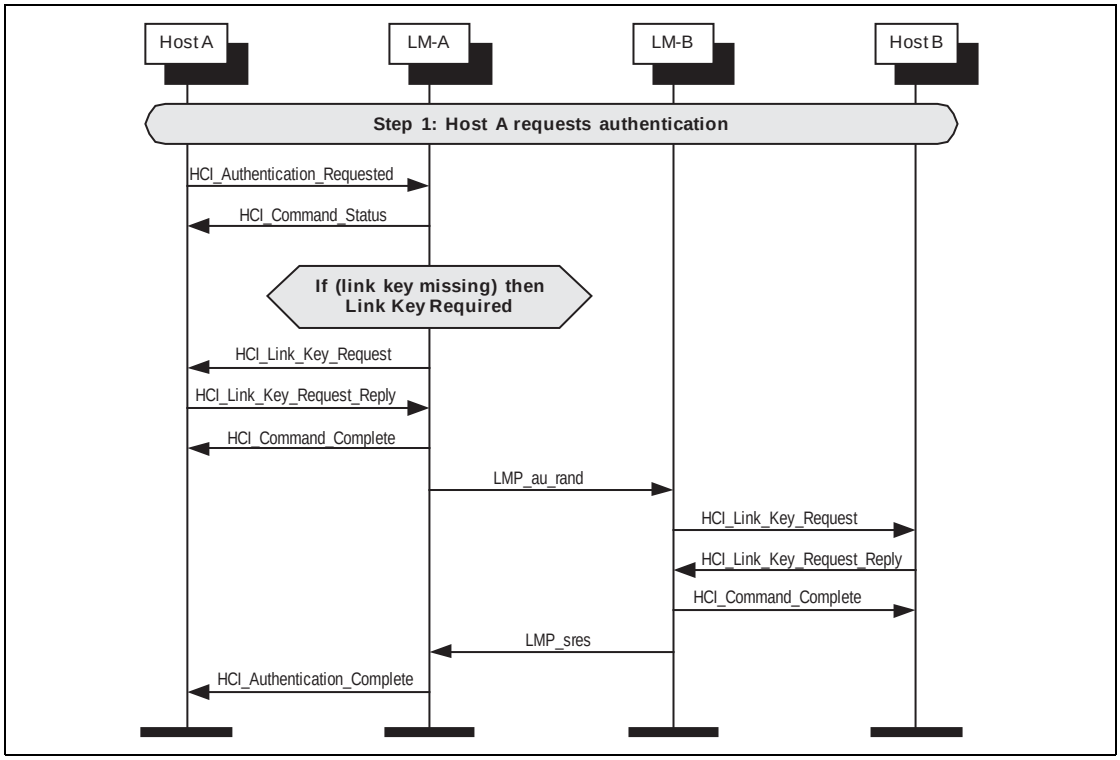


Figure 4.1: Authentication requested.

4.2 SIMPLE PAIRING MESSAGE SEQUENCE CHARTS

A flow diagram of simple pairing between two devices is shown in Figure 4.2. The process is illustrated in 11 distinct steps. A number of these steps have a number of different options.

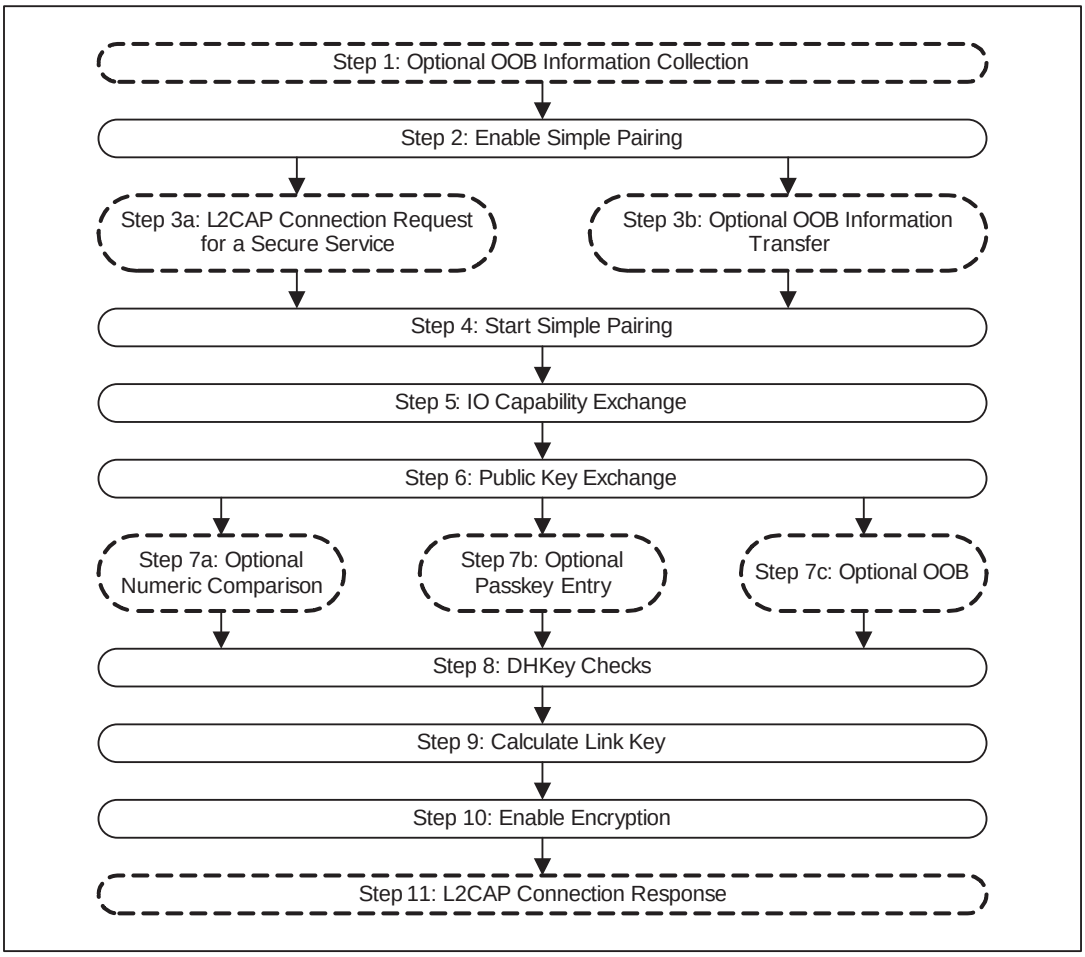


Figure 4.2: Simple Pairing Flow Diagram

4.2.1 Optional OOB Information Collection

If a device supports OOB information exchange, then the Host should request the C and R values from the controller that need to be sent by OOB. It is then assumed that the host transfers this information to the OOB system. Note: This could occur a long time before, for example at the factory for a passive tag.

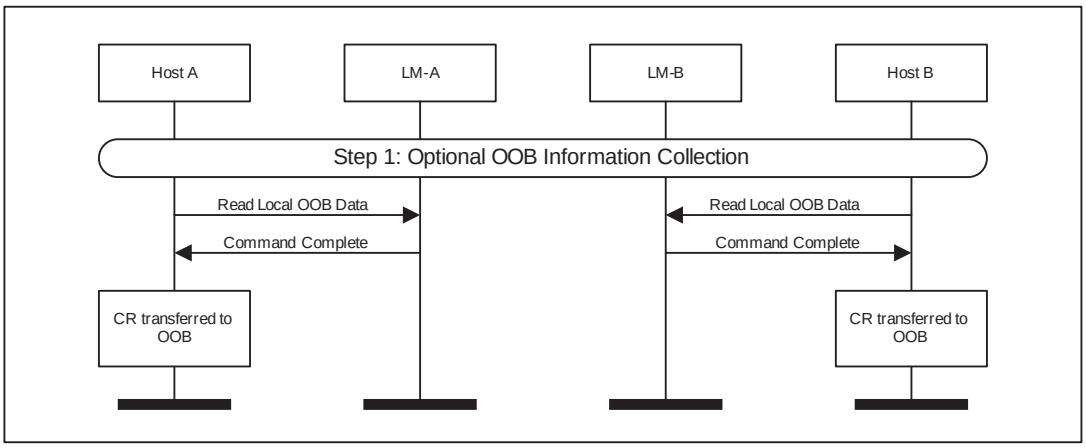


Figure 4.3: Optional OOB Information Collection

4.2.2 Enable Simple Pairing

To enable simple pairing, a device must use the Write Simple Pairing Mode command. This should be done before any other connections that may use simple pairing are created.

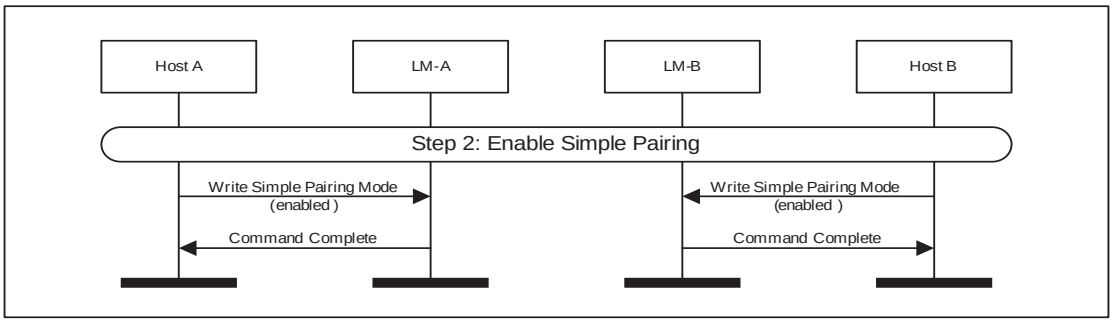


Figure 4.4: Enable Simple Pairing

4.2.3 Connection Establishment

Simple pairing, once it is enabled, is triggered by one of two possible actions. It could be triggered by an L2CAP connection request to a service that requires security, or it could be triggered by an OOB transfer of information.

4.2.4 L2CAP Connection Request for a Secure Service

Once a connection has been established between two devices, if a device requests an L2CAP connection to a service that requires authentication and encryption, then the device will start simple pairing.

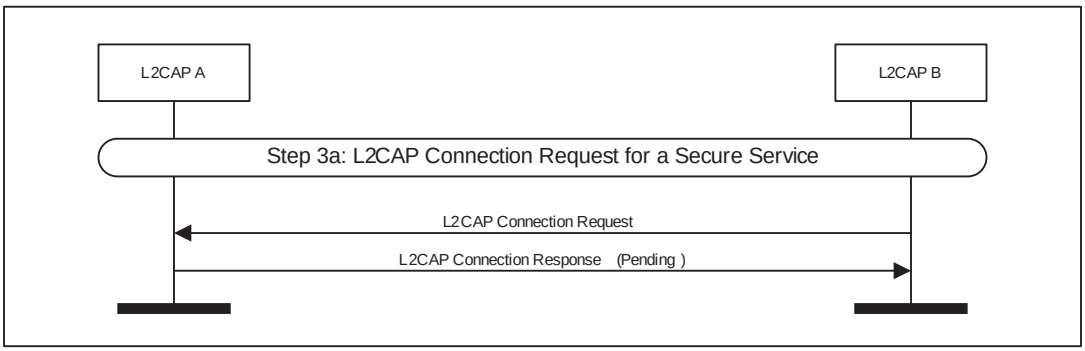


Figure 4.5: L2CAP Connection Request for a Secure Service

4.2.5 Optional OOB Information Transfer

Even if a Bluetooth connection has not been established between two devices, an OOB transfer can occur that transfers the Bluetooth Device Address of the device, and other OOB information for authentication. If an OOB transfer occurs, then the host can start simple pairing.

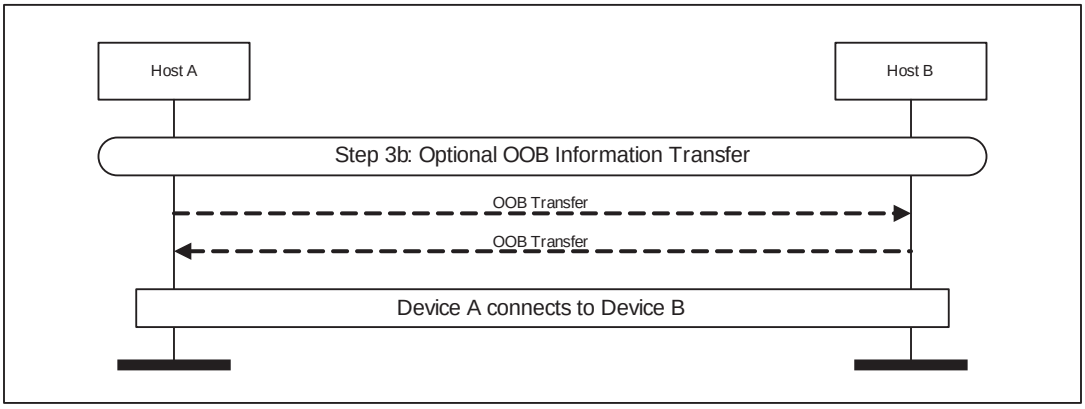


Figure 4.6: Optional OOB Information Transfer

4.2.6 Start Simple Pairing

Once the host has determined that simple pairing should start, it issues an Authentication Requested command to the controller. This will cause the controller to generate a request for a link key. If the host has a link key for this connection, then pairing is not required, and the link key can be used immediately once it has been authenticated. Simple Pairing will only be used if a Link_Key_Request_Negative_Reply Command is sent from the Host to the Controller on the initiating side.

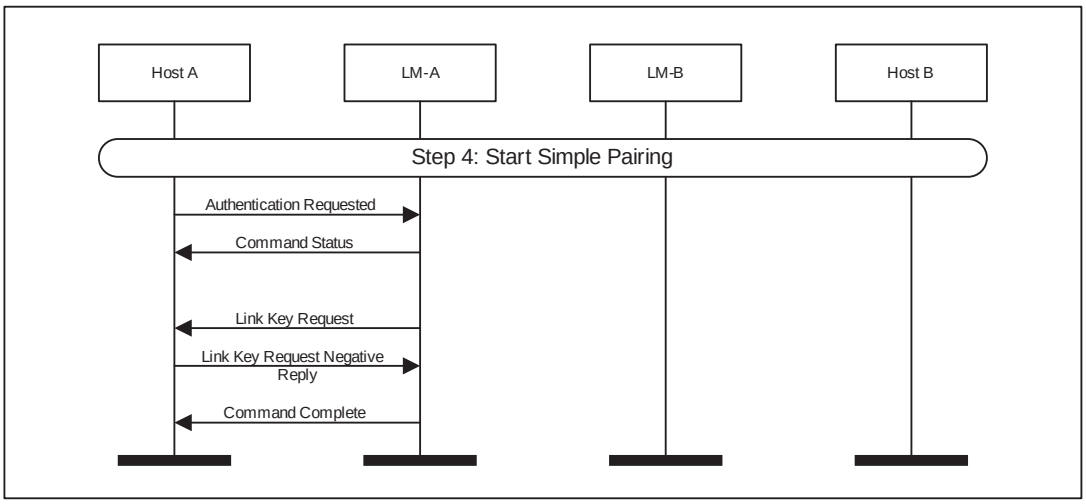


Figure 4.7: Start Simple Pairing

4.2.7 IO Capability Exchange

To be able to determine the correct authentication algorithm to use, the input / output capabilities of the two devices are exchanged.

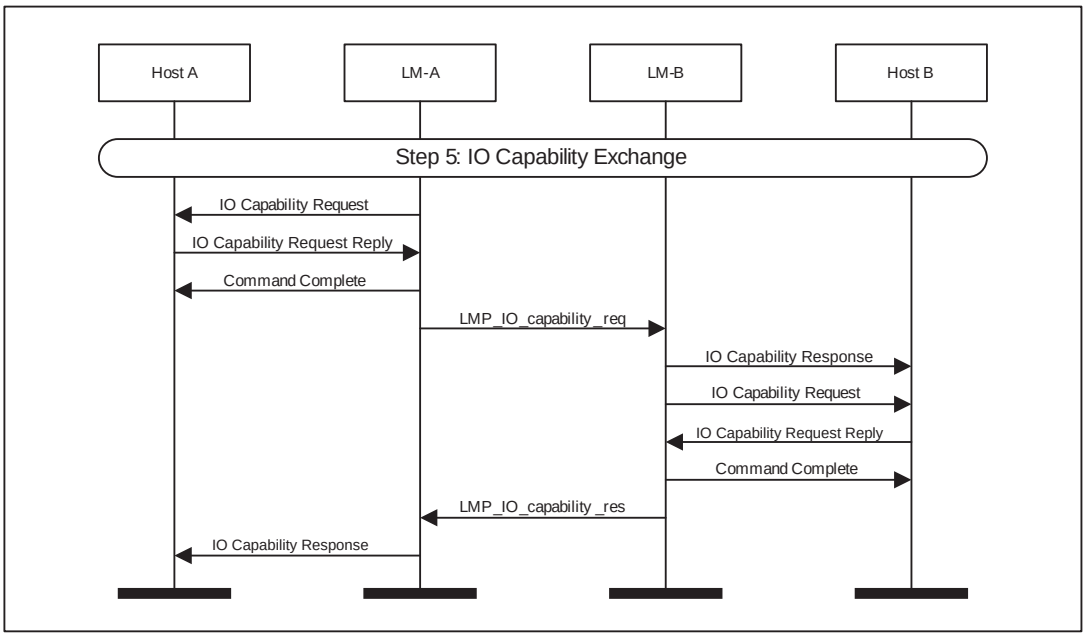


Figure 4.8: IO Capability Exchange

4.2.8 Public Key Exchange

Next the public keys are exchanged between the two devices. Once a device has received the public key of the peer device, it can start to calculate the Diffie Hellman Key (DHKey). This may take a long time, and should be started early, so that user interaction can hide the calculation time. The DHKey is not required until step 8.

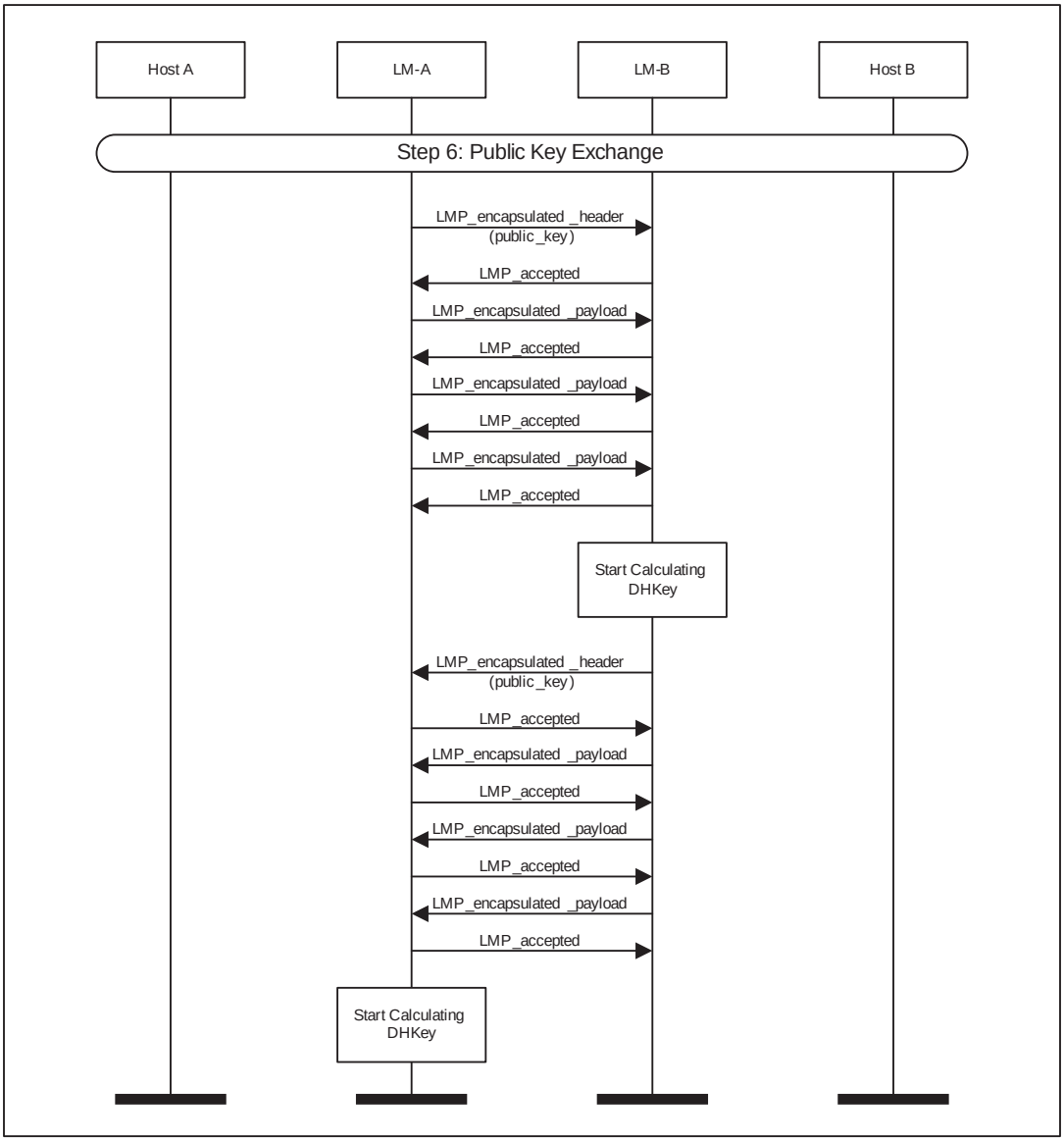


Figure 4.9: Public Key Exchange

4.2.9 Authentication

A device can be authenticated by using one of three algorithms. The choice of algorithm is determined by the combination of the IO capabilities of the two devices.

4.2.10 Numeric Comparison

The numeric comparison step will be done when both devices have output capabilities, or if one of the devices has no input or output capabilities. If both devices have output capabilities, this step requires the displaying of a user confirmation value. This value should be displayed until the end of step 8. If one or both devices do not have output capabilities, the same protocol is used but the Hosts will skip the step asking for the user confirmation.

Note: The sequence for Just Works is identical to that of Numeric Comparison with the exception that the Host will not show the numbers to the user.

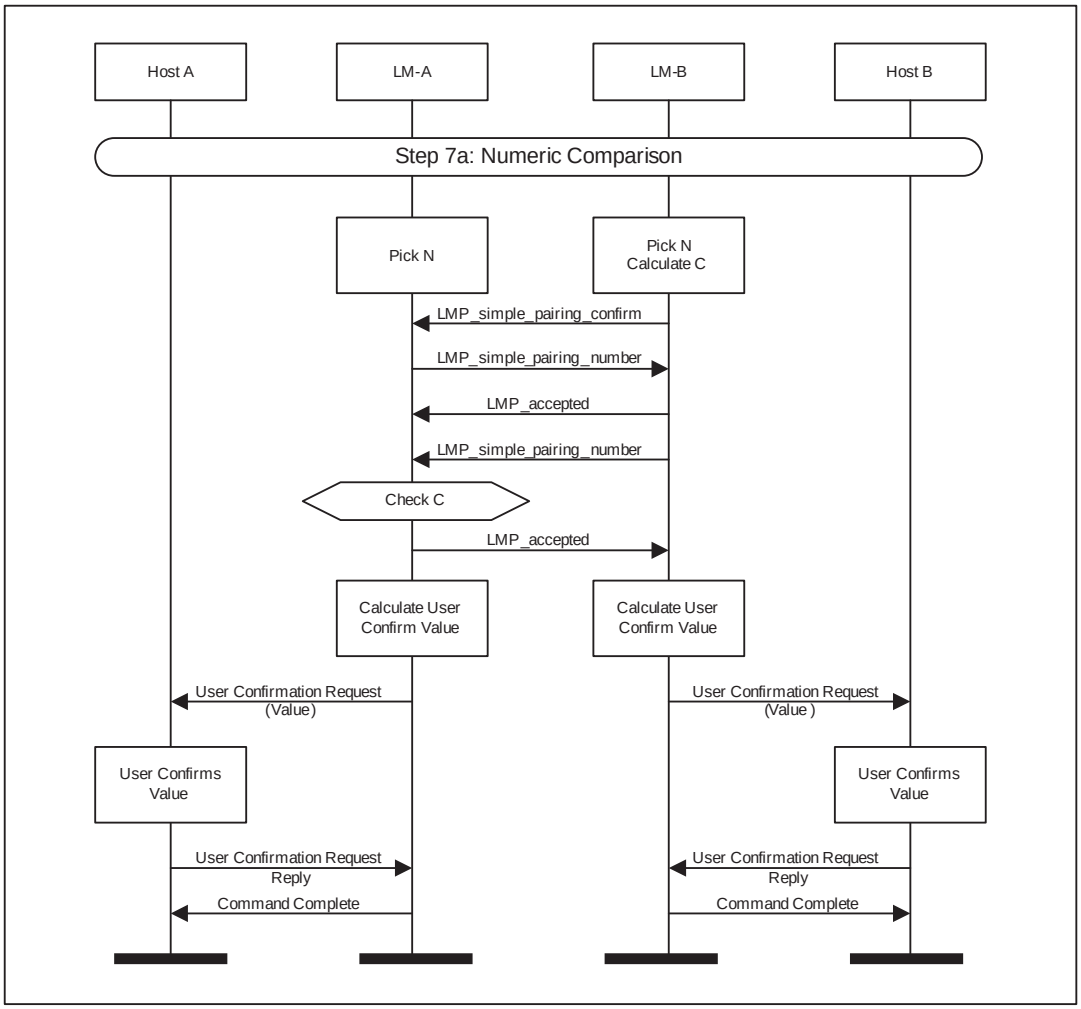


Figure 4.10: Numeric Comparison Authentication

4.2.11 Numeric Comparison Failure on Initiating Side

If the numeric comparison fails on the initiating side due to the user indicating that the confirmation values do not match, Simple Pairing is terminated.

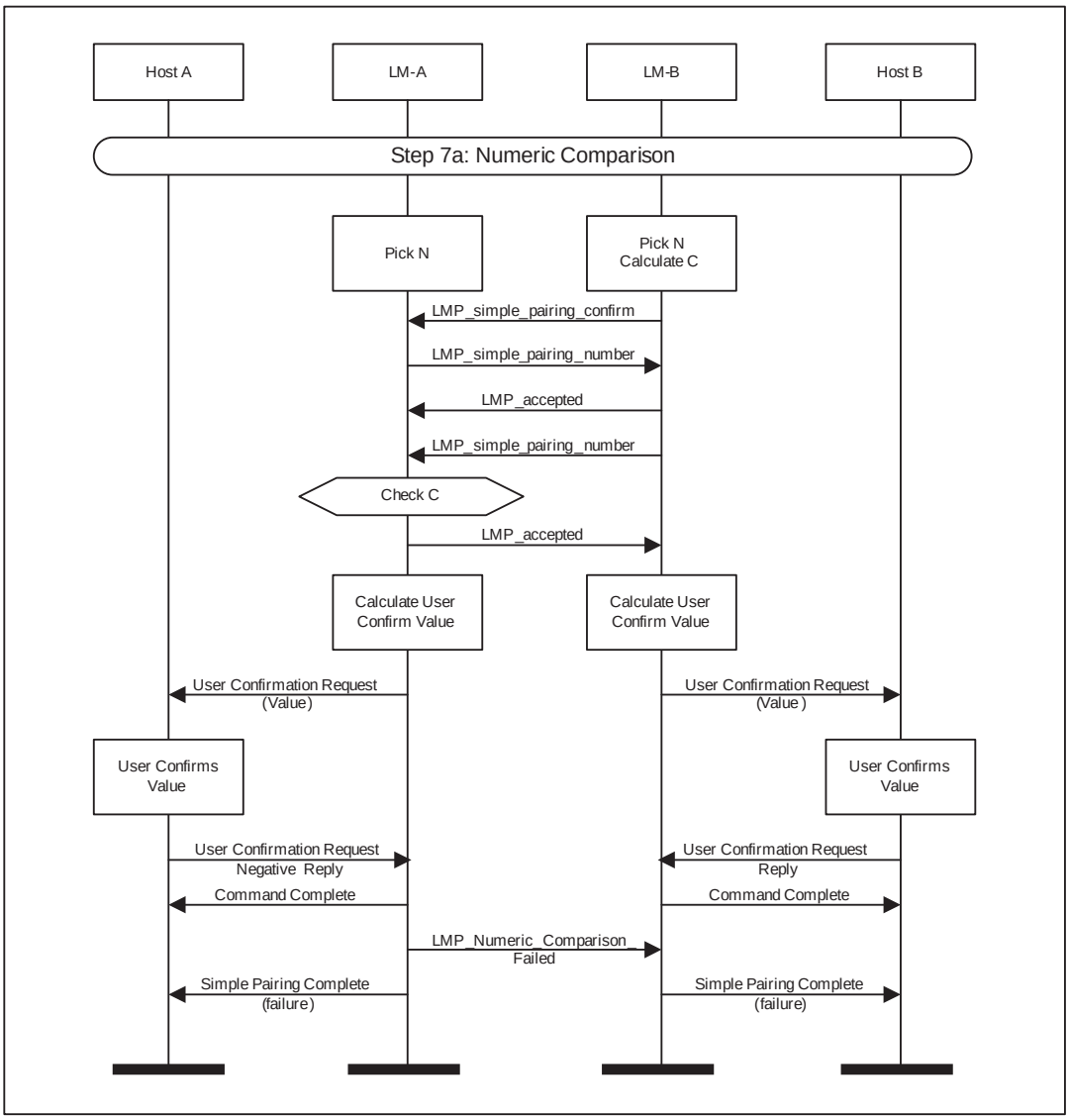


Figure 4.11: Numeric Comparison Authentication (Failure on Initiating Side)

4.2.12 Numeric Comparison Failure on Responding Side

If the numeric comparison fails on the responding side due to the user indicating that the confirmation values do not match, Simple Pairing is terminated.

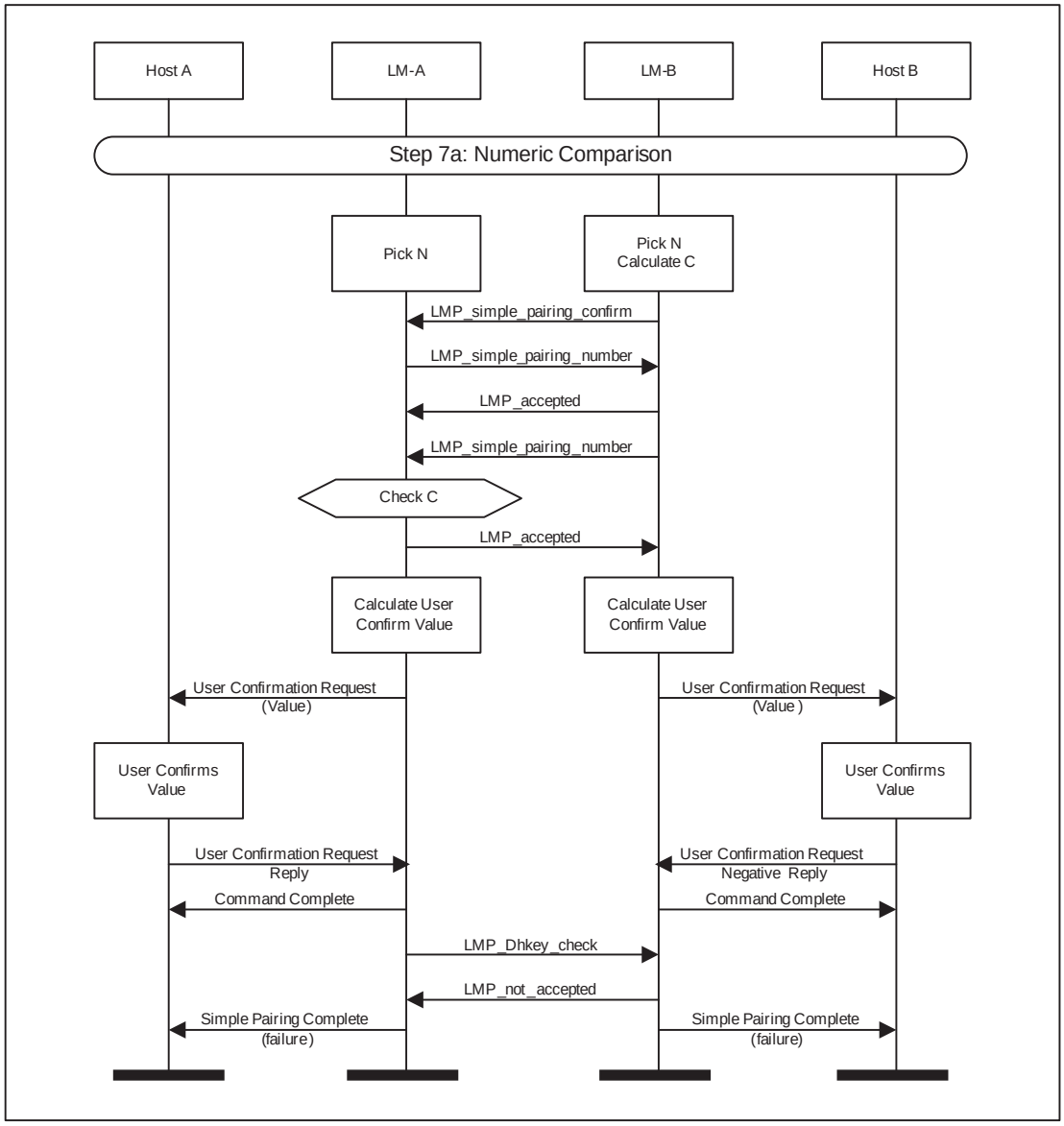


Figure 4.12: Numeric Comparison Failure on Responding Side

4.2.13 Passkey Entry

The Passkey entry step is used in two cases: when one device has numeric input only and the other device has either a display or numeric input capability or when both devices only have numeric input capability. In this step, one device display a number to be entered by the other device or the user enters a number on both devices. This number should be displayed until the end of step 8. Key press notification messages are shown during the user input phase.

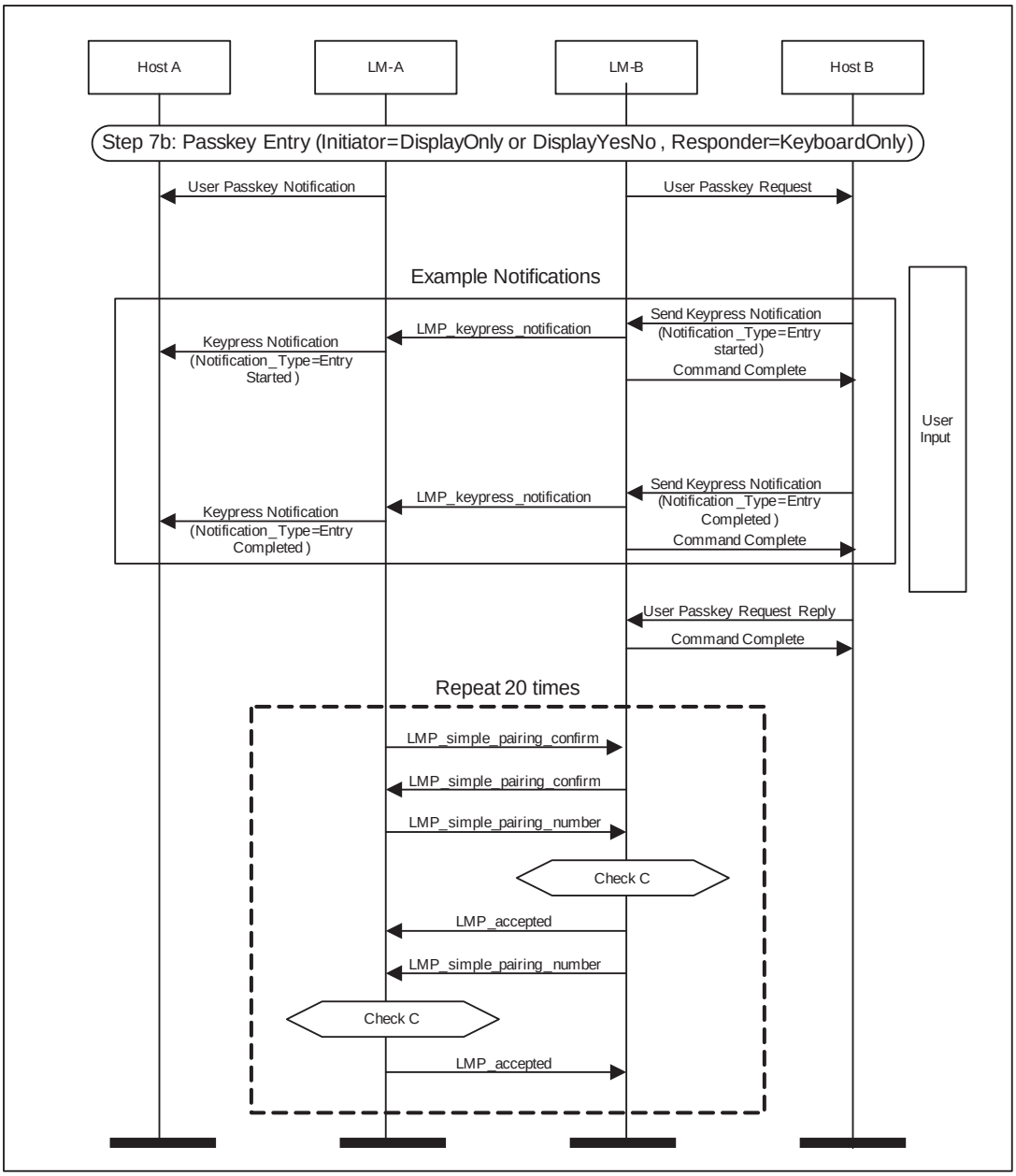


Figure 4.13: Passkey Entry Authentication

4.2.14 Passkey Entry Failure on Responding Side

If the passkey entry fails on the responding side, Simple Pairing is terminated.

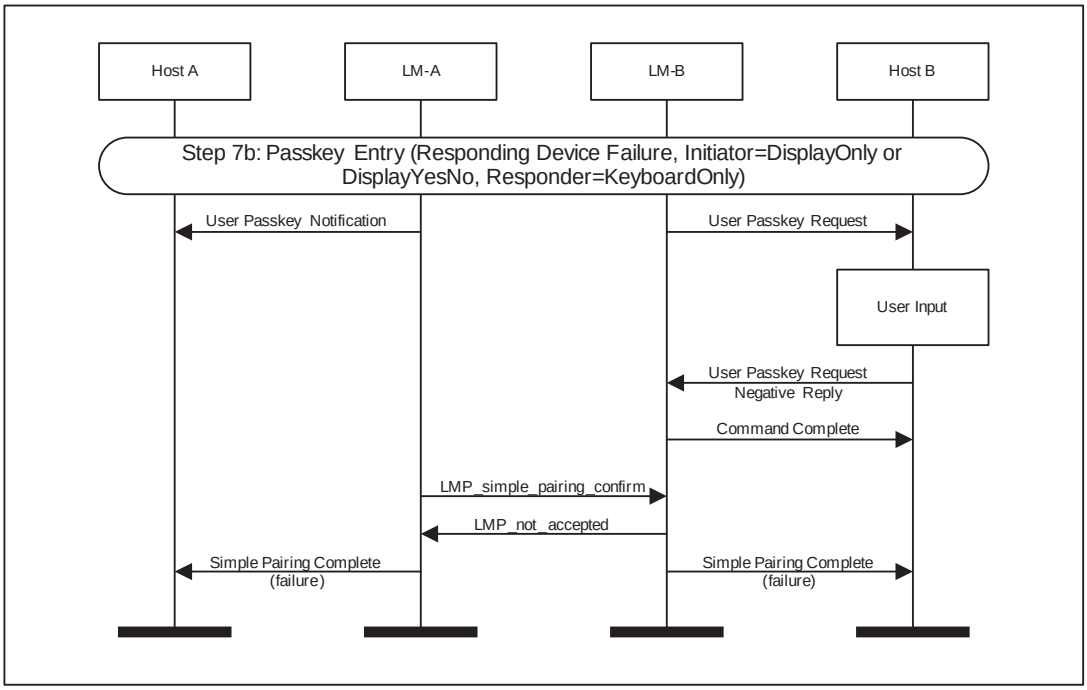


Figure 4.14: Passkey Entry Failure on Responding Side

4.2.15 Passkey Entry Failure on Initiator Side

If the passkey entry fails on the initiating side, Simple Pairing is terminated. Note that this is only possible if the initiating LM side sends an HCI User Passkey Request event.

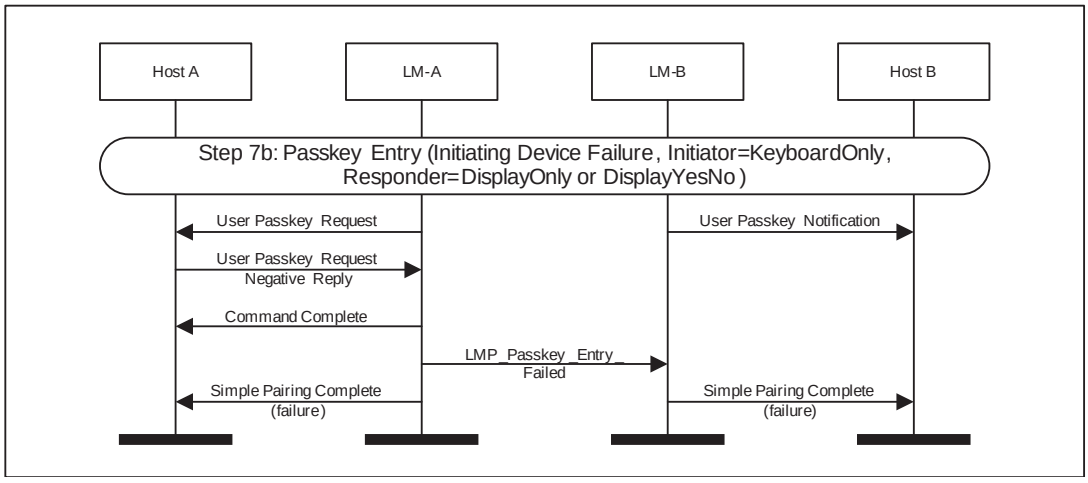


Figure 4.15: Passkey Entry Failure on Initiating Side

4.2.16 Out of Band

The OOB authentication will only be done when both devices have some OOB information to use. This step requires no user interaction.

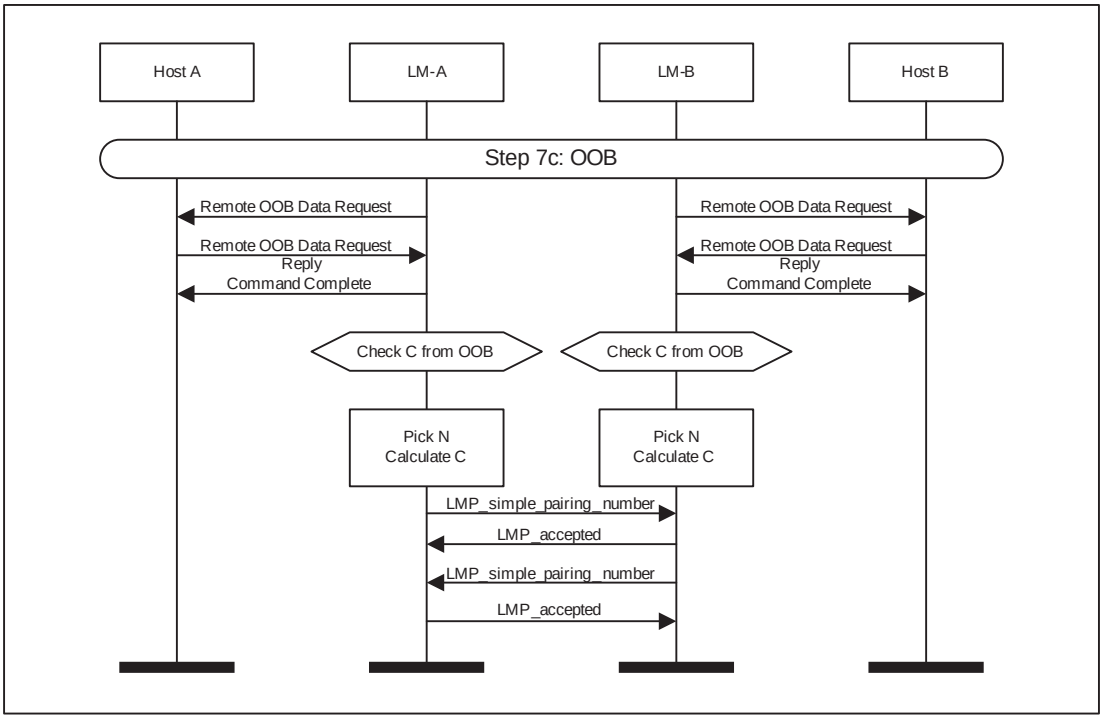


Figure 4.16: OOB Authentication

4.2.17 OOB Failure on Initiator Side

If the initiating side does not have OOB information, Simple Pairing is terminated.

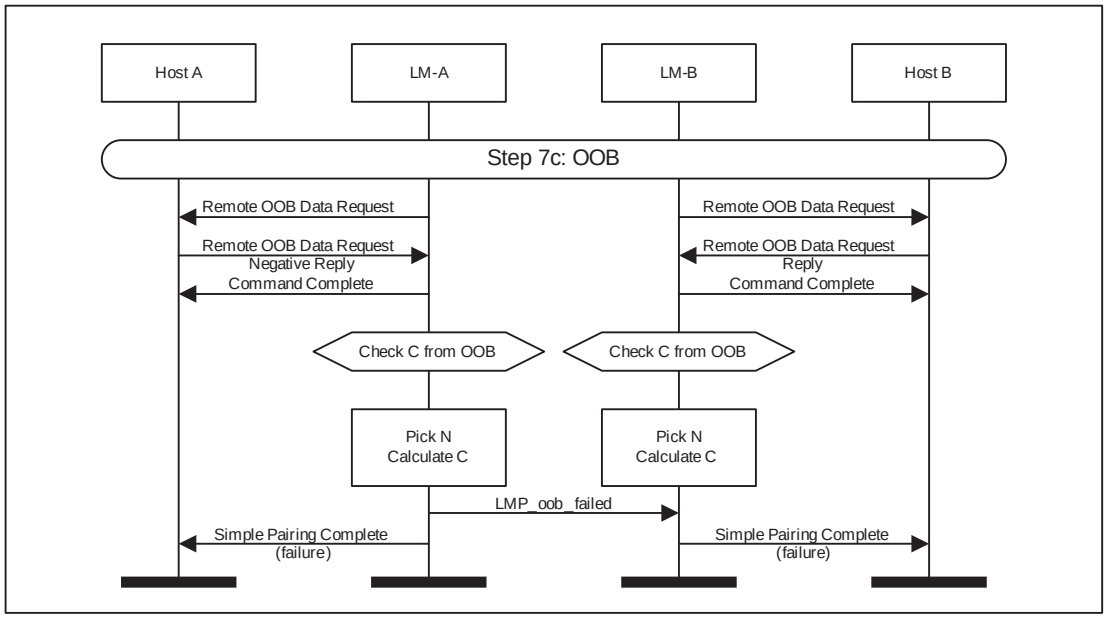


Figure 4.17: OOB Authentication failure on initiating side

4.2.18 DHKey Checks

Once the devices have been authenticated, and the DHKey calculation has completed, the DHKey value generated is checked. If this succeeds, then both devices would have finished displaying information to the user about the process, and therefore a message is sent from the controller to the host to notify it to stop displaying this information.

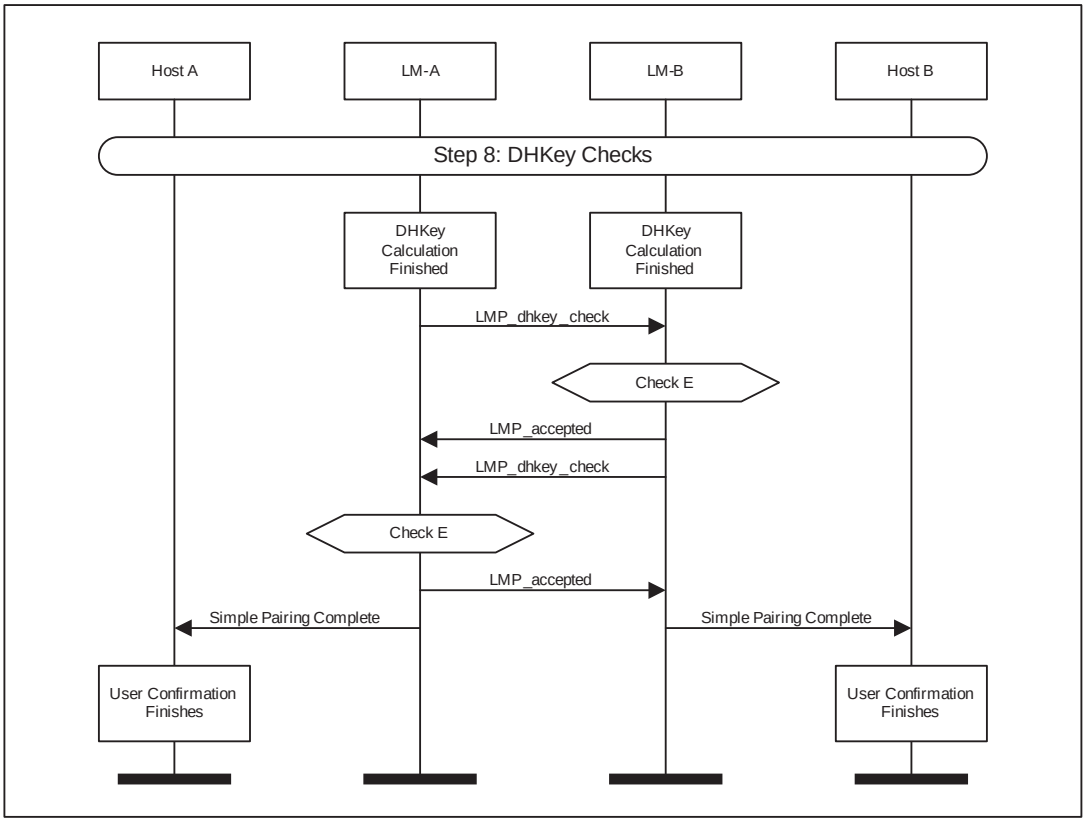


Figure 4.18: DHKey Checks

4.2.19 Calculate Link Key

Once simple pairing is complete, the link key can be calculated from the DHKey, and this should be used as input to a standard mutual authentication. Once this is complete, a Link Key Notification event will be generated.