

MOST

Media Oriented Systems Transport

Multimedia and Control
Networking Technology

MOST Specification

Rev 2.0

12/1999

Version 2.0-01



©

00 MOST Cooperation

Legal Notice

COPYRIGHT

© Copyright 1999 - 2000 MOST Cooperation. All rights reserved.

LICENSE DISCLAIMER

Nothing on any MOST Cooperation Web Site, or in any MOST Cooperation document, shall be construed as conferring any license under any of the MOST Cooperation or its members or any third party's intellectual property rights, whether by estoppel, implication, or otherwise.

CONTENT AND LIABILITY DISCLAIMER

MOST Cooperation or its members shall not be responsible for any errors or omissions contained at any MOST Cooperation Web Site, or in any MOST Cooperation document, and reserves the right to make changes without notice. Accordingly, all MOST Cooperation and third party information is provided "AS IS". In addition, MOST Cooperation or its members are not responsible for the content of any other Web Site linked to any MOST Cooperation Web Site. Links are provided as Internet navigation tools only.

MOST COOPERATION AND ITS MEMBERS DISCLAIM ALL WARRANTIES WITH REGARD TO THE INFORMATION (INCLUDING ANY SOFTWARE) PROVIDED, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to you.

In no event shall MOST Cooperation or its members be liable for any damages whatsoever, and in particular MOST Cooperation or its members shall not be liable for special, indirect, consequential, or incidental damages, or damages for lost profits, loss of revenue, or loss of use, arising out of or related to any MOST Cooperation Web Site, any MOST Cooperation document, or the information contained in it, whether such damages arise in contract, negligence, tort, under statute, in equity, at law or otherwise.

FEEDBACK INFORMATION

Any information provided to MOST Cooperation in connection with any MOST Cooperation Web Site, or any MOST Cooperation document, shall be provided by the submitter and received by MOST Cooperation on a non-confidential basis. MOST Cooperation shall be free to use such information on an unrestricted basis.

TRADEMARKS

MOST Cooperation and its members prohibit the unauthorized use of any of their trademarks. MOST Cooperation specifically prohibits the use of the MOST Cooperation LOGO unless the use is approved by the Steering Committee of MOST Cooperation.

SUPPORT AND FURTHER INFORMATION

For more information on the MOST technology, please contact:

MOST Cooperation

Administration
P. O. Box 4327
D-76028 Karlsruhe
Germany

Tel: (+49) (0) 721 966 50 00

Fax: (+49) (0) 721 966 50 01

E-mail: contact@mostcooperation.com

Web: www.mostcooperation.com



© Copyright 1999 - 2000 MOST Cooperation
All rights reserved

MOST is a registered trademark

Contents

1	INTRODUCTION	11
2	APPLICATION SECTION	12
2.1	Overview of Data Channels.....	12
2.1.1	Control Channel	12
2.1.2	Synchronous Channel.....	12
2.1.3	Asynchronous Channel.....	13
2.1.4	Managing Synch./Async. Bandwidth.....	13
2.2	Logical Device Model	14
2.2.1	Function Block.....	14
2.2.1.1	Slave, Controller, MMI.....	15
2.2.1.2	First Introduction to MOST Functions.....	15
2.2.2	Functions.....	16
2.2.3	Methods.....	16
2.2.4	Properties	17
2.2.4.1	Setting a Property	17
2.2.4.2	Reading a Property	18
2.2.5	Events	18
2.2.6	Function Interfaces.....	19
2.2.7	Definition Example	20
2.2.8	MOST Data Flow Model.....	22
2.2.9	MOST System Services	23
2.2.10	Delegation, Heredity, Device Hierarchy	23
2.2.10.1	Delegation	23
2.2.10.2	Heredity of Functions	25
2.2.10.3	Deriving Devices/Device Hierarchy	26
2.3	Protocols.....	29
2.3.1	Protocol Basics	29
2.3.2	Structure of MOST Protocols	29
2.3.2.1	DeviceID	29
2.3.2.2	FBlockID	30
2.3.2.3	InstID.....	31
2.3.2.4	FktID	32
2.3.2.5	OPType.....	33
2.3.2.5.1	Error	34
2.3.2.5.2	Start, Result, Processing, Error.....	37
2.3.2.5.3	StartResult, Result, Processing, Error.....	37
2.3.2.5.4	StartAck, ProcessingAck, ResultAck, and ErrorAck.....	39
2.3.2.5.5	Get, Status, Error	39
2.3.2.5.6	Set, Status, Error.....	39
2.3.2.5.7	SetGet, Status, Error.....	39
2.3.2.5.8	GetInterface, Interface, Error	40
2.3.2.5.9	Increment And Decrement	40
2.3.2.5.10	Abort	40
2.3.2.6	Length	41
2.3.2.7	Data	41
2.3.3	Function Formats in Documentation	43
2.3.4	Protocol Catalogs.....	43
2.3.5	Application Functions on MOST Network (Introduction).....	44
2.3.6	Controller/Slave Communication.....	47
2.3.6.1	Communication With Properties Using Shadows.....	47
2.3.6.2	Communication With Methods	52
2.3.6.2.1	Standard Case	52
2.3.6.2.2	Special Case Using Routing	53
2.3.7	Seeking Communication Partner	55
2.3.8	Requesting Function Block Information from a Device	55
2.3.9	Requesting Functions from a Function Block	56

2.3.10	Transmitting The Function Interface	57
2.3.10.1	Principle	57
2.3.10.2	Realization Of The Ability To Extract The Function Interface	57
2.3.11	Function Classes	58
2.3.11.1	Properties With A Single Variable	58
2.3.11.1.1	Function Class Switch	60
2.3.11.1.2	Function Class Number	61
2.3.11.1.3	Function Class Text	63
2.3.11.1.4	Function Class Enumeration	64
2.3.11.2	Properties with Multiple Variables	65
2.3.11.2.1	Function Class Record	66
2.3.11.2.2	Function Class Array	68
2.3.11.2.3	Function Class Dynamic Array	71
2.3.11.2.4	Function Class LongArray	73
2.3.11.3	Function Class For Methods	80
2.3.12	Handling Message Notification	81
3	NETWORK SECTION	84
3.1	MOSTTransceiver and its Internal Services	84
3.1.1	Electrical Bypass (All Bypass)	84
3.1.2	Source Data Bypass	84
3.1.3	Master/Slave, Active and Passive Components	84
3.1.4	Data Transport	85
3.1.4.1	Blocks	85
3.1.4.2	Frames	85
3.1.4.2.1	Preamble	87
3.1.4.2.2	Boundary Descriptor	87
3.1.4.2.3	MOST System Control Bits	87
3.1.4.3	Source Data	88
3.1.4.3.1	Definition of Control Data and Source Data	88
3.1.4.3.2	Differentiating Synchronous and Asynchronous Data	88
3.1.4.3.3	Source Data Interface	88
3.1.4.3.4	Transparent Channels	88
3.1.4.3.5	Synchronous Area	89
3.1.4.3.6	Asynchronous (Packet Data) Area	89
3.1.4.4	Control Data	91
3.1.4.4.1	Control Data Interface	91
3.1.4.4.2	Description	91
3.1.5	Internal Services	93
3.1.5.1	Addressing	93
3.1.5.2	Address Initialization (SAI)	93
3.1.5.3	Support at System Startup	94
3.1.5.4	Delay Recognition	94
3.1.5.5	Remote-Access	94
3.1.5.6	Automatic Channel Allocation	94
3.1.5.7	Power Management	95
3.1.5.8	Detection of Unused Channels	95
3.2	Dynamic Behavior of a Device	96
3.2.1	Overview	96
3.2.2	NetInterface	98
3.2.2.1	NetInterfacePowerOff	99
3.2.2.2	NetInterfaceInit	99
3.2.2.3	NetInterfaceNormalOperation	103
3.2.2.4	NetInterface Ring Break Diagnosis	105
3.2.3	Initialization on Application Level	111
3.2.3.1	Network Slave	111
3.2.3.2	Network Master	113
3.2.4	Power Management	117
3.2.4.1	General Procedure	117
3.2.4.2	Functions and Important Operations	120

3.2.5	Error Management	121
3.2.5.1	Handling of Light Off	121
3.2.5.2	Fatal Error	122
3.2.5.2.1	Waking	122
3.2.5.2.2	Operation	122
3.2.5.3	Unlock	123
3.2.5.4	NetworkChange Event	124
3.2.5.5	Low Voltage	124
3.2.5.6	"Hanging" of an Application	126
3.2.5.7	Diagnosis	126
3.3	Accessing Control Channel	127
3.3.1	Addressing	127
3.3.2	Assigning Priority Levels	129
3.3.3	Low Level Retries	129
3.3.4	High Level Retries	129
3.3.5	MOSTNetServices (Application Socket)	130
3.3.5.1	Basics for Automatic Adding of Physical Address	130
3.3.5.2	De-Central Registry	130
3.3.5.3	Central Registry	131
3.3.6	Handling Overload in a Message Sink	134
3.3.7	MOSTNetServices (Basic Layer)	135
3.3.7.1	Control Message Service	135
3.3.7.2	Application Message Service (AMS) And Application Protocols	135
3.3.8	Direct Access to OS8104	137
3.3.8.1	Sending Messages	137
3.3.8.2	Receiving Messages	138
3.3.8.3	Acknowledgement and Data Security	138
3.3.9	Remote Control	139
3.3.9.1	Remote Read Message	139
3.3.9.2	Remote Write Message	140
3.4	Handling Synchronous Data	141
3.4.1	MOSTNetServices (Application Socket)	141
3.4.1.1	Basic Functions on Application Level	142
3.4.1.1.1	NetBlock	142
3.4.1.1.2	Function Block	143
3.4.2	MOSTNetServices (Basic Layer)	149
3.4.3	Direct Access to OS8104	149
3.4.3.1	Serial Interface	149
3.4.3.2	Parallel Interface	149
3.4.3.3	Compensating Network Delay	149
3.5	Handling Asynchronous (Packet) Data	150
3.5.1	Direct Access to OS8104	150
3.5.1.1	Priorities	150
3.5.2	MOSTNetServices	151
3.5.2.1	Securing data	151
3.6	Controlling Synchronous/Asynchronous Bandwidth	153
3.7	Connections	154
3.7.1	Synchronous Connections	154
3.7.1.1	Administering (Connection Master)	154
3.7.1.2	Establishing Synchronous Connections	156
3.7.1.3	Removing Synchronous Connections	158
3.7.1.4	Supervising Synchronous Connections	158
3.8	Timeouts	159
4	HARDWARE SECTION	161
4.1	Basic HW Concept	161

4.2	Optical Interface Area.....	162
4.2.1	Overview	162
4.2.2	Optical Power Budget	164
4.2.3	POF	164
4.2.4	Connection Systems (Pig Tail).....	165
4.3	MOST Function Area.....	166
4.4	µC Area	166
4.5	Application Area	167
4.6	Power Supply Area.....	167
4.7	Voltage Levels	172
5	TOOLS	174
5.1	OptoLyzer4MOST [®]	174
5.2	MOSTRapidControl	175
5.3	OptoLyzer4MOST [®] Professional	176
5.3.1	Introduction.....	176
5.3.2	Architecture	176
5.3.3	OptoLyzer Control	176
5.3.4	Bus Analysis.....	177
5.3.5	MOST Simulation Interface	177
6	APPENDIX A: INDEX OF FIGURES	179
7	APPENDIX B: INDEX OF TABLES	181
8	APPENDIX C: INDEX.....	182

Bibliography

Number	Document
[1]	MOST Specification Framework
[2]	MOST Specification
[3]	MOSTHighProtocol Specification
[4]	MOSTNetServices "Basic Layer"; User Manual and Specification
[5]	MOSTNetServices "Application Socket"; User Manual and Specification
[6]	FOT Datasheet
[7]	MOSTTransceiver Datasheet
[8]	MOSTFunctionCatalog

Document History

Changes MOST Specification 2V0-00 to MOST Specification 2V0-01

Change Ref.	Section	Changes
2V01_001	General	Document no longer specified as "Confidential"; Legal Notice inserted.

Changes MOST Specification 1V0 to MOST Specification 2V0

Change Ref.	Section	Changes
2V0_001	3.3.1	Equation modified; Startup address 0xFFFF
2V0_002	2.1.2/ 2.2.5	Section 2.2.5 moved to 2.2.2
2V0_003	2.2.1	NetBlock "functions related to the entire device."
2V0_004	2.3.2.2	Table 2-5: Proprietary FBlockIDs 0xF0..0xFE
2V0_005	2.3.2.3	Completely revised
2V0_006	2.3.2.4	Minor modification
2V0_007	2.3.2.5	Completely revised
2V0_008	2.3.2.6	Completely revised
2V0_009	2.3.2.7	Bool-field introduced; Definition of STRING expanded, Examples for Exponent, Step and Unit
2V0_010	2.3.5	Minor modification
2V0_011	2.3.6	Distinguishing Properties and Methods; Communication with routing revised
2V0_012	2.3.10	Transmitting function interfaces. Introduced.
2V0_013	2.3.11	Function Classes (completely revised)
2V0_014	2.3.12	Notification for array properties; Notification re-build at system start
2V0_015	3.2.2.2	Error_t_slave replaced by Error_NSInit_Timeout
2V0_016	3.2.2.3	Completely revised
2V0_017	3.2.2.4	Completely revised
2V0_018	3.2.3.1	Completely revised
2V0_019	3.2.3.2	Completely revised
2V0_020	3.2.4.1	Completely revised
2V0_021	3.2.5	General rules added
2V0_022	3.2.5.1	Completely revised
2V0_023	3.2.5	Completely revised
2V0_024	3.3.5.3	- Table 3-15; - sample for receiving logical node address; - section below Table 3-16
2V0_025	3.3.7.2	TellIDs for MOST High Protocol removed
2V0_026	3.3.8.1	Figure 3-22; Set STX bit added
2V0_027	3.4.1.1.1	Replaced ".0." by ".Pos."
2V0_028	3.4.1.1.2	Completely revised

Change Ref.	Section	Changes
2V0_029	3.4.3.3	Equations
2V0_030	3.5.2.1	TelID and TelLen changed; One ID reserved for Ethernet frames
2V0_031	3.7.1.1	Revised (OPTypes)
2V0_032	3.7.1.2	Revised (OPTypes)
2V0_033	3.7.1.3	Revised (OPTypes)
2V0_034	3.1.4.2	Table 3-1
2V0_035	3.1.4.2.2	Revised
2V0_036	3.1.4.3.1	Handling of Isochronous data removed
2V0_037	3.1.4.3.6	Table 3-2; Table 3-3 added, Handling of Isochronous data removed
2V0_038	3.1.4.4.2	Completely revised
2V0_039	4.1	Figure 4-1
2V0_040	4.2.1	Completely revised
2V0_041	4.2.2	Revised
2V0_042	4.2.4	Completely revised
2V0_043	4.3	Revised
2V0_044	4.5	Completely revised
2V0_045	4.6	Completely revised
2V0_046	4.7	Completely revised
2V0_047	---	General changes in Structure: - Chapter 2.1 removed, contents included within 2.2.9 - Detailed descriptions of Control Channel (2.2) moved to 3.3 - Introduction of CMS/ AMS moved to 3.3.7 - Chapters 2.6 up to 2.12 moved to 3.2 up to 3.8 - Chapter 2.5 and 2.13 moved to Chapter 5

3 Network Section

3.1 MOSTTransceiver and its Internal Services

The MOSTTransceiver OS8104 of Oasis SiliconSystems provides extensive tools for operating the MOST bus simply and safely, and for the transmission of data of different origins. Based on these tools, higher layers are defined. For a more detailed description of the MOSTTransceiver please refer to [7]. The following sections give an overview of the features of the OS8104 that are available for simplifying the definition of higher layers.

3.1.1 Electrical Bypass (All Bypass)

If the bypass is closed, all signals received at the RX pin of the MOSTTransceiver are electrically connected to the RX pin (shortcut). In this state, the respective device is "invisible" to the network. The device will be considered for the automatic counting of bus components only after opening the bypass, which gives access to the bus.

After the MOSTTransceiver is reset, the bypass is closed. This allows a very fast startup of the system, especially on usage of an optical wakeup mechanism. The bypass must be opened by the controlling micro controller after wakeup of the component.

3.1.2 Source Data Bypass

In order to put data on the MOSTNetwork, the source data bypass must be opened in the device. That means that the data is no longer passed through the chip without being processed (that is, not handled by the routing engine RE), but can be routed now, e.g., from a source data port to the bus.

Based on the internal processing of data, a delay of two samples is added in the signal path. The source data bypass should be opened only in devices that put source data onto the bus on runtime.

3.1.3 Master/Slave, Active and Passive Components

Basically, a MOST system consists of up to 64 nodes with identical MOSTTransceivers. By configuration, any of the transceivers can be the timing master, all the others are slaves. The timing master provides generation and transporting of system clock, the frames, and blocks. All slave devices derive their clock from the MOST bus.

The timing master, as well as active slave devices (source data bypass is open, device can put source data on the bus) add two samples of delay to the path of source data.

A passive slave device has a closed source data bypass. Since in that case the routing engine is inactive, no delay is generated.

3.1.4 Data Transport

The bit stream is optimized in such a way that processing is easy and maximum functionality is supported. This includes mechanisms for automatic channel routing, network delay detection, and burst data channel management.

The MOST network technology defines an intelligent bit stream which is capable of providing all MOSTNetwork features as described above.

Data is transferred in a continuous bi-phase encoded bit stream yielding more than a 24.8Mbps data rate at a 44.1 kHz rate and a bit error rate of less than 10^{-10} .

Since the MOST system is fully synchronous, with all devices connected to the bus being synchronized to the bus, no memory buffering is needed (unlike isochronous, or asynchronous devices). This keeps cost low.

The sample frequency in a MOST system can be chosen in a range between 30kHz and 50kHz. The frequency depends directly on the application components. Some devices, for example, CD drives, work at a device-specific sample rate. In systems optimized for cost, such devices are regarded as fixed with respect to sample frequency. The sample frequency that is used most should be defined as the system frequency, to avoid sample rate conversion in the different devices.

3.1.4.1 Blocks

Organization of data transfer in blocks of frames is required for network management and control data transport tasks.

A block consists of 16 frames with 512 bits each. Per frame, 60 bytes of data are available for source data (synchronous and asynchronous packet data), while two bytes transport control data. The 2 bytes of 16 frames (1 block) are added to the control frame that transports a control telegram.

3.1.4.2 Frames

The MOST frame structure is designed in a way that provides maximum flexibility in terms of compatibility with a number of existing communication and data transport requirements without any drawbacks in implementation cost or processing overhead. It allows easy re-synchronization, clock and data recovery with the highest data quality and integrity. Built-in structures allow simple network management on the lowest layers avoiding overhead and cost shortcomings.

For synchronization, two different bus node types are required. A timing master that generates the frames, and slave devices that synchronize to the master clock on the bus.

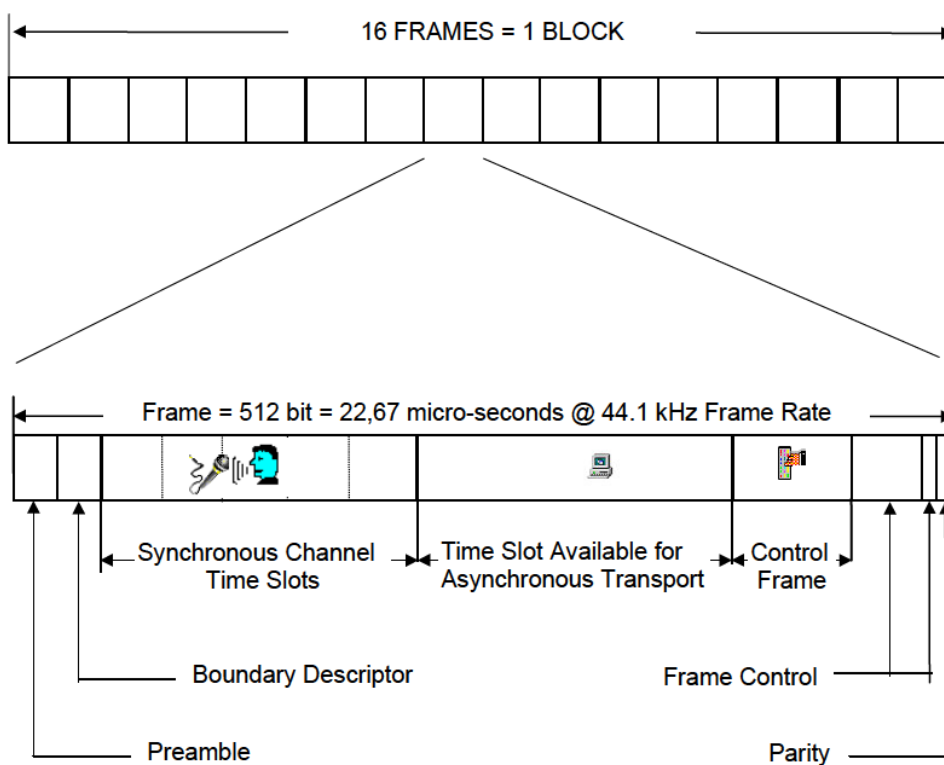


Figure 3-1: Structure of blocks and frames on the MOST bus.

The 64 bytes (512 bits) wide frame has the following structure:

Byte	Bit	Task
0	0-3	preamble
0	4-7	boundary descriptor (synchronous area count value)
1	8-15	data byte 0
2	16-23	data byte 1
...	...	...
60	480-487	data byte 59
61	488-495	control frame byte 0
62	496-503	control frame byte 1
63	504-510	frame control and status bits
63	511	parity bit

Table 3-1: Structure of the MOST frame

3.1.4.2.1 Preamble

The preambles are used internally to synchronize the MOST core and its on-chip functions to the bit stream.

For synchronization to a frame, two different mechanisms are used for slave and master nodes. For a slave node, the first reception of valid preambles after reset, power-up, or loss of lock indicates that phase lock on the input bit stream has been accomplished.

This method ensures that the slave node is phase- and frequency-locked to the bit stream, and hence the master node. In a master node, the transmitted bit stream is synchronized to an external timing source such as a crystal oscillator, SCK, FSX, or S/PDIF source.

Once all the nodes in the network have locked to the master's transmitted bit stream, the received bit stream has the correct frequency, but will be phase shifted with respect to the transmitted bit stream. This phase shift is due to delays from each active node, and additional accumulated delays due to tolerances in the phase lock within the slave nodes. The master node re-synchronizes the received data by the use of a PLL to lock onto the incoming bit stream, thereby re-synchronizing the incoming data to the proper bit alignment.

3.1.4.2.2 Boundary Descriptor

The boundary descriptor provides a flexible way of changing the bandwidth for synchronous and asynchronous data transmission. It represents the number of 4 byte blocks (quadlets) of data used for synchronous data. This value is used to determine the boundary between the synchronous and asynchronous data areas in the frame. A count value of 0 indicates no synchronous data and 15 quadlets of asynchronous data, while a count value of 15 indicates 15 quadlets of synchronous data and no asynchronous data.

By this means, a 60 byte data field can be allocated to either synchronous or asynchronous data on a 4 byte resolution. As such, it can be optimized to different requirements, depending on the amount of bandwidth required for each type of data.

Note that the maximum number of asynchronous data bytes per frame is 36 bytes, which means that the boundary descriptor values can be between 6 and 15.

The boundary descriptor is managed by the timing master of a MOSTNetwork. Please note that all synchronous connections must be re-built after having changed bSBC.

3.1.4.2.3 MOST System Control Bits

All other bits within the frame are for management purposes on the network level. While the preamble provides synchronization and clock regeneration, the parity bit indicates reliable data content and is used for error detection and phase lock loop operation.

3.1.4.3 Source Data

3.1.4.3.1 Definition of Control Data and Source Data

Depending on the kind of data and bandwidth, the MOST system provides different transmission procedures.

Telegrams for controlling devices or slow asynchronous data are transmitted via the control channel of the MOST_{Transceiver}. For transmitting asynchronous data of higher bandwidth, a packet-oriented asynchronous data area is available. Synchronous data, such as audio signals of a CD drive, can be transmitted directly in the synchronous data area of the network. A more detailed description of the different data areas can be found in the sections below.

3.1.4.3.2 Differentiating Synchronous and Asynchronous Data

Sixty data bytes (15 quadlets) total are available for synchronous and asynchronous (packet) data. The number of synchronous and asynchronous bytes is specified by the boundary descriptor value described above.

3.1.4.3.3 Source Data Interface

The MOST_{Transceiver} can handle a variety of different data formats at its source data port. The source data port formats are controlled via the internal registers of the transceiver. For more detailed information please refer to [7].

3.1.4.3.4 Transparent Channels

In addition to the different transmission procedures, the MOST_{Network} provides a transparent interface (transparent port). This port is oversampled (depending on the system's sample rate, and on the sampling rate chosen for the transparent port) and routed via the network. Therefore source data port 1 (SR1 and SX1) is available. It provides, for example, the transparent transmission of a RS232 interface, i.e., without synchronizing RS232 to the bus.

If no transparent channel is required, source data port 1 can be used as a standard source data port.

3.1.4.3.5 Synchronous Area

The synchronous channel time slots are available for real-time data such as audio/video or sensors and eliminate the need for additional buffering in analog-to-digital converters (and digital-to-analog converters) or in single speed CD devices for audio and video.

Accessing this data is provided by time division multiplexing (TDM) and allocation of quasi-static physical channels for a certain period of time (e.g., while playing an audio source). The bandwidth for such a channel can be adjusted by allocating any number of bytes to one logical channel. The maximum number of bytes available in a synchronous channel is 60 bytes/frame, which is corresponding to 60 x 8 bits or 15 stereo channels of CD-quality audio. The typical frame rate is 44,100 frames/second.

The routing engine (RE) is used to route data to and from the appropriate sources or sinks within a node. Internal synchronization is provided so input data does not need to be phase-aligned to the transceiver. The RE provides full flexibility in directing data from any source to any sink just by setting the appropriate value in the corresponding registers.

3.1.4.3.6 Asynchronous (Packet Data) Area

Another time slot is available for asynchronous data transport as required for more packet-oriented, burst-like data. In contrast to the control data channel, the asynchronous data channel provides transmission of longer data packets.

Access to this type of data is provided in a token ring manner. Each node has fair access to this channel and its bandwidth can be controlled using the boundary descriptor in a step of four bytes (quadlets). The maximum packet length on an asynchronous channel when using the 48 bytes data link layer, is 48 bytes. In case of using an alternative data link layer, the maximum packet length is 1014. The data on this channel is CRC protected. The asynchronous message is defined as follows:

Byte	Task
0	Arbitration
1-2	Target address
3	Length (in Quadlets = 4 Bytes)
4-5	Own address (Source address)
6-53	Data area
54-57	CRC

Table 3-2: Structure of a frame in the asynchronous area (48 bytes data link layer).

Byte	Task
0	Arbitration
1-2	Target address
3	Length (in Quadlets = 4 Bytes)
4-5	Own address (Source address)
6-1019	Data area
1020-1023	CRC

Table 3-3: Structure of a frame in the asynchronous area (alternative data link layer).

Since the asynchronous data area is variable, it can take several frames to complete a message. The corresponding management such as arbitration and channel allocation is provided by the chip. A hardware CRC is provided. The CRC is calculated in the background and can be indicated in a register at the end of each asynchronous message. A low level retry mechanism is not implemented.

3.1.4.4 Control Data

3.1.4.4.1 Control Data Interface

The transmission of control data to and from the MOSTTransceiver is done via the control bus (at the control port). Depending on the application, this is either an I²C bus, an SPI interface, or the parallel interface of the chip.

3.1.4.4.2 Description

The control data is used mainly for communication between the single nodes of the bus. This is where commands, status and diagnosis messages, as well as gateway messages are handled. The protocol on this channel runs in a carrier sense multiple access (CSMA) manner offering predictable response times which are considered essential in an audio/video control network. At a system sample frequency of 44.1 kHz, 2756 messages per second are transmitted, which corresponds to a gross data rate of 705.6 kBit/s.

Since 2 out of 64 messages are used for a system wide distributing of the allocation information by the network, the number of messages per second available for control messaging is 2670. When subtracting the data used for control and data securing, the net data rate (user data plus addressing) is 405.84 kBit/s, which corresponds to 19 bytes per message that can be read from the MOSTTransceiver.

A MOST Device can access every third message propagated through the network. So a single device has a maximum message rate of 890 per second, or a net data rate of 135.28kBit/s

There are two kinds of control messages. Normal messages provide control of applications, while system messages handle system-related operations such as resource handling or remote access. During remote access, additional handshaking guarantees reliable remote operation. A control data message is 32 bytes long and has the following structure:

Byte	Task
0-3	Arbitration
4-5	Target address
6-7	Own address (Source address)
8-25	Data area+1 byte message type
26-27	CRC
28-29	Transmission status
30-31	Reserved

Table 3-4: Structure of a control data frame.

Please note:

The contents of register bXTIM – which controls the delay time between two messages in case of low level retries – must be identical in all nodes of a MOST Network.

One complete message is buffered by the chip. Any important status changes can be flagged by interrupt (if desired). The data on the control channel is protected by CRC and acknowledge mechanisms.

Arbitration is provided automatically by the transceiver in case a node wants to send a message. In order to provide fair arbitration even at high bus loads, a double arbitration mechanism is used. This ensures that an access is not depending on the communication load of upstream devices and the priority is not depending on the network position. Rejection of messages is flagged and automatic retransmission is performed. The number of retries can be defined by the application software register bXRTY. If the maximum of retries is reached without success, a transmission error is indicated to the controlling device (e.g., external Microcontroller). The delay time between the sending out of a message (in case of low level retries only) is specified in register bXTIM. The contents of bXTIM must be equal in every node of a MOST Network.

In standalone mode, the MOST control messages can be used to indicate the occurrence of an interrupt.

3.1.5 Internal Services

3.1.5.1 Addressing

The MOST_{Transceiver} supports four different ways of addressing:

- Node position in the ring.
The node position is generated automatically in each node during the locking procedure of the MOST_{Network}.
- Unique node address (2 bytes).
This address can be set by the application. The SAI procedure helps to test on the chip level whether a desired address is unique or not.
- Group address (1 byte).
Group address can be set by the application. A group is made up of devices that have the same number in the group address register.
- Broadcast
The broadcast address is a special group address. When used, the message is received by all nodes in the ring. Until the last node in the ring has acknowledged a broadcast message, communication via the control channel is suppressed for other messages.

The different ways of addressing are mapped into the address area of a MOST_{Transceiver}:

Address range	Mode
0x0001..0x02FF 0x0500..0xFFFF	Normal addressing (Point to point) based on unique node address.
0x0400..0x04FF	Node position addressing: Address = 0x400 + Node position of target node
0x0300..0x03C7 0x03C9..0x03FF	Group addressed: Address = 0x300 + Address of desired group Group address 0xC8 reserved for broadcast
0x03C8	Broadcast addressing

Table 3-5: Addressing modes vs. address range.

Group addressing is typically used for controlling several devices of the same type (e.g., active speakers). The grouping of devices must be established during definition of the system.

3.1.5.2 Address Initialization (SAI)

The MOST_{Transceiver} supports a procedure on the chip level that can be used to verify whether an address is unique or not. The desired address must be written as a target address to the Xmit control message buffer (bytes XTAH and XTAL), and then the bit SAI in the message control register must be set to '1'. Now the transceiver checks whether the address is unique or not. If it is ok, then the address is immediately written to the node address registers of the transceiver. The result of the operation is written to bit TXR in the message status register, with a value of '1' meaning that the address was ok.

3.1.5.3 Support at System Startup

The MOSTTransceiver meets all requirements of a low level startup. It is so far able to run in standalone mode, as it provides a communication-ready network to all applications. In addition, several supporting mechanisms are provided. All components of the system get a unique number, with numbering starting at the timing master at 0x00, and then incremented by one. These numbers can be used for node position addressing. Furthermore, every device receives the information about the total number of devices in the ring. The MOSTTransceiver also provides a wakeup mechanism.

3.1.5.4 Delay Recognition

Based on the fact that every node may be active or passive with respect to source data handling (source data bypass open or closed), and that every active node generates two samples of delay, it is useful to have information about source data delay, for example, for noise compensation applications, or in high-end audio applications.

Therefore a mechanism is implemented in each MOSTTransceiver, giving access to information about the total delay of the system, and to the delay up to the local node with respect to the timing master.

3.1.5.5 Remote-Access

This feature provides a remote-controllable I²C bus in each MOSTTransceiver that runs in standalone mode. It can be controlled from any node in the network, by using certain system commands. Furthermore, all registers on page 0x0 of the MOSTTransceiver can be read or written via the network.

3.1.5.6 Automatic Channel Allocation

Since administration of up to 30 audio channels would need many resources on an application's side, the MOST system supports resource administration on chip level.

Allocating one or more audio channels (up to 64 bits per allocation procedure) is done via a request from an application to the timing master of the network. If there are enough channels, the application will get a handle, by which source data can be routed onto the network. The handle can also be used for de-allocating. A channel resource allocation table (mCRA), distributed automatically in the ring (on chip level), gives access to the current allocation status of the channels in each node.

The channel map that belongs to the handle can be retrieved from mCRA, or it can be delivered during connection management via control messages. The entire allocation takes about 25ms (maximum), making it possible to change allocation during runtime.

3.1.5.7 Power Management

The MOSTTransceiver has three power states:

- Normal operation mode
- Low power mode
- Zero power mode

In normal operation mode, all sections of the chip are running and the chip is fully accessible.

Low power mode is used to lower the power consumption of devices that are not in use at the moment. All sections of the chip that handle source data are switched off, source data bypass is active. The chip is visible from the network's point of view, but no communication is possible. Based on the timing master, all components that are in low power mode can be awakened by a wakeup-preamble.

On zero power mode, all sections of the chip are deactivated, except a small wakeup logic. The wakeup logic activates the receiving FOT unit by a special pin of the chip in regular intervals and scans for modulated light. If there is modulated light, the chip wakes up.

When using an FOT unit with wakeup feature, the zero power mode of the chip does not need to be used. The chip can be connected to the switched branch of the power supply.

3.1.5.8 Detection of Unused Channels

Detection of unused channels, i.e., channels that are allocated by a device, but which are no longer used, is done with the help of the allocation table mCRA. By checking the most significant bit, it can be determined whether a channel is in use or not. If there are unused but allocated channels, they should be de-allocated with respect to the network resources.