



PayPass – M/Chip 4

Card Technical Specification

Version 1.3.1 - September 2008

Proprietary Rights

The information contained in this document is proprietary and confidential to MasterCard International Incorporated, one or more of its affiliated entities (collectively "MasterCard"), or both.

This material may not be duplicated, published, or disclosed, in whole or in part, without the prior written permission of MasterCard.

Trademarks

Trademark notices and symbols used in this manual reflect the registration status of MasterCard trademarks in the United States. Please consult with the Customer Operations Services team or the MasterCard Law Department for the registration status of particular product, program, or service names outside the United States.

All third-party product and service names are trademarks or registered trademarks of their respective owners.

Media

This document is available in both electronic and printed format.

Address

MasterCard Worldwide
2200 MasterCard Boulevard
O'Fallon MO 63368-7263
USA

www.mastercard.com

Table of Contents

Table of Contents	iii
Using this Manual	v
Scope.....	v
Audience.....	v
Related Publications	vi
Abbreviations	vi
Notational Conventions	vii
Transition Flow Diagrams	viii
1 PPSE Application.....	9
1.1 Introduction	9
1.2 Application State Machine	10
1.3 Command Processing.....	11
1.3.1 C-APDU Recognition.....	11
1.3.2 C-APDU Acceptance	11
1.3.3 Select PPSE	12
1.3.4 Loop Back	14
2 PayPass – M/Chip 4 Application.....	17
2.1 Introduction	17
2.1.1 Overview	17
2.1.2 Assumptions	17
2.1.3 Data Objects	18
2.1.4 Offline Counters	18
2.1.5 Log of Transactions.....	18
2.2 Application States.....	19
2.2.1 Overview	19
2.2.2 Initiation Commands	21
2.3 C-APDU Pre-Processing	22
2.3.1 C-APDU Recognition.....	22
2.3.2 C-APDU Acceptance	23
2.3.3 Rejected C-APDU Processing.....	24
2.4 C-APDU Processing	25
2.4.1 Overview	25

Table of Contents

2.4.2	Instance of Data Objects Used	26
2.4.3	Compute Cryptographic Checksum	26
2.4.4	Get Data.....	29
2.4.5	Put Data	30
2.5	Key Management and Cryptographic Algorithms	31
2.5.1	Dynamic CVC3	31
2.5.2	Symmetric Key Management.....	32
2.6	Data Objects Location	33
2.6.1	Transient Data Objects that Span a Single C-APDU Processing	33
2.6.2	Additional Persistent Data Objects.....	33
2.6.3	Secret Keys.....	34
2.7	Personalization	35
2.7.1	Compute Cryptographic Checksum Data Objects.....	35
2.7.2	Persistent Data Referenced in the AFL (<i>PayPass</i>).....	36
2.7.3	Persistent Data Objects for Card Risk Management	36
2.7.4	Triple DES Key	36
2.7.5	Persistent Data Objects for GPO Response.....	37
Annex A	Data Objects Dictionary	39
A.1	AFL (<i>PayPass</i>)	39
A.2	AIP (<i>PayPass</i>).....	39
A.3	Application Control (<i>PayPass</i>)	40
A.4	Card Issuer Action Codes (<i>PayPass</i>) – Decline, Default, Online	40
A.5	CVC3 _{TRACK1}	41
A.6	CVC3 _{TRACK2}	41
A.7	IVCVC3 _{TRACK1}	41
A.8	IVCVC3 _{TRACK2}	41
A.9	KD _{CVC3}	41
A.10	Static CVC3 _{TRACK1}	42
A.11	Static CVC3 _{TRACK2}	42
Annex B	ICC Derived Key for CVC3 Generation (KD_{CVC3}).....	43
Annex C	PayPass Data Groupings.....	45

Using this Manual

This chapter contains information that helps you understand and use this document.

Scope

MasterCard *PayPass*™ technology enables fast, easy and globally accepted payments through the use of contactless chip technology. The *PayPass* – M/Chip 4 application is designed specifically for authorization networks that currently support chip card authorizations for credit or debit applications.

In conjunction with the *M/Chip 4 Card Application Specifications for Credit and Debit*, this document provides the specification of the implementation of the *PayPass* – M/Chip 4 application on a dual interface card (contact and contactless) with support for the PPSE. This document does not intend to include or exclude any particular platform.

This specification views support of the PPSE as separate and distinct from that of the *PayPass* – M/Chip 4 application. This is reflected in the document structure. Chapter 1 is concerned solely with the description of the PPSE application. The *PayPass* – M/Chip 4 application is described in Chapter 2.

Audience

This document is intended for use by vendors that want to implement the MasterCard *PayPass* – M/Chip 4 application on a card or other acceptance device.

This document is also intended for type approval services that would test the actual implementations against this specification.

It is assumed that the audience already has an understanding of chip card technology in general and of M/Chip 4 and ISO/IEC 14443 in particular.

Related Publications

The following publications contain information that is related to the contents of this manual.

[M/CHIP4]	<i>M/Chip 4 Card Application Specifications for Credit and Debit v1.0</i>
	<i>M/Chip 4 Card Application Specifications for Credit and Debit v1.1</i>
[M/CHIP4 CPS]	<i>M/Chip 4 Common Personalization Specifications</i>
[PP-MC-INT]	<i>PayPass – M/Chip Reader Card Application Interface Specification v2.0</i>

Abbreviations

The following abbreviations are used in this specification:

Abbreviation	Description
AC	Application Cryptogram
AEF	Application Elementary File
AFL	Application File Locator
AID	Application Identifier
AIP	Application Interchange Profile
an	Alphanumeric
ans	Alphanumeric Special
APDU	Application Protocol Data Unit
ARQC	Authorization Request Cryptogram
ATC	Application Transaction Counter
b	Binary
C-APDU	Command APDU
CDOL	Card Risk Management Data Object List
CLA	Class byte of command message
CVC	Card Verification Code
DF	Dedicated File
DES	Data Encryption Standard
DGI	Data Grouping Identifier
EMV	Europay MasterCard Visa
ICC	Integrated Circuit Card
INS	Instruction byte of command message
ISO	International Organization for Standardization
Lc	Number of bytes present in the data field of the C-APDU
Le	Maximum length of bytes expected in the data field of the R-APDU.

Abbreviation	Description
MAC	Message Authentication Code
MF	Master File
n	Numeric
NATC _{TRACK1}	Track 1 Number of ATC Digits
NATC _{TRACK2}	Track 2 Number of ATC Digits
PAN	Primary Account Number
PCVC3 _{TRACK1}	Track 1 Bit Map for CVC3
PCVC3 _{TRACK2}	Track 2 Bit Map for CVC3
PDOL	Processing Options Data Object List
PPSE	Proximity Payment System Environment
PUNATC _{TRACK1}	Track 1 Bit Map for UN and ATC
PUNATC _{TRACK2}	Track 2 Bit Map for UN and ATC
P1	Parameter 1
P2	Parameter 2
R-APDU	Response APDU
RFU	Reserved for Future Use
SW1	Status Byte One
SW2	Status Byte Two
TC	Transaction Certificate
TLV	Tag Length Value
UDOL	Unpredictable Number Data Object List

Notational Conventions

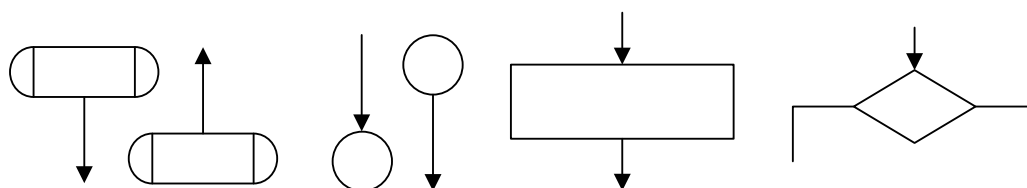
The following notations apply:

Notation	Description
'0' to '9' and 'A' to 'F'	Hexadecimal notation. Values expressed in hexadecimal form are enclosed in single quotes (i.e. '_').
1001b	Binary notation. Values expressed in binary form are followed by a lower case "b".
"abcd"	an or ans string
digit	Any of the ten Arabic numerals from 0 to 9.
[...]	Optional part.
xx	Any value.
A := B	A is assigned the value of B.
C := (A B)	The concatenation of an n-bit number A and an m bit number B, which is defined as $C = 2^m A + B$.
Y := ALG(K)[X]	Encipherment of a 64-bit data block X with a 64-bit block cipher using a secret key K.

Notation	Description
<i>Application Control</i>	Data objects used for this specification are written in <i>italics</i> to distinguish them from the text.
GENERATE AC	Command APDUs used for this specification are written in SMALL CAPITALS to distinguish them from the text.

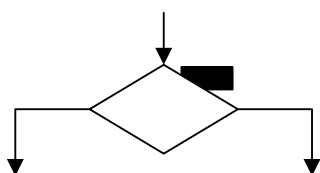
Transition Flow Diagrams

The following symbols are used in the transition flow diagrams:



In most cases a textual description accompanies the transition flow diagram. In this case the symbols in the transition flow diagram are identified with a symbol number. When a paragraph in the textual description starts with '**Symbol n**', then it corresponds to the symbol bearing the same number in the transition flow diagram. The following example illustrates how it works.

- The decision symbol is used in a flow diagram, identified with number 2.



- An explanation of the check done in symbol 2 is given:

Symbol 2

An explanation of how the application checks that the condition is satisfied.

1 *PPSE Application*

1.1 Introduction

This chapter specifies the behavior of the card for the selection of the PPSE. Support for the PPSE is mandatory for all *PayPass* cards.

The SELECT PPSE command processing is independent of the actual application(s) implemented on the card. The PPSE may be implemented as a separate application on a multi-application platform or may be mapped on a DF – which may or may not be the MF – of an ISO 7816-4 compatible file structure.

In addition to the directory function, the PPSE application provides support for loop-back functionality. Loop-back functionality is implemented by the LOOP BACK C-APDU. Upon receiving a LOOP BACK C-APDU the PPSE application returns without any further action the content of the data field of the C-APDU in the data field of the R-APDU. Loop-back functionality is used during the contactless communication protocol testing of the *PayPass* card.

1.2 Application State Machine

The behavior of the PPSE application is specified by its state machine. The application states used in this description are given in Table 1.1.

Table 1.1—Application States of the PPSE Application

State	Description
IDLE	Application is not currently selected
SELECTED	Application is selected

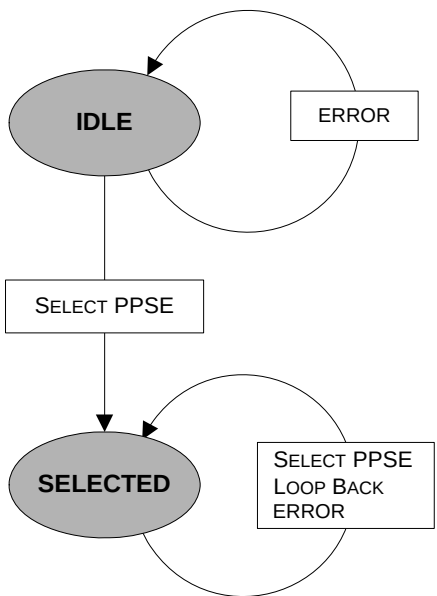
The PPSE application is in state IDLE if it is not currently activated. There is only one C-APDU which is handled in this state: the SELECT PPSE C-APDU, which activates the application.

Upon successfully processing of the SELECT PPSE C-APDU, the PPSE application goes to the state SELECTED. The PPSE application remains in the state SELECTED until the PPSE application is de-selected (i.e. another application is selected or the card is powered-off).

The PPSE application does not change state when an error occurs. An error means a command response with status bytes different from '9000'.

Figure 1.1 shows the state machine of the PPSE application.

Figure 1.1—State Machine of PPSE Application



1.3 Command Processing

This section specifies the command processing for the PPSE application.

1.3.1 C-APDU Recognition

C-APDU recognition is the procedure that identifies the C-APDU transmitted by the *PayPass* reader to the PPSE application. The recognition is based on the CLA and INS byte. Table 1.2 specifies the CLA and INS coding for the PPSE application.

Table 1.2—C-APDU Recognition of the PPSE Application

CLA	INS	C-APDU
'00'	'A4'	SELECT PPSE
'80'	'EE'	LOOP BACK

If the CLA and INS byte combination of the C-APDU is not one of those listed in Table 1.2, then the C-APDU recognition procedure returns status bytes '6E00' or '6D00' and the PPSE application remains in its current state.

1.3.2 C-APDU Acceptance

C-APDU acceptance is specified as the procedure that accepts or rejects the C-APDU, depending on the application state. Acceptance or rejection of a C-APDU by the PPSE application is specified in Table 1.3.

Table 1.3—Acceptance Matrix of the PPSE Application

	IDLE	SELECTED
SELECT PPSE	Accept	Accept
LOOP BACK	Reject	Accept

In the IDLE state, the LOOP BACK C-APDU is not passed to the PPSE application, but is handled by the multi-application manager (refer to [M/CHIP4] for more information about the multi-application manager) or operating system. In this case, the LOOP BACK command should be rejected. Native cards that map the PPSE on the MF file, may however accept the LOOP BACK command without first selecting the PPSE. If the LOOP BACK command is rejected in the IDLE state, then the value of the status bytes is left to the implementation.

If the C-APDU is accepted in the current application state, then the C-APDU is processed as specified in the section dedicated to the C-APDU.

1.3.3 Select PPSE

1.3.3.1 Command Message

The SELECT command message for the PPSE is coded as defined in Table 1.4.

Table 1.4—Select PPSE Command Message

Code	Value
CLA	'00'
INS	'A4'
P1	'04'
P2	'00'
Lc	'0E'
Data	'32 50 41 59 2E 53 59 53 2E 44 44 46 30 31'
Le	'00'

The data field of the command message contains the PPSE directory name ("2PAY.SYS.DDF01").

1.3.3.2 Response Message

The data field of the response message contains the *FCI* and is coded as specified in [PP-MC-INT]. For the PPSE application, the *FCI* is not interpreted by the application: it is data set at personalization and returned in the response to the SELECT PPSE.

1.3.3.3 Processing

Figure 1.2 specifies the processing of the SELECT PPSE command.

Symbol 0

If $P1 \neq '04'$ and $P2 \neq '00'$, then the C-APDU is rejected (SW1-SW2 = '6A86').

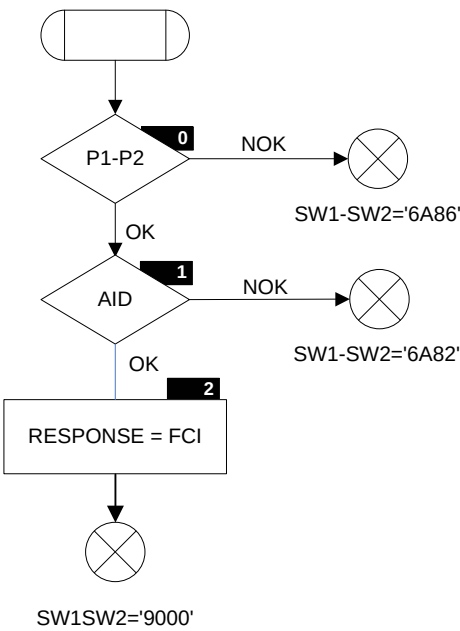
Symbol 1

If the *AID* in the command message data field is different from the PPSE directory name ("2PAY.SYS.DDF01"), then the C-APDU is rejected (SW1-SW2 = '6A82').

Symbol 2

The PPSE application builds the response message template containing the *FCI*.

Figure 1.2—Select PPSE processing



1.3.3.4 Destination States

The destination states for the SELECT PPSE command are listed in Table 1.5.

Table 1.5—Destination States for Select PPSE Command

SW1	SW2	IDLE	SELECTED
'6A'	'82'	IDLE	SELECTED
'6A'	'86'	IDLE	SELECTED
'90'	'00'	SELECTED	SELECTED
Other		IDLE	SELECTED

1.3.4 Loop Back

The LOOP BACK command returns without any further action the content of the data field of the C-APDU in the data field of the R-APDU.

1.3.4.1 Command Message

The LOOP BACK command message is coded according to Table 1.6.

Table 1.6—Loop Back Command Message

Code	Value
CLA	'80'
INS	'EE'
P1	'00'
P2	'00'
Lc	var
Data	Test Data
Le	'00'

The value of Lc defines the number of bytes included in the Test Data. The LOOP BACK command must work for Lc ranging from 1 to 250 and may optionally work for Lc greater than 250. The data field of the command message contains the Test Data to be returned in the data field of the response message.

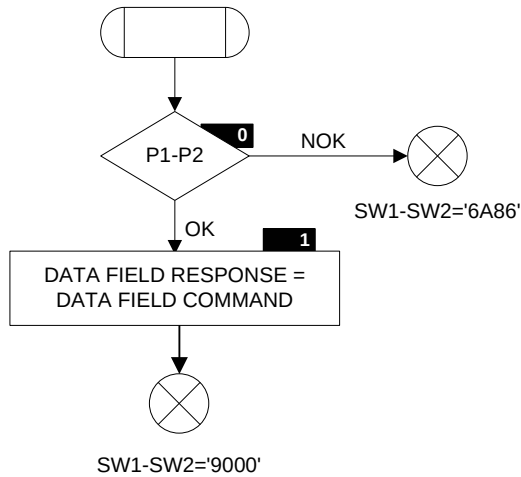
1.3.4.2 Response Message

The data field of the response message contains the Test Data included in the data field of the command message.

1.3.4.3 Processing

Figure 1.3 specifies the processing of the LOOP BACK command.

Figure 1.3—LOOP BACK Processing



Symbol 0

If P1 ≠ '00' or P2 ≠ '00', then the C-APDU is rejected (SW1-SW2 = '6A86').

Symbol 1

Build the data field of the response message. The data field of the response is set equal to the data field of the command message.

1.3.4.4 Destination States

The destination states for the LOOP BACK command are listed in Table 1.7.

Table 1.7—Destination States for LOOP BACK Command

SW1	SW2	SELECTED
'6A'	'86'	SELECTED
'90'	'00'	SELECTED
Other		SELECTED

2 *PayPass – M/Chip 4 Application*

2.1 Introduction

2.1.1 Overview

The *PayPass – M/Chip 4* application is an extension of the M/Chip 4 contact application for implementation on a dual interface card (i.e. a card with an EMV contact interface and a *PayPass* contactless interface). The references made in this document to *[M/CHIP4]* are applicable to all versions of that application, i.e.

- M/Chip 4 v1.0
- M/Chip 4 v1.1a
- M/Chip 4 v1.1b

This chapter uses the following terminology:

- "M/Chip 4 application"
All versions of the M/Chip Select 4 and M/Chip Lite 4 applications as specified in *[M/CHIP4]*.
- "*PayPass – M/Chip 4* application"
All versions of the M/Chip Select 4 and M/Chip Lite 4 dual interface applications as specified in this document.

The *PayPass – M/Chip 4* application supports the COMPUTE CRYPTOGRAPHIC CHECKSUM command. This is necessary for acceptance on a *PayPass – Mag Stripe* only terminal.

2.1.2 Assumptions

In this specification we make the following assumptions about the use of a dual interface card:

- Only one of the two interfaces is used between the power-on and power-off of the card.
- It is possible to know on the application layer whether the card is communicating via the contact or the contactless interface.

2.1.3 Data Objects

The *PayPass – M/Chip 4* application extends the data object dictionary of the M/Chip 4 application. The *PayPass – M/Chip 4* application supports the following new instances of existing M/Chip 4 application data objects:

- *AIP (PayPass)*
- *AFL (PayPass)*
- *Application Control (PayPass)*
- *Card Issuer Action Codes (PayPass)*

These data objects cannot be shared between the contact and contactless interfaces and are personalized with a specific value for the contactless interface. All other existing M/Chip 4 application data objects are shared between the contact and the contactless interface.

The *PayPass – M/Chip 4* application also supports the following data objects needed to support *PayPass – Mag Stripe* transactions:

- *IVCVC3_{TRACK1}, IVCVC3_{TRACK2}*
- *Static CVC3_{TRACK1}, Static CVC3_{TRACK2}*
- *KD_{CVC3}*

2.1.4 Offline Counters

The *PayPass – M/Chip 4* application shares the offline risk management counters between the contact and contactless interfaces. These offline counters will only be updated during a *PayPass* transaction if it is accepted offline. Bearing in mind that during a *PayPass* transaction the card is removed from the field after the first GENERATE AC command, this implies that the counters will only be updated if the cards replies with a TC to this GENERATE AC.

The counters are not updated if a *PayPass* transaction is completed online.

The counters can only be reset during an online contact transaction.

2.1.5 Log of Transactions

The *PayPass – M/Chip 4* application stores transaction information in the *Log of Transactions* whenever an AAC or TC is generated. Therefore, the issuer should notice that *PayPass* transactions completed online may not appear in the *Log of Transactions*. This is because the terminal does not send a second GENERATE AC command when the *PayPass – M/Chip 4* application generates an ARQC in response to the first GENERATE AC command.

2.2 Application States

2.2.1 Overview

The behavior of a personalized *PayPass* – M/Chip 4 application is specified as a state machine.

The application states used in this description are the same as the states defined for the M/Chip 4 application and are listed in Table 2.1.

Table 2.1—Application States of the *PayPass* – M/Chip 4 Application

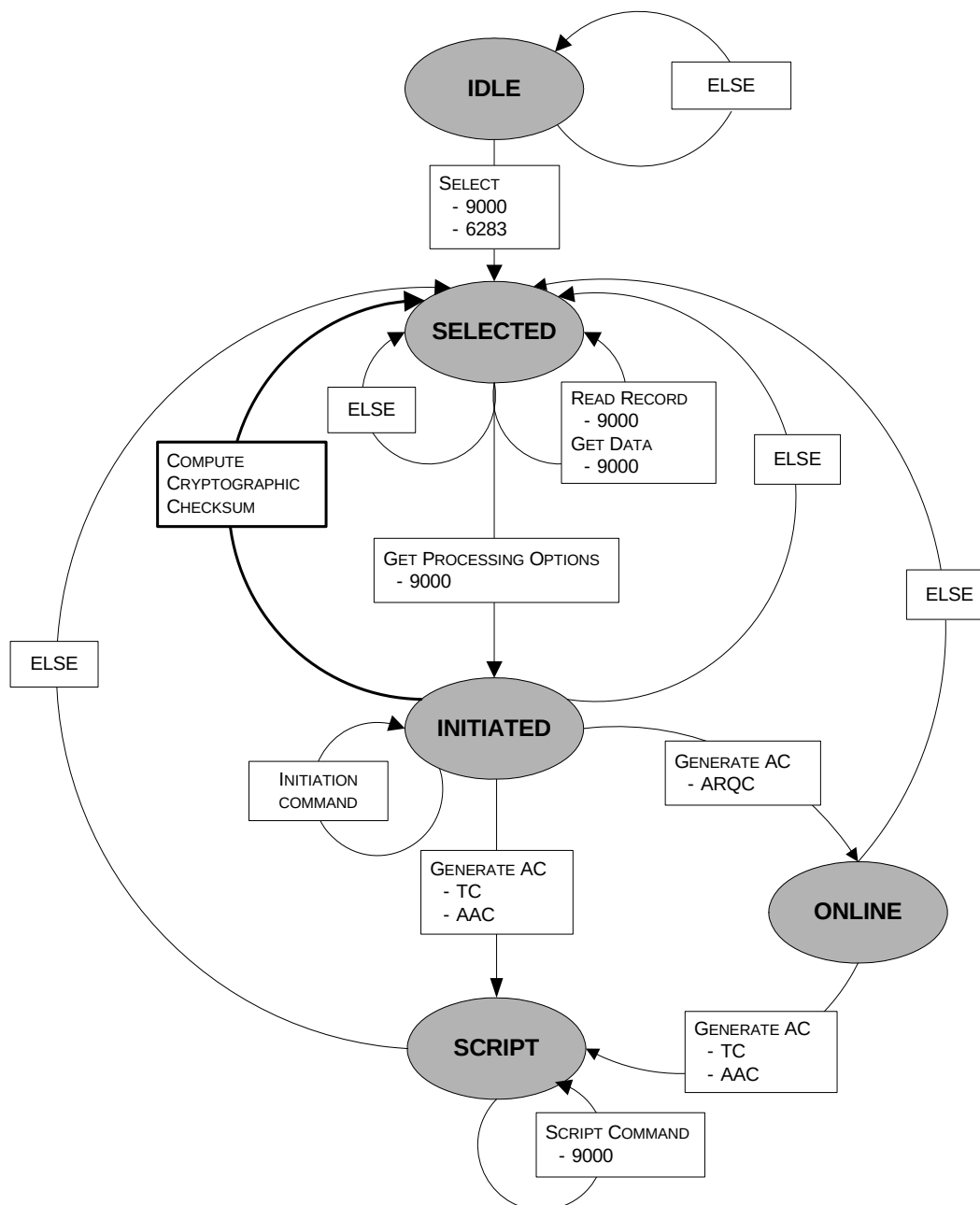
State	Description
IDLE	Application is not currently selected
SELECTED	Application is selected
INITIATED	Transaction is initiated
ONLINE	Application expects a connection with the issuer
SCRIPT	Application is ready to accept a script command

In addition to the state transitions supported by the M/Chip 4 application, the *PayPass* – M/Chip 4 application state machine supports also the following state transition:

- If the *PayPass* – M/Chip 4 application is in the state INITIATED, then it goes back to the state SELECTED after the processing, successful or not, of the COMPUTE CRYPTOGRAPHIC CHECKSUM command.

Figure 2.1 illustrates the state machine of the *PayPass* – M/Chip 4 application, showing the addition of the COMPUTE CRYPTOGRAPHIC CHECKSUM command.

Figure 2.1—State Machine of the *PayPass – M/Chip 4* Application



2.2.2 Initiation Commands

As shown in Figure 2.1, certain commands ("initiation commands") received when in the INITIATED state cause the application to return to the same state. The commands are different depending on the version of the *PayPass – M/Chip 4* application.

Table 2.2 provides the SW1-SW2 values for the initiation commands leading to the state INITIATED for the *PayPass – M/Chip Select 4* application.

Table 2.2—Initiation Commands for the *PayPass – M/Chip Select 4* Application

Initiation Commands	SW1-SW2
GET CHALLENGE	'9000'
GET DATA	'9000'
INTERNAL AUTHENTICATE	'9000'
VERIFY	'9000', '6983', '63Cx'
READ RECORD	'9000'

Table 2.3 provides the SW1-SW2 values for the initiation commands leading to the state INITIATED for the *PayPass – M/Chip Lite 4* application.

Table 2.3—Initiation Commands for the *PayPass – M/Chip Lite 4* Application

Initiation Commands	SW1-SW2
GET DATA	'9000'
VERIFY	'9000'
READ RECORD	'9000'

2.3 C-APDU Pre-Processing

2.3.1 C-APDU Recognition

C-APDU recognition is the procedure that identifies the C-APDU transmitted by the terminal to the *PayPass – M/Chip 4* application. The recognition is based firstly on the CLA byte and secondly on the INS byte. The *PayPass – M/Chip 4* application supports the combinations of CLA and INS bytes specified in Table 2.4.

The C-APDU recognition procedure takes as input the CLA and INS bytes and produces as output one of the responses as listed in the third column of Table 2.4.

If the CLA byte of the C-APDU is not one of those listed in Table 2.4, then the C-APDU Recognition procedure rejects the C-APDU and returns BAD CLA.

If the CLA byte is recognized, but the INS byte of the C-APDU is not one of those listed in Table 2.4, then the C-APDU Recognition procedure rejects the C-APDU and returns BAD INS.

Table 2.4—C-APDU Recognition

CLA	INS	C-APDU
'84'	'1E'	APPLICATION BLOCK
'84'	'18'	APPLICATION UNBLOCK
'80'	'2A'	COMPUTE CRYPTOGRAPHIC CHECKSUM
'80'	'AE'	GENERATE AC
'00'	'84'	GET CHALLENGE ^a
'80'	'CA'	GET DATA
'80'	'A8'	GET PROCESSING OPTIONS
'00'	'88'	INTERNAL AUTHENTICATE ^a
'84'	'24'	PIN CHANGE/UNBLOCK
'84'	'DA'	PUT DATA
'00'	'B2'	READ RECORD
'00'	'A4'	SELECT
'00'	'20'	VERIFY ^b
'84'	'DC'	UPDATE RECORD

^a Only applicable for M/Chip 4 Select.

^b Only applicable for the contact interface. If the C-APDU is received via the contactless interface, then the C-APDU Recognition must return BAD INS.

When the application has recognized the C-APDU it must perform a validity check on the following:

- Consistency between Lc and the length of data sent
- Le

These checks are protocol dependent and cannot be specified independently of the transport layer. However, when the validity check indicates an error in the lengths, the output of the procedure C-APDU Recognition is BAD LENGTH.

If the output of the C-APDU Recognition is BAD CLA, BAD INS or BAD LENGTH, then the C-APDU is not supported by the *PayPass – M/Chip 4* application over the active interface.

2.3.2 C-APDU Acceptance

C-APDU acceptance is the procedure that evaluates the validity of the C-APDU as a function of the current application state. Acceptance or rejection of a C-APDU by the *PayPass – M/Chip 4* application is specified in Table 2.5. The C-APDU acceptance procedure takes as input the response of the C-APDU recognition procedure.

If the C-APDU is rejected in the current state (R/CNS: rejected, conditions of use not satisfied), then the C-APDU is processed as specified in Section 2.3.3.

If the C-APDU is accepted in the current application state (P: processed), then the C-APDU is processed as specified in Section 2.4.

Table 2.5—Acceptance Matrix of *PayPass – M/Chip 4* Application

	SELECTED	INITIATED	ONLINE	SCRIPT
APPLICATION BLOCK	R/CNS	R/CNS	R/CNS	P
APPLICATION UNBLOCK	R/CNS	R/CNS	R/CNS	P
COMPUTE CRYPTOGRAPHIC CHECKSUM	R/CNS	P	R/CNS	R/CNS
GENERATE AC	R/CNS	P	P	R/CNS
GET CHALLENGE	R/CNS	P	R/CNS	R/CNS
GET DATA	P	P	R/CNS	R/CNS
GET PROCESSING OPTIONS	P	R/CNS	R/CNS	R/CNS
INTERNAL AUTHENTICATE	R/CNS	P	R/CNS	R/CNS
PIN CHANGE/UNBLOCK	R/CNS	R/CNS	R/CNS	P
PUT DATA	R/CNS	R/CNS	R/CNS	P
READ RECORD	P	P	R/CNS	R/CNS
SELECT	P	P	P	P
VERIFY	R/CNS	P	R/CNS	R/CNS
UPDATE RECORD	R/CNS	R/CNS	R/CNS	P

2.3.3 Rejected C-APDU Processing

A C-APDU may be rejected for two reasons:

- The bytes received are not recognized as a supported C-APDU (i.e. the CLA,INS pair does not correspond to a C-APDU supported by the *PayPass – M/Chip 4* application over the current active interface or there is an error in the lengths). In this case the rejection happens in the procedure C-APDU Recognition.
- The C-APDU is supported by the *PayPass – M/Chip 4* application, but the application is in a state where it is not accepted. In this case rejection happens during the C-APDU Acceptance procedure.

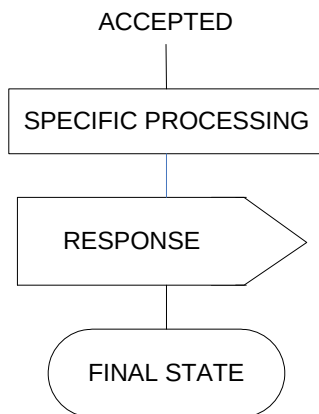
Refer to *[M/CHIP4]* for the description of the processing of the four cases R/CNS, BAD CLA, BAD INS and BAD LENGTH.

2.4 C-APDU Processing

2.4.1 Overview

Figure 2.2 illustrates the actions taken by the *PayPass – M/Chip 4* application when a C-APDU is processed.

Figure 2.2—Processing a C-APDU



A C-APDU is processed if the C-APDU Recognition and Acceptance procedures have not resulted in the rejection of the C-APDU (see Section 2.3.3).

The processing that is specific to the C-APDU is specified in Section 2.4.3 for the COMPUTE CRYPTOGRAPHIC CHECKSUM command and in *[M/CHIP4]* for all other commands.

The R-APDU resulting from the processing of a C-APDU and the destination state of the application when the C-APDU is processed are specified in the section dedicated to the C-APDU.

2.4.2 Instance of Data Objects Used

Commands that access the *AIP*, *AFL*, *Application Control* and *Card Issuer Action Codes* internal data objects must use the correct instance of the data object according to the active interface. This includes:

- The **GENERATE AC** command accessing the *Application Control* and *Card Issuer Action Codes* for the contact interface and the *Application Control (PayPass)* and *Card Issuer Action Codes (PayPass)* for the contactless interface. If the *AIP* is used as input to the generation of the *Application Cryptogram*, then the *AIP* must be used for the contact interface and the *AIP (PayPass)* must be used for the contactless interface.
- The **GET PROCESSING OPTIONS** command accessing the *AIP* and *AFL* for the contact interface and the *AIP (PayPass)* and *AFL (PayPass)* for the contactless interface.
- The **COMPUTE CRYPTOGRAPHIC CHECKSUM** command accessing the *Application Control (PayPass)* for both the contact and contactless interface.

2.4.3 Compute Cryptographic Checksum

2.4.3.1 Command Message

The **COMPUTE CRYPTOGRAPHIC CHECKSUM** command message is coded according to Table 2.6.

Table 2.6—Compute Cryptographic Checksum Command Message

Code	Value
CLA	'80'
INS	'2A'
P1	'8E'
P2	'80'
Lc	'04'
Data	<i>Unpredictable Number (Numeric)</i>
Le	'00'

As the *UDOL* is not provided by the *PayPass – M/Chip 4* application, the data field of the command message is the value field of the *Unpredictable Number (Numeric)* data object.

2.4.3.2 Data Field Returned in the Response Message

The data field of the response message is a constructed data object with tag '77'. The value field of the constructed data object includes the $CVC3_{TRACK1}$, the $CVC3_{TRACK2}$ and the ATC .

Table 2.7—Compute Cryptographic Checksum Response Message

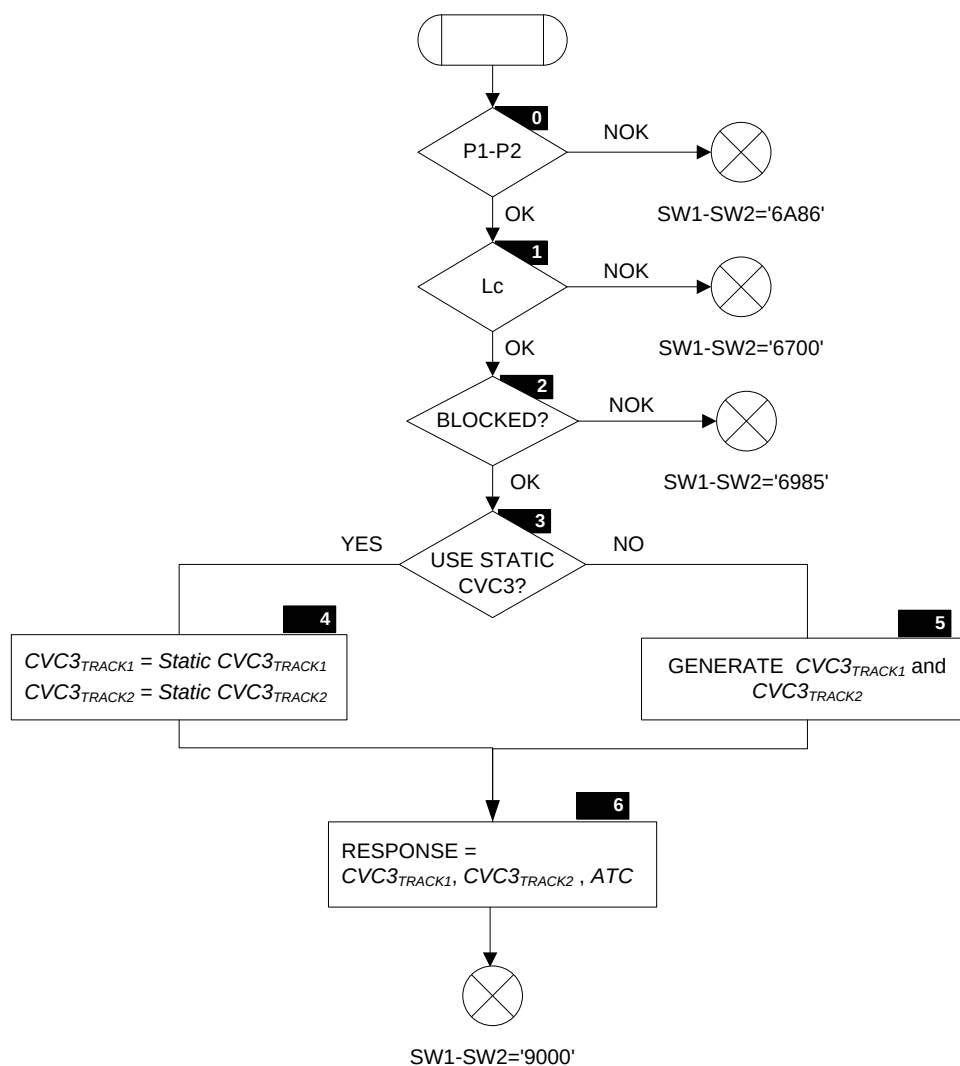
Data Object	Tag	Length
<i>Response Message Template</i>	'77'	15
$CVC3_{TRACK1}$	'9F60'	2
$CVC3_{TRACK2}$	'9F61'	2
ATC	'9F36'	2

The $CVC3_{TRACK1}$ and the $CVC3_{TRACK2}$ are cryptograms generated by the PayPass – M/Chip 4 application according to the algorithm specified in Section 2.5.1.

2.4.3.3 Processing

Figure 2.3 specifies the flow of the COMPUTE CRYPTOGRAPHIC CHECKSUM command processing.

Figure 2.3—Compute Cryptographic Checksum Processing



Symbol 0

If $P1 \neq '8E'$ or $P2 \neq '80'$, then the C-APDU is rejected (SW1-SW2 = '6A86').

Symbol 1

If $Lc \neq 4$, then the C-APDU is rejected (SW1-SW2 = '6700').

Symbol 2

If the application is blocked (i.e. if *Previous Transaction History*[5] = 1b), then the C-APDU is rejected (SW1-SW2 = '6985').

Symbol 3

The *PayPass – M/Chip 4* application checks if the *Static CVC3* must be used (i.e. *Application Control (PayPass)*[3][8] = 1b).

Symbol 4

The *PayPass* – M/Chip 4 application sets $CVC3_{TRACK1}$ equal to *Static* $CVC3_{TRACK1}$ and $CVC3_{TRACK2}$ equal to *Static* $CVC3_{TRACK2}$.

Symbol 5

The *PayPass* – M/Chip 4 application generates $CVC3_{TRACK1}$ and $CVC3_{TRACK2}$ as specified in Section 2.5.1.

Symbol 6

The *PayPass* – M/Chip 4 application generates the response message template containing the $CVC3_{TRACK1}$, the $CVC3_{TRACK2}$ and the *ATC*.

2.4.3.4 Destination State

The destination states for the COMPUTE CRYPTOGRAPHIC CHECKSUM command are listed in Table 2.8.

Table 2.8—Destination State for Compute Cryptographic Checksum Command

SW1	SW2	INITIATED
'67'	'00'	SELECTED
'69'	'85'	SELECTED
'6A'	'86'	SELECTED
'90'	'00'	SELECTED
Other		SELECTED

2.4.4 Get Data

The GET DATA command is processed as specified in [M/CHIP4]. This section specifies the additional tag values that must be supported by the GET DATA command of the *PayPass* – M/Chip 4 application.

Table 2.9—Additional Tag Value for Get Data

P1/P2	Data Object	Length
'00CD'	<i>Card Issuer Action Code (PayPass) – Default</i>	3
'00CE'	<i>Card Issuer Action Code (PayPass) – Online</i>	3
'00CF'	<i>Card Issuer Action Code (PayPass) – Decline</i>	3
'00D7'	<i>Application Control (PayPass)</i>	3

2.4.5 Put Data

The PUT DATA command is processed as specified in [M/CHIP4]. This section specifies the additional tag values that have to be supported by the PUT DATA command of the PayPass – M/Chip 4 application.

Table 2.10—Additional Tag Values for Put Data

P1/P2	Data Object	Length
'00CD'	<i>Card Issuer Action Code (PayPass) – Default</i>	3
'00CE'	<i>Card Issuer Action Code (PayPass) – Online</i>	3
'00CF'	<i>Card Issuer Action Code (PayPass) – Decline</i>	3
'00D7'	<i>Application Control (PayPass)</i>	3
'00D8'	<i>AIP (PayPass)</i>	2
'00D9'	<i>AFL (PayPass)</i>	var
'00DA'	<i>Static CVC3_{TRACK1}</i>	2
'00DB'	<i>Static CVC3_{TRACK2}</i>	2
'00DC'	<i>IVCVC3_{TRACK1}</i>	2
'00DD'	<i>IVCVC3_{TRACK2}</i>	2

2.5 Key Management and Cryptographic Algorithms

The *PayPass – M/Chip 4* application supports all cryptographic computations supported by the M/Chip 4 application as specified in [M/CHIP4]. This section lists only the additional cryptographic computations and key derivations that are supported by the *PayPass – M/Chip 4* application.

2.5.1 Dynamic CVC3

This section specifies how the *PayPass – M/Chip 4* application constructs the dynamic CVC3.

The *PayPass – M/Chip 4* application generates a dynamic CVC3 for the *Track 1 Data* ($CVC3_{TRACK1}$) and a dynamic CVC3 for the *Track 2 Data* ($CVC3_{TRACK2}$). Both cryptograms are generated with the same dynamic data (*Unpredictable Number (Numeric)* and *ATC*) and with the same secret key (KD_{CVC3}), but with a different initialization vector ($IVCVC3_{TRACK1}$ for $CVC3_{TRACK1}$ and $IVCVC3_{TRACK2}$ for $CVC3_{TRACK2}$).

The $CVC3_{TRACK1}$ is generated using DES3 encipherment as follows:

1. Concatenate the data listed in Table 2.11 in the order specified to obtain an 8 byte data block (D).

Table 2.11—Track 1 CVC3 Data Objects

Data Object	Length
$IVCVC3_{TRACK1}$	2 bytes
<i>Unpredictable Number (Numeric)</i>	4 bytes
<i>ATC</i> ^a	2 bytes

^a If *Application Control*[3][7] = 0b (Do not include the *ATC* in dynamic CVC3 generation), then the 2 bytes are filled with hexadecimal zeroes ('00 00').

2. Calculate O as follows:

$$O := \text{DES3}(KD_{CVC3})[D]$$

3. The two least significant bytes of O are the $CVC3_{TRACK1}$.

The $CVC3_{TRACK2}$ is generated in the same way by replacing $IVCVC3_{TRACK1}$ with $IVCVC3_{TRACK2}$.

2.5.2 Symmetric Key Management

The issuer of the *PayPass* – M/Chip 4 application must derive the double length DES3 key for CVC3 generation (KD_{CVC3}) for each card.

Table 2.12— KD_{CVC3}

Key	Description	length
KD_{CVC3}	ICC Derived Key for CVC3 Generation	16

Annex B specifies the key derivation method used to generate KD_{CVC3} .

2.6 Data Objects Location

2.6.1 Transient Data Objects that Span a Single C-APDU Processing

Some transient data objects have a lifetime that spans a single C-APDU processing. All these transient data objects are created during application selection and are listed in [M/CHIP4].

2.6.2 Additional Persistent Data Objects

All the persistent data objects that are listed in [M/CHIP4] are supported by the PayPass – M/Chip 4 application. Table 2.13 lists the additional persistent data objects of the PayPass – M/Chip 4 application and their access conditions.

Table 2.13—Additional Persistent Data Objects

Tag	Name	read record	update record	internal read	internal update	get data	put data
'56'	Track 1 Data	Yes	Yes	No	No	No	No
'9F62'	PCVC3 _{TRACK1}	Yes	Yes	No	No	No	No
'9F63'	PUNATC _{TRACK1}	Yes	Yes	No	No	No	No
'9F64'	NATC _{TRACK1}	Yes	Yes	No	No	No	No
'9F65'	PCVC3 _{TRACK2}	Yes	Yes	No	No	No	No
'9F66'	PUNATC _{TRACK2}	Yes	Yes	No	No	No	No
'9F67'	NATC _{TRACK2}	Yes	Yes	No	No	No	No
'9F6B'	Track 2 Data	Yes	Yes	No	No	No	No
'9F6C'	Mag Stripe Application Version Number (Card)	Yes	Yes	No	No	No	No
'CD'	Card Issuer Action Code (PayPass) – Default	No	No	Yes	No	Yes	Yes
'CE'	Card Issuer Action Code (PayPass) – Online	No	No	Yes	No	Yes	Yes
'CF'	Card Issuer Action Code (PayPass) – Decline	No	No	Yes	No	Yes	Yes
'D7'	Application Control (PayPass)	No	No	Yes	No	Yes	Yes
'D8'	AIP (PayPass)	No	No	Yes	No	No	Yes
'D9'	AFL (PayPass)	No	No	Yes	No	No	Yes
'DA'	Static CVC3 _{TRACK1}	No	No	Yes	No	No	Yes
'DB'	Static CVC3 _{TRACK2}	No	No	Yes	No	No	Yes
'DC'	IVCVC3 _{TRACK1}	No	No	Yes	No	No	Yes
'DD'	IVCVC3 _{TRACK2}	No	No	Yes	No	No	Yes

2.6.3 Secret Keys

All the secret keys that are listed in [M/CHIP4] are supported by the *PayPass – M/Chip 4* application. Table 2.14 defines the additional DES3 key for CVC3 generation that must be supported by the *PayPass – M/Chip 4* application.

Table 2.14— KD_{CVC3}

Data Object	length	get data	internal update	put data
KD_{CVC3}	16	No	No	No

2.7 Personalization

This section specifies the data objects that are available to the issuer for personalization. The personalization commands are not in the scope of this specification. They are left to the implementation.

All data objects available for personalization are stored in persistent memory of the card and are listed in [M/CHIP4]. This section specifies only the specific personalization requirements for the PayPass – M/Chip 4 application.



Note When the PayPass – M/Chip 4 application is personalized according to the EMV Card Personalization Specification as defined in [M/CHIP4 CPS], then the data groupings for the PayPass specific data objects are as listed in Annex B.

2.7.1 Compute Cryptographic Checksum Data Objects

Table 2.15 lists the persistent card data objects used during the generation of $CVC3_{TRACK1}$ and $CVC3_{TRACK2}$.

Table 2.15—Data Objects for CVC3 Generation

Tag	Data Object	Length (bytes)
'DA'	Static $CVC3_{TRACK1}$	2
'DB'	Static $CVC3_{TRACK2}$	2
'DC'	IVCVC3 $_{TRACK1}$	2
'DD'	IVCVC3 $_{TRACK2}$	2

2.7.2 Persistent Data Referenced in the AFL (*PayPass*)

Table 2.16 lists the additional persistent data objects referenced in the *AFL (PayPass)* that are needed for *PayPass – Mag Stripe* transactions.

Table 2.16—*PayPass – Mag Stripe* Data Objects

Tag	Name	Length (bytes)
'9F6C'	<i>Mag Stripe Application Version Number (Card)</i>	2
'9F62'	<i>PCVC3_{TRACK1}</i>	6
'9F63'	<i>PUNATC_{TRACK1}</i>	6
'56'	<i>Track 1 Data</i>	var up to 76
'9F64'	<i>NATC_{TRACK1}</i>	1
'9F65'	<i>PCVC3_{TRACK2}</i>	2
'9F66'	<i>PUNATC_{TRACK2}</i>	2
'9F6B'	<i>Track 2 Data</i>	var up to 19
'9F67'	<i>NATC_{TRACK2}</i>	1

2.7.3 Persistent Data Objects for Card Risk Management

The data objects listed in Table 2.17 are the *PayPass – M/Chip 4* specific data objects for card risk management.

Table 2.17—Persistent Data Objects for Card Risk Management

Tag	Name	Length (bytes)
'CD'	<i>Card Issuer Action Code (PayPass) – Default</i>	3
'CE'	<i>Card Issuer Action Code (PayPass) – Online</i>	3
'CF'	<i>Card Issuer Action Code (PayPass) – Decline</i>	3
'D7'	<i>Application Control (PayPass)</i>	3

2.7.4 Triple DES Key

Table 2.18 lists the DES3 key used for CVC3 generation.

Table 2.18—*KD_{CVC3}*

Data Object	Length
<i>KD_{CVC3}</i>	16

2.7.5 Persistent Data Objects for GPO Response

Table 2.19 lists the data objects returned by the *PayPass – M/Chip 4* application in response to the GET PROCESSING OPTIONS command.

Table 2.19—Persistent Data Objects for GPO Response

Tag	Name	Length (bytes)
'D8'	<i>AIP (PayPass)</i>	2
'D9'	<i>AFL (PayPass)</i>	12 or 16

Annex A Data Objects Dictionary

The *PayPass* – M/Chip 4 application supports all data objects supported by the M/Chip 4 application as listed in the Data Objects Dictionary of [M/CHIP4]. This annex lists only the additional data objects that are supported by the *PayPass* – M/Chip 4 application.

A.1 AFL (*PayPass*)

Tag: 'D9'

Format: b, var (a memory space of at least 32 bytes must be available)

Description: The *AFL (PayPass)* indicates the location (SFI and range of records) of the AEFs when the contactless interface is used.



Note The tag 'D9' of the *AFL (PayPass)* must only be used to identify the data object for the PUT DATA command. When the *AFL (PayPass)* is returned in the response message of the GET PROCESSING OPTIONS command, then the EMV tag '94' must be used.

A.2 AIP (*PayPass*)

Tag: 'D8'

Format: b, 2 bytes

Description: The *AIP (PayPass)* indicates the capabilities of the card to support specific functions in the application when the contactless interface is used.



Note The tag 'D8' of the *AIP (PayPass)* must only be used to identify the data object for the PUT DATA command. When the *AIP (PayPass)* is returned in the response message of the GET PROCESSING OPTIONS command, then the EMV tag '82' must be used.

A.3 Application Control (PayPass)

Tag: 'D7'
Format: b, 3 bytes
Description: The *Application Control (PayPass)* activates or de-activates functions in the application when the contactless interface is used. Byte 1 and byte 2 are coded as the *Application Control* (tag 'D5') specified in [M/CHIP4].
Byte 3 of the *Application Control (PayPass)* is coded as shown in Table A.1.

Table A.1—Byte 3 of the Application Control (PayPass)

b8	b7	b6	b5	b4	b3	b2	b1	Description
x								Use static CVC3
0								Do not use static CVC3
1								Use static CVC3
	x							Include ATC in dynamic CVC3 generation
	0							Do not include ATC in dynamic CVC3 generation
	1							Include ATC in dynamic CVC3 generation
		0	0	0	0	0	0	RFU

A.4 Card Issuer Action Codes (PayPass) – Decline, Default, Online

Tags: *Card Issuer Action Code (PayPass) – Default:* 'CD'
Card Issuer Action Code (PayPass) – Online: 'CE'
Card Issuer Action Code (PayPass) – Decline: 'CF'

Format: b, 3 bytes

Description: The *Card Issuer Action Codes (PayPass)* are represented by three *PayPass – M/Chip 4* proprietary data objects: *Card Issuer Action Code (PayPass) – Default*, *Card Issuer Action Code (PayPass) – Online* and *Card Issuer Action Code (PayPass) – Decline*. They are compared to the decisional part of the *Card Verification Results* to decide which cryptogram to include in the response to the GENERATE AC command.

The *Card Issuer Action Codes (PayPass)* are formatted as the *Card Issuer Action Codes* (tags 'C3', 'C4' and 'C5') specified in [M/CHIP4].

A.5 CVC3_{TRACK1}

Tag: '9F60'
Format: b, 2 bytes
Description: The CVC3_{TRACK1} is a 2 byte cryptogram returned in the response to the COMPUTE CRYPTOGRAPHIC CHECKSUM command.

A.6 CVC3_{TRACK2}

Tag: '9F61'
Format: b, 2 bytes
Description: The CVC3_{TRACK2} is a 2 byte cryptogram returned in the response to the COMPUTE CRYPTOGRAPHIC CHECKSUM command.

A.7 IVCVC3_{TRACK1}

Tag: 'DC'
Format: b, 2 bytes
Description: The IVCVC3_{TRACK1} is an issuer proprietary static data object that is used as input for the generation of the CVC3_{TRACK1} cryptogram.

A.8 IVCVC3_{TRACK2}

Tag: 'DD'
Format: b, 2 bytes
Description: The IVCVC3_{TRACK2} is an issuer proprietary static data object that is used as input for the generation of the CVC3_{TRACK2} cryptogram.

A.9 KD_{CVC3}

Tag: –
Format: b, 16 bytes
Description: KD_{CVC3} is the ICC derived key for CVC3 Generation. KD_{CVC3} is used directly to compute the dynamic CVC3. No session key is used.

A.10 Static CVC3_{TRACK1}

Tag: 'DA'

Format: b, 2 bytes

Description: The *Static CVC3_{TRACK1}* is the static version of the dynamic CVC3 of the track 1 data converted into binary format (e.g. a *Static CVC3_{TRACK1}* with value "812" in ans format is stored as '032C'). The *PayPass – M/Chip 4* application returns the *Static CVC3_{TRACK1}* instead of the dynamically calculated *CVC3_{TRACK1}* if *Application Control (PayPass)[3][8] = 1b*.

A.11 Static CVC3_{TRACK2}

Tag: 'DB'

Format: b, 2 bytes

Description: The *Static CVC3_{TRACK2}* is the static version of the dynamic CVC3 of the track 2 data converted into binary format (e.g. a *Static CVC3_{TRACK2}* with value 812 in numeric format is stored as '032C'). The *PayPass – M/Chip 4* application returns the *Static CVC3_{TRACK2}* instead of the dynamically calculated *CVC3_{TRACK2}* if *Application Control (PayPass)[3][8] = 1b*.

Annex B ICC Derived Key for CVC3 Generation (KD_{CVC3})

This annex specifies the key derivation method used to generate KD_{CVC3} .

KD_{CVC3} is a 16-byte DES3 key derived from the *Issuer Master Key for CVC3 Generation* (IMK_{CVC3}) as follows:

1. Concatenate from left to right the *PAN* (without any 'F' padding) with the *PAN Sequence Number* (if the *PAN Sequence Number* is not available, then it is replaced by a '00' byte). If the result X is less than 16 digits long, pad it to the left with hexadecimal zeros in order to obtain an eight-byte number Y in numeric (n) format. If X is at least 16 digits long, then Y consists of the 16 rightmost digits of X in numeric (n) format.
2. Compute the two eight-byte numbers:

$$Z_L := \text{DES3}(IMK_{CVC3})[Y]$$

$$Z_R := \text{DES3}(IMK_{CVC3})[Y \oplus ('FF' || 'FF' || 'FF' || 'FF' || 'FF' || 'FF' || 'FF' || 'FF')]$$

and define:

$$KD_{CVC3} := (Z_L || Z_R).$$

Annex C PayPass Data Groupings

This annex defines the structure of the data groupings that must be used to personalize the PayPass specific data objects when personalizing the PayPass – M/Chip 4 application according to the EMV Card Personalization Specification as defined in [M/CHIP4 CPS].

DGI 'B002'

Data Object	Length
<i>Application Control (PayPass)</i>	3
<i>Static CVC3_{TRACK1}</i>	2
<i>Static CVC3_{TRACK2}</i>	2
<i>IVCVC3_{TRACK1}</i>	2
<i>IVCVC3_{TRACK2}</i>	2
<i>Card Issuer Action Code (PayPass) – Default</i>	3
<i>Card Issuer Action Code (PayPass) – Online</i>	3
<i>Card Issuer Action Code (PayPass) – Decline</i>	3

DGI 'B005'

Data Object	Length
<i>AIP (PayPass)</i>	2
<i>AFL (PayPass)</i>	var.

DGI '8400' – Encrypted

Data Object	Length
<i>KD_{CVC3}</i>	16

***** End of Document *****