MOST

Media **O**riented **S**ystems **T**ransport

Multimedia and Control
Networking Technology

MOST Content Protection Scheme
DTCP Implementation
Revision 2.2
03/2007

MOSTCO CONFIDENTIAL
See page 3 for the terms of disclosure

Legal Notice

COPYRIGHT

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
D-76185 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

This Specification is Confidential Information of the MOST Cooperation. It may only be disclosed to member companies. Member companies wishing to discuss these Specifications with suppliers or other third parties must ensure that a commercially standard form of non-disclosure agreement has been previously executed by the party receiving such Specifications. Use of these Specifications may only be for purposes for which they are intended by the MOST Cooperation. Unauthorized use or disclosure is a violation of law.

Contents

Document History

Changes Specification 1.0-00 to Specification 2.2-00

Version	Date	Section	Comment on changes
1.0-00	2001-12-10	-	First version
1.1-06	2004-02-09	All	Major updates
1.1-07	2004-04-01	-	DRAFT removed
1.1-08	2004-05-28	3 5	Updated Function Catalog Added Chapter 5: Message Sequence Charts
1.1-09	2004-06-03	3, 5	Updates from WG meeting
1.1-10	2005-01-12	3	Updates from WG meeting
2.0-00	2005-02-28	All 3, 5	New template Updates from WG meeting
2.0-01	2005-03-14	2 5 (now 4)	Chapter 2 in 2.0-00 deleted, by request of WG-DA Introduction included, by request of WG-DA
2.0-02	2005-03-14	4	Collaboration diagram included, by request of WG-DA
2.0-03	2005-06-20	All	Update from WG Telephone Conference
	2006-06-28	All	WG work
	2006-09-12	All	TeleCon updates
	2006-10-12	All	WG work
2.1-06	2007-01-25	4.7,4.8,4.9	Update MSC Sequences according changes in DTCP_Status, DTCP_Control
2.1-07	2007-01-30	All	WG work
2.2	2007-03-12	All	Reference to MOST Stream Transmission Specification changed to Revision 1.3. Updated Enumerations. Renamed parameter Control to Control_5C. Renamed parameter Status to Status_5C. Renamed parameter SourceNr. / SinkNr. to SourceSinkNr.

1 Introduction

1.1 Purpose

Today, two ways exist for implementing DTCP mechanisms into MOST systems. These are:

- 5C DTCP specification Volume 1, Supplement B: Mapping DTCP to MOST
- 5C DTCP specification Volume 1, Supplement E: Mapping DTCP to IP

The details of DTCP can be found at www.dtcp.com.

This document describes the MOST functions and services required to enable Digital Transmission Content Protection (DTCP) protocols according '**Supplement B**' only.
For an implementation according to 'Supplement E', please refer to the MOST Cooperation document 'MOST_ContentProtectionScheme_DTCP-IP_Implementation'.

Please note:
Every usage of DCTP requires a license agreement with DTLA (Digital Transmission License Administrator). In particular, the implementation of this DTCP specification on the MOST Network requires full compliance with the DTCP license agreement and its procedural appendix, compliance rules, and policy statements.

1.2 Related Documents

- MOST Specification Rev 2.4
- MOST Content Security Specification Rev. 1.0
- MOST Specification for Stream Transmission Rev. 1.3
- Digital Transmission Content Protection Specification by Hitachi, Ltd., Intel Corporation, Matsushita Electric Industrial Co., Ltd., Sony Corporation, and Toshiba Corporation (collectively, the "5C"). Revision 1.1

1.3 Terms and Abbreviations

Sink	The target of a data transfer
Source	The origin of a data transfer
DTCP	Digital Transmission Content Protection
BW	BW relates to allocated block width
MOST	Media Oriented Systems Transport

2 DTCP Functions

2.1 DTCP_StartProcess (FktID = 0x120)

Section type: Coordination

Important:
Function ID 0x120 has an alias 0x12E. It is not recommended to use the alias.
However, system integrators are allowed to use Function ID 0x12E instead of 0x120 for compatibility reasons.

2.1.1 Format of function

Function Class: Unclassified Method

FBlock	Function	OPType	Parameter
GeneralFBlock	DTCP_StartProcess (0x120)	Abort	-
		StartResult	FBlockID, InstID
		Processing	-
		Result	FBlockID, InstID
		Error	ErrorCode, ErrorInfo

2.1.2 Parameters

FBlockID

Functional address of the function block of the source

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

InstID

Instance ID of the function block of the source

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

2.2 DTCP_Control (FktID = 0x121)

Section type: Coordination

Important:
Function ID 0x121 has an alias 0x12F. It is not recommended to use the alias.
However, system integrators are allowed to use Function ID 0x12F instead of 0x121 for compatibility reasons.

This function transmits DTCP control commands and corresponding responses.

2.2.1 Format of function

Function Class: Unclassified Method

FBlock	Function	OPType	Parameter
GeneralFBlock	DTCP_Control (0x121)	StartResult	RequesterFBlockID, RequesterInstID, Control_5C
		Processing	-
		Result	RequesterFBlockID, RequesterInstID, Control_5C
		Error	ErrorCode, ErrorInfo

2.2.2 Parameters

RequesterFBlockID

Functional address of the function block that sends a DTCP command.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

RequesterInstID

Instance ID of the function block that sends a DTCP command.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

Control_5C

The contents and structure of the Control fields are detailed in “Supplement B” of the 5C DTCP Specification

Basis datatype	Length	Description
Stream	-	Defined by 5C

2.3 DTCP_Status (FktID = 0x122)

Section type: Coordination

This function transmits DTCP status commands and corresponding responses.

2.3.1 Format of function

Function Class: Unclassified Method

FBlock	Function	OPType	Parameter
GeneralFBlock	DTCP_Status (0x122)	StartResult	RequesterFBlockID, RequesterInstID, Status_5C
		Processing	-
		Result	RequesterFBlockID, RequesterInstID, Status_5C
		Error	ErrorCode, ErrorInfo

2.3.2 Parameters

RequesterFBlockID

Functional address of the function block that sends a DTCP command.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

RequesterInstID

Instance ID of the function block that sends a DTCP command.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

Status_5C

The contents and structure of the Status fields are detailed in “Supplement B” of the 5C DTCP Specification

Basis datatype	Length	Description
Stream		Defined by 5C

2.4 DTCP_CipherStatus (FktID = 0x123)

Section type: Coordination

Important:
Function ID 0x123 has an alias 0x12D. It is not recommended to use the alias.
However, system integrators are allowed to use Function ID 0x12D instead of 0x123 for compatibility reasons.

This function gives information about the state of the AKE and Ciphering components.

2.4.1 Format of function

Function Class: Unclassified Property

FBlock	Function	OPType	Parameter
GeneralFBlock	DTCP_Cipher Status	Get	SourceSinkNr
		Status	SourceSinkNr, AuthenticationState, AvailableExchangeKeys, CipherError
		Error	ErrorCode, ErrorInfo

2.4.2 Parameters

SourceSinkNr

Number of a data source or sink.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

AuthenticationState

AuthenticationState gives the current state of Authentication as defined in Chapter 3 of the 5C DTCP Specification.

Basis datatype	Range of values	Code	Description
Enum	0x00..0x05	0x00	State A0: Unauthenticated
		0x01	State A1: Full Authentication
		0x02	State A2: Restricted Authentication
		0x03	State A3: Authenticated
		0x04	State A4: Send Content Channel Key
		0x05	State A5: Initialize Device

AvailableExchangeKeys

AvailableExchangeKeys gives the current set of available ExchangeKeys

Basis datatype	Bit #	Code	Description
BitField	Bit 0	False	ExchangeKey for EMI Mode A (Copy-never) not available
		True	ExchangeKey for EMI Mode A (Copy-never) available
	Bit 1	False	ExchangeKey for EMI Mode B (Copy-one-generation) not available
		True	ExchangeKey for EMI Mode B (Copy-one-generation) available
	Bit 2	False	ExchangeKey for EMI Mode C (No-more-copies) not available
		True	ExchangeKey for EMI Mode C (No-more-copies) available
	Bit 3	False	Reserved
		True	Reserved
	Bit 4	False	DTCP-IP ExchangeKey for all E-EMI modes not available
		True	DTCP-IP ExchangeKey for all E-EMI modes available
	Bit 5..7	False	Reserved
		True	Reserved

CipherError

CipherError gives the current state of the ciphering machines

Basis datatype	Range of values	Code	Description
Enum	0x00..0x20	0x00	No error
		0x10	Encoding Error
		0x20	Decoding Error

2.5 DTCP_Info (FktID = 0x124)

Section type: Coordination

This function gives information about MOST DTCP parameters.

2.5.1 Format of function

Function Class: Unclassified Property

FBlock	Function	OPType	Parameter
GeneralFBlock	DTCP_Info	Get	SourceSinkNr
		Status	SourceSinkNr, PacketFormat, MediaType, PacketLength, EncryptionFrameSize
		Error	ErrorCode, ErrorInfo

2.5.2 Parameters

SourceSinkNr

Number of data source / sink

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

PacketFormat

PacketFormat gives the packet format which is used by the source / sink (please see MOST Content Protection Scheme – DTCP Implementation).

Basis datatype	Range of values	Code	Description
Enum	0x00..0x02	0x00	Not defined
		0x01	Not applicable
		0x02	Generic MOST-DTCP Packet Format

MediaType

This parameter refers to the MediaType values, which are given in the MOST Specification for Stream Transmission.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

PacketLength

This parameter refers to the MOST Packet Length value, which is given in the MOST Specification for Stream Transmission.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Word	0	full range	1	none

EncryptionFrameSize

This parameter refers to the DTCP Encryption Frame Size value, which is given in the MOST Specification for Stream Transmission.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Word	0	full range	1	none

2.6 DTCP_ContentKeyProcess (FktID = 0x125)

Section type: Coordination

The method starts the establishing of Content Keys.

2.6.1 Format of function

Function Class: Unclassified Method

FBlock	Function	OPType	Parameter
GeneralFBlock	DTCP_ContentKey Process (0x125)	StartResult	FBlockID, InstID, SourceNr, SinkNr, PacketFormat, MediaType, PacketLength, EncryptionFrameSize
		Processing	None
		Result	FBlockID, InstID
		Error	ErrorCode, ErrorInfo

2.6.2 Parameters

FBlockID

Functional address of the function block of the source.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

InstID

Instance ID of the function block instance of the source.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

SourceNr

Number of data source

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

SinkNr

Number of data sink

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

PacketFormat

PacketFormat gives the packet format that is used by the source (please see MOST Content Protection Scheme – DTCP-Implementation).

Basis datatype	Range of values	Code	Description
Enum	0x00..0x02	0x00	Not defined
		0x01	Not applicable
		0x02	Generic MOST-DTCP Packet Format

MediaType

This parameter refers to the MediaType values, which are given in the MOST Specification for Stream Transmission.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Byte	0	full range	1	none

PacketLength

This parameter refers to the MOST Packet Length value, which is given in the MOST Specification for Stream Transmission.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Word	0	full range	1	none

EncryptionFrameSize

This parameter refers to the DTCP Encryption Frame Size value, which is given in the MOST Specification for Stream Transmission.

Basis datatype	Exp	Range of values	Step	Unit
Unsigned Word	0	full range	1	none

3 Protected Content

For implementing the DTCP mechanisms, the data to be protected is encrypted, transmitted and decrypted in packetized form. Embedded information has to be transported as part of the encrypted section.

Exchange key Expiration
The Exchange Keys of source devices expire when the sources stop output of protected content. Sources are considered to have stopped output when there are no synchronous connections or asynchronous data transfers for audiovisual or audio content

Detailed definition of a synchronous connection:

- A logical connection between MOST devices
- Is not bound to specific MOST Source Numbers or MOST Sink Numbers
- Is not bound to specific allocated channels
- Is not affected by low-level unlocks or bus-resets

Packetizing of streaming content
For all types of streaming content, a common generic packetizing scheme called ‘MOST-DTCP Packet format’ is used. It is based on the isochronous transmission protocol specified in the ‘MOST Stream Transmission’ document. Additionally, DTCP specific information is encoded directly after the synchronization headers.

Embedded Information
Depending on the stream type and origin, specific ‘Embedded Information’ is carried over a dedicated SAD channel called ‘Info’. The ‘MOST Stream Transmission’ document specifies the appropriate parameters and information to be used for the different streams available on MOST.

3.1 Generic MOST-DTCP Packet Format

To pack streaming data in DTCP packets two additional Stream-Associated-Data channels (SADs) are used:

- SAD0 transmits the unprotected header information.
  Unused areas between two headers are reserved and must be transmitted as ‘0x00’.

- SAD1 is defined to transmit the protected ‘Embedded Information’.
  Unused areas are reserved and must be transmitted as ‘0x00’.

Please note:
The correlation between the header and the packet is shown in the following figure. The packet starts one frame upon after reception of Header [3].

The gray areas refer to protected content (PC). The white areas are unprotected headers.

Frame	Byte [0] SAD0 - Header	Byte [1] SAD1 - Info	Byte [2]		…	Byte [BW-1]
…	Header [0]	Info [M-3]	Data	Data	Data	Data
…	Header [1]	Info [M-2]	Data	Data	Data	Data
…	Header [2]	Info [M-1]	Data	Data	Data	Data
…	Header [3]	Info [M]	Data	Data	Data	Data [N]
A	reserved	Info [0]	Data [0]	Data [1]	Data [..]	Data [BW-3]
A + 1	reserved	Info [1]	Data [BW-2]	Data [BW-1]	Data	Data
A + 2	reserved	Info [2]	Data	Data	Data	Data
A + 3	reserved	Info [3]	Data	Data	Data	Data
A + 4	reserved	…	Data	Data	Data	Data
A + 5	reserved	…	Data	Data	Data	Data
A + 6	Header [0]	Info [M-3]	Data	Data	Data	Data
…	Header [1]	Info [M-2]	Data	Data	Data	Data
…	Header [2]	Info [M-1]	Data	Data	Data	Data
…	Header [3]	Info [M]	Data	Data	Data	Data [N]
B	reserved	Info [0]	Data [0]	Data [1]	Data[..]	Data [BW-3]
B + 1	reserved	Info [1]	Data [BW-2]	Data [BW-1]	Data	Data
B + 2	reserved	Info [2]	Data	Data	Data	Data
…	reserved	Info [3]	Data	Data	Data	Data

Figure 3-1: Streaming Data with Additional SADs (Frame-by-Frame View)

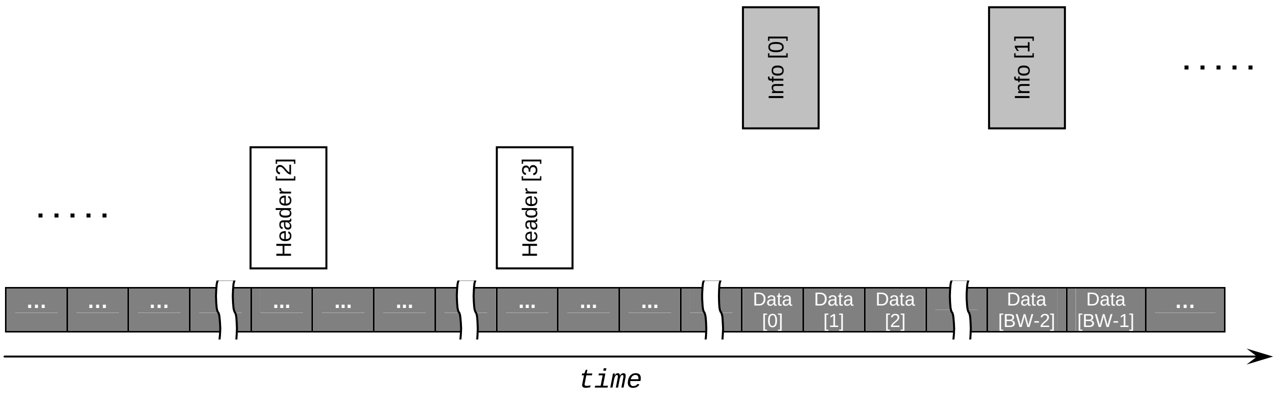

Figure 3-2: Streaming Data with Additional SADs (Byte-by-Byte View)

3.1.1 Embedded Information (Info Bytes)

SAD1 delivers the encrypted, flexible-length info field, which is carrying the ‘Embedded Information’. Info[0] indicates the number of info bytes following. Info [1] indicates the media type. The usage and mapping of ‘Embedded Information’ to Info [2]….Info [M] depends on the type and content of the stream and is specified in the document ‘MOST Stream Transmission’.

Please note:
Generally, unused info bytes are reserved and must be transmitted as ‘0x00’.

3.1.2 Generic MOST-DTCP Packet Length

A packet always consists of an unprotected header channel (SAD0), the protected info channel (SAD1) and a variable number of protected data channels (Byte[2..BW-1]).

Please note:
The protected channels (SAD1 / Byte[2..BW-1]) may be subdivided into several ‘Encryption frames’.

MOST-DTCP Packet Length = (k * EFS * BW) / (BW-1)
MOST-DTCP Packet Length = n * BW

k, n : Element of N

EFS: Encryption Frame Size as defined in ‘Supplement B’ of the 5C DTCP Specification

Based on the formula above, the ‘MOST Stream Transmission’ document defines for each supported stream the Encryption Frame Size and the MOST-DTCP Packet Length.

3.2 Definition of Header Bytes

This section describes the definition of the DTCP header bytes used in the previous chapter.

Name	**Purpose**	**MSB**							**LSB**
Header [0]	SyncHi 0x3C	0	0	1	1	1	1	0	0
Header [1]	SyncLo 0xB2	1	0	1	1	0	0	1	0
Header [2]	DTCP information	Defined by ‘Supplement B’ of DTCP specification							
Header [3]	Extension	Reserved, set to '0x00'							

Table 3-1: Definitions of the Header Bytes

The used sync pattern 0x3CB2 is a 4th order PN-series with a length of 15 bits, padded with a zero bit.

3.3 Example

Content: 'Audio of DVD-Audio' (MediaType 0x22)

Resulting MOST BlockWidth: 6 bytes per frame (2 bytes Info & 4 bytes LPCM)

DTCP Encryption Frame Size: 8 bytes
MOST Packet Length: 48 bytes

(Parameters taken from the 'MOST Stream Transmission specification 1.1')

Frame	Byte [0]	Byte [1]	Byte [2]	Byte [3]	Byte [4]	Byte [5]
1	0x3C	(0x00)	Data [16]	Data [17]	Data [18]	Data [19]
2	0xB2	(0x00)	Data [20]	Data [21]	Data [22]	Data [23]
3	DTCP	(0x00)	Data [24]	Data [25]	Data [26]	Data [27]
4	0x00	(0x00)	Data [28]	Data [29]	Data [30]	Data [31]
5	(0x00)	0x07	Data [0]	Data [1]	Data [2]	Data [3]
6	(0x00)	0x22	Data [4]	Data [5]	Data [6]	Data [7]
7	(0x00)	CCI	Data [8]	Data [9]	Data [10]	Data [11]
8	(0x00)	ISRC	Data [12]	Data [13]	Data [14]	Data [15]
9	0x3C	ISRC	Data [16]	Data [17]	Data [18]	Data [19]
10	0xB2	(0x00)	Data [20]	Data [21]	Data [22]	Data [23]
11	DTCP	(0x00)	Data [24]	Data [25]	Data [26]	Data [27]
12	0x00	(0x00)	Data [28]	Data [29]	Data [30]	Data [31]
13	(0x00)	0x07	Data [0]	Data [1]	Data [2]	Data [3]
14	(0x00)	0x22	Data [4]	Data [5]	Data [6]	Data [7]
15	(0x00)	CCI	Data [8]	Data [9]	Data [10]	Data [11]
16	(0x00)	ISRC	Data [12]	Data [13]	Data [14]	Data [15]

Figure 3-3: DVD-Audio Stream Protected by MOST-DTCP

4 Message Sequence Charts

The following dynamic specification is an implementation recommendation. There may exist valid reasons in particular circumstances to ignore a particular item, to change its detailed behavior or to add items, etc. However, the full implications (e.g. interoperability) must be understood and carefully weighted before choosing a different course.

4.1 Overview

In the example collaboration diagram, a complete DTCP connection establishment (Authentication followed by a Content Key Exchange) is figured out.

For reasons of clarity, requests and responses are merged, if possible (without referring to the relevant OP Types). Otherwise, the communication is outlined by explicitly using the OP Types.

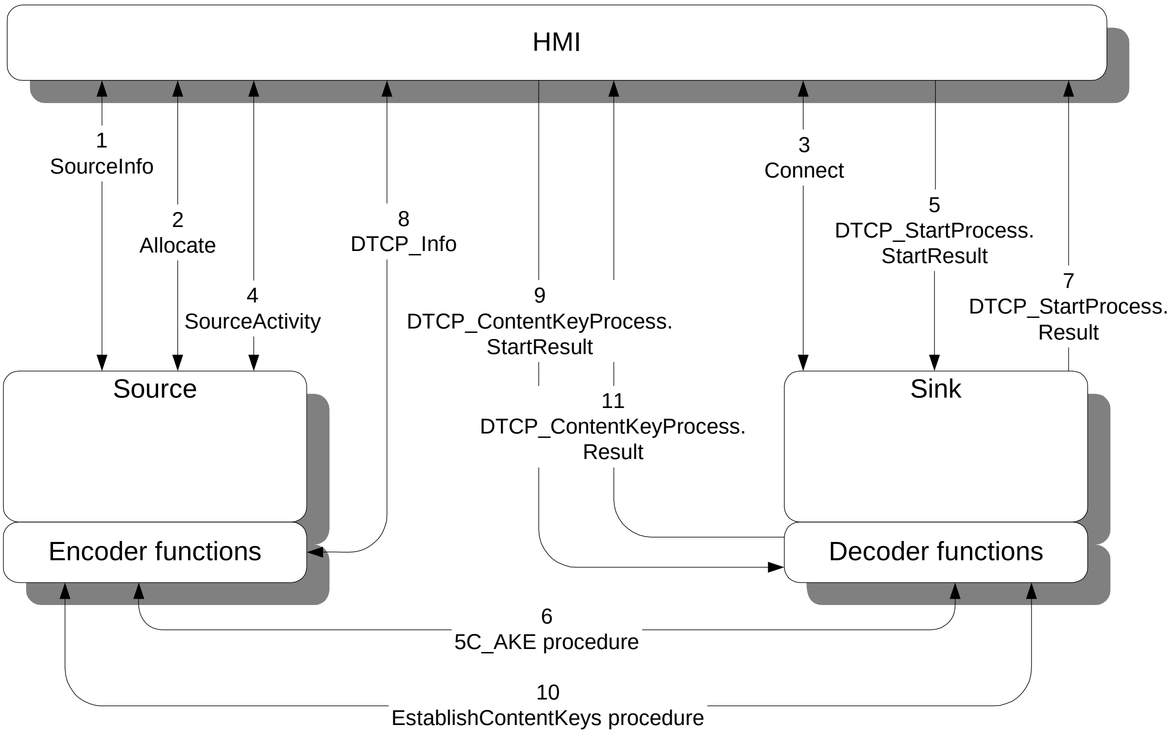

Figure 4-1: Collaboration Diagram 1: DTCP Connection Establishment (Authentication Followed by a Content Key Exchange)

4.2 Speculative Authentication

Use Case:	Speculative DTCP Authentication		
Description:	The Speculative DTCP Authentication takes place during the start of the MOST Network		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:	Speculative DTCP Authentication is only an optional element at this time.		

Table 4-1: MSC 1 Speculative Authentication

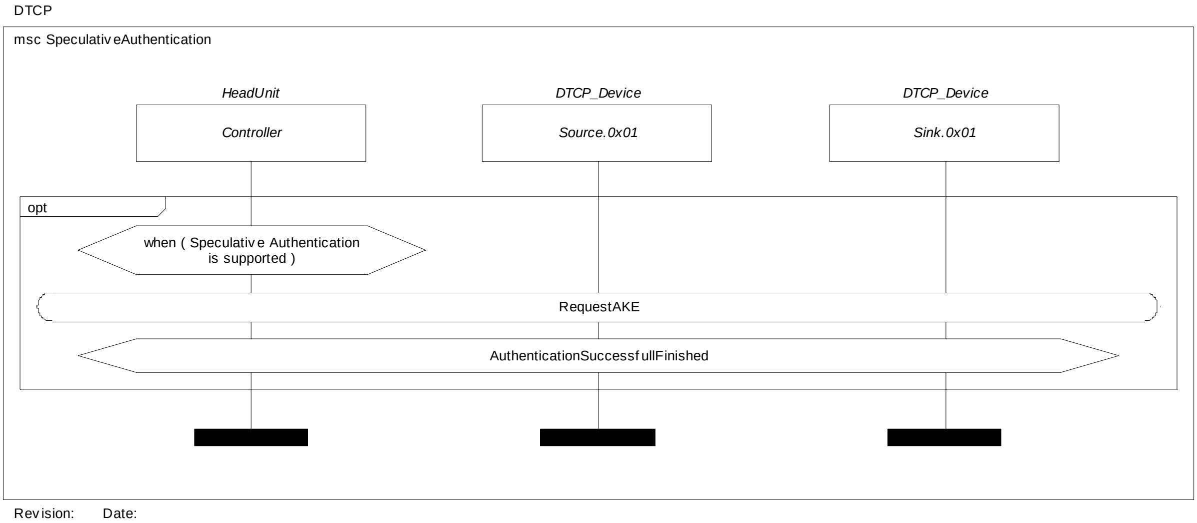
DTCP
msc SpeculativeAuthentication
HeadUnit
Controller
DTCP_Device
Source.0x01
DTCP_Device
Sink.0x01
opt
when (Speculative Authentication is supported)
RequestAKE
AuthenticationSuccessfullFinished
Revision: Date:

Figure 4-2: MSC 1 Speculative Authentication

4.3 The User Requests a DTCP Audio Connection

Use Case:	The passenger requests a DTCP audio connection		
Description:	The passenger initiates the establishment of a DTCP audio connection. If necessary, the DTCP Authentication Process is executed, before the Content Keys are calculated.		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
	X		
Remarks:			

Table 4-2: MSC 2 The User Requests a DTCP Audio Connection

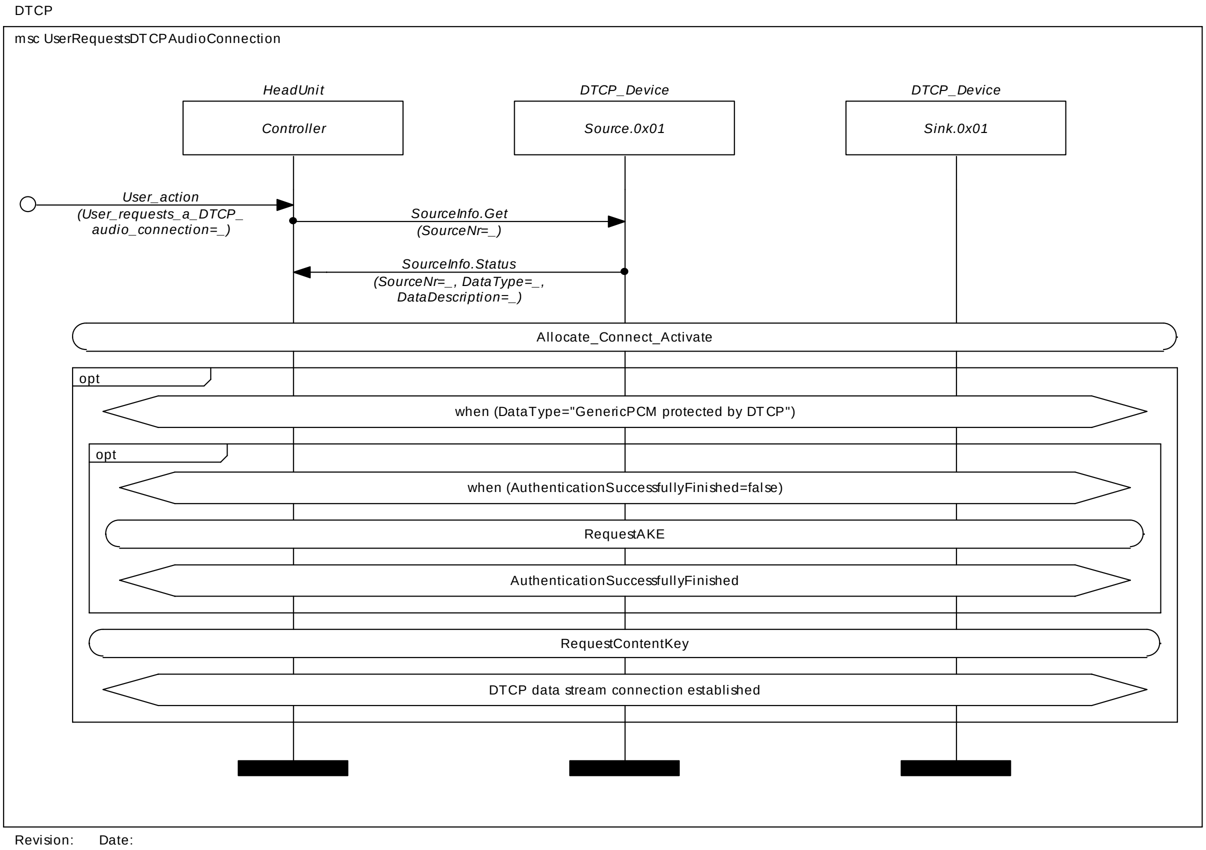

Figure 4-3: MSC 2 The User Requests a DTCP Audio Connection

4.4 Request Exchange Key Calculation

Use Case:	Request for calculating the Exchange Keys		
Description:	The HeadUnit starts the DTCP Authentication Procedure and therefore initiates the calculation of the Exchange Keys.		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:	The AKE is done on device-level		

Table 4-3: MSC 3 Request Exchange Key Calculation

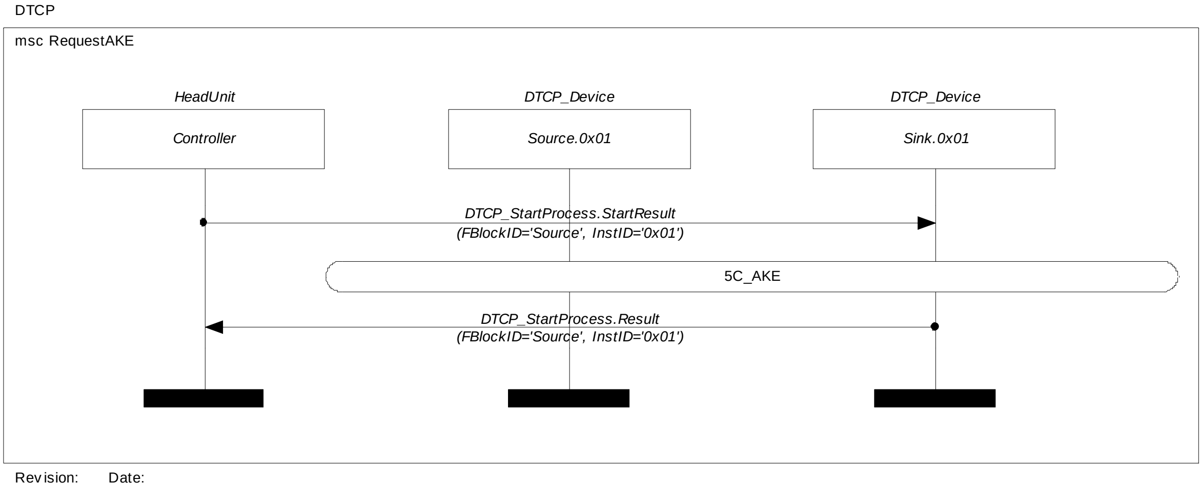

Figure 4-4: MSC 3 Request Exchange Key Calculation

4.5 Request Content Key Calculation

Use Case:	Request for calculating the Content Keys		
Description:	The HeadUnit initiates the calculation of the Content Keys. To do so, it delivers audio stream relevant information of the source device to the sink.		
Prior Condition:	The requested audio connection is to be protected in accordance to the DTCP specification		
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:	The establishing of Content Keys is done on “SourceNr”-level		

Table 4-4: MSC 4 Request for Calculating the Content Keys

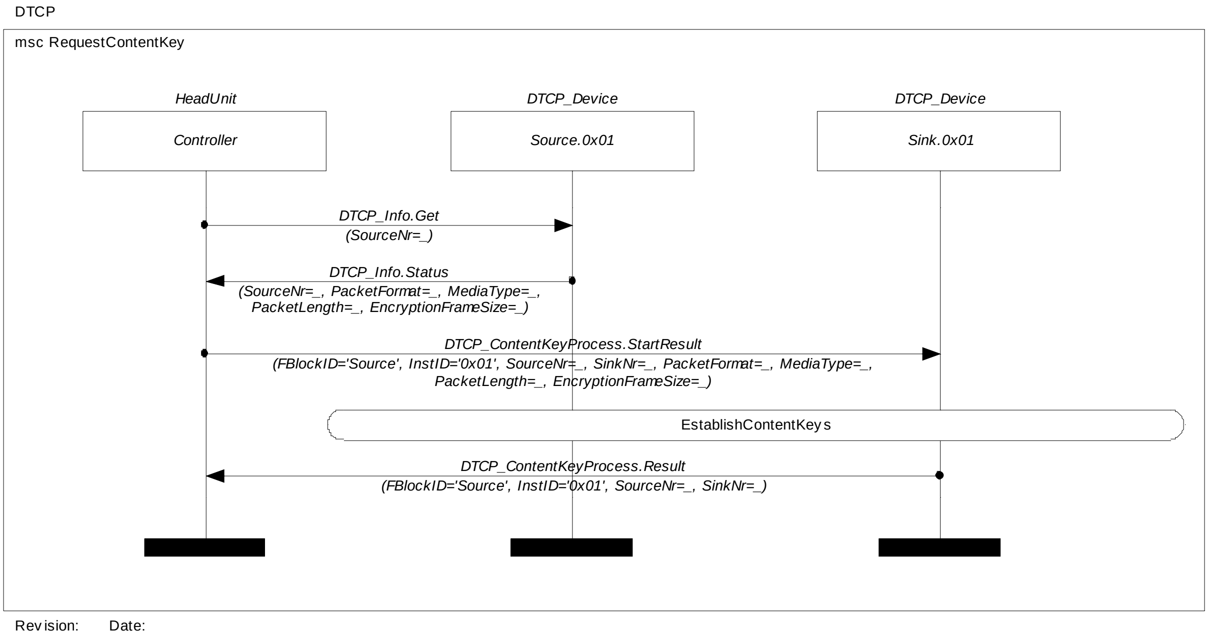
DTCP
msc RequestContentKey
HeadUnit
Controller
DTCP_Device
Source.0x01
DTCP_Device
Sink.0x01
DTCP_Info.Get
(SourceNr=_)
DTCP_Info.Status
(SourceNr=_, PacketFormat=_, MediaType=_,
PacketLength=_, EncryptionFrameSize=_)
DTCP_ContentKeyProcess.StartResult
(FBlockID='Source', InstID='0x01', SourceNr=_, SinkNr=_, PacketFormat=_, MediaType=_,
PacketLength=_, EncryptionFrameSize=_)
EstablishContentKeys
DTCP_ContentKeyProcess.Result
(FBlockID='Source', InstID='0x01', SourceNr=_, SinkNr=_)
Revision:
Date:

Figure 4-5: MSC 4 Request for Calculating the Content Keys

4.6 Allocate, Connect and Activate

Use Case:	Allocate, connect and activate		
Description:	The HeadUnit allocates synchronous timeslots on the MOST Network, connects the audio source and sink to it and optionally activates the output of audio data.		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:			

Table 4-5: MSC 5 Allocate, Connect and Activate

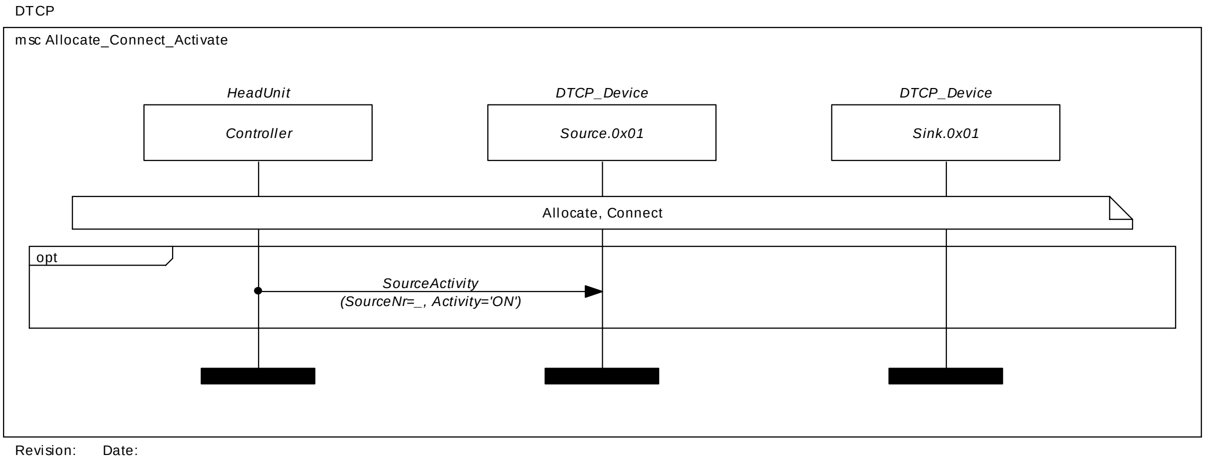

Figure 4-6: MSC 5 Allocate, Connect and Activate

4.7 Calculate Exchange Key (Example)

Use Case:	Calculate the Exchange Keys		
Description:	Calculate the Exchange Keys in accordance to the DTCP Specification Vol.1 (Informational Version)		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:	In this example, all three Exchange Keys are calculated		

Table 4-6: MSC 6 Calculate the Exchange Key

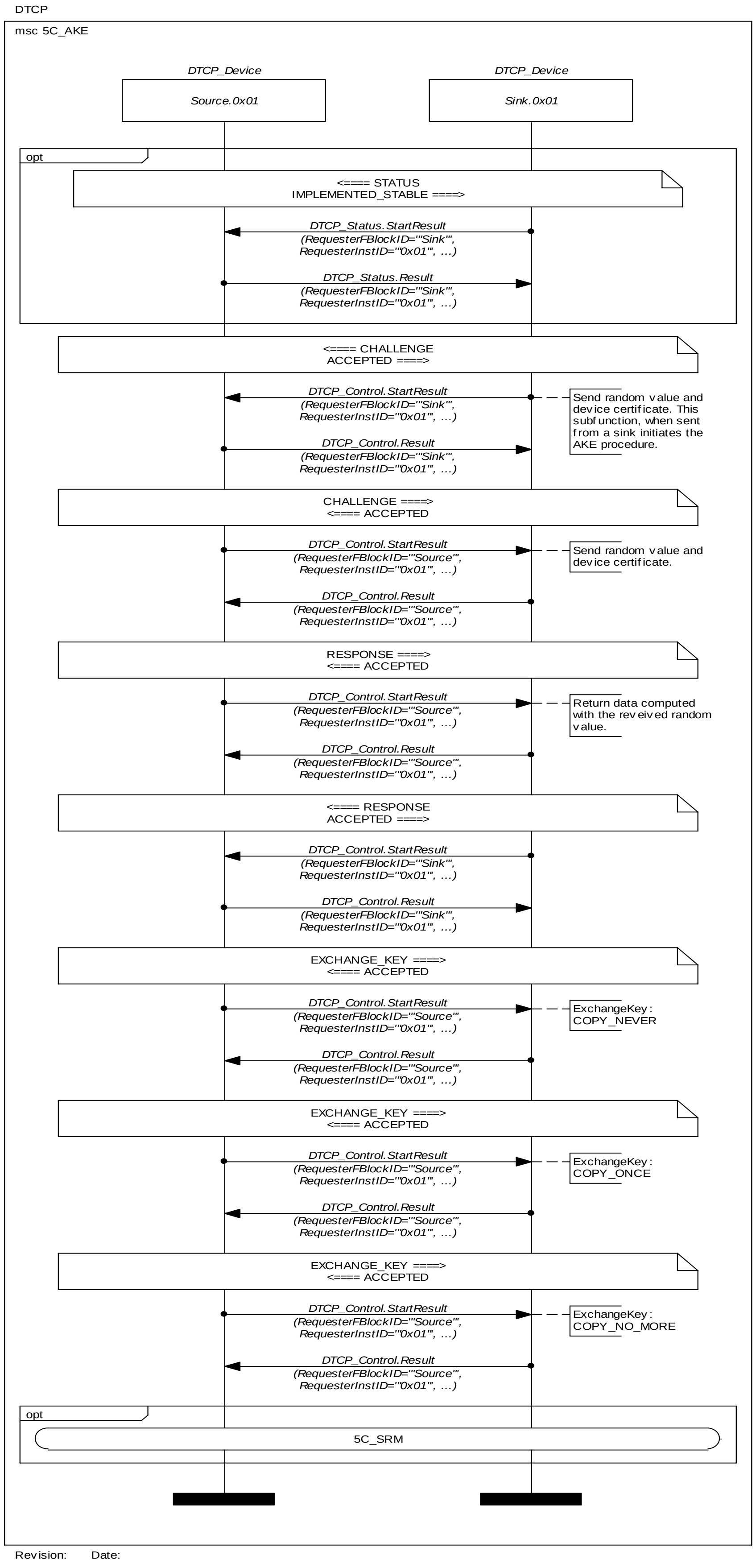
DTCP
msc 5C_AKE
DTCP_Device
Source.0x01
DTCP_Device
Sink.0x01
opt
<==== STATUS
IMPLEMENTED_STABLE ====>
DTCP_Status.StartResult
(RequesterFBlockID="'Sink'",
RequesterInstID="'0x01'", ...)
DTCP_Status.Result
(RequesterFBlockID="'Sink'",
RequesterInstID="'0x01'", ...)
<==== CHALLENGE
ACCEPTED ====>
DTCP_Control.StartResult
(RequesterFBlockID="'Sink'",
RequesterInstID="'0x01'", ...)
DTCP_Control.Result
(RequesterFBlockID="'Sink'",
RequesterInstID="'0x01'", ...)
Send random value and device certificate. This subfunction, when sent from a sink initiates the AKE procedure.
CHALLENGE ====>
<==== ACCEPTED
DTCP_Control.StartResult
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
DTCP_Control.Result
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
Send random value and device certificate.
RESPONSE ====>
<==== ACCEPTED
DTCP_Control.StartResult
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
DTCP_Control.Result
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
Return data computed with the reveived random value.
<==== RESPONSE
ACCEPTED ====>
DTCP_Control.StartResult
(RequesterFBlockID="'Sink'",
RequesterInstID="'0x01'", ...)
DTCP_Control.Result
(RequesterFBlockID="'Sink'",
RequesterInstID="'0x01'", ...)
EXCHANGE_KEY ====>
<==== ACCEPTED
DTCP_Control.StartResult
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
DTCP_Control.Result
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
ExchangeKey:
COPY_NEVER
EXCHANGE_KEY ====>
<==== ACCEPTED
DTCP_Control.StartResult
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
DTCP_Control.Result
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
ExchangeKey:
COPY_ONCE
EXCHANGE_KEY ====>
<==== ACCEPTED
DTCP_Control.StartResult
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
DTCP_Control.Result
(RequesterFBlockID="'Source'",
RequesterInstID="'0x01'", ...)
ExchangeKey:
COPY_NO_MORE
opt
5C_SRM
Revision: Date:

Figure 4-7: MSC 6 Calculate the Exchange Key

4.8 Establish Content Keys

Use Case:	Establish the Content Keys		
Description:	Establish the Content Keys in accordance to the DTCP Specification Vol.1 (Informational Version)		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:			

Table 4-7: MSC 7 Establish the Content Keys

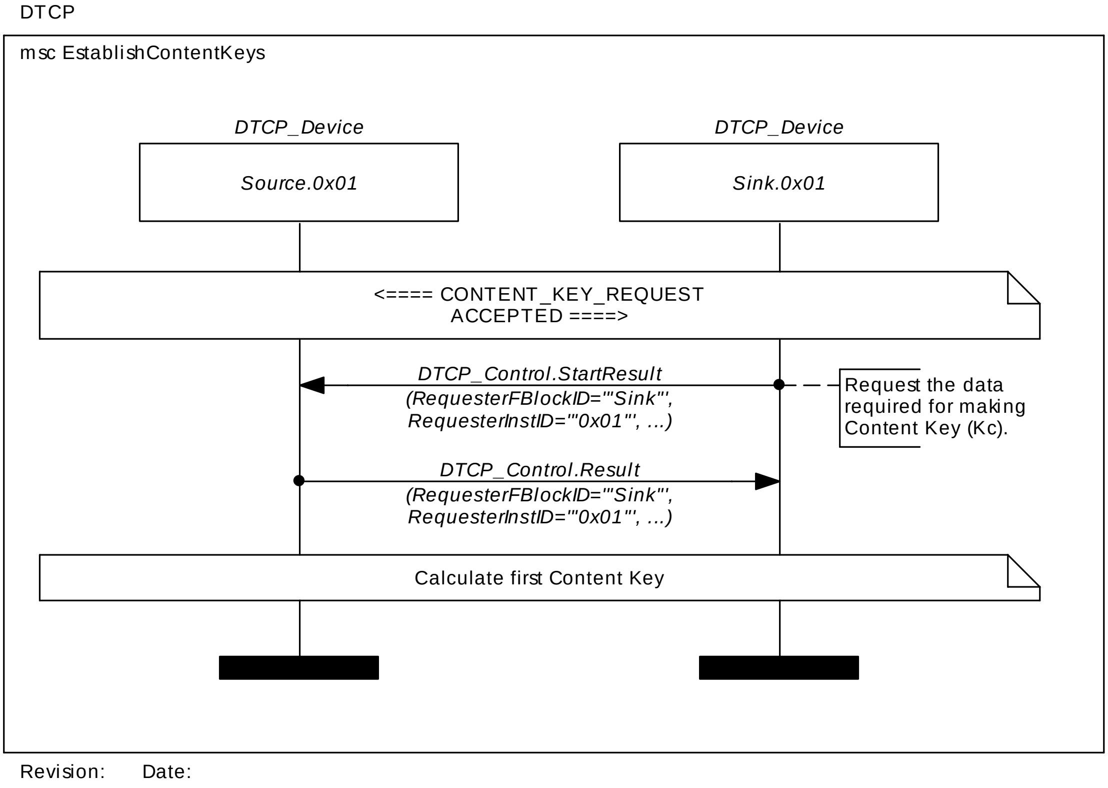
DTCP
msc EstablishContentKeys
DTCP_Device
Source.0x01
DTCP_Device
Sink.0x01
<==== CONTENT_KEY_REQUEST ACCEPTED ====>
DTCP_Control.StartResult (RequesterFBlockID="'Sink"', RequesterInstID="'0x01"', ...)
Request the data required for making Content Key (Kc).
DTCP_Control.Result (RequesterFBlockID="'Sink"', RequesterInstID="'0x01"', ...)
Calculate first Content Key
Revision: Date:

Figure 4-8: MSC 7 Establish the Content Keys

4.9 SRM

Use Case:	SRM update		
Description:	SRM update in accordance to the DTCP Specification Vol.1 (Informational Version)		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:			

Table 4-8: MSC 8 5C_SRM

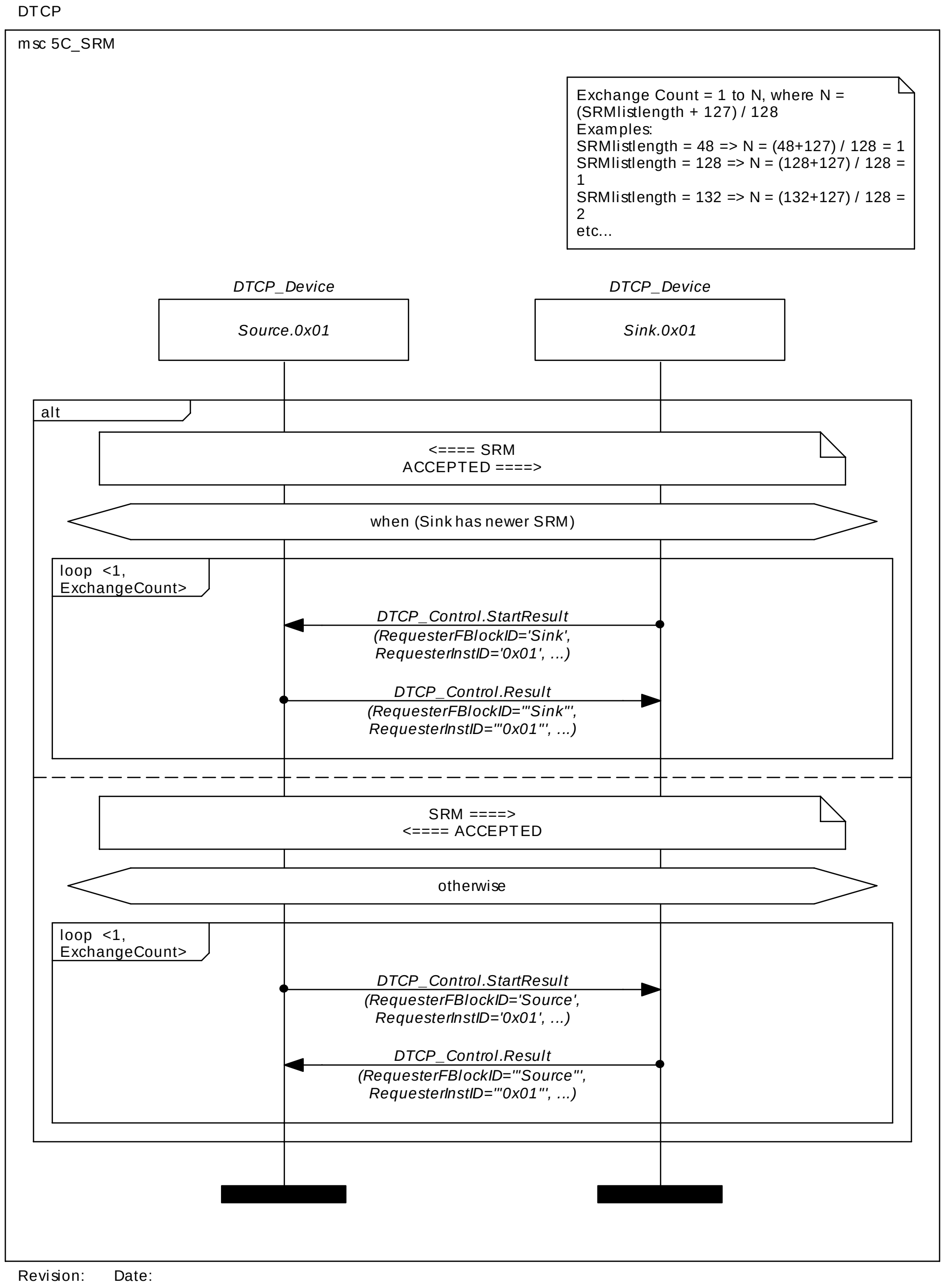
DTCP
msc 5C_SRM
Exchange Count = 1 to N, where N = (SRMlistlength + 127) / 128
Examples:
SRMlistlength = 48 => N = (48+127) / 128 = 1
SRMlistlength = 128 => N = (128+127) / 128 = 1
SRMlistlength = 132 => N = (132+127) / 128 = 2
etc...
DTCP_Device
Source.0x01
DTCP_Device
Sink.0x01
alt
<==== SRM ACCEPTED ====>
when (Sink has newer SRM)
loop <1, ExchangeCount>
DTCP_Control.StartResult (RequesterFBlockID='Sink', RequesterInstID='0x01', ...)
DTCP_Control.Result (RequesterFBlockID='''Sink''', RequesterInstID='''0x01''', ...)
SRM ====> <==== ACCEPTED
otherwise
loop <1, ExchangeCount>
DTCP_Control.StartResult (RequesterFBlockID='Source', RequesterInstID='0x01', ...)
DTCP_Control.Result (RequesterFBlockID='''Source''', RequesterInstID='''0x01''', ...)
Revision: Date:

Figure 4-9: MSC 8 5C_SRM

4.10 Error Handling: Software Error of the Source Device

Use Case:	Software error of the source device		
Description:	A software error of the source device occurs.		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:			

Table 4-9: MSC 9 Software Error of the Source Device

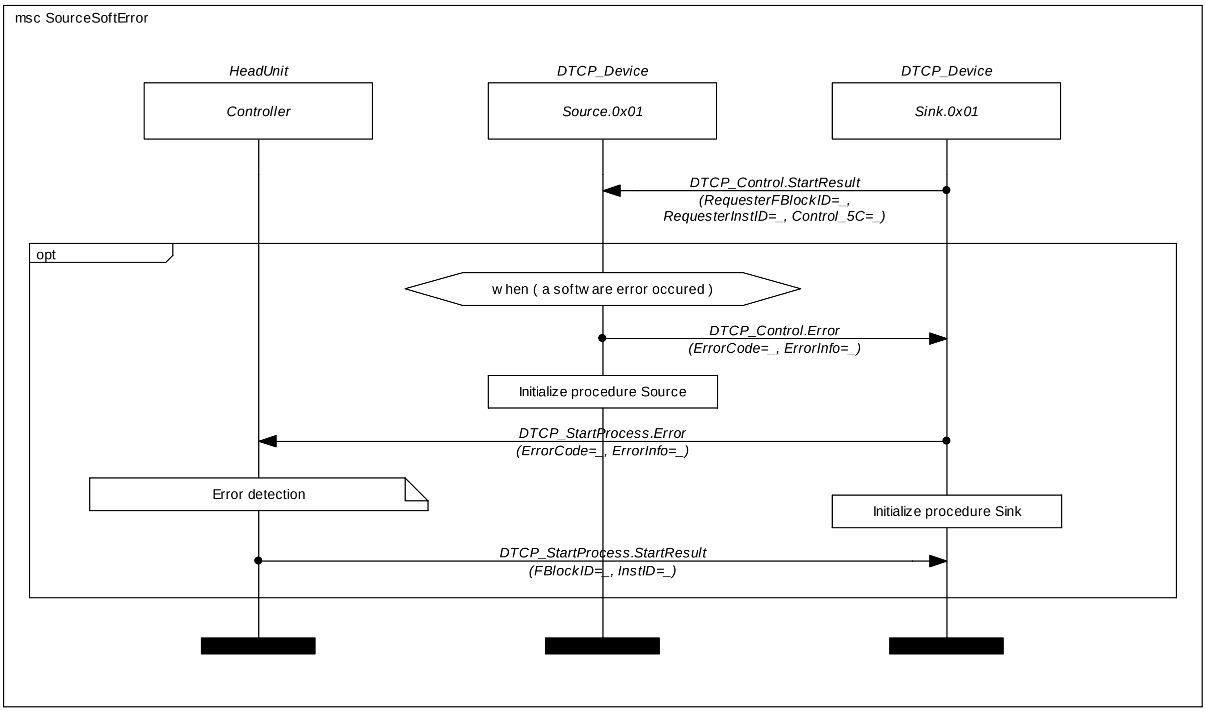
msc SourceSoftError
HeadUnit
Controller
DTCP_Device
Source.0x01
DTCP_Device
Sink.0x01
DTCP_Control.StartResult (RequesterFBlockID=_, RequesterInstID=_, Control_5C=_)
opt
when (a software error occured)
DTCP_Control.Error (ErrorCode=_, ErrorInfo=_)
Initialize procedure Source
DTCP_StartProcess.Error (ErrorCode=_, ErrorInfo=_)
Error detection
Initialize procedure Sink
DTCP_StartProcess.StartResult (FBlockID=_, InstID=_)

Figure 4-10: MSC 9 Software Error of the Source Device

4.11 Error Handling: Software Error of the Sink Device

Use Case:	Software error of the sink device		
Description:	A software error of the sink device occurs.		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:			

Table 4-10: MSC 10 Software Error of the Sink Device

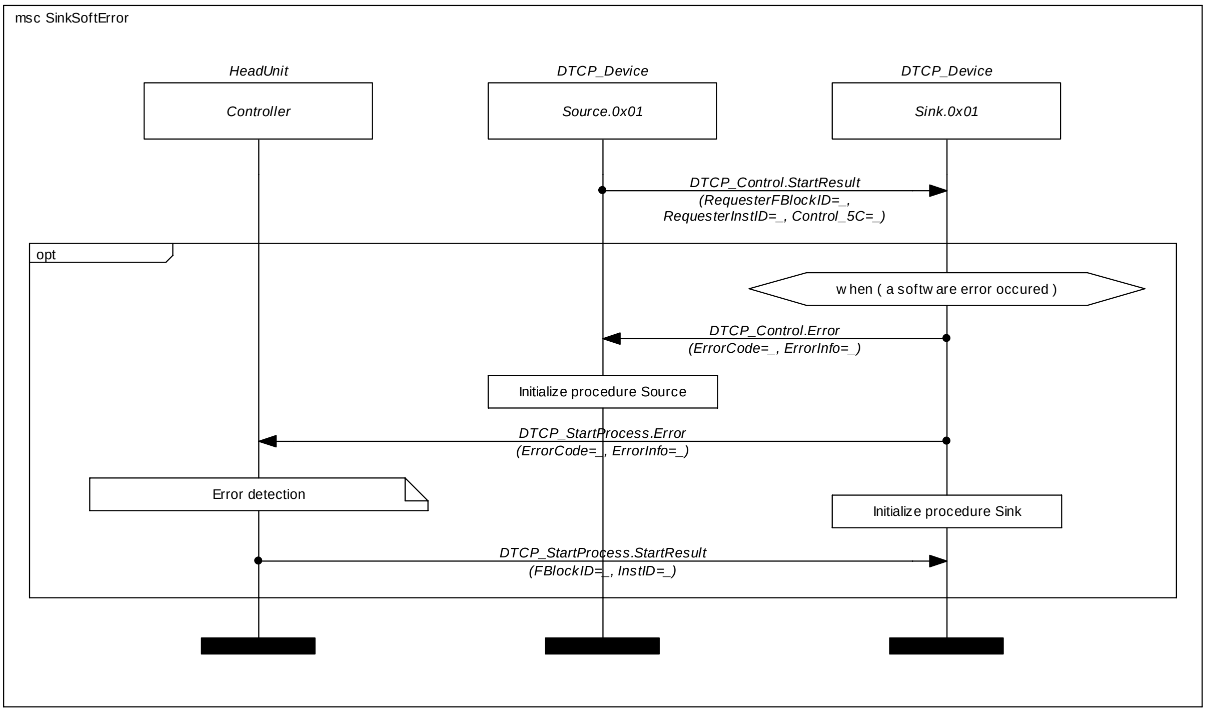

Figure 4-11: MSC 10 Software Error of the Sink Device

4.12 Error Handling: Hardware Error of the Source Device

Use Case:	Hardware error of the source device		
Description:	A hardware error of the source device occurs.		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:			

Table 4-11: MSC 11 Hardware Error of the Source Device

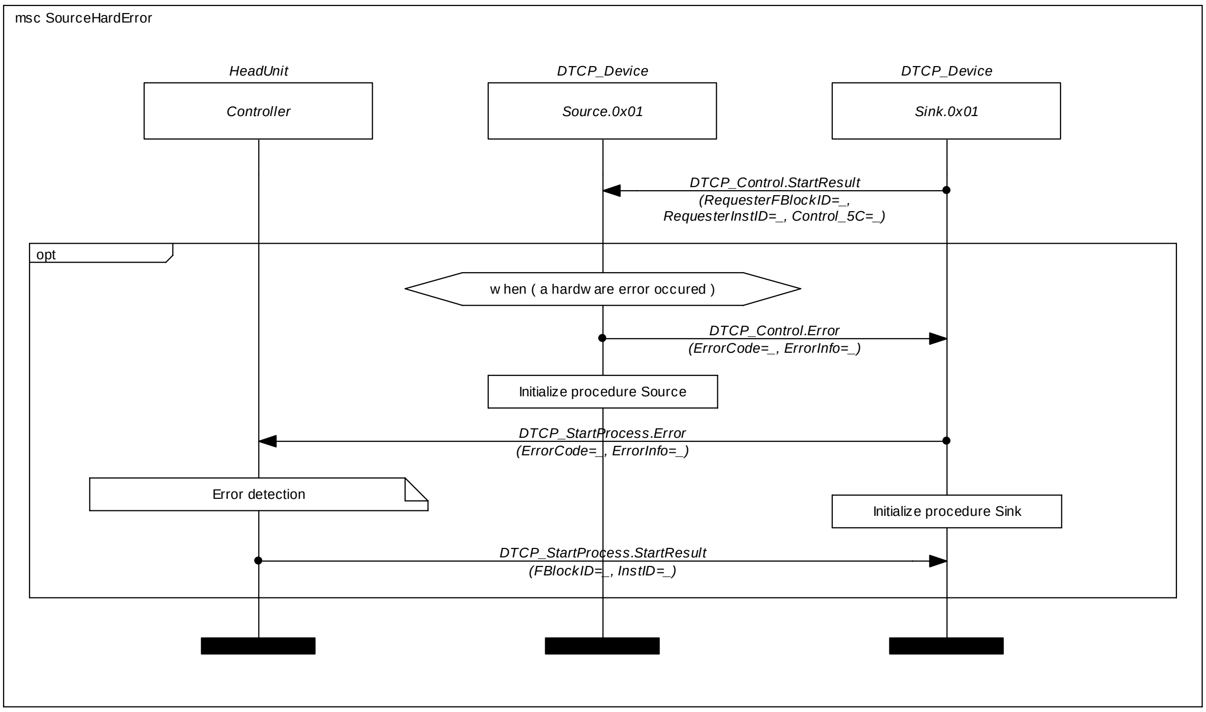

Figure 4-12: MSC 11 Hardware Error of the Source Device

4.13 Error Handling: Hardware Error of the Sink Device

Use Case:	Hardware error of the sink device		
Description:	A hardware error of the sink device occurs.		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:			

Table 4-12: MSC 12 Hardware Error of the Sink Device

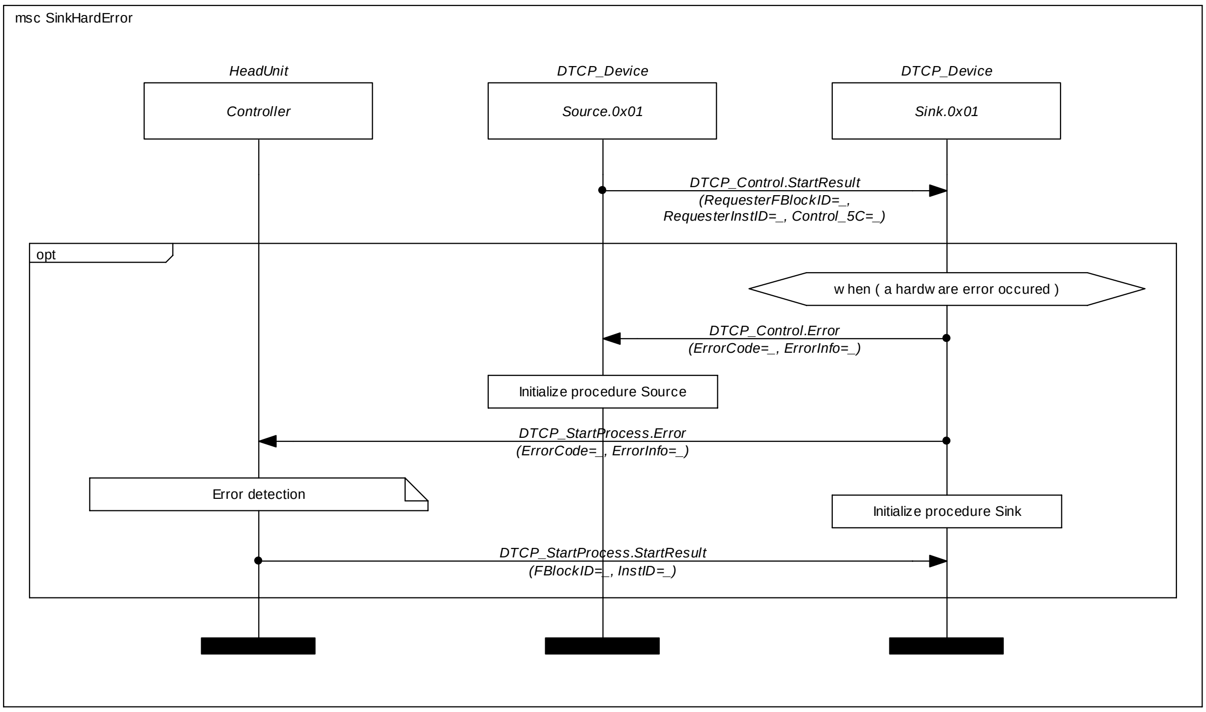

Figure 4-13: MSC 12 Hardware Error of the Sink Device

4.14 Error Handling: Decode Error of the Sink Device

Use Case:	Decode error of the sink device		
Description:	A decode error of the sink device occurs, while the sink receives decoded data.		
Prior Condition:			
Initiator:	**Passenger**	**Internal**	**Comment**
		X	
Remarks:			

Table 4-13: MSC 13 Decode Error of the Sink Device

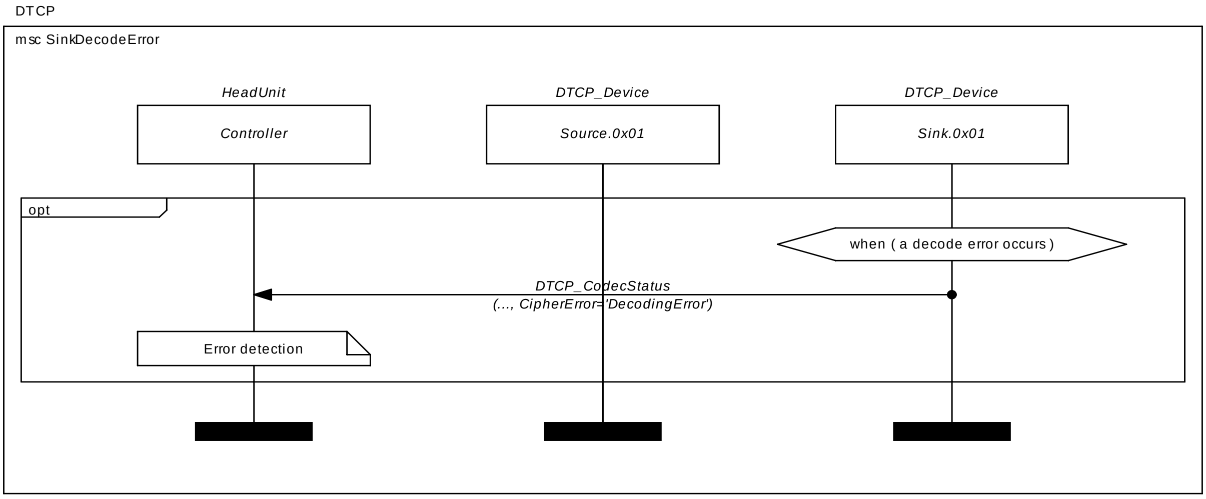

Figure 4-14: MSC 13 Decode Error of the Sink Device

Appendix A: List of Figures

Appendix B: List of Tables