

# **High-bandwidth Digital Content Protection System**

## **Mapping HDCP to DisplayPort**

Revision 2.2

21 December, 2012

## Notice

THIS DOCUMENT IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Intel Corporation disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

The cryptographic functions described in this specification may be subject to export control by the United States, Japanese, and/or other governments.

Copyright © 1999-2012 by Intel Corporation. Third-party brands and names are the property of their respective owners.

## Acknowledgement

STMicroelectronics and Parade Technologies have contributed to the development of this specification.

## Intellectual Property

Implementation of this specification requires a license from the Digital Content Protection LLC.

### Contact Information

Digital Content Protection LLC  
C/O Vital Technical Marketing, Inc.  
3855 SW 153rd Drive  
Beaverton, OR 97006

Email: [info@digital-cp.com](mailto:info@digital-cp.com)

Web: [www.digital-cp.com](http://www.digital-cp.com)

## Revision History

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Scope	5
1.2	Definitions	5
1.3	Overview	8
1.4	Terminology	9
1.5	References	9
<b>2</b>	<b>Authentication Protocol</b>	<b>10</b>
2.1	Overview	10
2.2	Authentication and Key Exchange	11
2.2.1	Pairing	14
2.3	Locality Check	15
2.4	Session Key Exchange	16
2.5	Authentication with Repeaters	17
2.5.1	Upstream Propagation of Topology Information	17
2.5.2	Downstream Propagation of Content Stream Management Information	22
2.6	Link Integrity Check	23
2.6.1	Link Integrity Check in MST Mode	23
2.6.2	Link Integrity Check in SST Mode	24
2.7	Key Derivation	25
2.8	HDCP Transmitter State Diagram	25
2.9	HDCP Receiver State Diagram	30
2.10	HDCP Repeater State Diagrams	32
2.10.1	Propagation of Topology Errors	33
2.10.2	HDCP Repeater Downstream State Diagram	33
2.10.3	HDCP Repeater Upstream State Diagram	38
2.11	Converters	42
2.11.1	HDCP 2 – HDCP 1.x Converters	42
2.11.2	HDCP 1.x – HDCP 2 Converters	44
2.12	Session Key Validity	45
2.13	Random Number Generation	45
2.14	CP_IRQ Interrupt Processing	46
2.15	HDCP Port	46
<b>3</b>	<b>HDCP Encryption</b>	<b>51</b>
3.1	Data Encryption	51
3.2	HDCP Cipher	54
3.3	Encryption Status Signaling in MST Mode	56
3.4	Encryption Status Signaling in SST Mode	58
3.5	Uniqueness of $k_s$ and $r_{iv}$	60
<b>4</b>	<b>Authentication Protocol Messages</b>	<b>62</b>
4.1	Overview	62
4.2	Message Format	62
4.2.1	AKE_Init (Write)	62
4.2.2	AKE_Send_Cert (Read)	62
4.2.3	AKE_No_Stored_km (Write)	62
4.2.4	AKE_Stored_km (Write)	62
4.2.5	AKE_Send_H_prime (Read)	62
4.2.6	AKE_Send_Pairing_Info (Read)	63
4.2.7	LC_Init (Write)	63
4.2.8	LC_Send_L_prime (Read)	63
4.2.9	SKE_Send_Eks (Write)	63
4.2.10	RepeaterAuth_Send_ReceiverID_List (Read)	63

4.2.11	RepeaterAuth_Send_Ack (Write).....	64
4.2.12	RepeaterAuth_Stream_Manage (Write).....	64
4.2.13	RepeaterAuth_Stream_Ready (Read) .....	65
<b>5</b>	<b>Renewability .....</b>	<b>66</b>
5.1	SRM Size and Scalability.....	67
5.2	Updating SRMs.....	68
<b>Appendix A.</b>	<b>Core Functions and Confidentiality and Integrity of Values ....</b>	<b>70</b>
<b>Appendix B.</b>	<b>DCP LLC Public Key.....</b>	<b>73</b>
<b>Appendix C.</b>	<b>Bibliography (Informative).....</b>	<b>74</b>

## 1 Introduction

### 1.1 Scope

This specification describes the mapping of High-bandwidth Digital Content Protection (HDCP) system to DisplayPort, Revision 2.20.

For the purpose of this specification, it is assumed that the Audiovisual content is transmitted over a DisplayPort based wired display link. In an HDCP System, two or more HDCP Devices are interconnected through an HDCP-protected Interface. The Audiovisual Content flows from the Upstream Content Control Function into the HDCP System at the most upstream HDCP Transmitter. From there the Audiovisual Content encrypted by the HDCP System, referred to as HDCP Content, flows through a tree-shaped topology of HDCP Receivers over HDCP-protected Interfaces. This specification describes a content protection mechanism for: (1) authentication of HDCP Receivers to their immediate upstream connection (i.e., an HDCP Transmitter), (2) revocation of HDCP Receivers that are determined by the Digital Content Protection, LLC, to be invalid, and (3) HDCP Encryption of Audiovisual Content over the HDCP-protected Interfaces between HDCP Transmitters and their downstream HDCP Receivers. HDCP Receivers may render the HDCP Content in audio and visual form for human consumption. HDCP Receivers may be HDCP Repeaters that serve as downstream HDCP Transmitters emitting the HDCP Content further downstream to one or more additional HDCP Receivers.

Unless otherwise specified, the term “HDCP Receiver” is also used to refer to the upstream HDCP-protected interface port of an HDCP Repeater. Similarly, the term “HDCP Transmitter” is also used to refer to the downstream HDCP-protected interface port of an HDCP Repeater. HDCP Transmitters must support HDCP Repeaters.

The state machines in this specification define the required behavior of HDCP Devices. The link-visible behavior of HDCP Devices implementing the specified state machines must be identical, even if implementations differ from the descriptions. The behavior of HDCP Devices implementing the specified state machines must also be identical from the perspective of an entity outside of the HDCP System.

Implementations must include all elements of the content protection system described herein, unless the element is specifically identified as informative or optional. Adopters must also ensure that implementations satisfy the robustness and compliance rules described in the technology license.

Device discovery and association, and link setup and teardown, is outside the scope of this specification.

### 1.2 Definitions

The following terminology, as used throughout this specification, is defined as herein:

**Audiovisual Content.** Audiovisual works (as defined in the United States Copyright Act as in effect on January 1, 1978), text and graphic images, are referred to as *AudioVisual Content*.

**Authorized Device.** An HDCP Device that is permitted access to HDCP Content is referred to as an *Authorized Device*. An HDCP Transmitter may test if a connected HDCP Receiver is an Authorized Device by successfully completing the following stages of the authentication protocol – Authentication and Key Exchange (AKE) and Locality check. If the authentication protocol successfully results in establishing authentication, then the other device is considered by the HDCP Transmitter to be an Authorized Device.

**Content Stream.** *Content Stream* consists of Audiovisual Content received from an Upstream Content Control Function that is to be encrypted and Audiovisual Content received from an Upstream Content Control Function that is encrypted by the HDCP System.

**Device Key Set.** An HDCP Receiver has a Device Key Set, which consists of its corresponding Device Secret Keys along with the associated Public Key Certificate.

**Device Secret Keys.** For an HDCP Transmitter, Device Secret Key consists of the secret Global Constant. For an HDCP Receiver, Device Secret Keys consists of the secret Global Constant and the RSA private key. The Device Secret Keys are to be protected from exposure outside of the HDCP Device.

**downstream.** The term, *downstream*, is used as an adjective to refer to being towards the sink of the HDCP Content. For example, when an HDCP Transmitter and an HDCP Receiver are connected over an HDCP-protected Interface, the HDCP Receiver can be referred to as the *downstream* HDCP Device in this connection. For another example, on an HDCP Repeater, the HDCP-protected Interface Port(s) which can emit HDCP Content can be referred to as its *downstream* HDCP-protected Interface Port(s). See also, *upstream*.

**Global Constant.** A 128-bit random, secret constant provided only to HDCP adopters and used during HDCP Content encryption or decryption

**HDCP 1.x.** *HDCP 1.x* refers to, specifically, the variant of HDCP described by Revision 1.00 (referred to as HDCP 1.0), Revision 1.10 (referred to as HDCP 1.1), Revision 1.20 (referred to as HDCP 1.2) and Revision 1.30 (referred to as HDCP 1.3) along with their associated errata, if applicable.

**HDCP 1.x-compliant Device.** An HDCP Device that is designed in adherence to HDCP 1.x, defined above, is referred to as an *HDCP 1.x-compliant Device*.

**HDCP 2.** *HDCP 2* refers to, specifically, the variant of HDCP mapping for all HDCP protected interfaces described by Revision 2.00 and higher versions along with their associated errata, if applicable.

**HDCP 2.0.** *HDCP 2.0* refers to, specifically, the variant of HDCP mapping for all HDCP protected interfaces described by Revision 2.00 of the corresponding specifications along with their associated errata, if applicable.

**HDCP 2.0-compliant Device.** An HDCP Device that is designed in adherence to HDCP 2.0 is referred to as an *HDCP 2.0-compliant Device*.

**HDCP 2.2.** *HDCP 2.2* refers to, specifically, the variant of HDCP mapping described by Revision 2.20 of this specification along with its associated errata, if applicable.

**HDCP 2.2-compliant Device.** An HDCP Device that is designed in adherence to HDCP 2.2 is referred to as an *HDCP 2.2-compliant Device*.

**HDCP Cipher.** The HDCP encryption module consisting of a 128-bit AES module that is operated in a Counter (CTR) mode is referred to as *HDCP Cipher*.

**HDCP Content.** *HDCP Content* consists of Audiovisual Content that is protected by the HDCP System. *HDCP Content* includes the Audiovisual Content in encrypted form as it is transferred from an HDCP Transmitter to an HDCP Receiver over an HDCP-protected Interface, as well as any translations of the same content, or portions thereof. For avoidance of doubt, Audiovisual Content that is never encrypted by the HDCP System is not *HDCP Content*.

**HDCP Device.** Any device that contains one or more HDCP-protected Interface Port and is designed in adherence to HDCP is referred to as an *HDCP Device*.

**HDCP Encryption.** *HDCP Encryption* is the encryption technology of HDCP when applied to the protection of HDCP Content in an HDCP System.

**HDCP Receiver.** An HDCP Device that can receive and decrypt HDCP Content through one or more of its HDCP-protected Interface Ports is referred to as an *HDCP Receiver*.

**HDCP Repeater.** An HDCP Device that can receive and decrypt HDCP Content through one or more of its HDCP-protected Interface Ports, and can also re-encrypt and emit said HDCP Content through one or more of its HDCP-protected Interface Ports, is referred to as an *HDCP Repeater*. An *HDCP Repeater* may also be referred to as either an HDCP Receiver or an HDCP Transmitter when referring to either the upstream side or the downstream side, respectively.

**HDCP Session.** An *HDCP Session* is established between an HDCP Transmitter and HDCP Receiver with the transmission or reception of the authentication initiation message, AKE\_Init. The established HDCP Session remains valid until it is aborted by the HDCP Transmitter or a new HDCP Session is established, which invalidates the HDCP Session that was previously established, by the transmission or reception of a new AKE\_Init message.

**HDCP System.** An *HDCP System* consists of an HDCP Transmitter, zero or more HDCP Repeaters and one or more HDCP Receivers connected through their HDCP-protected interfaces in a tree topology; whereas the said HDCP Transmitter is the HDCP Device most upstream, and receives the Audiovisual Content from one or more Upstream Content Control Functions. All HDCP Devices connected to other HDCP Devices in an *HDCP System* over HDCP-protected Interfaces are part of the *HDCP System*.

**HDCP Transmitter.** An HDCP Device that can encrypt and emit HDCP Content through one or more of its HDCP-protected Interface Ports is referred to as an *HDCP Transmitter*.

**HDCP.** *HDCP* is an acronym for High-bandwidth Digital Content Protection. This term refers to this content protection system as described by any revision of this specification and its errata.

**HDCP-protected Interface Port.** A connection point on an HDCP Device that supports an HDCP-protected Interface is referred to as an *HDCP-protected Interface Port*.

**HDCP-protected Interface.** An interface for which HDCP applies is described as an *HDCP-protected Interface*.

**Master Key.** A 128-bit random, secret cryptographic key negotiated between the HDCP Transmitter and the HDCP Receiver during Authentication and Key Exchange and used to pair the HDCP Transmitter with the HDCP Receiver.

**Public Key Certificate.** Each HDCP Receiver is issued a Public Key Certificate signed by DCP LLC, and contains the Receiver ID and RSA public key corresponding to the HDCP Receiver.

**Receiver Connected Indication.** An indication to the HDCP Transmitter that an active receiver has been connected to it. The format of the indication or the method used by the HDCP Transmitter to connect to or disconnect from a receiver is outside the scope of this specification.

**Receiver Disconnected Indication.** An indication to the HDCP Transmitter that the receiver has been disconnected from it. The format of the indication or the method used by the HDCP Transmitter to connect to or disconnect from a receiver is outside the scope of this specification.

**Receiver ID.** A 40-bit value that uniquely identifies the HDCP Receiver. It has the same format as an HDCP 1.x KSV i.e. it contains 20 ones and 20 zeroes.

**Session Key.** A 128-bit random, secret cryptographic key negotiated between the HDCP Transmitter and the HDCP Receiver during Session Key exchange and used during HDCP Content encryption or decryption.

**Upstream Content Control Function.** The HDCP Transmitter most upstream in the HDCP System receives Audiovisual Content to be protected from the *Upstream Content Control Function*. The *Upstream Content Control Function* is not part of the HDCP System, and the methods used, if any, by the *Upstream Content Control Function* to determine for itself the HDCP System is correctly authenticated or permitted to receive the Audiovisual Content, or to transfer the Audiovisual Content to the HDCP System, are beyond the scope of this specification. On a personal computer platform, an example of an *Upstream Content Control Function* may be software designed to emit Audiovisual Content to a display or other presentation device that requires HDCP.

**upstream.** The term, *upstream*, is used as an adjective to refer to being towards the source of the HDCP Content. For example, when an HDCP Transmitter and an HDCP Receiver are connected over an HDCP-protected Interface, the HDCP Transmitter can be referred to as the *upstream* HDCP Device in this connection. For another example, on an HDCP Repeater, the HDCP-protected Interface Port(s) which can receive HDCP Content can be referred to as its *upstream* HDCP-protected Interface Port(s). See also, *downstream*.

### 1.3 Overview

1. HDCP is designed to protect the transmission of Audiovisual Content between an HDCP Transmitter and an HDCP Receiver. The HDCP Transmitter may support simultaneous connections to HDCP Receivers through one or more of its HDCP-protected interface ports. The system also allows for HDCP Repeaters that support downstream HDCP-protected Interface Ports. The HDCP System allows up to four levels of HDCP Repeaters and as many as 32 total HDCP Devices, including HDCP Repeaters, to be connected to an HDCP-protected Interface port.

Figure 1.1 illustrates an example connection topology for HDCP Devices.

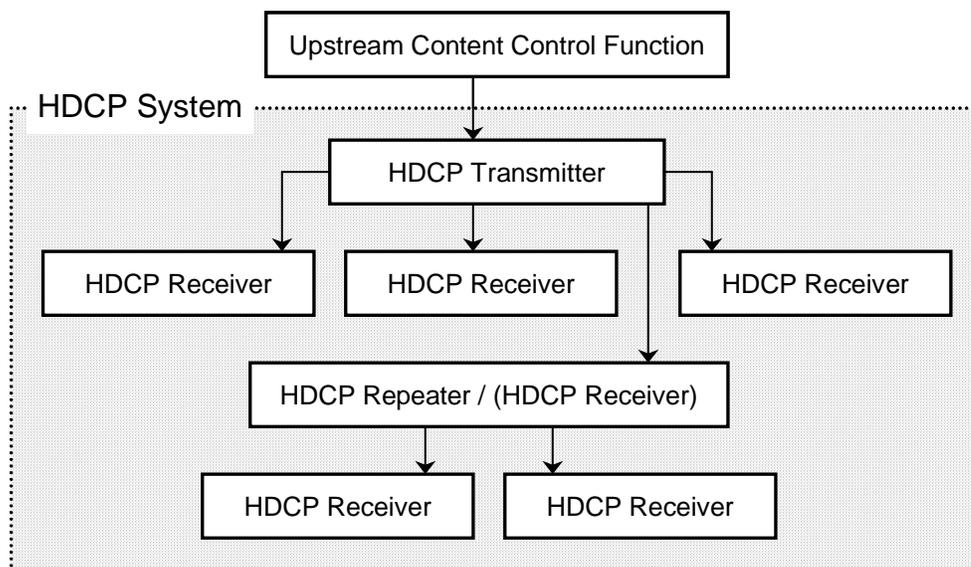


Figure 1.1. Sample Connection Topology of an HDCP System

There are three elements of the content protection system. Each element plays a specific role in the system. First, there is the authentication protocol, through which the HDCP Transmitter verifies that a given HDCP Receiver is licensed to receive HDCP Content. The authentication protocol is implemented between the HDCP Transmitter and its corresponding downstream HDCP Receiver. With the legitimacy of the HDCP Receiver determined, encrypted HDCP Content is transmitted between the two devices based on shared secrets established during the authentication protocol. This prevents eavesdropping devices from utilizing the content. Finally, in the event that legitimate devices are compromised to permit unauthorized use of HDCP Content, renewability allows an HDCP Transmitter to identify such compromised devices and prevent the transmission of HDCP Content.

This document contains chapters describing in detail the requirements of each of these elements. In addition, a chapter is devoted to describing the cipher structure that is used in the encryption of HDCP Content.

## 1.4 Terminology

Throughout this specification, names that appear in *italic* refer to values that are exchanged during the HDCP cryptographic protocol. C-style notation is used throughout the state diagrams and protocol diagrams, although the logic functions AND, OR, and XOR are written out where a textual description would be more clear.

This specification uses the big-endian notation to represent bit strings so that the most significant bit in the representation is stored in the left-most bit position. The concatenation operator ‘||’ combines two values into one. For eight-bit values  $a$  and  $b$ , the result of  $(a || b)$  is a 16-bit value, with the value  $a$  in the most significant eight bits and  $b$  in the least significant eight bits.

## 1.5 References

- [1]. Digital Content Protection (DCP) LLC, High-bandwidth Digital Content Protection System, Revision 1.4, July 8, 2009.
- [2]. Digital Content Protection (DCP) LLC, HDCP Specification 1.3 – Amendment for DisplayPort, Revision 1.0, December 19, 2006.
- [3]. National Institute of Standards and Technology (NIST), *Advanced Encryption Standard (AES)*, FIPS Publication 197, November 26, 2001.
- [4]. RSA Laboratories, *RSA Cryptography Standard*, PKCS #1 v2.1, June 14, 2002.
- [5]. National Institute of Standards and Technology (NIST), *Secure Hash Standard (SHS)*, FIPS Publication 180-2, August 1, 2002.
- [6]. Internet Engineering Task Force (IETF), *HMAC: Keyed-Hashing for Message Authentication*, Request for Comments (RFC) 2104, February 1997.
- [7]. National Institute of Standards and Technology (NIST), *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, Special Publication 800-90, March 2007
- [8]. VESA®DisplayPort®Standard, Version 1, Revision 2a, May 23, 2012

## 2 Authentication Protocol

### 2.1 Overview

The HDCP authentication protocol is an exchange between an HDCP Transmitter and an HDCP Receiver that affirms to the HDCP Transmitter that the HDCP Receiver is authorized to receive HDCP Content. It is comprised of the following stages

- Authentication and Key Exchange (AKE) – The HDCP Receiver’s public key certificate is verified by the HDCP Transmitter. A Master Key  $k_m$  is exchanged.
- Locality Check – The HDCP Transmitter enforces locality on the content by requiring that the Round Trip Time (RTT) between a pair of messages is not more than 7 ms.
- Session Key Exchange (SKE) – The HDCP Transmitter exchanges Session Key  $k_s$  with the HDCP Receiver.
- Authentication with Repeaters – The step is performed by the HDCP Transmitter only with HDCP Repeaters. In this step, the repeater assembles downstream topology information and forwards it to the upstream HDCP Transmitter.

Successful completion of AKE and locality check stages affirms to the HDCP Transmitter that the HDCP Receiver is authorized to receive HDCP Content. At the end of the authentication protocol, a communication path is established between the HDCP Transmitter and HDCP Receiver that only Authorized Devices can access.

All HDCP Devices contain a 128-bit secret Global Constant denoted by  $lc_{128}$ . All HDCP Devices share the same Global Constant.  $lc_{128}$  is provided only to HDCP adopters.

The HDCP Transmitter contains the 3072-bit RSA public key of DCP LLC denoted by  $kpub_{dcp}$ .

The HDCP Receiver is issued 1024-bit RSA public and private keys. The public key is stored in a Public Key Certificate issued by DCP LLC, denoted by  $cert_{rx}$ . Table 2.1 gives the fields contained in the certificate. All values are stored in big-endian format.

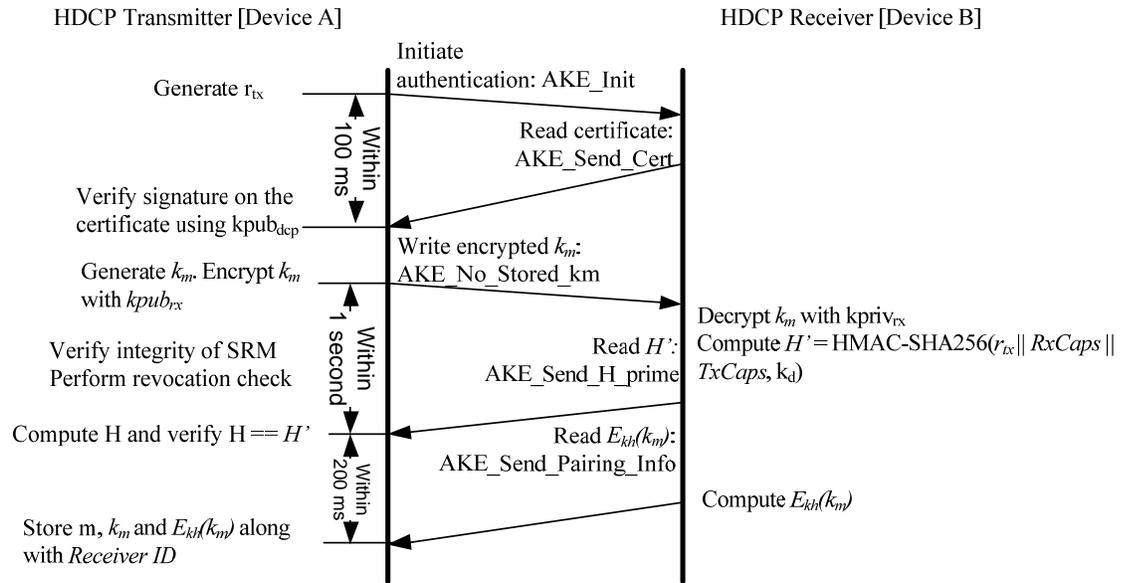
Name	Size (bits)	Bit position	Function
Receiver ID	40	4175:4136	Unique receiver identifier. It has the same format as an HDCP 1.x KSV i.e. it contains 20 ones and 20 zeroes
Receiver Public Key	1048	4135:3088	Unique RSA public key of HDCP Receiver denoted by $kpub_{rx}$ . The first 1024 bits is the big-endian representation of the modulus n and the trailing 24 bits is the big-endian representation of the public exponent e
Reserved2	4	3087:3084	Reserved for future definition. Must be 0x0 or 0x1.
Reserved1	12	3083:3072	Reserved for future definition. Must be 0x000
DCP LLC Signature	3072	3071:0	A cryptographic signature calculated over all preceding fields of the certificate. RSASSA-PKCS1-v1_5 is the signature scheme used as defined by PKCS #1 V2.1: RSA Cryptography Standard. SHA-256 is the underlying hash function

**Table 2.1. Public Key Certificate of HDCP Receiver**

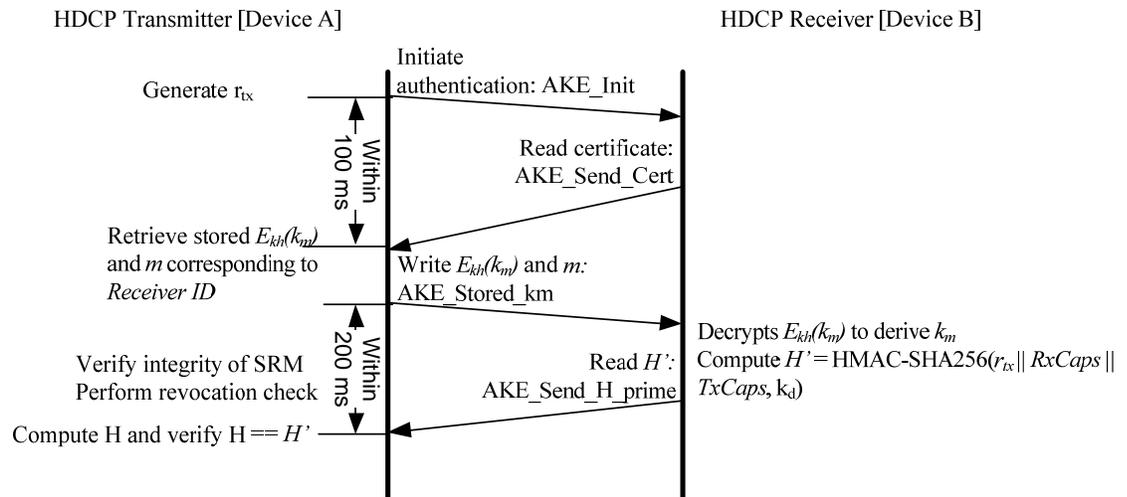
The secret RSA private key is denoted by  $kpriv_{rx}$ . The computation time of RSA private key operation can be reduced by using the Chinese Remainder Theorem (CRT) technique. Therefore, it is recommended that HDCP Receivers use the CRT technique for private key computations.

## 2.2 Authentication and Key Exchange

Authentication and Key Exchange (AKE) is the first step in the authentication protocol. Figure 2.1 and Figure 2.2 illustrates the AKE. The HDCP Transmitter (*Device A*) can initiate authentication at any time, even before a previous authentication exchange has completed. The HDCP Transmitter initiates a new HDCP Session by sending the authentication initiation message, *AKE\_Init*. Message formats are defined in Section 4.2.



**Figure 2.1. Authentication and Key Exchange (Without Stored  $k_m$ )**



**Figure 2.2. Authentication and Key Exchange (With Stored  $k_m$ )**

The HDCP Transmitter

- Initiates authentication by sending the initiation message, AKE\_Init, containing a 64-bit pseudo-random value ( $r_{tx}$ ) and  $TxCaps$  parameters.
- Reads AKE\_Send\_Cert from the receiver containing  $cert_{rx}$ , a 64-bit pseudo-random value ( $r_{rx}$ ) and  $RxCaps$ . REPEATER bit in  $RxCaps$  indicates whether the connected receiver is an HDCP Repeater. If REPEATER is set to one, it indicates the receiver is an HDCP Repeater. If REPEATER is zero, the receiver is not an HDCP Repeater. The AKE\_Send\_Cert message must be available for the transmitter to read within 100 ms from the time the transmitter finishes writing the AKE\_Init message parameters to the HDCP Receiver. The HDCP Transmitter must not attempt to read AKE\_Send\_Cert sooner than 100 ms after writing the AKE\_Init message. If the AKE\_Send\_Cert message is not available for the transmitter to read within 100 ms, the transmitter aborts the authentication protocol.
- Extracts *Receiver ID* from  $cert_{rx}$ 
  - If the HDCP Transmitter does not have a 128-bit Master Key  $k_m$  stored corresponding to the *Receiver ID* (See Section 2.2.1)
    - Verifies the signature on the certificate using  $k_{pub_{dcp}}$ . Failure of signature verification constitutes an authentication failure and the HDCP Transmitter aborts the authentication protocol.
    - Generates a pseudo-random 128-bit Master Key  $k_m$ . Encrypts  $k_m$  with  $k_{pub_{rx}}$  ( $E_{k_{pub}}(k_m)$ ) and sends AKE\_No\_Stored\_km message to the receiver containing the 1024-bit  $E_{k_{pub}}(k_m)$ . RSAES-OAEP encryption scheme must be used as defined by PKCS #1 V2.1: RSA Cryptography Standard. SHA-256 is the underlying hash function. The mask generation function used is MGF1 which uses SHA-256 as its underlying hash function.
    - Verifies integrity of the System Renewability Message (SRM). It does this by checking the signature of the SRM using  $k_{pub_{dcp}}$ . Failure of this integrity check constitutes an authentication failure and causes the HDCP Transmitter to abort authentication protocol.

The top-level HDCP Transmitter checks to see if the *Receiver ID* of the connected device is found in the revocation list. If the *Receiver ID* of the connected HDCP Device is found in the revocation list, authentication fails and the authentication protocol is aborted. SRM integrity check and revocation check are performed only by the top-level HDCP Transmitter.

- Performs key derivation as explained in Section 2.7 to generate 256-bit  $k_d$ .  $k_d = dkey_0 \parallel dkey_1$ , where  $dkey_0$  and  $dkey_1$  are derived keys generated when  $ctr = 0$  and  $ctr = 1$  respectively.  $dkey_0$  and  $dkey_1$  are in big-endian order.
- Computes 256-bit  $H = \text{HMAC-SHA256}(r_{tx} \parallel RxCaps \parallel TxCaps, k_d)$  where HMAC-SHA256 is computed over  $r_{tx} \parallel RxCaps \parallel TxCaps$  and the key used for HMAC is  $k_d$ .
- Reads the H'\_AVAILABLE status bit in the *RxStatus* register as soon as it receives the CP\_IRQ interrupt. If the H'\_AVAILABLE status bit is set, reads the AKE\_Send\_H\_prime message from the receiver

containing the 256-bit  $H'$ . The CP\_IRQ interrupt must be generated and the AKE\_Send\_H\_prime message must be available for the transmitter to read within one second from the time the transmitter finishes writing the AKE\_No\_Stored\_km message parameters to the HDCP Receiver. If the AKE\_Send\_H\_prime message is not available for the transmitter to read within one second or there is a mismatch between  $H$  and  $H'$ , the transmitter aborts the authentication protocol.

- If the HDCP Transmitter has a 128-bit Master Key  $k_m$  stored corresponding to the *Receiver ID* (See Section 2.2.1)
  - Sends AKE\_Stored\_km message to the receiver with the 128-bit  $E_{k_H}(k_m)$  and the 128-bit  $m$  corresponding to the *Receiver ID* of the HDCP Receiver
  - Verifies integrity of the System Renewability Message (SRM). It does this by checking the signature of the SRM using  $k_{pub\_dcp}$ . Failure of this integrity check constitutes an authentication failure and causes the HDCP Transmitter to abort the authentication protocol.

The top-level HDCP Transmitter checks to see if the *Receiver ID* of the connected device is found in the revocation list. If the *Receiver ID* of the connected HDCP Device is found in the revocation list, authentication fails and the authentication protocol is aborted.

- Performs key derivation as explained in Section 2.7 to generate 256-bit  $k_d$ .  $k_d = dkey_0 \parallel dkey_1$ , where  $dkey_0$  and  $dkey_1$  are derived keys generated when  $ctr = 0$  and  $ctr = 1$  respectively.  $dkey_0$  and  $dkey_1$  are in big-endian order.
- Computes 256-bit  $H = \text{HMAC-SHA256}(r_{tx} \parallel RxCaps \parallel TxCaps, k_d)$  where HMAC-SHA256 is computed over  $r_{tx} \parallel RxCaps \parallel TxCaps$  and the key used for HMAC is  $k_d$ .
- Reads the H\_AVAILABLE status bit in the *RxStatus* register as soon as it receives the CP\_IRQ interrupt. If the H\_AVAILABLE status bit is set, reads the AKE\_Send\_H\_prime message from the receiver containing the 256-bit  $H'$ . The CP\_IRQ interrupt must be generated and the AKE\_Send\_H\_prime message must be available for the transmitter to read within 200 ms from the time the transmitter finishes writing the AKE\_Stored\_km message parameters to the HDCP Receiver. If the AKE\_Send\_H\_prime message is not available for the transmitter to read within 200 ms or there is a mismatch between  $H$  and  $H'$ , the transmitter aborts the authentication protocol.

#### The HDCP Receiver

- Makes available the AKE\_Send\_Cert message for the transmitter to read in response to AKE\_Init. The AKE\_Send\_Cert message must be available for the transmitter to read within 100 ms from the time the transmitter finishes writing the AKE\_Init message parameters to the HDCP Receiver.
- If AKE\_No\_Stored\_km is received, the HDCP Receiver
  - Decrypts  $k_m$  with  $k_{priv\_rx}$  using RSAES-OAEP decryption scheme.

- Performs key derivation as explained in Section 2.7 to generate 256-bit  $k_d$ .  $k_d = dkey_0 \parallel dkey_1$ , where  $dkey_0$  and  $dkey_1$  are derived keys generated when  $ctr = 0$  and  $ctr = 1$  respectively.  $dkey_0$  and  $dkey_1$  are in big-endian order.
- Computes  $H' = \text{HMAC-SHA256}(r_{rx} \parallel RxCaps \parallel TxCaps, k_d)$ . Asserts  $H\_AVAILABLE$  bit and generates  $CP\_IRQ$  as soon as it makes  $AKE\_Send\_H\_prime$  message available for the transmitter to read. The  $AKE\_Send\_H\_prime$  message must be available for the transmitter to read within one second from the time the transmitter finishes writing the  $AKE\_No\_Stored\_km$  message parameters to the HDCP Receiver.
- If  $AKE\_Stored\_km$  is received, the HDCP Receiver
  - Computes 128-bit  $k_h = \text{SHA-256}(kpriv_{rx})[127:0]$
  - Decrypts  $E_{kh}(k_m)$  using AES with the received  $m$  as input and  $k_h$  as key in to the AES module as illustrated in Figure 2.3 to derive  $k_m$ .
  - Performs key derivation as explained in Section 2.7 to generate 256-bit  $k_d$ .  $k_d = dkey_0 \parallel dkey_1$ , where  $dkey_0$  and  $dkey_1$  are derived keys generated when  $ctr = 0$  and  $ctr = 1$  respectively.  $dkey_0$  and  $dkey_1$  are in big-endian order.
  - Computes  $H' = \text{HMAC-SHA256}(r_{rx} \parallel RxCaps \parallel TxCaps, k_d)$ . Asserts  $H\_AVAILABLE$  bit and generates  $CP\_IRQ$  as soon as it makes  $AKE\_Send\_H\_prime$  message available for the transmitter to read. The  $AKE\_Send\_H\_prime$  message must be available for the transmitter to read within 200 ms from the time the transmitter finishes writing the  $AKE\_Stored\_km$  message parameters to the HDCP Receiver.

On a decryption failure of  $k_m$  with  $kpriv_{rx}$ , the HDCP Receiver does not send  $H'$  and simply lets the timeout occur on the HDCP Transmitter.

## 2.2.1 Pairing

To speed up the AKE process, pairing must be implemented between the HDCP Transmitter and HDCP Receiver in parallel with AKE. When  $AKE\_No\_Stored\_km$  message is received from the transmitter, it is an indication to the receiver that the transmitter does not have  $k_m$  stored corresponding to the receiver. In this case, after computing  $H'$ , the HDCP Receiver

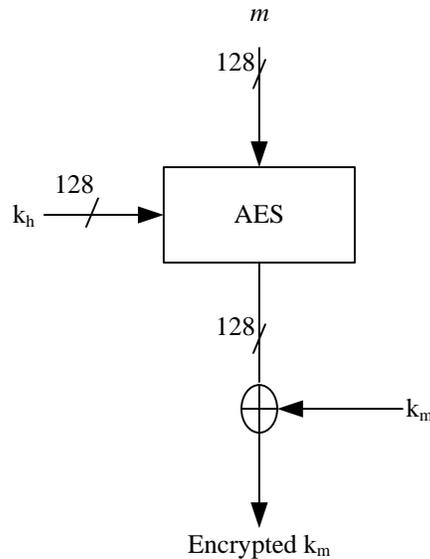
- Computes 128-bit  $k_h = \text{SHA-256}(kpriv_{rx})[127:0]$ .
- Generates 128-bit  $E_{kh}(k_m)$  by encrypting  $k_m$  with  $k_h$  using AES as illustrated in Figure 2.3.
- Asserts  $PAIRING\_AVAILABLE$  and generates  $CP\_IRQ$  when  $AKE\_Send\_Pairing\_Info$  message containing the 128-bit  $E_{kh}(k_m)$  is available for the transmitter to read. This message must be available for the transmitter to read within 200 ms from the time the transmitter begins reading the  $AKE\_Send\_H\_prime$  message parameters from the HDCP Receiver.

The transmitter reads the  $PAIRING\_AVAILABLE$  status bit in the  $RxStatus$  register as soon as it receives the  $CP\_IRQ$  interrupt. If the  $PAIRING\_AVAILABLE$  status bit is set, the transmitter reads the  $AKE\_Send\_Pairing\_Info$  message. If the  $AKE\_Send\_Pairing\_Info$  message is not available for the transmitter to read within 200 ms, authentication fails and the transmitter aborts the authentication protocol. On reading  $AKE\_Send\_Pairing\_Info$  message, the HDCP Transmitter

may persistently store  $m$  (which is  $r_{tx}$  concatenated with  $r_{rx}$  ( $r_{tx} \parallel r_{rx}$ )),  $k_m$  and  $E_{k_h}(k_m)$  along with *Receiver ID*

Note: The HDCP Transmitter may store in its non-volatile storage  $m$ ,  $k_m$  and  $E_{k_h}(k_m)$  along with corresponding *Receiver IDs* of all HDCP Receivers with which pairing was implemented by the HDCP Transmitter.

Figure 2.3 illustrates the encryption of  $k_m$  with  $k_h$ .



**Figure 2.3.  $E_{k_h}(k_m)$  Computation**

128-bit  $m$  is constructed by concatenating  $r_{tx}$  and  $r_{rx}$  ( $r_{tx} \parallel r_{rx}$ ). Both values are in big-endian order.

### 2.3 Locality Check

Locality check is performed after AKE and pairing. The HDCP Transmitter initiates locality check by sending a 64-bit pseudo-random nonce  $r_n$  to the downstream receiver.

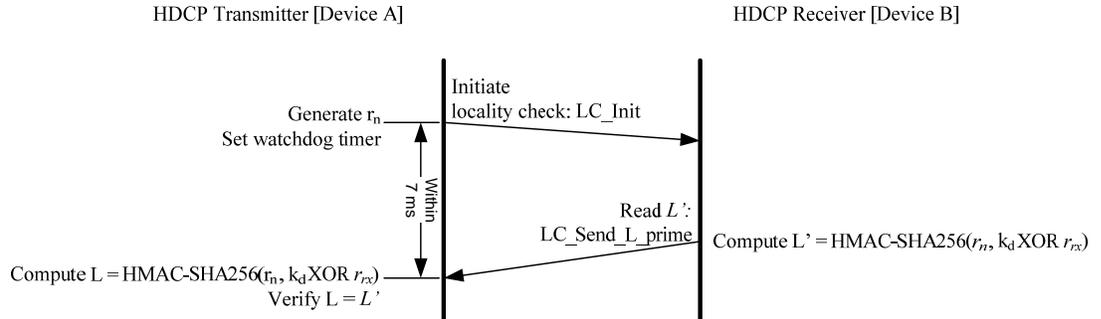
The HDCP Transmitter

- Initiates locality check by writing the LC\_Init message containing a 64-bit pseudo-random nonce  $r_n$  to the HDCP Receiver.
- Sets its watchdog timer to 7 ms. The LC\_Send\_L\_prime message must be received by the transmitter within 7 ms from the time the transmitter finishes writing the LC\_Init message parameters to the HDCP Receiver. The HDCP Transmitter must not attempt to read LC\_Send\_L\_prime sooner than 7 ms after writing the AKE\_Stored\_km message. If the LC\_Send\_L\_prime message is not available for the transmitter to read within 7 ms, locality check fails and the transmitter aborts the authentication protocol.
- Computes  $L = \text{HMAC-SHA256}(r_n, k_d \text{ XOR } r_{rx})$  where HMAC-SHA256 is computed over  $r_n$  and the key used for HMAC is  $k_d \text{ XOR } r_{rx}$ , where  $r_{rx}$  is XORed with the least-significant 64-bits of  $k_d$ .

- On reading LC\_Send\_L\_prime message from the receiver, compares L and L'. Locality check fails if L is not equal to L'.

An HDCP Repeater initiates locality check on all its downstream HDCP-protected interface ports by sending unique  $r_n$  values to the connected HDCP Devices.

Figure 2.4 illustrate locality check between the HDCP Transmitter and HDCP Receiver.



**Figure 2.4. Locality Check between HDCP Transmitter and HDCP Receiver**

The HDCP Receiver

- Computes a 256-bit value  $L' = \text{HMAC-SHA256}(r_n, k_d \text{ XOR } r_{rx})$ .
- Makes LC\_Send\_L\_prime message containing 256-bit  $L'$  available for the transmitter to read. The LC\_Send\_L\_prime message must be available for the transmitter to read within 7 ms from the time the transmitter finishes writing the LC\_Init message parameters to the HDCP Receiver.

In the case of a locality check failure due to expiration of the watchdog timer or due to mismatch of L and L' at the HDCP Transmitter, locality check may be reattempted by the HDCP Transmitter for a maximum of 1023 additional attempts (for a maximum allowed 1024 total trials) with the transmission of an LC\_Init message containing a new  $r_n$ . Failure of locality check on the first attempt and subsequent zero or more reattempts results in an authentication failure and the authentication protocol is aborted.

## 2.4 Session Key Exchange

Successful completion of AKE and locality check stages affirms to HDCP Transmitter that the HDCP Receiver is authorized to receive HDCP Content. Session Key Exchange (SKE) is initiated by the HDCP Transmitter after a successful locality check. The HDCP Transmitter sends encrypted Session Key to the HDCP Receiver at least 200 ms before enabling HDCP Encryption and beginning the transmission of HDCP Content. HDCP Encryption may be enabled 200 ms after the transmission of the encrypted Session Key to the HDCP Receiver and at no time prior. Content encrypted with the Session Key  $k_s$  starts to flow between the HDCP Transmitter and HDCP Receiver. HDCP Encryption must be enabled only after successful completion of AKE, locality check and SKE stages.

During SKE, the HDCP Transmitter

- Generates a pseudo-random 128-bit Session Key  $k_s$  and 64-bit pseudo-random number  $r_n$ .

- Performs key derivation as explained in Section 2.7 to generate 128-bit  $dkey_2$  where  $dkey_2$  is the derived key when  $ctr = 2$ .
- Computes 128-bit  $E_{dkey}(k_s) = k_s \text{ XOR } (dkey_2 \text{ XOR } r_{rx})$ , where  $r_{rx}$  is XORed with the least-significant 64-bits of  $dkey_2$ .
- Writes SKE\_Send\_Eks message containing  $E_{dkey}(k_s)$  and  $r_{iv}$  to the HDCP Receiver.

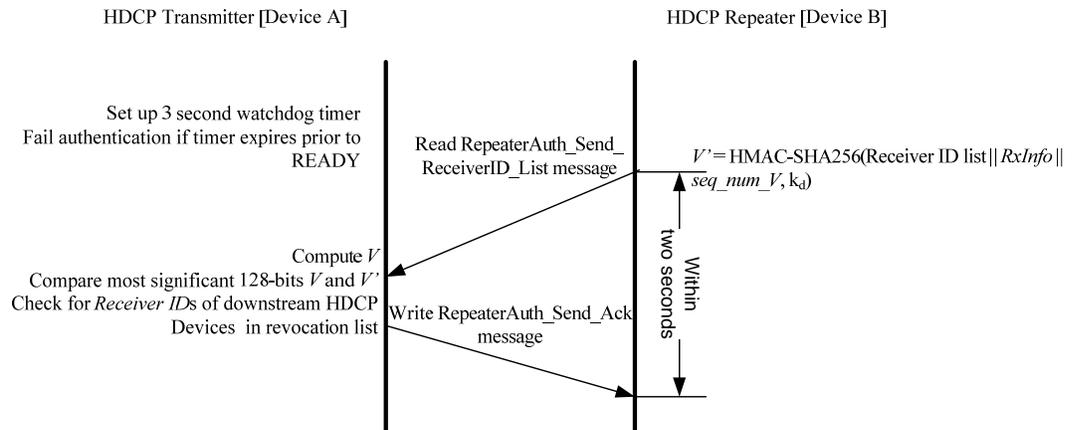
On receiving SKE\_Send\_Eks message, the HDCP Receiver

- Performs key derivation as explained in Section 2.7 to generate 128-bit  $dkey_2$  where  $dkey_2$  is the derived key when  $ctr = 2$ .
- Computes  $k_s = E_{dkey}(k_s) \text{ XOR } (dkey_2 \text{ XOR } r_{rx})$

## 2.5 Authentication with Repeaters

The HDCP Transmitter executes authentication with repeaters after Session Key exchange and only when REPEATER bit is set, indicating that the connected HDCP Receiver is an HDCP Repeater. Authentication with repeaters stage is used for the upstream propagation of topology information and the downstream propagation of Content Stream management information as explained in Section 2.5.1 and Section 2.5.2 respectively. Authentication with repeaters may be implemented by the HDCP Transmitter in parallel with the flow of encrypted content and Link Integrity Check. The Link Integrity Check process is explained in Section 2.6.

### 2.5.1 Upstream Propagation of Topology Information



**Figure 2.5. Upstream Propagation of Topology Information**

Figure 2.5 illustrates the upstream propagation of topology information. This stage assembles a list of all downstream *Receiver IDs* connected to the HDCP Repeater through a permitted connection tree, enabling revocation support upstream. This stage is implemented after successful completion of Session Key Exchange. This stage is used to assemble the latest topology information at the beginning of the HDCP Session immediately following an SKE or on subsequent changes to the topology due to connect or disconnect of an HDCP Receiver or HDCP Repeater.

HDCP Repeaters assemble the list of all connected downstream HDCP Receivers as the downstream HDCP-protected Interface Ports of the HDCP Repeater successfully complete the authentication protocol with connected HDCP Receivers. The list is represented by a contiguous set of bytes, with each *Receiver ID* occupying five bytes stored in big-endian order. The total length of the Receiver ID list is five bytes times the total number of connected and active downstream HDCP Devices, including downstream HDCP Repeaters, with which the HDCP

Repeater has successfully completed the authentication protocol. This total number is represented in the RepeaterAuth\_Send\_ReceiverID list message by the DEVICE\_COUNT value. An HDCP-protected Interface Port with no active device connected adds nothing to the list. Also, the *Receiver ID* of the HDCP Repeater itself at any level is not included in its own Receiver ID list. An HDCP-protected Interface Port connected to an HDCP Receiver that is not an HDCP Repeater adds the *Receiver ID* of the connected HDCP Receiver to the list. HDCP-protected Interface Ports that have an HDCP Repeater connected add the Receiver ID list received from the connected downstream HDCP Repeater, plus the *Receiver ID* of the connected downstream HDCP Repeater itself.

When the HDCP Repeater has assembled the complete list of *Receiver IDs* of connected and active HDCP Devices with which the HDCP Repeater has successfully completed the authentication protocol, it computes the 256-bit verification value  $V'$ .

An HDCP Repeater and an HDCP Transmitter compute respective  $V'$  and  $V$  values as given below. HMAC-SHA256 is computed over the concatenation of Receiver ID list, *RxInfo* and *seq\_num\_V* received as part of the RepeaterAuth\_Send\_ReceiverID\_List message. The key used for HMAC is  $k_d$ .

$$V' \text{ (or } V) = \text{HMAC-SHA256}(\text{Receiver ID list} \parallel \text{RxInfo} \parallel \text{seq\_num\_V}, k_d)$$

Receiver ID list is formed by appending downstream *Receiver IDs* in big-endian order. When the Receiver ID list,  $V'$ , DEPTH, DEVICE\_COUNT, HDCP2\_0\_REPEATER\_DOWNSTREAM and HDCP1\_DEVICE\_DOWNSTREAM are available, the HDCP Repeater asserts its READY status indicator in the *RxStatus* register and asserts the CP\_IRQ interrupt.

The HDCP Transmitter checks the READY bit when a CP\_IRQ interrupt is received. If READY is set, the HDCP Transmitter reads the RepeaterAuth\_Send\_ReceiverID\_List message. The HDCP Repeater makes available the most significant 128-bits of  $V'$  for the transmitter to read as part of the RepeaterAuth\_Send\_ReceiverID\_List message.

The HDCP Repeater initializes *seq\_num\_V* to 0 at the beginning of the HDCP Session i.e. after AKE\_Init is received. It is incremented by one after the transmission of every RepeaterAuth\_Send\_ReceiverID\_List message. *seq\_num\_V* must never be reused during an HDCP Session for the computation of  $V$  (or  $V'$ ). If *seq\_num\_V* rolls over, the HDCP Transmitter must detect the roll-over in the RepeaterAuth\_Send\_ReceiverID\_List read from the HDCP Repeater and the transmitter must disable HDCP Encryption if encryption is enabled, restart authentication by the transmission of a new AKE\_Init message.

When the HDCP Repeater receives HDCP2\_0\_REPEATER\_DOWNSTREAM or HDCP1\_DEVICE\_DOWNSTREAM bits that are set from a downstream HDCP Repeater, it must propagate this information to the upstream HDCP Transmitter by setting the corresponding bits in the RepeaterAuth\_Send\_ReceiverID\_List message.

If HDCP2\_0\_REPEATER\_DOWNSTREAM or HDCP1\_DEVICE\_DOWNSTREAM bit is set, the Upstream Content Control Function may instruct the most upstream HDCP Transmitter to abort the transmission of certain HDCP encrypted Type 1 Content Streams. The most upstream HDCP Transmitter must be prepared to process the request and immediately cease the transmission of specific Content Streams as instructed by the Upstream Content Control Function.

Whenever the HDCP Transmitter reads the RepeaterAuth\_Send\_ReceiverID\_List message, it verifies the integrity of the Receiver ID list by computing  $V$  and comparing the most significant 128-bits of  $V$  and  $V'$ . If the values do not match, authentication fails, the authentication protocol is aborted and HDCP Encryption is disabled.

On successful verification of Receiver ID list and topology information, i.e. if the values match, none of the reported *Receiver IDs* are in the current revocation list (in the case of the most upstream HDCP Transmitter), the HDCP Transmitter does not detect a roll-over of *seq\_num\_V*, the downstream topology does not exceed specified maximums (explained below), the HDCP Transmitter (including downstream port of HDCP Repeater) writes the least significant 128-bits of *V* to the HDCP Repeater as part of the *RepeaterAuth\_Send\_Ack* message. Every *RepeaterAuth\_Send\_ReceiverID\_List* message from the repeater to the transmitter must be followed by a *RepeaterAuth\_Send\_Ack* message from the transmitter to repeater on successful verification of Receiver ID list and topology information by the transmitter.

The *RepeaterAuth\_Send\_Ack* message must be received by the HDCP Repeater within two seconds from the time the *READY* status was set and the *CP\_IRQ* interrupt was asserted by the HDCP Repeater and the downstream topology does not exceed specified maximums. A match between the least significant 128-bits of *V* and *V'* indicates successful upstream transmission of topology information. If a mismatch occurs or the *RepeaterAuth\_Send\_Ack* message is not received by the repeater within two seconds, the HDCP Repeater must transition in to an unauthenticated state, set the *REAUTH\_REQ* status bit in the *RxStatus* register and assert *CP\_IRQ* interrupt (See Section 2.10.3).

If the upstream HDCP Transmitter receives a *CP\_IRQ* interrupt with *REAUTH\_REQ* status bit set, it may initiate re-authentication with the HDCP Repeater with the transmission of a new *AKE\_Init* message.

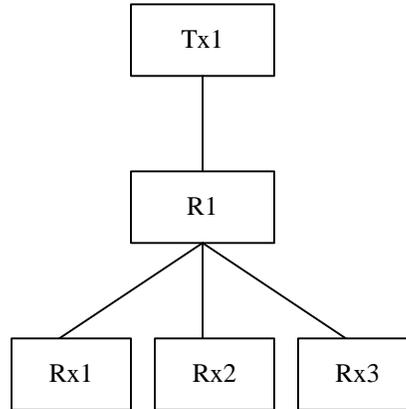
After transmitting the *SKE\_Send\_Eks* message, the HDCP Transmitter, having determined that *REPEATER* received earlier in the protocol is set, sets a three second watchdog timer. If the asserted *READY* status is not received by the HDCP Transmitter within a maximum-permitted time of three seconds after transmitting *SKE\_Send\_Eks* message, authentication of the HDCP Repeater fails. With this failure, the HDCP Transmitter disables HDCP Encryption and aborts the authentication protocol with the HDCP Repeater.

When an HDCP Receiver (including HDCP Repeater) is connected to the HDCP Repeater or when a connected, active HDCP Receiver with which the HDCP Repeater has successfully completed the authentication protocol is disconnected from the HDCP Repeater, the HDCP Repeater assert the *READY* status bit and the *CP\_IRQ* interrupt and must make the *RepeaterAuth\_Send\_ReceiverID\_List* message available for the upstream HDCP Transmitter to read. The *RepeaterAuth\_Send\_ReceiverID\_List* message must include the Receiver IDs of all connected and active downstream HDCP Receivers with which the HDCP Repeater has successfully completed the authentication protocol. This enables upstream propagation of the most recent topology information after changes to the topology without interrupting the transmission of HDCP Content.

Refer to Table 2.2 for the HDCP Repeater upstream and downstream propagation time.

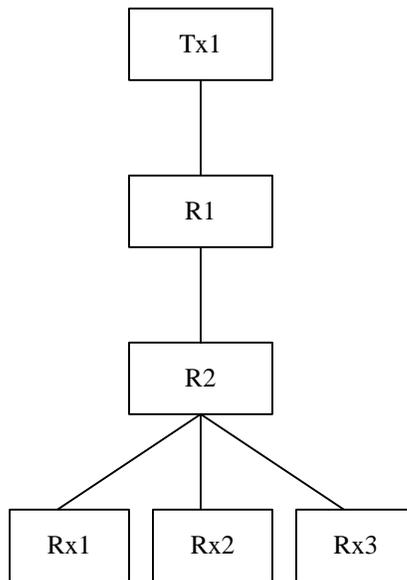
The HDCP Repeater propagates topology information upward through the connection tree to the HDCP Transmitter. An HDCP Repeater reports the topology status variables *DEVICE\_COUNT* and *DEPTH*. The *DEVICE\_COUNT* for an HDCP Repeater is equal to the total number of connected downstream HDCP Receivers and HDCP Repeaters. The value is calculated as the sum of the number of directly connected downstream HDCP Receivers and HDCP Repeaters plus the sum of the *DEVICE\_COUNT* received from all connected HDCP Repeaters. The *DEPTH* status for an HDCP Repeater is equal to the maximum number of connection levels below any of the downstream HDCP-protected Interface Ports. The value is calculated as the maximum *DEPTH* reported from downstream HDCP Repeaters plus one (accounting for the connected downstream HDCP Repeater).

In Figure 2.6, R1 has three downstream HDCP Receivers connected to it. It reports a DEPTH of one and a DEVICE\_COUNT of three.



**Figure 2.6. DEPTH and DEVICE\_COUNT for HDCP Repeater**

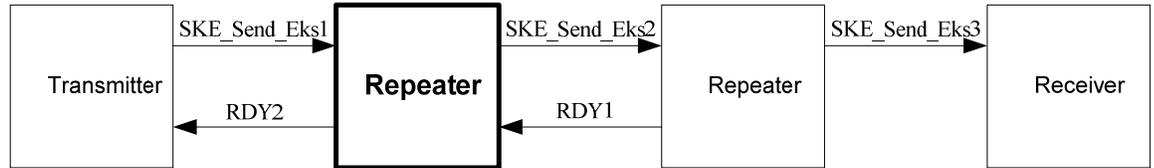
In Figure 2.7, R1 reports a DEPTH of two and a DEVICE\_COUNT of four.



**Figure 2.7. DEPTH and DEVICE\_COUNT for HDCP Repeater**

HDCP Repeaters must be capable of supporting DEVICE\_COUNT values of up to 31 and DEPTH values of up to 4. If the computed DEVICE\_COUNT for an HDCP Repeater exceeds 31, the error is referred to as MAX\_DEVS\_EXCEEDED error. The repeater sets MAX\_DEVS\_EXCEEDED bit to one in the RepeaterAuth\_Send\_ReceiverID\_List message. If the computed DEPTH for an HDCP Repeater exceeds four, the error is referred to as MAX\_CASCADE\_EXCEEDED error. The repeater sets MAX\_CASCADE\_EXCEEDED bit to one in the RepeaterAuth\_Send\_ReceiverID\_List message. When an HDCP Repeater receives a MAX\_DEVS\_EXCEEDED or a MAX\_CASCADE\_EXCEEDED error from a downstream HDCP Repeater, it must propagate the error to the upstream HDCP Transmitter, assert the READY bit and assert the CP\_IRQ interrupt. If a transmitter receives these errors, it must not read the most significant 128-bits of *V'*, Receiver ID list and *seq\_num\_V* from the HDCP Repeater.

Authentication fails if the topology maximums are exceeded. HDCP Encryption is disabled and the authentication protocol is aborted. The top-level HDCP Transmitter, having already performed SRM integrity check during AKE, proceeds to see if the *Receiver ID* of any downstream device from the Receiver ID list is found in the current revocation list, and, if present, authentication fails, HDCP Encryption is disabled and authentication protocol is aborted.



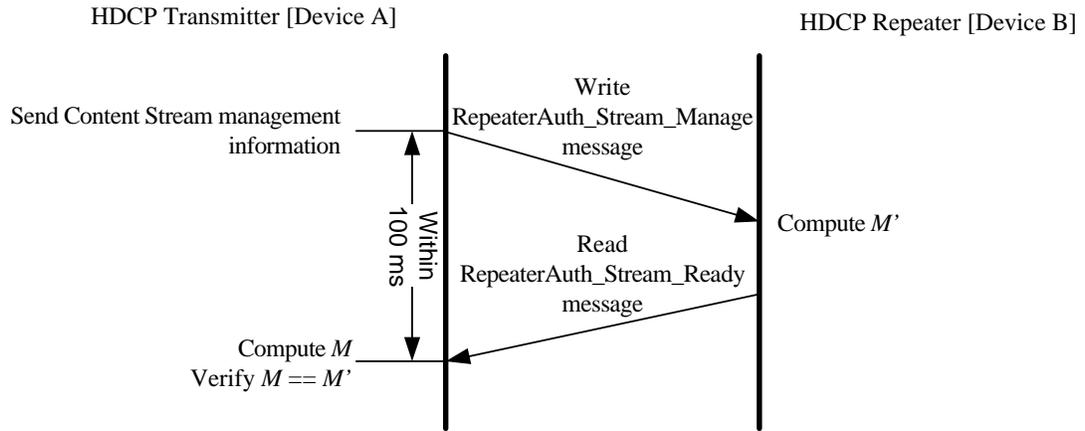
**Figure 2.8. HDCP Repeater Protocol Timing Requirements**

From	To	Max Delay	Conditions and Comments
SKE_Send_Eks1 Session Key received from Upstream HDCP Transmitter	SKE_Send_Eks2 $k_s$ generated by HDCP Repeater transmitted downstream	100 ms	Downstream propagation time.
SKE_Send_Eks3 $k_s$ transmitted to all downstream HDCP-protected Interface Ports	RDY1 Upstream READY asserted	200 ms	Upstream propagation time when no downstream HDCP Repeaters are attached (no downstream Receiver ID lists to process)
RDY1 Downstream Receiver IDs and topology information received	RDY2 Upstream READY asserted	200 ms	Upstream propagation time when one or more HDCP Repeaters are attached. From latest downstream READY (downstream Receiver ID lists must be processed)
SKE_Send_Eks1 Upstream HDCP Transmitter transmits $k_s$	RDY2 Upstream transmitter receives CP_IRQ with READY asserted	1.2 seconds	For the Maximum of four repeater levels, 4 * (100 ms + 200 ms)

**Table 2.2. HDCP Repeater Protocol Timing Requirements**

Table 2.2 specifies HDCP Repeater timing requirements that bound the worst-case propagation time for the Receiver ID list. A maximum delay of three seconds, to receive the RepeaterAuth\_Send\_ReceiverID\_List message by the upstream transmitter, has been provided after transmission of the SKE\_Send\_Eks message to account for authentication delays due to the presence of downstream receivers that have not been paired with the upstream HDCP Repeater. Note that because each HDCP Repeater does not know the number of downstream HDCP Repeaters, it must use the same three-second timeout used by the upstream HDCP Transmitter for receiving the RepeaterAuth\_Send\_ReceiverID\_List message.

## 2.5.2 Downstream Propagation of Content Stream Management Information



**Figure 2.9. Downstream Propagation of Content Stream Management Information**

The HDCP Transmitter may transmit multiple Content Streams to an HDCP Receiver during an HDCP Session. The HDCP Transmitter may use the same Session Key,  $k_s$ , negotiated during the HDCP Session for HDCP Encryption of the Content Streams.

The HDCP Transmitter propagates Content Stream management information, which includes Type values assigned to Content Streams, using the RepeaterAuth\_Stream\_Manage message to the attached HDCP Repeater. The HDCP Transmitter executes this step after successful completion of Session Key Exchange and before beginning the transmission of a Content Stream after HDCP Encryption to the HDCP Repeater. The RepeaterAuth\_Stream\_Manage message from an HDCP Transmitter to the attached HDCP Repeater identifies restrictions, as specified by the Upstream Content Control Function, on the transmission of Content Streams to specific devices.

Type values are assigned to all Content Streams by the most upstream HDCP Transmitter based on instructions received from the Upstream Content Control Function. The exact mechanism used by the Upstream Content Control Function to instruct the HDCP Transmitter is outside the scope of this specification. Type 0 Content Streams (see Section 4.2.12) may be transmitted by the HDCP Repeater to all HDCP Devices. Type 1 Content Streams (see Section 4.2.12) must not be transmitted by the HDCP Repeater through its HDCP-protected Interface Ports connected to HDCP 1.x-compliant Devices and HDCP 2.0-compliant Repeaters.

The HDCP Transmitter must write the RepeaterAuth\_Stream\_Manage message specifying Type values assigned to Content Streams, to the attached HDCP Repeater at least 100ms before the transmission of the corresponding Content Streams after HDCP Encryption. The HDCP Transmitter must only send the RepeaterAuth\_Stream\_Manage message corresponding to encrypted Content Streams it will transmit to the HDCP Repeater. The HDCP Transmitter initializes  $seq\_num\_M$  to 0 at the beginning of the HDCP Session i.e. after AKE\_Init is sent. It is incremented by one after the transmission of every RepeaterAuth\_Stream\_Manage message.

On receiving the RepeaterAuth\_Stream\_Manage message, the HDCP Repeater computes  $M'$  as given below. HMAC-SHA256 is computed over the concatenation of  $StreamID\_Type$  (see Section 4.2.12) and  $seq\_num\_M$  values received as part of the RepeaterAuth\_Stream\_Manage message. All values are in big-endian order. The key used for HMAC is  $SHA256(k_d)$ .  $seq\_num\_M$  must never be reused during an HDCP Session for the computation of  $M'$  (or  $M$ ). If  $seq\_num\_M$  rolls over, the HDCP Transmitter must disable HDCP Encryption if encryption is enabled, restart authentication by the transmission of a new AKE\_Init message.

$M'$  (or  $M$ ) = HMAC-SHA256(*StreamID\_Type* || *seq\_num\_M*, SHA256( $k_d$ )).

$M'$  must be made available by the HDCP Repeater for the HDCP Transmitter to read as part of the RepeaterAuth\_Stream\_Ready message.

The RepeaterAuth\_Stream\_Ready message must be available for the transmitter to read within 100 ms from the time the transmitter finishes writing the RepeaterAuth\_Stream\_Manage message parameters to the HDCP Receiver. The HDCP Transmitter must not attempt to read RepeaterAuth\_Stream\_Ready sooner than 100 ms after writing the RepeaterAuth\_Stream\_Manage message. Every RepeaterAuth\_Stream\_Manage message from the transmitter to the repeater must be followed by a RepeaterAuth\_Stream\_Ready message from the repeater to the transmitter.

When the RepeaterAuth\_Stream\_Ready message is read, the HDCP Transmitter verifies the integrity of the message by computing  $M$  and comparing this value to  $M'$ . If  $M$  is equal to  $M'$ , the HDCP Transmitter may transmit the Content Streams identified in the corresponding RepeaterAuth\_Stream\_Manage message. If the RepeaterAuth\_Stream\_Ready message is not available for the transmitter to read within 100 ms or if  $M$  is not equal to  $M'$ , the HDCP Transmitter must not transmit the Content Streams identified in the corresponding RepeaterAuth\_Stream\_Manage message.

An HDCP Repeater connected to an HDCP 2.0-compliant Transmitter or an HDCP 1.x-compliant Transmitter will not receive the RepeaterAuth\_Stream\_Manage message from the transmitter. In this case, the HDCP Repeater must assign a Type value of 0x00 to all Content Streams received from the HDCP Transmitter.

The HDCP Repeater must in turn propagate the received Content Stream management information using the RepeaterAuth\_Stream\_Manage message further downstream.

## 2.6 Link Integrity Check

After successful completion of SKE, HDCP Encryption is enabled and encrypted content starts to flow between the HDCP Transmitter and the HDCP Receiver. Once encryption is enabled, a periodic Link Integrity Check is performed to maintain cipher synchronization between the HDCP Transmitter and the HDCP Receiver.

### 2.6.1 Link Integrity Check in MST Mode

To perform link integrity check in the MST mode, two MTPH timeslots immediately following SR are used to transmit a known 16-bit pattern, 0x531F, from the transmitter to the receiver. This pattern is referred to as LINK\_VERIFICATION\_PATTERN and is transmitted least significant byte first. The LINK\_VERIFICATION\_PATTERN is duplicated per lane on 2-lane and 4-lane Main Links. The transmitter sets the two MTPH timeslots following a given SR symbol to the corresponding byte of the pattern, encrypts the MTPHs and sends the MTPHs to the receiver. The receiver decrypts the MTPHs and compares the decrypted byte values to the corresponding byte in the LINK\_VERIFICATION\_PATTERN. If the received pattern, which is transmitted least significant byte first, matches the LINK\_VERIFICATION\_PATTERN at the receiver, it indicates that the ciphers are in sync. An error is determined to have occurred if the decrypted byte does not match the corresponding byte in the LINK\_VERIFICATION\_PATTERN. No error correction techniques (e.g., majority voting) should be applied to the MTPH timeslots used to transmit the LINK\_VERIFICATION\_PATTERN. HDCP Encryption is only applied to the MTPH timeslots used to transmit LINK\_VERIFICATION\_PATTERN; other MTPH timeslots must not be encrypted.

A link integrity failure is determined to have occurred if pattern mismatches at the receiver are detected for two successive link frame periods. Two successive link frame periods are checked to enable recovery from simple transient synchronization errors (e.g., random bit error bursts). If a

failure is detected within two successive link frames then the receiver has experienced a non-recoverable loss of cipher synchronization.

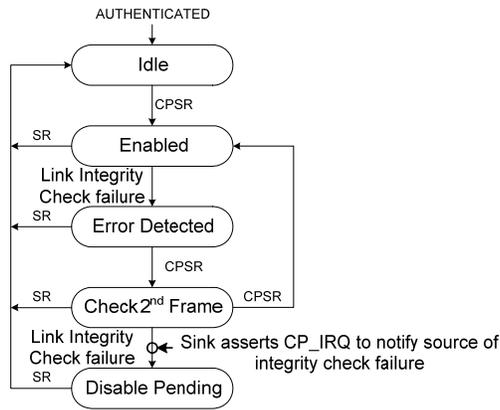
On detecting an unrecoverable loss of cipher synchronization, the HDCP Receiver must assert the `LINK_INTEGRITY_FAILURE` bit in the `RxStatus` register and generate a `CP_IRQ` interrupt. On receiving a `CP_IRQ` interrupt, the HDCP Transmitter is required to read the `RxStatus` register to determine the cause of the interrupt. The HDCP Transmitter must disable HDCP Encryption at the SR boundary as soon as feasible after receiving the `CP_IRQ` interrupt from the HDCP Receiver if the `LINK_INTEGRITY_FAILURE` bit is set and must initiate re-authentication with the transmission of a new `AKE_Init` message.

## 2.6.2 Link Integrity Check in SST Mode

To perform link integrity check in SST mode, Bit 5 of VB-ID is used to transmit a known 16-bit pattern, 0x531F, from the transmitter to the receiver one bit at a time. This pattern is referred to as `LINK_VERIFICATION_PATTERN`. The VB-ID is transmitted on all lanes after every BS/SR/CPBS/CPSR symbol, as described in the DisplayPort Specification. It is transmitted once per lane for 4-lane Main Link, twice per lane for 2-lane Main Link, and four times for 1-lane Main Link, resulting in a total of four VB-ID's following each CPBS/CPSR. The `LINK_VERIFICATION_PATTERN` is continuously and repeatedly transmitted least significant bit first. All four VB-ID's following each CPBS/CPSR shall carry the same Bit 5 value. After every CPSR symbol the pattern transmission is restarted, inserting the LSB of the pattern in all VB-IDs associated with the CPSR symbol. The transmitter sets Bit 5 of the VB-ID symbol associated with a given CPBS/CPSR symbol to the corresponding pattern bit value, encrypts the VB-ID and sends the VB-ID to the receiver. The receiver decrypts the VB-ID and compares Bit 5 of VB-ID to the corresponding bit value in the `LINK_VERIFICATION_PATTERN`. If the received pattern, which is transmitted one bit at a time, matches the `LINK_VERIFICATION_PATTERN` at the receiver, it indicates that the ciphers are in sync. An error is determined to have occurred if the bit pattern in any of the VB-ID symbols is found to not match the expected bit of the `LINK_VERIFICATION_PATTERN`. No error correction techniques (e.g., majority voting) should be applied to Bit 5 of the VB-ID symbols associated with a given CPBS/CPSR symbol.

A link integrity failure is determined to have occurred if three consecutive pattern mismatches at the receiver (in  $16 * 3 = 48$  VB-ID transmissions) are detected within two successive frame periods. Two successive frame periods are checked to enable recovery from simple transient synchronization errors (e.g., random bit error bursts). If a failure is detected within two successive frames then the receiver has experienced a non-recoverable loss of cipher synchronization. The state machine shown in Figure 2-10 illustrates the expected HDCP Receiver link integrity check behavior.

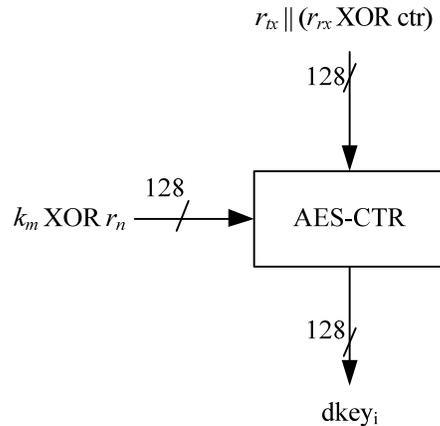
On detecting an unrecoverable loss of cipher synchronization (e.g., transition from "Check 2nd Frame" to "Disable Pending" in Figure 2-10), the HDCP Receiver must assert the `LINK_INTEGRITY_FAILURE` bit in the `RxStatus` register and generate a `CP_IRQ` interrupt. On receiving a `CP_IRQ` interrupt, the HDCP Transmitter is required to read the `RxStatus` register to determine the cause of the interrupt. The HDCP Transmitter must disable HDCP Encryption at the CPSR/SR transmission boundary as soon as feasible after receiving the `CP_IRQ` interrupt from the HDCP Receiver if the `LINK_INTEGRITY_FAILURE` bit is set and must initiate re-authentication with the transmission of a new `AKE_Init` message.



**Figure 2-10. HDCP Receiver Link Integrity Check State Machine**

## 2.7 Key Derivation

Key derivation is illustrated in Figure 2.11.



**Figure 2.11. Key Derivation**

$r_{tx}$  is concatenated with  $r_{rx} \text{ XOR } \text{ctr}$  ( $r_{tx} \parallel (r_{rx} \text{ XOR } \text{ctr})$ ). All values are in big-endian order. ctr is a 64-bit counter and is initialized to 0 at the beginning of the HDCP Session i.e. after AKE\_Init is sent or received. It is incremented by one after every derived key computation.  $dkey_i$  is the 128-bit derived key when  $\text{ctr} = i$ . ctr must never be reused during an HDCP Session.

$r_n$  is initialized to 0 during AKE i.e. during the generation of  $dkey_0$  and  $dkey_1$ . It is set to a pseudo-random value during locality check as explained in Section 2.3. The pseudo-random  $r_n$  is XORed with the least-significant 64-bits of  $k_m$  during generation of  $dkey_2$ .

## 2.8 HDCP Transmitter State Diagram

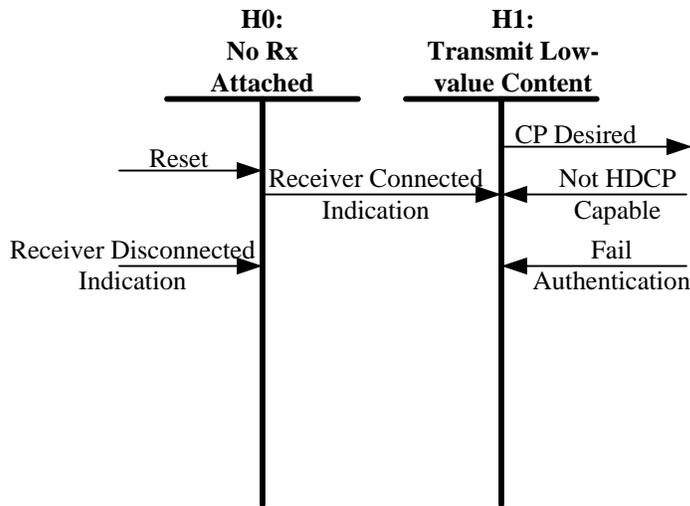
As explained in Section 1.3, the HDCP Transmitter may support simultaneous connections to HDCP Receivers through one or more of its HDCP-protected interface ports. The HDCP Transmitter state diagram is implemented independently on each HDCP-protected interface port.

The HDCP Transmitter Link State Diagram and HDCP Transmitter Authentication Protocol State Diagram (Figure 2.12 and Figure 2.13) illustrate the operation states of the authentication protocol

for an HDCP Transmitter that is not an HDCP Repeater. For HDCP Repeaters, the downstream (HDCP Transmitter) side is covered in Section 2.10.2.

Transmitter's decision to begin authentication is dependent on events such as detection of an HDCP Receiver, availability of premium content or other implementation dependent details in the transmitter. In the event of authentication failure, an HDCP Receiver must be prepared to process subsequent authentication attempts. The HDCP Transmitter may cease to attempt authentication for transmitter-specific reasons, which include receiving a Receiver Disconnected Indication or after a certain number of authentication re-attempts by the transmitter.

The transmitter must not initiate authentication unless it determines that the receiver is HDCP-capable.



Note: Transition arrows with no connected state (e.g. Reset) indicate transitions that can occur from multiple states

**Figure 2.12. HDCP Transmitter Link State Diagram**

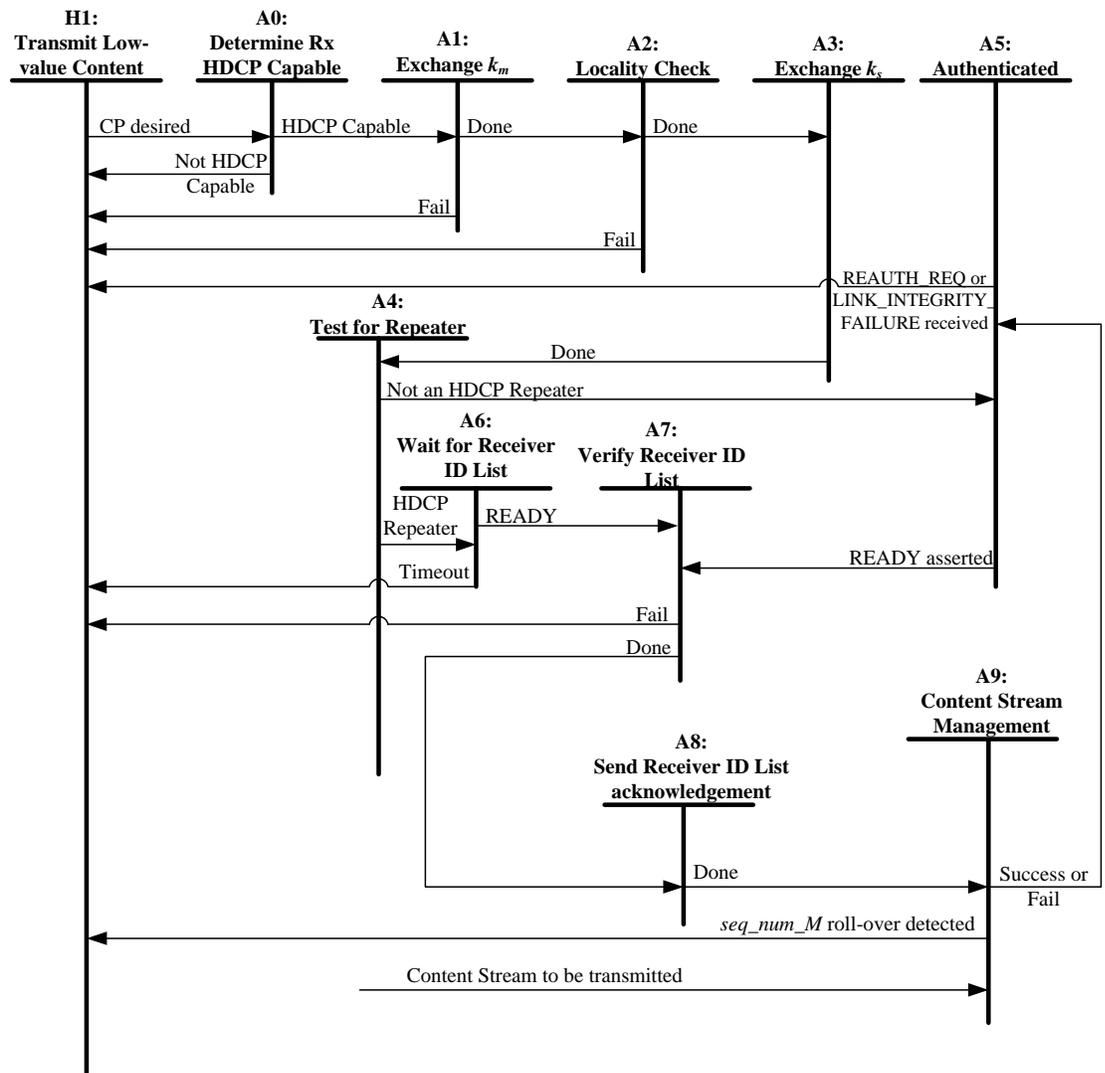


Figure 2.13. HDCP Transmitter Authentication Protocol State Diagram

**Transition Any State:H0.** Reset conditions at the HDCP Transmitter or disconnect of the connected HDCP capable receiver cause the HDCP Transmitter to enter the No Receiver Attached state.

**Transition H0:H1.** The detection of a sink device (through Receiver Connected Indication) indicates to the transmitter that a sink device is connected and ready to display the received content. When the receiver is no longer active, the transmitter is notified through Receiver Disconnected Indication.

**State H1: Transmit Low-value Content.** In this state the transmitter should begin sending an unencrypted signal with HDCP Encryption disabled. The transmitted signal can be a low value content or informative on-screen display. This will ensure that a valid video signal is displayed to the user before and during authentication.

**Transition H1:A0.** If content protection is desired by the Upstream Content Control Function, then the HDCP Transmitter should immediately attempt to determine whether the receiver is HDCP capable.

**State A0: Determine Rx HDCP Capable.** The transmitter determines that the receiver is HDCP capable by reading the HDCP\_CAPABLE bit in the receiver's *RxCaps* register. If this bit is set to 1, it indicates that the receiver is HDCP capable. If state A0 is reached when content protection is desired by the Upstream Content Control Function, authentication must be started immediately by the transmitter if the receiver is HDCP capable. A valid video screen is displayed to the user with encryption disabled during this time.

**Transition A0:H1.** If the receiver is not HDCP capable, the transmitter continues to transmit low value content or informative on-screen display.

**Transition A0:A1.** If the receiver is HDCP capable, the transmitter initiates the authentication protocol.

**State A1: Exchange  $k_m$ .** In this state, the HDCP Transmitter initiates authentication by writing AKE\_Init message to the HDCP Receiver. It reads AKE\_Send\_Cert from the receiver within 100 ms after writing the AKE\_Init message.

If the HDCP Transmitter does not have  $k_m$  stored corresponding to the *Receiver ID*, it generates  $E_{k_{pub}}(k_m)$  and sends  $E_{k_{pub}}(k_m)$  as part of the AKE\_No\_Stored\_km message to the receiver after verification of signature on  $cert_{rx}$ . It performs integrity check on the SRM and checks to see whether the *Receiver ID* of the connected HDCP Device is in the revocation list. It computes H, reads AKE\_Send\_H\_prime message from the receiver containing  $H'$  within one second after writing AKE\_No\_Stored\_km to the receiver and compares  $H'$  against H.

If the HDCP Transmitter has  $k_m$  stored corresponding to the *Receiver ID*, it writes AKE\_Stored\_km message containing  $E_{kh}(k_m)$  and  $m$  to the receiver, performs integrity check on the SRM and checks to see whether the *Receiver ID* of the connected HDCP Device is in the revocation list. It computes H, reads AKE\_Send\_H\_prime message from the receiver containing  $H'$  within 200 ms after writing AKE\_Stored\_km to the receiver and compares  $H'$  against H.

If the HDCP Transmitter does not have a  $k_m$  stored corresponding to the *Receiver ID*, it implements pairing with the HDCP Receiver as explained in Section 2.2.1.

**Transition A1:H1.** This transition occurs on failure of signature verification on  $cert_{rx}$ , failure of SRM integrity check, if *Receiver ID* of the connected HDCP Device is in the revocation list or if there is a mismatch between H and  $H'$ . This transition also occurs if AKE\_Send\_H\_prime message is not received within one second after writing AKE\_No\_Stored\_km or within 200 ms after writing AKE\_Stored\_km to the receiver.

**Transition A1:A2.** The HDCP Transmitter implements locality check after successful completion of AKE and pairing.

**State A2: Locality Check.** In this state, the HDCP Transmitter implements the locality check as explained in Section 2.3 with the HDCP Receiver.

**Transition A2:H1.** This transition occurs on one or more consecutive locality check failures. Locality check fails when  $L'$  is not available for the transmitter to read within 7 ms and the watchdog timer at the HDCP Transmitter expires or on a mismatch between L and  $L'$ .

**Transition A2:A3.** The HDCP Transmitter implements SKE after successful completion of locality check.

**State A3: Exchange  $k_s$ .** The HDCP Transmitter sends encrypted Session Key,  $E_{dk_s}(k_s)$ , and  $r_{iv}$  to the HDCP Receiver as part of the SKE\_Send\_Eks message. It may enable HDCP Encryption 200 ms after sending encrypted Session Key. HDCP Encryption must be enabled only after successful completion of AKE, locality check and SKE stages.

**Transition A3:A4.** This transition occurs after completion of SKE.

**State A4: Test for Repeater.** The HDCP Transmitter evaluates the REPEATER value that was received in State A1.

**Transition A4:A5.** REPEATER bit is not set (the HDCP Receiver is not an HDCP Repeater).

**State A5: Authenticated.** At this time, and at no prior time, the HDCP Transmitter has completed the authentication protocol.

A periodic Link Integrity Check is performed to maintain cipher synchronization between the HDCP Transmitter and the HDCP Receiver.

**Transition A4:A6.** REPEATER bit is set (the HDCP Receiver is an HDCP Repeater).

**State A6: Wait for Receiver ID List.** The HDCP Transmitter sets up a three-second watchdog timer after sending SKE\_Send\_Eks.

**Transition A6:H1.** The watchdog timer expires before READY has been asserted by the repeater.

**Transition A6:A7.** READY bit is asserted.

**State A7: Verify Receiver ID List.** If a transition in to this state occurs from State A6, the watchdog timer is cleared. The transmitter reads the RepeaterAuth\_Send\_ReceiverID\_List message. If both MAX\_DEVS\_EXCEEDED and MAX\_CASCADE\_EXCEEDED bits are not set, the transmitter computes  $V$  and compares the most significant 128-bits of  $V$  and  $V'$ . The *Receiver IDs* from the Receiver ID list are compared against the current revocation list.

**Transition A7:H1.** This transition is made if a mismatch occurs between the most significant 128-bits of  $V$  and  $V'$ . This transition is also made if any of the *Receiver IDs* in the Receiver ID list are found in the current revocation list or if the HDCP Transmitter detects a roll-over of  $seq\_num\_V$ . A MAX\_CASCADE\_EXCEEDED or MAX\_DEVS\_EXCEEDED error also causes this transition.

**Transition A7:A8.** This transition occurs on successful verification of the most significant 128-bits of  $V$  and  $V'$ , none of the reported *Receiver IDs* are in the current revocation list, the HDCP Transmitter does not detect a roll-over of  $seq\_num\_V$  and the downstream topology does not exceed specified maximums.

**State A8: Send Receiver ID list acknowledgement.** , The HDCP Transmitter sends the least significant 128-bits of  $V$  to the HDCP Repeater as part of the RepeaterAuth\_Send\_Ack message.

The RepeaterAuth\_Send\_Ack message must be received by the HDCP Repeater within two seconds from the time the READY status was set and the CP\_IRQ interrupt was asserted by the HDCP Repeater..

**Transition A8:A9.** This transition occurs after the RepeaterAuth\_Send\_Ack message has been written to the repeater.

**Transition A5:H1.** This transition occurs if a CP\_IRQ interrupt was received with the REAUTH\_REQ or LINK\_INTEGRITY\_FAILURE status bit set.

**Transition A5:A7.** This transition occurs whenever a CP\_IRQ interrupt is received from the connected HDCP Repeater with READY bit set.

**State A9: Content Stream Management.** This stage is implemented if Content Stream is to be transmitted. The HDCP Transmitter sends the RepeaterAuth\_Stream\_Manage message specifying Type values assigned to Content Streams, to the attached HDCP Repeater at least 100ms before the transmission of the corresponding Content Streams after HDCP Encryption. It must receive the RepeaterAuth\_Stream\_Ready message from the HDCP Repeater within 100 ms after the transmission of RepeaterAuth\_Stream\_Manage message and verifies  $M'$ . This step fails if the RepeaterAuth\_Stream\_Ready message is not available to read within 100 ms or if  $M$  is not equal to  $M'$ .

This stage may be implemented in parallel with the upstream propagation of topology information (State A4, State A6, State A7 and State A8) and with the flow of encrypted content and Link Integrity Check (State A5). This state may be implemented asynchronously from the rest of the state diagram. A transition in to this state may occur from State A4, State A5, State A6, State A7 or State A8 if Content Stream is to be transmitted. Also, the transition from State A9 must return to the appropriate state to allow for uninterrupted operation.

**Transition A9:A5.** This transition occurs on success or failure of the Content Stream management stage.

**Transition A9:H1.** This transition occurs if  $seq\_num\_M$  rolls over.

Note: Since Link Integrity Check (State A5) may be implemented in parallel with the upstream propagation of topology information (State A4, State A6, State A7 and State A8) and Content Stream management (State A9) stages, the link integrity check process (i.e. State A5) may be implemented asynchronously from the rest of the state diagram. The transition into State A5 may occur from any state for which encryption is currently enabled. Also, the transition from State A5 returns to the appropriate state to allow for uninterrupted operation.

The HDCP Transmitter may support simultaneous connections to HDCP Receivers through one or more of its HDCP-protected interface ports. It may share the same Session Key and  $r_{iv}$  across all its HDCP-protected interface ports, as explained in Section 3.5. However, the HDCP Transmitter must ensure that each connected HDCP Receiver receives distinct  $k_m$  and  $r_{ix}$  values.

## 2.9 HDCP Receiver State Diagram

The operation states of the authentication protocol for an HDCP Receiver that is not an HDCP Repeater are illustrated in Figure 2.14 **Error! Reference source not found.** For HDCP Repeaters, the upstream (HDCP Receiver) side is covered in Section 2.10.3.

The HDCP Receiver must be ready to re-authenticate with the HDCP Transmitter at any point in time. In particular, the only indication to the HDCP Receiver of a re-authentication attempt by the HDCP Transmitter is the reception of the AKE\_Init message from the HDCP Transmitter.

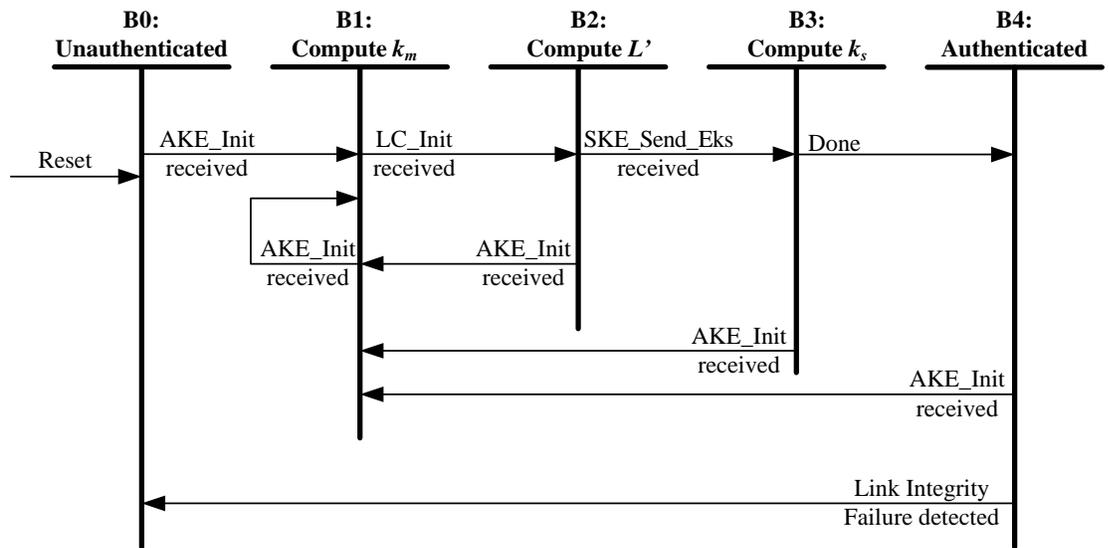


Figure 2.14. HDPC Receiver Authentication Protocol State Diagram

**Transition Any State:B0.** Reset conditions at the HDPC Receiver cause the HDPC Receiver to enter the unauthenticated state.

**State B0: Unauthenticated.** The HDPC Receiver is awaiting the reception of AKE\_Init from the HDPC Transmitter to trigger the authentication protocol.

**Transition B0:B1.** AKE\_Init message is received from the HDPC Transmitter.

**State B1: Compute  $k_m$ .** In this state, the HDPC Receiver makes the AKE\_Send\_Cert message available for reading by the transmitter in response to AKE\_Init. If AKE\_No\_Stored\_km is received, the receiver decrypts  $k_m$  with  $k_{priv_{rx}}$ , calculates  $H'$ . It makes AKE\_Send\_H\_prime message available for reading immediately after computation of  $H'$  to ensure that the message is received by the transmitter within the specified one second timeout at the transmitter.

If AKE\_Stored\_km is received, the HDPC Receiver decrypts  $E_{k_H}(k_m)$  to derive  $k_m$  and calculates  $H'$ . It makes AKE\_Send\_H\_prime message available for reading immediately after computation of  $H'$  to ensure that the message is received by the transmitter within the specified 200 ms timeout at the transmitter

If AKE\_No\_Stored\_km is received, this is an indication to the HDPC Receiver that the HDPC Transmitter does not contain a  $k_m$  stored corresponding to its *Receiver ID*. It implements pairing with the HDPC Transmitter as explained in Section 2.2.1.

**Transition B1: B1.** Should the HDPC Transmitter write an AKE\_Init while the HDPC Receiver is in State B1, the HDPC Receiver abandons intermediate results and restarts computation of  $k_m$ .

**Transition B1: B2.** The transition occurs when  $r_n$  is received as part of LC\_Init message from the transmitter.

**State B2: Compute  $L'$ .** The HDPC Receiver computes  $L'$  required during locality check and makes the LC\_Send\_L\_prime message available for reading by the transmitter.

**Transition B2: B1.** Should the HDCP Transmitter write an AKE\_Init while the HDCP Receiver is in State B2, the HDCP Receiver abandons intermediate results and restarts computation of  $k_m$ .

**Transition B2: B3.** The transition occurs when SKE\_Send\_Eks message is received from the transmitter.

**State B3: Compute  $k_s$ .** The HDCP Receiver decrypts  $E_{dkey}(k_s)$  to derive  $k_s$ .

**Transition B3: B1.** Should the HDCP Transmitter write an AKE\_Init while the HDCP Receiver is in State B3, the HDCP Receiver abandons intermediate results and restarts computation of  $k_m$ .

**Transition B3: B4.** Successful computation of  $k_s$  transitions the receiver into the authenticated state.

**State B4: Authenticated.** The HDCP Receiver has completed the authentication protocol. Periodically, it performs a link integrity check.

**Transition B4: B1.** Should the HDCP Transmitter write an AKE\_Init while the HDCP Receiver is in State B4, the HDCP Receiver abandons intermediate results and restarts computation of  $k_m$ .

**Transition B4: B0.** This transition occurs when a link integrity failure is detected by the HDCP Receiver as explained in Section 2.6. The HDCP Receiver asserts the LINK\_INTEGRITY\_FAILURE bit in the *RxStatus* register and generates a CP\_IRQ interrupt.

## 2.10 HDCP Repeater State Diagrams

The HDCP Repeater has one HDCP-protected Interface connection to an upstream HDCP Transmitter and one or more HDCP-protected Interface connections to downstream HDCP Receivers. The state diagram for each downstream connection (Figure 2.15 and Figure 2.16) is substantially the same as that for the host HDCP Transmitter (Section 2.8), with the exception that the HDCP Repeater is not required to check for downstream Receiver IDs in a revocation list.

When the upstream HDCP-protected interface port of the HDCP Repeater is in an unauthenticated state, it signals the detection of an active downstream HDCP Receiver to the upstream HDCP Transmitter in one of the following ways. The HDCP Repeater must generate the CONNECTION\_STATUS\_NOTIFY message to indicate plug of an active HDCP Receiver when the most upstream HDCP Transmitter is capable of operating in the MST mode (MST-capable). It must pulse IRQ\_HPD when the most upstream HDCP Transmitter is capable of operating only in the SST mode (SST-capable only) and has enabled IRQ\_HPD for upstream notification. Hot plug, CONNECTION\_STATUS\_NOTIFY and IRQ\_HPD are collectively referred to as Receiver Connected Indication in this specification.

Hot Unplug is referred to as Receiver Disconnected Indication in this specification.

Whenever authentication is initiated by the upstream HDCP Transmitter by sending AKE\_Init, the HDCP Repeater immediately initiates authentication on all its downstream HDCP-protected interface ports if its downstream ports are in an unauthenticated state.

The HDCP Repeater may cache the latest Receiver ID list and topology information received on its downstream ports. Whenever authentication is attempted by the upstream transmitter by writing the AKE\_Init message, the HDCP Repeater may propagate the cached Receiver ID list upstream without initiating a re-authentication on all its downstream ports.

The HDCP Repeater must generate unique  $k_m$  values for HDCP Devices connected to each of its downstream HDCP-protected Interface Ports.

The HDCP Repeater may transmit the same session key,  $k_s$ , to all its authenticated and active downstream HDCP-protected Interface Ports before beginning the transmission of HDCP Content to any of its downstream ports. After beginning the transmission of HDCP Content by the HDCP Repeater to any of its downstream ports, subsequent connection of a new HDCP Receiver to its downstream port must result in (a) a unique session key,  $k_s$ , exchanged with that HDCP Receiver or (b) a new authentication attempt with all its downstream HDCP-protected Interface ports and subsequent exchange of the same session key,  $k_s$ , to all its authenticated and active downstream HDCP-protected Interface Ports.

If an HDCP Repeater has no active downstream HDCP Devices, it must authenticate as an HDCP Receiver with REPEATER bit set to zero if it wishes to receive HDCP Content, but must not pass HDCP Content to downstream devices.

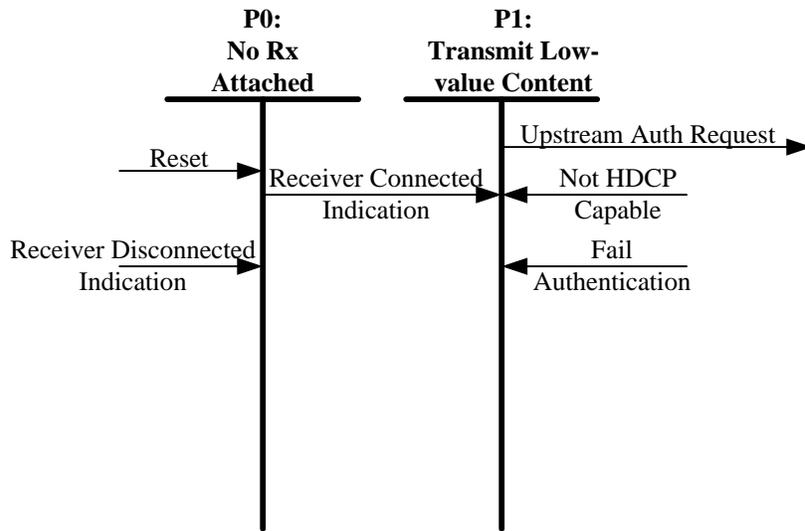
When the upstream HDCP-protected Interface Port of the HDCP Repeater transitions in to an unauthenticated state from an authenticated state (See Transition C5:C0 and Transition C6:C0 in Section 2.10.3), the HDCP Repeater must set the REAUTH\_REQ status bit in the *RxStatus* register and assert CP\_IRQ interrupt. If the upstream HDCP Transmitter receives a CP\_IRQ interrupt with REAUTH\_REQ status bit set, it may initiate re-authentication with the HDCP Repeater with the transmission of a new AKE\_Init message.

### 2.10.1 Propagation of Topology Errors

**MAX\_DEVS\_EXCEEDED and MAX\_CASCADE\_EXCEEDED:** HDCP Repeaters must be capable of supporting DEVICE\_COUNT values of up to 31 and DEPTH values of up to 4. If the computed DEVICE\_COUNT for an HDCP Repeater exceeds 31, the error is referred to as MAX\_DEVS\_EXCEEDED error. The repeater sets MAX\_DEVS\_EXCEEDED bit to one in the RepeaterAuth\_Send\_ReceiverID\_List message. If the computed DEPTH for an HDCP Repeater exceeds four, the error is referred to as MAX\_CASCADE\_EXCEEDED error. The repeater sets MAX\_CASCADE\_EXCEEDED bit to one in the RepeaterAuth\_Send\_ReceiverID\_List message. When an HDCP Repeater receives a MAX\_DEVS\_EXCEEDED or a MAX\_CASCADE\_EXCEEDED error from a downstream HDCP Repeater, it must propagate the error to the upstream HDCP Transmitter and must not transmit  $V'$  and Receiver ID list.

### 2.10.2 HDCP Repeater Downstream State Diagram

In this state diagram and its following description, the downstream (HDCP Transmitter) side refers to the HDCP Transmitter functionality within the HDCP Repeater for its corresponding downstream HDCP-protected Interface Port.



Note: Transition arrows with no connected state (e.g. Reset) indicate transitions that can occur from multiple states

**Figure 2.15. HDCP Repeater Downstream Link State Diagram**

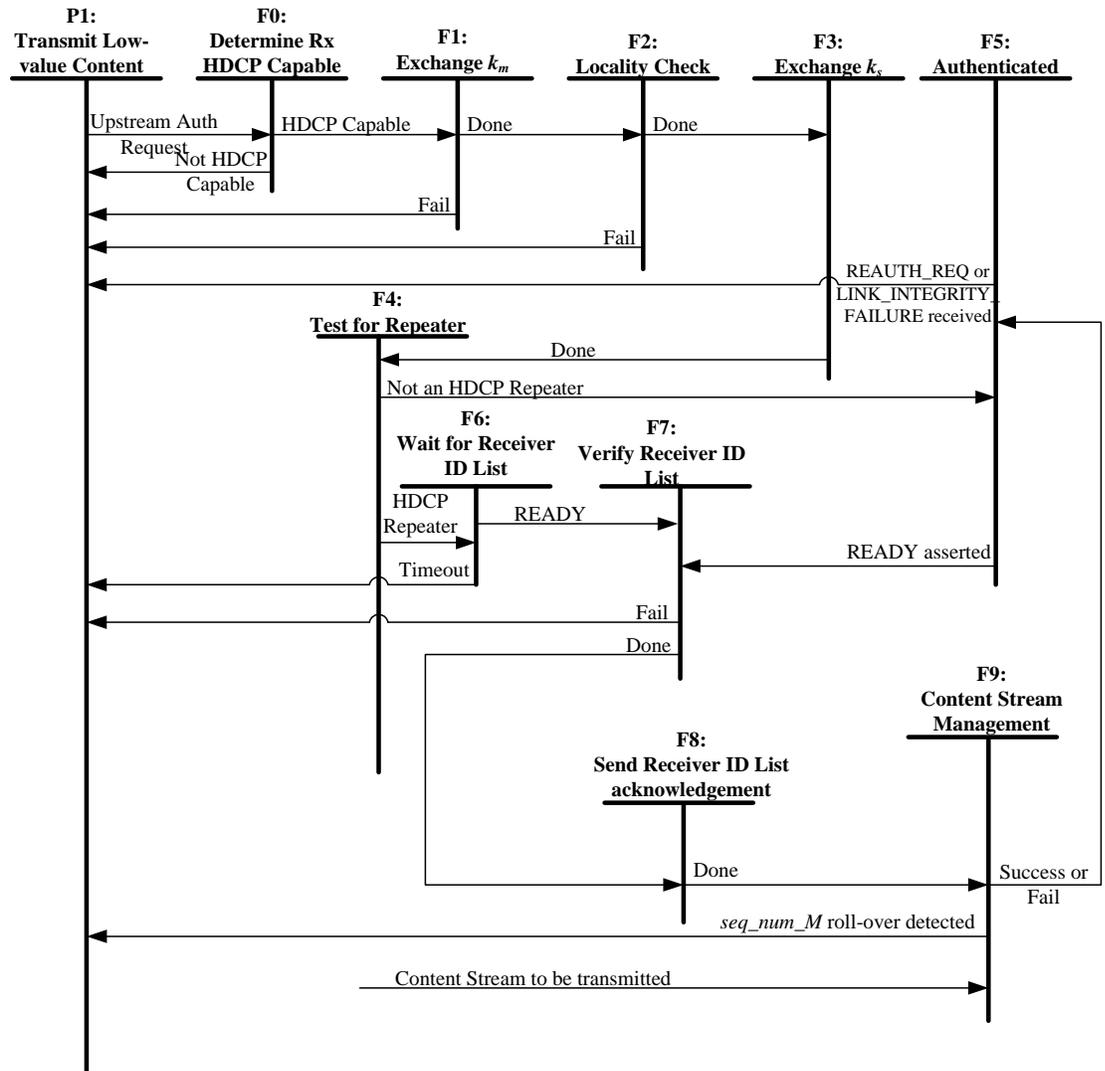


Figure 2.16. HDCP Repeater Downstream Authentication Protocol State Diagram

**Transition Any State:P0.** Reset conditions at the HDCP Repeater or disconnect of the connected HDCP capable receiver cause the HDCP Repeater to enter the No Receiver Attached state for this port.

**Transition P0:P1.** The detection of a sink device (through Receiver Connected Indication) indicates that the receiver is available and active (ready to display received content). When the receiver is no longer active, the downstream (HDCP Transmitter) side is notified through Receiver Disconnected Indication.

**State P1: Transmit low-value content.** In this state the downstream side should begin sending the unencrypted video signal received from the upstream HDCP Transmitter with HDCP Encryption disabled.

**Transition P1:F0.** Upon an Upstream Authentication Request, the downstream side should immediately attempt to determine whether the receiver is HDCP capable.

**State F0: Determine Rx HDCP Capable.** The downstream side determines that the receiver is HDCP capable by reading the HDCP\_CAPABLE bit in the receiver's *RxCaps* register. If this bit is set to 1, it indicates that the receiver is HDCP capable. If state F0 is reached upon an Upstream Authentication Request, authentication must be started immediately by the downstream side if the receiver is HDCP capable. A valid video screen is displayed to the user with encryption disabled during this time.

Note: The downstream side may initiate authentication with the attached HDCP Receiver before an Upstream Authentication Request is received.

**Transition F0:P1.** If the receiver is not HDCP capable, the downstream side continues to transmit low value content or informative on-screen display received from the upstream HDCP Transmitter.

**Transition F0:F1.** If the receiver is HDCP capable, the downstream side initiates the authentication protocol.

**State F1: Exchange  $k_m$ .** In this state, the downstream side initiates authentication by writing AKE\_Init message to the HDCP Receiver. It reads AKE\_Send\_Cert from the receiver within 100 ms after writing AKE\_Init message.

If the downstream side does not have  $k_m$  stored corresponding to the *Receiver ID*, it generates  $E_{k_{pub}}(k_m)$  and sends  $E_{k_{pub}}(k_m)$  as part of the AKE\_No\_Stored\_km message to the receiver after verification of signature on  $cert_{rx}$ . It computes H, receives AKE\_Send\_H\_prime message from the receiver containing  $H'$  within one second after writing AKE\_No\_Stored\_km to the receiver and compares  $H'$  against H.

If the downstream side has  $k_m$  stored corresponding to the *Receiver ID*, it sends AKE\_Stored\_km message containing  $E_{k_h}(k_m)$  and  $m$  to the receiver. It computes H, receives AKE\_Send\_H\_prime message from the receiver containing  $H'$  within 200 ms after writing AKE\_Stored\_km to the receiver and compares  $H'$  against H.

If the downstream side does not have a  $k_m$  stored corresponding to the *Receiver ID*, it implements pairing with the HDCP Receiver as explained in Section 2.2.1.

**Transition F1:P1.** This transition occurs on failure of signature verification on  $cert_{rx}$  or if there is a mismatch between H and  $H'$ . This transition also occurs if AKE\_Send\_H\_prime message is not received within one second after writing AKE\_No\_Stored\_km or within 200 ms after writing AKE\_Stored\_km to the receiver.

**Transition F1:F2.** The downstream side implements locality check after successful completion of AKE and pairing.

**State F2: Locality Check.** In this state, the downstream side implements the locality check as explained in Section 2.3 with the HDCP Receiver..

**Transition F2:P1.** This transition occurs on one or more consecutive locality check failures. Locality check fails when  $L'$  is not available for the transmitter to read within 7 ms and the watchdog timer at the downstream side expires or on a mismatch between L and  $L'$ .

**Transition F2:F3.** The downstream side implements SKE after successful completion of locality check.

**State F3: Exchange  $k_s$ .** The downstream side sends encrypted Session Key,  $E_{dkey}(k_s)$ , and  $r_{iv}$  to the HDCP Receiver as part of the SKE\_Send\_Eks message. It may enable HDCP Encryption 200 ms

after sending encrypted Session Key. HDPC Encryption must be enabled only after successful completion of AKE, locality check and SKE stages.

**Transition F3:F4.** This transition occurs after completion of SKE.

**State F4: Test for Repeater.** The downstream side evaluates the REPEATER value that was received in State F1.

**Transition F4:F5.** REPEATER bit is not set (the HDPC Receiver is not an HDPC Repeater).

**State F5: Authenticated.** At this time, and at no prior time, the downstream side has completed the authentication protocol.

A periodic Link Integrity Check is performed to maintain cipher synchronization between the downstream side and the HDPC Receiver.

**Transition F4:F6.** REPEATER bit is set (the HDPC Receiver is an HDPC Repeater).

**State F6: Wait for Receiver ID List.** The downstream side sets up a three-second watchdog timer after sending SKE\_Send\_Eks.

**Transition F6:P1.** The watchdog timer expires before READY has been asserted by the repeater.

**Transition F6:F7.** READY bit is asserted.

**State F7: Verify Receiver ID List.** If a transition in to this state occurs from State F6, the watchdog timer is cleared. The downstream side reads the RepeaterAuth\_Send\_ReceiverID\_List message. If both MAX\_DEVS\_EXCEEDED and MAX\_CASCADE\_EXCEEDED bits are not set, the downstream side computes  $V$  and compares the most significant 128-bits of  $V$  and  $V'$ . The Receiver IDs from this port are added to the Receiver ID list for this HDPC Repeater. The upstream HDPC Transmitter must be informed if topology maximums are exceeded.

**Transition F7:P1.** This transition is made if a mismatch occurs between the most significant 128-bits of  $V$  and  $V'$ . This transition is also made if the downstream side detects a roll-over of  $seq\_num\_V$ . A MAX\_CASCADE\_EXCEEDED or MAX\_DEVS\_EXCEEDED error also causes this transition.

**Transition F7:F8.** This transition occurs on successful verification of the most significant 128-bits of  $V$  and  $V'$ , the downstream side does not detect a roll-over of  $seq\_num\_V$  and the downstream topology does not exceed specified maximums.

**State F8: Send Receiver ID list acknowledgement.** , The downstream side sends the least significant 128-bits of  $V$  to the attached HDPC Repeater as part of the RepeaterAuth\_Send\_Ack message.

The RepeaterAuth\_Send\_Ack message must be received by the attached HDPC Repeater within two seconds from the time the READY status was set and the CP\_IRQ interrupt was asserted by the attached HDPC Repeater..

**Transition F8:F9.** This transition occurs after the RepeaterAuth\_Send\_Ack message has been written to the repeater.

**Transition F5:P1.** This transition occurs if a CP\_IRQ interrupt was received with the REAUTH\_REQ or LINK\_INTEGRITY\_FAILURE status bit set.

**Transition F5:F7.** This transition occurs whenever a CP\_IRQ interrupt is received from the connected HDCP Repeater with READY bit set.

**State F9: Content Stream Management.** This stage is implemented if Content Stream is to be transmitted. The downstream side propagates the Content Stream management information, received from the upstream transmitter, using the RepeaterAuth\_Stream\_Manage message to the attached HDCP Repeater at least 100ms before the transmission of the corresponding Content Streams after HDCP Encryption. If the upstream transmitter is HDCP 2.0-compliant or HDCP 1.x-compliant, the downstream side will not receive the RepeaterAuth\_Stream\_Manage message from the upstream transmitter and assigns a Type value of 0x00 to all Content Streams received from the upstream transmitter and propagates the Content Stream management information using the RepeaterAuth\_Stream\_Manage message.

The downstream side must receive the RepeaterAuth\_Stream\_Ready message from the HDCP Repeater within 100 ms after the transmission of RepeaterAuth\_Stream\_Manage message and verifies  $M'$ . This step fails if the RepeaterAuth\_Stream\_Ready message is not available to read within 100 ms or if  $M$  is not equal to  $M'$ .

This stage may be implemented in parallel with the upstream propagation of topology information (State F4, State F6, State F7 and State F8) and with the flow of encrypted content and Link Integrity Check (State F5). This state may be implemented asynchronously from the rest of the state diagram. A transition in to this state may occur from State F4, State F5, State F6, State F7 or State F8 if Content Stream is to be transmitted and the Content Stream management information is received from the upstream HDCP Transmitter. Also, the transition from State F9 must return to the appropriate state to allow for uninterrupted operation.

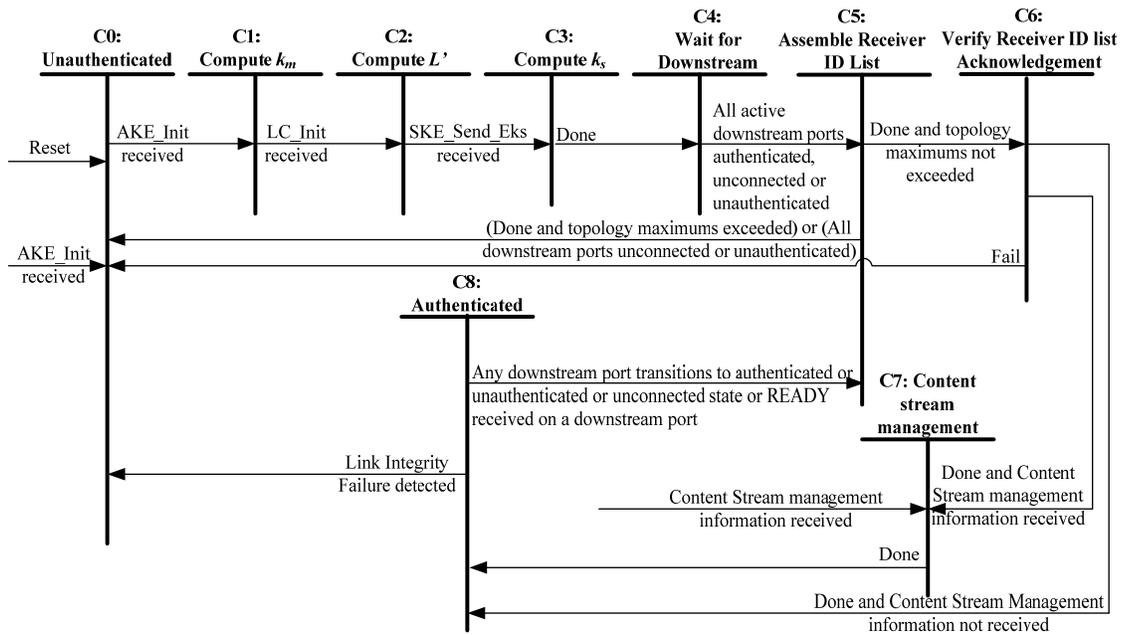
**Transition F9:F5.** This transition occurs on success or failure of the Content Stream management stage.

**Transition F9:P1.** This transition occurs if *seq\_num\_M* rolls over.

Note: Since Link Integrity Check may be implemented in parallel with the upstream propagation of topology information (State F4, State F6, State F7 and State F8) and Content Stream management (State F9) stages, the link integrity checkprocess (i.e. State F5) may be implemented asynchronously from the rest of the state diagram. The transition into State F5 may occur from any state for which encryption is currently enabled. Also, the transition from State F5 returns to the appropriate state to allow for uninterrupted operation.

### 2.10.3 HDCP Repeater Upstream State Diagram

The HDCP Repeater upstream state diagram, illustrated in Figure 2.17, makes reference to states of the HDCP Repeater downstream state diagram. In this state diagram and its following description, the upstream (HDCP Receiver) side refers to the HDCP Receiver functionality within the HDCP Repeater for its corresponding upstream HDCP-protected Interface Port.



**Figure 2.17. HDCP Repeater Upstream Authentication Protocol State Diagram**

**Transitions Any State:C0.** Reset conditions at the HDCP Repeater cause the HDCP Repeater to enter the unauthenticated state. Re-authentication is forced any time AKE\_Init is received from the connected HDCP Transmitter, with a transition through the unauthenticated state.

**State C0: Unauthenticated.** The device is idle, awaiting the reception of AKE\_Init from the HDCP Transmitter to trigger the authentication protocol.

If a transition in to this state occurred from State C5, when State C5 is implemented in parallel with State C8, or from State C6, the HDCP Repeater must set the REAUTH\_REQ status bit in the *RxStatus* register and assert CP\_IRQ interrupt.

If a transition in to this state occurred from State C8, the HDCP Repeater must set the LINK\_INTEGRITY\_FAILURE bit in the *RxStatus* register and assert CP\_IRQ interrupt.

**Transition C0:C1.** AKE\_Init message is received from the HDCP Transmitter.

**State C1: Compute  $k_m$ .** In this state, the upstream (HDCP Receiver) side makes available the AKE\_Send\_Cert message for the transmitter to read in response to AKE\_Init. If AKE\_No\_Stored\_ $k_m$  is received, it decrypts  $k_m$  with  $k_{priv_{rx}}$ , calculates  $H'$ . It makes available the AKE\_Send\_H\_prime message within one second from the time the transmitter finishes writing the AKE\_No\_Stored\_ $k_m$  message parameters.

If AKE\_Stored\_ $k_m$  is received, the upstream side decrypts  $E_{id}(k_m)$  to derive  $k_m$  and calculates  $H'$ . It makes available the AKE\_Send\_H\_prime message within 200 ms from the time the transmitter finishes writing the AKE\_Stored\_ $k_m$  message parameters.

If AKE\_No\_Stored\_ $k_m$  is received, this is an indication to the upstream side that the HDCP Transmitter does not contain a  $k_m$  stored corresponding to its *Receiver ID*. It implements pairing with the HDCP Transmitter as explained in Section 2.2.1.

**Transition C1:C2.** The transition occurs when  $r_n$  is received as part of LC\_Init message from the transmitter.

**State C2: Compute  $L'$ .** The upstream side computes  $L'$  required during locality check and sends LC\_Send\_L\_prime message.

**Transition C2: C3.** The transition occurs when SKE\_Send\_Eks message is received from the transmitter.

**State C3: Compute  $k_s$ .** The upstream side decrypts  $E_{dk_{key}}(k_s)$  to derive  $k_s$ .

**Transition C3: C4.** Successful computation of  $k_s$  causes this transition.

**State C4: Wait for Downstream.** The upstream state machine waits for all downstream HDCP-protected Interface Ports of the HDCP Repeater to enter the unconnected (State P0), unauthenticated (State P1), or the authenticated state (State F5).

**Transition C4:C5.** All downstream HDCP-protected Interface Ports with connected HDCP Receivers have reached the state of authenticated, unconnected or unauthenticated state.

**State C5: Assemble Receiver ID List.** The upstream side assembles the list of all connected downstream topology HDCP Devices as the downstream HDCP-protected Interface Ports reach terminal states of the authentication protocol. An HDCP-protected Interface Port that advances to State P0, the unconnected state, or P1, the unauthenticated state, does not add to the list. A downstream HDCP-protected Interface Port that arrives in State F5 that has an HDCP Receiver that is not an HDCP Repeater connected, adds the *Receiver ID* of the connected HDCP Receiver to the list. Downstream HDCP-protected Interface Ports that arrive in State F5 that have an HDCP Repeater connected will cause the Receiver ID list read from the connected HDCP Repeater, plus the *Receiver ID* of the connected HDCP Repeater itself, to be added to the list.

Note: The upstream side may add the Receiver ID list read from the HDCP Repeater connected to the downstream HDCP-protected Interface port, plus the *Receiver ID* of the connected HDCP Repeater itself to the list after the downstream port has transitioned in to State F8.

When the Receiver ID list for all downstream HDCP Receivers has been assembled, the upstream side computes DEPTH, DEVICE\_COUNT and the upstream  $V'$  and asserts its READY status indicator in the *RxStatus* register and asserts the CP\_IRQ interrupt. In the case of a MAX\_DEVS\_EXCEEDED or a MAX\_CASCADE\_EXCEEDED error, it asserts the corresponding bits to the upstream transmitter. When an HDCP Repeater receives a MAX\_DEVS\_EXCEEDED or MAX\_CASCADE\_EXCEEDED error from a downstream HDCP Repeater, it is required to inform the upstream HDCP Transmitter.

If any downstream port connected to an HDCP Repeater detects the HDCP2\_0\_REPEATER\_DOWNSTREAM or HDCP1\_DEVICE\_DOWNSTREAM bits read from the repeater to be set to one, the upstream side sets the corresponding bits to one in the *RxInfo* register which is read by the upstream HDCP Transmitter as part of the RepeaterAuth\_Send\_ReceiverID\_List message.

**Transition C5:C0.** This transition occurs if RepeaterAuth\_Send\_ReceiverID\_List message has been read by the upstream HDCP Transmitter and topology maximums are exceeded i.e. on a MAX\_DEVS\_EXCEEDED or MAX\_CASCADE\_EXCEEDED error. This transition also occurs if all downstream HDCP-protected Interface Ports have reached the state of unconnected or unauthenticated.

**Transition C5:C6.** RepeaterAuth\_Send\_ReceiverID\_List message has been read by the upstream HDCP Transmitter and topology maximums are not exceeded.

**State C6. Verify Receiver ID list acknowledgement.** In this state, the upstream side receives the RepeaterAuth\_Send\_Ack message from the upstream transmitter and compares the least significant 128-bits of  $V$  and  $V'$ . A match between the least significant 128-bits of  $V$  and  $V'$  indicates successful upstream transmission of topology information. The last byte of the RepeaterAuth\_Send\_Ack message must be written to the upstream side by the transmitter within two seconds from the time the READY status was set and the CP\_IRQ interrupt was asserted by the upstream side..

**Transition C6:C0.** This transition occurs if the RepeaterAuth\_Send\_Ack message is not received by the upstream side within two seconds or on a mismatch between the least significant 128-bits of  $V$  and  $V'$ . If this transition occurs, the upstream side must set the REAUTH\_REQ status bit in the *RxStatus* register and assert CP\_IRQ interrupt.

**Transition C6:C7.** This transition occurs if the RepeaterAuth\_Send\_Ack message is received by the upstream side within two seconds, on a successful match between the least significant 128-bits of  $V$  and  $V'$  and if Content Stream management information is received from the upstream transmitter.

**Transition C6:C8.** This transition occurs if the RepeaterAuth\_Send\_Ack message is received by the upstream side within two seconds, on a successful match between the least significant 128-bits of  $V$  and  $V'$  and if Content Stream management information is not received from the upstream transmitter.

**State C7: Content Stream Management.** On receiving the RepeaterAuth\_Stream\_Manage message, the upstream side computes  $M'$  and makes it available for the the upstream Transmitter to read as part of the RepeaterAuth\_Stream\_Ready message.

This stage may be implemented in parallel with the upstream propagation of topology information (State C4, State C5 and State C6) and with the flow of encrypted content and link integrity check (State C8). This state may be implemented asynchronously from the rest of the state diagram. A transition in to this state may occur from State C4, State C5, State C6 or State C8 if Content Stream management information is received from the upstream transmitter. Also, the transition from State C7 may return to the appropriate state to allow for uninterrupted operation.

The upstream side must be prepared to implement this stage in parallel with the upstream propagation of topology information and with the flow of encrypted content and link integrity check if these stages are implemented in parallel by the upstream transmitter.

**Transition C7:C8.** This transition occurs after RepeaterAuth\_Stream\_Ready message has been read by the upstream transmitter.

**State C8: Authenticated.** The upstream side has completed the authentication protocol. Periodically, it performs a link integrity check.

**Transition C8:C0.** This transition occurs when a link integrity failure is detected by the upstream side as explained in Section 2.6.

**Transition C8:C5.** This transition occurs on detection of any changes to the topology.

This transition occurs when a downstream port that was previously in the unauthenticated (State P1) or unconnected (State P0) state transitions in to the authenticated (State F5) state. For example, the transition may occur when a new HDCP Receiver is connected to a downstream port, that previously had no receivers connected, and the downstream port completes the authentication protocol with the HDCP Receiver.

This transition also occurs when a downstream port that was previously in an authenticated state transitions in to an unauthenticated or unconnected state. For example, the transition may occur when an active, authenticated HDCP Receiver attached to the downstream port is disconnected.

Reception of a CP\_IRQ interrupt with the READY status bit asserted on a downstream port from the connected downstream HDCP Repeater also causes this transition.

Note: Since Link Integrity Check may be implemented in parallel with the upstream propagation of topology information (State C4, State C5 and State C6) and Content Stream management (State C7), the link integrity check process (i.e. State C8) may be implemented asynchronously from the rest of the state diagram. The transition into State C8 may occur from any state for which encryption is currently enabled. Also, the transition from state C8 may return to the appropriate state to allow for uninterrupted operation.

The upstream side must be prepared to implement the link integrity check process in parallel with the upstream propagation of topology information and Content Stream management if these stages are implemented in parallel by the upstream transmitter.

## 2.11 Converters

### 2.11.1 HDCP 2 – HDCP 1.x Converters

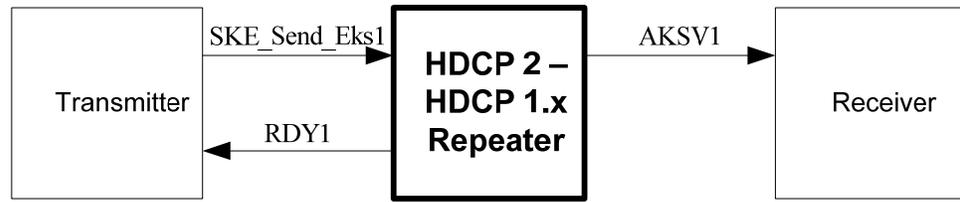
HDCP 2 – HDCP 1.x converters are HDCP Repeaters with an HDCP 2 compliant interface port on the upstream (HDCP Receiver) side and one or more HDCP 1.x compliant interface ports on the downstream (HDCP Transmitter) side.

The HDCP 1.x compliant downstream side implements the state diagram explained in the corresponding HDCP 1.x specification (See Section 1.5).

The HDCP 2 compliant upstream side implements the state diagram as explained in Section 2.10.3 with these modifications.

- **State C5: Assemble Receiver ID List.** The upstream side assembles the list of all connected downstream topology HDCP Devices as the downstream HDCP-protected Interface Ports reach terminal states of the authentication protocol. An HDCP-protected Interface Port that advances to the unconnected state or the unauthenticated state does not add to the list. A downstream HDCP-protected Interface Port that arrives in an authenticated state that has an HDCP Receiver that is not an HDCP Repeater connected, adds the *Bksv* of the connected HDCP Receiver to the Receiver ID list. Downstream HDCP-protected Interface Ports that arrive in an authenticated state that have an HDCP Repeater connected will cause the KSV list read from the connected HDCP Repeater, plus the *Bksv* of the connected HDCP Repeater itself, to be added to the list. KSVs are used in place of *Receiver IDs* and are added to the Receiver ID list in big-endian order

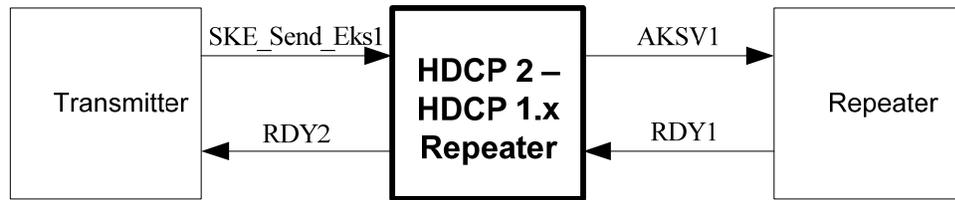
When the Receiver ID list (comprising KSVs of connected downstream HDCP 1.x Receivers, where the KSVs are added to the list in big-endian order) for all downstream HDCP Receivers has been assembled, the upstream side computes DEPTH, DEVICE\_COUNT and the upstream *V'* and asserts its READY status indicator in the *RxStatus* register and asserts the CP\_IRQ interrupt. In the case of a MAX\_DEVS\_EXCEEDED or a MAX\_CASCADE\_EXCEEDED error, it asserts the corresponding bits to the upstream transmitter. When an HDCP Repeater receives a MAX\_DEVS\_EXCEEDED or MAX\_CASCADE\_EXCEEDED error from a downstream HDCP Repeater, it is required to inform the upstream HDCP Transmitter.



**Figure 2.18. HDCP 2 – HDCP 1.x Repeater Protocol Timing with Receiver Attached**

From	To	Max Delay	Conditions and Comments
SKE_Send_Eks1 Session Key received from Upstream HDCP Transmitter	AKSV1 HDCP Repeater's <i>Aksv</i> transmitted downstream	100 ms	Downstream propagation time.
AKSV1 HDCP Repeater's <i>Aksv</i> transmitted downstream	RDY1 Upstream READY asserted	200 ms	Upstream propagation time when no downstream HDCP Repeaters are attached (no downstream KSV lists to process)

**Table 2.3. HDCP 2 – HDCP 1.x Repeater Protocol Timing with Receiver Attached**



**Figure 2.19. HDCP 2 – HDCP 1.x Repeater Protocol Timing with Repeater Attached**

From	To	Max Delay	Conditions and Comments
SKE_Send_Eks1 Session Key received from Upstream HDCP Transmitter	AKSV1 HDCP Repeater's <i>Aksv</i> transmitted downstream	100 ms	Downstream propagation time.
RDY1 Downstream Receiver IDs and topology information received	RDY2 Upstream READY asserted	200 ms	Upstream propagation time when one or more HDCP 1.x-compliant Repeaters are attached. From latest downstream READY. (downstream KSV lists must be processed)

**Table 2.4. HDCP 2 – HDCP 1.x Repeater Protocol Timing with Repeater Attached**

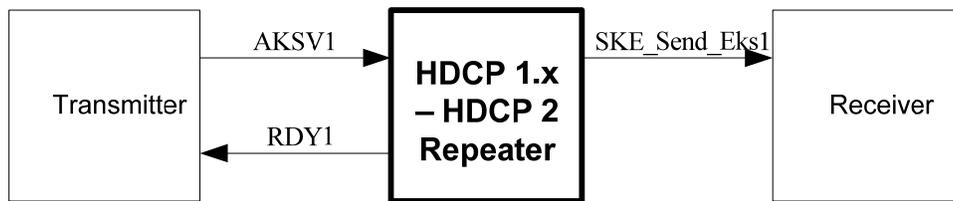
### 2.11.2 HDCP 1.x – HDCP 2 Converters

HDCP 1.x – HDCP 2 converters are HDCP Repeaters with an HDCP 1.x compliant interface port on the upstream (HDCP Receiver) side and one or more HDCP 2 compliant interface ports on the downstream (HDCP Transmitter) side.

The HDCP 1.x compliant upstream side implements the state diagram explained in the corresponding HDCP 1.x specification (See Section 1.5).

The HDCP 2 compliant downstream side implements the state diagram as explained in Section 2.10.2 with these modifications.

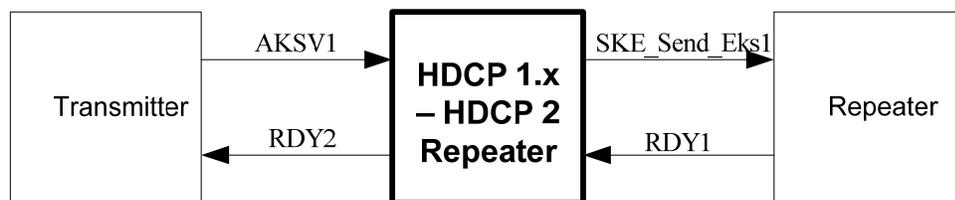
- State F7: Verify Receiver ID List.** If a transition in to this state occurs from State F6, the watchdog timer is cleared. The downstream side reads the RepeaterAuth\_Send\_ReceiverID\_List message. If both MAX\_DEVS\_EXCEEDED and MAX\_CASCADE\_EXCEEDED bits are not set, the downstream side computes  $V$  and compares the most significant 128-bits of  $V$  and  $V'$ . The *Receiver IDs* from this port are used in place of KSVs and are added to the KSV list for this HDCP Repeater. KSV list is constructed by appending *Receiver IDs* in little-endian order. The upstream HDCP Transmitter must be informed if topology maximums are exceeded.



**Figure 2.20. HDCP 1.x – HDCP 2 Repeater Protocol Timing with Receiver Attached**

From	To	Max Delay	Conditions and Comments
AKSV1 Upstream HDCP Transmitter $A_{ksv}$ received	SKE_Send_Eks1 $k_s$ generated by HDCP Repeater transmitted downstream	400 ms	Downstream propagation time.
SKE_Send_Eks1 $k_s$ generated by HDCP Repeater transmitted downstream	RDY1 Upstream READY asserted	500 ms	Upstream propagation time when no downstream HDCP Repeaters are attached (no downstream Receiver ID lists to process)

**Table 2.5. HDCP 1.x – HDCP 2 Repeater Protocol Timing with Repeater Attached**



**Figure 2.21. HDCP 1.x – HDCP 2 Repeater Protocol Timing with Repeater Attached**

From	To	Max Delay	Conditions and Comments
AKSV1 Upstream HDCP Transmitter $A_{ksv}$ received	SKE_Send_Eks1 $k_s$ generated by HDCP Repeater transmitted downstream	400 ms	Downstream propagation time.
RDY1 READY asserted by downstream repeater	RDY2 Upstream READY asserted	500 ms	Upstream propagation time when one or more HDCP Repeaters are attached. From latest downstream READY (downstream Receiver ID lists must be processed)

**Table 2.6. HDCP 1.x – HDCP 2 Repeater Protocol Timing with Repeater Attached**

### 2.12 Session Key Validity

When HDCP Encryption is disabled, the transmitter and receiver ceases to perform HDCP Encryption and stops incrementing the *inputCtr*.

If HDCP Encryption was disabled, from its enabled state, due to the detection of Receiver Connected Indication, Receiver Disconnected Indication or authentication failures, the HDCP Transmitter expires the Session Key. The HDCP Transmitter initiates re-authentication with the transmission of a new AKE\_Init message. In all other cases, where HDCP Encryption was disabled, from its enabled state, while the link was still active and authenticated (for e.g., HDCP Encryption may be briefly disabled during transmission of low value content), the HDCP Transmitter need not expire the Session Key . The HDCP Transmitter may maintain the encryption parameters used during the HDCP Session i.e. *inputCtr* value after the last HDCP Encryption operation (after which HDCP Encryption was disabled),  $k_s$  and  $r_{iv}$ . When encryption is re-enabled, HDCP Encryption may be applied seamlessly, without requiring re-authentication, by using the same stored encryption parameters.

If HDCP Encryption was disabled, from its enabled state, the HDCP Receiver must maintain the *inputCtr* value after the last HDCP Encryption operation (after which HDCP Encryption was disabled),  $k_s$  and  $r_{iv}$  used during the HDCP Session. If encryption was re-enabled, without intervening re-authentication requests from the transmitter, the HDCP Receiver must use the same stored *inputCtr*,  $k_s$  and  $r_{iv}$ .

### 2.13 Random Number Generation

Random number generation is required both in the HDCP Transmitter logic and in the HDCP Receiver logic. Counter mode based deterministic random bit generator using AES-128 block cipher specified in NIST SP 800-90 is the recommended random number generator. The minimum entropy requirement for random values that are not used as secret key material (i.e.  $r_{tx}$ ,  $r_{rx}$ ,  $r_{iv}$ ,  $r_n$ ) is 40 random bits out of 64-bits. This means that a reasonable level of variability or entropy is established if out of 1,000,000 random ( $r_{tx}$ ,  $r_{rx}$ ,  $r_{iv}$  or  $r_n$ ) values collected after the first authentication attempt (i.e. after power-up cycles on the HDCP Transmitter or HDCP Receiver logic), the probability of there being any duplicates in this list of 1,000,000 random values is less than 50%.

For randomly generated secret key material ( $k_m$ ,  $k_s$ ) the minimum entropy requirement is 128-bits of entropy (i.e. the probability of there being any duplicates in the list of  $2^{128}$  secret values ( $k_m$  or  $k_s$ ) collected after power-up and first authentication attempt on the HDCP Transmitter logic is less than 50%).

A list of possible entropy sources that may be used for generation of random values used as secret key material include

- a true Random Number Generator or analog noise source, even if a poor (biased) one
- a pseudo-random number generator (PRNG), seeded by a true RNG with the required entropy, where the state is stored in non-volatile memory after each use. The state must be kept secret. Flash memory or even disk is usable for this purpose as long as it is secure from tampering.

A list of possible entropy sources that may be used for generation of random values not used as secret key material include

- timers, network statistics, error correction information, radio/cable television signals, disk seek times, etc.
- a reliable (not manipulatable by the user) calendar and time-of-day clock. For example, some broadcast content sources may give reliable date and time information.

## 2.14 CP\_IRQ Interrupt Processing

HDCP Transmitters must process CP\_IRQ interrupts when they are received from HDCP Receivers or HDCP Repeaters. HDCP Receivers and HDCP Repeaters must generate CP\_IRQ interrupts as specified in this specification.

The HDCP Transmitter uses the following steps when processing HPD interrupts.

1. If CP\_IRQ is not set, process the interrupt as specified in the DisplayPort specification and exit
2. Read *RxStatus* register
3. If LINK\_INTEGRITY\_FAILURE, abort HDCP session
4. If REAUTH\_REQ, abort HDCP Session
5. If H<sub>1</sub>\_AVAILABLE, read AKE\_Send\_H\_prime message
6. If PAIRING\_AVAILABLE, read AKE\_Send\_Pairing\_Info message
7. If READY, continue with Authentication with Repeaters stage as specified in Section 2.5.1
8. Else ignore interrupt and continue HDCP Session without aborting

## 2.15 HDCP Port

HDCP Transmitters and HDCP Receivers communicate HDCP register values over the AUX channel. HDCP Receivers and HDCP Repeaters must support these HDCP registers. Within the DPCD address space, addresses from 0x69000 are reserved for HDCP2. Multi-byte values are stored in big-endian format.

Offset (hex)	Name	Size in Bytes	Rd/ Wr	Function
0x69000	<i>r<sub>ix</sub></i>	8	Wr	Pseudo-random number written as part of AKE_Init message. This multi-byte value must be written by the HDCP Transmitter before <i>TxCaps</i> is written.
0x69008	<i>TxCaps</i>	3	Wr	This multi-byte value is written as part of AKE_Init message. Refer to Table 2.11 for definitions.

0x6900B	$cert_{rx}$	522	Rd	HDPC Receiver Public Key Certificate read by the HDPC Transmitter as part of AKE_Send_Cert message. This value must be available less than 100 ms from the time the transmitter finishes writing the AKE_Init message parameters i.e. from the time the last byte of $TxCaps$ has been written.
0x69215	$r_{rx}$	8	Rd	Pseudo-random value read as part of AKE_Send_Cert message. This value must be available less than 100 ms from the time the transmitter finishes writing the AKE_Init message parameters i.e. from the time the last byte of $TxCaps$ has been written.
0x6921D	$RxCaps$	3	Rd	This multi-byte value is read as part of AKE_Send_Cert message. Refer to Table 2.10 for definitions. This value must be available less than 100 ms from the time the transmitter finishes writing the AKE_Init message parameters i.e. from the time the last byte of $TxCaps$ has been written.
0x69220	$E_{k_{pub},k_m}$	128	Wr	Encrypted $k_m$ written as part of AKE_No_Stored_km message.
0x692A0	$E_{kh,k_m}$	16	Wr	Encrypted $k_m$ written as part of AKE_Stored_km message. This multi-byte value must be written by the HDPC Transmitter before $m$ is written
0x692B0	$m$	16	Wr	Initialization Vector used for encryption of $k_m$ during the pairing process and written as part of the AKE_Stored_km message.
0x692C0	$H'$	32	Rd	HMAC computed during AKE and read as part of AKE_Send_H_prime message. This value must be available less than one second from the time the transmitter finishes writing the AKE_No_Stored_km message parameters i.e. from the time the last byte of $E_{k_{pub},k_m}$ has been written. This value must be available less than 200 ms from the time the transmitter finishes writing the AKE_Stored_km message parameters i.e. from the time the last byte of $m$ has been written.
0x692E0	$E_{kh,k_m}$	16	Rd	Encrypted $k_m$ read as part of AKE_Send_Pairing_Info message. This value must be available less than 200 ms from the time the transmitter begins reading the AKE_Send_H_prime message i.e. the $H'$ value.
0x692F0	$r_n$	8	Wr	Pseudo-random number written as part of LC_Init message.
0x692F8	$L'$	32	Rd	HMAC computed during Locality Check and read as part of LC_Send_L_prime message. This value must be available for the transmitter to read within 7 ms from the time the transmitter finishes writing the LC_Init message parameters i.e. from the time the last byte of $r_n$ has been written.
0x69318	$E_{dkey,k_s}$	16	Wr	Encrypted session key written as part of SKE_Send_Eks message. This multi-byte value must be written by the HDPC Transmitter before $r_{iv}$ is written. This value must be written to the HDPC Receiver at least 200 ms before enabling HDPC Encryption.
0x69328	$r_{iv}$	8	Wr	Pseudo-random number written as part of SKE_Send_Eks message. This value must be written to the HDPC Receiver at least 200 ms before enabling HDPC Encryption.
0x69330	$RxInfo$	2	Rd	Refer to Table 2.9 for definitions.

0x69332	<i>seq_num_V</i>	3	Rd	Sequence number used for the computation of $V/V'$ and read as part of RepeaterAuth_Send_ReceiverID_List message.
0x69335	$V'$	16	Rd	Most-significant 128-bits of HMAC computed during upstream propagation of topology information and read as part of RepeaterAuth_Send_ReceiverID_List message.
0x69345	Receiver ID List	155	Rd	List of Receiver ID read as part of RepeaterAuth_Send_ReceiverID_List message. All bytes read as 0x00 for HDCP Receivers that are not HDCP Repeaters (REPEATER = 0).
0x693E0	$V$	16	Wr	Least-significant 128-bits of HMAC computed during upstream propagation of topology information and written as part of RepeaterAuth_Send_Ack message. The last byte of this value must be written less than two seconds from the time the READY status was set and the CP_IRQ interrupt was asserted by the HDCP Repeater..
0x693F0	<i>seq_num_M</i>	3	Wr	Sequence number used for the computation of $M/M'$ and written as part of RepeaterAuth_Stream_Manage message. This multi-byte value is written before $k$ is written. This value must be written at least 100ms before the transmission of the corresponding Content Streams after HDCP Encryption.
0x693F3	$k$	2	Wr	This value indicates the number of Content Streams that are being transmitted by the HDCP Transmitter to the attached HDCP Repeater during the HDCP Session and is written as part of RepeaterAuth_Stream_Manage message. This multi-byte value is written before <i>StreamID_Type</i> is written. This value must be written at least 100ms before the transmission of the corresponding Content Streams after HDCP Encryption.
0x693F5	<i>StreamID_Type</i>	126	Wr	Concatenation of Stream Identifiers and Type values of Content Streams that are being transmitted by the HDCP Transmitter to the attached HDCP Repeater during the HDCP Session and is written as part of RepeaterAuth_Stream_Manage message. This value must be written at least 100ms before the transmission of the corresponding Content Streams after HDCP Encryption.
0x69473	$M'$	32	Rd	HMAC computed during downstream propagation of Content Stream management information and read as part of RepeaterAuth_Stream_Ready message. This value must be available less than 100 ms from the time the transmitter finishes writing the RepeaterAuth_Stream_Manage message parameters i.e. from the time the last byte of <i>StreamID_Type</i> has been written.
0x69493	<i>RxStatus</i>	1	Rd	Refer to Table 2.8 for definitions.
0x69494	Rsvd	132	Rd	All bytes read as 0x00
0x69518	dbg	64	Rd/ Wr	Implementation-specific debug registers. Confidential values must not be exposed through these registers.

**Table 2.7. HDCP Addresses in DPCD**

Name	Bit Field	Rd/ Wr	Description
Rsvd	7:5	Rd	Reserved. Read as zero
LINK_INTEGRITY_FAILURE	4	Rd	When set to one, indicates that loss of cipher synchronization was detected at the HDCP Receiver during a link integrity check. This value must be reset by the HDCP Receiver on every new authentication request by the HDCP Transmitter as indicated by a write of the AKE_Init message.
REAUTH_REQ	3	Rd	When set to one, indicates that the upstream side of the HDCP Repeater has transitioned in to an unauthenticated state from State C5, when State C5 is implemented in parallel with State C8, or from State C6 (See Section 2.10.3). The HDCP Transmitter may initiate re-authentication with the HDCP Repeater. This value must be reset by the HDCP Receiver on every new authentication request by the HDCP Transmitter as indicated by a write of the AKE_Init message.
PAIRING_AVAILABLE	2	Rd	When set to one, indicates that $E_{kh}(k_m)$ is available for reading at the HDCP Receiver. This value must be reset by the HDCP Receiver as soon as $E_{kh}(k_m)$ is read by the HDCP Transmitter.
H'_AVAILABLE	1	Rd	When set to one, indicates that $H'$ is available for reading at the HDCP Receiver. This value must be reset by the HDCP Receiver as soon as $H'$ is read by the HDCP Transmitter.
READY	0	Rd	When set to one, this HDCP Repeater has built the list of downstream Receiver IDs and computed the verification value $V'$ . READY must be reset by the HDCP Repeater as soon as $Rxinfo$ has been read by the HDCP Transmitter. This value is always zero during the computation of $V'$ . READY bit must be set less than three seconds from the time the transmitter finishes writing the SKE_Send_Eks message parameters i.e. from the time the last byte of $r_{iv}$ has been written

**Table 2.8. RxStatus Register Bit Field Definitions**

Name	Bit Field	Rd/Wr	Description
Rsvd	15:12	Rd	Reserved. Read as zero
DEPTH	11:9	Rd	Repeater cascade depth. This value gives the number of attached levels through the connection topology.
DEVICE_COUNT	8:4	Rd	Total number of attached downstream devices. Always zero for HDCP Receivers. This count does not include the HDCP Repeater itself, but only devices downstream from the HDCP Repeater.
MAX_DEVS_EXCEEDED	3	Rd	Topology error indicator. When set to one, more than 31 downstream devices are attached.
MAX_CASCADE_EXCEEDED.	2	Rd	Topology error indicator. When set to one, more than four levels of repeaters have been cascaded together.
HDCP2_0_REPEATER_DOWNSTREAM	1	Rd	When set to one, indicates presence of an HDCP2.0-compliant Repeater in the topology
HDCP1_DEVICE_DOWNSTREAM	0	Rd	When set to one, indicates presence of an HDCP 1.x-compliant Device in the topology

**Table 2.9. RxInfo Register Bit Field Definitions**

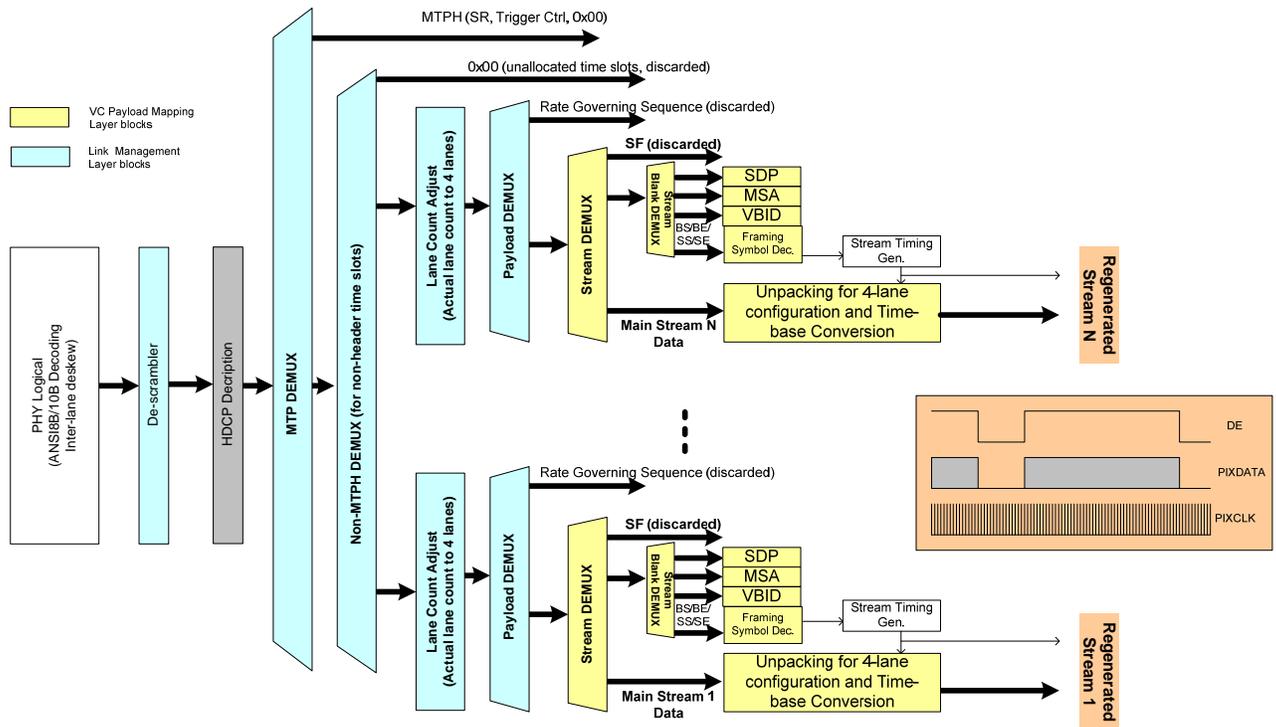
Name	Bit Field	Rd/Wr	Description
VERSION	23:16	Rd	The HDCP Receiver must set VERSION to 0x02
RECEIVER_CAPABILITY_MASK	15:2	Rd	Reserved. Read as zero
HDCP_CAPABLE	1	Rd	When set to one, indicates that the receiver is HDCP capable. This bit does not change while the HDCP Receiver is active.
REPEATER	0	Rd	When set to one, this HDCP Receiver supports downstream connections as permitted by the Digital Content Protection LLC license. This bit does not change while the HDCP Receiver is active.

**Table 2.10. RxCaps Register Bit Field Definitions**

Name	Bit Field	Rd/Wr	Description
VERSION	23:16	Wr	The HDCP Transmitter must set VERSION to 0x02
TRANSMITTER_CAPABILITY_MASK	15:0	Wr	Reserved. Read as zero

**Table 2.11. TxCaps Register Bit Field Definitions**





**Figure 3-2. HDCP Decryption in the DisplayPort Receiver**

When HDCP Encryption is applied to a timeslot in the MST mode or when HDCP Encryption is enabled in the SST mode, all data symbols (including video data, secondary data and dummy symbols) must be encrypted and K-codes must not be encrypted. Section 3.3 and Section 3.4 explains in detail the encryption signaling protocol that is used to enable/disable HDCP Encryption.

The HDCP Cipher is clocked at the following rates.

- For 4-lane Main link configurations, the HDCP Cipher is clocked for every four link symbol clocks ( $LS\_CLK/4$ )
- For 2-lane Main link configurations, the HDCP Cipher is clocked at  $LS\_CLK/8$
- For 1-lane Main link configurations, the HDCP Cipher is clocked  $LS\_CLK/16$

Unless otherwise specified, all references to clock in this specification denote the HDCP Cipher clock. The clock is applied regardless of K-codes or data symbols.

The mappings of the 128-bit HDCP Cipher key stream to the DisplayPort Lanes are shown in Table 3-1, Table 3-2 and Table 3-3.

DisplayPort Lane	Cipher Key Stream	Symbol
3	127:120	15

	95:88	11
	63:56	7
	31:24	3
2	119:112	14
	87:80	10
	55:48	6
	23:16	2
1	111:104	13
	79:72	9
	47:40	5
	15:8	1
0	103:96	12
	71:64	8
	39:32	4
	7:0	0

**Table 3-1. Encryption Stream Mapping for 4-lane Main Link Configuration**

<b>DisplayPort Lane</b>	<b>Cipher Key Stream</b>	<b>Symbol</b>
1	127:120	15
	111:104	13
	95:88	11
	79:72	9
	63:56	7
	47:40	5
	31:24	3
	15:8	1
0	119:112	14
	103:96	12

	87:80	10
	71:64	8
	55:48	6
	39:32	4
	23:16	2
	7:0	0

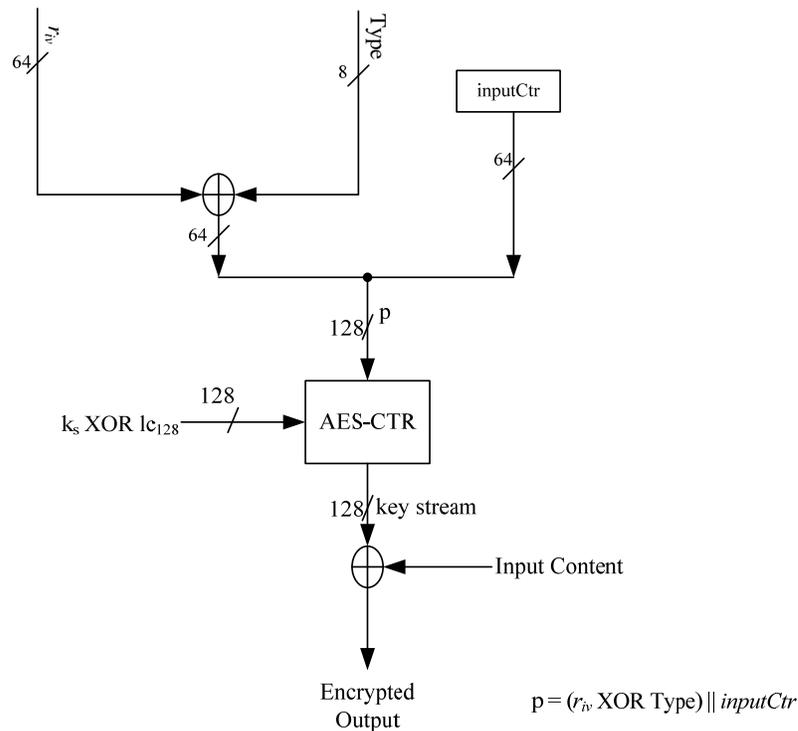
**Table 3-2. Encryption Stream Mapping for 2-lane Main Link Configuration**

DisplayPort Lane	Cipher Key Stream	Symbol
0	127:120	15
	119:112	14
	111:104	13
	103:96	12
	95:88	11
	87:80	10
	79:72	9
	71:64	8
	63:56	7
	55:48	6
	47:40	5
	39:32	4
	31:24	3
	23:16	2
	15:8	1
7:0	0	

**Table 3-3. Encryption Stream Mapping for 1-lane Main Link Configuration**

### 3.2 HDCP Cipher

The HDCP cipher consists of a 128-bit AES module that is operated in a Counter (CTR) mode as illustrated in Figure 3-3.



**Figure 3-3. HDCP Cipher Structure**

$k_s$  is the 128-bit Session Key which is XORed with  $lc_{128}$ .

$p = (r_{iv} \text{ XOR } \text{Type}) \parallel \text{inputCtr}$ . All values are in big-endian order.

The Type value, associated with the Content Stream to be encrypted, is XORed with the least significant 8-bits of  $r_{iv}$ . Type value is associated with the STREAM\_ID corresponding to a Content Stream as explained in Section 4.2.12. *inputCtr* is a 64-bit counter. It is initialized to zero when HDCP Encryption is enabled for the first time during the HDCP Session i.e. immediately after SKE and must not be reset at any other time. HDCP Encryption of data symbols begins with an *inputCtr* value of zero.

The HDCP Cipher, illustrated in Figure 3-3, may be viewed as two logical HDCP Cipher modules where one module is associated with a Type value of 0x00 and the other module is associated with a Type value of 0x01. Both logical HDCP Cipher modules are clocked at the same rate and the *inputCtr* associated with both modules are always in sync.

When the HDCP Cipher is clocked, it produces a 128-bit block of key stream and increments the *inputCtr* following generation of the key stream. The key stream is XORed with 32-bit data symbols as shown in Table 3-1, Table 3-2 and Table 3-3. The value of *inputCtr* must never be reused for a given set of encryption parameters i.e.  $k_s$  and  $r_{iv}$ .

When HDCP Encryption is enabled, the HDCP Cipher continues to be clocked irrespective of whether HDCP Encryption is applied to the HDCP Content i.e. the HDCP Cipher continues to be clocked even when control symbols are detected or when the XOR Enable/Disable Indicator bits indicate HDCP Encryption must not be applied to the corresponding timeslots. The HDCP Cipher is not clocked when HDCP Encryption is disabled.

### 3.3 Encryption Status Signaling in MST Mode

Encryption Status Signaling is the process by which an HDCP Transmitter signals to a downstream device that encryption is enabled or disabled and indicates whether or not encryption must be applied to the following data symbols.

In the MST mode, an HDCP Encryption Indicator bit is used to indicate whether HDCP Encryption is enabled or disabled. XOR Enable/Disable Indicator bits are used to indicate whether HDCP Encryption must be applied to a specific MTP timeslot. HDCP Encryption is enabled or disabled at link frame boundaries using an HDCP Encryption Indicator bit in the MTP Header.

A multi-stream link may be comprised of different Content Streams where some Content Streams may require HDCP Encryption to be applied and some for which HDCP Encryption is not required. An XOR Enable/Disable indication, corresponding to each of the 63 MTP timeslots, contained in MTP Header bits indicates whether or not HDCP Encryption must be applied to each timeslot. XOR Enable/Disable indication is valid only when HDCP Encryption is enabled as indicated by the HDCP Encryption Indicator bit. When HDCP Encryption is disabled as indicated by the HDCP Encryption Indicator bit, any further XOR Enable/Disable indication must be ignored.

The HDCP Encryption Indicator bit and the bits used for XOR Enable/Disable indication are represented by a 64-bit data structure referred to as `HDCP_Encryption_Control`, where

`HDCP_Encryption_Control[0]` = HDCP Encryption Indicator bit. When HDCP Encryption is disabled, the HDCP Encryption Indicator bit is set to 0. When HDCP Encryption is enabled, the HDCP Encryption Indicator bit is set to 1.

`HDCP_Encryption_Control[1..63]` = XOR Enable/Disable Indicator bits. `HDCP_Encryption_Control[1]` corresponds to XOR Enable/Disable indication for timeslot 1 and so on. XOR Enable/Disable Indicator bits are set to 0 (XOR Disabled) when HDCP Encryption must not be applied to the corresponding timeslots and they are set to 1 (XOR Enabled) when HDCP Encryption must be applied to the corresponding timeslots.

The 64-bit `HDCP_Encryption_Control` is contained in the `Encryption_Control_Field` which is described in the DisplayPort Specification (see References). The `HDCP_Encryption_Control` occupies eight MTP Headers.

The `HDCP_Encryption_Control` is transmitted in the `Encryption_Control_Field` as explained in the DisplayPort specification (see References). The `HDCP_Encryption_Control` consists of an 8 (scrambled) data code sequence spanning consecutive MTPH's. The data code sequence is identical per-lane, regardless of lane count. The `HDCP_Encryption_Control` is repeated four consecutive times, resulting in a total sequence length of 32 MTPs. HDCP Receivers apply majority voting to the repeated sequence for error correction, as described in DisplayPort specification. Unless otherwise noted, references in the HDCP specification to receiver use of the `HDCP_Encryption_Control` refer to the post-error corrected result.

The `HDCP_Encryption_Control` must be transmitted starting exactly 36 MTPs prior to each link frame boundary SR signal, and immediately prior to any standalone ACT<sup>1</sup> sequence. A single `HDCP_Encryption_Control` must be transmitted immediately preceding any (optional) back to back ACT/SR sequence.

The `HDCP_Encryption_Control` starting 36 MTPs before the SR is used to enable or disable HDCP Encryption in addition to indicating XOR Enable/Disable status for timeslots. The HDCP

---

<sup>1</sup> For this section, a *standalone ACT* is defined as an ACT not part of a back-to-back ACT/SR sequence

Encryption Indicator bit is valid only at link frame boundaries and HDCP Encryption is enabled/disabled only at link frame boundaries.

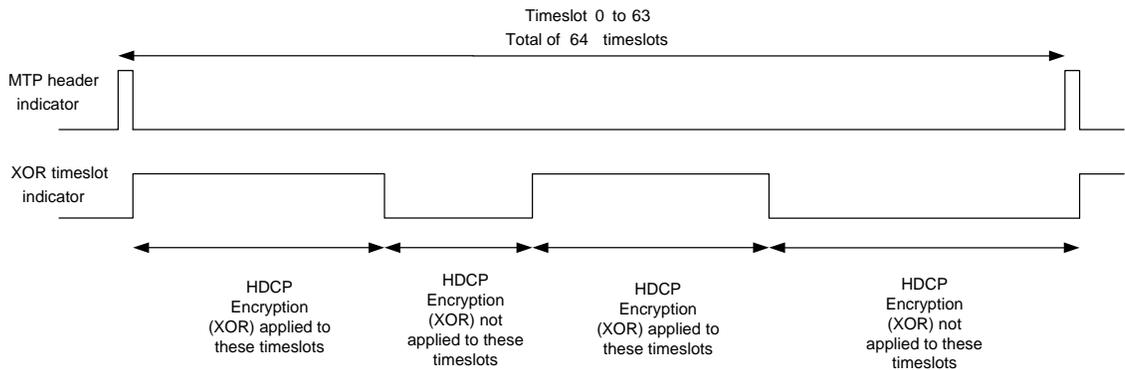
The HDCP\_Encryption\_Control preceding a standalone ACT must not be used to enable or disable HDCP Encryption. It is only used to indicate XOR Enable/Disable status for every timeslot. The HDCP Encryption Indicator bit in the HDCP\_Encryption\_Control preceding a standalone ACT must be ignored.

In cases of the (optional) back-to-back ACT/SR sequence an HDCP Transmitter must transmit a single HDCP\_Encryption\_Control sequence preceding the ACT/SR pair. The HDCP\_Encryption\_Control preceding an ACT/SR pair is used to enable/disable HDCP Encryption in addition to indicating XOR Enable/Disable status for timeslots.

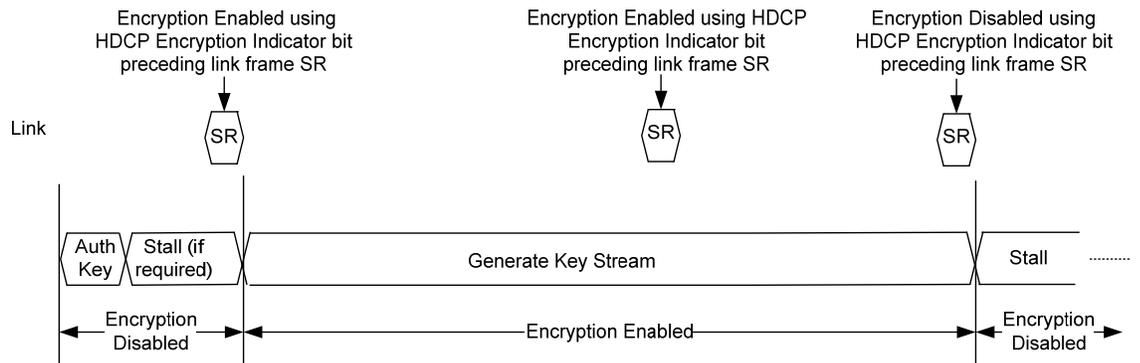
HDCP Devices must support and use Enhanced Framing Mode. Refer to the DisplayPort Specification for more details regarding enhanced framing mode.

When HDCP Encryption is enabled, the HDCP Cipher is clocked, as explained in Section 3.1, following the subsequent SR. The resulting key stream bits are used to encrypt the data symbols following SR, as explained in Section 3.2, if the XOR Enable/Disable Indicator bits indicate encryption must be applied to the corresponding timeslots. The HDCP Cipher is not clocked when HDCP Encryption is disabled.

HDCP Encryption is disabled at the link frame boundary by the transmission of the HDCP\_Encryption\_Control starting 36 MTPs before an SR or immediately preceding a back-to-back ACT/SR sequence with HDCP Encryption Indicator bit set to 0.

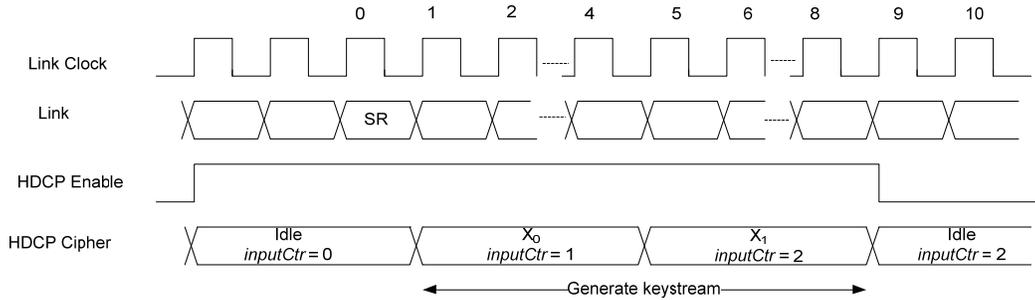


**Figure 3-4. HDCP Encryption Applied Based on HDCP\_Encryption\_Control Bits 1:63**

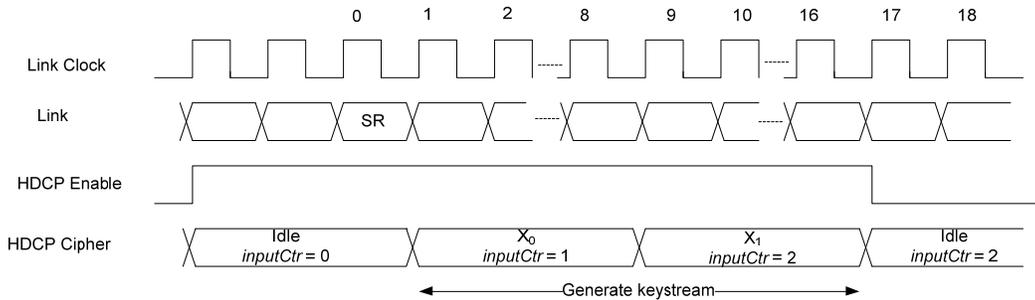


**Figure 3-5. Encryption Status Signaling in MST Mode and HDCP Cipher Operations**

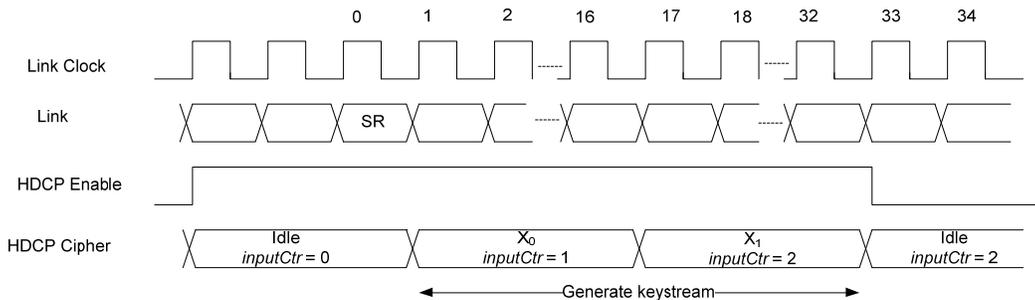
In Figure 3-6, Figure 3-7 and Figure 3-8,  $X_i$  denotes the HDCP Cipher state that is used to generate the 128-bit keystream where  $i$  is the value of the *inputCtr* used during keystream generation. For each HDCP Cipher clock, the HDCP Cipher generates the keystream and increments the *inputCtr* following keystream generation.



**Figure 3-6. 4-Lane HDCP Cipher Operations in MST Mode**



**Figure 3-7. 2-Lane HDCP Cipher Operations in MST Mode**



**Figure 3-8. 1-Lane HDCP Cipher Operations in MST Mode**

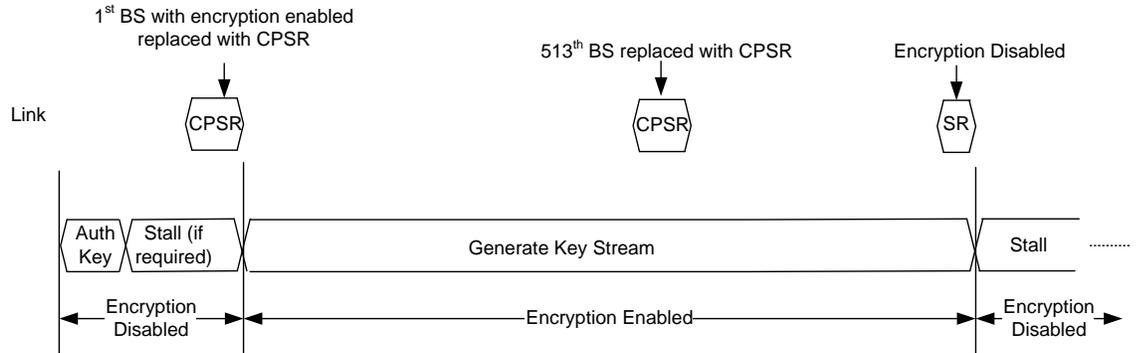
**3.4 Encryption Status Signaling in SST Mode**

The DisplayPort transmitter inserts a blanking start (BS) symbol after each line of video is transmitted including the last line of a frame. For audio only transmissions, a BS symbol is transmitted every 8192 symbols per lane. Every 512<sup>th</sup> BS symbol is replaced by a scrambler reset (SR) symbol. When encryption is currently disabled, the BS and SR control symbols are

transmitted. When encryption is currently enabled, BS and SR control symbols are replaced by CPBS and CPSR control symbols respectively.

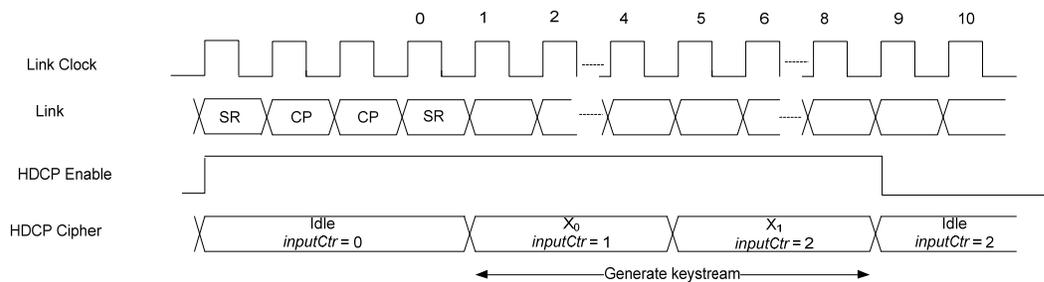
In the SST mode, detection of SR indicates that encryption is disabled and detection of CPSR indicates that encryption is enabled.

Following the transmission or detection of CPSR, the HDCP Cipher is clocked as explained in Section 3.1 and the resulting key stream bits are used to encrypt the data symbols following the CPSR as explained in Section 3.2. The HDCP Cipher is not clocked when HDCP Encryption is disabled.

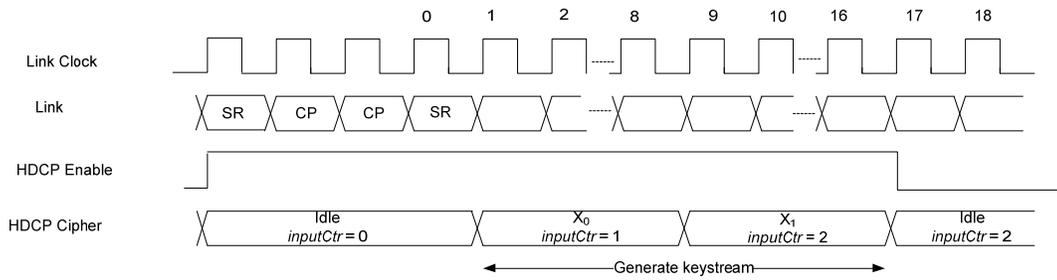


**Figure 3-9. Encryption Status Signaling in SST Mode and HDCP Cipher Operations**

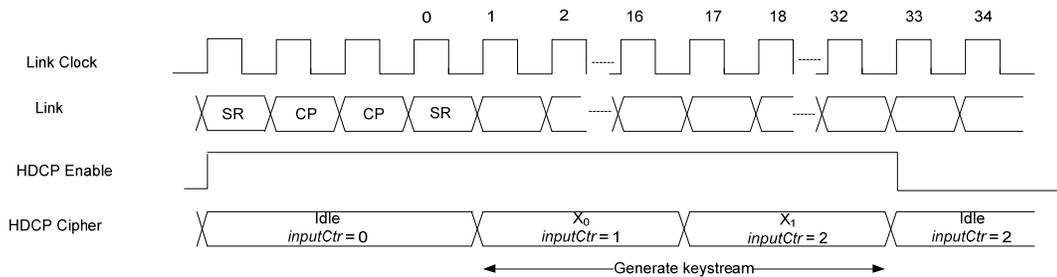
In Figure 3-10, Figure 3-11 and Figure 3-12,  $X_i$  denotes the HDCP Cipher state that is used to generate the 128-bit keystream where  $i$  is the value of the *inputCtr* used during keystream generation. For each HDCP Cipher clock, the HDCP Cipher generates the keystream and increments the *inputCtr* following keystream generation.



**Figure 3-10. 4-Lane HDCP Cipher Operations in SST Mode**



**Figure 3-11. 2-Lane HDCP Cipher Operations in SST Mode**



**Figure 3-12. 1-Lane HDCP Cipher Operations in SST Mode**

**3.5 Uniqueness of  $k_s$  and  $r_{iv}$**

HDPC Receivers and HDPC Repeaters with multiple inputs may share the same Public Key Certificates and Private Keys across all inputs. The HDPC Transmitter (including downstream side of HDPC Repeater) must negotiate distinct  $k_m$  with each directly connected downstream HDPC Device. While  $r_{tx}$  used during each HDPC Session is required to be fresh, transmitters with multiple downstream HDPC links must ensure that each link receives a distinct  $r_{tx}$  value.

As illustrated in Figure 3-13, HDPC Transmitters, including downstream side of HDPC Repeaters, with multiple downstream HDPC links may share the same  $k_s$  and  $r_{iv}$  across those links only if HDPC Content from the same HDPC Cipher module following all the requirements specified in Section 3.2 is transmitted to those links.

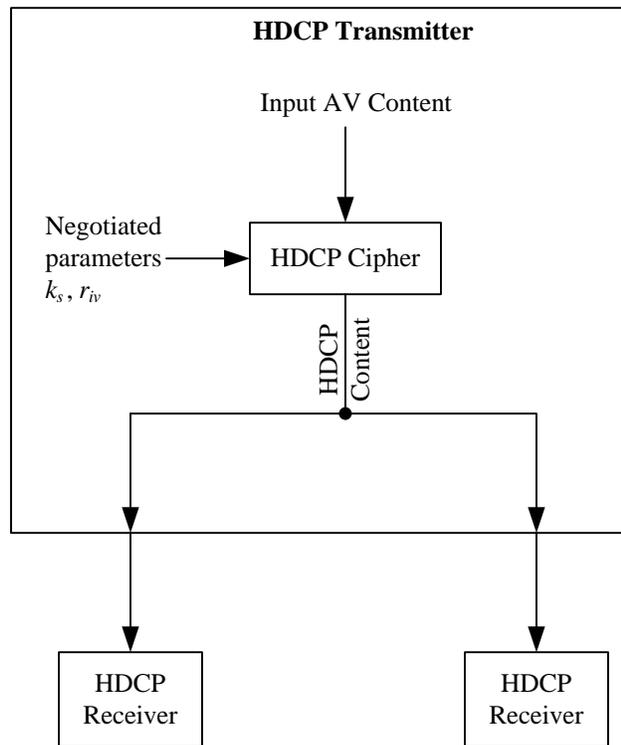


Figure 3-13.  $k_s$  and  $r_{iv}$  Shared across HDCP Links

## 4 Authentication Protocol Messages

### 4.1 Overview

Message parameters are specified in the order that they are read or written by the HDCP Transmitter. For example, in the case of the AKE\_Init message, the transmitter first writes the  $r_{tx}$  parameter to the receiver followed by  $TxCaps$ . In the case of the AKE\_Send\_Cert message, the transmitter first reads the  $cert_{rx}$  parameter followed by  $r_{rx}$  and finally  $RxCaps$ .

### 4.2 Message Format

#### 4.2.1 AKE\_Init (Write)

Syntax	No. of Bytes
AKE_Init { $r_{tx}[63..0]$ $TxCaps$ }	8 3

Table 4.1. AKE\_Init Format

#### 4.2.2 AKE\_Send\_Cert (Read)

The HDCP Transmitter attempts to read AKE\_Send\_Cert beginning with  $cert_{rx}$  within 100 ms after writing the AKE\_Init message i.e. after the last byte of  $TxCaps$  has been written. The HDCP Transmitter must not attempt to read AKE\_Send\_Cert sooner than 100 ms after writing the AKE\_Init message.

Syntax	No. of Bytes
AKE_Send_Cert { $cert_{rx}[4175..0]$ $r_{rx}[63..0]$ $RxCaps$ }	522 8 3

Table 4.2. AKE\_Send\_Cert Format

#### 4.2.3 AKE\_No\_Stored\_km (Write)

Syntax	No. of Bytes
AKE_No_Stored_km { $E_{kpub\_k_m}[1023..0]$ }	128

Table 4.3. AKE\_No\_Stored\_km Format

#### 4.2.4 AKE\_Stored\_km (Write)

Syntax	No. of Bytes
AKE_Stored_km { $E_{k_t\_k_m}[127..0]$ $m[127..0]$ }	16 16

Table 4.4. AKE\_Stored\_km Format

#### 4.2.5 AKE\_Send\_H\_prime (Read)

AKE\_Send\_H\_prime must be available for the transmitter to read within one second after writing the AKE\_No\_Stored\_km message i.e. after the last byte of  $E_{kpub\_k_m}$  has been written or within 200 ms after writing the AKE\_Stored\_km message i.e. after the last byte of  $m$  has been written.

Syntax	No. of Bytes
AK_Send_H_prime{ <i>H</i> [255..0] }	32

**Table 4.5. AKE\_Send\_H\_prime Format**

#### 4.2.6 AKE\_Send\_Pairing\_Info (Read)

AKE\_Send\_Pairing\_Info must be available for the transmitter to read within 200 ms from the time the transmitter finishes writing the AKE\_Send\_H\_prime message parameters to the HDCP Receiver i.e. after the last byte of *H'* has been written

Syntax	No. of Bytes
AKE_Send_Pairing_Info{ <i>E<sub>kh</sub>-k<sub>m</sub></i> [127..0] }	16

**Table 4.6. AKE\_Send\_Pairing\_Info Format**

#### 4.2.7 LC\_Init (Write)

Syntax	No. of Bytes
LC_Init { <i>r<sub>n</sub></i> [63..0] }	8

**Table 4.7. LC\_Init Format**

#### 4.2.8 LC\_Send\_L\_prime (Read)

The LC\_Send\_L\_prime message must be available for the transmitter to read within 7 ms from the time the transmitter finishes writing the LC\_Init message parameters to the HDCP Receiver i.e. after the last byte of *r<sub>n</sub>* has been written.

Syntax	No. of Bytes
LC_Send_L_prime{ <i>L</i> [255..0] }	32

**Table 4.8. LC\_Send\_L\_prime Format**

#### 4.2.9 SKE\_Send\_Eks (Write)

Syntax	No. of Bytes
SKE_Send_Eks{ <i>E<sub>dkey</sub>-k<sub>s</sub></i> [127..0] <i>r<sub>iv</sub></i> [63..0] }	16 8

**Table 4.9. SKE\_Send\_Eks Format**

#### 4.2.10 RepeaterAuth\_Send\_ReceiverID\_List (Read)

Receiver ID list is constructed by appending *Receiver IDs* in big-endian order.

Receiver ID list = *Receiver ID<sub>0</sub>* // *Receiver ID<sub>1</sub>* // ... // *Receiver ID<sub>n-1</sub>*, where n is the DEVICE\_COUNT.

If the computed DEVICE\_COUNT for an HDCP Repeater exceeds 31, the repeater sets the *RxInfo*.MAX\_DEVS\_EXCEEDED bit to one. If the computed DEPTH for an HDCP Repeater

exceeds four, the repeater sets *RxInfo*.MAX\_CASCADE\_EXCEEDED bit to one. If topology maximums are not exceeded, *RxInfo*.MAX\_DEVS\_EXCEEDED and *RxInfo*.MAX\_CASCADE\_EXCEEDED are set to zero.

The HDCP Repeater sets *RxInfo*.HDCP2\_0\_REPEATER\_DOWNSTREAM bit to one if an HDCP 2.0-compliant repeater is attached to any one of its downstream port, else it sets *RxInfo*.HDCP2\_0\_REPEATER\_DOWNSTREAM to zero.

The HDCP Repeater sets *RxInfo*.HDCP1\_DEVICE\_DOWNSTREAM to one if an HDCP 1.x-compliant Device i.e. an HDCP 1.x-compliant Receiver or an HDCP 1.x-compliant Repeater is attached to any one of its downstream port, else it sets *RxInfo*.HDCP1\_DEVICE\_DOWNSTREAM to zero.

When the HDCP Repeater receives HDCP2\_0\_REPEATER\_DOWNSTREAM or HDCP1\_DEVICE\_DOWNSTREAM bits that are set from a downstream HDCP Repeater, it must propagate this information to the upstream HDCP Transmitter by setting the corresponding bits in the RepeaterAuth\_Send\_ReceiverID\_List message.

Syntax	No. of Bytes
RepeaterAuth_Send_ReceiverID_List{	
<i>RxInfo</i>	2
<i>seq_num_V</i>	3
<i>V</i> [255..128]	16
Receiver ID List	5*DEVICE_COUNT
}	

**Table 4.10. RepeaterAuth\_Send\_ReceiverID\_List Format**

#### 4.2.11 RepeaterAuth\_Send\_Ack (Write)

The last byte of the RepeaterAuth\_Send\_Ack message i.e the last byte of *V*'[127..0] must be written to the repeater by the transmitter within two seconds from the time the READY status was set and the CP\_IRQ interrupt was asserted by the HDCP Repeater and the downstream topology does not exceed specified maximums.

Syntax	No. of Bytes
RepeaterAuth_Send_Ack{	
<i>V</i> [127..0]	16
}	

**Table 4.11. RepeaterAuth\_Send\_Ack Format**

#### 4.2.12 RepeaterAuth\_Stream\_Manage (Write)

Content Streams are assigned a Type value by the most upstream HDCP Transmitter based on instructions received from the Upstream Content Control Function.

The STREAM\_ID, assigned to a Content Stream as specified in the DisplayPort specification, is followed by its assigned Type value in the RepeaterAuth\_Stream\_Manage message. All Content Streams transmitted by the HDCP Transmitter to the HDCP Repeater, after HDCP Encryption, are assigned Type values.

Syntax	No. of Bytes
RepeaterAuth_Stream_Manage{ <i>seq_num_M</i> <i>k</i> <i>StreamID_Type</i> }	3 2 2*k

**Table 4.12. RepeaterAuth\_Stream\_Manage Format**

$StreamID\_Type = STREAM\_ID_1 \parallel Type \parallel STREAM\_ID_2 \parallel Type \parallel \dots \parallel STREAM\_ID_k \parallel Type$

STREAM\_ID assigned to a Content Stream is concatenated with its assigned Type value. All values are in big-endian order.

Parameter *k* is the number of Content Streams that are being transmitted by the HDCP Transmitter to the attached HDCP Repeater during the HDCP Session.

Parameter	No. of Bytes	Description
STREAM_ID	1	Stream Identifier, corresponding to the Content Stream, as defined in the DisplayPort specification
Type	1	0x00 : Type 0 Content Streams. May be transmitted by the HDCP Repeater to all HDCP Devices. 0x01 : Type 1 Content Streams. Must not be transmitted by the HDCP Repeater to HDCP 1.x-compliant Devices and HDCP 2.0-compliant Repeaters 0x02 – 0xFF : Reserved for future use only. Content Streams with reserved Type values must be treated similar to Type 1 Content Streams i.e. must not be transmitted by the HDCP Repeater to HDCP 1.x-compliant Devices and HDCP 2.0-compliant Repeaters

**Table 4.13. STREAM\_ID, Type Description**

#### 4.2.13 RepeaterAuth\_Stream\_Ready (Read)

The RepeaterAuth\_Stream\_Ready message must be available for the transmitter to read within 100 ms from the time the transmitter finishes writing the RepeaterAuth\_Stream\_Manage message parameters to the HDCP Receiver i.e. after the last byte of *StreamID\_Type* has been written.

Syntax	No. of Bytes
RepeaterAuth_Stream_Ready{ <i>M</i> [255..0] }	32

**Table 4.14. RepeaterAuth\_Stream\_Ready Format**

## 5 Renewability

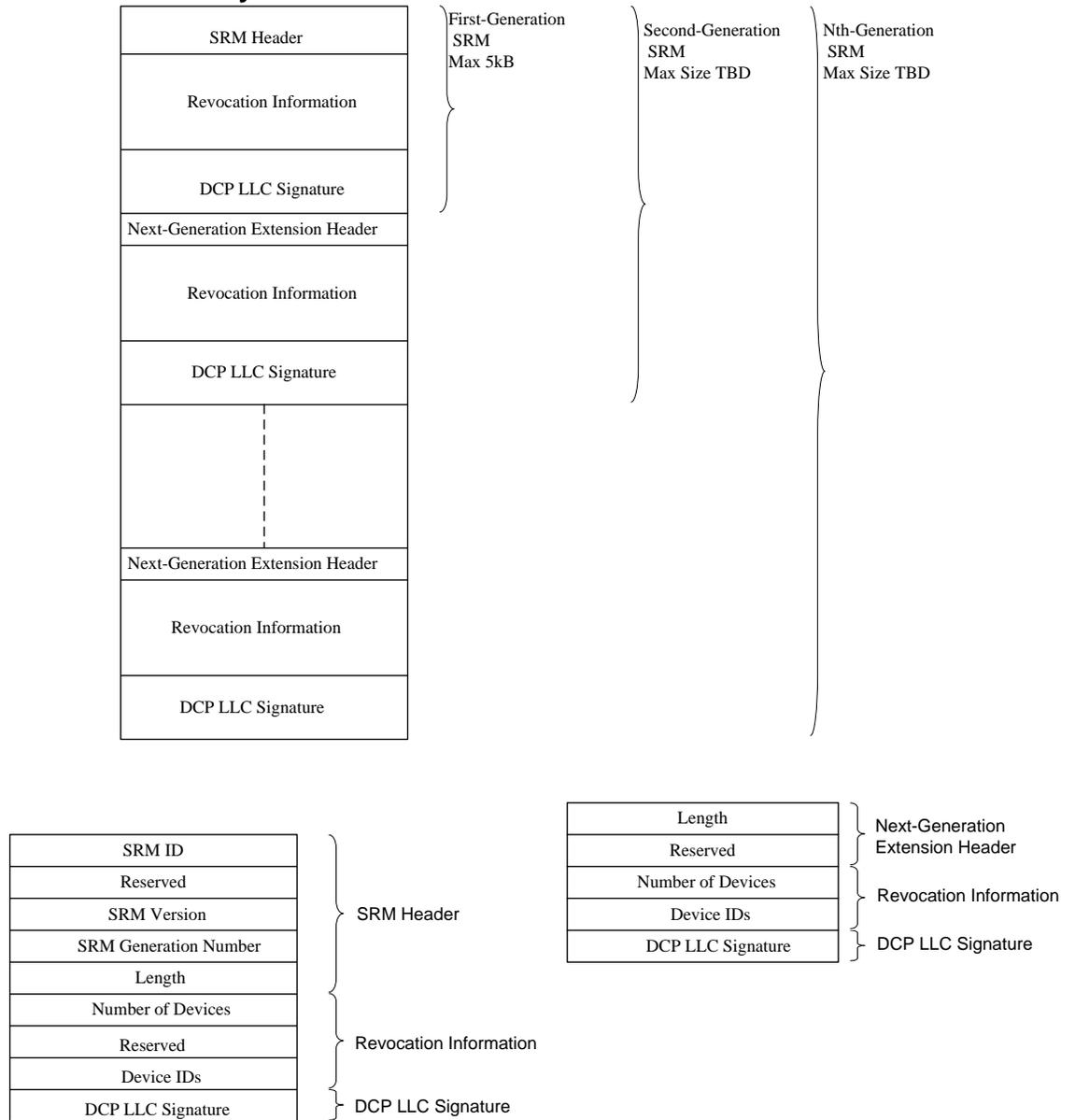
It is contemplated that an authorized participant in the authentication protocol may become compromised so as to expose the RSA private keys it possesses for misuse by unauthorized parties. In consideration of this, each HDCP Receiver is issued a unique Receiver ID which is contained in  $cert_{rx}$ . Through a process defined in the HDCP Adopter's License, the Digital Content Protection LLC may determine that an HDCP Receiver's RSA private key,  $kpriv_{rx}$ , has been compromised. If so, it places the corresponding Receiver ID on a revocation list that the HDCP Transmitter checks during authentication.

The HDCP Transmitter is required to manage system renewability messages (SRMs) carrying the Receiver ID revocation list. The validity of an SRM is established by verifying the integrity of its signature with the Digital Content Protection LLC public key, which is specified by the Digital Content Protection LLC.

For interoperability with HDCP 1.x, KSVs of revoked HDCP 1.x devices will be included in the HDCP 2 SRM, in addition to the HDCP 1.x SRM. Similarly, Receiver IDs of revoked HDCP 2 devices will be included in the HDCP 1.x SRM, in addition to the HDCP 2 SRM.

The SRMs are delivered with content and must be checked when available. The Receiver IDs must immediately be checked against the SRM when a new version of the SRM is received. Additionally, devices compliant with HDCP 2.0 and higher must be capable of storing at least 5kB of the SRM in their non-volatile memory. The process by which a device compliant with HDCP 2.0 or higher updates the SRM stored in its non-volatile storage when presented with a newer SRM version is explained in Section 5.2.

### 5.1 SRM Size and Scalability



**Figure 5.1. SRM Generational Format**

As illustrated in Figure 5.1, the size of the First-Generation HDCP SRM will be limited to a maximum of 5kB. The actual size of the First-Generation SRM is 5116 bytes. For scalability of the SRM, the SRM format supports next-generation extensions. By supporting generations of SRMs, an HDCP SRM can, if required in future, grow beyond the 5kB limit to accommodate more Receiver IDs. Next-generation extensions are appended to the current-generation SRM in order to ensure backward compatibility with devices that support only previous-generation SRMs.

Table 5.1 gives the format of the HDCP 2 SRM. All values are stored in big endian format.

Name	Size (bits)	Function
SRM ID	4	A value of 0x9 signifies that the message is for HDCP 2. All other

		values are reserved. SRMs with values other than 0x9 must be ignored.
HDCP2 Indicator	4	A value of 0x1 signifies that the message is for HDCP2
Reserved	8	Reserved for future definition. Must be 0x00
SRM Version	16	Sequentially increasing unique SRM numbers. Higher numbered SRMs are more recent
SRM Generation Number	8	Indicates the generation of the SRM. The generation number starts at 1 and increases sequentially
Length	24	Length in bytes and includes the combined size of this field (three bytes) and all following fields contained in the first-generation SRM i.e. size of this field, Number of Devices field, Reserved (22 bits) field, Device IDs field and Digital Content Protection LLC signature field (384 bytes) in the first-generation SRM
Number of Devices	10	Specifies the number (N1) of Receiver IDs / KSVs contained in the first-generation SRM
Reserved	22	Reserved for future definition. All bits set to 0
Device IDs	40 * N1 Max size for this field is 37760 (4720 bytes)	40-bit Receiver IDs / KSVs
DCP LLC Signature	3072	A cryptographic signature calculated over all preceding fields of the SRM. RSASSA-PKCS1-v1_5 is the signature scheme used as defined by PKCS #1 V2.1: RSA Cryptography Standard. SHA-256 is the underlying hash function

**Table 5.1. System Renewability Message Format**

Each subsequent next-generation extensions to the first-generation SRM will have the following fields.

Name	Size (bits)	Function
Length	16	Length in bytes and includes the combined size of this field (two bytes) and all following fields contained in this next-generation extension i.e. size of this field, Number of Devices field, Reserved (6 bits) field, Device IDs field and Digital Content Protection LLC signature field (384 bytes) in this next-generation SRM
Reserved	6	Reserved for future definition. All bits set to 0
Number of Devices	10	Specifies the number (N2) of Receiver IDs / KSVs contained in this next generation extension
Device IDs	40 * N2	40-bit Receiver IDs / KSVs
DCP LLC Signature	3072	A cryptographic signature calculated over all preceding fields of the SRM. RSASSA-PKCS1-v1_5 is the signature scheme used as defined by PKCS #1 V2.1: RSA Cryptography Standard. SHA-256 is the underlying hash function

**Table 5.2. Next-generation extension format**

## 5.2 Updating SRMs

The stored HDCP SRM must be updated when a newer version of the SRM is delivered with the content. The procedure for updating an SRM is as follows:

1. Verify that the version number of the new SRM is greater than the version number of the SRM currently stored in the device's non-volatile storage
2. If the version number of the new SRM is greater (implying that it is a more recent version), verify the signature on the new SRM

On successful signature verification, replace the current SRM in the device's non-volatile storage with the new SRM. If, for instance, the device supports only second-generation SRMs and the new SRM is a third-generation SRM, the device is not required to store the third-generation extension. Devices compliant with HDCP 2.0 or higher must be capable of storing at least 5kB (actual size is 5116 bytes) of the SRM (First-Generation SRM).

## Appendix A. Core Functions and Confidentiality and Integrity of Values

Table A.1 identifies the requirements of confidentiality and integrity for values within the protocol. A *confidential* value must never be revealed. The *integrity* of many values in the system is protected by fail-safe mechanisms of the protocol. Values that are not protected in this manner require active measures beyond the protocol to ensure integrity. Such values are noted in the table as requiring integrity. Core Functions must be implemented in Hardware. The values used by Core Functions, along with the corresponding Core Functions by which they are used, are identified in the table.

Value	Confidentiality Required <sup>±</sup> ?	Integrity Required <sup>±</sup> ?	Value used by Core Functions?	Core Function
$lc_{128}$	Yes	Yes	Yes	HDCP Encryption and Decryption
$k_{pub_{dcp}}$	No	Yes	No	N/A
$cert_{rx}$	No	No	No	N/A
$k_{pub_{rx}}$	No	Yes	No	N/A
Receiver ID	No	Yes	No	N/A
$k_{priv_{tx}}$	Yes	Yes	Yes	Handling of Device Secret Key, during AKE, in plaintext form
$r_{tx}$	No	Yes*	Yes	
$r_{iv}$	No	Yes*	Yes	N/A
REPEATER	No	Yes	No	N/A
$r_{rx}$	No	Yes**	Yes	N/A
$k_m$	Yes	Yes*	Yes	Handling of Master Key, during AKE (including Pairing) and Key Derivation, in plaintext form
$k_d$	Yes	Yes*	No	N/A
$dkey_0, dkey_1$	Yes	Yes*	No	N/A

<sup>±</sup> According to the robustness rules in the HDCP Adopter's License

\* Only within the transmitter

\* Only within the transmitter

\*\* Only within the receiver

$dkey_2$	Yes	Yes*	Yes	Handling of information or materials during Key Derivation and SKE, including but not limited to cryptographic keys used to encrypt or decrypt HDCP Core Keys ( $k_s$ ), from which HDCP Core Keys could reasonably be derived
ctr	No	Yes*	Yes	N/A
H	Yes	Yes	No	N/A
$H'$	No	No	No	N/A
$m$	No	No	Yes	N/A
$k_h$	Yes	Yes	Yes	Handling of information or materials during Pairing, including but not limited to cryptographic keys used to encrypt or decrypt HDCP Core Keys ( $k_m$ ), from which HDCP Core Keys could reasonably be derived
$r_n$	No	Yes*	Yes	N/A
L	Yes	Yes	No	N/A
$L'$	No	No	No	N/A
$k_s$	Yes	Yes*	Yes	Handling of Session Key, during SKE and HDCP Encryption/Decryption, in plaintext form
V	Yes	Yes	No	N/A
$V'$	No	No	No	N/A
Receiver ID list	No	Yes	No	N/A
DEPTH	No	Yes	No	N/A
DEVICE_COUNT	No	Yes	No	N/A
MAX_DEVS_EXCEEDED	No	Yes	No	N/A
MAX_CASCADE_EXCEEDED	No	Yes	No	N/A
<i>inputCtr</i>	No	Yes*	Yes	HDCP Encryption and Decryption

<i>Type</i>	No	Yes*	Yes	HDCP Encryption and Decryption
p	No	Yes*	Yes	HDCP Encryption and Decryption

**Table A.1. Core Functions and Confidentiality and Integrity of Values**

**Appendix B. DCP LLC Public Key**

Table B.1 gives the production DCP LLC public key.

Parameter	Value (hexadecimal)	
Modulus n	B0E9 AA45 F129 BA0A 1CBE 1757 28EB 2B4E	
	8FD0 C06A AD79 980F 8D43 8D47 04B8 2BF4	
	1521 5619 0140 013B D091 9062 9E89 C227	
	8ECF B6DB CE3F 7210 5093 8C23 2983 7B80	
	64A7 59E8 6167 4CBC D858 B8F1 D4F8 2C37	
	9816 260E 4EF9 4EEE 24DE CCD1 4B4B C506	
	7AFB 4965 E6C0 0083 481E 8E42 2A53 A0F5	
	3729 2B5A F973 C59A A1B5 B574 7C06 DC7B	
	7CDC 6C6E 826B 4988 D41B 25E0 EED1 79BD	
	3985 FA4F 25EC 7019 23C1 B9A6 D97E 3EDA	
	48A9 58E3 1814 1E9F 307F 4CA8 AE53 2266	
	2BBE 24CB 4766 FC83 CF5C 2D1E 3AAB AB06	
	BE05 AA1A 9B2D B7A6 54F3 632B 97BF 93BE	
	C1AF 2139 490C E931 90CC C2BB 3C02 C4E2	
	BDBD 2F84 639B D2DD 783E 90C6 C5AC 1677	
	2E69 6C77 FDED 8A4D 6A8C A3A9 256C 21FD	
	B294 0C84 AA07 2926 46F7 9B3A 1987 E09F	
	EB30 A8F5 64EB 07F1 E9DB F9AF 2C8B 697E	
	2E67 393F F3A6 E5CD DA24 9BA2 7872 F0A2	
	27C3 E025 B4A1 046A 5980 27B5 DAB4 B453	
	973B 2899 ACF4 9627 0F7F 300C 4AAF CB9E	
	D871 2824 3EBC 3515 BE13 EBAF 4301 BD61	
	2454 349F 733E B510 9FC9 FC80 E84D E332	
	968F 8810 2325 F3D3 3E6E 6DBB DC29 66EB	
	Public Exponent e	03

**Table B.1. DCP LLC Public Key**

## Appendix C. Bibliography (Informative)

These documents are not normatively referenced in this specification, but may provide useful supplementary information.

ITU-T Recommendation H.222.0 / ISO/IEC 13818-1 (2006) Amendment 1 (Jan. 2007), *Transport of MPEG-4 streaming text and MPEG-4 lossless audio over MPEG-2 systems*

ITU-T Recommendation H.222.0 / ISO/IEC 13818-1 (2006) Amendment 2 (Aug. 2007), *Carriage of auxiliary video data*

SMPTE 2022-1-2007, *Forward Error Correction for Real-Time Video/Audio Transport Over IP Networks*, May 2007

SMPTE 2022-2-2007, *Unidirectional Transport of Constant Bit Rate MPEG-2 Transport Streams on IP Networks*, May 2007

*Interoperability for Professional Video Streaming over IP Networks*, SMPTE Motion Imaging Journal, Feb./March 2005,  
<http://www.broadcastpapers.com/whitepapers/Path1InteropVideoIP.pdf?CFID=16660544&CFTOKEN=dd0a39cb99517fc5-3203F7CF-F879-0B3E-45C4A402626C372C>