High-bandwidth Digital Content Protection System

Revision 1.0

17 February 2000

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.

Acknowledgement

Silicon Image Inc. has 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, JF2-53
C/O Intel Corporation
2111 NE 25th Ave
Hillsboro, OR 97124

Telephone: (503) 264-6576

Fax: (503) 264-4151

Email: info@digital-cp.com

Web: www.digital-cp.com

Revision History

1 September 99 – 0.80 Revision. Initial publication at Intel Developer Forum
13 October 99 – 0.89 Revision. Publication at Copy Protection Technical Working Group
11 November 99 – 0.90 Revision. Publication at Copy Protection Technical Working Group
11 January 00 – 0.95 Revision. Publication at Copy Protection Technical Working Group
17 February 00 – 1.00 Revision. Publication at Intel Developer Forum

ISBN 0-9675129-4-8

1 Introduction

1.1 Scope

This specification describes the High-bandwidth Digital Content Protection (HDCP) system for protecting Digital Visual Interface (DVI) outputs from being copied. The system requires modifications to both display devices and to host graphics systems to provide a protected link between the host (video transmitter) and the display device (video receiver).

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. Additionally, video transmitters may be subject to additional robustness and compliance rules associated with other content protection technologies.

1.2 Overview

HDCP is designed to protect the video transmission between a DVI video transmitter and a DVI video receiver. The system also allows for video receivers that support protected downstream DVI connections. These devices are referred to as video repeaters in Figure 1–1, which illustrates an example connection topology for video transmitters, receivers, and repeaters. The HDCP system allows up to seven levels of video repeaters and as many as 128 total devices, including repeaters, to be attached to a host DVI port.

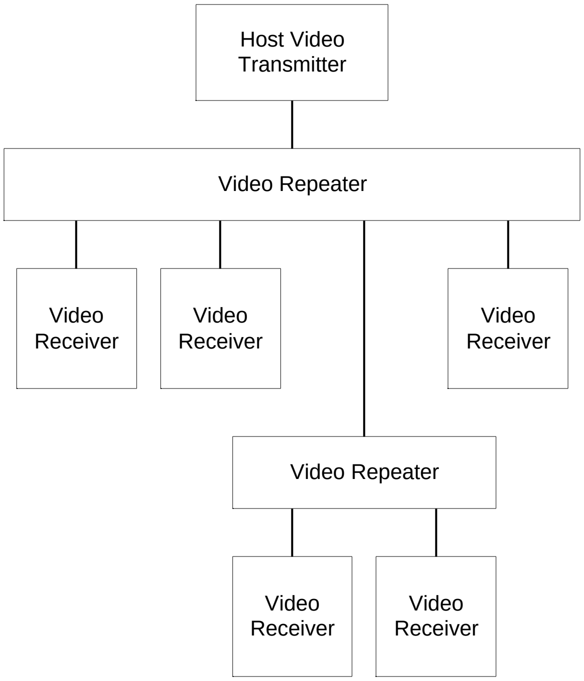

Figure 1–1. HDCP Connection Topology

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 video transmitter verifies that a given video receiver is licensed to receive protected content. With the legitimacy of the video receiver determined, encrypted data 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 content, renewability allows a video transmitter to identify such compromised devices and prevent the transmission of protected content.

This document contains chapters describing in detail the requirements of each of these elements. In addition, a chapter is devoted describing the cipher that is used in both the authentication protocol and in the encryption of the video. All aspects of HDCP map easily onto the existing DVI specification.

1.3 Terminology

Throughout this specification, names that appear in italic refer to values that are exchanged during the HDCP cryptographic protocol. Names that appear in CAPS refer to status values from the video receiver. 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.

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.

1.4 References

Digital Display Working Group (DDWG), *Digital Visual Interface (DVI) Revision 1.0*, April 2, 1999.

National Institute of Standards and Technology (NIST), *Digital Signature Standard (DSS)*, FIPS Publication 186-1, December 15, 1998.

National Institute of Standards and Technology (NIST), *Secure Hash Standard (SHS)*, FIPS Publication 180-1, April 17, 1995.

Philips Semiconductors, *The I^2C-Bus Specification*, Version 2.0, December 1998.

2 Authentication

The HDCP Authentication protocol is an exchange between a video transmitter and a video receiver that affirms to the transmitter that the receiver is authorized to receive the protected information. This affirmation is in the form of the receiver demonstrating knowledge of a set of secret device keys. Each authorized device is provided with a unique set of secret device keys from the Digital Content Protection LLC. The communication exchange, which allows for the receiver to demonstrate knowledge of such secret device keys, also provides for both parties to generate a shared secret value that cannot be determined by eavesdroppers on this exchange. By having this shared secret formation melded into the demonstration of authorization, the shared secret can then be used as a symmetric key to encrypt video content intended only for the authorized receiver. Thus, a communication path is established between the transmitter and receiver that only authorized parties can access.

2.1 Overview

Each authorized participant (e.g. licensed monitor device, graphics controller device, etc.) receives an array of 40, 56-bit secret device keys and a corresponding identifier from the Digital Content Protection LLC. This identifier is the Key Selection Vector (KSV) assigned to the device. The KSV is a 40-bit binary value.

The HDCP Authentication Protocol can be considered in three parts. The first part establishes shared values between the two devices if both devices have a valid array of secret device keys and corresponding KSVs. The second part allows a video repeater to report the KSVs of attached video receivers. The third part occurs during the vertical blanking interval preceding each video frame for which encryption is enabled, and provides an initialization state for the HDCP Cipher for encrypting the RGB pixel stream of that frame.

2.2 Protocol

Figure 2–1 illustrates the first part of the authentication exchange. The video transmitter (*Device A*) can initiate authentication at any time, even before a previous authentication exchange has completed. Authentication is initiated by the video transmitter by sending an initiation message containing its KSV (*Aksv*) and a 64-bit pseudo-random value (*An*) generated by the HDCP Cipher function hdcpRngCipher (Section 4.5) to the video receiver (*Device B*). The video receiver responds by sending a response message containing the video receiver’s KSV (*Bksv*) and the REPEATER bit, which indicates if the video receiver is a repeater. The video transmitter verifies that the video receiver’s KSV has not been revoked (section 5), and that the received KSV contains 20 ones and 20 zeros.

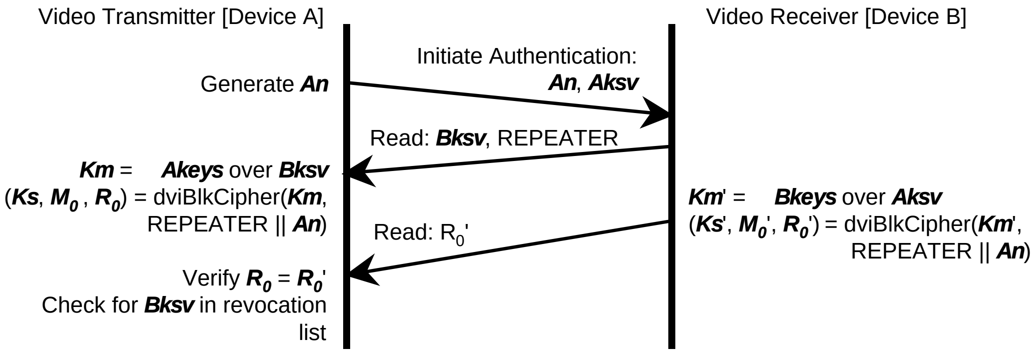
Video Transmitter [Device A]
Video Receiver [Device B]
Generate An
Initiate Authentication:
An, Aksv
Read: Bksv, REPEATER
Km = Akeys over Bksv
(Ks, M0 , R0) = dviBlkCipher(Km, REPEATER || An)
Km' = Bkeys over Aksv
(Ks', M0', R0') = dviBlkCipher(Km', REPEATER || An)
Read: R0'
Verify R0 = R0'
Check for Bksv in revocation list

Figure 2–1. First Part of Authentication Protocol

At this point, if both devices have a valid array of secret device keys and corresponding KSV from the Digital Content Protection LLC, then they can each calculate a 56-bit shared secret value, *Km* (or *Km*' in the video receiver). Each device calculates *Km* (or *Km*') by adding a selection of its private device keys described by the other device's KSV, using 56-bit binary addition (i.e. unsigned addition modulo 2^{56}). The selection of secret device keys that are added together consists of those corresponding to the bit indexes of all of the 1-bits of the binary representation of the KSV.

For example, suppose *Bksv* equals 0x5A3. For the binary representation of 0x5A3, bit positions 0, 1, 5, 7, 8, and 10 are ones and all other bit positions are zeros. Therefore, *Device A* will add it's own secret device keys at array indexes 0, 1, 5, 7, 8, and 10 together to calculate the shared secret value, *Km*. *Device B* will perform an analogous calculation using its own private key array and *Device A*'s KSV to get *Km*'.

If either device has an invalid set of secret device keys or corresponding KSV, then *Km* will not be equal to *Km*'.

The HDCP Cipher function hdcpBlockCipher (Section 4.5) is then used to calculate three values, *Ks*, M_0, and R_0. The cipher initialization values for this calculation are *Km* (or *Km*'), and the 65-bit concatenation of REPEATER with *An*. The video receiver status bit REPEATER indicates that the video receiver supports retransmission of video to additional DVI video receivers. The session key *Ks* is a 56-bit secret key for the HDCP Cipher. M_0 is a 64-bit secret value used in the second part of the authentication protocol, and as a supplemental HDCP Cipher initialization value. R_0' is a 16-bit response value that the video receiver returns to the video transmitter to provide an indication as to the success of the authentication exchange. R_0' must be available for the video transmitter to read within 100 milliseconds from the time that the video transmitter finishes writing *Aksv* to the video receiver.

If authentication was successful, then R_0' will be equal to R_0. If authentication was unsuccessful, then R_0' and R_0 will, in most cases, differ. Future R_i' values, produced during the third part of the authentication protocol, will reveal that authentication has failed in the event that the R_0 values erroneously indicate that authentication was successful.

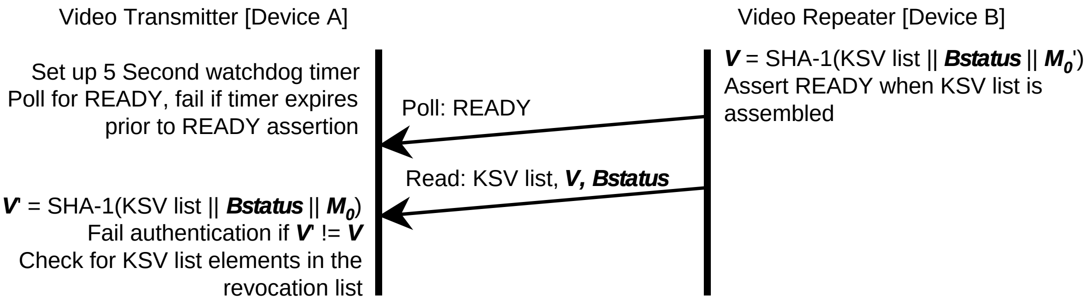
Video Transmitter [Device A]
Video Repeater [Device B]
Set up 5 Second watchdog timer
Poll for READY, fail if timer expires prior to READY assertion
V = SHA-1(KSV list || $Bstatus$ || M_0')
Assert READY when KSV list is assembled
Poll: READY
Read: KSV list, V, $Bstatus$
V' = SHA-1(KSV list || $Bstatus$ || M_0)
Fail authentication if V' != V
Check for KSV list elements in the revocation list

Figure 2–2. Second Part of Authentication Protocol

The second part of the authentication protocol (Figure 2–2) is required for all video transmitters and video repeaters. The video transmitter executes the second part of the protocol only when the REPEATER bit is set, indicating that the attached device is a video repeater. This part of the protocol assembles a list of all KSVs attached to the DVI host through a the permitted connection tree, enabling video source functions at the host to perform revocation.

Video repeaters assemble the list of all attached devices as the downstream ports of the video repeater complete the authentication protocol with attached video receivers. The list is represented by a contiguous set of bytes, with each KSV occupying five bytes stored in little-endian order. The total length of the KSV list is five times the total number of attached devices. An unconnected port adds nothing to the list. A port connected to a video receiver (as opposed to a video repeater), adds the *Bksv* of the attached video receiver to the list. Ports that have a video repeater attached add the KSV list of the attached video repeater, plus the *Bksv* of the attached video repeater. In order to add the KSV list of the attached video repeater, it is necessary for the video repeater to verify the integrity of the list by computing *V* and checking this value against *V*' received from the attached video repeater. If *V* does not equal *V*', the downstream KSV list integrity check fails, and the upstream video repeater must not assert its READY status. Upstream devices will detect this failure by the expiration of a watchdog time set in the video transmitter.

When the video repeater has assembled the complete list of attached devices' KSVs, itcomputes and appends to the list the verification value *V*. This value is the SHA–1 hash of the concatenation of the KSV list, *Bstatus*, and the secret value M_0. When both the KSV list and *V* are available, the video repeater asserts its READY status indicator.

The video transmitter, having determined that the REPEATER bit read earlier in the protocol is set, sets a five-second watchdog timer and polls the receiver's READY status bit. When READY is set, the video transmitter reads the KSV list and *V* from the repeater. The video transmitter verifies the integrity of the KSV list by computing the SHA–1 hash value *V* and comparing this value to *V*'. If *V* is not equal to *V*', then the authentication protocol is aborted.

If the asserted READY status is not received within a maximum-permitted time of five seconds, authentication of the video repeater fails. With this failure, the video transmitter abandons the authentication protocol with the video repeater. Authentication can be reattempted with the transmission of a new value *An* and the *Aksv*.

In addition to assembling the KSV, list video repeater propagates topology information upward through the connection tree to the DVI host. A video repeater reports the topology status variables DEVICE_COUNT and DEPTH. The DEVICE_COUNT for a video repeater is equal to the total number of attached downstream video receivers. The value is calculated as the sum of the number of attached downstream ports plus the sum of the DEVICE_COUNT of all attached video repeaters. The DEPTH status for a video repeater is equal to the maximum number of connection levels below any of the downstream DVI ports. The value is calculated as the maximum DEPTH reported from downstream video repeaters plus one (accounting for the attached downstream video repeater). For example, a video repeater with zero downstream devices reports a value of zero for both the DEPTH and the DEVICE_COUNT. A video repeater with four downstream video receivers (not repeaters) reports a DEPTH of one and a DEVICE_COUNT of four. If the computed DEVICE_COUNT for a repeater exceeds 127, the repeater must assert the MAX_DEVS_EXCEEDED status bit. If the computed DEPTH for a repeater exceeds seven, the repeater must assert the MAX_CASCADE_EXCEEDED status bit. When a repeater receives a MAX_DEVS_EXCEEDED or a MAX_CASCADE_EXCEEDED status from a downstream repeater, it is required to assert the corresponding status bits upstream.

Authentication fails if the topology maximums are exceeded. All video transmitters check to see if the KSV of any attached device is found in the current revocation list, and . if present, the authentication fails. The video transmitter verifies the integrity of the current revocation list by checking the signature of the system renewability message (SRM) using the Digital Content Protection LLC public key L[1]. Failure of this integrity check constitutes an authentication failure.

From	To	Max Delay	Conditions and Comments
Upstream *Aksv* received	*Aksv* transmitted downstream	100 ms	Downstream propagation time. To latest *Aksv* transmission when more than one receiver is attached.
Aksv transmitted to all downstream ports	Upstream READY asserted	500 ms	Upstream propagation time when no downstream video repeaters are attached. (no downstream KSV lists to process)
Downstream READY asserted	Upstream READY asserted	500 ms	Upstream propagation time when one or more video repeaters are attached. From latest downstream READY. (downstream KSV lists must be processed)
Host transmits *Aksv*	Host polls asserted READY	4.2 seconds	For the Maximum of seven repeater levels, 7 * (100 ms + 500 ms)

Table 2–1. Video Repeater Protocol Timing Requirements

Table 2–1 specifies video repeater timing requirements that bound the worst-case propagation time for the KSV list. Note that because each video repeater does not know the number of downstream video repeaters, it must use the same five-second timeout used by the host when polling for downstream READY.

The video transmitter enables data encryption when the second part of the authentication protocol successfully completes.

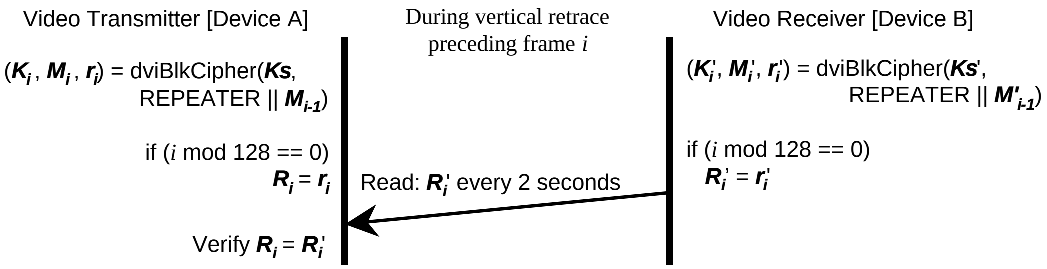
Video Transmitter [Device A]
During vertical retrace preceding frame i
Video Receiver [Device B]
(K_i, M_i, r_i) = dviBlkCipher(Ks, REPEATER || M_{i-1})
(K_i', M_i', r_i') = dviBlkCipher(Ks', REPEATER || M'_{i-1})
if (i mod 128 == 0)
$R_i = r_i$
if (i mod 128 == 0)
$R_i' = r_i'$
Read: R_i' every 2 seconds
Verify $R_i = R_i'$

Figure 2–3. Third Part of Authentication Protocol

The third part of the authentication protocol, illustrated in Figure 2–3, occurs during the vertical blanking interval preceding the frame for which it applies. Each of the two devices calculates new cipher initialization values, K_i and M_i, and a third value R_i, where i is the frame number starting with one for the first video frame for which content protection is enabled. K_i is a 56-bit key used to initialize the HDCP cipher for encryption or decryption of the RGB information for the video frame. M_i is a new 64-bit initialization value for the HDCP cipher. R_i is a 16-bit value used for link integrity verification, and is updated for every 128th frame, starting with the 128th frame. The video transmitter verifies this value against its own calculations to insure that the video receiver is still able to correctly decrypt the information. This verification is made at the rate of once every two seconds, plus or minus one-half second. It is required that the R_i' read operation complete within 250 milliseconds from the time that it is initiated by the video transmitter. Failure for any reason causes the video transmitter to consider the DVI link to be unauthenticated.

2.3 Video Transmitter State Diagram

The transmitter device state diagram (Figure 2–4) illustrates the operation states of the authentication protocol for a video transmitter.

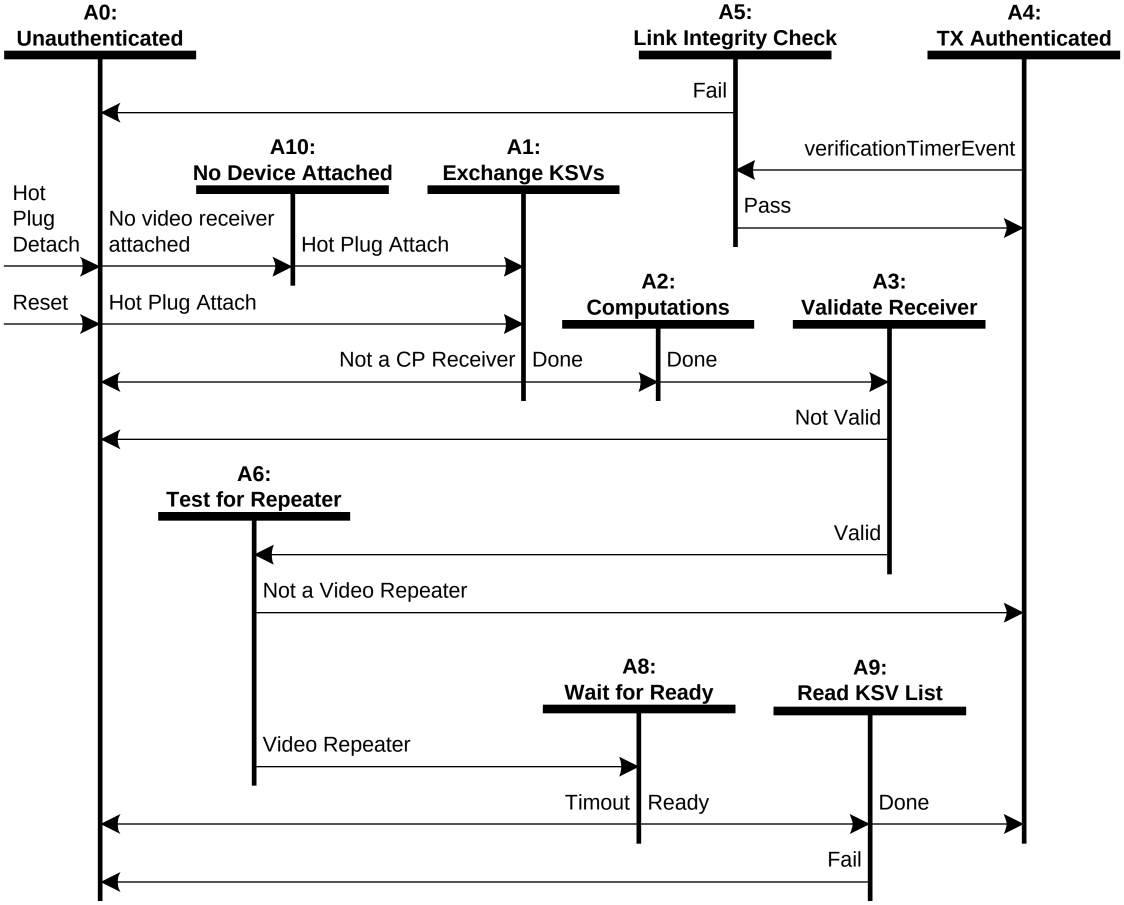

Figure 2–4. Video Transmitter Authentication Protocol State Diagram

Transition Any State:A0. Reset conditions at the video transmitter cause the video transmitter to enter the unauthenticated state. Video receiver detach as sensed by the hot plug pin of the DVI interface also cause a transition to the unauthenticated state.

State A0: Unauthenticated. In this state the device is idle, with encryption disabled, awaiting an event to trigger the authentication protocol. Such events include completion of certain phases of the operating system startup and the hot-plug detection of an attached video receiver.

Transition A0:A1. A trigger event, such as hot-plug detection of an attached video receiver, initiates the authentication protocol.

Transition A0:A10. This transition is made when the hot plug pin of the DVI interface indicates that no device is attached.

State A1: Exchange KSVs. In this state, the video transmitter generates a 64-bit pseudo-random value (*An*) in hardware and writes that value and its key selection vector (*Aksv*) to the video receiver. The video transmitter also reads the video receiver key selection vector (*Bksv*) and the REPEATER status bit necessary for cipher initialization. Hardware generation of *An* using the HDCP Cipher is described in section 4.5.

Transition A1:A0. Failure to read a key selection vector containing 20 zeros and 20 ones is considered a protocol failure and causes this state transition to State A0.

Transition A1:A2. The random value *An* and video transmitter KSV have been written, and a valid video receiver *Bksv* and REPEATER bit have been read. Video transmitter hardware is required to check that *Bksv* contains 20 ones and 20 zeros.

State A2: Computations. In this state, the video transmitter computes the values *Km*, *Ks*, $M_{0,}$ and R_0, using the video transmitter private keys, *Bksv* read during State A1, and the random number *An* written to the video receiver during state A1.

Transition A2:A3. When the computed results from State A2 are available, the video transmitter proceeds to State A3.

State A3: Validate Receiver. The video transmitter reads R_0' from the video receiver and compares it with the corresponding R_0 produced by the video transmitter during the computations of State A2. If R_0 is equal to R_0', then data encryption is immediately enabled. The video transmitter must allow the video receiver up to 100 ms to make R_0' available from the time that *Aksv* is written. The video transmitter also checks the current revocation list for the video receiver's KSV *Bksv*. If *Bksv* is in the revocation list, then the video receiver is considered to have failed the authentication and is not allowed to receive protected content. Note: checking the revocation list for *Bksv* may begin as soon as the *Bksv* has been read in State A1, asynchronously to the other portions of the protocol, but it must complete prior to the transition into the authenticated state (State A4).

The integrity of the current revocation list must be verified by checking the signature of the SRM using the Digital Content Protection LLC public key L^1, as specified in Section 5.

Transition A3:A0. The link integrity message R_0 received from the video receiver does not match the value calculated by the video transmitter, or *Bksv* is in the current revocation list.

Transition A3:A6. The link integrity message R_0 received from the video receiver matches the expected value calculated by the video transmitter and *Bksv* is not in the current revocation list.

State A4: Authenticated. The device has completed the authentication protocol. The verification timer is set up to generate timer events at the nominal rate of once every two seconds, plus or minus one-half second. At this time, and at no time prior, the HDCP system may indicate to upstream content protection technologies (eg. conditional access technologies for direct broadcast satellite) that the system is fully engaged and able to deliver protected content.

Transition A4:A5. A verification timer event causes this transition to State A5.

State A5: Link Integrity Check. In this state, the video transmitter reads R_i' from the video receiver and compares that value against its value R_i. If the values are equal, then the video receiver is correctly decrypting the transmitted stream. The R_i' value may be re-read to allow for synchronization and I^2C bus errors.

Transition A5:A4. The link integrity message from the video receiver correctly matches the expected value.

Transition A5:A0. The link integrity message from the video receiver does not match the expected value, or the value was not returned to the video transmitter within 250 milliseconds from the initiation of the read operation.

State A6: Test for Repeater. The video transmitter evaluates the state of the video repeater capability bit (REPEATER) that was read in State A1.

Transition A6:A4. The REPEATER bit is not set (the video receiver is not a repeater).

Transition A6:A8. The REPEATER bit is set (the video receiver is a repeater).

State A8: Wait for Ready. The video transmitter polls the video receiver READY bit.

Transition A8:A0. The watchdog timer expires before the READY indication is received.

Transition A8:A9. The asserted READY signal is received.

State A9: Read KSV List. The watchdog timer is cleared. The video transmitter reads the list of attached KSVs from the KSV FIFO, reads V, computes V' and verifies $V == V'$, and the KSVs from the list are compared against the current revocation list.

The integrity of the current revocation list must be verified by checking the signature of the SRM using the Digital Content Protection LLC public key L^1, as specified in Section 5.

Transition A9:A0. This transition is made if $V \neq V'$, verification of the SRM fails, or if the any of the KSVs in the list are found in the current revocation list. A retry of the entire KSV FIFO read operation may be implemented in the case of an incorrect V value. Two additional status bits cause this transition when asserted. These are MAX_CASCADE_EXCEEDED and MAX_DEVS_EXCEEDED.

Transition A9:A4. If $V == V'$, the SRM is valid, none of the reported KSVs are in the current revocation list, and the downstream topology does not exceed specified maximums.

State A10: No Device Attached. The hot plug pin of the DVI interface indicates that there is no DVI device attached. Video data is not transmitted.

Transition A10:A1. The authentication protocol begins when the hot plug pin of the DVI interface indicates attachment of a device.

2.4 Video Receiver State Diagram

The operation states of the authentication protocol for a video receiver are illustrated in Figure 2–5.

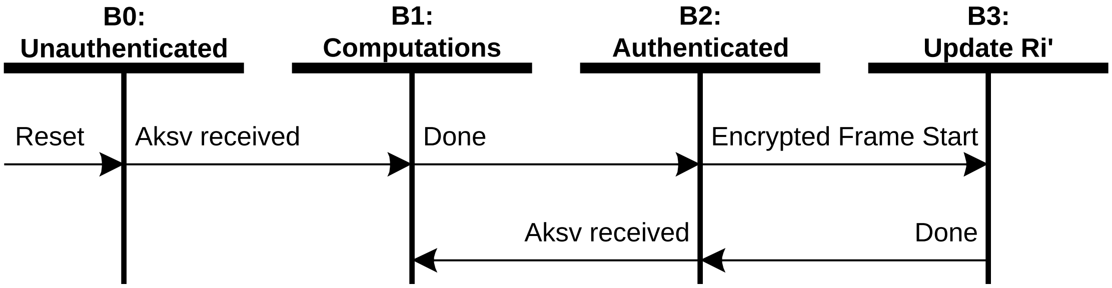

Figure 2–5. Video Receiver Authentication State Diagram

Transition Any State:B0. Reset conditions at the video receiver cause the video receiver to enter the unauthenticated state.

State B0: Unauthenticated. The device is idle, awaiting the reception of *An* and *Aksv* from the video transmitter to trigger the authentication protocol.

Transition B0:B1. The final byte of *Aksv* is received from a video transmitter.

State B1: Computations. In this state, the video receiver calculates the values *Km*', *Ks*', M_0', and R_0' using the video receiver private keys and the received values of *An* and *Aksv*. The video receiver is allowed a maximum time of 100 milliseconds to complete the computations and make R_0' available to the video transmitter. Should the video transmitter write the *Aksv* while the video receiver is in this state (State B1), the video receiver abandons intermediate results and restarts the computations.

Transition B1:B2. The computations are complete and the results are available for reading by the video transmitter.

State B2: Authenticated. The video receiver has completed the authentication protocol and is ready to generate the first video frame key when signaled by the video transmitter.

Transition B2:B1. Re-authentication is forced any time the *Aksv* is written by the attached video transmitter.

Transition B2:B3. The third part of the authentication protocol requires periodic updates to the *Ri*' value.

State B3: Update Ri'. During the vertical blank interval preceding each encrypted frame the video receiver determines whether or not to update the response value *Ri*' with HDCP Cipher output value available during the frame key calculation. The *Ri*' value is updated when (*i* mod 128 == 0). The updated value must be available through the HDCP Port no more than 128 pixel clocks from the time that CTL3 signals data encryption for the next frame. Section 2.7 specifies CTL3 signaling

Transition B3:B2. Once R_i' has been updated, return to the authenticated state.

2.5 Video Repeater State Diagrams

The video repeater has one connection to an upstream video transmitter and one or more connections to downstream video receivers connected via DVI and HDCP as permitted in the Digital Content Protection LLC license. The state diagram for each downstream connection (Figure 2–6) is substantially the same as that for the host video transmitter (Section 2.3), with two exceptions. First, the repeater is not required to check for downstream KSVs in a revocation list. Second, the video repeater initiates authentication downstream only when it receives an authentication request from upstream, rather than at hot plug detection on the downstream port. The video repeater signals the hot plug event to the upstream host by pulsing the HPG signal of the upstream DVI interface. The pulse width must be greater than 100 ms.

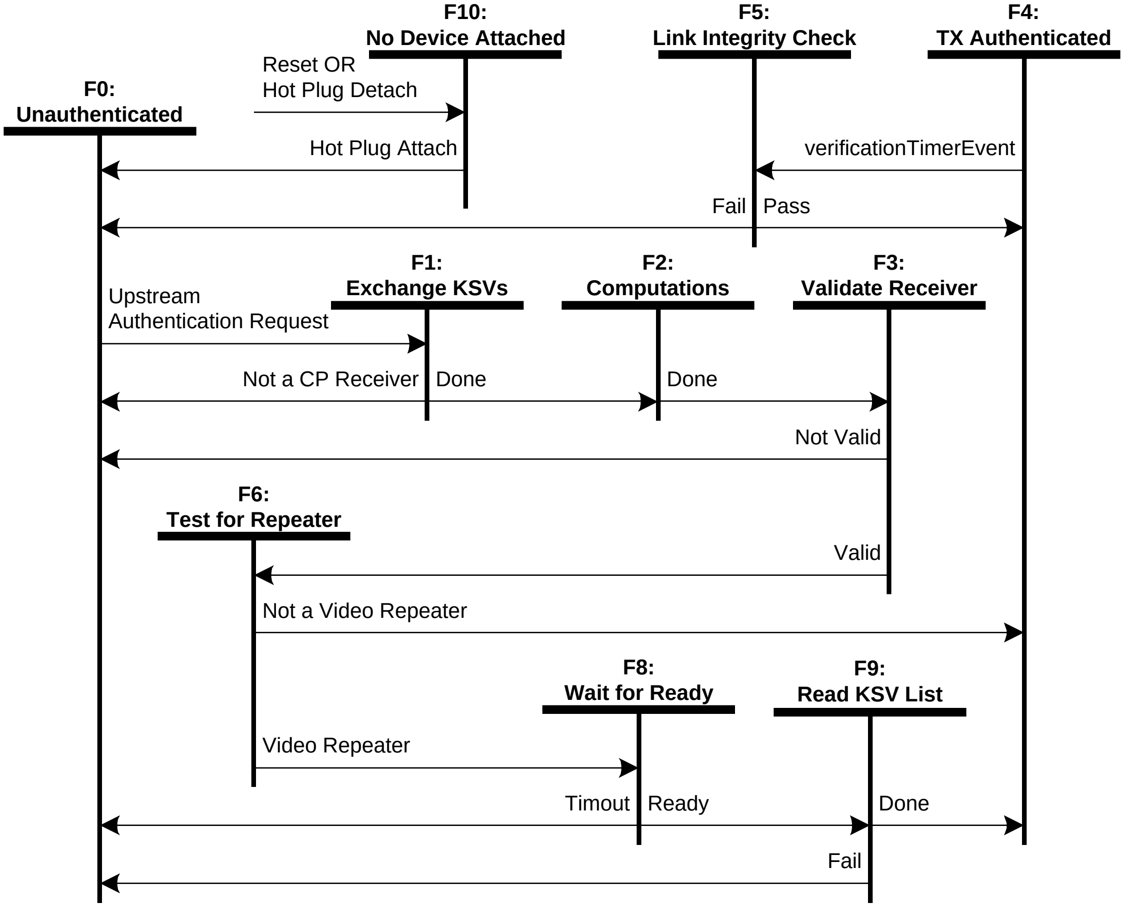

Figure 2–6. Video Repeater Downstream Authentication Protocol State Diagram

Transition Any State:F10. Reset conditions at the video repeater and downstream hot plug detach cause the video repeater port to enter state F10, no device attached.

State F0: Unauthenticated. In this state the device is idle, with encryption disabled, awaiting an upstream authentication request (upstream Aksv write) to trigger the authentication protocol.

Transition F0:F1. The upstream authentication request initiates the authentication protocol.

State F1: Exchange KSVs. In this state, the downstream transmitter of the video repeater generates a 64-bit pseudo-random value (*An*) in hardware and writes that value and its key selection vector (*Aksv*) to the video receiver. The video repeater also reads the video receiver key selection vector (*Bksv*) and the repeater capability bit (REPEATER) necessary for cipher initialization. Hardware generation of *An* using the HDCP Cipher is described in section 4.5.

Transition F1:F0. Failure to read a key selection vector containing 20 zeros and 20 ones is considered a protocol failure and causes this state transition to State F0.

Transition F1:F2. The random value *An* and downstream transmitter KSV have been written, and a valid video receiver *Bksv* and REPEATER bit have been read. Downstream transmitter hardware is required to validate that *Bksv* contains 20 ones and 20 zeros.

State F2: Computations. In this state, the downstream transmitter computes the values *Km*, *Ks*, $M_{0,}$ and R_0, using the downstream transmitter private keys, *Bksv* read during State F1, and the random number *An* written to the video receiver during state F1.

Transition F2:F3. When the computed results from State F2 are available, the downstream transmitter proceeds to State F3.

State F3: Validate Receiver. The downstream transmitter reads R_0' from the video receiver and compares it with the corresponding R_0 produced by the downstream transmitter during the computations of State F2, then immediately enables data encryption if R_0' is equal to R_0. The video receiver must be allowed up to 100 ms to make R_0' available from the time that *Aksv* is written. The downstream *Bksv* is added to the KSV list for this video repeater.

Transition F3:F0. The link integrity message R_0' received from the video receiver does not match the value calculated by the downstream transmitter.

Transition F3:F6. The link integrity message R_0' received from the video receiver matches the expected value calculated by the downstream transmitter.

State F4: Authenticated. At this time, and at no prior time, the device has completed the authentication protocol and is fully operational, able to deliver protected content. The verification timer is set up to generate timer events at the nominal rate of once every two seconds, plus or minus one-half second.

Transition F4:F5. A verification timer event causes this transition to State F5.

State F5: Link Integrity Check. In this state, the downstream transmitter reads R_i' from the video receiver and compares that value against its value R_i. If the values are equal, then the video receiver is correctly decrypting the transmitted stream. The R_i' value may be re-read to allow for synchronization and I^2C bus errors.

Transition F5:F4. The link integrity message from the video receiver correctly matches the expected value.

Transition F5:F0. The link integrity message from the video receiver does not match the expected value, or the value was not returned to the downstream transmitter within 250 milliseconds from the initiation of the read operation.

State F6: Test for Repeater. The downstream transmitter evaluates the state of the video repeater capability bit (REPEATER) that was read in State F1.

Transition F6:F4. The REPEATER bit is not set (the video receiver is not a repeater).

Transition F6:F8. The REPEATER bit is set (the video receiver is a repeater).

State F8: Wait for Ready. The downstream transmitter sets up a five-second watchdog timer and polls the video receiver READY bit.

Transition F8:F0. The watchdog timer expires before the READY indication is received.

Transition F8:F9. The asserted READY signal is received.

State F9: Read KSV List. The watchdog timer is cleared. The downstream transmitter reads the list of attached KSVs through the KSV FIFO, reads *V*, computes *V'* and verifies *V* == *V'*, and the KSVs from this port are added to the KSV list for this video repeater. Two additional status bits (MAX_CASCADE_EXCEEDED and MAX_DEVS_EXCEEDED) from the downstream video receiver are read and if asserted, cause the repeater to also assert them upstream.

Transition F9:F0. This transition is made if *V* != *V'*. A retry of the entire KSV FIFO read operation may be implemented in the case of an incorrect *V* value. It is also made if either MAX_CASCADE_EXCEEDED or MAX_DEVS_EXCEEDED are asserted.

Transition F9:F4. This transition is made if *V* == *V'* and the downstream topology does not exceed specified maximums.

State F10: No Device Attached. The hot plug pin of the DVI interface indicates that there is no DVI device attached. Video data is not transmitted.

Transition F10:F0. The downstream port transitions to the unauthenticated state when the hot plug pin of the DVI interface indicates attachment of a device.

The video repeater upstream state diagram, illustrated in Figure 2–7, makes reference to states of the video repeater downstream state diagram.

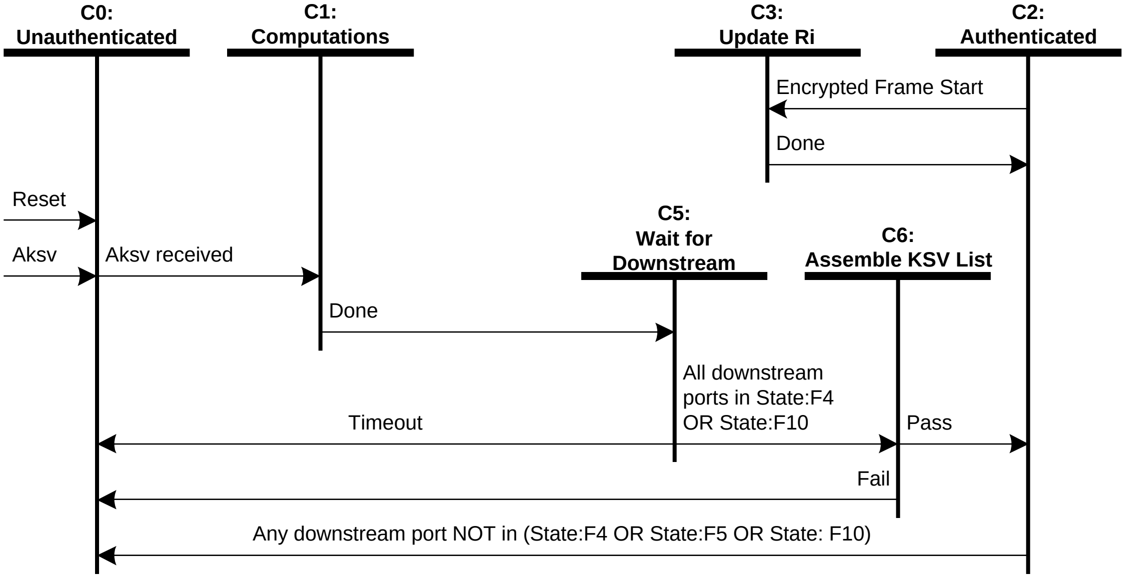

Figure 2–7. Video Repeater Upstream Authentication Protocol State Diagram

Transitions Any State:C0. Reset conditions at the video repeater cause the video repeater to enter the unauthenticated state. Re-authentication is forced any time the *Aksv* is written by the attached video transmitter, with a transition through the unauthenticated state.

State C0: Unauthenticated. The device is idle, awaiting the reception of *An* and *Aksv* from the video transmitter to trigger the authentication protocol. The READY status bit, in the HDCP port, is de-asserted.

Transition C0:C1. The final byte of *Aksv* is received from a video transmitter.

State C1: Computations. In this state, the video repeater calculates the values *Km*', *Ks*', M_0', and R_0' using its private keys and the received values of *An* and *Aksv*. The video repeater is allowed a maximum time of 100 milliseconds to complete the computations and make R_0' available to the video transmitter. Should the video transmitter write the *Aksv* while the video repeater is in this state (State C1), the video repeater abandons intermediate results and restarts the computations.

Transition C1:C5. The computations are complete and the results are available for reading by the video transmitter.

State C2: Authenticated. The video repeater has completed the authentication protocol and is ready to generate the first video frame key when signaled by the video transmitter. The READY status bit is asserted.

Transition C2:C0. The upstream connection becomes unauthenticated if any downstream video receiver enters the unauthenticated state OR if a downstream port that previously had no downstream device attached senses an attachment via the hot plug detection pin.

Transition C2:C3. This transition is made during the vertical blank interval preceding encrypted frames.

State C3: Update Ri'. During the vertical blank interval preceding each encrypted frame the video repeater determines whether or not to update the response value R_i' with HDCP Cipher output value available during the frame key calculation. The R_i' value is updated when (*i* mod 128 == 0).

Transition C3:C2. Once R_i' has been updated, return to the authenticated state.

State C5:Wait for Downstream. The upstream state machine waits for all downstream ports of the video repeater to enter either the unconnected (State F10) or the authenticated state (State F4).

Transition C5:C0. The watchdog timer expires before all downstream video ports enter the authenticated or unconnected state.

Transition C5:C6. All downstream ports with attached video receivers have reached the state of authenticated or unconnected.

State C6: Assemble KSV List. The video repeater assembles the list of all attached devices as the downstream ports reach terminal states of the authentication protocol. A port that advances to State F10, the unconnected state, does not add to the list. A port that arrives in State F4 that has a video receiver attached (as opposed to a video repeater), adds the *Bksv* of the attached video receiver to the list. Ports that arrive in State F4 that have a video repeater attached will cause the KSV list of the attached video repeater, plus the *Bksv* of the attached video repeater, to be added to the list. The video repeater must verify the integrity of the downstream list by computing *V* and checking this value against *V*' received from the attached video repeater. If *V* does not equal *V*', the downstream KSV list integrity check fails. When the KSV list for all downstream video receivers has been assembled, the video repeater computes the upstream *V*.

The DEVICE_COUNT for a video repeater is equal to the total number of attached video receivers. The value is calculated as the sum of the number of attached downstream ports plus the sum of the DEVICE_COUNT of all attached video repeaters. The DEPTH for a video repeater is equal to the maximum number of connection levels below any of the downstream DVI ports. The value is calculated as the maximum DEPTH reported from downstream video repeaters plus one (accounting for the attached downstream video repeater). If the computed DEVICE_COUNT for a repeater exceeds 127, the repeater must assert the MAX_DEVS_EXCEEDED status bit. If the computed DEPTH for a repeater exceeds seven, the repeater must assert the MAX_CASCADE_EXCEEDED status bit. When a repeater receives a MAX_DEVS_EXCEEDED or a MAX_CASCADE_EXCEEDED status from a downstream repeater, it is required to assert its corresponding upstream status bit.

Transition C6:C0. If any downstream port should transition to the unauthenticated state, the upstream connection transitions to the unauthenticated state. This transition is also made when any downstream video repeater reports a topology error, or when the KSV list integrity check for a downstream video repeater fails.

Transition C6:C2. The KSV list and *V* are ready for reading by the upstream video transmitter.

2.6 HDCP Port

The values that must be exchanged between the video transmitter and the video receiver are communicated over the I^2C serial interface of the DVI interface. The video receiver or video repeater must present a logical device on the I^2C bus for each T.M.D.S. link that it supports. No equivalent interface to video transmitters is specified. The seven-bit I^2C device address for the primary link is 0111010x binary, or 0x74 in the usual hexadecimal representation of I^2C device addresses where the read/write bit is set to zero. The device address for the secondary link is 0x76. Table 2–2 and Table 2–3 specify the address space for these devices, which act only as slaves on the I^2C bus. Multi-byte values are stored in little-endian format.

Read and write operations must complete within 100 ms per byte transferred. It is strongly recommended that slave devices never stretch the I^2C clock. Master devices may elect to repeat any transfers believed to have previously completed with errors.

Offset (hex)	**Name**	**Size in Bytes**	**Rd/ Wr**	**Function**
0x00	*Bksv*	5	Rd	Video receiver KSV. This value must always be available for reading, and may be used to determine that the video receiver is HDCP capable. Valid KSVs contain 20 ones and 20 zeros, a characteristic that must be verified by video transmitter hardware before encryption is enabled.
0x05	Rsvd	3	Rd	All bytes read as 0x00
0x08	*Ri'*	2	Rd	Link verification response. Updated every 128th frame. It is recommended that graphics systems protect against errors in the I^{2}C transmission by re-reading this value when unexpected values are received. This value must be available at all times between updates. R_0' must be available a maximum of 100 ms after *Aksv* is received. Subsequent R_i' values must be available a maximum of 128 pixel clocks following the assertion of CTL3.
0x0A	Rsvd	6	Rd	All bytes read as 0x00
0x10	*Aksv*	5	Wr	Video transmitter KSV. Writes to this multi-byte value are written least significant byte first. The final write to 0x14 triggers the authentication sequence in the display device.
0x15	Rsvd	3	Rd	All bytes read as 0x00
0x18	*An*	8	Wr	Session random number. This multi-byte value must be written by the graphics system before the KSV is written.
0x20	*V*	20	Rd	Twenty-byte SHA–1 hash value used in the second part of the authentication protocol for video repeaters.
0x34	Rsvd	12	Rd	All bytes read as 0x00
0x40	*Bcaps*	1	Rd	Bit 7: Reserved. Read as zero. Bit 6: REPEATER, Video repeater capability. When set to one, this device supports downstream DVI connections as permitted by the Digital Content Protection LLC license. Bit 5: READY, KSV FIFO ready. When set to one, the device has built the list of attached KSVs and appended the verification value *V*. This value is always zero during the computation of *V*. Bit 4: FAST. When set to one, this device supports 400 KHz transfers. When zero, 100 KHz is the maximum transfer rate supported.
0x41	*Bstatus*	2	Rd	Refer to Table 2–4 for definitions.
0x43	KSV FIFO	1	Rd	Key selection vector FIFO. Used to pull KSVs from devices with downstream DVI outputs. Must be read in a single, auto-incrementing access. All bytes read as 0x00 for video receivers (REPEATER == 0).
0x44	Rsvd	176	Rd	All bytes read as 0x00
0xFF	dbg	1	Rd/ Wr	Implementation-specific debug register. Confidential values must not be exposed through this register.

Table 2–2. Primary Link HDCP Port (I^{2}C device address 0x74)

Offset (hex)	Name	Size (Bytes)	Rd/Wr	Function
0x00	*Bksv*	5	Rd	Video receiver KSV. See primary link comments. This value must match the value of Bksv for the primary link.
0x05	Rsvd	3	Rd	All bytes read as 0x00
0x08	*Ri'*	2	Rd	Link verification response. See primary link comments. This value will differ from the value of *Ri*' for the primary link.
0x0A	Rsvd	6	Rd	All bytes read as 0x00
0x10	*Aksv*	5	Wr	Video transmitter KSV. See primary link comments. This value must be programmed to the same value of *Aksv* for the primary link.
0x15	Rsvd	3	Rd	All bytes read as 0x00
0x18	*An*	8	Wr	Session random number. See primary link comments. This value must **differ** from the programmed value of *An* for the primary link.
0x20	Rsvd	36	Rd	All bytes read as 0x00
0x43	dbg	1	Rd/Wr	Implementation-specific debug register. Confidential values must not be exposed through this register.
0x44	Rsvd	187	Rd	All bytes read as 0x00

Table 2–3. Secondary Link HDCP Port (I^{2}C device address 0x76)

Name	Bit Field	Rd/ Wr	Description
Rsvd	15:12	Rd	Reserved. Read as zero.
MAX_CASCADE_EXCEEDED.	11	Rd	Topology error indicator. When set to one, more than seven levels of video repeater have been cascaded together.
DEPTH	10:8	Rd	Three-bit repeater cascade depth. This value gives the number of attached levels through the connection topology.
MAX_DEVS_EXCEEDED	7	Rd	Topology error indicator. When set to one, more than 127 downstream devices are attached.
DEVICE_COUNT	6:0	Rd	Total number of attached devices. Always zero for video receivers. Video repeater count does not include the repeater

Table 2–4. *Bstatus* Register Bit Field Definitions

The CP devices at these slave addresses respond to I^{2}C accesses as diagrammed in Figures 2–7, 2–8, and 2–9. The nomenclature within these diagrams, and used to describe them, is the same as found in *The I^{2}C Bus Specification Version 2.0*.

Figure 2–8 illustrates a combined-format byte read, in which the master writes a one-byte address to the slave, followed by a repeated start condition (Sr) and the data read. With the exception of combined-format reads from the KSV FIFO, HDCP port devices must support multi-byte reads, with auto-increment. Combined-format reads from the KSV FIFO have an implicit address increment though the FIFO data structures.

S	Slave Addr (7)	W	A	Offset Addr (8)	A	Sr	Slave Addr (7)	R	A	Read Data (8)	A	P

Figure 2–8. HDCP Port Combined-Format Byte Read

Figure 2–9 illustrates a byte write access. As for combined-format read accesses, the HDCP port must support multi-byte writes with auto-increment, again with an exception for KSV FIFO writes where the implicit address increment moves through the KSV FIFO data structure rather than through the HDCP port address space. Auto-incremented sequential accesses that start before the KSV FIFO address and cross through the KSV FIFO address read only the first byte of the KSV FIFO and then continue incrementing through the HDCP port address space.

S	Slave Addr (7)	W	A	Offset Addr (8)	A	Write Data (8)	A	P

Figure 2–9. HDCP Port Byte Write

In order to minimize the number of bits that must be transferred for the link integrity check, a second read format must be supported by all video receivers and by video transmitters that do not implement a hardware I^2C master. This access, shown in Figure 2–10, has an implicit address equal to 0x08, the starting location for R_i'. The short read format may be uniquely differentiated from combined reads by tracking STOP conditions (P) on the bus. Short reads must be supported with auto-incrementing addresses.

S	Slave Addr (7)	R	A	Read Data (8)	A	P

Figure 2–10. HDCP Port Link Integrity Message Read

2.7 DVI Control Signal Protocol

The transmitter signals the receiver to begin the second part of the authentication protocol through the previously reserved control signal CTL3 in the DVI interface. This is done with a single high-going pulse, during the vertical blanking interval, of sufficient width that it may be distinguished from bit errors on the channel or any effects due to resynchronization events in the receiver.

The timing requirements for CTL3 are specified in Table 2–5. Note that for typical display timings with positive polarity vertical sync, it is possible to satisfy these requirements by tying the CTL3 signal to the vertical sync signal when content protection is enabled.

Parameter	Time (Pixel Clocks)
Minimum Pulse Width	8
Minimum time from first assertion of CTL3 to end of vertical blank interval	128

Table 2–5. DVI Control Signal Timing Requirements

3 Data Encryption

Data encryption is applied at the input to the T.M.D.S. transmitter and decryption is applied at the output of the T.M.D.S. receiver (Figure 3–1). Data encryption consists of a bit-wise exclusive-or (XOR) of the video data with a pseudo-random data stream produced by the HDCP Cipher. In dual-link implementations the video data is 48-bits wide and requires two HDCP Ciphers to produce the required pseudo-random streams.

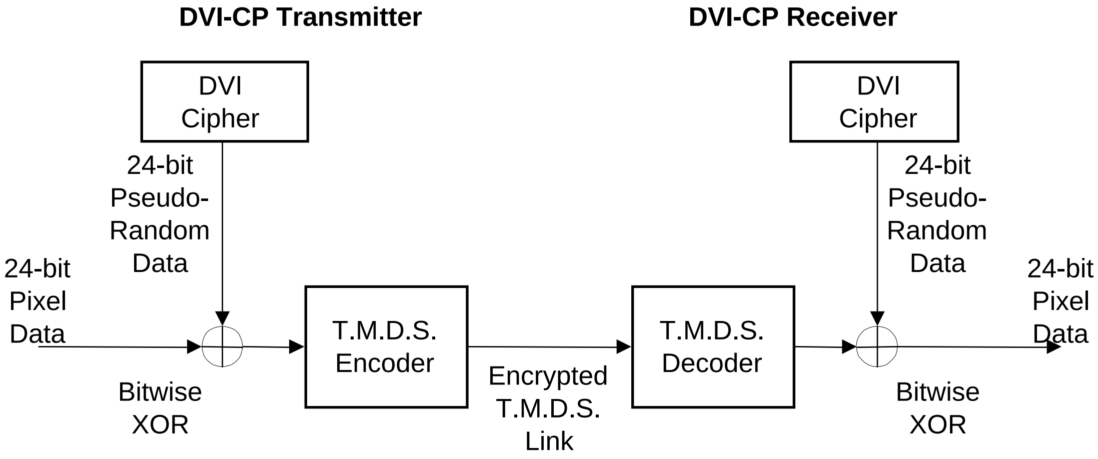

Figure 3–1. HDCP Encryption and Decryption

During the vertical-blanking interval, the hdcpBlockCipher function prepares the HDCP Cipher to produce the 24-bit wide key-dependent pseudo-random stream during active pixel data. The HDCP Cipher generates a new 24-bit word of pseudo-random data for every active pixel of video data, as defined on the interface by the data enable (DE) signal. The 24-bits of cipher output are applied to the RGB video data as shown in Table 3–1.

Cipher Output	Video Stream Bits
23:16	Red[7:0]
15:8	Green[7:0]
7:0	Blue[7:0]

Table 3–1. Encryption Stream Mapping

During horizontal-blanking intervals on the interface, the HDCP Cipher is re-keyed for 56 pixel clocks as described in Section 4.5. This complicates the task of breaking the encryption from line to line.

Figure 3–2 illustrates the encryption functions as they relate to horizontal sync (HSync), vertical sync (VSync), data enable (DE), and Control3. Because this diagram is applicable to both transmitters and receivers, the state and transition descriptions below use the term "transceiver" to refer to either transmitters or receivers. Encrypt/Decrypt refers to the appropriate operation for the transceiver.

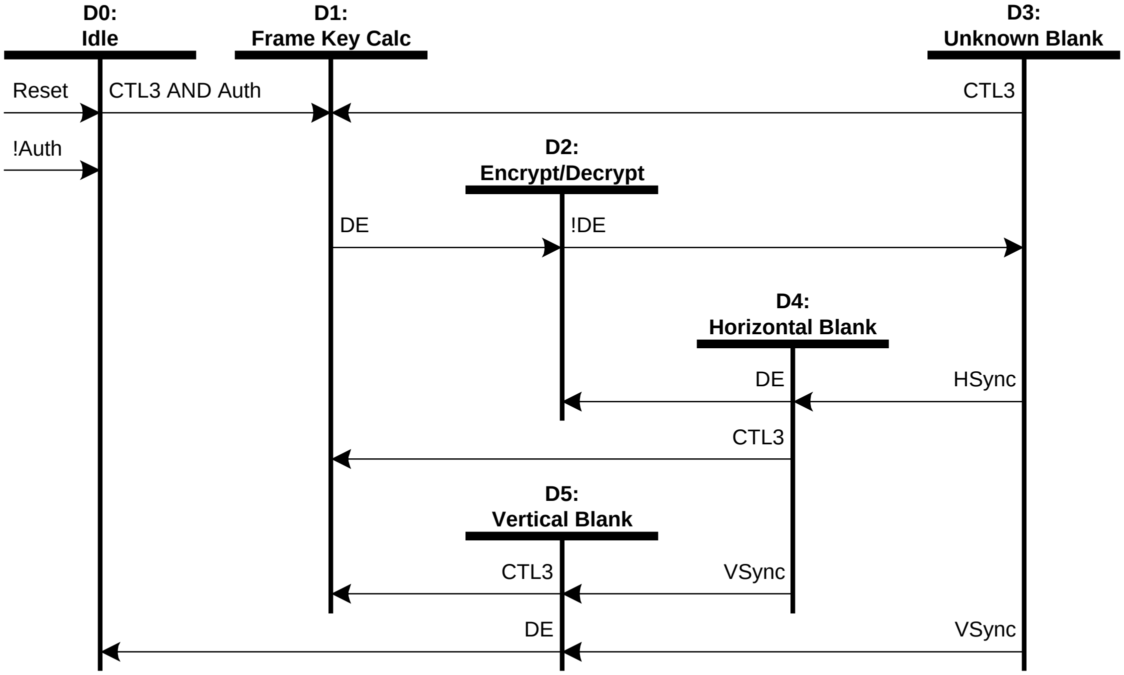

Figure 3–2. Encryption/Decryption State Diagram

Transition Any State:D0. Reset conditions or transitions into the unauthenticated state at the video transceiver cause the encryption state machine to transition to the idle state.

State D0: Idle. The HDCP Cipher is free running and available for use as hdcpRngCipher (Section 4.5).

Transition D0:D1. The assertion of CTL3 (as specified in Table 2–4) at a time when the video transceiver is authenticated causes frame key calculation. Authenticated states for video transmitters are State A4 and State A5. Authenticated states for video receivers are State B2 and State B3. Authenticated states for video repeaters are State C2 and State C3.

State D1: Frame Key Calculation. The frame key for the next video frame is calculated as described in section 4.5, using hdcpBlockCipher.

Transition D1:D2. Data enable (DE) on the T.M.D.S. link signals the beginning of active pixel data to be encrypted/decrypted by the transceiver.

State D2: Encrypt/Decrypt. Video transmitters encrypt pixel data in this state, while video receiver decrypt data. Both use the hdcpStreamCipher as described in section 4.5.

Transition D2:D3. The end of pixel data is signaled by DE.

State D3: Unknown Blank. At the end of active pixel data, it is not assumed that video transceivers are able to distinguish between horizontal and vertical sync. In this state, video transceivers must begin to rekey the HDCP cipher using hdcpRekeyCipher as described in section 4.5.

Transition D3:D1 Frame Key Calculation. The assertion of CTL3 as specified in Table 2–4 results in the generation of a new frame key.

Transition D3:D4. The assertion of HSync identifies the horizontal blank.

Transition D3:D5. The assertion of VSync identifies the vertical blank.

State D4: Horizontal Blank. The rekey operation continues if not completed during State D3.

Transition D4:D2. The assertion of DE begins encryption/decryption of the next line of video data.

Transition D4:D1. The assertion of CTL3 as specified in Table 2–4 results in the generation of a new frame key.

Transition D4:D5. The assertion of VSync identifies the vertical blank.

State D5: Vertical Blank. This state waits for one of the exit conditions.

Transition D5:D1. The assertion of CTL3 as specified in Table 2–4 results in the generation of new frame key.

Transition D5:D0. The return to active pixel data signaled by the assertion of DE prior without a CTL3 assertion during the vertical blank indicates that encryption has been disabled for the next frame. This path might be taken in the event of link integrity message failure.

4 HDCP Cipher

4.1 Overview

The HDCP cipher is a special-purpose cipher designed for both the appropriate robustness of the authentication protocol as well as for the high-speed streaming requirement of uncompressed video data encryption.

The overall structure of the HDCP Cipher can be thought of as three layers. The first layer consists of a set of four Linear Feedback Shift Registers (LFSRs) that are combined to one bit. This one bit feeds into the middle layer when enabled via the rekey enable signal. The middle layer consists of two halves that are very similar in design. One half, *Round Function B*, performs one round of a block cipher using three 28-bit registers, *Bx*, *By*, and *Bz*. The other half, *Round Function K*, is similar in structure to Round Function B, but provides the output of latch *Ky* as a stream of 28-bit round keys to Round Function B at the rate of one 28-bit round key per clock pulse. The final layer takes four 28-bit register outputs from the round functions, *By*, *Bz*, *Ky*, and *Kz*, through a compression function to produce a 24-bit block of pseudo-random bits for every clock pulse.

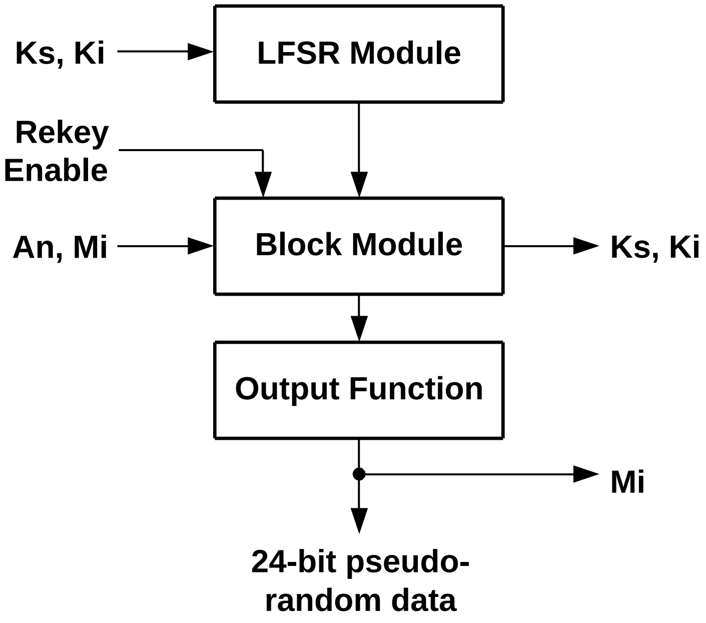

Figure 4–1. HDCP Cipher Structure

The block module operates as a block cipher during the authentication protocol. There is a single sequence, hdcpBlockCipher, which is used for both parts of the authentication protocol. Although decryption in block mode is possible for the HDCP cipher, it is not necessary for this application and thus is not described in this document.

The block module and the output function are used together to produce the 24-bit pseudo random sequence that is used to encrypt the video data. In this mode, hdcpStreamCipher, the module produces 24 bits of output for every input clock.

The LFSR module is used to re-key the block module between lines of video.

4.2 Linear Feedback Shift Register Module

The linear feedback shift register module consists of four LFSRs of different lengths and a combining function that produces a single bit stream from them. The combining function takes three taps from each LFSR. The generator polynomials and combining function taps for the LFSRs are specified in Table 4–1.

		Combining Function Taps		
LFSR	**Polynomial**	**0**	**1**	**2**
3	$x^{17} + x^{15} + x^{11} + x^{5} + 1$	5	11	16
2	$x^{16} + x^{15} + x^{12} + x^{8} + x^{7} + x^{5} + 1$	5	9	15
1	$x^{14} + x^{11} + x^{10} + x^{7} + x^{6} + x^{4} + 1$	4	8	13
0	$x^{13} + x^{11} + x^{9} + x^{5} + 1$	3	7	12

Table 4–1. LFSR Generation and Tapping

Figure 4–2 illustrates the tap locations of LFSR0 as well as the XOR term feedback into the least significant bit of LFSR0.

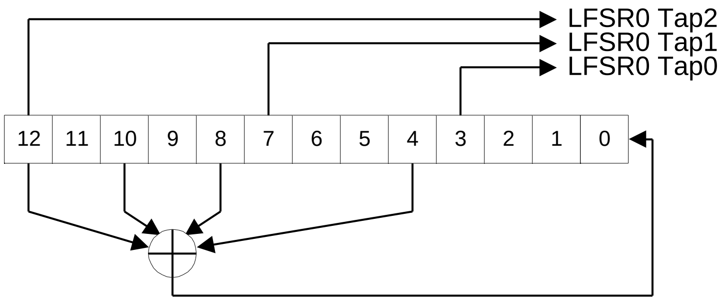

Figure 4–2. LFSR0

The combining function contains four cascaded shuffle networks, each of which includes two state bits. One tap from each of the four LFSRs is XORed together to form the data input to the first shuffle network. One tap from each of the four LFSRs is used as the select input to one of the four shuffle networks. The output of the fourth shuffle network is XORed together with one tap from each of the LFSRs. The Combiner Function illustrated in Figure 4–3.

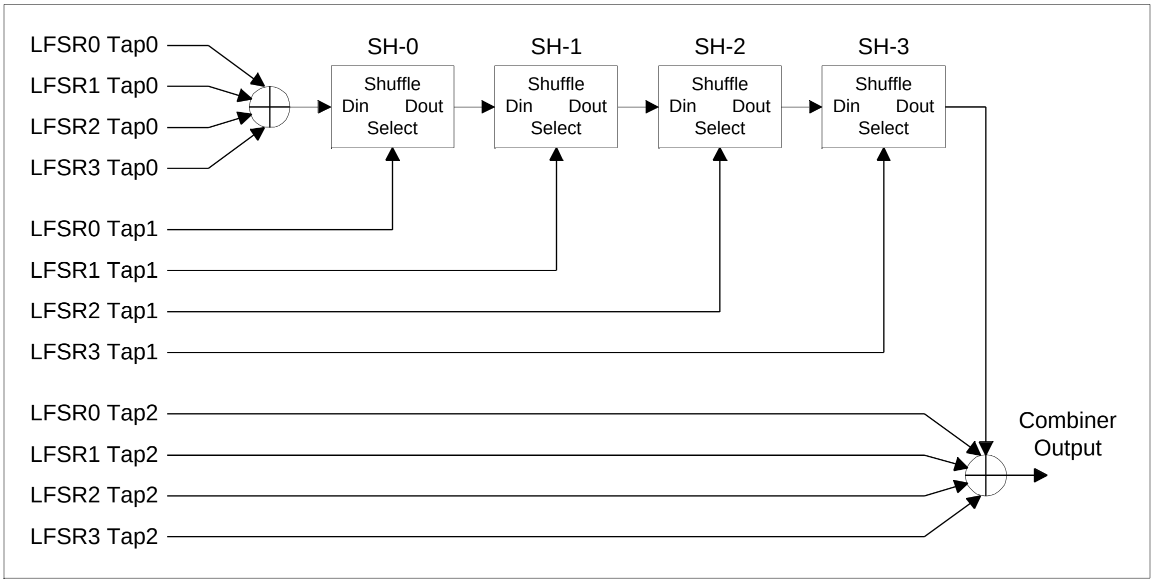

Figure 4–3. LFSR Module Combiner Function

The shuffle network is represented schematically in Figure 4–4. If the shuffle network contains the ordered pair of boolean values (A, B) and has boolean data input D and selection input S, the S value controls the next state. If S is zero, it outputs A and assumes state (B, D). If S is one, it outputs B and assumes state (D, A).

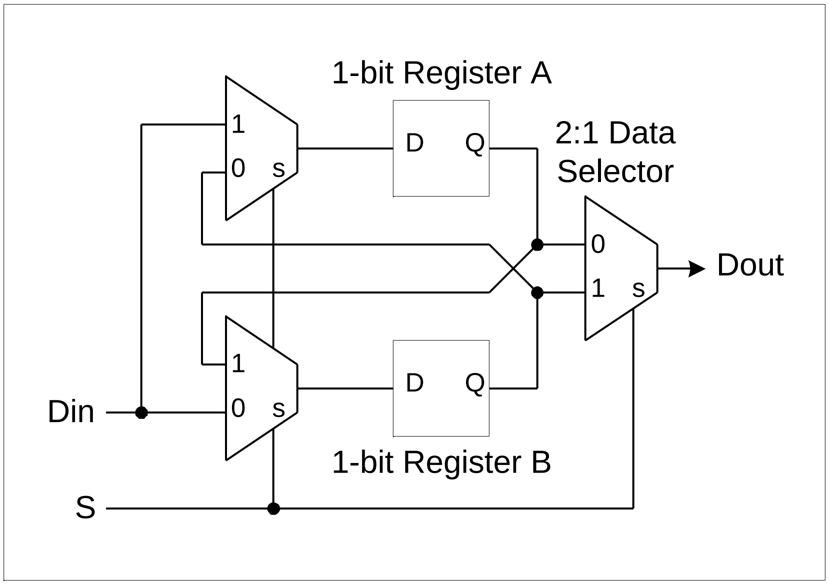

Figure 4–4. Shuffle Network

In all modes of operation the LFSRs and combining function are initialized by a 56-bit value. The 60 bits of LFSR state use these 56 bits directly plus the complements of four of the bits. The shuffle networks are each initialized with the same constant value. The initialization of the LFSR module is specified in Table 4–2 for a 56-bit initialization value.

	Bit Field	Initial Value
LFSR3	[16]	Complement of input bit 47
	[15:0]	Input bits [55:40]
LFSR2	[15]	Complement of input bit 32
	[14:0]	Input bits [39:25]
LFSR1	[13]	Complement of input bit 18
	[12:0]	Input bits [24:12]
LFSR0	[12]	Complement of input bit 6
	[11:0]	Input bits [11:0]
Shuffle Networks	Register A	0
	Register B	1

Table 4–2. LFSR Module Initialization

This one-bit stream output of the combining function is the only output from the LFSR module. This bit stream provides key material to the block module when the rekey enable signal is active.

4.3 Block Module

The block module consists of two separate "round function" components. One of these components, *Round Function K*, provides a key stream for the other component, *Round Function B*. Each of these two components operates on a corresponding set of three 28-bit registers. The structure of the block module is diagrammed in Figure 4–5.

For Round Function K, bit 13 of the Ky register takes its input from the LFSR module output stream when the external rekey enable signal is asserted.

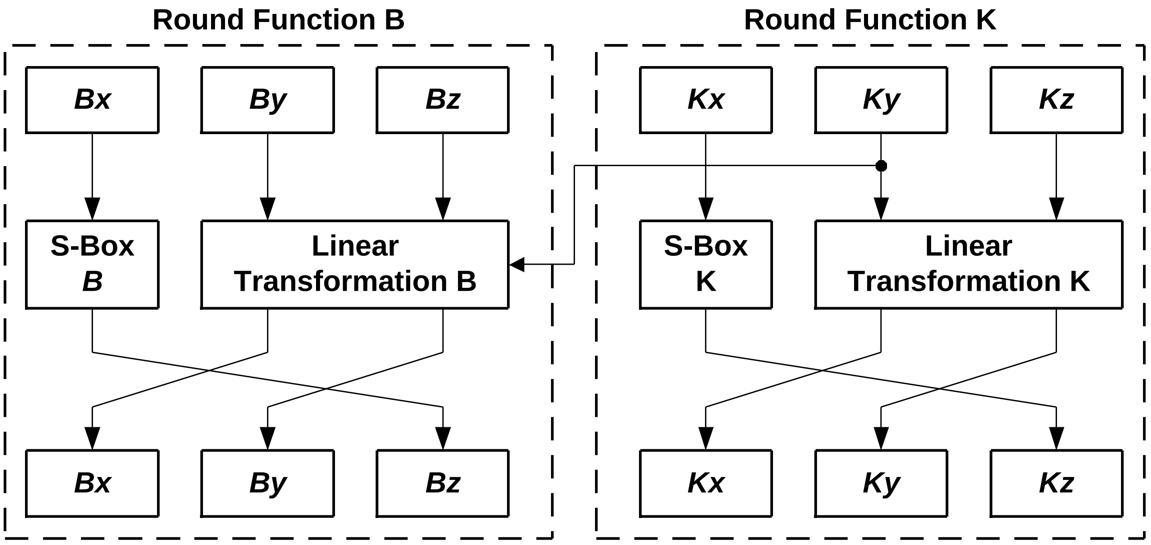

Figure 4–5. Block Module

The S-Boxes for both round functions consist of seven 4 input by 4 output S-boxes. Round function K S-Boxes are labeled SK0 through SK6 and round function B S-Boxes are labeled SB0 through SB6. The Ith input to box J is bit I*7+J from the round x register (*Bx* or *Kx*), and output I of box J goes to bit I*7+J of register z of the round function (*Bz* or *Kz*). Bit 0 is the least significant bit. The S-box permutations of round functions K and B are specified in Table 4–3.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
SK0	8	14	5	9	3	0	12	6	1	11	15	2	4	7	10	13
SK1	1	6	4	15	8	3	11	5	10	0	9	12	7	13	14	2
SK2	13	11	8	6	7	4	2	15	1	12	14	0	10	3	9	5
SK3	0	14	11	7	12	3	2	13	15	4	8	1	9	10	5	6
SK4	12	7	15	8	11	14	1	4	6	10	3	5	0	9	13	2
SK5	1	12	7	2	8	3	4	14	11	5	0	15	13	6	10	9
SK6	10	7	6	1	0	14	3	13	12	9	11	2	15	5	4	8
SB0	12	9	3	0	11	5	13	6	2	4	14	7	8	15	1	10
SB1	3	8	14	1	5	2	11	13	10	4	9	7	6	15	12	0
SB2	7	4	1	10	11	13	14	3	12	15	6	0	2	8	9	5
SB3	6	3	1	4	10	12	15	2	5	14	11	8	9	7	0	13
SB4	3	6	15	12	4	1	9	2	5	8	10	7	11	13	0	14
SB5	11	14	6	8	5	2	12	7	1	4	15	3	10	13	9	0
SB6	1	11	7	4	2	5	12	9	13	6	8	15	14	0	3	10

Table4–3. Block Module S-Box Values

Both linear transformation K and linear transformation B produce 56 output values. These values are the combined outputs from eight diffusion networks that each produces seven outputs. The diffusion network function is specified in Table 4–4. Each diffusion network has seven data inputs labeled I_0 - I_6, seven outputs O_0 – O_6, plus an additional seven optional key inputs K_0 – K_6.

The diffusion networks of round function K are specified in Table 4–5. Note that none of the round function K diffusion networks have the optional key inputs. The diffusion units of round function B are specified in Table 4–6. Half of these diffusion networks have key inputs that are driven from the Ky register of round function K. A dash in the table indicates that the key input is not present.

	Diffusion Network Logic Function							
O_0	K_0		I_1	I_2	I_3	I_4	I_5	I_6
O_1	K_1	I_0		I_2	I_3	I_4	I_5	I_6
O_2	K_2	I_0	I_1		I_3	I_4	I_5	I_6
O_3	K_3	I_0	I_1	I_2		I_4	I_5	I_6
O_4	K_4	I_0	I_1	I_2	I_3		I_5	I_6
O_5	K_5	I_0	I_1	I_2	I_3	I_4		I_6
O_6	K_6	I_0	I_1	I_2	I_3	I_4	I_5	I_6

Table 4–4. Diffusion Network Logic Function

	K1	K2	K3	K4	K5	K6	K7	K8
I_0	Kz0	Kz7	Kz10	Kz13	Kz16	Ky16	Ky20	Ky24
I_1	Kz1	Kz8	Kz11	Kz14	Kz17	Ky17	Ky21	Ky25
I_2	Kz2	Kz9	Kz12	Kz15	Kz18	Ky18	Ky22	Ky26
I_3	Kz3	Ky0	Ky3	Ky6	Ky9	Ky19	Ky23	Ky27
I_4	Kz4	Ky1	Ky4	Ky7	Ky10	Kz19	Kz22	Kz25
I_5	Kz5	Ky2	Ky5	Ky8	Ky11	Kz20	Kz23	Kz26
I_6	Kz6	Ky12	Ky13	Ky14	Ky15	Kz21	Kz24	Kz27
O_0	Kx0	Ky0	Ky1	Ky2	Ky3	Kx1	Kx2	Kx3
O_1	Kx4	Ky4	Ky5	Ky6	Ky7	Kx5	Kx6	Kx7
O_2	Kx8	Ky8	Ky9	Ky10	Ky11	Kx9	Kx10	Kx11
O_3	Kx12	Ky12	Ky13	Ky14	Ky15	Kx13	Kx14	Kx15
O_4	Kx16	Ky16	Ky17	Ky18	Ky19	Kx17	Kx18	Kx19
O_5	Kx20	Ky20	Ky21	Ky22	Ky23	Kx21	Kx22	Kx23
O_6	Kx24	Ky24	Ky25	Ky26	Ky27	Kx25	Kx26	Kx27

Table 4–5. K Round Input and Output Mapping

	B1	B2	B3	B4	B5	B6	B7	B8
I_0	Bz0	Bz7	Bz10	Bz13	Bz16	By16	By20	By24
I_1	Bz1	Bz8	Bz11	Bz14	Bz17	By17	By21	By25
I_2	Bz2	Bz9	Bz12	Bz15	Bz18	By18	By22	By26
I_3	Bz3	By0	By3	By6	By9	By19	By23	By27
I_4	Bz4	By1	By4	By7	By10	Bz19	Bz22	Bz25
I_5	Bz5	By2	By5	By8	By11	Bz20	Bz23	Bz26
I_6	Bz6	By12	By13	By14	By15	Bz21	Bz24	Bz27
K_0	Ky0	–	–	–	–	Ky7	Ky14	Ky21
K_1	Ky1	–	–	–	–	Ky8	Ky15	Ky22
K_2	Ky2	–	–	–	–	Ky9	Ky16	Ky23
K_3	Ky3	–	–	–	–	Ky10	Ky17	Ky24
K_4	Ky4	–	–	–	–	Ky11	Ky18	Ky25
K_5	Ky5	–	–	–	–	Ky12	Ky19	Ky26
K_6	Ky6	–	–	–	–	Ky13	Ky20	Ky27
O_0	Bx0	By0	By1	By2	By3	Bx1	Bx2	Bx3
O_1	Bx4	By4	By5	By6	By7	Bx5	Bx6	Bx7
O_2	Bx8	By8	By9	By10	By11	Bx9	Bx10	Bx11
O_3	Bx12	By12	By13	By14	By15	Bx13	Bx14	Bx15
O_4	Bx16	By16	By17	By18	By19	Bx17	Bx18	Bx19
O_5	Bx20	By20	By21	By22	By23	Bx21	Bx22	Bx23
O_6	Bx24	By24	By25	By26	By27	Bx25	Bx26	Bx27

Table 4–6. B Round Input and Output Mapping

4.4 Output Function

The Ky, Kz,, By, and Bz registers drive the final output function. Each of the 24 outputs consists of the XOR of nine terms given by the following formula:

(B0· K0) (B1· K1) (B2· K2) (B3· K3) (B4· K4) (B5· K5) (B6· K6) B7 K7

Where “ ” represents a logical XOR function and “· ” represents a logical AND function. Table 4–7 specifies the input values B and K to the 24 logic functions.

Input	B0	B1	B2	B3	B4	B5	B6	B7	K0	K1	K2	K3	K4	K5	K6	K7
Origin	Bz	Bz	Bz	Bz	Bz	Bz	Bz	By	Kz	Kz	Kz	Kz	Kz	Kz	Kz	Ky
Output bit																
0	17	26	22	27	21	18	2	5	3	6	0	9	4	22	5	10
1	5	20	15	24	2	25	0	16	20	18	7	23	15	5	3	25
2	22	5	14	16	25	17	20	11	7	19	2	10	22	4	13	21
3	19	3	15	11	21	16	27	1	6	14	9	8	17	18	12	24
4	19	6	17	18	22	7	9	12	25	6	5	2	10	15	21	8
5	3	7	4	8	16	6	5	17	27	14	2	4	24	19	1	12
6	8	21	27	2	11	24	12	3	17	26	4	16	27	7	22	11
7	9	5	7	4	8	13	3	15	9	10	19	11	7	6	8	23
8	26	13	23	10	11	7	15	19	13	12	18	24	15	23	7	16
9	1	0	19	11	13	16	24	18	0	5	20	25	1	24	9	27
10	26	13	9	14	10	4	1	2	14	23	27	25	17	19	1	22
11	21	15	5	3	13	25	16	27	6	21	17	15	26	11	16	7
12	20	7	18	12	17	1	16	0	11	22	20	0	26	23	17	2
13	14	23	1	12	24	6	18	9	8	4	3	14	20	26	23	15
14	19	6	21	25	23	1	10	8	19	0	18	2	13	8	24	14
15	3	0	27	23	19	8	4	7	16	21	24	25	12	27	15	18
16	6	5	14	22	24	18	2	21	3	5	8	25	7	27	2	26
17	3	4	2	6	22	14	12	26	11	14	23	17	22	13	19	4
18	25	21	19	9	10	15	13	22	1	16	14	11	12	6	10	19
19	23	11	10	20	1	12	14	4	21	1	10	20	18	26	9	13
20	11	26	20	17	8	23	0	24	20	21	9	25	12	3	15	0
21	9	17	26	4	27	0	15	6	18	12	21	27	1	16	24	20
22	22	12	2	10	7	20	25	13	13	0	3	16	22	11	26	9
23	27	24	26	8	0	9	18	23	2	0	13	5	4	8	10	3

Table 4–7. Output Function Input and Output Mapping

4.5 Operation

The HDCP cipher is used in four different ways during operation: hdcpBlockCipher, hdcpStreamCipher, hdcpRekeyCipher, and hdcpRngCipher. No change in HDCP cipher state occurs that is not explicitly identified in the following descriptions.

hdcpBlockCipher

This sequence is used during the first part of authentication to establish the session key, *Ks*, and during the vertical blanking interval preceding encrypted video frames to establish the frame key, *Ki*. Table 4–8 and Table 4–9 describes this sequence. The initial value for the B round register is specified with the concatenation operator "||". 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.

Step	Activity
1	Load B and K registers of the block module
2	Apply 48 clocks to the block module registers
3	Save the least significant 56 bits of the B register for future use as Ks/K_i
4	Transfer 84-bit B register values to the K registers
5	Reload B registers
6	Initialize the LFSR module
7	Assert rekey enable
8	Apply 56 clocks to the LFSR and block modules, saving the 64-bit M_i value during the last four clocks as specified in Table 4–11.
9	De-assert rekey enable

Table 4–8. hdcpBlockCipher Sequence

	Steps	clocks	LFSR init (56 bits)	K init	B init (65 bits)	B output (84 bits)	Output Function
hdcpBlockCipher at Authentication	1-3	48	–	*Km* (56 bits)	REPEATER || *An*	*Ks*	–
	6-9	56	*Ks*	*Ks* (84 bits)	REPEATER || *An*	–	R_0, M_0
hdcpBlockCipher at Vertical Blank	1-3	48	–	*Ks* (56 bits)	REPEATER || M_{i-1}	K_i	–
	6-9	56	K_i	K_i (84 bits)	REPEATER || M_{i-1}	–	R_i, M_i

Table 4–9. hdcpBlockCipher Initial Values and Outputs

For both the B and K round functions, the x, y, and z registers may be viewed as comprising a single register 84 bits in length, identified by B[83:0] and K[83:0]. The mapping of the x, y, and z registers into the full round register is specified by Table 4–10.

Round Register	B[83:56]	B[55:28]	B[27:0]	K[83:56]	K[55:28]	K[27:0]
Sub Register	Bz[27:0]	By[27:0]	Bx[27:0]	Kz[27:0]	Ky[27:0]	Kx[27:0]

Table 4–10. Round Register Bit Precedence

When fewer than 84 bits of output of a round register are required, the least significant bits are used. When fewer than 84 bits are available for initialization, the least significant bits are filled and the most significant bits are set to zero. For example, the 65-bit concatenation of REPEATER with *An* will be loaded into the Bx and By registers, plus the least significant nine bits of the Bz register, and the most significant 15 bits of the Bz register are set to zero. Similarly, the 56 bits from the Bx and By registers are saved as *Ks* or K_i during hdcpBlockCipher.

The origin of the M_i and r_i bits from the output function is specified by Table 4–11.

Warm-up Clock (Step 8)	Output Function Bits 23……16	Output Function Bits 15 ………… 0
53	–	M_i [63:48]
54	–	M_i [47:32]
55	r_i [15:8]	M_i [31:16]
56	r_i [7:0]	M_i [15:0]

Table 4–11. hdcpBlockCipher Output Function Bit Map

hdcpStreamCipher

For every video pixel as defined by the T.M.D.S. data enable (DE) signal, hdcpStreamCipher produces 24-bits of output data. Both the LFSR and block modules are clocked. The rekey enable signal is de-asserted.

hdcpRekeyCipher

During horizontal blanking intervals that immediately follow active lines of pixel data, hdcpRekeyCipher moves new key material from the LFSR module into the Block module. No other initialization of the cipher state is made, and no outputs are taken from the cipher during re-keying. Both the LFSR and block modules are clocked 56 times. The rekey enable signal is asserted.

hdcpRngCipher

The HDCP Cipher must be used as defined in Figure 4–6 to produce the value *An* required for the authentication protocol. This state diagram references video transmitter states from Figure 2–4.

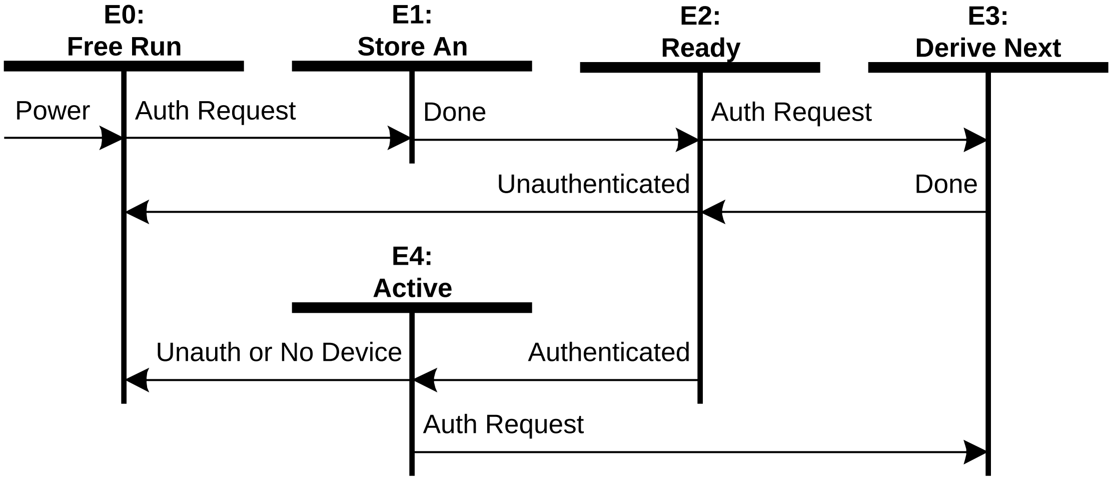

Figure 4–6. hdcpRngCipher State Diagram

Transition Any State:E0. On power up the HDCP Cipher is allowed to free run from its initial state, clocked by the pixel clock.

State E0: Free Run. The HDCP Cipher is clocked, from its current state, using the pixel clock.

Transition E0:E1. An authentication request to the video transmitter causes this transition. Authentication requests are identified by a video transmitter state transition to State:A1.

State E1: Store An. *An* is taken from the HDCP Cipher output function bits that are ordinarily used to produce *Mi*. This requires four pixel clocks.

Transition E1:E2. This transition is made immediately upon storage of *An*.

State E2: Ready. The *An* value is available for the authentication protocol.

Transition E2:E0. This transition is made if the current authentication fails, as indicated by a video transmitter state transition to State:A0.

Transition E2:E3. A new authentication request causes a new *An* value to be derived.

Transition E2:E4. The authentication protocol using the derived *An* is successful, as indicated by a video transmitter state transition to State:A4.

State E3: Derive Next. A new *An* is derived using the hdcpBlockCipher sequence, using the current values stored in the *Mi* and *Ki* registers.

Transition E3:E2. This transition is made immediately upon storage of *An*.

State E4: Active. The video transmitter is authenticated with a video receiver.

Transition E4:E0. This transition is made whenever the video transmitter becomes unauthenticated or if the video receiver is detached, as sensed by the hot plug pin of the DVI interface.

Transition E4:E3. An authentication request to the video transmitter causes this transition.

5 Renewability

It is contemplated that an authorized participant in the authentication protocol may become compromised so as to expose the secret device keys it possesses for misuse by unauthorized parties. In consideration of this, each video receiver is issued a unique set of secret device keys, matched with a non-secret identifier (the KSV). Through a process defined in the HDCP Adopter's License, the Digital Content Protection LLC may determine that a set of secret device keys has been compromised. If so, it places the corresponding KSVon a revocation list that the video transmitter checks during authentication. Other, authorized, receivers have different sets of secret device keys and, thus, are not affected by this revocation.

The video transmitter is required to manage system renewability messages (SRMs) carrying the KSV revocation list. These messages are delivered with content and must be checked when available. 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.

Table 5–1 gives the format of the HDCP SRM. All values are stored in big endian format.

Name	Size (bits)	Function
SRM ID	4	A value of 0x8 signifies that the message is for HDCP. All other values are reserved.
Reserved	36	Reserved for future definition. Must be 0x000.
Vector Revocation List Length	24	Specifies the combined length of all vector revocation lists contained in this SRM. The length is in bytes and includes the three bytes of this field, the combined size of the vector revocation lists, and the 40 bytes of the Digital Content Protection LLC signature.
Vector Revocation Lists (VRLs)	Variable	One or more VRLs, each in the format specified by Table 5–2.
Digital Content Protection LLC signature	320	A cryptographic signature of the SRM as defined by the Digital Signature Algorithm (DSA), as described in FIPS Publication 186-1 dated December 15, 1998. The first 160 bits is the big endian representation of the "r" value of the signature and the trailing 160 bits is the big endian representation of the "s" value produced by DSA.

Table 5–1. System Renewability Message Format

The SRM contains the vector revocation list, variable-length list of KSVs that belong to compromised devices. The format of the revocation list is specified in Table 5–2.

Name	Size (bits)	Function
Reserved	1	Set to 0.
Number of Devices	7	Specifies the number of device KSVs in this list.
Device KSVs	40	Forty-bit KSVs follow the type/number byte. The first byte following the type byte is the most significant byte of the first KSV in the list.

Table 5–2. Vector Revocation List Format

Appendix A. Test Vectors

Table A–1 gives facsimile key information for test purposes.

	Transmitter A1	Transmitter A2	Receiver B1	Receiver B2
Key Selection Vector	b70361f714	43f72d5066	511ef21acd	e72697f401
Key 0	4da4588f131e69	9aaba1f9ef907c	bc13e0c75bf0fd	93afe1ff4ca0ed
Key 1	1f823558e65009	34a0407731d1d0	ae0d2c7f76443b	efb49d4a25a4e4
Key 2	8a6a47abb9980d	97c682992dc5d9	24bf2185a36c60	e822d8a9335346
Key 3	f3181b52cbc5ca	da80caca68ed15	f4bc6cbcd7a32f	8812c3004e23d2
Key 4	fb147f6896d8b4	1866d9b51462a6	a72e69c5eb6388	dc63ba78d94263
Key 5	e08bc978488f81	d9fc9599bb7498	7fa2d27a37d9f8	47ebdf52776fd5
Key 6	a0d064c8112c41	7a062ac883f528	32fd3529dea3d1	4bce49472e0464
Key 7	b39d5a28242044	f5938c662af454	485fc240cc9bae	0479bed7732682
Key 8	b928b2bdad566b	ec3075e82d3ef2	3b9857797d5103	c5f800fad716d5
Key 9	91a47b4a6ce4f6	536e376e7ffc49	0dd170be615250	f53fd67ba9b9ec
Key 10	5600f8205e9d58	51c83a6cbeb116	1a748be4866bb1	6fb3901e5867f2
Key 11	8c7fb706ee3fa0	79d44ae1bd5f50	f9606a7c348cca	24c46f520f1be5
Key 12	c02d8c9d7cbc28	674b2563e27393	4bbb037899eea1	2038176d369ed7
Key 13	561261e54b9f05	7a1357efc538a2	190ecf9cc095a9	9ba9cd6a077a57
Key 14	74f0de8ccac1cb	6486e57ea46b02	a821c46897447f	5f2764b35c5591
Key 15	3bb8f60efcdb6a	bdf27a1ce8a299	1a8a0bc4298a41	ee32f1171f5356
Key 16	a02bbb16b22fd7	dc8bd1fa5b46b9	aefc0853e62082	d20a9e2f4d57fa
Key 17	482f8e46785498	27ef71efef9b73	f75d4a0c497ba4	439eb96d2daff0
Key 18	66ae2562274738	187599f603c947	ad6495fc8a06d8	1c68df6f868aaf
Key 19	3d4952a323ddf2	023ae9da303ecb	67c2020c2b2e02	dd50d7551dc6fb
Key 20	e2d231767b3a54	3d1cf6533dea8e	8f116b18f4ae8d	50b85379165c5f
Key 21	4d581aede66125	34dd5525f1890c	e3053fa3e9fa69	f45d64b097d6b5
Key 22	326082bf7b22f7	367dd774a07f4c	37d8002881c7d1	a1a154e07adb4d
Key 23	f61b463530ce6b	cdc34c8a6f56d1	c3a5fd1c15669c	0755ea83e47e71
Key 24	360409f0d7976b	de3413927363a8	9e93d41e0811f7	e1dca26293efe4
Key 25	a1e105618d49f9	21b11c739f45b3	2c4074509eec6c	e1092507ab8f45
Key 26	c98e9dd1053406	84440fadd281ac	8b7fd819279b61	3d56680db98e15
Key 27	20c36794426190	10f7900c65fef4	d7caada0a06ce9	0a49af413de66b
Key 28	964451ceac4fc3	30070704c8aa06	9297dca1f8c1db	90a814bbf971a0
Key 29	3e904504e18c8a	f287cb4063cb9d	5d1aaa99dea489	626b121ca0504f
Key 30	290010579c2dfc	97033445a4d587	60cb56ddbaa1d9	00f9bb7a94a1a7
Key 31	d7943b69e5b180	8051045091c10b	85d4ad5e5ff2e0	f485290cc5c1ba
Key 32	54c7ea5bdd7b43	d18f282074da20	1280161221df6d	baa873c54fdedf
Key 33	74fb5887c790ba	f2679a98828400	ca31a5f2406589	2d6a56233b8aba
Key 34	935cfa364e1de0	a6f0b6042a3dd7	1d30e8cb198e6f	a60d0379512312
Key 35	03075e159a11ae	3e5ddad097f5e1	d1c18bed07d3fa	942582078dadb8
Key 36	05d3408a78fb01	3ad1f8a2e5958f	cec7ec09245b43	8395a4b022082f
Key 37	0059a5d7a04db3	f025bb1c085d4f	b08129efedd583	cb12fe97842b60
Key 38	373b634a2c9e40	0864213d6d50c1	2134cf4ce286e5	282ffe78f2f95c
Key 39	2573bbb4562041	9018b0ff3ab170	edeef9d099b78c	f6491f33c7ef53

Table A–1. Sample Device Keys

Transmitter Device #1 examines the KSV of Receiver Device #1 and combines its own secret device keys that correspond to the bit positions of all of the ones in the KSV. Receiver Device #1 examines the KSV of Transmitter Device #1 and combines its own secret device keys that correspond to the bit positions of all of the ones in the KSV. Table A–2 shows the 56-bit binary addition of keys performed by Transmitter Device #1 and Receiver Device #1, and the corresponding equivalent values derived for Km and Km'.

Transmitter Device #1 Sum of Keys Calculation		Receiver Device #1 Sum of Keys Calculation	
Key 0	`4da4588f131e69`	Key 2	`24bf2185a36c60`
Key 2	`8a6a47abb9980d`	Key 4	`a72e69c5eb6388`
Key 3	`f3181b52cbc5ca`	Key 8	`3b9857797d5103`
Key 6	`a0d064c8112c41`	Key 9	`0dd170be615250`
Key 7	`b39d5a28242044`	Key 10	`1a748be4866bb1`
Key 9	`91a47b4a6ce4f6`	Key 12	`4bbb037899eea1`
Key 11	`8c7fb706ee3fa0`	Key 13	`190ecf9cc095a9`
Key 12	`c02d8c9d7cbc28`	Key 14	`a821c46897447f`
Key 17	`482f8e46785498`	Key 15	`1a8a0bc4298a41`
Key 20	`e2d231767b3a54`	Key 16	`aefc0853e62082`
Key 21	`4d581aede66125`	Key 21	`e3053fa3e9fa69`
Key 22	`326082bf7b22f7`	Key 22	`37d8002881c7d1`
Key 23	`f61b463530ce6b`	Key 24	`9e93d41e0811f7`
Key 25	`a1e105618d49f9`	Key 25	`2c4074509eec6c`
Key 26	`c98e9dd1053406`	Key 32	`1280161221df6d`
Key 27	`20c36794426190`	Key 33	`ca31a5f2406589`
Key 28	`964451ceac4fc3`	Key 34	`1d30e8cb198e6f`
Key 32	`54c7ea5bdd7b43`	Key 36	`cec7ec09245b43`
Key 36	`05d3408a78fb01`	Key 37	`b08129efedd583`
Key 38	`373b634a2c9e40`	Key 39	`edeef9d099b78c`
RESULT (*Km*):	**`5309c7d22fcecc`**	**RESULT (*Km*')**	**`5309c7d22fcecc`**

Table A–2. Sample Km Calculation

Table A–3 gives test vectors for the four possible authentication pairs of test keys in Table A–1. The test vectors cover two lines of video data, each with eight pixels per line. The video receiver does not support downstream DVI connections (REPEATER = 0).

	A1 – B1	A1 – B2	A2 – B1	A2 – B2
Km	5309c7d22fcecc	f6aee46089c923	4afe34dbec1205	a423d78b8676a7
REPEATER || An	034271c130c070403	0445e62a53ad10fe5	083bec2bb01c66e07	00351f7175406a74d
Ks	54294b7c040e35	4e60d941d0e8b1	2c9bef71df792e	1963deb799ee82
M_0	a02bc815e73d001c	e7d28b9b2f46c49d	8e1e91f6d8ae4c25	d05d8c26378a126e
R_0	8ae0	fb65	3435	4fd5
K_1	d692b7ee1d40e8	e46f51311a959a	f3e27849d067c1	65f793e160ec27
M_1	1dbf44e50f523e56	445b5c6eebf657ff	23d89127a5ee6c26	68be984885aafef7
Line 1, Pixel 1	R 59 G c0 B 3e	R 56 G bf B 8a	R 11 G 07 B d2	R b8 G 2c B 9c
Line 1, Pixel 2	R 9e G e5 B fe	R 2c G 26 B 03	R b1 G 8f B 7f	R 9b G 34 B e3
Line 1, Pixel 3	R 9a G f9 B 19	R 88 G 43 B dc	R 3c G fb B 8c	R 1c G fa B d7
Line 1, Pixel 4	R 5b G 5d B 6c	R 1d G db B bd	R a3 G 97 B 0c	R 00 G A0 B 08
Line 1, Pixel 5	R 55 G dc B de	R e6 G 32 B 13	R 38 G 94 B 3e	R ce G c3 B f4
Line 1, Pixel 6	R e5 G 87 B 63	R 36 G 34 B 24	R ac G 84 B da	R f4 G 36 B 27
Line 1, Pixel 7	R be G fc B c7	R 48 G 82 B 8f	R b8 G a4 B 73	R b6 G 36 B f7
Line 1, Pixel 8	R a1 G b5 B 65	R 99 G b9 B db	R 2f G c5 B c0	R 24 G bd B 8b
Horizontal Blank Re-Key				
Line 2, Pixel 1	R 12 G 6b B 14	R 9c G ac B 7b	R 6c G 64 B c7	R 73 G 9f B 2e
Line 2, Pixel 2	R 06 G 4a B 73	R 40 G 11 B d0	R ba G 05 B 8d	R f6 G 1e B 16
Line 2, Pixel 3	R f8 G bb B 15	R aa G 3c B e6	R 62 G 17 B ff	R e2 G 8c B 59
Line 2, Pixel 4	R cc G e6 B 21	R e6 G e9 B ac	R f1 G 35 B df	R d9 G 8a B 86
Line 2, Pixel 5	R 87 G 95 B 78	R 7a G d5 B 2e	R c2 G 36 B 92	R c5 G eb B 96
Line 2, Pixel 6	R d2 G 03 B f7	R 94 G 1f B 35	R 47 G a4 B 94	R c0 G b3 B ce
Line 2, Pixel 7	R 62 G 81 B 44	R a7 G 85 B 64	R 59 G b7 B a1	R eb G 26 B f3
Line 2, Pixel 8	R 80 G d8 B 75	R f7 G 45 B 16	R 9d G 96 B ea	R f4 G 9e B e1

Table A–3. Sample Authentication and Encryption Values (REPEATER = 0)

Table A–4 gives test vectors for the four possible authentication pairs of test keys in Table A–1. The test vectors cover two lines of video data, each with eight pixels per line. The video receiver supports downstream DVI connections (REPEATER = 1).

	A1 – B1	**A1 – B2**	**A2 – B1**	**A2 – B2**
Km	5309c7d22fcecc	f6aee46089c923	4afe34dbec1205	a423d78b8676a7
REPEATER || An	134271c130c070403	1445e62a53ad10fe5	183bec2bb01c66e07	10351f7175406a74d
Ks	bc607b21d48e97	b7894f1754caaa	fe3717c12f3bb1	aac4147081a2d0
M_0	372d3dce38bbe78f	43d609c682c956e1	536dee1e44a58bf4	38b57ad3cdd1b266
R_0	6485	1367	dd9b	7930
K_1	98b281e1876a9a	ffbfea4bc7fd2c	a1ec276b2ddaf0	0f0b83888e3209
M_1	016f9561e001f80d	2a067368042fa1aa	b365f8813c45db0b	06471e358f601ce4
Line 1, Pixel 1	R 33 G 4e B 55	R bc G 9c B a4	R 4a G c7 B d3	R c2 G c8 B 84
Line 1, Pixel 2	R d2 G 37 B 4e	R 43 G 10 B df	R 30 G a7 B ec	R 2f G 7c B 68
Line 1, Pixel 3	R 0e G 22 B f5	R b1 G e0 B 12	R 2d G 6e B 36	R 90 G 0b B e5
Line 1, Pixel 4	R c1 G 31 B 8f	R 27 G d0 B 5a	R e1 G 75 B b6	R 9e G de B 54
Line 1, Pixel 5	R dc G a1 B a7	R d8 G aa B 3d	R 94 G ff B fb	R 78 G cd B 8c
Line 1, Pixel 6	R 27 G e7 B c3	R 3f G 2a B 64	R 11 G aa B c1	R 38 G a5 B b8
Line 1, Pixel 7	R 56 G 3e B c9	R 2e G 00 B 0a	R 5c G 71 B 66	R 32 G ff B 1e
Line 1, Pixel 8	R 10 G dc B 2f	R f2 G 47 B 63	R be G 33 B 6f	R e4 G d9 B 0c
Horizontal Blank Re-Key				
Line 2, Pixel 1	R 73 G 03 B 22	R e4 G 97 B f1	R 0b G a7 B ec	R 62 G 0f B 61
Line 2, Pixel 2	R 69 G 01 B 36	R df G 15 B 0e	R 4f G 10 B 1e	R 33 G 73 B 52
Line 2, Pixel 3	R 3d G 27 B 53	R 2f G 44 B 7b	R fe G 16 b 16	R cd G 96 B fd
Line 2, Pixel 4	R fe G 41 B 50	R 0c G 9b B ae	R 52 G e6 B 35	R 53 G ea B d5
Line 2, Pixel 5	R a8 G 18 B 8d	R 93 G db B da	R db G 8d B b7	R 33 G a9 B 31
Line 2, Pixel 6	R 1a G 02 B 91	R a7 G f9 B 01	R 18 G f0 B d9	R cc G 34 B 86
Line 2, Pixel 7	R 8c G 29 B ce	R 1a G 39 B 9a	R f5 G 9a B 63	R 6e G e0 B bb
Line 2, Pixel 8	R 89 G cd B bf	R 4b G 54 B 00	R d4 G ac B aa	R d2 G fc B 4b

Table A–4. Sample Authentication and Encryption Values (REPEATER = 1)

Sequence	LFSR 0	LFSR 1	LFSR 2	LFSR 3	SH 0	SH 1	SH 2	SH 3
Load								
1								
2								
3								
…								
47								
48								
Load	0x01e35	0x00040	0x025be	0x15429	01	01	01	01
1	0x01c6b	0x00081	0x04b7c	0x0a853	10	10	10	10
2	0x018d6	0x00102	0x096f8	0x150a7	00	01	11	01
3	0x011ac	0x00204	0x02df0	0x0a14e	00	00	11	11
4	0x00358	0x00409	0x05be0	0x1429c	00	00	10	11
5	0x006b0	0x00812	0x0b7c0	0x08539	00	00	01	10
6	0x00d60	0x01024	0x06f81	0x10a72	00	00	00	01
7	0x01ac0	0x02049	0x0df03	0x014e4	01	00	00	00
8	0x01581	0x00093	0x0be07	0x029c9	10	01	00	00
…	…	…	…	…	…	…	…	…
49	0x01cbc	0x03218	0x05712	0x0ab75	10	10	01	11
50	0x01979	0x02431	0x0ae24	0x156eb	11	00	10	11
51	0x012f3	0x00863	0x05c48	0x0add7	10	01	01	10
52	0x005e6	0x010c6	0x0b891	0x15bae	01	10	10	01
53	0x00bcc	0x0218d	0x07122	0x0b75c	10	01	01	10
54	0x01799	0x0031a	0x0e245	0x16eb8	01	00	11	00
55	0x00f32	0x00634	0x0c48b	0x0dd70	10	10	01	10
56	0x01e65	0x00c69	0x08917	0x1bae1	00	01	11	01

Table A–5. LFSR Module States During A1 - B1 Authentication (REPEATER = 0)

Sequence	Kx	Ky	Kz	Bx	By	Bz	Output
Load	0x22fcecc	0x5309c7d	0x0000000	0xc070403	0x271c130	0x0000034	
1	0x000084c	0xf458fff	0x7f722dc	0xa5d4b70	0x8ea8888	0x9f6066d	0xbe70ee
2	0x0ed9f8a	0xb444236	0x3b62e76	0x8fa5383	0x5d17cd7	0x2e71e83	0x007023
3	0x70ef0ef	0x9aa103f	0x8aa659d	0x49d0347	0xe71b545	0xd39af92	0xdd51b7
4	0xc8f3da5	0x8bbb85f	0x58047e6	0x05add47	0xaf2ff95	0x4371447	0xeae10f
5	0x6b68710	0x1826042	0xc20a675	0x5693206	0xd034757	0x71f4c59	0xe0e624
6	0xd4c9cf4	0x0014506	0x6c11733	0xf679cf3	0xbe06351	0x412aafc	0x6104f9
7	0x2ff2231	0x059031a	0xd84c367	0x7c6878b	0x735a2d2	0x2d4fba7	0x12c5e4
8	0x1c13406	0x516f805	0x3e231f5	0x61f3f4d	0xccb03b9	0x3030a78	0x9f08dc
…	…	…	…	…	…	…	…
41	0x7dc29a3	0x5895932	0x26047a5	0x12b9cbd	0xe40581a	0xc892f27	0x1cfd71
42	0xba7d2b0	0xf1cfeac	0x36eb45d	0xa8bab0f	0x083213e	0x38fd0ef	0xb90f28
43	0xdd26650	0x29e8ca4	0xbf0109c	0x04a0c9b	0xf8cd136	0xb6b8827	0xf32344
44	0xf928c5b	0xc70cecd	0xcc71bb9	0x004c69f	0xf8cfb57	0x20d8664	0xff2c26
45	0x491d801	0xf630446	0x43655f6	0x26727b8	0xb6866b1	0x48253f0	0xead81d
46	0x9281463	0x891c25b	0x2c40a10	0xe2e3627	0xce25f1d	0x6fd76d2	0x7cb35d
47	0x37ef335	0xbb8429b	0xfad91c5	0x8bb8770	0x94322d6	0xbc24e18	0x4ac7aa
48	0x7bd96ba	0xee950f7	0x749f3d9	0xc040e35	0x54294b7	0x1c61d8e	0x37d937
Load	0xc040e35	0x54294b7	0x1c61d8e	0xc070403	0x271c130	0x0000034	
1	0x3772e0b	0x6595cd5	0x93d46aa	0xf5f1bea	0x8ea8888	0x9f6066d	0x5d74aa
2	0xfcdc369	0x18f685a	0x22626f1	0x48ec1f7	0x5d17cd7	0x083878b	0x1e60bc
3	0x67f044d	0xd5eb45a	0x8ca9144	0x034b338	0x3ac66a8	0xdc9e6f6	0x4c29b4
4	0x046af2c	0x992df09	0xd7b21a9	0x845e47f	0xce06983	0xc50059e	0x1c3d69
5	0x1a7c13c	0x6aed6fb	0x57ba318	0xea50517	0xc09dcdf	0xcdbf157	0x2d0855
6	0x82ff268	0xfd00a63	0xf4c6f06	0x00bc25d	0xb24cd67	0xa94407a	0xddb851
7	0xe602372	0xe4f1798	0x6487e18	0x47a81d0	0x3ca6b73	0x90eea67	0x5605dd
8	0xa251408	0x26ca144	0x2c8a821	0x700ece4	0x1f2ccf5	0x575dec4	0x44236d
…	…	…	…	…	…	…	…
49	0xade5581	0x026eead	0x58676ad	0x19978d8	0x207678c	0x552b693	0x65e697
50	0xc1cdfad	0x29eb9e5	0x85864c6	0x3a260ed	0xd817a5a	0xf2e4743	0xa341ef
51	0x75114c3	0x6923621	0xc5367fa	0x4c7b24b	0x4c7ad96	0x4bf179e	0x6c2f44
52	0x5e00de1	0x31ba2ec	0x9352a05	0x21f7177	0x1ce1a8a	0x5fe9127	0xdce5b0
53	0xa8a8b05	0x470ad68	0x35c28f6	0x3eaf43f	0x194bf81	0xb8d5477	0x14a02b
54	0x56a5801	0x5bd1d70	0xd724992	0xf41fb7d	0x6aafc2c	0x3fbf3ef	0x54c815
55	0x6c30c38	0xf15bf0e	0xfc5799d	0xb673b37	0x921be44	0x956fe75	0x8ae73d
56	0x8451307	0x58cff28	0x9ee2338	0x346ebe6	0x189def7	0xf04cb0e	0xe0001c

Table A–6. Block Module States During A1 - B1 Authentication (REPEATER = 0)

Sequence	LFSR 0	LFSR 1	LFSR 2	LFSR 3	SH 0	SH 1	SH 2	SH 3
Load								
1								
2								
3								
…								
47								
48								
Load	0x018b1	0x03d0e	0x06ca0	0x14e60	01	01	01	01
1	0x01162	0x03a1d	0x0d941	0x09cc1	00	10	11	00
2	0x002c4	0x0343b	0x0b282	0x13983	01	00	11	10
3	0x00588	0x02876	0x06504	0x07307	10	01	01	11
4	0x00b10	0x010ed	0x0ca09	0x0e60f	01	10	10	10
5	0x01620	0x021db	0x09413	0x1cc1e	10	00	11	00
6	0x00c40	0x003b7	0x02826	0x1983c	01	10	10	10
7	0x01881	0x0076e	0x0504d	0x13078	11	01	00	11
8	0x01103	0x00edd	0x0a09a	0x060f0	11	10	01	10
…	…	…	…	…	…	…	…	…
49	0x005c3	0x016e4	0x0917e	0x1efbd	01	00	00	01
50	0x00b86	0x02dc8	0x022fd	0x1df7a	00	01	00	00
51	0x0170d	0x01b90	0x045fb	0x1bef4	00	00	10	00
52	0x00e1b	0x03721	0x08bf6	0x17de9	00	00	00	10
53	0x01c36	0x02e42	0x017ed	0x0fbd3	01	00	00	01
54	0x0186d	0x01c84	0x02fda	0x1f7a6	11	00	00	00
55	0x010db	0x03909	0x05fb4	0x1ef4d	10	01	00	00
56	0x001b6	0x03212	0x0bf68	0x1de9b	01	00	10	00

Table A–7. LFSR Module States During A1 – B2 Authentication (REPEATER = 0)

Sequence	Kx	Ky	Kz	Bx	By	Bz	Output
Load	0x089c923	0xf6aee46	0x0000000	0xad10fe5	0x5e62a53	0x0000044	
1	0x000ace8	0x2bbe222	0xa84ba32	0xf8ee8f0	0x4c79444	0x649180e	0xb24463
2	0xbe2db4d	0xced43e8	0x6cf4c5d	0xb0bccb3	0xcd48ee4	0xfbde86b	0x0ff14d
3	0x59aaa16	0x420acae	0x948ddf1	0x4f31d66	0x5e99939	0x8945bd4	0x5a7c22
4	0x6716e27	0xc71eabf	0x728216a	0x948e7ab	0xb5980ca	0x3969dfa	0xe29870
5	0x2b8be74	0xc7b7cd8	0x1896efd	0xdd99072	0xdd8b36e	0x9005894	0x252d85
6	0x417f923	0xf719e90	0xd5c1459	0xdc0bba0	0x6178407	0x066cb0a	0x5195fa
7	0x6c1faa9	0xf7175fd	0x50bb276	0xcafbc7c	0x32a2ec3	0xa479ab9	0xced7d1
8	0x90a1447	0xad4dd26	0x59afdb6	0xfa48546	0x6ebb9cf	0x890acc2	0xd92360
…	…	…	…	…	…	…	…
41	0x456a8de	0x218a73d	0xefe8143	0xdb40d6f	0x8adb81b	0x7f17e90	0x4b21a1
42	0x5bb75c0	0x9e32509	0xcd4d66f	0x94b2edc	0x91aaaf6	0x3894216	0x537e81
43	0x692b31d	0x40c7b06	0xeb692c8	0x5b4a26a	0x7c0b63f	0xb5e23ed	0x71f997
44	0x4ac7e44	0x584dad4	0x2606dca	0xb41c724	0xde66448	0x90f07c0	0x9b4c0f
45	0x995c381	0xe782e99	0x500545a	0x296761d	0x33b5aa8	0xd7c96dd	0xcce274
46	0x2a39ef6	0xb3509f9	0xbd26dfe	0xf7d1275	0xd7972de	0xa1c5513	0xa9e21a
47	0xe937d30	0x7910780	0x03575d7	0x0e9e5a9	0x235c870	0x246431c	0x8d7b49
48	0xb9af224	0x04c8a5f	0x49c96b1	0x1d0e8b1	0x4e60d94	0x072bad0	0x1cfb41
Load	0x1d0e8b1	0x4e60d94	0x072bad0	0xad10fe5	0x5e62a53	0x0000044	
1	0x8adc6e8	0xb659c1e	0x70ae5ce	0x4c36286	0x4c79444	0x649180e	0xfeaeeb
2	0xe647934	0x7ec73a0	0xae21cfc	0x57c3737	0xcd48ee4	0x131ec75	0xe6e976
3	0xfa28037	0x602e4c5	0xcc87a66	0x1fe7698	0xf433b91	0x990c71a	0x47ee81
4	0x0d609b0	0x76b0413	0xbb909ab	0xc160202	0x2e4b770	0xd5b0319	0x09463e
5	0x8f2b473	0x00b1039	0x54e4007	0xf914da7	0xbd17a23	0x9746424	0x341d4a
6	0x91fb8aa	0x6445ea6	0x8649c97	0x623f7e9	0xf5e67b9	0xb986c8a	0x61be45
7	0x88d8719	0x4f9ea67	0x5195717	0x2f6bf08	0x42af423	0x0f517b2	0x38c278
8	0x4e72913	0x5e4a60f	0xef64d8e	0xa7afa70	0x46d5f5f	0x8599680	0x366d9f
…	…	…	…	…	…	…	…
49	0x4dda715	0x5cf4582	0x66dc877	0x4e69fc3	0x6790add	0x692ce89	0x40f21c
50	0x4db2b7f	0xfb2f397	0x76dedec	0x20ef253	0x81e7d6b	0xf0b76f9	0x9c8062
51	0x6f8bf8a	0x0579c7f	0xa79d4cc	0xf23684b	0x79e04b8	0x71c4515	0xef455b
52	0x57b4273	0x7cc013c	0x4a37fd9	0xa63e183	0x13f3943	0xaf26eed	0x9b00a8
53	0x6a718ef	0x43667bb	0x91c7a99	0x9383356	0x3f262d4	0xda416b4	0xbee7d2
54	0x5764f30	0xca377a9	0x61cb7fc	0x75526c2	0x5439e56	0xc8e2a8a	0x168b9b
55	0x1aac873	0xf9340e8	0x0ce402a	0x8504037	0x18ad8b4	0xb818ef9	0xfb2f46
56	0x365eb8d	0x02468c0	0x31071ef	0x01c71f2	0xc7ac9e7	0xc1ffc01	0x65c49d

Table A–8. Block Module States During A1 – B2 Authentication (REPEATER = 0)

Sequence	LFSR 0	LFSR 1	LFSR 2	LFSR 3	SH 0	SH 1	SH 2	SH 3
Load								
1								
2								
3								
…								
47								
48								
Load	0x0192e	0x01df7	0x077b8	0x02c9b	01	01	01	01
1	0x0125c	0x03bef	0x0ef71	0x05936	11	00	10	10
2	0x004b8	0x037df	0x0dee3	0x0b26c	11	10	01	01
3	0x00970	0x02fbf	0x0bdc7	0x164d8	01	11	00	11
4	0x012e0	0x01f7f	0x07b8e	0x0c9b0	11	01	01	10
5	0x005c1	0x03eff	0x0f71d	0x19360	01	10	10	11
6	0x00b82	0x03dfe	0x0ee3b	0x126c1	00	01	11	10
7	0x01705	0x03bfd	0x0dc76	0x04d82	00	00	11	01
8	0x00e0b	0x037fb	0x0b8ed	0x09b04	00	00	10	10
…	…	…	…	…	…	…	…	…
49	0x016ef	0x004ea	0x08ffb	0x18374	01	11	11	10
50	0x00dde	0x009d4	0x01ff7	0x106e8	10	11	11	01
51	0x01bbd	0x013a9	0x03fee	0x00dd0	01	01	11	11
52	0x0177b	0x02753	0x07fdd	0x01ba0	00	10	11	11
53	0x00ef6	0x00ea6	0x0ffbb	0x03740	01	01	01	11
54	0x01dec	0x01d4d	0x0ff77	0x06e81	10	11	00	11
55	0x01bd9	0x03a9b	0x0feef	0x0dd02	01	01	10	01
56	0x017b3	0x03537	0x0fddf	0x1ba04	10	11	01	00

Table A–9. LFSR Module States During A2 – B1 Authentication (REPEATER = 0)

Sequence	Kx	Ky	Kz	Bx	By	Bz	Output
Load	0xbec1205	0x4afe34d	0x0000000	0x1c66e07	0xbec2bb0	0x0000083	
1	0x888e2ea	0x414b444	0x97a0589	0xf087578	0x3d5b332	0x610f071	0x001a91
2	0x4625e41	0xcd48c5f	0x3a77722	0x17b01a9	0x0638644	0xb71a3c5	0x892758
3	0xc9402d8	0x5ce2e8b	0x2d46dd1	0xcba2da3	0x45c8159	0x0c27e9f	0xd3c6e1
4	0x9f4f7b0	0x4c9fc33	0x7975e63	0xb1a5c1f	0x37140d4	0x78f6cfb	0x916ff8
5	0xa52c6b9	0x0ab1bea	0x3f59b80	0x66c7c4e	0xef8a601	0xd5f6819	0x21475c
6	0xe828e8c	0x1f4fe28	0xf9ae9ca	0xa6e1944	0x11989fd	0x4338020	0x729008
7	0x3d9656f	0x9313d6c	0xd525839	0x3d3cf97	0x2d456aa	0x5592482	0x2c2762
8	0x0b5904f	0xe168c0e	0x8549a6c	0x8e384cb	0xfd25ff0	0x40578b4	0xa66b25
…	…	…	…	…	…	…	…
41	0xf907779	0x8add56d	0xa2bf28b	0xb6d2591	0x8cbe163	0x1db3ce9	0x55f6f1
42	0xbb149e8	0x34b44fe	0xe899a28	0x7ec27a0	0xbdae914	0xbcc46bf	0xb1c490
43	0x852bc22	0x30c541b	0x4ba8ad0	0xbacaa81	0xf2df6bc	0x7796efa	0x134543
44	0xe0dcc66	0x3380692	0x2f59c16	0x5875f9a	0x03ea16f	0x80bc2ab	0xf8b3c8
45	0xbd69a67	0x11e9f3b	0xb0d15db	0xcd318e7	0xbcace72	0x5aa586f	0x49d410
46	0x992aba4	0x79ccd6c	0x374d0da	0x4a507c8	0xd761f3d	0x3849c30	0x4d30b7
47	0x02d7a9c	0x69e0827	0x75c491b	0x1c3734c	0x1ebaf33	0x8e6e1e4	0x9df48b
48	0x28d5897	0x4f55c34	0x1bf2686	0x1df792e	0x2c9bef7	0x07b1c9f	0xebdeef
Load	0x1df792e	0x2c9bef7	0x07b1c9f	0x1c66e07	0xbec2bb0	0x0000083	
1	0xfd88a6c	0x1aec3ba	0x548b6d5	0xfb705c6	0x3d5b332	0x610f071	0x636064
2	0x0876369	0x710f070	0x03a9952	0x68afa97	0x0638644	0x2a048b2	0x3a375c
3	0xfdcf763	0x64400d6	0x6888c5c	0x81f7bc9	0xab26acb	0x5146df0	0x1b8dbf
4	0x0cb1f80	0x6710244	0xd810320	0x8a558ef	0xc4934bb	0xfcbe390	0x2fba5d
5	0x7a77bb1	0x545b44d	0xacc6c17	0xefc1031	0x8a7bd55	0x6f02498	0x66bde4
6	0x629697d	0xdc585bb	0x5b8f82d	0x9e3cd09	0xe34bee9	0xad76510	0x9b04a5
7	0x2d0fd29	0x6095002	0x10fd4d1	0x161afae	0x9356147	0xf76daf9	0x9467c6
8	0x7745ff4	0xddcd316	0x042bd5c	0x9cc0fc2	0x7262896	0x73c7ad4	0xa7a735
…	…	…	…	…	…	…	…
49	0x3e266d1	0xc895108	0x65cffa5	0xbbf95cd	0x063edad	0x9f1843e	0xd2a1f8
50	0x1aff812	0xc8cc3bb	0x2e34b69	0x548d48b	0x0fc340a	0x7ca499b	0xdeebe6
51	0xeb214ef	0x067b1f8	0x19c630a	0xe7c0a44	0x66f4697	0x541cbf6	0x4420a7
52	0x2403450	0x5331c01	0x59f99e8	0xa39e281	0x8971df1	0x4c21780	0x9f6e12
53	0x96b81f7	0xc44f275	0x3e91d6c	0x644040d	0xd338e4e	0x0afa6f2	0xd38e1e
54	0xaf435aa	0x8ba5ab2	0x90519f8	0x72a4777	0xc552143	0x2630971	0x6c91f6
55	0x011f064	0x0a7aa39	0x072d48d	0x2802af7	0x15041a9	0xea862e3	0x34d8ae
56	0x7532414	0x0a296c3	0xa5510c1	0x6891e10	0x5316410	0x45e1c10	0x354c25

Table A–10. Block Module States During A2 – B1 Authentication (REPEATER = 0)

Sequence	LFSR 0	LFSR 1	LFSR 2	LFSR 3	SH 0	SH 1	SH 2	SH 3
Load								
1								
2								
3								
…								
47								
48								
Load	0x01e82	0x0399e	0x0ef5b	0x11963	01	01	01	01
1	0x01d04	0x0333c	0x0deb7	0x032c7	00	10	10	10
2	0x01a09	0x02678	0x0bd6f	0x0658e	00	01	01	00
3	0x01413	0x00cf0	0x07adf	0x0cb1c	01	10	10	00
4	0x00827	0x019e1	0x0f5bf	0x19638	11	00	11	00
5	0x0104e	0x033c2	0x0eb7e	0x12c71	10	10	10	01
6	0x0009d	0x02785	0x0d6fd	0x058e3	01	11	01	00
7	0x0013b	0x00f0b	0x0adfb	0x0b1c7	10	11	10	10
8	0x00276	0x01e17	0x05bf7	0x1638e	00	11	01	01
…	…	…	…	…	…	…	…	…
49	0x0055e	0x02e73	0x08f69	0x07085	11	00	11	10
50	0x00abd	0x01ce7	0x01ed3	0x0e10b	11	01	01	01
51	0x0157b	0x039cf	0x03da6	0x1c217	11	11	00	11
52	0x00af6	0x0339f	0x07b4c	0x1842f	10	11	01	10
53	0x015ed	0x0273f	0x0f699	0x1085e	01	01	10	01
54	0x00bdb	0x00e7f	0x0ed32	0x010bc	00	10	11	00
55	0x017b6	0x01cff	0x0da64	0x02179	00	00	11	01
56	0x00f6c	0x039fe	0x0b4c8	0x042f3	10	00	01	11

Table A-11. LFSR Module States During A2 – B2 Authentication (REPEATER = 0)

Sequence	Kx	Ky	Kz	Bx	By	Bz	Output
Load	0xb8676a7	0xa423d78	0x0000000	0x406a74d	0x51f7175	0x0000003	
1	0x666e2c6	0x1fb7111	0x802f1c8	0xf7f1edb	0x7052777	0x0f40723	0xf05140
2	0x222564c	0xeacf83b	0x56392e2	0xf8c5faf	0x9e2408b	0x787caa9	0x91937d
3	0x3a7d9e3	0x39004ba	0x11f7a6a	0xd50bb43	0x88db561	0x91040c2	0x026852
4	0x47614d8	0x6494d8a	0x3b4f25b	0x4395a00	0x53d0514	0xe2e383d	0x3bc587
5	0xdb4e14e	0x845a7cc	0xbf7698d	0xbeab442	0xbe1b11f	0x6a72f32	0xb649af
6	0x9f50e9a	0x72b9f8a	0xe83d832	0x2446aa1	0x2711b9c	0xcdda1d2	0x76b8c5
7	0x3ea1bc9	0x2ef84ca	0x8b460ed	0xff20d53	0x0d6ac1d	0x45a75c4	0x1cfba1
8	0x16166f2	0xaa7c2ef	0x1d92ed2	0x962b376	0x2b810f5	0x085c932	0x34494d
…	…	…	…	…	…	…	…
41	0x2b7a4ee	0x76aaca6	0x990b686	0xe19348b	0xfea6035	0xa9afaf0	0x37e446
42	0x2420fda	0xc71cbcb	0xd3a43cf	0x3b01c23	0xa98bd4f	0x4c62274	0x58a13f
43	0x1b38c46	0x7b286a6	0x1d6e079	0x7fd5dd1	0xd04a459	0x7c16c08	0xd854bb
44	0x9ecc174	0xa97266e	0xa162b3f	0xbab8ead	0xff58f91	0x7740eea	0x5b3ceb
45	0x039d3b7	0x039e9b4	0xbc7dd68	0xfa0a1ce	0xb752298	0xb13d8cf	0xdf6e53
46	0x5096513	0xc3ac236	0x4adda17	0xdc0290a	0xff95916	0x9f7e6f6	0x1dbde4
47	0xc0f65b9	0x566da3d	0x55dab36	0x179735f	0x586589a	0xba7cd32	0xc580c5
48	0x83f87f0	0xd6f60e1	0xb0ffacc	0x799ee82	0x1963deb	0xd2ecfc7	0x531799
Load	0x799ee82	0x1963deb	0xd2ecfc7	0x406a74d	0x51f7175	0x0000003	
1	0xc4e8ff1	0x68b3b95	0x5a86976	0x3729648	0x7052777	0x0f40723	0xda19ca
2	0xf2c964d	0x2f49256	0x8ec9541	0xb06dc21	0x9e2408b	0x11e91dc	0xa8a0b8
3	0x26464e7	0xab964b8	0xc6112c9	0x72cfc92	0x4417ad5	0xc11c247	0xe28985
4	0x3b7c3f4	0x20c212b	0x5a8464d	0x235fdd1	0xc5a1984	0x7152f6d	0x8d3851
5	0x0c23381	0x1700053	0xf79219e	0x593da63	0xc18c5f2	0xaec1bce	0xb484bf
6	0x6c9733a	0xaa9fab7	0x3ff3223	0x3295feb	0x8e7c3b9	0x394597d	0x30ed7d
7	0xf811f2c	0x5e2ced9	0x7d2aca5	0xe469c78	0xacc10da	0xba93ae2	0xa60a41
8	0x1ed5c78	0xc42186b	0xc39983c	0x0c80d4e	0xccbafe1	0x235ff24	0x25ab7f
…	…	…	…	…	…	…	…
49	0x7d252c0	0x081db0e	0x329083e	0x3036a4c	0x4c638fc	0x9042db0	0x9c7024
50	0xba0eaa9	0x1c0b139	0x9f56b08	0x4771510	0x4f22c73	0x6321faf	0x4732f1
51	0x531015d	0xe8cd792	0xceb6a51	0x9327e2f	0xd768e6e	0x5ca36be	0x45edc6
52	0xd1a375c	0xd925c31	0xc37b8b1	0xb098639	0x8316b0f	0x7e66ad9	0x62404c
53	0xb0a7396	0xd77e370	0xc279e10	0x0b2b48e	0x3e28ad6	0xbb19243	0xc8d05d
54	0xd5c53b3	0x9fb7633	0xb69eb4a	0x88af562	0x5c2925d	0x8b95f94	0x5c8c26
55	0x33dc74d	0x9b22ce5	0xfd6ece8	0x2de6f79	0xab859d1	0x9fbbcfb	0x4f378a
56	0x96549f5	0x5e909b2	0xcd1638f	0x7ed9156	0x95fcf36	0xa455e43	0xd5126e

A–12. Block Module States During A2 – B2 Authentication (REPEATER = 0)

Sequence	LFSR 0	LFSR 1	LFSR 2	LFSR 3	SH 0	SH 1	SH 2	SH 3
Load								
1								
2								
3								
…								
47								
48								
Load	0x01e97	0x01d48	0x03d90	0x1bc60	01	01	01	01
1	0x01d2f	0x03a91	0x07b21	0x178c0	10	10	11	00
2	0x01a5f	0x03522	0x0f642	0x0f180	01	01	11	10
3	0x014be	0x02a45	0x0ec85	0x1e301	11	00	11	01
4	0x0097d	0x0148b	0x0d90a	0x1c602	11	01	10	11
5	0x012fa	0x02916	0x0b215	0x18c05	11	11	00	11
6	0x005f4	0x0122d	0x0642a	0x1180a	01	11	10	01
7	0x00be9	0x0245b	0x0c855	0x03015	10	11	01	10
8	0x017d3	0x008b6	0x090ab	0x0602b	01	10	11	00
…	…	…	…	…	…	…	…	…
49	0x01f26	0x01ba1	0x004d1	0x01eb1	01	10	01	00
50	0x01e4d	0x03742	0x009a3	0x03d62	11	01	10	00
51	0x01c9a	0x02e84	0x01346	0x07ac5	11	10	01	10
52	0x01935	0x01d09	0x0268d	0x0f58b	11	01	10	11
53	0x0126b	0x03a12	0x04d1b	0x1eb16	10	10	11	10
54	0x004d7	0x03424	0x09a37	0x1d62d	00	01	11	11
55	0x009ae	0x02849	0x0346f	0x1ac5b	00	10	01	11
56	0x0135d	0x01093	0x068df	0x158b7	00	00	11	01

Table A–13. LFSR Module States During A1 – B1 Authentication (REPEATER = 1)

Sequence	Kx	Ky	Kz	Bx	By	Bz	Output
Load	0x22fcecc	0x5309c7d	0x0000000	0xc070403	0x271c130	0x0000134	
1	0x000084c	0xf458fff	0x7f722dc	0xa5d4b70	0x9fb9989	0x9f6066d	0xbe70ee
2	0x0ed9f8a	0xb444236	0x3b62e76	0x614bd63	0x1d52893	0x2e71e83	0x102031
3	0x70ef0ef	0x9aa103f	0x8aa659d	0xe37a3ed	0x6e17dcd	0x861926f	0xff57a7
4	0xc8f3da5	0x8bbb85f	0x58047e6	0x0ed0c42	0xe3299e6	0xb4a6b97	0xb351be
5	0x6b68710	0x1826042	0xc20a675	0x7e45c24	0xc398d39	0xa08a2f8	0x785499
6	0xd4c9cf4	0x0014506	0x6c11733	0x1395270	0xf15cafa	0x1e1176c	0xe2b59c
7	0x2ff2231	0x059031a	0xd84c367	0x2769c98	0x7d0946d	0x0bf1b6a	0xaaa109
8	0x1c13406	0x516f805	0x3e231f5	0xe99e086	0xde5a665	0x22dff84	0x2ce1f3
…	…	…	…	…	…	…	…
41	0x7dc29a3	0x5895932	0x26047a5	0x0755719	0x935cfbf	0xb95d7e0	0x24e15b
42	0xba7d2b0	0xf1cfeac	0x36eb45d	0x2a92c58	0x699d93d	0x0eb7293	0x87309b
43	0xdd26650	0x29e8ca4	0xbf0109c	0xfa8cac0	0x1e322dc	0x01e0bb2	0xb0f7f3
44	0xf928c5b	0xc70cecd	0xcc71bb9	0x9b0f0e5	0x89e6139	0x613ba0b	0x800977
45	0x491d801	0xf630446	0x43655f6	0x4b35863	0x06237ac	0xca3aa9e	0x4fdd1d
46	0x9281463	0x891c25b	0x2c40a10	0xd0db4ac	0x07ca5ad	0x3745ef1	0x4fd875
47	0x37ef335	0xbb8429b	0xfad91c5	0x1f0f4dc	0xcb0f7af	0x9858087	0x08d905
48	0x7bd96ba	0xee950f7	0x749f3d9	0x1d48e97	0xbc607b2	0x98d9b45	0x2247f5
Load	0x1d48e97	0xbc607b2	0x98d9b45	0xc070403	0x271c130	0x0000134	
1	0x371f49a	0x53afa6d	0x1648023	0x7f3108b	0x9fb9989	0x9f6066d	0x7ccafe
2	0x3271b4e	0x7c7ab77	0x269baee	0x879d9dd	0x1d52893	0x40ef6b9	0xf3e3bb
3	0x76928cd	0x3c0c41e	0x3ddb777	0x56aff98	0x80f974f	0x6ed848c	0x387685
4	0xcb38955	0x45f4b5a	0x44b09f0	0x84f827e	0xd8421d6	0x756a06d	0xcac318
5	0x7e05951	0x7b4b7ce	0x77213e7	0x8a65060	0x41308c0	0x172f316	0xbba079
6	0xf43b422	0x63ba5f7	0x15664df	0xa546f91	0x6e221b2	0x5b52502	0x15723b
7	0x02539f7	0x43b1c83	0xc6fba6e	0x8c6d674	0x4234c5a	0x64478ee	0x6d962d
8	0xf69c689	0xc41f360	0x04591c2	0xde7e4f0	0x803e2ed	0x532a599	0xa8de7e
…	…	…	…	…	…	…	…
49	0x7bf9fa7	0x1a284c6	0x739fd87	0x461f4a1	0xf717fe1	0x32b1a29	0xf7f563
50	0xd779ca4	0xef3a891	0x60780be	0xaa1ce2e	0x9754a31	0x0b0bbfc	0x664b98
51	0x900446f	0x80e9401	0xc3bf1fb	0xfebca94	0x4e6d371	0xe3b1944	0xd1dc3b
52	0x83b3ab9	0x66e50bb	0xe8c834c	0xea84947	0x53787ed	0xd15995d	0xc6c650
53	0xd17e23d	0xfd8c2ef	0x618168a	0x5091ea5	0x9e567a1	0x6b37e87	0x49372d
54	0x6cc9afa	0x560a656	0x3dd0e24	0xc214d9d	0x71be498	0x3040f5e	0x0e3dce
55	0xcb2c184	0xdc614f7	0x5d3ee63	0x0bba955	0xaa48398	0xaf781e4	0x6438bb
56	0x692a85f	0xde2a833	0xff731e2	0xafa1960	0xc8a6055	0xbcc4562	0x85e78f

Table A–14. Block Module States During A1 – B1 Authentication (REPEATER = 1)

Sequence	LFSR 0	LFSR 1	LFSR 2	LFSR 3	SH 0	SH 1	SH 2	SH 3
Load								
1								
2								
3								
…								
47								
48								
Load	0x01aaa	0x0154c	0x0278b	0x0b789	01	01	01	01
1	0x01555	0x02a99	0x04f17	0x16f13	10	10	10	11
2	0x00aaa	0x01533	0x09e2f	0x0de26	01	01	11	01
3	0x01554	0x02a66	0x03c5e	0x1bc4c	00	10	11	10
4	0x00aa8	0x014cc	0x078bd	0x17898	00	00	11	11
5	0x01550	0x02999	0x0f17a	0x0f131	00	00	10	11
6	0x00aa0	0x01332	0x0e2f4	0x1e262	01	00	00	11
7	0x01540	0x02664	0x0c5e9	0x1c4c4	10	10	00	10
8	0x00a81	0x00cc9	0x08bd2	0x18989	01	01	01	00
…	…	…	…	…	…	…	…	…
49	0x00c45	0x01b77	0x08130	0x052e4	00	01	00	01
50	0x0188b	0x036ef	0x00260	0x0a5c9	01	00	01	10
51	0x01117	0x02dde	0x004c1	0x14b93	00	01	00	01
52	0x0022f	0x01bbc	0x00982	0x09727	01	00	01	00
53	0x0045e	0x03779	0x01304	0x12e4f	11	00	10	00
54	0x008bc	0x02ef2	0x02608	0x05c9e	10	10	01	00
55	0x01179	0x01de5	0x04c10	0x0b93d	01	00	10	10
56	0x002f3	0x03bcb	0x09821	0x1727b	10	00	00	11

Table A–15. LFSR Module States During A1 – B2 Authentication (REPEATER = 1)

Sequence	Kx	Ky	Kz	Bx	By	Bz	Output
Load	0x089c923	0xf6aee46	0x0000000	0xad10fe5	0x5e62a53	0x0000144	
1	0x000ace8	0x2bbe222	0xa84ba32	0xf8ee8f0	0x5d68545	0x649180e	0xb24463
2	0xbe2db4d	0xced43e8	0x6cf4c5d	0x5e52253	0x8d0daa0	0xfbde86b	0x1fa15f
3	0x59aaa16	0x420acae	0x948ddf1	0xe59bdcc	0xd7951b1	0x092c03c	0x787a32
4	0x6716e27	0xc71eabf	0x728216a	0x84926be	0xcaad80c	0xec3a8a5	0xf27cef
5	0x2b8be74	0xc7b7cd8	0x1896efd	0x7d66727	0x5c571f8	0x8069a85	0x88a3ad
6	0x417f923	0xf719e90	0xd5c1459	0x76bb30d	0x5333af4	0xa18c913	0xd01f1b
7	0x6c1faa9	0xf7175fd	0x50bb276	0xd91bfa4	0x1a7d561	0x456e67c	0xdc6f7c
8	0x90a1447	0xad4dd26	0x59afdb6	0xa59b390	0x1794cd7	0x3453dff	0x9276f6
…	…	…	…	…	…	…	…
41	0x456a8de	0x218a73d	0xefe8143	0x4705e66	0xa0ab473	0x77d249d	0x40cba0
42	0x5bb75c0	0x9e32509	0xcd4d66f	0x4d4a0e2	0x02b580f	0x2b49a78	0x1a3445
43	0x692b31d	0x40c7b06	0xeb692c8	0x0d36661	0x3a20c13	0x8cf85c3	0x02f684
44	0x4ac7e44	0x584dad4	0x2606dca	0xb39da54	0xc47d057	0xdca5d5d	0xf7ef88
45	0x995c381	0xe782e99	0x500545a	0x0710574	0x54607a7	0x42e8a1e	0xf1a5cc
46	0x2a39ef6	0xb3509f9	0xbd26dfe	0x284e17f	0x439d9e4	0x4dd18ce	0x23402b
47	0xe937d30	0x7910780	0x03575d7	0xdf9ad7d	0x3c7791a	0x6ddd61f	0x95dc64
48	0xb9af224	0x04c8a5f	0x49c96b1	0x754caaa	0xb7894f1	0xfcce020	0xcdaa1d
Load	0x754caaa	0xb7894f1	0xfcce020	0xad10fe5	0x5e62a53	0x0000144	
1	0x1cfb5dd	0xce2b088	0x2eec032	0x93dabe7	0x5d68545	0x649180e	0x4bbc20
2	0xfa0338f	0xdd9d11d	0x26e8f45	0x91d34c5	0x8d0daa0	0xa42f29f	0x0c1351
3	0x11ffc1e	0xd8fc06f	0x846a9c2	0x575d169	0x5f1d290	0xd8d250e	0x14f5d7
4	0x004ea3a	0xb8ae70e	0x00f25c3	0x807911a	0x442cc5a	0x1f6d6e5	0xa0c9b8
5	0xffd1f46	0x63fcef9	0x59e2583	0x0965cff	0x912f65a	0x9fad256	0x28067a
6	0x86aa27f	0x1bfc986	0x7559055	0xd307ffb	0x11af6d1	0x4d14ec4	0xa73184
7	0xe438d81	0x2f72c2a	0x065bebb	0x2c48a34	0x00ed16b	0xb2430a6	0x62d500
8	0xdc88b2a	0x1b83e3e	0xc719f35	0x3530afd	0x2435827	0x62edd40	0xe4b982
…	…	…	…	…	…	…	…
49	0x2afbb7	0x6e1ecc7	0x2126ced	0xa7ac884	0x0a7c511	0x278da73	0x3c52476
50	0xbf090b	0x9b7983d	0xd61a93c	0x560de7f	0x47467e0	0xf5c27f1	0x56257fb
51	0x6fffc7	0x1848c4a	0x6946104	0x97436c5	0x0ac81df	0xac47979	0x84c004f
52	0x8f5af2	0xb9ff03e	0xfafd4f8	0x030217e	0xb570368	0x4a63c44	0x8c9e6ff
53	0xda43d6	0x031fbfa	0x20c4236	0x7181797	0xa99940c	0x810cdc7	0x6eb5e1a
54	0xc409c6	0xc67ef5d	0xdee5ece	0xb3296c2	0xd4f4edd	0xe33bd04	0xcbee012
55	0x3f82c9	0xa8244d2	0x3aef4b0	0x5c7f3ad	0x7eb9d86	0xa72a66e	0x5527b8c
56	0x6856e1	0xe3a9d07	0xce2e311	0xa20cd64	0xe15b166	0x74e9482	0x6a048e0

Table A–16. Block Module States During A1 – B2 Authentication (REPEATER = 1)

Sequence	LFSR 0	LFSR 1	LFSR 2	LFSR 3	SH 0	SH 1	SH 2	SH 3
Load								
1								
2								
3								
…								
47								
48								
Load	0x01bb1	0x012f3	0x00be0	0x1fe37	01	01	01	01
1	0x01763	0x025e7	0x017c1	0x1fc6e	10	11	00	10
2	0x00ec7	0x00bce	0x02f82	0x1f8dd	01	11	10	01
3	0x01d8f	0x0179d	0x05f04	0x1f1bb	00	11	11	00
4	0x01b1f	0x02f3b	0x0be08	0x1e377	10	01	11	01
5	0x0163f	0x01e77	0x07c10	0x1c6ef	01	10	11	11
6	0x00c7f	0x03cee	0x0f821	0x18ddf	11	00	11	11
7	0x018fe	0x039dd	0x0f043	0x11bbf	10	01	10	11
8	0x011fc	0x033bb	0x0e087	0x0377e	11	00	01	11
…	…	…	…	…	…	…	…	…
49	0x00d13	0x03c38	0x09f02	0x16ea7	00	11	11	01
50	0x01a27	0x03870	0x03e04	0x0dd4f	00	10	11	10
51	0x0144f	0x030e1	0x07c09	0x1ba9e	01	00	11	11
52	0x0089e	0x021c3	0x0f812	0x1753c	11	00	10	11
53	0x0113d	0x00386	0x0f024	0x0ea78	01	10	00	11
54	0x0027b	0x0070d	0x0e048	0x1d4f0	11	01	00	01
55	0x004f7	0x00e1b	0x0c091	0x1a9e0	10	10	01	10
56	0x009ee	0x01c37	0x08122	0x153c1	01	00	11	01

Table A-17. LFSR Module States During A2 – B1 Authentication (REPEATER = 1)

Sequence	Kx	Ky	Kz	Bx	By	Bz	Output
Load	0xbec1205	0x4afe34d	0x0000000	0x1c66e07	0xbec2bb0	0x0000183	
1	0x888e2ea	0x414b444	0x97a0589	0xf087578	0x2c4a233	0x610f071	0x001a91
2	0x4625e41	0xcd48c5f	0x3a77722	0xf95ef49	0x467d200	0xb71a3c5	0x99774a
3	0xc9402d8	0x5ce2e8b	0x2d46dd1	0x6108d09	0xccc49d1	0x9c06127	0xf1c0f1
4	0x9f4f7b0	0x4c9fc33	0x7975e63	0xe5ed94e	0xa6cfafe	0x2632b27	0x3ce478
5	0xa52c6b9	0x0ab1bea	0x3f59b80	0xc0165ea	0xb0c5a07	0x52300a1	0x8091f8
6	0xe828e8c	0x1f4fe28	0xf9ae9ca	0x7849ad5	0x5c4c5dc	0x8ba6a57	0xa1cf90
7	0x3d9656f	0x9313d6c	0xd525839	0xb882808	0xaf4cb4e	0xe0eb86a	0xd6d500
8	0x0b5904f	0xe168c0e	0x8549a6c	0x720eb74	0xe3f004a	0xbab4d22	0x1000c1
…	…	…	…	…	…	…	…
41	0xf907779	0x8add56d	0xa2bf28b	0x170a7c3	0x35dc444	0x8e8c9fa	0xa24983
42	0xbb149e8	0x34b44fe	0xe899a28	0x298b048	0x32b7742	0xd005cfd	0xea1835
43	0x852bc22	0x30c541b	0x4ba8ad0	0x3eae65f	0x158d372	0xcadc45a	0xe1162f
44	0xe0dcc66	0x3380692	0x2f59c16	0xe406ae7	0x605aa2c	0x37ac1ab	0x9e5a09
45	0xbd69a67	0x11e9f3b	0xb0d15db	0xedd1223	0x38397e2	0xa9aeec0	0xb5955f
46	0x992aba4	0x79ccd6c	0x374d0da	0x50ca3ca	0x24fe7c5	0xab2ac15	0x8680ef
47	0x02d7a9c	0x69e0827	0x75c491b	0xc2e075e	0x27ef684	0x5569487	0x2f26b1
48	0x28d5897	0x4f55c34	0x1bf2686	0x12f3bb1	0xfe3717c	0x4903692	0x490497
Load	0x12f3bb1	0xfe3717c	0x4903692	0x1c66e07	0xbec2bb0	0x0000183	
1	0xa4b6650	0x0726307	0x51cb288	0x775f7b9	0x2c4a233	0x610f071	0xc6a91b
2	0xb19afdf	0x140ae14	0x6402f81	0xe318db4	0x467d200	0xbf592b0	0x5dcbb5
3	0x9159d90	0x4dec573	0xca5821f	0xc90434c	0x333bc3a	0x8fd699e	0x93cd20
4	0x958e6ac	0x17a4c19	0x95d7367	0xf18d3a1	0xa0182d7	0x0608db9	0xa81d43
5	0x5637028	0x7fd4c2b	0x235d32a	0x012244a	0x760a344	0x856619e	0x73e788
6	0x30b4ded	0x6cf793e	0x75d7724	0x29dc723	0x363fbe6	0xc615e74	0x18faae
7	0x0be6fa2	0x96a92c7	0x013fcf0	0x40c3e38	0x693a50c	0x2c0f81f	0x429d33
8	0x302975b	0x762a198	0x0e1b7f2	0x0b403f5	0x1493775	0x0326946	0x743991
…	…	…	…	…	…	…	…
49	0xaf2d2bb	0xe13c1bf	0xd5bf725	0xa861b70	0x30baed9	0x595a054	0xaee82d
50	0xd6b547a	0xbcc8c65	0xaf1fe4b	0x5e1ed44	0x3bdcf3f	0x775ef00	0x574a8e
51	0x8e47e11	0x1a9467f	0xc074e74	0xf94ad69	0x78cca09	0x3f48c38	0x6d424b
52	0x819e9c2	0xed51704	0x9cd77e9	0x03dd484	0x3b38f11	0x9e92103	0xbcdd40
53	0x274fca5	0x50dde0a	0xe25ca16	0x462e7d7	0xa603ab6	0x48da00f	0x97536d
54	0x910b283	0x5dcf83d	0x3a4f75f	0xecacd6b	0x7c0fb7b	0x1b60ea8	0x0eee1e
55	0xea791f3	0x92b86cf	0x3be152b	0xe0f4dc5	0xd3e247e	0x6996c21	0xdd44a5
56	0xcb67cb7	0xab75038	0xf8a92f2	0x754b3d8	0x47f242a	0x5d3f58c	0x9b8bf4

Table A–18. Block Module States During A2 – B1 Authentication (REPEATER = 1)

Sequence	LFSR 0	LFSR 1	LFSR 2	LFSR 3	SH 0	SH 1	SH 2	SH 3
Load								
1								
2								
3								
…								
47								
48								
Load	0x002d0	0x0281a	0x08a38	0x0aac4	01	01	01	01
1	0x005a1	0x01034	0x01471	0x15588	00	11	00	10
2	0x00b42	0x02069	0x028e2	0x0ab11	00	10	01	00
3	0x01685	0x000d2	0x051c5	0x15623	01	00	11	00
4	0x00d0a	0x001a5	0x0a38b	0x0ac47	00	01	10	01
5	0x01a14	0x0034b	0x04716	0x1588f	01	00	11	00
6	0x01428	0x00697	0x08e2c	0x0b11e	10	00	01	10
7	0x00850	0x00d2e	0x01c58	0x1623d	01	01	00	01
8	0x010a1	0x01a5d	0x038b1	0x0c47b	11	00	01	10
…	…	…	…	…	…	…	…	…
49	0x017d1	0x002a0	0x0c549	0x10b2f	10	00	00	11
50	0x00fa2	0x00540	0x08a93	0x0165f	11	00	00	01
51	0x01f44	0x00a80	0x01526	0x02cbe	01	10	00	10
52	0x01e89	0x01501	0x02a4c	0x0597c	10	00	01	01
53	0x01d12	0x02a03	0x05498	0x0b2f8	01	00	00	10
54	0x01a24	0x01406	0x0a931	0x165f1	11	00	00	00
55	0x01449	0x0280d	0x05263	0x0cbe2	10	01	00	00
56	0x00892	0x0101a	0x0a4c6	0x197c5	01	11	00	00

Table A–19. LFSR Module States During A2 – B2 Authentication (REPEATER = 1)

Sequence	Kx	Ky	Kz	Bx	By	Bz	Output
Load	0xb8676a7	0xa423d78	0x0000000	0x406a74d	0x51f7175	0x0000103	
1	0x666e2c6	0x1fb7111	0x802f1c8	0xf7f1edb	0x6143676	0x0f40723	0xf05140
2	0x222564c	0xeacf83b	0x56392e2	0x162b14f	0xde614cf	0x787caa9	0x81c36f
3	0x3a7d9e3	0x39004ba	0x11f7a6a	0x7fa1be9	0x01d7de9	0x01b5c18	0x206e42
4	0x47614d8	0x6494d8a	0x3b4f25b	0x25cec72	0x4a836ae	0x2534ecb	0xeaf263
5	0xdb4e14e	0x845a7cc	0xbf7698d	0x4a208a3	0x30e92d8	0xa659bcf	0x84539a
6	0x9f50e9a	0x72b9f8a	0xe83d832	0xe5d510e	0x442ab7d	0x3cd4cd1	0xc822c1
7	0x3ea1bc9	0x2ef84ca	0x8b460ed	0x1b4eb4a	0xd2f25b6	0xeb1adbf	0x37ed7a
8	0x16166f2	0xaa7c2ef	0x1d92ed2	0x1b5c7a1	0x25d261d	0xf639672	0x0312ca
…	…	…	…	…	…	…	…
41	0xab9a38	0x2b7a4ee	0x76aaca6	0x990b686	0x7b9285b	0xcea3e3a	0xf0550a8
42	0x6ccddb	0x2420fda	0xc71cbcb	0xd3a43cf	0xaca9532	0xf5455b6	0xd465e50
43	0x0532d5	0x1b38c46	0x7b286a6	0x1d6e079	0xf25ba51	0xad5a148	0xbbb5468
44	0x59de69	0x9ecc174	0xa97266e	0xa162b3f	0x3954aab	0xc8cae06	0xe9ffa6a
45	0xa81811	0x039d3b7	0x039e9b4	0xbc7dd68	0x76e0d88	0xf667013	0x5ca7484
46	0x1d33c5	0x5096513	0xc3ac236	0x4adda17	0x96a7579	0xccfde0b	0x56352ce
47	0x1ad8fa	0xc0f65b9	0x566da3d	0x55dab36	0x6ff16c4	0x198a2d8	0x97f7aef
48	0xd23a1e	0x83f87f0	0xd6f60e1	0xb0ffacc	0x081a2d0	0xaac4147	0x7734dfc
Load	0x081a2d0	0xaac4147	0x7734dfc	0x406a74d	0x51f7175	0x0000103	
1	0x4e6747	0x1ace2d1	0x14061ea	0x0c44875	0xd086746	0x6143676	0x0f40723
2	0x8d2332	0xd88d8d4	0xdb895bd	0x7e74e49	0x413ed54	0xde614cf	0xdb03edb
3	0xb7c367	0x95561d4	0xe90f704	0xfe35448	0x1cdbacf	0xcd1bfeb	0xbe705ef
4	0x1c6484	0x6aabee2	0xeb64c24	0xb674c2a	0xef4f673	0xd302546	0x75b8516
5	0xb37177	0xfe3250b	0xb039351	0x4a14ff3	0x5a879c9	0xd849947	0xa65f3bb
6	0xe41787	0x7a6f7cc	0xfbd0e84	0xce6bee1	0x0ad85e1	0x7a6282a	0x7f78db0
7	0x85be3b	0x581bf9a	0xf637058	0x06205c2	0x0ff292e	0x7d65bcc	0x84473cb
8	0x5857cf	0x662ea9c	0x99bf90a	0x290e00f	0xbad8a31	0x94d72cc	0xb929192
…	…	…	…	…	…	…	…
49	0x68a55fc	0x5bc6412	0x5ca2595	0x14cc21e	0x30c7bd6	0xb826f67	0x06a265
50	0xb7cd0f6	0x33813a4	0x7b3e868	0x78c9a94	0x94e586f	0x1ea87f3	0x18c4db
51	0x3cb03ff	0xcb86820	0x7fa96de	0x71c1620	0x7c602e4	0x60688eb	0xc9abf0
52	0x1fee845	0x0a02783	0x371bc65	0x7d3cf2c	0xcf8006d	0x3206d1e	0xb00bfa
53	0x8b4c9c9	0x8c51ea6	0xd91c1db	0xec51ba3	0x5652523	0x36ba88d	0xb238b5
54	0xb5a6da8	0x7caf32e	0x1724577	0x1a1a940	0xf96eb52	0x8929566	0x1c7ad3
55	0x8bde531	0xcbd6c1e	0x0f35c36	0xc66fea6	0x0c3c692	0x6561bba	0x79cdd1
56	0x6138d30	0x09b02ea	0x3d45fab	0x81c0f48	0xaa5211b	0xbc2973b	0x30b266

Table A–20. Block Module States During A2 – B2 Authentication (REPEATER = 1)

Appendix B. Confidentiality and Integrity of Values

Table B-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 Table B-1 as requiring integrity.

Value	Size (Bytes)	Confidentiality Required[†]?	Integrity Required[†]?	Function
Aksv	5	No	No	Video transmitter’s Key Selection Vector
An	8	No	Yes[*]	Pseudo-random value sent to video receiver/repeater by transmitter
Bksv	5	No	Yes[*]	Video receiver/repeater’s Key Selection Vector
Km,Km'	7	Yes	Yes	Secret value generated by video transmitter and receiver/repeater during authentication
Ks,Ks'	84 bits	Yes	Yes	Secret session key
K_i, K_i'	84 bits	Yes	Yes	Secret frame key
Akeys	280	Yes	Yes	Video transmitter’s device keys
Bkeys	280	Yes	Yes	Video receiver/repeater’s device keys
M_i, M_i'	8	Yes	Yes	Integrity verification key and HDCP cipher initialization value
r_i, r_i'	2	No	No	HDCP Cipher outputs during frame key calculations. Every 128th output becomes the video transmitter and receiver link synchronization verification value
R_i, R_i'	2	No	No	Video transmitter and receiver link synchronization verification values
REPEATER	1 bit	No	Yes	Video repeater capability status bit
MAX_CASCADE_EXCEEDED	1 bit	No	Yes	Video repeater topology error status bit
MAX_DEVS_EXCEEDED	1 bit	No	Yes	Video repeater topology error status bit
DEV_COUNT	7 bits	No	Yes	Video repeater topology status bit
DEPTH	3 bits	No	Yes	Video repeater topology status bit
V	20	No	No	KSV list integrity value generated by video repeater
V'	20	Yes	Yes	KSV list integrity verification value generated by video transmitter
KSV List	Varies	No	Yes	List of downstream KSV gathered by video repeater devices
Bx, By, Bx, Kx, Ky, Kz	28 bits	Yes	Yes	Internal HDCP cipher values
L[1]	128	No	Yes	Digital Content Protection LLC DSS Public Key

Table B–1 Confidentiality and Integrity of Values

[†] According to the robustness rules in the HDCP Adopter’s License.
[*] Only within the video transmitter

ISBN 0-9675129-4-8