SD Specifications

Part A1

Advanced Security SD Extension Simplified Specification

Version 2.00

May 18, 2010

Technical Committee
SD Card Association

Revision History

Date	Version	Changes compared to previous issue
May 18, 2010	2.00	Simplified Specification initial release.

Conditions for publication

Publisher and Copyright Holder:

SD Card Association
2400 Camino Ramon, Suite 375
San Ramon, CA 94583 USA
Telephone: +1 (925) 275-6615,
Fax: +1 (925) 886-4870
E-mail: office@sdcard.org

Disclaimers:

The information contained herein is presented only as a standard specification for SD Card and SD Host/Ancillary products. No responsibility is assumed by SD Card Association for any damages, any infringements of patents or other right of the third parties, which may result from its use. No license is granted by implication or otherwise under any patent or rights of SD Card Association or others.

Conventions Used in This Document

Naming Conventions

- Some terms are capitalized to distinguish their definition from their common English meaning. Words not capitalized have their common English meaning.

Numbers and Number Bases

- Hexadecimal numbers are written with a lower case "h" suffix, e.g., FFFFh and 80h.
- Binary numbers are written with a lower case "b" suffix (e.g., 10b).
- Binary numbers larger than four digits are written with a space dividing each group of four digits, as in 1000 0101 0010b.
- All other numbers are decimal.

Key Words

- May: Indicates flexibility of choice with no implied recommendation or requirement.
- Shall: Indicates a mandatory requirement. Designers shall implement such mandatory requirements to ensure interchangeability and to claim conformance with the specification.
- Should: Indicates a strong recommendation but not a mandatory requirement. Designers should give strong consideration to such recommendations, but there is still a choice in implementation.

Application Notes

Some sections of this document provide guidance to the host implementers as follows:

Application Note: This is an example of an application note.

Table of Content

Table of Figures

Table of Tables

1. General Description

1.1 Scope

This document provides the foundation of additional (to the existing copy protection system based on the CPRM technology) Security Systems to be optionally implemented on SD cards. The SD cards with the additional Security Systems are called Advanced Security SD (ASSD) cards.

A Security System is a set of card commands performing security related operations (i.e. authentication, enforcing access control policies to assets stored on the card and providing data confidentiality and non repudiation services). The definition of a Security Systems will typically include the commands format and the associated card behavior. The assumption is that the Security Systems specification is “form factor independent” in nature.

This document serves as an extension to the SD card Physical Layer Specification and provides all the definitions required to transfer Security System specific command packets from the ASSD enabled host to the ASSD card, and vice versa. The protocol extension is defined over the existing mechanisms of the SD card, making it an easy add-on. The extension is available in both, SD and SPI modes.

1.2 The ASSD Security Platform

The security market tends to be very fragmented and the traditional way of doing business is creating vertical markets using application specific Security Systems. Security standards are slowly emerging, however, currently there is no clear market direction. As an example, the legacy Smartcard market is using the ISO-7816 security system while the next generation Smartcard will allow an HTTPS based protocols. The availability of security features on mass storage devices is creating new standards based on SCSI protocols, etc…

The ASSD card, therefore, is defined as a security platform which may implement several Security Systems. The ASSD platform is designed as an extension of the legacy, CPRM based, security system and supports up to 15 additional security systems plus a proprietary test channel.

This part was removed from the simplified specification.

An ASSD card may optionally support any subset of the ASSD security systems and associated security applications. An ASSD enabled host shall be designed to enable security applications to query the ASSD card for available, supported, Security Systems, activate and use the appropriate Security System for the application or, reject the card if no appropriate Security Systems is found.

Security System specifications as well as Application specifications for the ASSD card will be independently added to the SDA specification library as they are created. Any application spec will define the Security System it is using. Host application developers will have to make sure the ASSD card inserted in the device uses the security specification expected by the application.

An important characteristic of the ASSD card (unlike a traditional Smartcard) is the availability of significantly larger non-volatile memory for data storage. Therefore, in addition to the Security System command transfer protocol, this document describes an interface to transfer data securely between the host and the memory area of the card. This feature may optionally be used by any Security System implemented on the card.

Figure 1-1 : was removed from the simplified specification.

1.3 ASSD Authorized Security Systems

This part was removed from the simplified specification.
Table 1-1 : was removed from the simplified specification.

1.4 Backwards Compatibility

This part was removed from the simplified specification.
Figure 1-2 : was removed from the simplified specification.

2. Functional Description

2.1 Switching to the ASSD Mode

The ASSD protocol is implemented in a dedicated SD command space, using the SD switch function. The ASSD host has to use the SD SWITCH mechanism to query the card, find out if the ASSD feature is supported and set the card into the right mode.

Starting revision 2.0, the ASSD extension specification introduced a conceptual change from a single security system card (as was defined in version 1.1 of the specification) to a multiple Security systems card. In order to maintain backward and forward compatibility, as well as proper version detection and communication, ASSD cards will use two switch codes.

The switch codes defined for ASSD cards are (refer to SD physical layer specification rev 2.0, Table 4-9, section 4.3.10) both allocated in the “Command System” group (group 0x2):

ASSD_1.1 (0x1)

ASSD_2.0 (0x4)

Legacy hosts and cards compliant with revision 1.1 (or older) of the extension specification will continue and use the ASSD_1.1 switch code while ASSD cards and hosts which conform to revision 2.0 (or a newer one) will use the ASSD_2.0 switch code. ASSD V1.1 (previously titled Mc-EX 1.1) supports only one ISO-7816 based security system as defined in the previous spec. It is implicitly selected and activated as soon as the card is switched into ASSD mode. In this version of the spec the card may support up to 16 different security systems and the host shall explicitly select and activate one of them.

This part was removed from the simplified specification.

Once the card is switched to the ASSD mode, the following three registers and six commands are made available to the host:

- ASSD Properties Register
- ASSD Status Register
- ASSD Random Number Register

- READ_SEC_CMD command
- WRITE_SEC_CMD command
- SEND_PSI command
- CONTROL_ASSD_SYSTEM command
- DIRECT_SECURE_READ command
- DIRECT_SECURE_WRITE command

2.2 Protected Memory

ASSD card may have large non-volatile memory areas which are protected by the ASSD Security System implemented on the card. The protected memory is managed only by the ASSD security system and shall only be accessed either by the security system commands or (optionally) by using the direct secure memory access commands defined in this specification.

This part was removed from the simplified specification.

2.2.1 Properties of the Protected Memory

This part was removed from the simplified specification.

2.2.2 Security Requirement of the Protected Memory

This part was removed from the simplified specification.

2.2.3 Direct Access to the Protected Memory

The protected area may be accessed either indirectly through any of the available security system commands or directly (if this optional feature is supported by the card) using the direct secure read and write commands described in this document.

A typical usage of this direct memory access commands would be to read or write memory spaces which are allocated to files (e.g. in the file system managed by the ASSD security system) for an ASSD applications. These files may be assigned an Area # by the ASSD security system.. Each file has a space of unique size. This document does not define the detail of the file system structure.

To gain access to files via the direct memory access commands, the host should request an Area# as a file handle from the ASSD security system. The host can then access a file in the protected area using the Area# as a file handle.

The session keys which are used in Direct Secure Memory access commands (see 2.4.5 and 2.4.6) may be shared by the host and the ASSD application before Direct Memory Access sequence is initiated. The protocol of the key exchange is not defined in this document. It is dependent on the ASSD application, so the host and the ASSD application may, as an example, share or update the session keys on each Direct Secure Memory access, or may use the same session keys at all times. The management of the session keys (frequency of the key exchange, etc) is the responsibility of the ASSD application.

2.3 Registers

There are three registers defined by the ASSD extension spec. The registers altogether are referred to as the PSI (**P**roperties and **S**tatus **I**nformation). Any of the registers can be read using the SEND_PSI command. The relevant register is selected by setting the [2:0] bits of the command argument.
The registers include information related to the physical aspects of the card as well as to the secure interface, and the SD protocol.

Table 2-1 : PSI Register ID

ID	Register Name	Register Abbreviation
0	ASSD Status Register	ASSD-SR
1,2,3	Reserved for future use	
4	ASSD Properties Register	ASSD-PR
5	Reserved for future use	
6	ASSD Random Number Register	ASSD-RNR
7	Reserved for future use	

2.3.1 ASSD Status Register (ASSD-SR)

This register is 32 bytes long. The ASSD-SR contains error statuses that are set by events in the card operations and are cleared when the register is read by the host.

Table 2-2 : ASSD Status Register

Name	Field	Width	Bit Slice
ASSD State	ASSD_STATE	8	[255:248]
ASSD Error	ASSD_ERR_STATE	8	[247:240]
ASSD Security System Error	ASSD_SEC_SYS_ERR	1	[239]
Reserved (Shall be set to 0)		7	[238:232]
Protected Memory Area State	PMEM_STATE	8	[231:224]
Direct Memory Access Authentication Algorithm	AUTH_ALG	8	[223:216]
Direct memory access Encryption Algorithm	ENC_ALG	8	[215:208]
Active Security System	ACTIVE_SEC_SYSTEM	8	[207:200]
Secure Token Protocol	SEC_TOKEN_PROT	8	[199:192]
Block count for READ_SEC_CMD	READ_BLOCK_COUNT	16	[191:176]
Reserved (Shall be set to 0)		176	[175:0]

ASSD_STATE

This byte indicates the current state of the ASSD Security System (refer to Table 2-3).

Table 2-3 : ASSD States

Code	State	Description
0	Idle	No secure sequence[1] is currently active
1	Secure Command in Progress	A WRITE_SEC_CMD has been received and it is being processed
2	Secure Command Completed	The last received WRITE_SEC_CMD has been completed and the host can send the READ_SEC_CMD
3	Secure Command Aborted	A legacy SD command was issued during a secure commands sequence, and the card had to abort the secure sequence to be able to execute the legacy command
4-255	Reserved	

ASSD_ERR_STATE

This byte indicates the current error state of the ASSD. This is cleared only when the ASSD-SR is read. And then, this field is set to "NO_ERROR". If another command is issued without reading the register, it is overwritten with the new error state.

Table 2-4 : ASSD Error

Code	State	Description
0	NO_ERROR	No error has occurred.
1	AUTH_ERROR	Authentication has failed when using the direct secure memory access command
2	AREA_NOT_FOUND	The area specified by Area# is wrong.
3	RANGE_OVER	Block offset or block count, of a direct secure memory access command, is wrong.
4	CONDITION_ERROR	Condition is not satisfied.
5-255	Reserved	

ASSD SEC_SYS_ERR

This bit is set when there is a general security system error (a WRITE_SEC_CMD failed) and no secure token level response is available. The card response to a subsequent READ_SEC_CMD, if issued, depends on the Secure Token Protocol the security system is using. Refer to section 3 for more details. This bit is set to '0' when ASSD-SR is read.

PMEM_STATE

This byte is set when a valid WRITE_SEC_CMD (*param*) is processed and indicates the current selected Area# (1~255) or IDLE state (0).

This is cleared when DIRECT_SECURE_READ or DIRECT_SECURE_WRITE has finished.

[1] See Section 2.5 for the description of secure commands sequences.

Table 2-5 : Protected Memory Area State

Code	State	Description
0	IDLE	No Area# is selected.
1~255	AREA_SELECTED	These values indicate the current selected Area# In this state, a host is allowed to issue DIRECT_SECURE_READ or DIRECT_SECURE_WRITE;, on the other hand, a host is NOT allowed to issue WRITE_SEC_CMD (param) (the card sets CONDITION_ERROR).

AUTH_ALG

This byte indicates the algorithm used in the authentication process for accessing the protected memory Area specified by "Protected Area State" using DIRECT_SECURE_READ (CMD50) or DIRECT_SECURE_WRITE (CMD57).

Table 2-6 : Direct Memory Access Authentication Algorithm

Code	Description
0	3-key Triple DES. MAC algorithm is ISO9797-1 algorithm 1. Most significant 4 bytes are used as the authentication code. The data string to be input to the MAC algorithm is '|PMEM__AUTH_CHALLE NGE | DATA_PAYLOAD of previous WRITE_SEC_CMD in param-mode'|.
1	128bits key AES MAC algorithm is ISO9797-1 algorithm 1. Most significant 4bytes are used as the authentication code. The data string to be input to the algorithm is '|PMEM_AUTH_CHALLENGE | DATA_PAYLOAD of previous WRITE_SEC_CMD in param-mode|'.
2-254	Reserved
255	No authentication

ENC_ALG

This byte indicates the algorithm used in the encryption of the data of DIRECT_SECURE_READ (CMD50) or DIRECT_SECURE_WRITE (CMD57).

Table 2-7 : Direct Memory Access Encryption Algorithm

Code	Description
0	3-key Triple DES- CBC and Initial Vector is '00 … 00'. CBC process is continued between continuous blocks.
1	128bits key AES and Initial Vector is '00 … 00'. CBC process is continued between continuous blocks.
2-254	Reserved
255	The data is not encrypted.

ACTIVE_SEC_SYSTEM

An 8 bit unsigned integer. This field indicates the currently active security system. An ASSD card will always have one of its security systems selected and activated (refer to section 2.5.1 for more details).

Table 2-8 : was removed from the simplified specification.

SEC_TOKEN_PROT

An 8 bit unsigned integer. This field indicates the Secure Token protocol used by the active security system (refer to section 3 for more details).

Table 2-9 : Secure Token Protocol

Code	Description
0	APDU based protocol
1	Data Block based protocol
2-255	Reserved for future use

READ_BLOCK_COUNT

A 16 bit unsigned integer. This field is valid only when the security system is using the Data Block protocol for transferring secure Tokens and ASSD_STATE is Secure_command_completed. When valid the field will indicate the number of data blocks that can be read out of the card. When the field in not valid, it shall be set to 0.

2.3.2 ASSD Properties Register (ASSD-PR)

This register is 32 bytes long. The ASSD-PR reflects the capabilities of the card. The content of the register is card dependent and cannot be modified by the host.

When an ASSD security system supports application loading (e.g. SW modules to be executed on the card, similar to the JavaCard concept), the content of this register may be application specific and may change as a result of loading of a new application. Therefore, ASSD host should query this register when new application is selected.

Table 2-10 : ASSD Properties Register

Name	Field	Symbol	Width	Bit Slice
Secure Read Latency	SEC_READ_LATENCY	Nsrl	8	[255:248]
Secure Write Latency	SEC_WRITE_LATENCY	Nswb	8	[247:240]
ASSD protocol version	ASSD_VERSION	Never	8	[239:232]
Contact-less Support	CL_SUPPORT	N_{CL}	15	[231:217]
Direct Access to Protected memory	PMEM_SUPPORT	N_{NVM}	1	[216:]
Protected Memory Read Access Time	PMEM_RD_TIME	N_{EXR}	8	[215:208]
Protected Memory Write Busy Time	PMEM_WR_TIME	N_{EXW}	8	[207:200]
WRITE_SEC_CMD Bus Busy	WR_SEC_BUS_BUSY	N_{WSB}	8	[199:192]
Supported Authentication Algorithm for Protected Memory Direct access	SUP_AUTH_ALG	N_{SAA}	16	[191:176]
Supported Encryption Algorithm for Protected Memory Direct Access	SUP_ENC_ALG	N_{SEA}	16	[175:160]
Security System	ASSD_SEC_SYS		16	[159:144]
Reserved (Shall be set to 0)			144	[143:0]

SEC_READ_LATENCY

An 8 bit, unsigned integer. This byte defines the maximum time allowed for the card to output the data of a READ_SEC_CMD command. As in a standard SD read operation this time is measured from the end bit of the response to the READ_SEC_CMD command to the start bit of the first data block. The value is given in units of 250ms. A value of [Dec]10, as an example, means maximum read latency of 2.5 seconds. Refer to section 2.5 for more details.

SEC_WRITE_LATENCY

An 8 bit, unsigned integer. This byte defines the maximum time allowed for the card to execute a WRITE_SEC_CMD command and set ASSD_STATE to Secure-command-completed. As in a standard SD write operation this time is measured from the end bit of the CRC response of the last data block. The value is given in units of 250ms. A value of [Dec]10, as an example, means maximum read latency of 2.5 seconds. If this field is set to 0, the maximum latency time is not defined. Refer to section 2.5 for more details.

ASSD_VERSION

An 8 bit, unsigned integer. This byte indicates the ASSD protocol version number supported by the card. The number means:

Table 2-11 : ASSD Version

Code	Description
0	This code is reserved for Mc-EX1.0 and means that this card conforms to the specification part A1 ver.1.0
1	This code is reserved for Mc-EX1.1 and means that this card conforms to the specification part A1 ver.1.1
2	This card conforms to the specification part A1 ver.2.0
3-255	Reserved for future use

CL_SUPPORT

A set of 15 bits. This field indicates which contactless interface is supported by the card.

Table 2-12 : CL_SUPPORT

Bit Position	Description
[14]	If set to '1': this card supports ISO/IEC 14443 Type A.
[13]	IF set to '1': this card supports ISO/IEC 14443 Type B.
[12:0]	Reserved for future use. Shall be set to '0'

Examples:

If the card supports only ISO/IEC 14443 Type A, CL_SUPPORT is set to "1000 0000 0000 000b".
If the card supports ISO/IEC 14443 Type A and Type B, CL_SUPPORT is set to "1100 0000 0000 000b".
A value of "0000 0000 0000 000b" indicates no CL support

Protected Memory Direct Access SUPPORT

A single bit field; This field indicates whether the card supports direct Protected Memory Access commands. Supporting of protected memory access includes: WRITE_SEC_CMD in param-mode, DIRECT_SECURE_READ, DIRECT_SECURE_WRITE, and ASSD-RNR register

.

Table 2-13 : Support of Protected Memory Direct Access

Bit Position	Description
[0]	If set to '1': the card supports Protected Memory Direct Access commands

Protected Memory_RD_TIME

An 8 bit, unsigned integer. This field defines the maximum read access time in DIRECT_SECURE_READ.

The value is given in units of 100ms. This value defines the maximum delay between the end bit of the DIRECT_SECURE_READ command and the start bit of the data block.

If direct memory access is not supported, this field shall be set to 0.

Protected Memory_WR_TIME

An 8 bit, unsigned integer. This field defines the maximum busy time from the end bit of the CRC response of a data block of DIRECT_SECURE_WRITE. The value is given in units of 250ms.

If direct memory access is not supported, this field shall be set to 0.

WR_SEC_BUS_BUSY

An 8 bit unsigned integer. This field indicates the maximum busy time from the end bit of the CRC response of a data block of WRITE_SEC_CMD. The value is given in units of 250ms. If this field is set to 0, the maximum busy time is not defined. Refer to section 2.5 for more details.

SUP_AUTH_ALG

A set of 16 bits. This field indicates which algorithms the card supports for the authentication of DIRECT_SECURE_READ and DIRECT_SECURE_WRITE.

If direct memory access is not supported, this field shall be set to 0.

Table 2-14 : SUP_AUTH_ALG

Bit Position	Description
[15]	If set to '1': "ISO/IEC9797-1 alg1 with 3-key TDES " is supported
[14]	If set to '1': "ISO/IEC9797-1 alg1 with 128bit key AES " is supported
[13:0]	Reserved for future use. Shall be set to '0'

SUP_ENC_ALG

A set of 16 bits. This field indicates which algorithms the card supports for the encryption of DIRECT_SECURE_READ and DIRECT_SECURE_WRITE.

If direct memory access is not supported, this field shall be set to 0.

Table 2-15 : SUP_ENC_ALG

Bit Position	Description
[15]	If set to '1': "3-key Triple DES-CBC" is supported
[14]	If set to '1': "128bit key AES-CBC " is supported.
[13:0]	Reserved for future use. Shall be set to '0'

ASSD_SEC_SYS

A set of 16 bits. This field indicates the list of security systems implemented by this ASSD card. If a specific Security System is implemented on the card the corresponding bit will be set to '1'. Otherwise it will be set to '0'. The bit position of any security system corresponds to the integer value of the security system indicator used in the ACTIVE_SEC_SYSTEM field in the status register, and Security System Code argument in the CONTROL_ASSD_SYSTEM command.

Table 2-16 : was removed from the simplified specification.

2.3.3 ASSD Random Number Register (ASSD-RNR)

This register is 32 bytes long. The ASSD-RNR stores the random number used during the authentication algorithm required for direct memory access operations. When this register is not implemented the card will return 32 zero (0) bytes when the host issues a SEND_PSI command, addressing this register.

Table 2-17 : ASSD Random Number Register

Name	Field	Width	Bit Slice
Direct memory access Authentication Challenge	PMEM_AUTH_CHALLENGE	128	[255:128]
Reserved (Shall be set to 0)		128	[127:0]

PMEM_AUTH_CHALLENGE

This field is a 16 bytes long random number. This random number is used to generate authentication code on DIRECT_SECURE_READ and DIRECT_SECURE_WRITE. The value of this field should be changed every time ASSD-RNR is read by the host. The host should use the latest value before WRITE_SEC_CMD (param-mode).

After this field has been read by the host once, the card has the latest value and uses it to verify the authentication code in the argument of DIRECT_SECURE_READ and DIRECT_SECURE_WRITE. After DIRECT_SECURE_READ or DIRECT_SECURE_WRITE has finished, the value in the card should be cleared, and not be used. When ASSD-RNR is read again, the card will keep the new value.

2.4 Commands

While in either ASSD 1.1 or ASSD 2.0 modes, the SD card allocates the “reserved command set” (refer to SD physical layer specification rev 2.0, section 4.3.9) to the ASSD functions. Table 2-18 includes a general overview of the commands. Detailed description of the commands functional behavior and parameters are provided in the following sections.

Table 2-18 : ASSD Mode Specific Commands

CMD INDEX	CMD Name	CMD Type	CMD Arguments	RSP Type	CMD Description
CMD 34	READ_SEC_CMD	Adtc	**Block count:** specifies the number of data blocks the host expects the card to output.	R1	This command is using an SD multiple block read bus transaction. The data blocks contain a Secure Token (Refer to section 3 for more details).
CMD 35	WRITE_SEC_CMD	Adtc	**Command mode:** indicates whether “command” or “param” mode is used. **Block count:** specifies the number of data blocks the host will transfer to the card.	R1	This command is using an SD multiple block write bus transaction. The data blocks contain a Secure Token. In command mode the Secure Token defines an ASSD command while in Param mode the Secure Token defines parameters for the next direct memory access command (CMD 50 or 57)
CMD 36	SEND_PSI	Adtc	**Register ID:** defines the register to be read.	R1	The card returns a, single, 32 byte data block containing the selected PSI register
CMD 37	CONTROL_ASSD_SYSTEM	Ac	**Security System Index:** Identifies the Security system to be Selected. **Operation:** May be either select or no-operation.	R1b	This command selects and resets the ASSD security system inside the SD card.
CMD 50	DIRECT_SECURE_READ	Adtc	Authentication code	R1	This command reads blocks of data continuously from an area memory specified by Area# in a previously sent WRITE_SEC_CMD in param-mode. Data to be read shall be encrypted with the session key (S-ENC) by the card.

CMD INDEX	CMD Name	CMD Type	CMD Arguments	RSP Type	CMD Description
CMD 57	DIRECT_SECURE_WRITE	Adtc	Authentication code	R1	This command writes blocks of data continuously to a memory area specified by Area# in a previously sent. WRITE_SEC_CMD in param-mode. Data to be written shall be encrypted with the session key (S-ENC) by the host.

2.4.1 WRITE_SEC_CMD Command

The WRITE_SEC_CMD is available in ASSD mode. This command uses the multiple block write transaction of the SD bus (refer to **Figure** 2-1 for quick reference. The complete description of this bus transaction is provided in the SD physical specification).

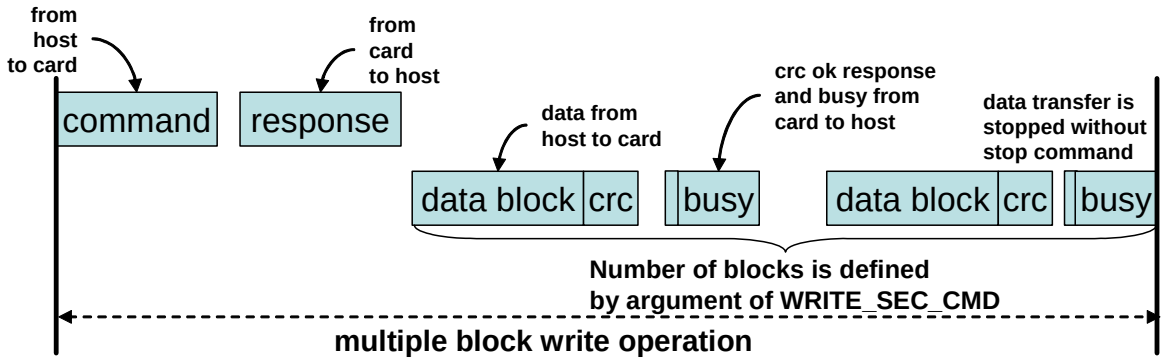

Figure 2-1 : Multiple Block Write Operation of WRITE_SEC_CMD

The WRITE_SEC_CMD command is used to transfer a Secure Token (refer to section 3 Secure Token definition) from the host to the card. This command continuously transfers blocks of data until all of the specified data blocks have been transferred. The number of data blocks to be transferred is defined in the argument of the command (refer to Table 2-19 for details).

Unlike standard WRITE_MULTIPLE_BLOCK command the usage of the STOP_TRANSMISSION command is not mandatory. The card will terminate the data transfer phase and start executing the command as soon as the specified number of data blocks is transferred. At the end of the Secure Token execution the card returns to TRAN state. The host may, however, terminate or interrupt the data transfer by issuing STOP_TRANSMISSION command. If STOP_TRANMISSION command is used the bus timing shall be identical to the standard SD write cycle.

In WRITE_SEC_CMD command, the block length shall be set to 512bytes. Before WRITE_SEC_CMD command, if it is a value other than 512, the host shall set the block length to 512 by SET_BLOCK_LEN command. If WRITE_SEC_CMD is issued without 512bytes block length, the result is not guaranteed.

Table 2-19 : Argument Field of WRITE_SEC_CMD

Argument Name	Description	Width	Bit Slice
Mode bit	A value of '0' indicates command mode. A value of '1' indicates parameter mode.	1	[31]
Reserved bits	Shall be set to '0'	15	[30:16]
Block Count	Indicates the number of blocks to be transferred from the host to the card. The value 0 indicates 2^{16}(65536).	16	[15:0]

This command has a cmd-mode and a param-mode. The mode is indicated by the argument. In cmd-mode, WRITE_SEC_CMD writes a Secure Token as defined by the security system implemented in the card. In param-mode, the secure token includes the parameter for the following DIRECT_SECURE_READ or DIRECT_SECURE_WRITE.

The host is not allowed to issue a WRITE_SEC_CMD when the ASSD_STATE is "secure command in progress". WRITE_SEC_CMD command received by the card while a previous command is still in progress will be ignored (although the data transfer will be accepted with no errors).

If the host issues a WRITE_SEC_CMD in "param-mode" when PMEM_STATE is "AREA_SELECTED", the card sets ASSD_ERR_STATE to "CONDITION_ERROR" in ASSD-SR.

If the data transfer is terminated by the host prior to the card accepting all data blocks, the command will be ignored. If the number of blocks sent to the card exceeds the number of blocks defined in the command argument, the card will start processing the command, ignore further incoming blocks and will not issue the data CRC response. If any of the data blocks has a CRC error, the host shall issue the respective WRITE_SEC_CMD again and transfer all blocks again.

After every data block the card may assert the BUSY signal, on the DAT line. The BUSY is released when the card is ready to accept the next data block of the secure token. The card may choose to execute any Secure Token in either blocking or non blocking mode. In blocking mode, after the last block of the Secure Token had been received, the card will assert the BUSY signal for the duration of the execution of the Secure Token. In non blocking mode the card will release the BUSY Signal while executing the Secure Token. Partial blocking is also allowed.

The card shall not assert the BUSY signal for time periods longer then the WR_SEC_BUS_BUSY as defined in the card properties register. The only indication for completion of a secure Token execution is the ASSD_STATE field in the card status register. The host shall validate that the card is in the correct state before attempting any ASSD command.

2.4.2 READ_SEC_CMD Command

The READ_SEC_CMD command is available in ASSD mode. This command uses the multiple block read transaction of the SD bus (refer to Figure 2-2 for quick reference. The complete description of this bus transaction is provided in the SD physical specification).

This command continuously transfers data blocks from the card to the host until completing a transfer of previously determined number of blocks. The number of data blocks expected by the host is defined in the command argument (refer to Table 2-20).

Unlike standard READ_MULTIPLE_BLOCK command the usage of the STOP_TRANSMISSION

command is not mandatory. The card will terminate the data transfer phase and return to TRAN state as soon as the requested number of blocks has been transferred. The host may, however, terminate or interrupt the data transfer by issuing STOP_TRANSMISSION command. If STOP_TRANMISSION command is used the bus timing shall be identical to the standard SD read cycle.

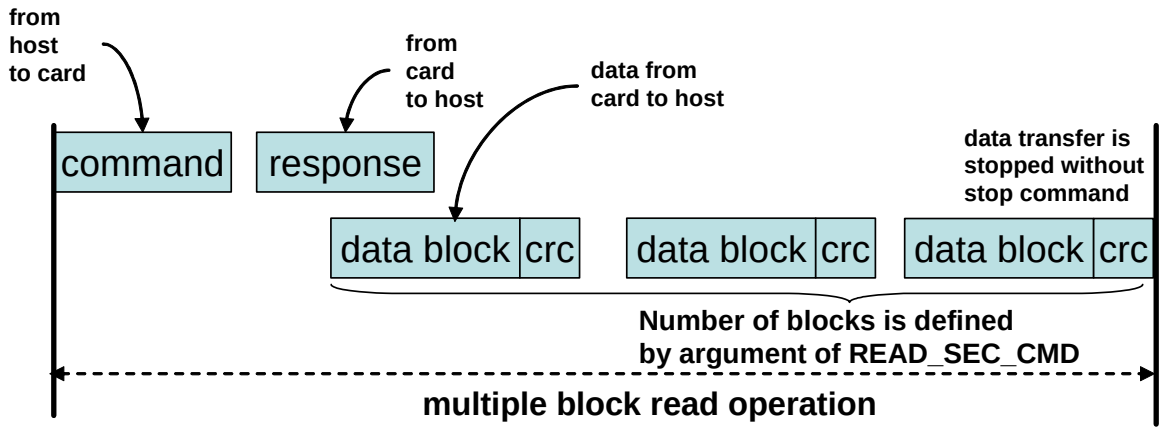

Figure 2-2 : Multiple Block Read Operation of READ_SEC_CMD

Table 2-20 : Argument Field of READ_SEC_CMD

Argument Name	Description	Width	Bit Slice
Reserved bits	Shall be set to '0'	16	[31:16]
Block Count	Indicates the number of blocks to be transferred from the card to the host. The value 0 indicates 2^{16}(65536).	16	[15:0]

The host may issue the READ_SEC_CMD multiple times. The response will include the result of the last WRITE_SEC_CMD.

In READ_SEC_CMD command, the block length shall be set to 512bytes. Before READ_SEC_CMD command, if it is a value other than 512, the host shall set the block length to 512 by SET_BLOCK_LEN command. If READ_SEC_CMD is issued without 512bytes block length, the result is not guaranteed.

There are several failure scenarios in which the card may not have valid data to output:

- The host issues the READ_SEC_CMD while the card is still processing the previous WRITE_SEC_CMD.
- No WRITE_SEC_CMD was issued prior to the first READ_SEC_CMD
- The previous WRITE_SEC_CMD was terminated with a ASSD_SEC_SYS_ERR status.
- The previous WRITE_SEC_CMD was issued in param-mode
- The host issues the READ_SEC_CMD correctly but attempts to read more data blocks then available for output

Although the host is controlling the number of data blocks to be read out of the card, the amount of available data is determined by the card. It is the host responsibility to read the correct number of data

blocks (refer to Sec 3 for detailed information about determining the amount of available data). If the host is reading more blocks then required to contain the available data the card will send out zeros (0).

When the card encounters a processing error while the data is being read and is unable to provide all the available data it shall stop transmitting the data. The host will detect this condition by watching the SEC_READ_LATENCY timeout value.

2.4.3 SEND_PSI Command

The PSI registers (**P**roperties and **S**tatus **I**nformation) are three registers, each 32 bytes long:
The SEND_PSI command is available in ASSD mode. It behaves as a single block read command. Upon reception of a SEND_PSI command, the card returns the data block selected by the register ID command argument (Refer to **Table** 2-21). If SEND_PSI is addressing the ASSD_RNR register while this register is not supported, or a reserved address, the card will return a data block with all zeros.

Table 2-21 : Argument Field of SEND_PSI Command

Argument Name	Description	Width	Bit Slice
Reserved bits	Shall be set to '0'	29	[31:3]
Register ID	See **Table** 2-1 **:** PSI Register ID	3	[2:0]

The host has to issue a SET_BLOCK_LEN command (and set the block length to 32) prior to a SEND_PSI command. The card sends out the number of bytes specified by the SET_BLOCK_LEN command. If the block length is less than 32, not all the register content is sent[2]. If the block length is greater than 32, the card sends undefined padding data. Nevertheless, the CRC checksum is calculated for all the bytes that are transmitted by the card.

2.4.4 CONTROL_ASSD_SYSTEM Command

The CONTROL_ASSD_SYSTEM command shall be used by the host to select, make active and reset, the security system of the ASSD. The host may use this command to explicitly abort the last WRITE_SEC_CMD when its completion is no longer required, or to reset the security system after a WRITE_SEC_CMD times out or terminated with a ASSD_SEC_SYS_ERR status. Upon reception of a CONTROL_ASSD_SYSTEM command the ASSD card will terminate any pending WRITE_SEC_CMD sequence and bring the ASSD security system to its idle state, ready to receive a new WRITE_SEC_CMD command.

The response to the CONTROL_ASSD_SYSTEM command is R1b. As soon as the card clears the busy signal, the ASSD_STATE state will be idle and the card is ready to accept a new WRITE_SEC_CMD command.

The argument field of the command defines the security system and includes an operation bit (refer to **Table** 2-22). If the CONTROL_ASSD_SYSTEM command addresses a non existing security system, or the operation bit is set to '0' the card will ignore the command.

[2] This can be used by a secure host who doesn't want to read all the 32 bytes of the register.

Table 2-22 : Argument Field of CONTROL_ASSD_SYSTEM Command

Argument Name	Description	Width	Bit Slice
Reserved bits	Shall be set to '0'	20	[31:12]
Security System Index	4 bit unsigned integer. Shall be set to the bit position of the addressed security system.	4	[11:8]
Reserved bits	Shall be set to '0'	7	[7:1]
Operation	'1' – Select and Reset '0' – No Operation	1	[0]

2.4.5 DIRECT_SECURE_READ Command

DIRECT_SECURE_READ is a command for reading data from Protected Memory. This command continuously transfers data blocks from card to host until completing a transfer of the previously determined number of blocks. The host can interrupt the transfer by issuing a STOP_TRANSMISSION command. This command uses the multiple block read transaction of the SD bus (refer to **Figure** 2**-**2 for quick reference. The complete description of this bus transaction is provided in the SD physical specification).

The essential parameters, which include the number of blocks, of this command is set by WRITE_SEC_CMD in param-mode, so DIRECT_SECURE_READ shall always follow WRITE_SEC_CMD in param-mode. But, the SEND_PSI command or normal SD commands can be interleaved between the WRITE_SEC_CMD and the DIRECT_SECURE_READ.

If the parameters set by WRITE_SEC_CMD in param-mode are wrong, the card sets ASSD_ERR_STATE to "AREA_NOT_FOUND" or "RANGE_OVER" in ASSD-SR. If the parameters are yet to be set (a valid WRITE_SEC_CMD in param-mode has not been issued previously), the card sets "CONDITION_ERROR". If the authentication code in the command argument is invalid, the card sets "AUTH_ERROR". When an error occurs, the card does not transfer data. So, the host should wait for timeout, issue STOP_TRANSMISSION for the card to return to TRAN state, and issue SEND_PSI command to get the ASSD_ERR_STATE. If the host is trying to read more data blocks that specified in the previous WRITE_SEC_CMD in param-mode the card will time out after the available data block have been sent out.

If the STOP_TRANSMISSION command is issued during the DIRECT_SECURE_READ, the data transfer should be stopped under the timing indicated in 2.7.4. The transferred data is encrypted with the algorithm indicated by ENC_ALG in ASSD-SR, so the odd data of the cipher block size (e.g. if the Triple DES, it is 8 bytes) cannot be decrypted correctly.
In DIRECT_SECURE_READ command, the block length is fixed to 512bytes. Before DIRECT_SECURE_READ command, if it is a value other than 512, the host shall set the block length to 512 by SET_BLOCK_LEN command. If DIRECT_SECURE_READ is issued without 512bytes block length, the result is not guaranteed.

Authentication Code

The authentication code in the command argument is generated with the latest Direct Memory Access Authentication Challenge, the Data Payload of previous WRITE_SEC_CMD in param-mode, and the S-AUTH key.

For generation of the authentication code, the algorithm indicated by AUTH_ALG in ASSD-SR shall be used. Before issuing DIRECT_SECURE_READ, The host should get the S-AUTH key the same as the card holds. If "No Authentication" is indicated by AUTH_ALG, the command argument is ineffective.

Read Data

Read data shall be encrypted with the S-ENC key by the card and transferred unless “No Encryption” is indicated by ENC_ALG.

Before issuing DIRECT_SECURE_READ, The host should get the S-ENC key the same as the card holds.

The algorithm indicated by ENC_ALG in ASSD-SR should be used for data encryption.

2.4.6 DIRECT_SECURE_WRITE Command

DIRECT_SECURE_WRITE is a command for writing data to Protected Memory. This command continuously writes blocks of data until all of the specified data blocks have been written. The host can interrupt the transfer by issuing STOP_TRANSMISSION command. This command uses the multiple block write transaction of the SD bus (refer to Figure 2-1 for quick reference. The complete description of this bus transaction is provided in the SD physical specification).

The essential parameters, which include the number of blocks, of this command are set by WRITE_SEC_CMD in param-mode, so DIRECT_SECURE_WRITE shall always follow WRITE_SEC_CMD in param-mode. But, the SEND_PSI command or normal SD commands can be interleaved between the WRITE_SEC_CMD and the DIRECT_SECURE_WRITE.

If the parameters set by WRITE_SEC_CMD in param-mode are wrong, the card sets ASSD_ERR_STATE to “AREA_NOT_FOUND” or “RANGE_OVER” in ASSD-SR. If the parameters are yet to be set (a valid WRITE_SEC_CMD in param-mode has not been issued previously), the card sets “CONDITION_ERROR”. If the authentication code in the command argument is invalid, the card sets “AUTH_ERROR”. When an error occurs, the card does not receive data and may not send the CRC response to the data block. The host should issue STOP_TRANSMISSION for the card to return to TRAN state, and issue SEND_PSI command to get the ASSD_ERR_STATE. If the host is trying to write more data blocks that specified in the previous WRITE_SEC_CMD in param-mode the card will ignore further incoming blocks and will not issue the data CRC response.

Before issuing DIRECT_SECURE_WRITE, The host should get the S-ENC key the same as the card holds.

If the STOP_TRANSMISSION command is issued during the DIRECT_SECURE_WRITE, the data transfer should be stopped under the timing indicated in 2.7.5. The card should write the complete data block before receiving the STOP_TRANSMISSION command.

In DIRECT_SECURE_WRITE command, the block length is fixed to 512bytes. Before DIRECT_SECURE_WRITE command, if it is a value other than 512, the host shall set the block length to 512 by SET_BLOCK_LEN command. If DIRECT_SECURE_WRITE is issued without 512bytes block length, the result is not guaranteed.

This command writes blocks of data continuously to an area specified by Area#. This command should always follow WRITE_SEC_CMD in param-mode.

Data to be written should be encrypted with the session key (S-ENC) by the host.

Authentication Code

The authentication code in arguments of this command is the same as DIRECT_SECURE_READ.

Written Data

Written data should be encrypted with the S-ENC key by the host and transferred to the card.

The algorithm for encrypting written data is the same as DIRECT_SECURE_READ.

The card does not verify validity of written data, so validity of written data is not ensured if a host encrypts data with an invalid key or it does not encrypt the data.

2.5 Command Sequences

2.5.1 Security System Selection

The security system selection process, (refer to Figure 2-3), starts with querying the card and checking availability of the ASSD security system. This is done by issuing an SD Switch command in sense mode. In this mode the card reports its switching capabilities. Depending on the implementation of the host side application, the host may test for support of either ASSD 1.1 or ASSD 2.0 Extensions. If the appropriate ASSD extension is found on the card, the host shall switch the card into the required mode by using the SD switch command in Set mode.

An ASSD card will always have one of its security systems selected and activated. A default security system is activated by the card as soon as it is switched into ASSD mode. The default security system is at the card discretion and not defined by this specification. Once a security system is selected by the host, the card will keep it selected and active until it switched out of ASSD mode or the host selects another one.

As ASSD 1.1 extension protocol does not support selecting security system, as soon as the card is switched into ASSD 1.1 mode the host shall start communicating with the card security application using the APDU secure token protocol (see section 3.1). Host and card applications shall be designed to identify each other using the application specific APDUs.

In ASSD 2.0 the extension protocol is designed to support ASSD cards with multiple security systems. Therefore, after switching the card into this mode, the host shall use command SEND_PSI (CMD36) to query the card for the available security systems. After the required system is selected, the host will use the appropriate secure token protocol, as indicated in the status register, to communicate with the card. The card will always have a security system selected. The default security system to be selected immediately after the card is switched into ASSD 2.0 mode is at the card discretion in not defined by this standard.

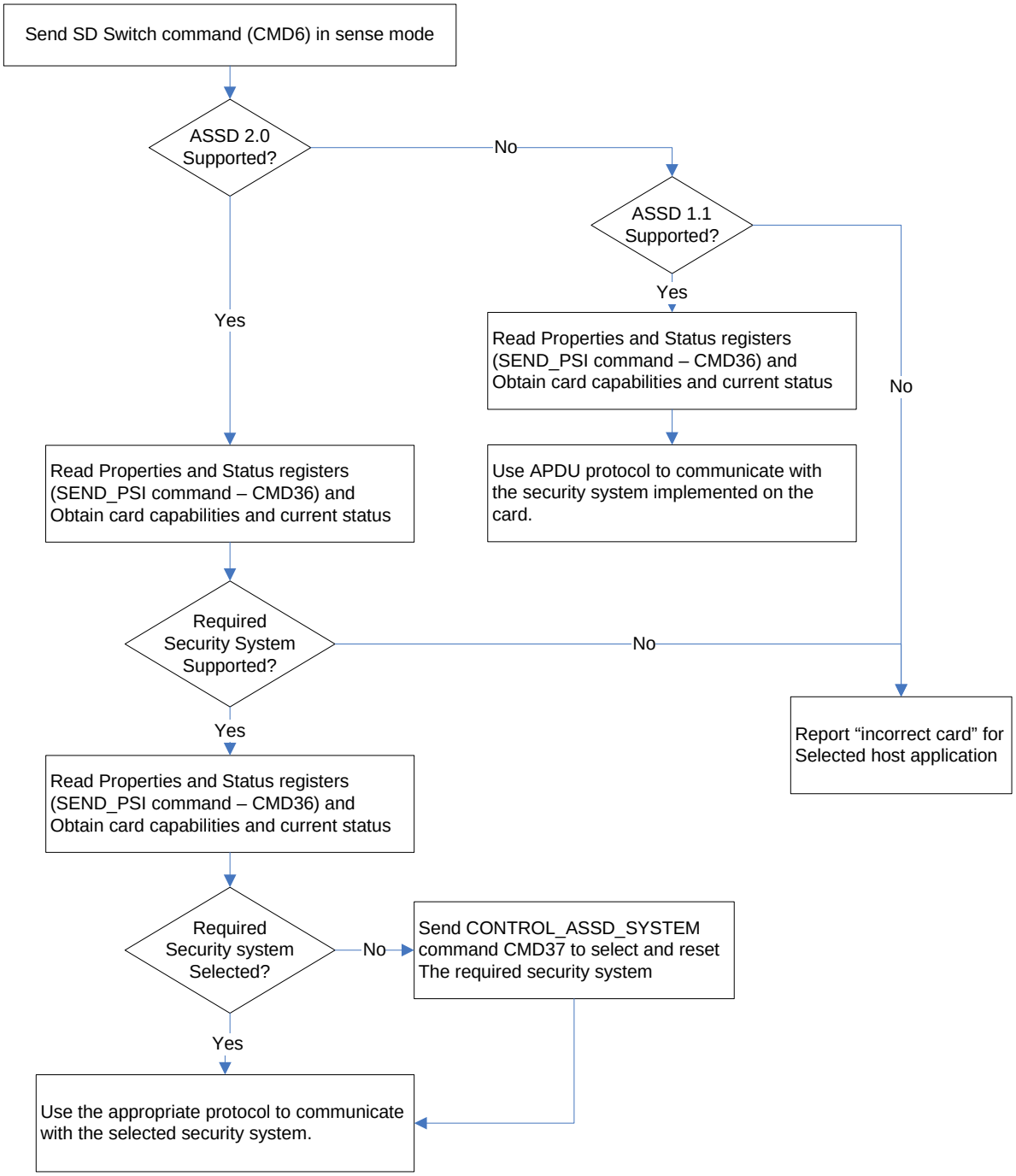

Figure 2-3 : Security System Selection

2.5.2 Secure Command Sequence

A secure commands sequence is composed of a WRITE_SEC_CMD command in cmd-mode optionally followed by a READ_SEC_CMD command. If the host is executing a secure transaction with data transferred into the card and no expected output, the READ_SEC_CMD maybe omitted. An overview of the secure command sequence is provided in Figure 2-4; the figure does no show the initial operations of switching the card to the ASSD mode and selecting the appropriate Security System.

ASSD cards may choose to implement a WRITE_SEC_COMMAND either in blocking or non blocking mode. If non blocking mode is selected, the ASSD cards will release the bus busy signal before the termination of the WRITE_SEC_CMD. In this non blocking mode the host is allowed to interleave legacy commands within the secure command sequence. Interleaving of secure commands is not allowed. Upon the de-assertions of the bus busy signal, and as long as the ASSD_STATE is in-progress, the host may only issue standard SD card commands and the ASSD SEND_PSI command. The host shall make sure that the WRITE_SEC_CMD is terminated by reading the ASSD status register and checking the ASSD_STATE field, before issuing any other ASSD command.
Two optional timeout variables may be used by the card to publish the maximum time periods allocated for secure command execution. WR_SEC_BUS_BUSY indicates the maximum time the card will assert the busy signal while executing an ASSD command. SEC_WRITE_LATENCY indicates the maximum time the card will spend executing a command. In blocking mode these two variables will show the same value. In non blocking mode the WR_SEC_BUS_BUSY will always be smaller then SEC_WRITE_LATENCY.

The assumption is that ASSD cards will, typically, make use of the timeout option and allow the host to optimize its SW driver design by detecting card failures which manifest themselves as failure to complete command execution. However, since the security operations themselves are left for security application specs and not defined in this standard, the protocol is providing for a scenario where a card is incapable of assuring command execution within the time limits defined in this standard. In this case the card shall set its time out registers to (0) and the host shall disable its time out counters and rely solely on the card state to detect end of execution.
Legacy commands have priority over secure commands, and, if non blocking mode is enabled, the card shall execute them. If, as a consequence of executing the higher priority legacy commands, a secure command cannot be completed before SEC_WRITE_LATENCY has elapsed, the card shall abort the execution of the secure command and set the ASSD_STATE field, in the ASSD-SR register to “Secure Command Aborted”.
It is the responsibility of the host to follow the correct secure commands sequence. If a host does not follow the secure sequence, the command execution is not guaranteed and the secure response is not predictable.

The command sequence for direct memory access is as follows:

The WRITE_SEC_CMD is sent in param-mode and followed by either DIRECT_SECURE_WRITE or DIRECT_SECURE_READ.

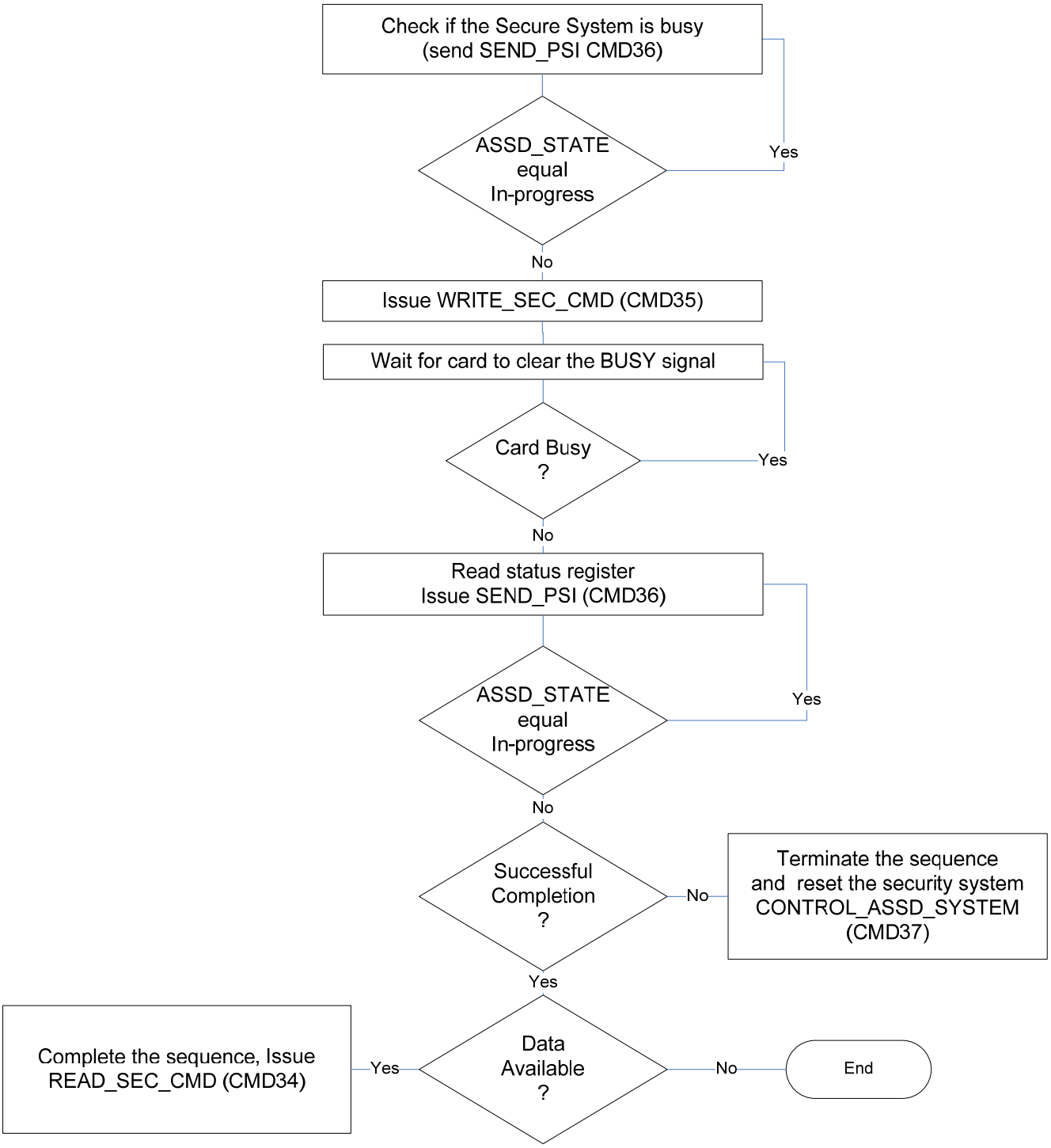

Figure 2-4 : Secure Command Sequence

2.6 Card State Transition Table

Table 2-23 defines the legal card state transitions for the ASSD card extensions.

Table 2-23 : Card State Transition

Current State		idle	ready	ident	stby	tran	data	rcv	prg	dis	ina
CMD	Comment	Next State									
CMD34	READ_SEC_CMD	-	-	-	-	data	-	-	-	-	-
CMD35	WRITE_SEC_CMD	-	-	-	-	rcv	-	-	-	-	-
CMD36	SEND_PSI	-	-	-	-	data	-	-	-	-	-
CMD37	CONTROL_ASSD_SYSTEM	-	-	-	-	prg	-	-	-	-	-
CMD50	DIRECT_SECURE_READ	-	-	-	-	data	-	-	-	-	-
CMD57	DIRECT_SECURE_WRITE	-	-	-	-	rcv	-	-	-	-	-

While in ASSD mode, the ASSD card is concurrently maintaining the legacy SD and the ASSD state transition tables (refer to Figure 2-5). For every state (a circle in the diagram) the SD state is defined above the horizontal line while the ASSD state is shown below. The shaded circles define the states where the card is not busy and, therefore, interleaving of SD legacy command, as well as SEND_PSI and CONTROL_ASSD_SYSTEM, within an ASSD secure command sequence, is allowed. The *Tran/Idle* state is, of course, the trivial case of ASSD being inactive; it is also the state the card enters after completion processing of the SD switch command (CMD6) and entering ASSD mode.

The SEND_PSI and the CONTROL_ASSD_SYSTEM commands are, in general, not shown in the state diagram. The only exception is the transition from *Tran/Aborted* to *Prg/Aborted* where the CONTROL_ASSD_COMMAND is required and not optional. Both commands are allowed whenever the SD card is in *Tran* state, regardless of the ASSD state. SEND_PSI command does not trigger an ASSD state change. CONTROL_ASSD_SYSTEM will return the card to the ASSD idle state.

The state transition diagram describes only the allowed commands per state and the valid state transitions. Issuing a command while the card is in the wrong state will not trigger any state change. The error reporting and recovery procedures are included in the command description sections.

As can be seen from the diagram, the command flow without interleaving is pretty straight forward.

- While the system is in *Tran/idle* state either SD or ASSD commands can be issued. An SD command will take the card through a series of SD legacy states as defined in the SD physical spec (denoted as SD-varies in the diagram).
- WRITE SEC_CMD command issued while the card is either in Idle or Completed states will start a secure token exchange sequence. If issue while the card is in Completed state the result of the previous transition is discarded, regardless if the command used in Command or Param modes, and the card starts a new sequence.
- READ SEC_CMD command issued either in *Idle* or *Completed* states will yield the currently available data, or all-zero - if there is no available data.

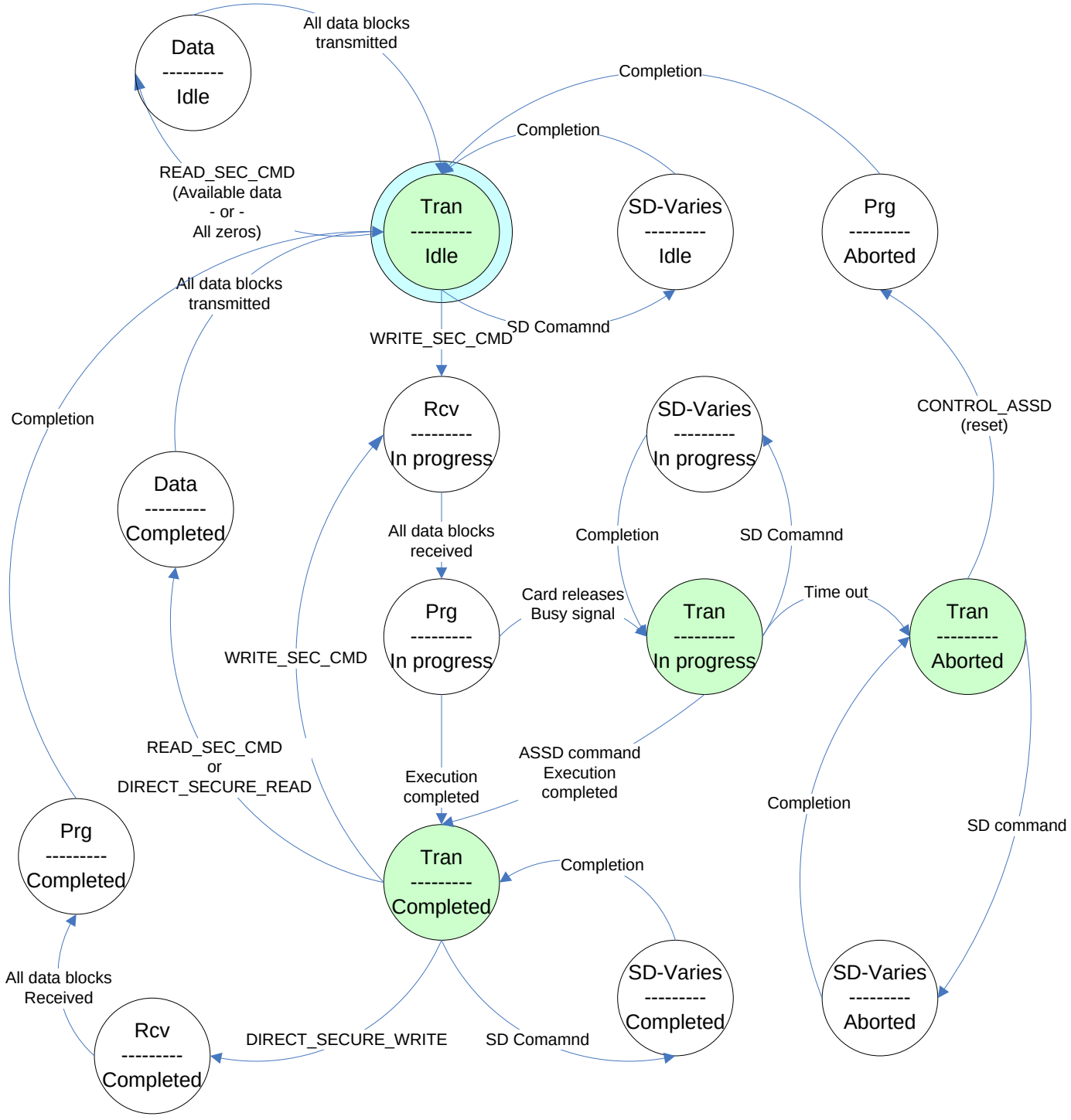

Figure 2-5 : Card State Transition Diagram

- DIRTECT_SECURE_WRITE and DIRECT_SECURE_READ are allowed only in Completed state and only if the WRITE_SEC_CMD was used in Param mode. The card will commence either a read or write transaction and as soon as all data blocks are transferred will move to *Idle* state.

If non blocking mode is allowed (the card releases busy signal while in *in-progress* state) the card moves into *Tran/In-progress* state, in which, it will accept and execute legacy SD command, pausing the ASSD command execution as needed. If too many legacy commands are issued and, as a result, the card is unable to complete the ASSD command execution within SEC_WRITE_LATENCY, the ASSD command will eventually be aborted. In this case the host shall reset the ASSD system and reissue the command.

The *Tran/Completed* state is, by definition, an interleaving state. The host is allowed to issue legacy SD commands prior to issuing a appropriate ASSD command required to complete the secure token transaction.

2.7 Command Timing

2.7.1 SEND_PSI

The SEND_PSI command follows the timing of the read single block transaction as defined in Figure 4-18 of the SD Card Physical layer specification Rev 2.0

2.7.2 READ_SEC_CMD

The read secure command follows the timing of the multiple block read transaction as defined in Figures 4-19 and 4-20 of the SD Card Physical layer specification rev 2.0. The timings given in SD Card Physical Spec are the same except the STOP TRAN command which may be eliminated in normal operation (as shown in Figure 2-2). In case that the process is terminated with STOP_TRAN command the timing is exactly as given in the SD card Physical Spec.

2.7.3 WRITE_SEC_CMD

The write secure command follows the timing of the multiple block write transaction as defined in Figures 4-22 to 4-23 of the SD Card Physical layer specification rev 2.0. The timings given in SD Card Physical Spec are the same except the STOP TRAN command which may be eliminated in normal operation (as shown in Figure 2-1). In case that the process is terminated with STOP_TRAN command the timing is exactly as given in the SD card Physical Spec.

2.7.4 DIRECT_SECURE_READ

The direct secure read command follows the timing of the multiple block read transaction as defined in Figure 4-19 and 4-20 of the SD Card Physical layer specification rev 2.0. The timings given in SD Card Physical Spec are the same except the STOP TRAN command which may be eliminated in normal operation (as shown in Figure 2-2) and Nac(max) is calculated from PMEM_RD_TIME (refer to section 2.3.2). In case that the process is terminated with STOP_TRAN command the timing is exactly as given in the SD card Physical Spec.

2.7.5 DIRECT_SECURE_WRITE

The direct secure write command follows the timing of the multiple block write transaction as defined in Figures 4-22 to 4-23 of the SD Card Physical layer specification rev 2.0. The timings given in SD Card Physical Spec are the same except the STOP TRAN command which may be eliminated in normal operation (as shown in Figure 2-1). In case that the process is terminated with STOP_TRAN command the timing is exactly as given in the SD card Physical Spec.

3. Secure Tokens

The commands to, and the responses from, the various ASSD Security Systems are transferred as Secure Tokens. Although, each security system may use different formats for the secure tokens they are always transferred to/from the card over the data blocks of the WRITE_SEC_CMD and the READ_SEC_CMD using the standard SD physical protocol for transmission of data blocks.

ASSD security systems may choose to use any of the secure token protocols defined in the following pages. When a security system is selected, the SECURE_TOKEN_PROT field of the status register indicates the secure token protocol used by the selected security system.

3.1 APDU Protocol

APDU data formats are defined in the ISO-7816 standards. As defined in the standard specification, security related commands are transferred to/from the card via Command and Response APDUs.

The APDU protocol enables a security system, designed to use the APDU data formats, to transfer the APDU packets between the host and the card. The 5C security system (Index 0) is an example of a security system using this protocol (refer to the 5C specification titled M-Commerce Extension Part 1) as well as the ASSD SDA core security system (Index 1).

In the APDU protocol, a Secure Token is composed of the Command or Response APDU prefixed with a Secure Token Length (STL) field. This is a 2 bytes field equal to the length (in bytes) of the Secure Token length (APDU-Length + 2) which includes two bytes for the STL field itself. If the total length of the secure token is smaller than the currently defined block length, the transmitted SD data block is padded with zeros [‘0’] (refer to Figure 3-1).

When the total length of a Secure Token exceeds the currently defined SD data block length it will be split over several SD data blocks (refer to Figure 3-2). When the length of the last segment of the Secure Token is smaller than the currently defined block length the transmitted block is padded with zeros (0).

When a secure Token is transmitted from the card to the host the STL field of the Secure Token will indicate the length of the valid data within the received data block. The content of data bytes beyond the length of the Secure Token shall be set to (0).

If the host attempts to read a Secure Token from the card when none is available the card will set the STL field to 2 (indicating zero length payload field) and shift the required amount of bits out to the bus. The data beyond the STL two bytes is set to (0).

The size of the response APDU (and therefore the number of blocks required to read it out of the card) is defined either in Security System or Security application specifications and are beyond the scope of this document.

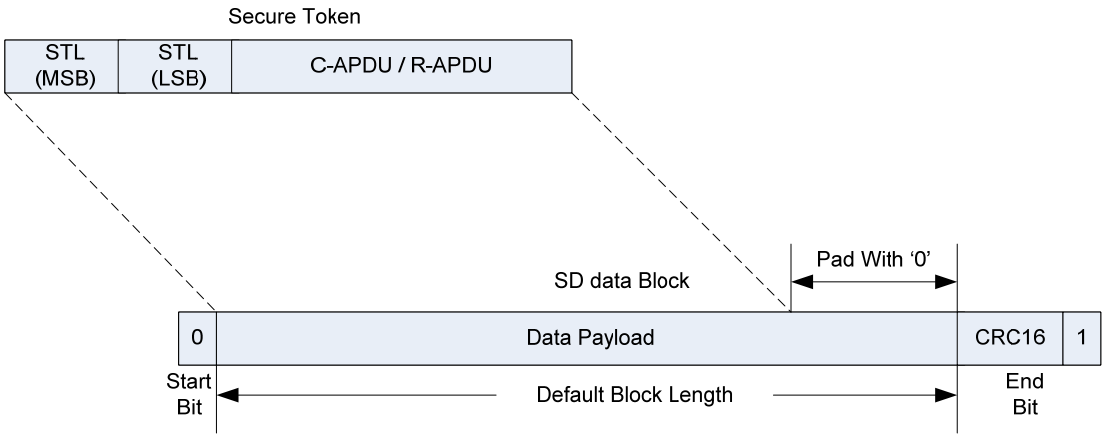

Figure 3-1 : Secure Token Smaller than Default Block (includes APDU)

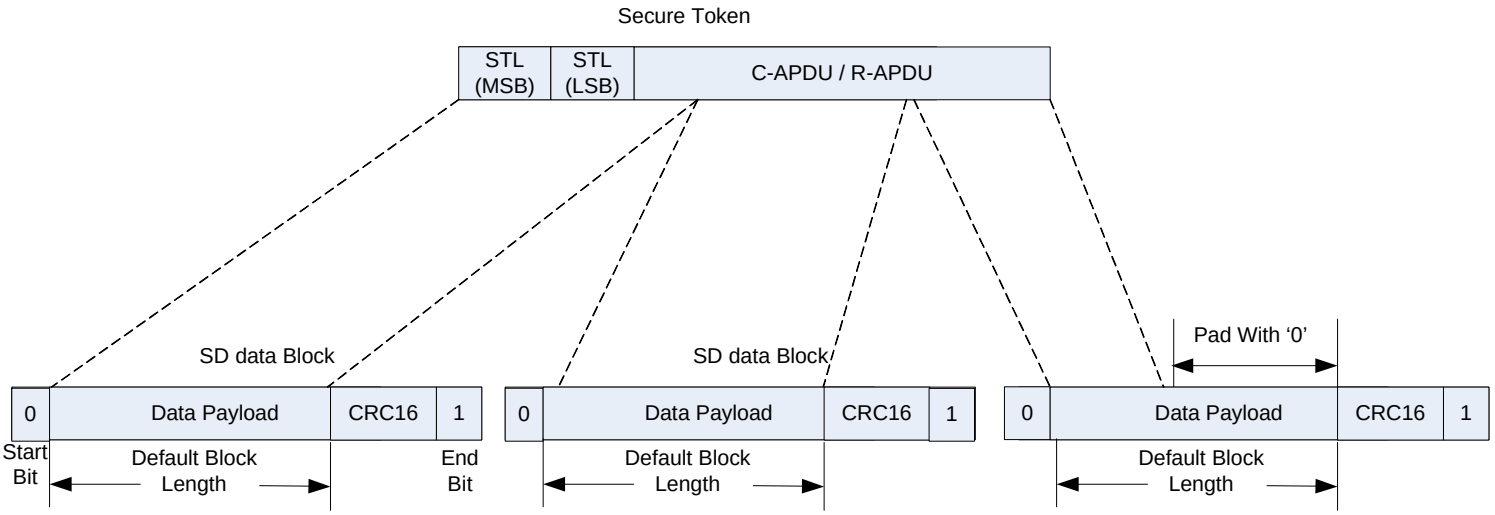

Figure 3-2 : Secure Token Larger than Default Block (includes APDU)

3.2 Data Block Protocol

Data Block protocol defines how the command packets and responses are transfered between the host and card without providing any definition for the data formats. These are left to be defined by the security system using this protocol.

In data block protocol the Secure Tokens are always a multiple of 512 bytes, therefore, there is no need for padding. The Secure Token is split over as many data block as needed, according to the length of the Secure Token.

The bus transaction is identical to the APDU protocol with one exception. As for the APDU protocol, the data length for WRITE_SEC_CMD command is calculated by the card from the block count argument of the command. However, for READ_SEC_CMD command the number of relevant data blocks to be read out is defined in the READ_BLOCK_COUNT field in the status register.

If the host attempts to read a Secure Token from the card when none is available, (READ_BLOCK_COUNT field is set to zero (0)) or attempts to read more data blocks then available, the card will shift the required amount of bits out to the bus. The data sent out is all zero (0).

3.3 WRITE_SEC_CMD in param_mode

The parameters for Direct Memory Access commands are also packaged into Secure Tokens and transmitted to the card. The parameters consist of 1 byte of Area#, 3 bytes of Block offset, and 3 bytes of Block count. Block offset is the offset from the start of the Protected Memory Area specified by Area# in block units. Block counts are also specified in units of blocks. Similarly to the way APDUs are packages the parameters will be prefixed by a 2 bytes Length field. The value of the length field shall be parameter-length + 2. Padding conventions are identical to the APDU protocol.(refer to Figure 3-1)

The bus transaction is similar to the APDU and Data block protocols. The only exception is that in case of error the card, similar to standard SD write and read transactions, times out.

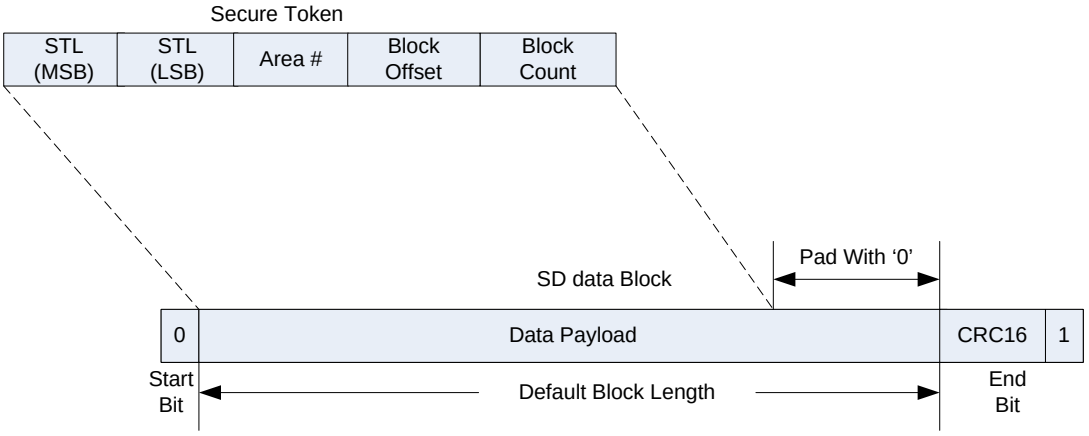

Figure 3-3 : Secure Token (includes Parameters for Direct Memory Access)

Appendix A (Normative) : Reference

A.1 Referenced Documents

This specification refers the following documents.

[1] Part 1, Physical Layer Specification Version 2.00 – Available from www.sdcard.org.

[2] Part A1, Mc-EX Extension Specification Version 1.10 – Available from www.sdcard.org

[3] Part A1, ASSD Extension Specification Version 2.00 – Available from www.sdcard.org

[4] Part A3, ASSD SDA Core Specification Version 1.00 – Available from www.sdcard.org

[5] 5C Mc-EX Core specification – Available from 5C consortium.

Appendix B (Normative) : Special Terms

B.1 Terminology

Authentication	The process of verifying the identity of someone or the authenticity of data. The process will typically use cryptographic algorithms.
Block I/O	A communication protocol used to communicate between host computer systems and mass storage devices. The standard unit of data transferred is a block of 512 bytes which is the standard storage unit of the mass storage devices.
Copy Protection	This term is used to describe a system that prevents creation of unauthorized copies of computer data files. Usually the discussion is in the context of digital multi media content stored in files.
Form Factor	The mechanical shape and size of a physical electrical component. In this document the discussion is restricted to mass memory devices.
Non Volatile memory	A memory component which maintain the data written on it without the need for external power. In this document the text refers to removable memory devices (cards) which retain the data even after it is disconnected from the host device.
Plaintext	A cryptographic term used to denote the original (readable) form of a data element as opposed to the encrypted form.
Smart Card	A generic name for a series of products using a physically secure hardware design that protect the data stored on this chips from being tampered with. These products are usually used as personal identification devices, bank cards, preloaded payment cards, etc…

B.2 Abbreviations

APDU	Application Protocol Data Unit
ASSD	Advanced Security SD
CPRM	Content Protection for Recordable Media
PSI	Properties and Status Information
TRM	Tamper Resistant Module