SD Memory Card Specifications

Part 1
PHYSICAL LAYER SPECIFICATION
Version 1.01

April 15 2001

SD Group

Matsushita Electric Industrial Co., Ltd. (MEI)

SanDisk Corporation

Toshiba Corporation

Copyright (C)2000 (C)2001 by SD Group (MEI, SanDisk, Toshiba)
Conditions for publication:

- **Publisher and Copyright Holder:**
  SD Group (MEI, SanDisk, Toshiba)

- **Exemption:**
  None will be liable for any damage from use of this document.

Revision History

Date	Version	Changes compared to previous issue
March 22th, 2000	1.0	Base version
April 15th, 2001	1.01	- The Supplementary Note (June 2000) that includes clarifications to the spec was incorporated into the spec. - Reliability/Durability - the open issues were defined (Torque/ Bending/WP Switch cycles) - ESD: higher voltages for Non-contact/Air discharge were defined. - Card's Thickness tolerance were re-defined for the center area of the card and more clarifications were added to the WP switch diagram - Thin SD Card: Mechanical Drawing was added. - More clarifications were added for SD Read Only Cards. - Maximum time out for Write/Erase was changed to 250ms. - Underrun/Overrun status bits were removed (non-relevant in SD Card) from Table 22. - Typo fixes and some clarification notes - Update Fig 31 - Stop Tran during CRC response at Mult Blk WR. - Add SD ROM Card type in SD_STATUS. - Fix error in SEND_CID/CSD Ncr timing. - A clarification about the operating frequency range were added (0-25Mz mandatory for SD Card). - Max Current during initialization period was re-defined. - Initialization sequence in SPI mode for SD was updated.

1 General description - 6

2 System features - 8

3 SD Memory Card System Concept - 9

3.1 Bus Topology - 9
3.1.1 SD bus - 10
3.1.2 SPI bus - 11
3.2 Bus Protocol - 12
3.2.1 SD bus - 12
3.2.2 SPI Bus - 15
3.3 SD Memory Card - Pins and Registers - 17
3.4 Compatibility to MultiMediaCard - 19

4 SD Memory Card Functional Description - 22
4.1 General - 22
4.2 Card Identification Mode - 23
4.2.1 Card Reset - 23
4.2.2 Operating Voltage Range Validation - 23
4.2.3 Card Identification Process - 25
4.3 Data Transfer Mode - 25
4.3.1 Wide Bus Selection/Deselection - 27
4.3.2 Data Read - 28
4.3.3 Data Write - 28
4.3.4 Erase - 29
4.3.5 Write Protect Management - 30
4.3.6 Card Lock/Unlock Operation (Optional) - 31
4.3.7 Copyright Protection - 33
4.3.8 Application specific commands - 34
4.4 Clock Control - 35
4.5 Cyclic redundancy codes (CRC) - 36
4.6 Error Conditions - 37
4.6.1 CRC and Illegal Command - 37
4.6.2 Read, Write and Erase Time-out Conditions - 37
4.7 Commands - 38
4.7.1 Command Types - 38
4.7.2 Command Format - 39
4.7.3 Command Classes (Redefined for SD Memory Card) - 39
4.7.4 Detailed Command Description - 40
4.8 Card State Transition Table - 45
4.9 Responses - 47
4.10 SD Memory Card Status - 49
4.10.1 Card Status - 49
4.10.2 SD Status - 52
4.11 Memory Array Partitioning - 53

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

4.12 Timings - 55
4.12.1 Command and Response - 55
4.12.2 Data Read - 56
4.12.3 Data Write - 57
4.12.4 Timing Values - 60

5 Card Registers - 61
5.1 OCR Register - 61
5.2 CID Register - 62
5.3 CSD Register - 63
5.4 RCA Register - 72
5.5 DSR Register (Optional) - 72
5.6 SCR Register - 72

6 SD Memory Card Hardware Interface - 74
6.1 Hot insertion and removal - 74
6.2 Card Detection (Insertion/Removal) - 75
6.3 Power protection (Insertion/Removal) - 75
6.4 Power up - 76
6.5 Programmable card output driver (Optional) - 78
6.6 Bus operating conditions - 80
6.7 Bus signal levels - 81
6.8 Bus timing - 82
6.9 Low Voltage (1.8v) SD Memory Cards (Preliminary) - 83

7 SPI Mode - 85
7.1 Introduction - 85
7.2 SPI Bus Protocol - 85
7.2.1 Mode Selection - 85
7.2.2 Bus Transfer Protection - 86
7.2.3 Data Read - 86
7.2.4 Data Write - 87
7.2.5 Erase & Write Protect Management - 89
7.2.6 Read CID/CSD Registers - 89
7.2.7 Reset Sequence - 89
7.2.8 Error Conditions - 90
7.2.9 Memory Array Partitioning - 90
7.2.10 Card Lock/unlock - 90
7.2.11 Application Specific commands - 90
7.2.12 Copyright Protection commands - 90
7.3 SPI Mode Transaction Packets - 90
7.3.1 Command Tokens - 90
7.3.2 Responses - 95
7.3.3 Data Tokens - 98
7.3.4 Data Error Token - 99
7.3.5 Clearing Status Bits - 99

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

7.4 Card Registers - 101
7.5 SPI Bus Timing Diagrams - 101
7.5.1 Command / Response - 102
7.5.2 Data read - 103
7.5.3 Data write - 103
7.5.4 Timing Values - 104
7.6 SPI Electrical Interface - 104
7.7 SPI Bus Operating Conditions - 105
7.8 Bus Timing - 105

8 SD Memory Card mechanical specification - 106
8.1 Card package - 106
8.1.1 External signal contacts (ESC) - 106
8.1.2 Design and format - 107
8.1.3 Reliability and durability - 107
8.1.4 Electrical Static Discharge (ESD) Requirements - 108
8.1.5 Quality assurance - 108
8.2 Mechanical form factor - 109
8.3 System: card and connector - 112
8.3.1 Card hot insertion - 112
8.3.2 Inverse insertion - 112
8.3.3 Card Orientation - 113
8.4 Thin (1.4mm) SD Memory Card (Preliminary) - 113

9 Appendix - 116
9.1 Power Supply Decoupling - 116
9.2 Connector - 116
9.2.1 General - 116
9.2.2 Card Insertion and Removal - 117
9.2.3 Characteristics - 118

10 Abbreviations and terms - 121

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

1 General description

SD Memory Card (Secure Digital Memory Card) is a memory card that is specifically designed to meet the security, capacity, performance and environment requirements inherent in newly emerging audio and video consumer electronic devices. The SD Memory Card will include a copyright protection mechanism that complies with the security of the SDMI standard and will be faster and capable for higher Memory capacity. The SD Memory Card security system uses mutual authentication and a "new cipher algorithm" to protect from illegal usage of the card content. A none secured access to the user's own content is also available. The physical form factor, pin assignment and data transfer protocol are forward compatible with the MultiMediaCard with some additions.

The SD Memory Card communication is based on an advanced 9-pin interface (Clock, Command, 4xData and 3xPower lines) designed to operate in at maximum operating frequency of 25MHz of and low voltage range. The communication protocol is defined as a part of this specification. The SD Memory Card host interface supports regular MultiMediaCard operation as well. In other words, MultiMediaCard forward compatibility was kept. Actually the main difference between SD Memory Card and MultiMediaCard is the initialization process.

The SD Memory Card Specifications were divided to several documents. The SD Memory Card documentation structure is given in Figure 1.

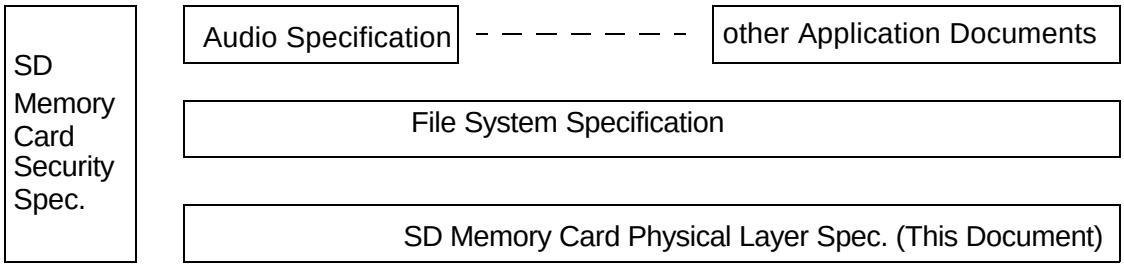


Figure 1: SD Memory Card Documentation Structure

- **SD Memory Card Audio Specification:**

This specification along with other application specifications describe the specification of certain application (in this case - Audio Application) and the requirements to implement it.

- **SD Memory Card File System Specification:**

Describes the specification of the file format structure of the data saved in the SD Memory Card (in protected and un-protected areas).

- **SD Memory Card Security Specification:**

Describes the copyright protection mechanism and the application specific commands that support it.

- **SD Memory Card Physical Layer Specification (this document):**

Describes the physical interface and the command protocol used by the SD Memory Card.

The purpose of the SD Memory Card Physical Layer specification is the definition of the SD Memory Card, its environment and handling.

The document is split up into several portions. Chapter 3 gives a general overview of the system

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

concepts. The common SD Memory Card characteristics are described in Chapter 4. As this description defines an overall set of card properties, we recommend to use the product documentation in parallel. The card registers are described in Chapter 5.

Chapter 6 defines the electrical parameters of the SD Memory Card's hardware interface.

Chapter 8 describes the physical and mechanical properties of the SD Memory Cards and the minimal recommendations to the card slots or cartridges.

As used in this document, "shall" or "will" denotes a mandatory provision of the standard. "Should" denotes a provision that is recommended but not mandatory. "May" denotes a feature whose presence does not preclude compliance, that may or may not be present at the option of the implementor.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

2 System features

- **Targeted for portable and stationary applications**
- **Voltage range:**

  SD Memory Card -
  Basic communication (CMD0, CMD15, CMD55, ACMD41): 2.0 - 3.6V
  Other commands and memory access: 2.7 - 3.6V

  SDLV Memory Card (low voltage) - Operating voltage range: 1.6 - 3.6V
- **Designed for read-only and read/write cards.**
- **Variable clock rate 0 - 25 MHZ**
- **Up to 10MByte/sec Read/Write rate (using 4 parallel data lines).**
- **Maximum data rate with up to 10 cards**
- **Correction of memory field errors**
- **Card removal during read operation will never harm the content**
- **Forward compatibility to MultiMediaCard**
- **Copyrights Protection Mechanism - Complies with highest security of SDMI standard.**
- **Password Protection of cards (option)**
- **Write Protect feature using mechanical switch**
- **Built-in write protection features (permanent and temporary)**
- **Card Detection (Insertion/Removal)**
- **Application specific commands**
- **Comfortable erase mechanism**
- **Protocol attributes of the communication channel:**

SD Memory Card Communication Channel
Six-wire communication channel (clock, command, 4 data lines)
Error-protected data transfer
Single or Multiple block oriented data transfer

- **SD Memory Card thickness is defined as either 2.1mm (normal) and 1.4mm (Thin SD Memory Card) .**

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

3 SD Memory Card System Concept

The SD Memory Card provides application designers with a low cost mass storage device, implemented as a removable card, that supports high security level for copyright protection and a compact, easy-to-implement interface.

SD Memory Cards can be grouped into several card classes which differ in the functions they provide (given by the subset of SD Memory Card system commands):

- Read/Write (RW) cards (Flash, One Time Programmable - OTP, Multiple Time Programmable - MTP). These cards are typically sold as blank (empty) media and are used for mass data storage, end user recording of video, audio or digital images.
- Read Only Memory (ROM) cards. These cards are manufactured with a fixed data content. They are typically used as a distribution media for software, audio, video etc.

In terms of operating supply voltage, two types of SD Memory Cards are defined:

- SD Memory Cards which supports initialization/identification process with a range of 2.0-3.6v and operating voltage within this range as defined in the CSD register.
- SDLV Memory Cards - Low Voltage SD Memory Cards, that can be operate in voltage range of 1.6-3.6V. The SDLV Memory Cards will be labeled differently then SD Memory Cards.

SD Memory Card system includes the SD Memory Card (or several cards) the bus and their Host / Application. The Host and Application specification is beyond the scope of this document. The following sections provides an overview of the card, bus topology and communication protocols of the SD Memory Card system. The copyright protection (security) system description is given in “SD Memory Card Security Specification” document.

3.1 Bus Topology

The SD Memory Card system defines two alternative communication protocols: SD and SPI. Applications can choose either one of modes. Mode selection is transparent to the host. The card automatically detects the mode of the reset command and will expect all further communication to be in the same communication mode. Therefore, applications which uses only one communication mode do not have to be aware of the other.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

3.1.1 SD bus

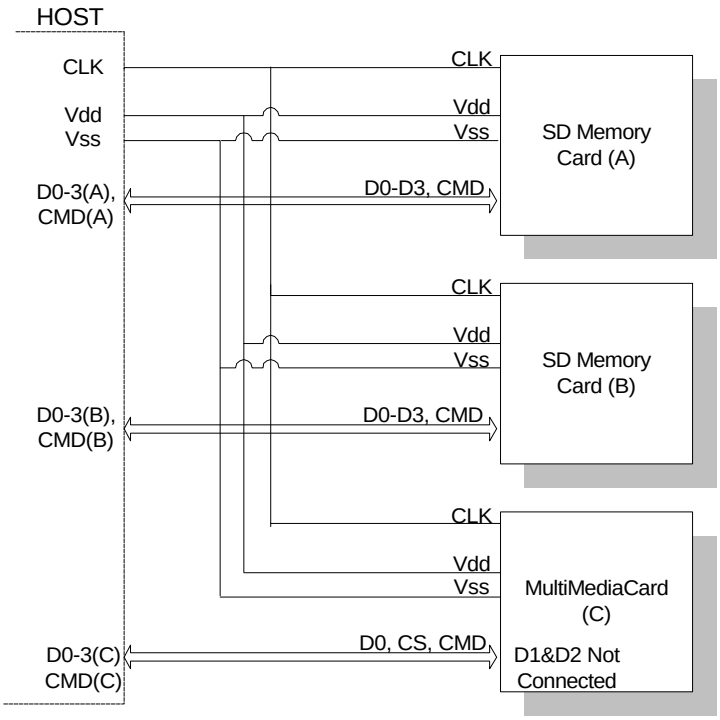


Figure 2: SD Memory Card system bus Topology

The SD bus includes the following signals:

CLK:	Host to card clock signal
CMD:	Bidirectional Command/Response signal
DAT0 - DAT3:	4 Bidirectional data signals.
VDD, VSS1, VSS2:	Power and ground signals.

The SD Memory Card bus has a single master (application), multiple slaves (cards), synchronous star topology (refer to Figure 2). Clock, power and ground signals are common to all cards. Command (CMD) and data (DAT0 - DAT3) signals are dedicated to each card providing continues point to point connection to all the cards.

During initialization process commands are sent to each card individually, allowing the application to detect the cards and assign logical addresses to the physical slots. Data is always sent (received) to (from) each card individually. However, in order to simply the handling of the card stack, after the initialization process, all commands may be sent concurrently to all cards. Addressing information is provided in the command packet.

SD bus allows dynamic configuration of the number of data lines. After power up, by default, the SD Memory Card will use only DAT0 for data transfer. After initialization the host can change the bus width (number of active data lines). This feature allows easy trade off between HW cost and system performance. **Note that while DAT1-DAT3 are not in use, the related Host's DAT lines should be in tri-state (input mode).**

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

3.1.2 SPI bus

The SPI compatible communication mode of the SD Memory Card is designed to communicate with a SPI channel, commonly found in various microcontrollers in the market. The interface is selected during the first reset command after power up and cannot be changed as long as the part is powered on.

The SPI standard defines the physical link only, and not the complete data transfer protocol. The SD Memory Card SPI implementation uses the same command set of the SD mode. From the application point of view, the advantage of the SPI mode is the capability of using an off-the-shelf host, hence reducing the design-in effort to minimum. The disadvantage is the loss of performance, relatively to the SD mode which enables the wide bus option.

The SD Memory Card SPI interface is compatible with SPI hosts available on the market. As any other SPI device the SD Memory Card SPI channel consists of the following four signals:

CS: Host to card Chip Select signal.

CLK: Host to card clock signal

DataIn: Host to card data signal.

DataOut: Card to host data signal.

Another SPI common characteristic are byte transfers, which is implemented in the card as well. All data tokens are multiples of bytes (8 bit) and always byte aligned to the CS signal.

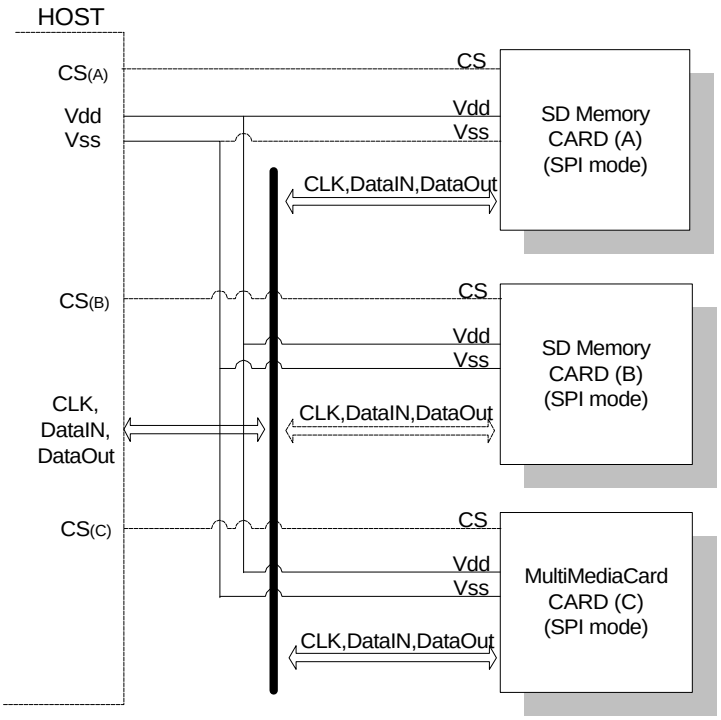


Figure 3: SD Memory Card system (SPI mode) bus topology

The card identification and addressing methods are replaced by a hardware Chip Select (CS) signal. There are no broadcast commands. For every command, a card (slave) is selected by asserting

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

(active low) the CS signal (see Figure 3).

The CS signal must be continuously active for the duration of the SPI transaction (command, response and data). The only exception occurs during card programming, when the host can de-assert the CS signal without affecting the programming process.

The SPI interface uses the 7 out of the SD 9 signals (DAT1 and DAT 2 are not used, DAT3 is the CS signal) of the SD bus.

3.2 Bus Protocol

3.2.1 SD bus

Communication over the SD bus is based on command and data bit streams which are initiated by a start bit and terminated by a stop bit.

- **Command**: a command is a token which starts an operation. A command is sent from the host either to a single card (addressed command) or to all connected cards (broadcast command). A command is transferred serially on the CMD line.
- **Response**: a response is a token which is sent from an addressed card, or (synchronously) from all connected cards, to the host as an answer to a previously received command. A response is transferred serially on the CMD line.
- **Data**: data can be transferred from the card to the host or vice versa. Data is transferred via the data lines.

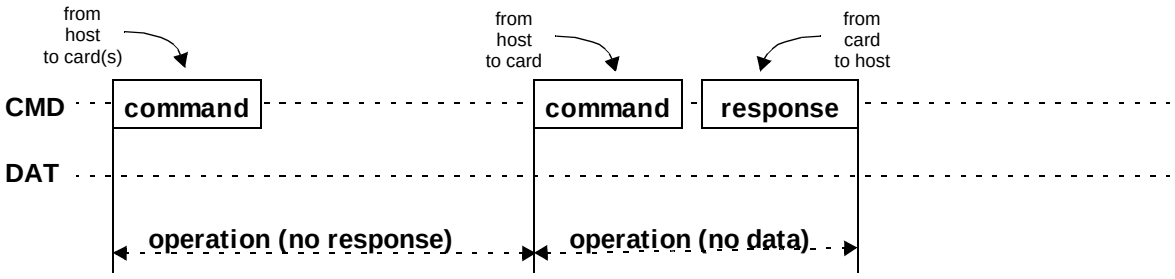


Figure 4: "no response" and "no data" operations

Card addressing is implemented using a session address, assigned to the card during the initialization phase. The structure of commands, responses and data blocks is described in Chapter 4. The basic transaction on the SD bus is the command/response transaction (refer to Figure 4). This type of bus transactions transfer their information directly within the command or response structure. In addition, some operations have a data token.

Data transfers to/from the SD Memory Card are done in blocks. Data blocks always succeeded by CRC bits. Single and multiple block operations are defined. Note that the Multiple Block operation mode is better for faster write operation. A multiple block transmission is terminated when a stop command follows on the CMD line. Data transfer can be configured by the host to use single or multiple data lines.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

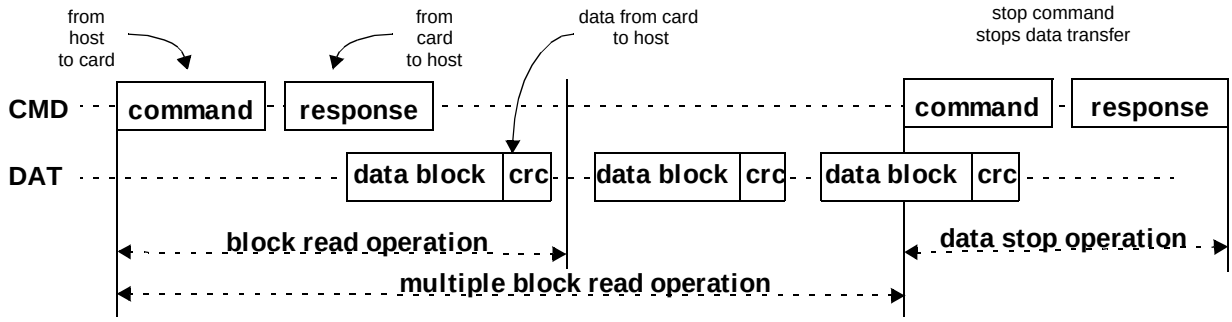


Figure 5: (Multiple) Block read operation

The block write operation uses a simple busy signaling of the write operation duration on the DAT0 data line (see Figure 6) regardless of the number of data lines used for transferring the data.

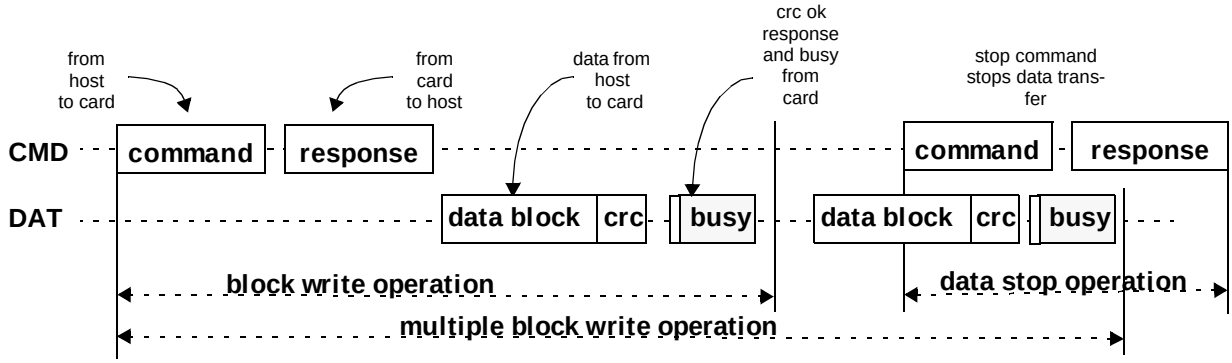


Figure 6: (Multiple) Block write operation

Command tokens have the following coding scheme:

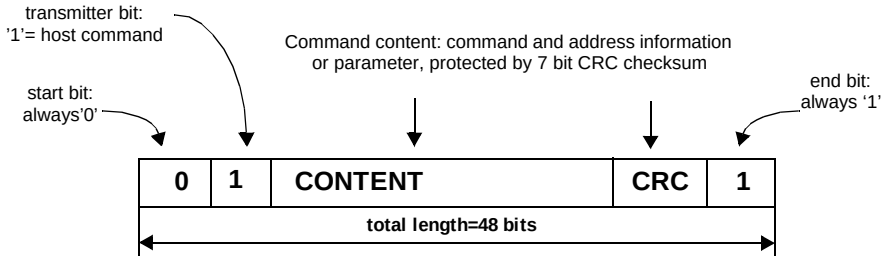


Figure 7: Command token format

Each command token is preceded by a start bit ('0') and succeeded by an end bit ('1'). The total length is 48 bits. Each token is protected by CRC bits so that transmission errors can be detected and the operation may be repeated.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

Response tokens have four coding schemes depending on their content. The token length is either 48 or 136 bits. The detailed commands and response definition is given in Chapter 4.7. The CRC protection algorithm for block data is a 16 bit CCITT polynomial. All used CRC types are described in Chapter 4.5.

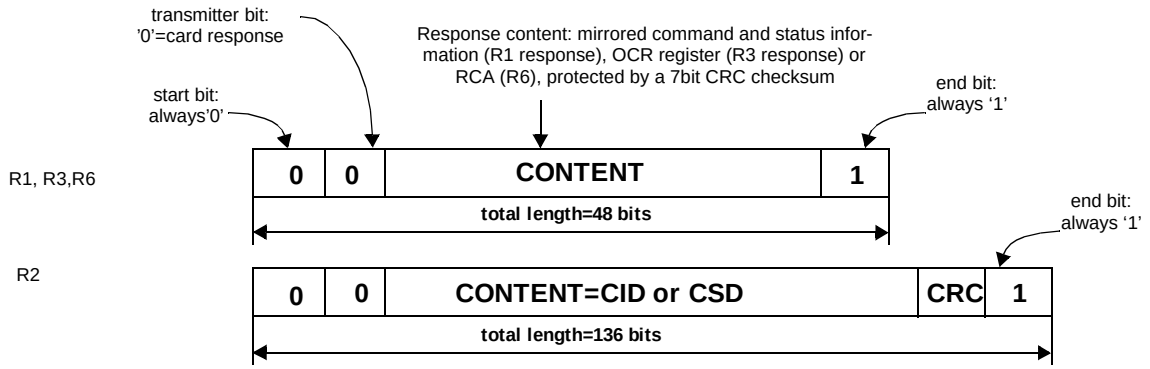


Figure 8: Response token format

In the CMD line the MSB bit is transmitted first the LSB bit is the last.

when the wide bus option is used, the data is transferred 4 bits at a time (refer to Figure 9). Start and end bits, as well as the CRC bits, are transmitted for every one of the DAT lines. CRC bits are calculated and checked for every DAT line individually. The CRC status response and Busy indication will be sent by the card to the host on DAT0 only (DAT1-DAT3 during that period are don't care).

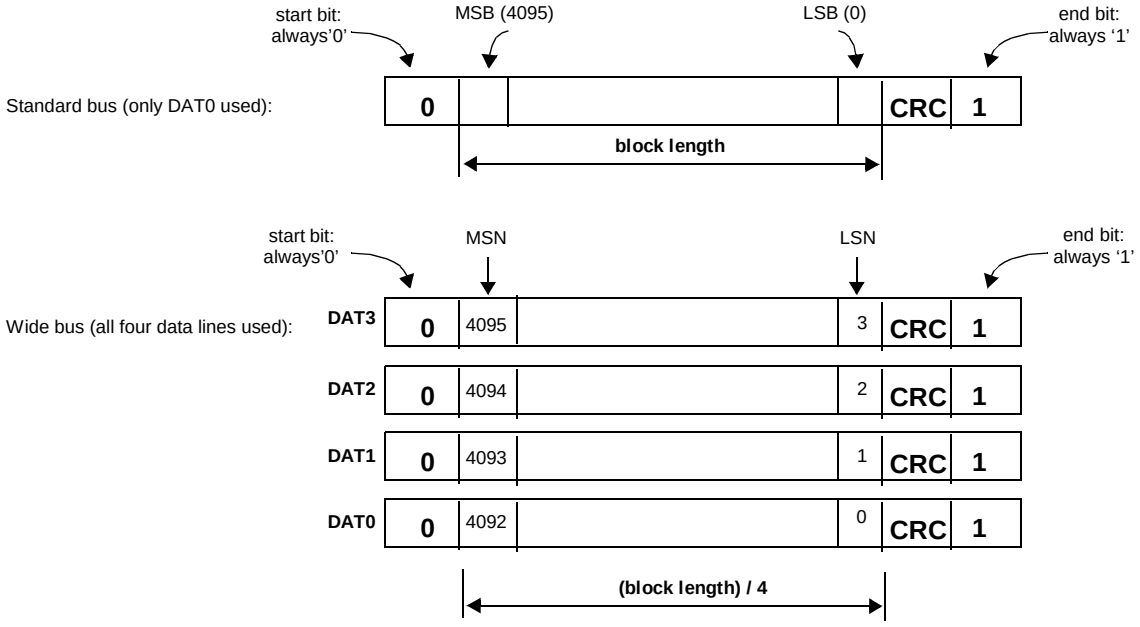


Figure 9: Data packet format

3.2.2 SPI Bus

While the SD channel is based on command and data bit streams which are initiated by a start bit and terminated by a stop bit, the SPI channel is byte oriented. Every command or data block is built of 8-bit bytes and is byte aligned to the CS signal (i.e. the length is a multiple of 8 clock cycles).

Similar to the SD protocol, the SPI messages consist of command, response and data-block tokens All communication between host and cards is controlled by the host (master). The host starts every bus transaction by asserting the CS signal low.

The response behavior in the SPI mode differs from the SD mode in the following three aspects:

- The selected card always responds to the command.
- Two new (8 & 16 bit) response structure is used
- When the card encounters a data retrieval problem, it will respond with an error response (which replaces the expected data block) rather than by a time-out as in the SD mode.

In addition to the command response, every data block sent to the card during write operations will be responded with a special data response token.

- **Data Read**

Single and multiple block read commands are supported in SPI mode. However, in order to comply with the SPI industry standard, only two (unidirectional) signal are used (refer to Chapter 10). Upon reception of a valid read command the card will respond with a response token followed by a data token of the length defined in a previous SET_BLOCKLEN (CMD16) command. A multiple block read operation is terminated, similar to the SD protocol, with the STOP_TRANSMISSION command.

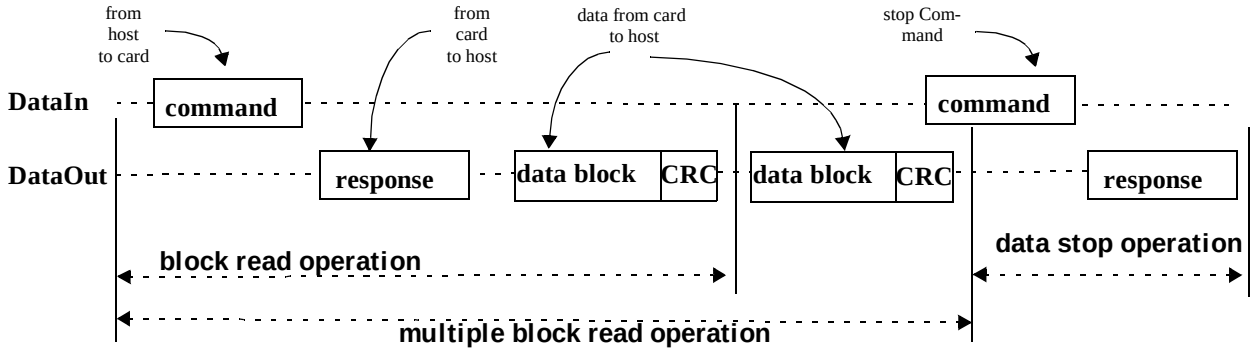


Figure 10: Read operation

A valid data block is suffixed with a 16 bit CRC generated by the standard CCITT polynomial $x^{16}+x^{12}+x^5+1$.

In case of a data retrieval error, the card will not transmit any data. Instead, a special data error token will be sent to the host. Figure 11 shows a data read operation which terminated with an error

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

token rather than a data block.

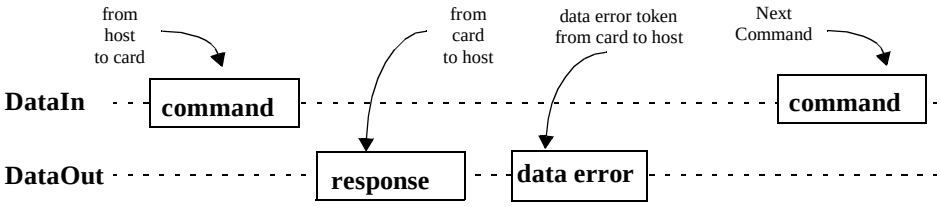


Figure 11: Read operation - data error

- **Data Write**

Single and multiple block write operations are supported in SPI mode. Upon reception of a valid write command, the card will respond with a response token and will wait for a data block to be sent from the host. CRC suffix, block length and start address restrictions are identical to the read operation (see Figure 12).

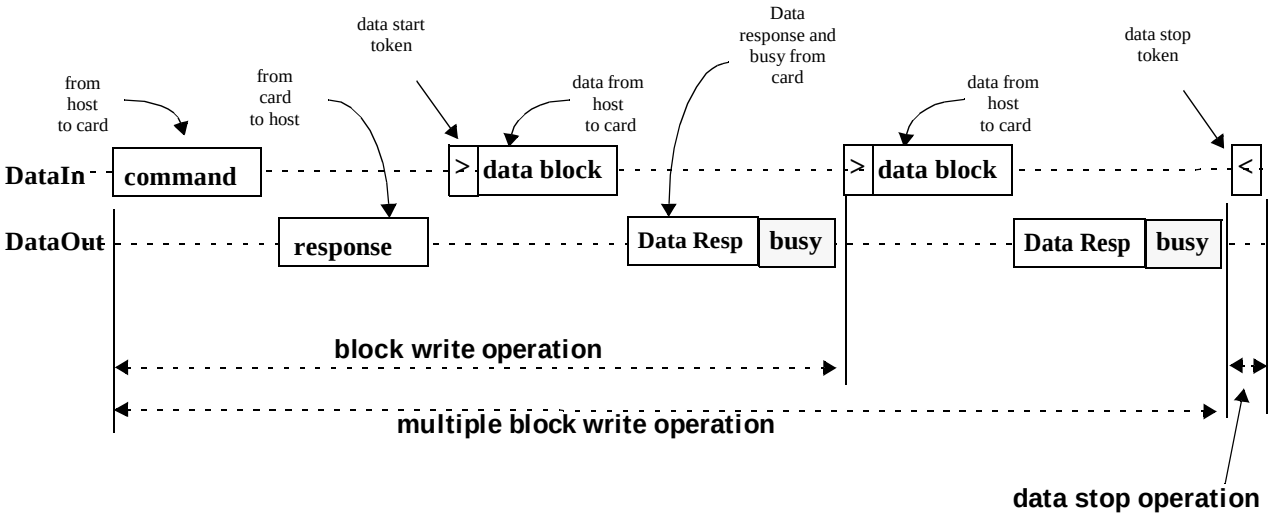


Figure 12: Write operation

After a data block has been received, the card will respond with a data-response token. If the data block has been received without errors, it will be programmed. As long as the card is busy programming, a continuous stream of busy tokens will be sent to the host (effectively holding the DataOut line low).

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

3.3 SD Memory Card - Pins and Registers

The SD Memory Card has the form factor 24mm x 32mm x 2.1mm.

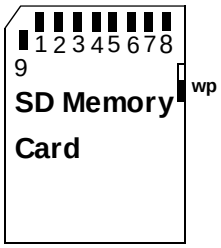


Figure 13: SD Memory Card shape and interface (top view)

Figure 13 describes the general idea of the shape and interface contacts of SD Memory Card. The detailed physical dimensions and mechanical description is given in chapter 9.

The following table defines the card contacts:

Pin #	SD Mode			SPI Mode		
	Name	Type[1]	Description	Name	Type	Description
1	CD/DAT3[2]	I/O/PP[3]	Card Detect / Data Line [Bit 3]	CS	I	Chip Select (neg true)
2	CMD	PP	Command/Response	DI	I	Data In
3	V_{SS1}	S	Supply voltage ground	VSS	S	Supply voltage ground
4	V_{DD}	S	Supply voltage	VDD	S	Supply voltage
5	CLK	I	Clock	SCLK	I	Clock
6	V_{SS2}	S	Supply voltage ground	VSS2	S	Supply voltage ground
7	DAT0	I/O/PP	Data Line [Bit 0]	DO	O/PP	Data Out
8	DAT1	I/O/PP	Data Line [Bit 1]	RSV		
9	DAT2	I/O/PP	Data Line [Bit 2]	RSV		

Table 1: SD memory Card Pad Assignment

1) S: power supply; I: input; O: output using push-pull drivers; PP: I/O using push-pull drivers;
2) The extended DAT lines (DAT1-DAT3) are input on power up. They start to operate as DAT lines after SET_BUS_WIDTH command. The Host shall keep its own DAT1-DAT3 lines in input mode, as well, while they are not used. It is defined so, in order to keep compatibility to MultiMediaCards.
3) After power up this line is input with 50KOhm pull-up (can be used for card detection or SPI mode selection). The pull-up should be disconnected by the user, during regular data transfer, with SET_CLR_CARD_DETECT (ACMD42) command

Each card has a set of information registers (see also Chapter 5 in the SD Memory Card Physical

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

Layer Specification):

Name	Width	Description
CID	128	Card identification number; card individual number for identification. **Mandatory**.
RCA[1]	16	Relative card address; local system address of a card, dynamically suggested by the card and approved by the host during initialization. **Mandatory**.
DSR	16	Driver Stage Register; to configure the card's output drivers. **Optional.**
CSD	128	Card Specific Data; information about the card operation conditions. **Mandatory**
SCR	64	SD Configuration Register; information about the SD Memory Card's Special Features capabilities. **Mandatory**
OCR	32	Operation condition register. **Mandatory.**

Table 2: SD Memory Card registers

1) RCA register is not used (available) in SPI mode.

The host may reset the cards by switching the power supply off and on again. Each card shall have its own power-on detection circuitry which puts the card into a defined state after the power-on. No explicit reset signal is necessary. The cards can also be reset by sending the GO_IDLE (CMD0) command.

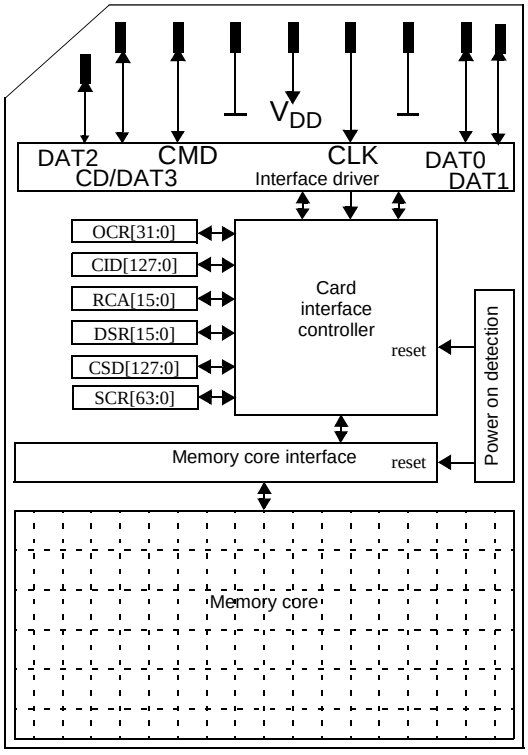


Figure 14: SD Memory Card architecture

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

3.4 Compatibility to MultiMediaCard

The SD Memory Card protocol is designed to be a super-set of the MultiMediaCard protocol[1]. The main additions are the wide bus option and the content protection support (refer to Table 3 for details). It is very easy to design host systems, capable of supporting both types of cards. The intent is to enable application designers to make use of the exiting install base of MultiMediaCard, unless the application cannot do without either the fast data transfer rate (wide bus), or content security.

	SD Memory Card	MultiMediaCard	Comments
Bus width	1bit or 4 bits	1 bit only	
System bus organization (multiple cards connection)	Star Topology	Bus Topology	
Initialization commands	CMD0 ACMD41 CMD2 CMD3	CMD0 CMD1 CMD2 CMD3	In MultiMediaCard CMD1 and CMD2 are sent concurrently from all cards using the OD drivers. In SD memory card each card is reset and identified indepen-dently and the RCA (CMD3) is assigned by the card.
Operation Commands	SEND_NUM_WR_BLOCK SET_WR_BLK_COUNT		two new command for improved write performance (ACMD23, ACMD22) are supported in SD.
Maximum Clock rate	25MHz	20MHz	
Copyright protection	supported (optional in Read Only type of cards)	not supported	
Write protect switch	supported	not supported	When MultiMediaCard is inserted into an SD Memory Card slot, it always perceived as non-write-protected card (window closed)
Feature of Pin #1	Has a card internal pull-up resistor	Defined as "not connected"	In SD Memory Card pin #1 may be used for card detection.
CSD Structure	Different from MMC (mainly Sector Size/ Groups is different)		
CID Structure	Different from MMC		In SD the Manufacturing date field is bigger. Product ID field is smaller.
SPI R/W Multiple Block	Supported	Not supported	
Stream R/W mode	not supported	supported (optional)	SD Memory Card supports only single and multiple block read/ write operations.

1. As defined in the MultiMediaCard system specification V2.11. Published by the MMCA technical commitee.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

	SD Memory Card	MultiMediaCard	Comments
I/O Mode	not supported	supported (optional)	I/O (Interrupt) mode is not supported in SD Memory Card.

Table 3: Differences between SD Memory Card and MultiMediaCard Feature set

The only difference (as opposed to addition) between the SD Memory Card and the MultiMediaCard is the bus topology and initialization protocol. While the MultiMediaCard stack is connected on the same bus and being identified using synchronous transmission of Open-drain outputs, each SD Memory Card has an independent point-to-point connection to the host, and the cards are identified serially, one at a time (refer to Table 4 for a command set comparison between SD Memory Card and MultiMediaCard. Detailed description of the SD Memory card protocol commands can be found in Chapter 4).

The initializing procedure in the SD protocol is defined to successfully identify either a MultiMediaCard or a SD Memory Card, which ever is currently connected on the bus. After card detection, the host executes the initializing procedure and ends up with an identified card of a known type.

Once the card is initialized the application can determine the card capabilities by querying the various configuration registers, and decide whether or not to use it.

The physical dimension of the SD Memory Card is thicker than MultiMediaCard (2.1mm vs. 1.4mm; refer to Chapter 9) but it is defined in such a way that a MultiMediaCard can be inserted into SD Memory Card socket. Note that because of the small differences between the mechanical definition of the pads layout of MultiMediaCard it is required that the SD Host will set its own DAT1-DAT3 lines to be in Input Mode (Tri-State) while they are not in use.

Three different card detect mechanisms are defined for the SD Memory Card (e.g. mechanical insertion which can be sensed using the WP switch, Electrical insertion which can be sensed using the pull-up resistor on DAT3 and periodical attempts to initialize the card). Since some of these methods may be not relevant (or behave differently) for the MultiMediaCard, it is recommended not to depend on the preemptive card detects methods only. The host should implement a polling mechanism or allow the operator to request card identification.

Class	CMD	SD Memory Card	MultiMedia Card	Comment
Class 0	CMD0	CMD0 (Mandatory)	CMD0	Same command.
	CMD1	Reserved	CMD1	In SD Memory Card ACMD41 is used instead of CMD1
	CMD2	CMD2 (Mandatory)	CMD2	Similar command except buffer type used to transmit to response of the card. (SD Memory Card: push-pull, MultiMediaCard: open-drain)
	CMD3	CMD3 (Mandatory)	CMD3	In both protocols this command is used to assign a logical address to the card. While In MultiMediaCard the host assigned the address, in SD memory Card it is the responsibility of the card.
	CMD 4-10	CMD4-10 (Mandatory)	CMD4-10	Same commands.

Table 4: Commands comparison table.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

Class	CMD	SD Memory Card	MultiMedia Card	Comment
Class 1	CMD11	Reserved	CMD11	SD Memory Card doesn't support stream access.
Class 0	CMD12 -15	CMD12-15 (Mandatory)	CMD12-15	Same commands.
Class 2	CMD16 -19	CMD16-19 (Mandatory)	CMD16-19	Same commands.
Class 3	CMD20	Reserved	CMD20	SD Memory Card dose not support stream access.
	CMD21 -23	Reserved	Reserved	All reserved.
Class 4	CMD24 -27	CMD24-27 (Mandatory for Writable card)	CMD24-27	Same commands.
Class 6	CMD28 -31	CMD28-31 (Optional)	CMD28-31	Same commands.
Class 5	CMD32 -33	CMD32-33 (Mandatory for Writable card)	CMD32-33	Same commands.
	CMD34 -37	Reserved	CMD34-37	SD Memory Card dose not support TAG and Erase Group commands
	CMD38	CMD38 (Mandatory for Writable card)	CMD38	Same Command.
Class 9	CMD39 -41	Reserved	CMD39-41	SD Memory Card dose not support I/O mode.
Class 7	CMD42 -54	CMD42-54 (Optional)	CMD42-54	Same commands.
Class 8	CMD55 -56	CMD55-56 (Mandatory)	CMD55-56	Same commands.
	CMD60 -63	CMD60-63 (reserved for manufac-turer)	CMD60-63 (reserved for manu-facturer)	

Table 4: Commands comparison table.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

4 SD Memory Card Functional Description

4.1 General

All communication between host and cards is controlled by the host (master). The host sends commands of two types: broadcast and addressed (point-to-point) commands.

- **Broadcast commands**

Broadcast commands are intended for all cards. Some of these commands require a response.

- **Addressed (point-to-point) commands**

The addressed commands are sent to the addressed card and cause a response from this card.

A general overview of the command flow is shown in Figure 15 for the card identification mode and in Figure for the data transfer mode. The commands are listed in the command tables (Table 9 - Table 16). The dependencies between current state, received command and following state are listed in Table 17. In the following sections, the different card operation modes will be described first. Thereafter, the restrictions for controlling the clock signal are defined. All SD Memory Card commands together with the corresponding responses, state transitions, error conditions and timings are presented in the succeeding sections.

Two operation modes are defined for the SD Memory Card system (host and cards):

- **Card identification mode**
  The host will be in card identification mode after reset and while it is looking for new cards on the bus. Cards will be in this mode after reset until the SEND_RCA command (CMD3) is received (in case of MultiMediaCard - SET_RCA command).

- **Data transfer mode**
  Cards will enter data transfer mode once their RCA is first published. The host will enter data transfer mode after identifying all the cards on the bus.

The following table shows the dependencies between operation modes and card states. Each state in the SD Memory Card state diagram (see Figure 15 and Figure) is associated with one operation mode:

Card state	Operation mode
Inactive State	inactive
Idle State	card identification mode
Ready State	
Identification State	
Stand-by State	data transfer mode
Transfer State	
Sending-data State	
Receive-data State	
Programming State	
Disconnect State	

Table 5: Overview of Card States vs. Operation modes

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

4.2 Card Identification Mode

While in card identification mode the host resets all the cards that are in card identification mode, validates operation voltage range, identifies cards and asks them to publish Relative Card Address (RCA). This operation is done to each card separately on its own CMD line. All data communication in the Card Identification Mode uses the command line (CMD) only.

4.2.1 Card Reset

The command GO_IDLE_STATE (CMD0) is the software reset command and sets each card into *Idle State* regardless of the current card state. Cards in *Inactive State* are not affected by this command.

After power-on by the host, all cards are in *Idle State*, including the cards that have been in *Inactive State* before.

After power-on or CMD0, all cards' CMD lines are in input mode, waiting for start bit of the next command. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

4.2.2 Operating Voltage Range Validation

All cards shall be able to establish communication with the host using any operating voltage in the maximal allowed voltage range specified in this standard (see Chapter 6.6). However, the supported minimum and maximum values for V_{DD} are defined in the Operation Conditions Register (OCR) and may not cover the whole range. Cards that store the CID and CSD data in the payload memory would be able to communicate these information only under data transfer V_{DD} conditions. That means if host and card have non compatible V_{DD} ranges, the card will not be able to complete the identification cycle, nor to send CSD data.

Therefore, a special command SD_SEND_OP_COND (ACMD41) is designed to provide SD Memory Card hosts with a mechanism to identify and reject cards which do not match the V_{DD} range desired by the host. This is accomplished by the host sending the required V_{DD} voltage window as the operand of this command (See Chapter 5.1). Cards which can not perform data transfer in the specified range must discard themselves from further bus operations and go into *Inactive State*. The levels in the OCR register shall be defined accordingly (see Chapter 5.1). Note that ACMD41 is application specific command, therefore APP_CMD (CMD55) shall always precede ACMD41. The RCA to be used for CMD55 in *idle_state* shall be the card's default RCA = 0x0000.

The MultiMediaCard will not respond to ACMD41 (actually it will not respond to APP_CMD - CMD55, that preceding it). MultiMediaCard shall be initialized as per the MultiMediaCard spec, using SEND_OP_COND command (CMD1 of MultiMediaCard). The host should ignore an ILLEGAL_COMMAND status in the MultiMediaCard response to CMD3, as it is a residue of ACMD41 which is invalid in MultiMediaCard (CMD0, 1, 2 do not clear the status register). Actually, ACMD41 and CMD1 will be used by the host to distinguish between MultiMediaCard and SD Memory Cards in a system.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

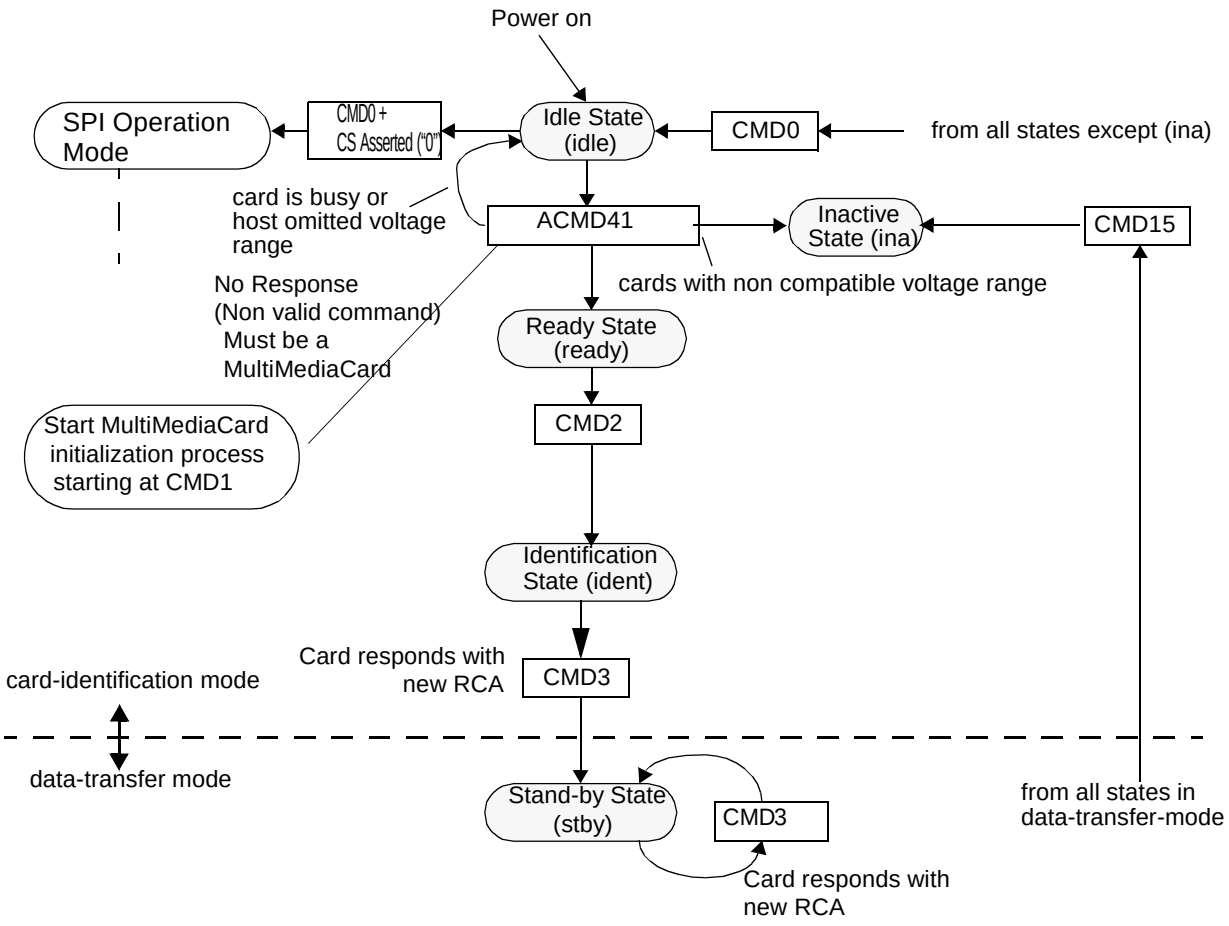


Figure 15: SD Memory Card state diagram (card identification mode)

By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the *Inactive State*. This query should be used if the host is able to select a common voltage range or if a notification to the application of non usable cards in the stack is desired. Afterwards, the host must choose a voltage for operation and reissue ACMD41 with this condition, sending incompatible cards into the *Inactive State*.

The busy bit in the ACMD41 response can be used by a card to tell the host that it is still working on its power-up/reset procedure (e.g. downloading the register information from memory field) and is not ready yet for communication. In this case the host must repeat ACMD41 until the busy bit is cleared.

During the initialization procedure, the host is not allowed to change the operating voltage range. Such changes shall be ignored by the card. If there is a real change in the operating conditions, the host must reset the card stack (sending CMD0 to all cards) and restart the initialization procedure. However, for accessing also the cards being already in *Inactive State*, a hard reset must be done by switching the power supply off and on.

The command GO_INACTIVE_STATE (CMD15) can be used to send an addressed card into the

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

Inactive State. This command is used when the host explicitly wants to deactivate a card (e.g. host is changing V_{DD} into a range which is known to be not supported by this card).

4.2.3 Card Identification Process

The host starts the card identification process with the identification clock rate f_{OD} (see Chapter 6.8). In SD Memory Card the CMD line output drives are push-pull drivers.

After the bus is activated the host will request the cards to send their valid operation conditions (ACMD41 preceding with APP_CMD - CMD55 with RCA=0x0000). The response to ACMD41 is the operation condition register of the card. The same command shall be send to all of the new cards in the system. Incompatible cards are sent into *Inactive State*. The host then issues the command ALL_SEND_CID (CMD2), to each card to get its unique card identification (CID) number. Card that is unidentified (i.e. which is in *Ready State*) sends its CID number as the response (on the CMD line). After the CID was sent by the card it goes into *Identification State*. Thereafter, the host issues CMD3 (SEND_RELATIVE_ADDR) asks the card to publish a new relative card address (RCA), which is shorter than CID and which will be used to address the card in the future data transfer mode (typically with a higher clock rate than f_{OD}). Once the RCA is received the card state changes to the *Stand-by State.* At this point, if the host wants that the card will have another RCA number, it may ask the card to publish a new number by sending another SEND_RELATIVE_ADDR command to the card. The last published RCA is the actual RCA number of the card.

The host repeats the identification process, i.e. the cycles with CMD2 and CMD3 for each card in the system.

After all the SD Memory Cards were initialized the host shall initialize the MultiMediaCards that are in the system (if any), using the CMD2 and CMD3 as given in the MultiMediaCard spec. Note that in the SD system all the cards are connected separately so each MultiMediaCard shall be initialized individually.

4.3 Data Transfer Mode

Until the end of Card Identification Mode the host must remain at f_{OD} frequency because some cards may have operating frequency restrictions during the card identification mode. In Data Transfer Mode the host may operate the card in f_{PP} frequency range (see chapter 6.8). The host issues SEND_CSD (CMD9) to obtain the Card Specific Data (CSD register), e.g. block length, card storage capacity, etc.

The broadcast command SET_DSR (CMD4) configures the driver stages of all identified cards. It programs their DSR registers corresponding to the application bus layout (length) and the number of cards on the bus and the data transfer frequency. The clock rate is also switched from f_{OD} to f_{PP} at that point. SET_DSR command is an option for the card and the host.

CMD7 is used to select one card and put it into the *Transfer State*. Only one card can be in the *Transfer State* at a given time. If a previously selected card is in the *Transfer State* its connection with the host is released and it will move back to the *Stand-by State*. When CMD7 is issued with the reserved relative card address “0x0000”, all cards are put back to *Stand-by State* (Note that it is the

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

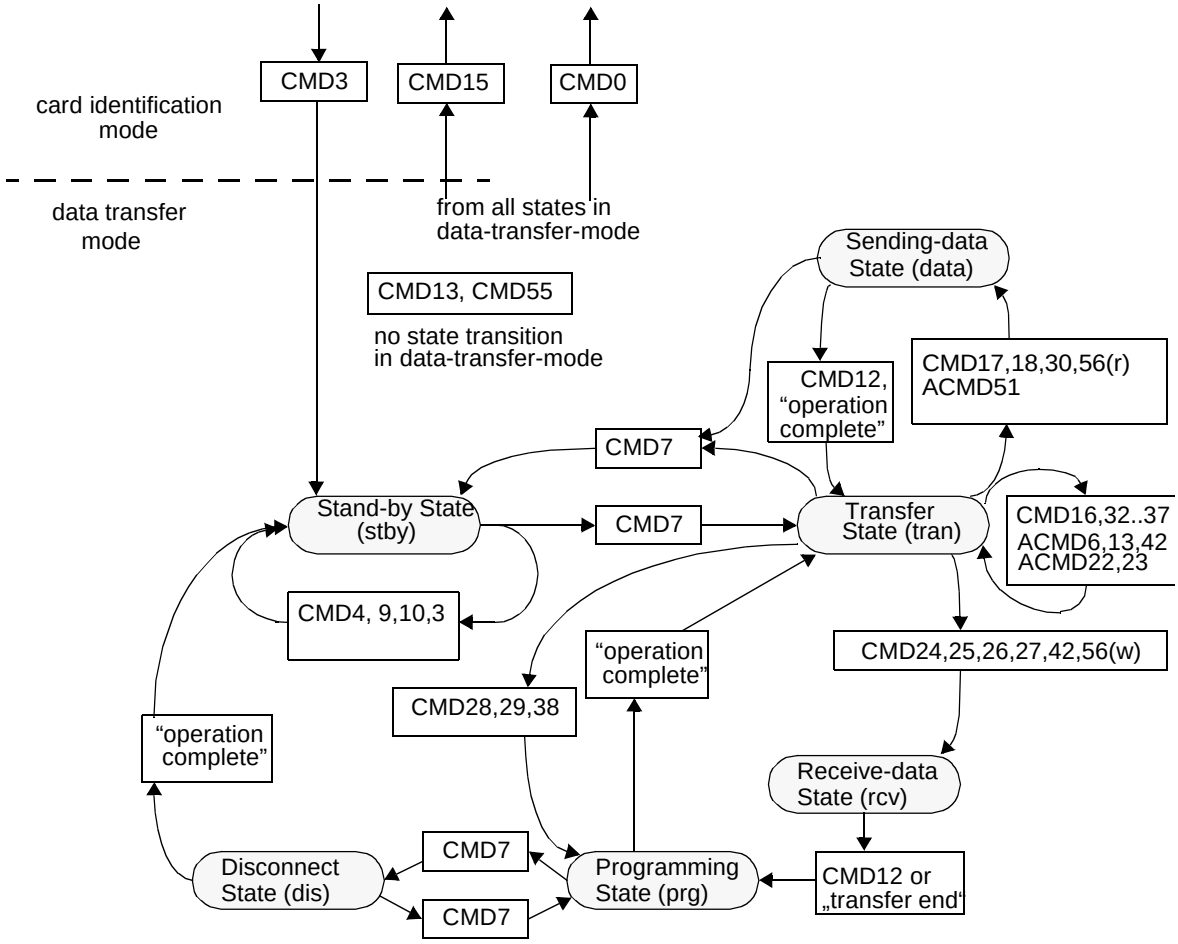


Figure 16: SD Memory Card state diagram (data transfer mode)

responsibility of the Host to reserve the RCA=0 for card de-selection - refer to Table 9, CMD7 description). This may be used before identifying new cards without resetting other already registered cards. Cards which already have an RCA do not respond to identification commands (CMD41, CMD2, CMD3, see Chapter 4.2.3) in this state.

Important Note: The card de-selection is done if certain card gets CMD7 with un-matched RCA. That happens automatically if selection is done to another card and the CMD lines are common. So, in SD Memory Card system it will be the responsibility of the host either to work with common CMD line (after initialization is done) - in that case the card de-selection will be done automatically (as in MultiMediaCard system) or if the CMD lines are separate then the host shall be aware to the necessity to de-select cards.

All data communication in the Data Transfer Mode is point-to point between the host and the selected card (using addressed commands). All addressed commands get acknowledged by a response on the CMD line.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

The relationship between the various data transfer modes is summarized below (see Figure):

- All data read commands can be aborted any time by the stop command (CMD12). The data transfer will terminate and the card will return to the *Transfer State.* The read commands are: block read (CMD17), multiple block read (CMD18), send write protect (CMD30), send scr (ACMD51) and general command in read mode (CMD56).
- All data write commands can be aborted any time by the stop command (CMD12). The write commands must be stopped prior to deselecting the card by CMD7. The write commands are: block write (CMD24 and CMD25), write CID (CMD26), write CSD (CMD27), lock/unlock command (CMD42) and general command in write mode (CMD56).
- As soon as the data transfer is completed, the card will exit the data write state and move either to the *Programming State* (transfer is successful) or *Transfer State* (transfer failed).
- If a block write operation is stopped and the block length and CRC of the last block are valid, the data will be programmed.
- The card may provide buffering for block write. This means that the next block can be sent to the card while the previous is being programmed.
  If all write buffers are full, and as long as the card is in *Programming State* (see SD Memory Card state diagram Figure), the DAT0 line will be kept low (BUSY).
- There is no buffering option for write CSD, write CID, write protection and erase. This means that while the card is busy servicing any one of these commands, no other data transfer commands will be accepted. DAT0 line will be kept low as long as the card is busy and in the *Programming State*. Actually if the CMD and DAT0 lines of the cards are kept separated and the host keep the busy DAT0 line disconnected from the other DAT0 lines (of the other cards) the host may access the other cards while the card is in busy.
- Parameter set commands are *not* allowed while card is programming.
  Parameter set commands are: set block length (CMD16), erase block start (CMD32) and erase block end (CMD33).
- Read commands are *not* allowed while card is programming.
- Moving another card from *Stand-by* to *Transfer State* (using CMD7) will not terminate erase and programming operations. The card will switch to the *Disconnect State* and will release the DAT line.
- A card can be reselected while in the *Disconnect State*, using CMD7. In this case the card will move to the *Programming State* and reactivate the busy indication.
- Resetting a card (using CMD0 or CMD15) will terminate any pending or active programming operation. This may destroy the data contents on the card. It is the host's responsibility to prevent this.

4.3.1 Wide Bus Selection/Deselection

Wide Bus (4 bit bus width) operation mode may be selected/deselected using ACMD6. The default bus width after power up or GO_IDLE (CMD0) is 1 bit bus width. ACMD6 command is valid in '*tran state*' only. That means that the bus width may be changed only after a card was selected (CMD7).

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

4.3.2 Data Read

The DAT bus line level is high by the pull-up when no data is transmitted. A transmitted data block consists of start bits (1 or 4 bits LOW), followed by a continuous data stream. The data stream contains the payload data (and error correction bits if an off-card ECC is used). The data stream ends with end bits (1 or 4 bits HIGH) (see Figure 25-Figure 27). The data transmission is synchronous to the clock signal. The payload for block oriented data transfer is protected by 1 or 4 bits CRC check sum (see Chapter 3.2).

The Read operation from SD Memory Card may be interrupted by turning the power off. The SD Memory Card ensures that data is not destroyed during all the conditions except write or erase operations issued by the host even in the event of sudden shut down or removal.

- **Block Read**

Block read is block oriented data transfer. The basic unit of data transfer is a block whose maximum size is defined in the CSD (READ_BL_LEN). Smaller blocks whose starting and ending address are entirely contained within one physical block (as defined by READ_BL_LEN) may also be transmitted. It is a mandatory requirement that SD Memory Card shall have a capability to transfer blocks of 512 Bytes.

A CRC is appended to the end of each block ensuring data transfer integrity. CMD17 (READ_SINGLE_BLOCK) initiates a block read and after completing the transfer, the card returns to the *Transfer State*. CMD18 (READ_MULTIPLE_BLOCK) starts a transfer of several consecutive blocks. Blocks will be continuously transferred until a STOP_TRANSMISSION command (CMD12) is issued. The stop command has an execution delay due to the serial command transmission. The data transfer stops after the end bit of the stop command.

If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed, the card shall detect a block misalignment at the beginning of the first misaligned block, set the ADDRESS_ERROR error bit in the status register, abort transmission and wait in the *Data State* for a stop command.

4.3.3 Data Write

The data transfer format is similar to the data read format. For block oriented write data transfer, the CRC check bits are added to each data block. The card performs 1 or 4 bits CRC parity check (see Chapter 7.2) for each received data block prior to the write operation. By this mechanism, writing of erroneously transferred data can be prevented.

- **Block Write**

During block write (CMD24 - 27,42,56(w)) one or more blocks of data are transferred from the host to the card with 1 or 4 bits CRC appended to the end of each block by the host. A card supporting block write shall always be able to accept a block of data defined by WRITE_BL_LEN and its 512bytes derivatives (for example: If write block length=1024bytes then write blocks of 1024 and 512bytes are supported). If WRITE_BL_PARTIAL is allowed (=1) then smaller blocks, up to resolution of one byte, can be used as well. If the CRC fails, the card shall indicate the failure on the DAT line (see below); the transferred data will be discarded and not written, and all further transmitted blocks (in multiple block write mode) will be ignored.

Multiple block write command shall be used rather than continuous single write command to make faster write operation.

If the host uses partial blocks whose accumulated length is not block aligned and block misalign-

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

ment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card shall detect the block misalignment error and abort programming before the beginning of the first misaligned block. The card shall set the ADDRESS_ERROR error bit in the status register, and while ignoring all further data transfer, wait in the *Receive-data-State* for a stop command.

The write operation shall also be aborted if the host tries to write over a write protected area. In this case, however, the card shall set the WP_VIOLATION bit.

Programming of the CID and CSD registers does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card will report an error and not change any register contents.

Some cards may require long and unpredictable times to write a block of data. After receiving a block of data and completing the CRC check, the card will begin writing and hold the DAT0 line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host may poll the status of the card with a SEND_STATUS command (CMD13) at any time, and the card will respond with its status. The status bit READY_FOR_DATA indicates whether the card can accept new data or whether the write process is still in progress). The host may deselect the card by issuing CMD7 (to select a different card) which will displace the card into the *Disconnect State* and release the DAT line without interrupting the write operation. When reselecting the card, it will reactivate busy indication by pulling DAT to low if programming is still in progress and the write buffer is unavailable. Actually, the host may perform simultaneous write operation to several cards with interleaving process. The interleaving process can be done by accessing each card separately while other cards are in busy. This process can be done by proper CMD and DAT0-3 line manipulations (disconnection of busy cards).

- **Pre-erase setting prior to a multiple block write operation**

Setting a number of write blocks to be pre_erased (ACMD23) will make a following Multiple Block Write operation faster compared to the same operation without preceding ACMD23. The host will use this command to define how many number of write blocks are going to be send in the next write operation. If the host will terminate the write operation (Using stop transmission) before all the data blocks sent to the card the content of the remaining write blocks is undefined(can be either erased or still have the old data). If the host will send more number of write blocks than defined in ACMD23 the card will erase block one by one(as new data is received). This number will be reset to the default(=1) value after Multiple Blocks Write operation.

It is recommended using this command preceding CMD25, some of the cards will be faster for Multiple Write Blocks operation. Note that The host must send ACMD23 just before WRITE command if the host wants to use the pre-erase feature. If not, pre-erase-count might be cleared automatically when another commands (ex: Security Aplication Commands) are executed.

- **Send Number of Written Blocks**

Systems that use PipeLine mechanism for data buffers management are, in some cases, unable to determine which block was the last to be well written to the flash if an error occurs in the middle of a Multiple Blocks Write operation. The card will respond to ACMD22 with the number of well written blocks.

4.3.4 Erase

It is desirable to erase many write blocks simultaneously in order to enhance the data throughput.

Identification of these write blocks is accomplished with the ERASE_WR_BLK_START(CMD32), ERASE_WR_BLK_END(CMD33) commands.

The host must adhere to the following command sequence: ERASE_WR_BLK_START, ERASE_WR_BLK_END and ERASE (CMD38).

If an erase (CMD38) or address setting (CMD32, 33) command is received out of sequence, the card shall set the ERASE_SEQ_ERROR bit in the status register and reset the whole sequence.

If an out of sequence command (except SEND_STATUS) is received, the card shall set the ERASE_RESET status bit in the status register, reset the erase sequence and execute the last command.

If the erase range includes write protected sectors, they shall be left intact and only the non protected sectors shall be erased. The WP_ERASE_SKIP status bit in the status register shall be set.

The address field in the address setting commands is a write block address in byte units. The card will ignore all LSB's below the WRITE_BLK_LEN (see CSD) size.

As described above for block write, the card will indicate that an erase is in progress by holding DAT0 low. The actual erase time may be quite long, and the host may issue CMD7 to deselect the card or perform card disconnection, as described in the Block Write section, above.

The data at the card after an erase operation is either '0' or '1', depends on the card vendor.

The SCR register bit DATA_STAT_AFTER_ERASE (bit 55) defines whether it is '0' or '1'.

4.3.5 Write Protect Management

Three write protect methods are supported in the SD Memory Card as follows:

- Mechanical write protect switch (Host responsibility only)
- Card internal write protect (Card's responsibility)
- Password protection card lock operation.

• Mechanical Write Protect Switch

A mechanical sliding tablet on the side of the card (refer to the mechanical description Chapter 8) will be used by the user to indicate that a given card is write protected or not. If the sliding tablet is positioned in such a way that the window is open it means that the card is write protected. If the window is close the card is not write protected.

A proper, matched, switch on the socket side will indicate to the host that the card is write protected or not. It is the responsibility of the host to protect the card. The position of the write protect switch is un-known to the internal circuitry of the card.

• Card's Internal Write Protection (Optional)

Card data may be protected against either erase or write. The entire card may be permanently write protected by the manufacturer or content provider by setting the permanent or temporary write protect bits in the CSD. For cards which support write protection of groups of sectors by setting the WP_GRP_ENABLE bit in the CSD, portions of the data may be protected (in units of WP_GRP_SIZE sectors as specified in the CSD), and the write protection may be changed by the application. The SET_WRITE_PROT command sets the write protection of the addressed write-protect group, and the CLR_WRITE_PROT command clears the write protection of the addressed write-protect group.

The SEND_WRITE_PROT command is similar to a single block read command. The card shall

send a data block containing 32 write protection bits (representing 32 write protect groups starting at the specified address) followed by 16 CRC bits. The address field in the write protect commands is a group address in byte units. The card will ignore all LSB's below the group size.

The Password Card Lock protection is described in the following section.

4.3.6 Card Lock/Unlock Operation (Optional)

The password protection feature enables the host to lock a card while providing a password, which later will be used for unlocking the card. The password and its size is kept in an 128 bit PWD and 8 bit PWD_LEN registers, respectively. These registers are non-volatile so that a power cycle will not erase them.

Locked cards respond to (and execute) all commands in the "basic" command class (class 0), ACMD41 and "lock card" command class. Thus the host is allowed to reset, initialize, select, query for status, etc., but not to access data on the card. If the password was previously set (the value of PWD_LEN is not '0') will be locked automatically after power on.

Similar to the existing CSD and CID register write commands the lock/unlock command is available in "transfer state" only. This means that it does not include an address argument and the card has to be selected before using it.

The card lock/unlock command has the structure and bus transaction type of a regular single block write command. The transferred data block includes all the required information of the command (password setting mode, PWD itself, card lock/unlock etc.). The following table describes the structure of the command data block.

Byte #	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved				ERASE	LOCK_ UNLOCK	CLR_ PWD	SET_ PWD
1	PWDS_LEN							
2	Password data							
...								
PWDS_LEN + 1								

Table 6: Lock card data structure

- **ERASE:** '1' Defines Forced Erase Operation (all other bits shall be '0') and only the cmd byte is sent.
- **LOCK/UNLOCK**: '1' = Locks the card. '0' = Unlock the card (note that it is valid to set this bit together with SET_PWD but it is not allowed to set it together with CLR_PWD).
- **CLR_PWD**: '1' = Clears PWD.
- **SET_PWD**: '1' = Set new password to PWD
- **PWDS_LEN**: Defines the following password/s length (in bytes). In case of Password change, this field include the total password lengths of old and new passwords.
- **PWD:** In case of set new password contains the new password. In case of password change it contains the old password followed by new password. All other commands it contains the current password.

The data block size shall be defined by the host before it sends the card lock/unlock command. This will allow different password sizes.

The following paragraphs define the various lock/unlock command sequences:

- **Setting the Password**
  - Select a card (CMD7), if not previously selected already
  - Define the block length (CMD16), given by the 8bit card lock/unlock mode, the 8 bits password size (in bytes), and the number of bytes of the new password. In case that a password *replacement* is done, then the block size shall consider that both passwords, the old and the new one, are sent with the command.
  - Send Card Lock/Unlock command with the appropriate data block size on the data line including 16 bit CRC. The data block shall indicate the mode (SET_PWD), the length (PWDS_LEN) and the password itself. In case that a password *replacement* is done, then the length value (PWDS_LEN) shall include both passwords, the old and the new one, and the PWD field shall include the old password (currently used) followed by the new password. Note that card shall handle internally the calculation of the new password length by subscribing the old password length from PWDS_LEN field.
  - In case that the sent old password is not correct (not equal in size and content) then LOCK_UNLOCK_FAILED error bit will be set in the status register and the old password does not change. In case that PWD matches the sent old password then the given new password and its size will be saved in the PWD and PWDS_LEN fields, respectively.

Note that the password length register (PWD_LEN) indicates if a password is currently set. When it equals '0' there is no password set. If the value of PWD_LEN is not equal to zero the card will lock itself after power up. It is possible to lock the card immediately in the current power session by setting the LOCK/UNLOCK bit (while setting the password) or sending additional command for card lock.

- **Reset the Password:**
  - Select a card (CMD7), if not previously selected already
  - Define the block length (CMD16), given by the 8 bit card lock/unlock mode, the 8 bit password size (in bytes), and the number of bytes of the currently used password.
  - Send the card lock/unlock command with the appropriate data block size on the data line including 16 bit CRC. The data block shall indicate the mode CLR_PWD, the length (PWDS_LEN) and the password (PWD) itself (LOCK/UNLOCK bit is don't care). If the PWD and PWD_LEN content match the sent password and its size, then the content of the PWD register is cleared and PWD_LEN is set to 0. If the password is not correct then the LOCK_UNLOCK_FAILED error bit will be set in the status register.

- **Locking a card:**
  - Select a card (CMD7), if not previously selected already
  - Define the block length (CMD16), given by the 8 bit card lock/unlock mode, the 8 bit password size (in bytes), and the number of bytes of the currently used password.
  - Send the card lock/unlock command with the appropriate data block size on the data line including 16 bit CRC. The data block shall indicate the mode LOCK, the length (PWDS_LEN) and the password (PWD) itself .

If the PWD content equals to the sent password then the card will be locked and the card-locked status bit will be set in the status register. If the password is not correct then LOCK_UNLOCK_FAILED error bit will be set in the status register.

Note that it is possible to set the password and to lock the card in the same sequence. In such case the host shall perform all the required steps for setting the password (as described above) including the bit LOCK set while the new password command is sent.

If the password was previously set (PWD_LEN is not '0'), then the card will be locked automatically after power on reset.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

An attempt to lock a locked card or to lock a card that does not have a password will fail and the LOCK_UNLOCK_FAILED error bit will be set in the status register.

- **Unlocking the card:**
  - Select a card (CMD7), if not previously selected already.
  - Define the block length (CMD16), given by the 8 bit card lock/unlock mode, the 8 bit password size (in bytes), and the number of bytes of the currently used password.
  - Send the card lock/unlock command with the appropriate data block size on the data line including 16 bit CRC. The data block shall indicate the mode UNLOCK, the length (PWDS_LEN) and the password (PWD) itself.

If the PWD content equals to the sent password then the card will be unlocked and the card-locked status bit will be cleared in the status register. If the password is not correct then the LOCK_UNLOCK_FAILED error bit will be set in the status register.

Note that the unlocking is done only for the current power session. As long as the PWD is not cleared the card will be locked automatically on the next power up. The only way to unlock the card is by clearing the password.

An attempt to unlock an unlocked card will fail and LOCK_UNLOCK_FAILED error bit will be set in the status register.

- **Forcing Erase:**

In case that the user forgot the password (the PWD content) it is possible to erase all the card data content along with the PWD content. This operation is called *Forced Erase*.

  - Select a card (CMD7), if not previously selected already.
  - Define the block length (CMD16) to 1 byte (8bit card lock/unlock command). Send the card lock/unlock command with the appropriate data block of one byte on the data line including 16 bit CRC. The data block shall indicate the mode ERASE (the ERASE bit shall be the only bit set).

If the ERASE bit is not the only bit in the data field then the LOCK_UNLOCK_FAILED error bit will be set in the status register and the erase request is rejected. If the command was accepted then ALL THE CARD CONTENT WILL BE ERASED including the PWD and PWD_LEN register content and the locked card will get unlocked. An attempt to force erase on an unlocked card will fail and LOCK_UNLOCK_FAILED error bit will be set in the status register.

4.3.7 Copyright Protection

Detailed description of the Copyrights Protection mechanism and the related security SD Memory Card commands can be found in “SD Memory Card Security Specification” document. All the SD Memory Card security related commands shall be operated in data transfer mode of operation.

As defined in the SDMI spec the data content that is saved in the card is saved already encrypted and it pass transparently to/from the card. NO operation is done on the data and there is no restriction to read the data at any time. Associated to every data packet (song, for example) that is saved in the un-protected memory there is a special data that shall be saved in a protected memory area. For any access (any Read or Write or Erase Command) from/to the data in the protected area, an authentication procedure shall be done between the card and the connected device, either the LCM (PC for example) or the PD (Portable Device - SD Player for example). After the authentication process was passed OK, the card is ready to accept or give data from/to the connected device. While the card is in the secured mode of operation (after the authentication succeeded) the argument and the associated data that is sent to the card or read from the card are encrypted. At the end of the

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

Read/Write/Erase operation the card gets out automatically of its secured mode.

4.3.8 Application specific commands:

The SD Memory Card is defined to be protocol forward compatible to the MultiMediaCard Standard.

The SD Memory Card system is designed to provide a standard interface for a variety applications types. In order to keep future compatibility to the MultiMediaCard standard together with new SD Memory Card specific commands the SD Memory Card use the Application Specific commands feature to implement its proprietary commands. Following is a description of the APP_CMD and GEN_CMD as were defined in the MultiMediaCard spec.

- **Application Specific Command – APP_CMD (CMD55)**

This command, when received by the card, will cause the card to interpret the following command as an application specific command, ACMD. The ACMD has the same structure as of regular MultiMediaCard standard commands and it may have the same CMD number. The card will recognize it as ACMD by the fact that it appears after APP_CMD.

The only effect of the APP_CMD is that if the command index of the, immediately, following command has an ACMD overloading it, the non standard version will be used. If, as an example, a card has a definition for ACMD13 but not for ACMD7 then, if received immediately after APP_CMD command, Command 13 will be interpreted as the non standard ACMD13 but, command 7 as the standard CMD7. In order to use one of the manufacturer specific ACMD's the host will:

- Send APP_CMD. The response will have the APP_CMD bit (new status bit) set signaling to the host that ACMD is now expected.
- Send the required ACMD. The response will have the APP_CMD bit set, indicating that the accepted command was interpreted as ACMD. If a non-ACMD is sent then it will be respected by the card as normal SD Memory Card command and the APP_CMD bit in the Card Status stays clear.

If a non valid command is sent (neither ACMD nor CMD) then it will be handled as a standard SD Memory Card illegal command error.

From the SD Memory Card protocol point of view the ACMD numbers will be defined by the manufacturers with some restrictions. The following ACMD numbers are reserved for the SD Memory Card proprietary applications and may not be used by any SD Memory Card manufacturer:

ACMD6, ACMD13, ACMD17-25, ACMD38-49, ACMD51.

- **General Command - GEN_CMD (CMD56)**

The bus transaction of the GEN_CMD is the same as the single block read or write commands (CMD24 or CMD17). The difference is that the argument denotes the direction of the data transfer (rather than the address) and the data block is not a memory payload data but has a vendor specific format and meaning. The card shall be selected ('*tran_state*') before sending CMD56. The data block size is the BLOCK_LEN that was defined with CMD16. The response to CMD56 will be R1.

Note that there are no reserved data pattern (of the associated data block) for SD Memory Card specific applications.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

4.4 Clock Control

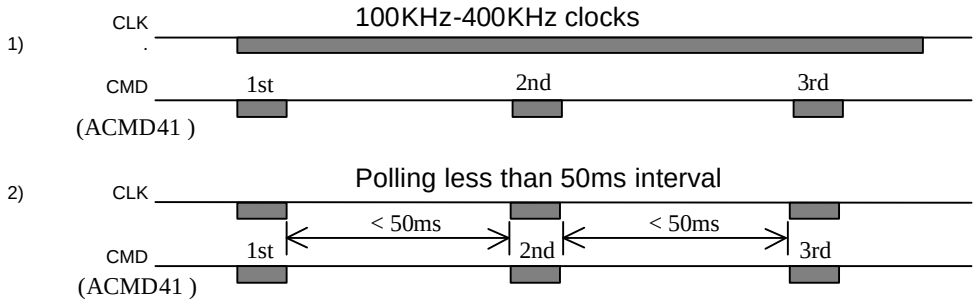
The SD Memory Card bus clock signal can be used by the host to turn the cards into energy saving mode or to control the data flow (to avoid under-run or over-run conditions) on the bus. The host is allowed to lower the clock frequency or shut it down. For example, in a case that a host with 512Bytes of data buffer would like to transfer data to a card with 1KByte write blocks. So, in order to preserve a continuous data transfer, from the cards point of view, the clock to the card shall be stopped after the first 512Bytes. Then the host will fill it internal buffer with another 512Bytes. After the second half of the write block is ready in the host, it will continue the data transfer to the card by re-starting the clock supply. In such a way the card does not recognize any interruptions in the data transfer.

There are a few restrictions the host must follow:

- The bus frequency can be changed at any time (under the restrictions of maximum data transfer frequency and the identification frequency defined by the specification document).
- En examption to the above is ACMD41(SD_APP_OP_COND). After issuing command ACMD41 the following 1) or 2) procedures shall be done by the host until the card becomes ready.

  1) Issue continues clock in frequency range of 100KHz-400KHz.

  2) If the host wants to stop the clock, poll busy bit by ACMD41 command at less than 50ms intervals.



- It is an obvious requirement that the clock must be running for the card to output data or response tokens. After the last SD Memory Card bus transaction, the host is required, to provide **8 (eight)** clock cycles for the card to complete the operation before shutting down the clock. Following is a list of the various bus transactions:
  - A command with no response. 8 clocks after the host command end bit.
  - A command with response. 8 clocks after the card response end bit.
  - A read data transaction. 8 clocks after the end bit of the last data block.
  - A write data transaction. 8 clocks after the CRC status token.
- The host is allowed to shut down the clock of a “busy” card. The card will complete the programming operation regardless of the host clock. However, the host must provide a clock edge for the card to turn off its busy signal. Without a clock edge the card (unless previously disconnected by a deselect command -CMD7) will force the DAT line down, forever.

4.5 Cyclic redundancy codes (CRC)

The CRC is intended for protecting SD Memory Card commands, responses and data transfer

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

against transmission errors on the SD Memory Card bus. One CRC is generated for every command and checked for every response on the CMD line. For data blocks one CRC per transferred block is generated. The CRC is generated and checked as described in the following.

- **CRC7**

The CRC7 check is used for all commands, for all responses except type R3, and for the CSD and CID registers. The CRC7 is a 7-bit value and is computed as follows:

generator polynomial: $G(x) = x^7 + x^3 + 1$.

$M(x) = (\text{first bit}) * x^n + (\text{second bit}) * x^{n-1} + ... + (\text{last bit}) * x^0$

$CRC[6...0] = \text{Remainder } [(M(x) * x^7) / G(x)]$

The first bit is the most left bit of the corresponding bitstring (of the command, response, CID or CSD). The degree *n* of the polynomial is the number of CRC protected bits decreased by one. The number of bits to be protected is 40 for commands and responses (n = 39), and 120 for the CSD and CID (n = 119).

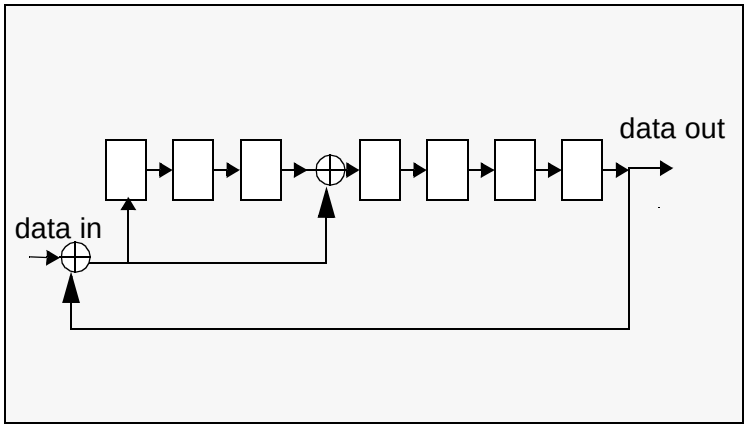


Figure 17: CRC7 generator/checker

- **CRC16**

In case of one DAT line usage (as in MultiMediaCard) than the CRC16 is used for payload protection in block transfer mode. The CRC check sum is a 16-bit value and is computed as follows:

generator polynomial $G(x) = x^{16} + x^{12} + x^5 + 1$

$M(x) = (\text{first bit}) * x^n + (\text{second bit}) * x^{n-1} + ... + (\text{last bit}) * x^0$

$CRC[15...0] = \text{Remainder } [(M(x) * x^{16}) / G(x)]$

The first bit is the first data bit of the corresponding block. The degree *n* of the polynomial denotes the number of bits of the data block decreased by one (e.g. n = 4095 for a block length of 512 bytes). The generator polynomial G(x) is a standard CCITT polynomial. The code has a minimal distance d=4 and is used for a payload length of up to 2048 Bytes (n <= 16383).

The same CRC16 method shall be used in single DAT line mode and in wide bus mode.

In wide bus mode, the CRC16 is done on each line separately.

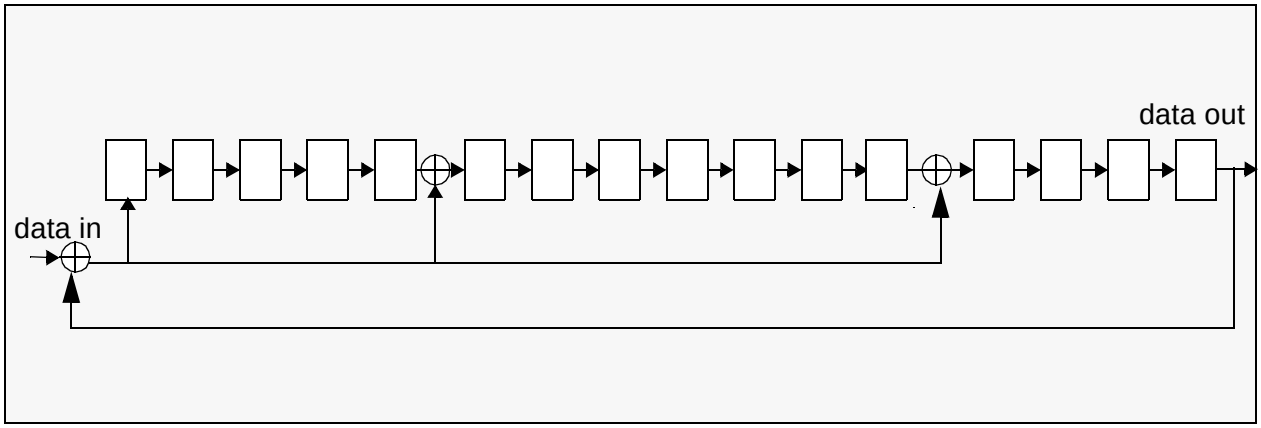


Figure 18: CRC16 generator/checker

4.6 Error Conditions

4.6.1 CRC and Illegal Command

All commands are protected by CRC (cyclic redundancy check) bits. If the addressed card's CRC check fails, the card does not respond and the command is not executed. The card does not change its state, and COM_CRC_ERROR bit is set in the status register.

Similarly, if an illegal command has been received, a card shall not change its state, shall not response and shall set the ILLEGAL_COMMAND error bit in the status register. Only the non-erroneous state branches are shown in the state diagrams (see Figure 15 to Figure). Table 17 contains a complete state transition description.

There are different kinds of illegal commands:

- Commands which belong to classes not supported by the card (e.g. write commands in read only cards).
- Commands not allowed in the current state (e.g. CMD2 in Transfer State).
- Commands which are not defined (e.g. CMD5).

4.6.2 Read, Write and Erase Time-out Conditions

The times after which a time-out condition for read operations occurs are (card independent) **either 100 times longer** than the typical access times for these operations given below **or 100ms (the lower of them)**. The times after which a time-out condition for Write/Erase operations occurs are (card independent) **either 100 times longer** than the typical program times for these operations given below **or 250ms (the lower of them).** A card shall complete the command within this time

period, or give up and return an error message. If the host does not get any response with the given time out it should assume the card is not going to respond anymore and try to recover (e.g. reset the card, power cycle, reject, etc.). The typical access and program times are defined as follows:

- **Read**

The read access time is defined as the sum of the two times given by the CSD parameters TAAC and NSAC (see Chapter 4.12). These card parameters define the typical delay between the end bit of the read command and the start bit of the data block. This number is card dependent and should be used by the host to calculate throughput and the maximal frequency for stream read.

- **Write**

The R2W_FACTOR field in the CSD is used to calculate the typical block program time obtained by multiplying the read access time by this factor. It applies to all write/erase commands (e.g. SET(CLEAR)_WRITE_PROTECT, PROGRAM_CSD(CID) and the block write commands).

- **Erase**

The duration of an erase command will be (order of magnitude) the number of write blocks (WRITE_BL) to be erased multiplied by the block write delay.

4.7 Commands

4.7.1 Command Types

There are four kinds of commands defined to control the SD Memory Card:

- broadcast commands (bc), no response - The broadcast feature is only if all the CMD lines are connected together in the host. If they are separated then each card will accept it separately on his turn.
- broadcast commands with response (bcr)
  response from all cards simultaneously - Since there is no Open Drain mode in SD Memory Card this type of commands shall be used only if all the CMD lines are separated - the command will be accepted and responded by every card separately.
- addressed (point-to-point) commands (ac)
  no data transfer on DAT
- addressed (point-to-point) data transfer commands (adtc)
  data transfer on DAT

All commands and responses are sent over the CMD line of the SD Memory Card. The command transmission always starts with the left bit of the bitstring corresponding to the command codeword.

4.7.2 Command Format

All commands have a fixed code length of 48 bits, needing a transmission time of 2.4 ?s @ 20 MHz

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'1'	x	x	x	'1'
Description	start bit	transmission bit	command index	argument	CRC7	end bit

Table 7: Command Format

A command always starts with a start bit (always '0'), followed by the bit indicating the direction of transmission (host = '1'). The next 6 bits indicate the index of the command, this value being interpreted as a binary coded number (between 0 and 63). Some commands need an argument (e.g. an address), which is coded by 32 bits. A value denoted by 'x' in the table above indicates this variable is dependent on the command. All commands are protected by a CRC (see Chapter 7.2 for the definition of CRC7). Every command codeword is terminated by the end bit (always '1'). All commands and their arguments are listed in Table 9-Table 16.

4.7.3 Command Classes (Redefined for SD Memory Card)

The command set of the SD Memory Card system is divided into several classes (See Table 8). Each class supports a set of card functionalities.

Class 0, 2, 4, 5 and 8 are mandatory and shall be supported by all SD Memory Cards. The other classes are optional. The supported Card Command Classes (CCC) are coded as a parameter in the card specific data (CSD) register of each card, providing the host with information on how to access the card.

	Card Command Class (CCC)	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9-11**
Supported commands	**class description**	basic	reserved	block read	reserved	block write	erase	write protection	lock card	application specific	reserved
CMD0	Mandatory	+									
CMD2	Mandatory	+									
CMD3	Mandatory	+									
CMD4	Mandatory	+									
CMD7	Mandatory	+									
CMD9	Mandatory	+									
CMD10	Mandatory	+									
CMD12	Mandatory	+									

Table 8: Card Command Classes (CCCs)

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

	Card Command Class (CCC)	0	1	2	3	4	5	6	7	8	9-11
Supported commands	**class description**	basic	reserved	block read	reserved	block write	erase	write protection	lock card	application specific	reserved
CMD13	Mandatory	+									
CMD15	Mandatory	+									
CMD16	Mandatory			+		+					
CMD17	Mandatory			+							
CMD18	Mandatory			+							
CMD24	Mandatory(1)					+					
CMD25	Mandatory(1)					+					
CMD27	Mandatory(1)					+					
CMD28	Optional							+			
CMD29	Optional							+			
CMD30	Optional							+			
CMD32	Mandatory(1)						+				
CMD33	Mandatory(1)						+				
CMD38	Mandatory(1)						+				
CMD42	Optional								+		
CMD55	Mandatory									+	
CMD56	Mandatory									+	
ACMD6	Mandatory									+	
ACMD13	Mandatory									+	
ACMD22	Mandatory(1)									+	
ACMD23	Mandatory(1)									+	
ACMD41	Mandatory									+	
ACMD42	Mandatory									+	
ACMD51	Mandatory									+	

Table 8: Card Command Classes (CCCs)

Note (1): The write related commands are mandatory only for the Writable type of Cards (OTP and R/W).

4.7.4 Detailed Command Description

The following tables define in detail all SD Memory Card bus commands. The responses R1-R3, R6

are defined in Chapter 4.9. The registers CID, CSD and DSR are described in Chapter 5.

CMD INDEX	type	argument	resp	abbreviation	command description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	resets all cards to idle state
CMD1	reserved				
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	asks any card to send the CID numbers on the CMD line (any card that is connected to the host will respond)
CMD3	bcr	[31:0] stuff bits	R6	SEND_RELATIVE_ ADDR	ask the card to publish a new relative address (RCA)
CMD4	bc	[31:16] DSR [15:0] stuff bits	-	SET_DSR	programs the DSR of all cards
CMD5	reserved				
CMD6	reserved				
CMD7	ac	[31:16] RCA [15:0] stuff bits	R1b (only from the selected card)	SELECT/ DESELECT_ CARD	command toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases the card is selected by its own relative address and gets deselected by any other address; address 0 deselects all. In case that the RCA equal 0 then the host may do one of the following: - Use other RCA number to perform card deselection. - Re-send CMD3 to change its RCA number to other than 0 and then use CMD7with RCA=0 for card deselection.
CMD8	reserved				
CMD9	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CSD	addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CID	addressed card sends its card identification (CID) on CMD the line.
CMD11	reserved				
CMD12	ac	[31:0] stuff bits	R1b	STOP_ TRANSMISSION	forces the card to stop transmission
CMD13	ac	[31:16] RCA [15:0] stuff bits	R1	SEND_STATUS	addressed card sends its status register.
CMD14	reserved				
CMD15	ac	[31:16] RCA [15:0] stuff bits	-	GO_INACTIVE_ STATE	sets the card to inactive state in order to protect the card stack against communication breakdowns.

Table 9: Basic commands (class 0)

CMD INDEX	type	argument	resp	abbreviation	command description
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD. Supported only if Partial block RD/WR operation are allowed in CSD.
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_ BLOCK	reads a block of the size selected by the SET_BLOCKLEN command.[1]
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_ BLOCK	continuously transfers data blocks from card to host until interrupted by a STOP_TRANSMISSION command.
CMD19 ... CMD23	reserved				

Table 10: Block oriented read commands (class 2)

1)The data transferred must not cross a physical block boundary unless READ_BLK_MISALIGN is set in the CSD.

CMD INDEX	type	argument	resp	abbreviation	command description
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	writes a block of the size selected by the SET_BLOCKLEN command.[1]
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPL E_BLOCK	continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	Reserved For Manufacturer				
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	programming of the programmable bits of the CSD.

Table 11: Block oriented write commands (class 4)

1)The data transferred must not cross a physical block boundary unless WRITE_BLK_MISALIGN is set in the CSD. In case that write partial blocks is not supported then the block length=default block length (given in CSD).

CMD INDEX	type	argument	resp	abbreviation	command description
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	if the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	if the card provides write protection features, this command clears the write protection bit of the addressed group.

Table 12: Block oriented write protection commands (class 6)

CMD INDEX	type	argument	resp	abbreviation	command description
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_ PROT	if the card provides write protection features, this command asks the card to send the status of the write protection bits. [1]
CMD31	reserved				

Table 12: Block oriented write protection commands (class 6)

1)32 write protection bits (representing 32 write protect groups starting at the specified address) followed by 16 CRC bits are transferred in a payload format via the data line. The last (least significant) bit of the protection bits corresponds to the first addressed group. If the addresses of the last groups are outside the valid range, then the corresponding write protection bits shall be set to zero.

CMD INDEX	type	argument	resp	abbreviation	command description
CMD32	ac	[31:0] data address	R1	ERASE_WR_BLK_ START	sets the address of the first write-block to be erased.
CMD33	ac	[31:0] data address	R1	ERASE_WR_BLK_END	sets the address of the last write block of the continuous range to be erased.
CMD34 ... CMD37	reserved				
CMD38	ac	[31:0] stuff bits	R1b	ERASE	erases all previously selected write blocks.
CMD39	reserved				
CMD40					Non Valid in SD Memory Card - Reserved for MultiMediaCard I/O mode
CMD41	reserved				

Table 13: Erase commands (class 5)

CMD INDEX	type	argument	resp	abbreviation	command description
CMD42	adtc	[31:0] stuff bits.	R1	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43 ... CMD54	reserved				

Table 14: Lock card (class 7)

CMD INDEX	type	argument	resp	abbreviation	command description
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather than a standard command
CMD56	adtc	[31:1] stuff bits. [0]: RD/ WR[1]	R1	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application specific commands. The size of the data block shall be set by the SET_BLOCK_LEN command.
CMD57 ... CMD59	reserved				
CMD60 -63	reserved for manufacturer				

Table 15: Application specific commands (class 8)

1) RD/WR: "1" the host gets a block of data from the card.
"0" the host sends block of data to the card.

All the application specific commands (given in Table 15) are supported if Class 8 is allowed (mandatory in SD Memory Card).

All future reserved commands shall have a codeword length of 48 bits, as well as their responses (if there are any).

The following table describes all the application specific commands supported/reserved by the SD Memory Card. All the following ACMDs shall be preceded with APP_CMD command (CMD55).

ACMD INDEX	type	argument	resp	abbreviation	command description
ACMD6	ac	[31:2] stuff bits [1:0]bus width	R1	SET_BUS_WIDTH	Defines the data bus width ('00'=1bit or '10'=4 bits bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status. The status fields are given in Table 24.
ACMD17	reserved				
ACMD18	--	--	--	--	Reserved for SD security applications[1]
ACMD19 to ACMD21	reserved				
ACMD22	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_ BLOCKS	Send the number of the written (without errors) write blocks. Responds with 32bit+CRC data block.

ACMD INDEX	type	argument	resp	abbreviation	command description
ACMD23	ac	[31:23] stuff bits [22:0]Number of blocks	R1	SET_WR_BLK_ ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for faster Multiple Block WR command). "1"=default (one wr block)(2).
ACMD24	reserved				
ACMD25	--	--	--	--	Reserved for SD security applications[1]
ACMD26	--	--	--	--	Reserved for SD security applications[1]
ACMD38	--	--	--	--	Reserved for SD security applications[1]
ACMD39 to ACMD40	reserved				
ACMD41	bcr	[31:0]OCR without busy	R3	SD_SEND_OP_COND	Asks the accessed card to send its operating condition register (OCR) content in the response on the CMD line.
ACMD42	ac	[31:1] stuff bits [0]set_cd	R1	SET_CLR_CARD_ DETECT	Connect[1]/Disconnect[0] the 50KOhm pull-up resistor on CD/DAT3 (pin 1) of the card. The pull-up may be used for card detection.
ACMD43 ACMD49	--	--	--	--	Reserved for SD security applications[1]
ACMD51	adtc	[31:0] staff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

(1) Refer to "SD Memory Card Security Specification" for detailed explanation about the SD Security Features
(2) Command STOP_TRAN (CMD12) shall be used to stop the transmission in Write Multiple Block whether the pre-erase (ACMD23) feature is used or not.

Table 16: Application Specific Commands used/reserved by SD Memory Card

4.8 Card State Transition Table

Table 17 defines the card state transitions in dependency of the received command.

	current state									
	idle	ready	ident	stby	tran	data	rcv	prg	dis	ina
command	changes to									
class independent										
CRC error	-	-	-	-	-	-	-	-	-	-
command not supported	-	-	-	-	-	-	-	-	-	-
class 0										

Table 17: Card state transition table

	current state									
	idle	**ready**	**ident**	**stby**	**tran**	**data**	**rcv**	**prg**	**dis**	**ina**
CMD0	idle	idle	idle	idle	idle	idle	idle	idle	idle	-
CMD2	-	ident	-	-	-	-	-	-	-	-
CMD3	-	-	stby	stby	-	-	-	-	-	-
CMD4	-	-	-	stby	-	-	-	-	-	-
CMD7, card is addressed	-	-	-	tran	-	-	-	-	prg	-
CMD7, card is not addressed	-	-	-	stby	stby	stby	-	dis	-	-
CMD9	-	-	-	stby	-	-	-	-	-	-
CMD10	-	-	-	stby	-	-	-	-	-	-
CMD12	-	-	-	-	-	tran	prg	-	-	-
CMD13	-	-	-	stby	tran	data	rcv	prg	dis	-
CMD15	-	-	-	ina	ina	ina	ina	ina	ina	-
class 2										
CMD16	-	-	-	-	tran	-	-	-	-	-
CMD17	-	-	-	-	data	-	-	-	-	-
CMD18	-	-	-	-	data	-	-	-	-	-
class 4										
CMD16	see class 2									
CMD24	-	-	-	-	rcv	-	-	-	-	-
CMD25	-	-	-	-	rcv	-	-	-	-	-
CMD27	-	-	-	-	rcv	-	-	-	-	-
class 6										
CMD28	-	-	-	-	prg	-	-	-	-	-
CMD29	-	-	-	-	prg	-	-	-	-	-
CMD30	-	-	-	-	data	-	-	-	-	-
class 5										
CMD32	-	-	-	-	tran	-	-	-	-	-
CMD33	-	-	-	-	tran	-	-	-	-	-
CMD38	-	-	-	-	prg	-	-	-	-	-
class 7										
CMD42	-	-	-	-	rcv	-	-	-	-	-
class 8										
CMD55	idle	-	-	stby	tran	data	rcv	prg	dis	-
CMD56; RD/WR = 0	-	-	-	-	rcv	-	-	-	-	-
CMD56; RD/WR = 1	-	-	-	-	data	-	-	-	-	-
ACMD6	-	-	-	-	tran	-	-	-	-	-

Table 17: Card state transition table

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

	current state									
	idle	**ready**	**ident**	**stby**	**tran**	**data**	**rcv**	**prg**	**dis**	**ina**
ACMD13	-	-	-	-	tran	-	-	-	-	-
ACMD22	-	-	-	-	tran	-	-	-	-	-
ACMD23	-	-	-	-	tran	-	-	-	-	-
ACMD18,25,26,38, 43,44,45,46,47,48,49	Refer to “SD Memory Card Security Specification” for explanation about the SD Security Features									
ACMD41, card V_{DD} range compatible	ready	-	-	-	-	-	-	-	-	-
ACMD41, card is busy	idle	-	-	-	-	-	-	-	-	-
ACMD41, card V_{DD} range not compatible	ina	-	-	-	-	-	-	-	-	-
ACMD42	-	-	-	-	tran	-	-	-	-	-
ACMD51	-	-	-	-	data	-	-	-	-	-
class 9- 11										
CMD41; CMD43...CMD54, CMD57-CMD59	reserved									
CMD60...CMD63	reserved for manufacturer									

Table 17: Card state transition table

The state transitions of the SD Memory Card application specific commands are given under Class 8, above.

4.9 Responses

All responses are sent via the command line CMD. The response transmission always starts with the left bit of the bitstring corresponding to the response codeword. The code length depends on the response type.

A response always starts with a start bit (always ‘0’), followed by the bit indicating the direction of transmission (card = ‘0’). A value denoted by ‘x’ in the tables below indicates a variable entry. All responses except for the type R3 (see below) are protected by a CRC (see Chapter 7.2 for the definition of CRC7). Every command codeword is terminated by the end bit (always ‘1’).

There are four types of responses. Their formats are defined as follows:

- **R1** (normal response command): code length 48 bit. The bits 45:40 indicate the index of the command to be responded to, this value being interpreted as a binary coded number (between 0 and 63). The status of the card is coded in 32 bits. Note that in case that data transfer to the card is involved then a busy signal may appear on the data line after the transmission of each block of data. The host shell check for busy after data block transmission.

The card status is described in Chapter 4.10.

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'0'	x	x	x	'1'
Description	start bit	transmission bit	command index	card status	CRC7	end bit

Table 18: Response R1

- **R1b** is identical to R1 with an optional busy signal transmitted on the data line. The card may become busy after receiving these commands based on its state prior to the command reception. The Host shell check for busy at the response. Refer to Chapter 4.12.3 for detailed description and timing diagrams.
- **R2** (CID, CSD register): code length 136 bits. The contents of the CID register are sent as a response to the commands CMD2 and CMD10. The contents of the CSD register are sent as a response to CMD9. Only the bits [127...1] of the CID and CSD are transferred, the reserved bit [0] of these registers is replaced by the end bit of the response.

Bit position	135	134	[133:128]	[127:1]	0
Width (bits)	1	1	6	127	1
Value	'0'	'0'	'111111'	x	'1'
Description	start bit	transmission bit	reserved	CID or CSD register incl. internal CRC7	end bit

Table 19: Response R2

- **R3** (OCR register): code length 48 bits. The contents of the OCR register is sent as a response to ACMD41.

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'0'	'111111'	x	'1111111'	'1'
Description	start bit	transmission bit	reserved	OCR register	reserved	end bit

Table 20: Response R3

- **R6** (Published RCA response): code length 48 bit. The bits 45:40 indicate the index of the

Bit position	47	46	[45:40]	[39:8] Argument field		[7:1]	0
Width (bits)	1	1	6	16	16	7	1
Value	'0'	'0'	x	x	x	x	'1'
Description	start bit	transmission bit	command index ('000011')	New published RCA [31:16] of the card	[15:0] *card status* bits: 23,22,19,12:0 (see Table 22)	CRC7	end bit

Table 21: Response R6

command to be responded to - in that case it will be '000011' (together with bit 5 in the status bits it means = CMD3). The 16 MSB bits of the argument field are used for the Published RCA number.

4.10 SD Memory Card Status

SD Memory Card supports two card status field as follows:

- '*Card Status*': compatible to the MultiMediaCard protocol.

- '*SD_Status*': Extended status field of 512bits that supports special features of the SD Memory Card and future Application Specific features.

4.10.1 Card Status

The response format R1 contains a 32-bit field named *card status*. This field is intended to transmit the card's status information (which may be stored in a local status register) to the host. If not specified otherwise, the status entries are always related to the previous issued command. The semantics of this register is according to the CSD entry SPEC_VERS (see Chapter 5.3), indicating the version of the response formats (possibly used for later extensions).

Table 22 defines the different entries of the status. The type and clear condition fields in the table are abbreviated as follows:

- **Type:**
  - E: Error bit.
  - S: Status bit.
  - R: Detected and set for the actual command response.
  - X: Detected and set during command execution. The host must poll the card by issuing the status command in order to read these bits.
- **Clear Condition:**
  - A: According to the card current state.
  - B: Always related to the previous command. Reception of a valid command will clear it (with a delay of one command).
  - C: Clear by read

.

Bits	Identifier	Type	Value	Description	Clear Condition
31	OUT_OF_RANGE	E R X	'0'= no error '1'= error	The command's argument was out of the allowed range for this card.	C
30	ADDRESS_ERROR	E R	'0'= no error '1'= error	A misaligned address which did not match the block length was used in the command.	C
29	BLOCK_LEN_ERROR	E R	'0'= no error '1'= error	The transferred block length is not allowed for this card, or the number of transferred bytes does not match the block length.	C
28	ERASE_SEQ_ERROR	E R	'0'= no error '1'= error	An error in the sequence of erase commands occurred.	C
27	ERASE_PARAM	E R X	'0'= no error '1'= error	An invalid selection of write-blocks for erase occurred.	C
26	WP_VIOLATION	E R X	'0'= not protected '1'= protected	Attempt to program a write protected block.	C
25	CARD_IS_LOCKED	S X	'0' = card unlocked '1' = card locked	When set, signals that the card is locked by the host	A
24	LOCK_UNLOCK_FAILED	E R X	'0' = no error '1' = error	Set when a sequence or password error has been detected in lock/ unlock card command or if there was an attempt to access a locked card	C
23	COM_CRC_ERROR	E R	'0'= no error '1'= error	The CRC check of the previous command failed.	B
22	ILLEGAL_COMMAND	E R	'0'= no error '1'= error	Command not legal for the card state	B
21	CARD_ECC_FAILED	E R X	'0'= success '1'= failure	Card internal ECC was applied but failed to correct the data.	C
20	CC_ERROR	E R X	'0'= no error '1'= error	Internal card controller error	C
19	ERROR	E R X	'0'= no error '1'= error	A general or an unknown error occurred during the operation.	C
18	reserved				
17	reserved				

Table 22: Card status

Bits	Identifier	Type	Value	Description	Clear Condition
16	CID/ CSD_OVERWRITE	E R X	'0'= no error '1'= error	can be either one of the following errors: - The CID register has been already written and can not be overwritten - The read only section of the CSD does not match the card content. - An attempt to reverse the copy (set as original) or permanent WP (unprotected) bits was made.	C
15	WP_ERASE_SKIP	S X	'0'= not protected '1'= protected	Only partial address space was erased due to existing write protected blocks.	C
14	CARD_ECC_DISABLED	S X	'0'= enabled '1'= disabled	The command has been executed without using the internal ECC.	A
13	ERASE_RESET	S R	'0'= cleared '1'= set	An erase sequence was cleared before executing because an out of erase sequence command was received	C
12:9	CURRENT_STATE	S X	0 = idle 1 = ready 2 = ident 3 = stby 4 = tran 5 = data 6 = rcv 7 = prg 8 = dis 9-15 = reserved	The state of the card when receiving the command. If the command execution causes a state change, it will be visible to the host in the response to the next command. The four bits are interpreted as a binary coded number between 0 and 15.	B
8	READY_FOR_DATA	S X	'0'= not ready '1'= ready	corresponds to buffer empty signalling on the bus	A
7:6					
5	APP_CMD	S R	'0' = Disabled '1' = Enabled	The card will expect ACMD, or indication that the command has been interpreted as ACMD	C
4	reserved				
3	AKE_SEQ_ERROR (SD Memory Card app. spec.)	E R	'0' = no error '1' = error	Error in the sequence of authentication process	C
2	reserved for application specific commands				
1, 0	reserved for manufacturer test mode				

Table 22: Card status

The following table defines for each command responded by a R1 response the affected bits in the

status field. An 'x' means the error/status bit may be set in the response to the respective command.

CMD#	Response Format 1 Status bit #																					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12:9	8	5
3(1)									x	x			x							x		
7					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
12	x	x				x	x		x	x	x	x	x	x	x			x		x		
13	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	
16			x		x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
17	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
18	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
24	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
25	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
26					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
27					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
28	x				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
29	x				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
30	x				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
32	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
33	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
38				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
42					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
55					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
56					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
ACMD6	x				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD13					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD22					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD23					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD42					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD51					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x

Table 23: Card status field / command - cross reference

(1) The response to CMD3 is R6 that includes only bits 23, 22, 19 and 12:9 out of the Card Status

4.10.2 *SD Status*

The SD Status contains status bits that are related to the SD Memory Card proprietary features and may be used for future application specific usage. The size of the SD Status is one data block of 512bit. The content of this register is transmitted to the Host over the DAT bus along with 16 bit CRC. The SD Status is sent to the host over the DAT bus if ACMD13 is sent (CMD55 followed with CMD13). ACMD13 can be sent to a card only in '*tran_state*' (card is selected). SD Status structure

is described in bellow.

The same abbreviation for ‘type’ and ‘clear condition’ were used as for the Card Status above.

Bits	Identifier	Type	Value	Description	Clear Condition
511:510	DAT_BUS_WIDTH	S R	’00’= 1 (default) ‘01’= reserved ‘10’= 4 bit width ‘11’= reserved	Shows the currently defined data bus width that was defined by SET_BUS_WIDTH command	A
509	SECURED_MODE	S R	’0’= Not in the mode ’1’= In Secured Mode	Card is in Secured Mode of operation (refer to “SD Security Specification”).	A
508:496	reserved				
495:480	SD_CARD_TYPE	SR	’00xxh’= SD Memory Cards as defined in Physical Spec Ver.1 (’x’=don’t care). The following cards are currently defined: ’0000’= Regular SD RD/WR Card. ’0001’= SD ROM Card	In the future the 8LSBs will be used to define different variations of a SD Memory Card (Each bit will define different SD Type). The 8MSBs will be used to define SD Cards that do not comply with SD Memory Card as was defined in Spec Ver.1	A
479:448	SIZE_OF_PROTECTED_AREA	SR	Size of protected area (in units of MULT*BLOCK_LEN refer to CSD register Table 5.3)	Shows the size of protected area. The actual area = (SIZE_OF_PROTECTED_AREA) * MULT * BLOCK_LEN.	A
447:312	reserved				
311:0	reserved for manufacturer				

Table 24: SD Card Status

4.11 Memory Array Partitioning

The basic unit of data transfer to/from the SD Memory Card is one byte. All data transfer operations which require a block size always define block lengths as integer multiples of bytes. Some special functions need other partition granularity.

For block oriented commands, the following definition is used:

- Block: is the unit which is related to the block oriented read and write commands. Its size is the number of bytes which will be transferred when one block command is sent by the host. The size of a block is either programmable or fixed. The information about allowed block sizes and the programmability is stored in the CSD.

For devices which have erasable memory cells, special erase commands are defined. The granularity of the erasable units is in general not the same as for the block oriented commands:

- Sector: is the unit which is related to the erase commands. Its size is the number of blocks which will be erased in one portion. The size of a sector is fixed for each device. The information about the sector size (in blocks) is stored in the CSD.

For devices which include a write protection:

- WP-Group: is the minimal unit which may have individual write protection. Its size is the number of groups which will be write protected by one bit. The size of a WP-group is fixed for each device. The information about the size is stored in the CSD.

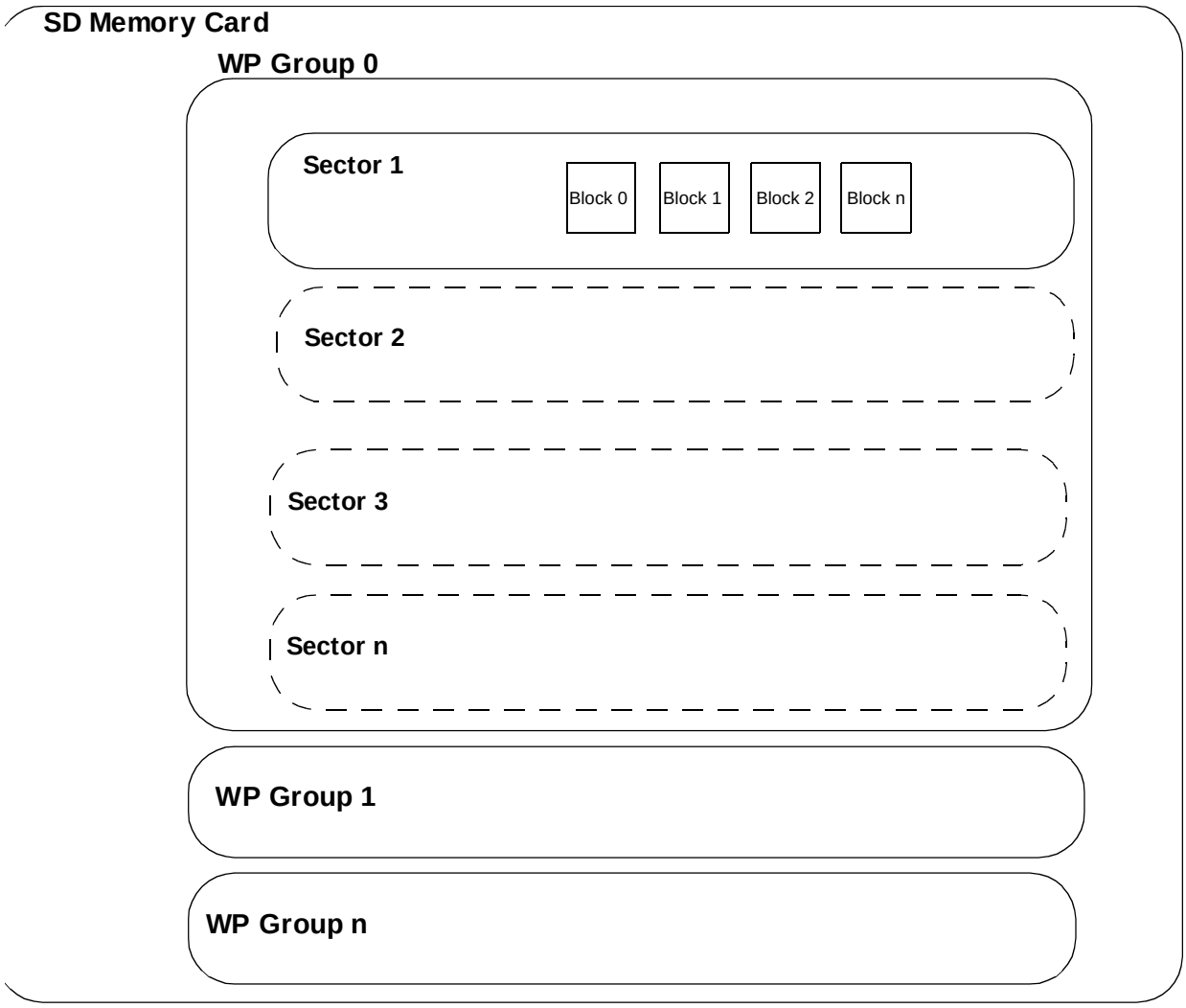


Figure 19: Write Protection hierarchy

Each WP-group may have an additional write protection bit. The write protection bits are programmable via special commands (see Chapter 4.7.4).

Both functions are optional and only useful for writable/erasable devices. The write protection may also be useful for multi type cards (e.g. a ROM - Flash combination). The information about the availability is stored in the CSD.

4.12 Timings

All timing diagrams use the following schematics and abbreviations:

S	Start bit (= '0')
T	Transmitter bit (Host = '1', Card = '0')
P	One-cycle pull-up (= '1')
E	End bit (=1)
Z	High impedance state (-> = '1')
D	Data bits
X	Don't Care data bits (from card)
*	Repetition
CRC	Cyclic redundancy check bits (7 bits)
	Card active
	Host active

Table 25: Timing diagram symbols

The difference between the P-bit and Z-bit is that a P-bit is actively driven to HIGH by the card respectively host output driver, while Z-bit is driven to (respectively kept) HIGH by the pull-up resistors R_{CMD} respectively R_{DAT}. Actively-driven P-bits are less sensitive to noise.

All timing values are defined in Table 26.

4.12.1 Command and Response

Both host command and card response are clocked out with the rising edge of the host clock.

- **Card identification and card operation conditions timing**

The timing for CMD2 and ACMD41 is given bellow. The command is followed by a period of two Z bits (allowing time for direction switching on the bus) and then by P bits pushed up by the responding card. The card response to the host command starts after N_{ID} clock cycles.

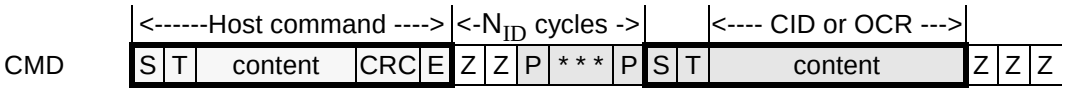


Figure 20: Identification timing (card identification mode)

- **Assign a card relative address**

The SEND_RELATIVE_ADDR (CMD 3) for SD Memory Card timing is given bellow. Note that CMD3 command's content, functionality and timing are different for MultiMediaCard. The minimum delay between the host command and card response is N_{CR} clock cycles.

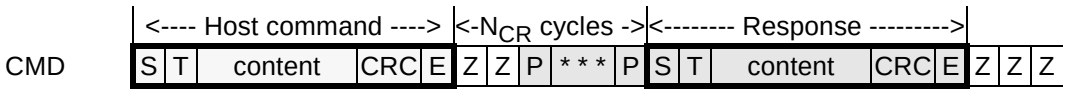


Figure 21: SEND_RELATIVE_ADDR timing

- **Data transfer mode.**

After the card published it own RCA it will switch to data transfer mode. The command is followed by a period of two Z bits (allowing time for direction switching on the bus) and then by P bits pushed up by the responding card. This timing diagram is relevant for all responded host commands except and ACMD41 and CMD2:

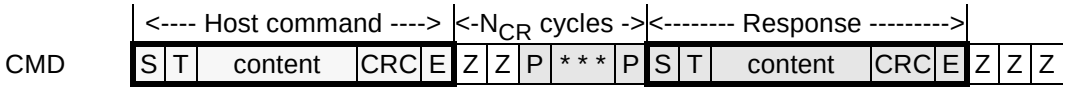


Figure 22: Command response timing (data transfer mode)

- **Last Card Response - Next Host Command Timing**

After receiving the last card response, the host can start the next command transmission after at least N_{RC} clock cycles. This timing is relevant for any host command.

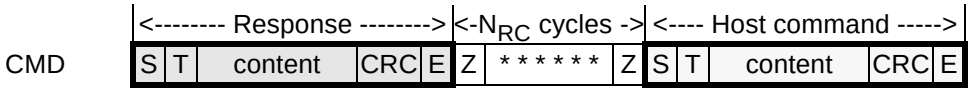


Figure 23: Timing response end to next CMD start (data transfer mode)

- **Last Host Command - Next Host Command Timing**

After the last command has been sent, the host can continue sending the next command after at least N_{CC} clock periods.

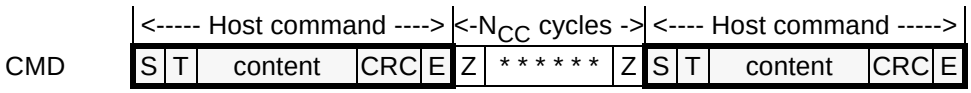


Figure 24: Timing of command sequences (all modes)

4.12.2 Data Read

*** Note: DAT line represents data bus (either 1 or 4 bits).**

- **Single Block Read**

The host selects one card for data read operation by CMD7, and sets the valid block length for block oriented data transfer by CMD16. The basic bus timing for a read operation is given in Figure 25. The sequence starts with a single block read command (CMD17) which specifies the start address

in the argument field. The response is sent on the CMD line as usual.

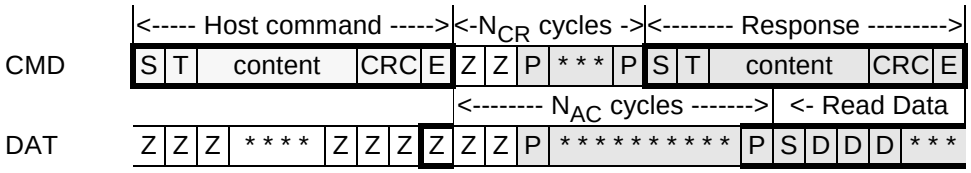


Figure 25: Timing of single block read

Data transmission from the card starts after the access time delay N_{AC} beginning from the end bit of the read command. After the last data bit, the CRC check bits are suffixed to allow the host to check for transmission errors.

- **Multiple Block Read**

In multiple block read mode, the card sends a continuous flow of data blocks following the initial host read command. The data flow is terminated by a stop transmission command (CMD12). Figure 26 describes the timing of the data blocks and Figure 27 the response to a stop command. The data transmission stops two clock cycles after the end bit of the stop command.

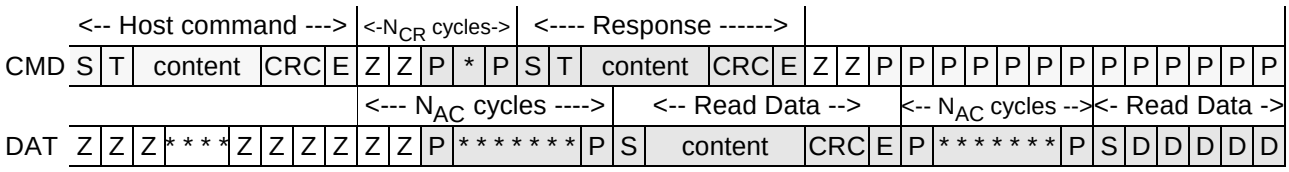


Figure 26: Timing of multiple block read command

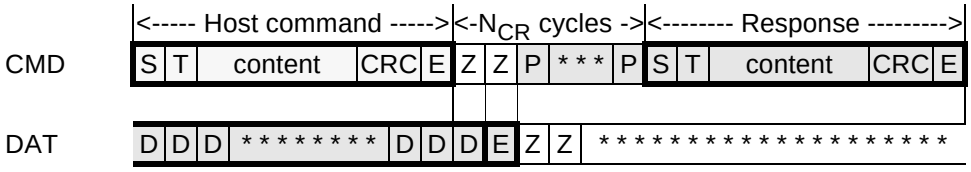


Figure 27: Timing of stop command (CMD12, data transfer mode)

4.12.3 Data Write

- **Single Block Write**

The host selects one card for data write operation by CMD7.

The host sets the valid block length for block oriented data transfer by CMD16.

The basic bus timing for a write operation is given in Figure 28. The sequence starts with a single block write command (CMD24) which determines (in the argument field) the start address. It is responded by the card on the CMD line as usual. The data transfer from the host starts N_{WR} clock cycles after the card response was received.

The data is suffixed with CRC check bits to allow the card to check it for transmission errors. The card sends back the CRC check result as a CRC status token on the DAT0 line. In the case of transmission error the card sends a negative CRC status ('101'). In the case of non erroneous transmission the card sends a positive CRC status ('010') and starts the data programming procedure.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

When a flash programming error occurs the card will ignore all further data blocks. In this case no CRC response will be sent to the host and, therefore, there will not be CRC start bit on the bus and the three CRC status bits will read (‘111‘).

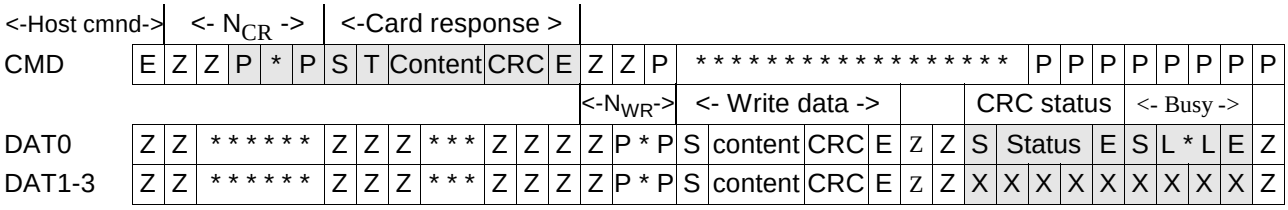


Figure 28: Timing of the block write command

Note that the CRC response output is always two clocks after the end of data.

If the card does not have a free data receive buffer, the card indicates this condition by pulling down the data line to LOW. The card stops pulling down the DAT0 line as soon as at least one receive buffer for the defined data transfer block length becomes free. This signalling does not give any information about the data write status which must be polled by the host.

- **Multiple Block Write**

In multiple block write mode, the card expects continuous flow of data blocks following the initial host write command.

As in the case of single block write, the data is suffixed with CRC check bits to allow the card to check it for transmission errors. The card sends back the CRC check result as a CRC status token on the DAT0 line. In the case of transmission error the card sends a negative CRC status (‘101’). In the case of non erroneous transmission the card sends a positive CRC status (‘010’) and starts the data programming procedure. When a flash programming error occurs the card will ignore all further data blocks. In this case no CRC response will be sent to the host and, therefore, there will not be CRC start bit on the bus and the three CRC status bits will read (‘111‘);

The data flow is terminated by a stop transmission command (CMD12). Figure 29 describes the timing of the data blocks with and without card busy signal.

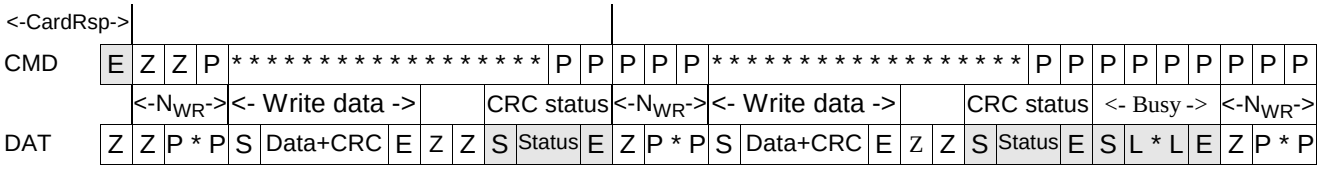


Figure 29: Timing of the multiple block write command

The stop transmission command works similar as in the read mode. Figure 30 to Figure 33 describe

the timing of the stop command in different card states.

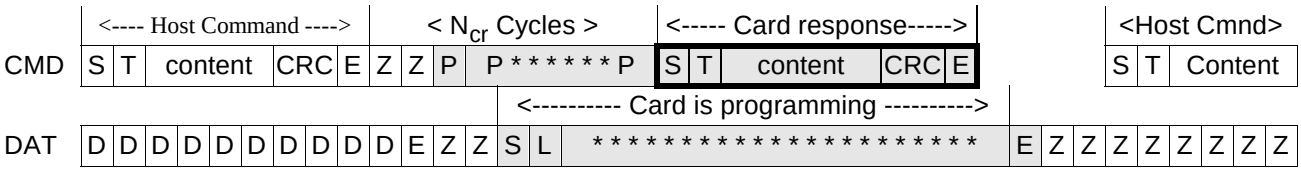
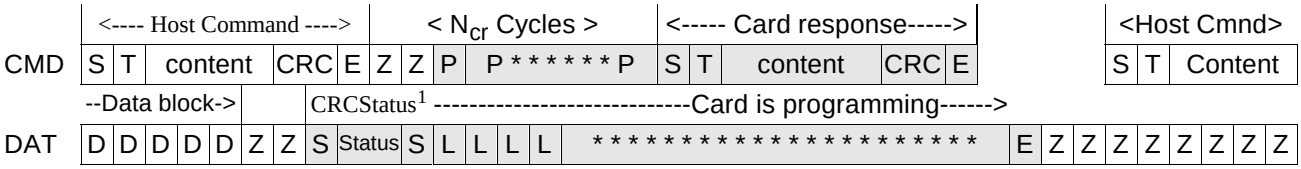


Figure 30: Stop transmission during data transfer from the host

The card will treat a data block as successfully received and ready for programming only if the CRC data of the block was validated and the CRC status token sent back to the host. Figure 31 is an example of an interrupted (by a host stop command) attempt to transmit the CRC status block. The sequence is identical to all other stop transmission examples. The end bit of the host command is followed, on the data line, with one more data bit and start of busy signalling. In that case there are no Z clocks, for switching the bus direction, because the bus direction is already towards the host. The received data block, in this case is considered incomplete and will not be programmed.



1) The card CRC status response was interrupted by the host.

Figure 31: Stop transmission during CRC status transfer from the card

All previous examples dealt with the scenario of the host stopping the data transmission during an active data transfer. The following two diagrams describe a scenario of receiving the stop transmission between data blocks. In the first example the card is busy programming the last block while in the second the card is idle. However, there are still unprogrammed data blocks in the input buffers. These blocks are being programmed as soon as the stop transmission command is received and the card activates the busy signal.

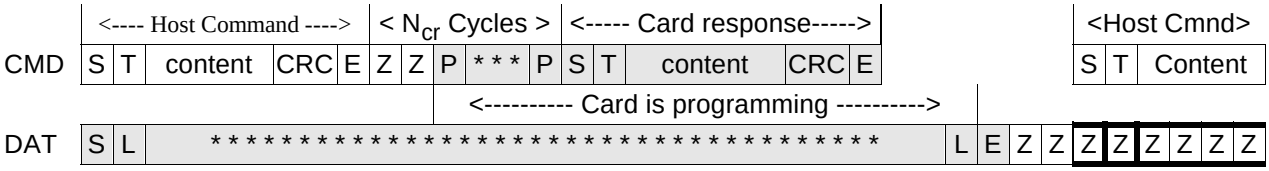


Figure 32: Stop transmission received after last data block. Card is busy programming.

:

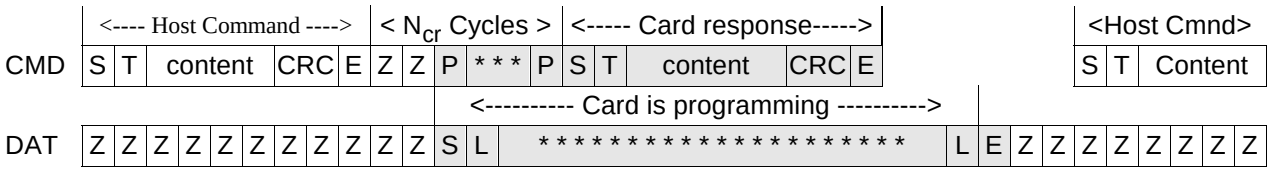


Figure 33: Stop transmission received after last data block. Card becomes busy.

- **Erase, Set and Clear Write Protect Timing.**

The host must first tag the start (CMD32) and end (CMD33) addresses of the range to be erased. The erase command (CMD38), once issued, will erase all the selected write blocks. Similarly, set and clear write protect commands start a programming operation as well. The card will signal “busy” (by pulling the DAT line low) for the duration of the erase or programming operation. The bus transaction timings are the same as given for stop tran command in Figure 33.

- **Reselecting a busy card**

When a busy card which is currently in the dis state is reselected it will reinstate its busy signaling on the data line. The timing diagram for this command / response / busy transaction is the same as given for stop tran command in Figure 33.

4.12.4 Timing Values

Table 26 defines all timing values.

	Min	Max	Unit
N_{CR}	2	64	clock cycles
N_{ID}	5	5	clock cycles
N_{AC}(note 1)	2	-	clock cycles
N_{RC}	8	-	clock cycles
N_{CC}	8	-	clock cycles
N_{WR}	2	-	clock cycles

(1) N_{AC} is the sum of TAAC + NSAC (refer to Chapter 5.3)

Table 26: Timing values

5 Card Registers

Within the card interface six registers are defined: OCR, CID, CSD, RCA, DSR and SCR. These can be accessed only by corresponding commands (see Chapter 4.7). The OCR, CID, CSD and SCR registers carry the card/content specific information, while the RCA and DSR registers are configuration registers storing actual configuration parameters.

5.1 OCR Register

The 32-bit operation conditions register stores the V_{DD} voltage profile of the card. In addition, this register includes a status information bit. This status bit is set if the card power up procedure has been finished. The OCR register shall be implemented by the cards which do not support the full operating voltage range of the SD Memory Card bus, or if the card power up extends the definition in the timing diagram (Figure 37).

OCR bit position	VDD voltage window
0-3	reserved
4	1.6-1.7
5	1.7-1.8
6	1.8-1.9
7	1.9-2.0
8	2.0-2.1
9	2.1-2.2
10	2.2-2.3
11	2.3-2.4
12	2.4-2.5
13	2.5-2.6
14	2.6-2.7
15	2.7-2.8
16	2.8-2.9
17	2.9-3.0
18	3.0-3.1
19	3.1-3.2
20	3.2-3.3
21	3.3-3.4
22	3.4-3.5
23	3.5-3.6
24-30	reserved
31	card power up status bit (busy)[1]

Table 27: OCR register definition

1)This bit is set to LOW if the card has not finished the power up routine

The supported voltage range is coded as shown in Table 27. A voltage range is not supported if the corresponding bit value is set to LOW. As long as the card is busy, the corresponding bit (31) is set

to LOW.

5.2 CID Register

The Card IDentification (CID) register is 128 bits wide. It contains the card identification information used during the card identification phase. Every individual flash card shall have a unique identification number. The structure of the CID register is defined in the following paragraphs:

Name	Field	Width	CID-slice
Manufacturer ID	MID	8	[127:120]
OEM/Application ID	OID	16	[119:104]
Product name	PNM	40	[103:64]
Product revision	PRV	8	[63:56]
Product serial number	PSN	32	[55:24]
reserved	--	4	[23:20]
Manufacturing date	MDT	12	[19:8]
CRC7 checksum	CRC	7	[7:1]
not used, always '1'	-	1	[0:0]

Table 28: The CID fields

- **MID**

An 8 bit binary number that identifies the card manufacturer. The MID number is controlled, defined and allocated to a SD Memory Card manufacturer by the SD Group. This procedure is established to ensure uniqueness of the CID register.

- **OID**

A 2 ASCII string characters that identifies the card OEM and/or the card contents (when used as a distribution media either on ROM or FLASH cards). The OID number is controlled, defined and allocated to a SD Memory Card manufacturer by the SD Group. This procedure is established to ensure uniqueness of the CID register.

- **PNM**

The product name is a string, 5 ASCII characters long.

- **PRV**

The product revision is composed of two Binary Coded Decimal (BCD) digits, four bits each, representing an "n.m" revision number. The "n" is the most significant nibble and "m" is the least significant nibble.

As an example, the PRV binary value field for product revision "6.2" will be: 0110 0010

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

- **PSN**

The Serial Number is 32 bits of binary number.

- **MDT**

The manufacturing date composed of two hexadecimal digits, one is 8 bit representing the year(y) and the other is four bits representing the month(m).

The “m” field [11:8] is the month code. 1 = January.

The “y” field [19:12] is the year code. 0 = 2000.

As an example, the binary value of the Date field for production date “April 2001” will be:

00000001 0100.

- **CRC**

CRC7 checksum (7 bits). This is the checksum of the CID contents computed according to Chapter 7.

5.3 CSD Register

The Card-Specific Data register provides information on how to access the card contents. The CSD defines the data format, error correction type, maximum data access time, whether the DSR register can be used etc. The programmable part of the register (entries marked by W or E, see below) can be changed by CMD27. The type of the entries in the table below is coded as follows: R = readable, W(1) = writable once, W = multiple writable.

Name	Field	Width	Cell Type	CSD-slice
CSD structure	CSD_STRUCTURE	2	R	[127:126]
reserved	-	6	R	[125:120]
data read access-time-1	TAAC	8	R	[119:112]
data read access-time-2 in CLK cycles (NSAC*100)	NSAC	8	R	[111:104]
max. data transfer rate	TRAN_SPEED	8	R	[103:96]
card command classes	CCC	12	R	[95:84]
max. read data block length	READ_BL_LEN	4	R	[83:80]
partial blocks for read allowed	READ_BL_PARTIAL	1	R	[79:79]
write block misalignment	WRITE_BLK_MISALIGN	1	R	[78:78]
read block misalignment	READ_BLK_MISALIGN	1	R	[77:77]
DSR implemented	DSR_IMP	1	R	[76:76]
reserved	-	2	R	[75:74]
device size	C_SIZE	12	R	[73:62]
max. read current @V_{DD} min	VDD_R_CURR_MIN	3	R	[61:59]
max. read current @V_{DD} max	VDD_R_CURR_MAX	3	R	[58:56]
max. write current @V_{DD} min	VDD_W_CURR_MIN	3	R	[55:53]

Table 29: The CSD Register fields

Name	Field	Width	Cell Type	CSD-slice
max. write current @V_{DD} max	VDD_W_CURR_MAX	3	R	[52:50]
device size multiplier	C_SIZE_MULT	3	R	[49:47]
erase single block enable	ERASE_BLK_EN	1	R	[46:46]
erase sector size	SECTOR_SIZE	7	R	[45:39]
write protect group size	WP_GRP_SIZE	7	R	[38:32]
write protect group enable	WP_GRP_ENABLE	1	R	[31:31]
reserved for MultiMediaCard compatibility		2	R	[30:29]
write speed factor	R2W_FACTOR	3	R	[28:26]
max. write data block length	WRITE_BL_LEN	4	R	[25:22]
partial blocks for write allowed	WRITE_BL_PARTIAL	1	R	[21:21]
reserved	-	5	R	[20:16]
File format group	FILE_FORMAT_GRP	1	R/W(1)	[15:15]
copy flag (OTP)	COPY	1	R/W(1)	[14:14]
permanent write protection	PERM_WRITE_PROTECT	1	R/W(1)	[13:13]
temporary write protection	TMP_WRITE_PROTECT	1	R/W	[12:12]
File format	FILE_FORMAT	2	R/W(1)	[11:10]
reserved		2	R/W	[9:8]
CRC	CRC	7	R/W	[7:1]
not used, always'1'	-	1	-	[0:0]

Table 29: The CSD Register fields

The following sections describe the CSD fields and the relevant data types. If not explicitly defined otherwise, all bit strings are interpreted as binary coded numbers starting with the left bit first.

- **CSD_STRUCTURE**

Version number of the related CSD structure.

CSD_STRUCTURE	CSD structure version	Valid for SD Memory Card Physical Specification Version
0	CSD version No. 1.0	Version 1.0
1-3	reserved	

Table 30: CSD register structure

- **TAAC**

Defines the asynchronous part of the data access time.

TAAC bit position	code
2:0	time unit 0=1ns, 1=10ns, 2=100ns, 3=1µs, 4=10µs, 5=100µs, 6=1ms, 7=10ms
6:3	time value 0=reserved, 1=1.0, 2=1.2, 3=1.3, 4=1.5, 5=2.0, 6=2.5, 7=3.0, 8=3.5, 9=4.0, A=4.5, B=5.0, C=5.5, D=6.0, E=7.0, F=8.0
7	reserved

Table 31: TAAC access time definition

- **NSAC**

Defines the worst case for the clock dependent factor of the data access time. The unit for NSAC is 100 clock cycles. Therefore, the maximal value for the clock dependent part of the data access time is 25.5k clock cycles.

The total access time N_{AC} as expressed in the Table is the sum of TAAC and NSAC. It has to be computed by the host for the actual clock rate. The read access time should be interpreted as a typical delay for the first data bit of a data block or stream.

- **TRAN_SPEED**

The following table defines the maximum data transfer rate per one data line - TRAN_SPEED:

TRAN_SPEED bit	code
2:0	transfer rate unit 0=100kbit/s, 1=1Mbit/s, 2=10Mbit/s, 3=100Mbit/s, 4... 7=reserved
6:3	time value 0=reserved, 1=1.0, 2=1.2, 3=1.3, 4=1.5, 5=2.0, 6=2.5, 7=3.0, 8=3.5, 9=4.0, A=4.5, B=5.0, C=5.5, D=6.0, E=7.0, F=8.0
7	reserved

Table 32: Maximum data transfer rate definition

Note that for current SD Memory Cards that field must be always 032h which is equal to 25MHz - the mandatory maximum operating frequency of SD Memory Card.

- **CCC**

The SD Memory Card command set is divided into subsets (command classes). The card command class register CCC defines which command classes are supported by this card. A value of ‘1’ in a CCC bit means that the corresponding command class is supported. For command class definition refer to Table 8.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

CCC bit	Supported card command class
0	class 0
1	class 1
.....	
11	class 11

Table 33: Supported card command classes

- **READ_BL_LEN**

The maximum read data block length is computed as $2^{READ_BL_LEN}$. The maximum block length might therefore be in the range 512...2048 bytes (see Chapter 4.11 for details). Note that in SD Memory Card the WRITE_BL_LEN is always equal to READ_BL_LEN

READ_BL_LEN	Block length	Remark
0-8	reserved	
9	2^9 = 512 Bytes	
.....		
11	2^{11} = 2048 Bytes	
12-15	reserved	

Table 34: Data block length

- **READ_BL_PARTIAL (always = 1 in SD Memory Card)**

Partial Block Read is always allowed in SD Memory Card. It means that smaller blocks can be used as well. The minimum block size will be one byte.

- **WRITE_BLK_MISALIGN**

Defines if the data block to be written by one command can be spread over more than one physical block of the memory device. The size of the memory block is defined in WRITE_BL_LEN.

WRITE_BLK_MISALIGN=0 signals that crossing physical block boundaries is invalid.

WRITE_BLK_MISALIGN=1 signals that crossing physical block boundaries is allowed.

- **READ_BLK_MISALIGN**

Defines if the data block to be read by one command can be spread over more than one physical block of the memory device. The size of the memory block is defined in READ_BL_LEN.

READ_BLK_MISALIGN=0 signals that crossing physical block boundaries is invalid.

READ_BLK_MISALIGN=1 signals that crossing physical block boundaries is allowed.

- **DSR_IMP**

Defines if the configurable driver stage is integrated on the card. If set, a driver stage register (DSR)

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

must be implemented also (see Chapter 5.5).

DSR_IMP	DSR type
0	no DSR implemented
1	DSR implemented

Table 35: DSR implementation code table

- **C_SIZE**

This parameter is used to compute the user's data card capacity (not include the security protected area). The memory capacity of the card is computed from the entries C_SIZE, C_SIZE_MULT and READ_BL_LEN as follows:

memory capacity = BLOCKNR * BLOCK_LEN

where

BLOCKNR = (C_SIZE+1) * MULT

MULT = $2^{C_SIZE_MULT+2}$ (C_SIZE_MULT < 8)

BLOCK_LEN = $2^{READ_BL_LEN}$, (READ_BL_LEN < 12)

Therefore, the maximal capacity which can be coded is 4096*512*2048 = 4 GBytes. Example: A 32 MByte card with BLOCK_LEN = 512 can be coded by C_SIZE_MULT = 3 and C_SIZE = 2000.

- **VDD_R_CURR_MIN, VDD_W_CURR_MIN**

The maximum values for read and write currents at the minimal power supply V_{DD} are coded as follows:

VDD_R_CURR_MIN VDD_W_CURR_MIN	code for current consumption @ V_{DD}
2:0	0=0.5mA; 1=1mA; 2=5mA; 3=10mA; 4=25mA; 5=35mA; 6=60mA; 7=100mA

Table 36: $V_{DD,min}$ current consumption

- **VDD_R_CURR_MAX, VDD_W_CURR_MAX**

The maximum values for read and write currents at the maximal power supply V_{DD} are coded as follows:

VDD_R_CURR_MAX VDD_W_CURR_MAX	code for current consumption @ V_{DD}
2:0	0=1mA; 1=5mA; 2=10mA; 3=25mA; 4=35mA; 5=45mA; 6=80mA; 7=200mA

Table 37: $V_{DD,max}$ current consumption

- **C_SIZE_MULT**

This parameter is used for coding a factor MULT for computing the total device size (see 'C_SIZE').

The factor MULT is defined as $2^{C_SIZE_MULT+2}$.

C_SIZE_MULT	MULT	Remark
0	$2^2 = 4$	
1	$2^3 = 8$	
2	$2^4 = 16$	
3	$2^5 = 32$	
4	$2^6 = 64$	
5	$2^7 = 128$	
6	$2^8 = 256$	
7	$2^9 = 512$	

Table 38: Multiply factor for the device size

- **ERASE_BLK_EN**

Defines whether erase of one write block (see WRITE_BL_LEN) is allowed (beside the SECTOR_SIZE given bellow).

If ERASE_BLK_EN = '0' erase area is unit of SECTOR_SIZE.

if ERASE_BLK_EN = '1' erase area is unit of SECTOR_SIZE or unit of WRITE_BL_LEN.

- **SECTOR_SIZE**

The size of an erasable sector. The contents of this register is a 7 bit binary coded value, defining the number of write blocks (see WRITE_BL_LEN). The actual size is computed by increasing this number by one. A value of zero means 1 write block, 127 means 128 write blocks.

- **WP_GRP_SIZE**

The size of a write protected group. The contents of this register is a 7 bit binary coded value, defining the number of erase sectors (see SECTOR_SIZE). The actual size is computed by increasing this number by one. A value of zero means 1 erase sector, 127 means 128 erase sectors.

- **WP_GRP_ENABLE**

A value of '0' means no group write protection possible.

- **R2W_FACTOR**

Defines the typical block program time as a multiple of the read access time. The following table defines the field format.

R2W_FACTOR	Multiples of read access time
0	1
1	2 (write half as fast as read)

Table 39: R2W_FACTOR

R2W_FACTOR	Multiples of read access time
2	4
3	8
4	16
5	32
6,7	reserved

Table 39: R2W_FACTOR

- **WRITE_BL_LEN**

The maximum write data block length is computed as $2^{WRITE_BL_LEN}$. The maximum block length might therefore be in the range from 512 up to 2048 bytes. Write Block Length of 512 bytes is always supported.

Note that in SD Memory Card the WRITE_BL_LEN is always equal to READ_BL_LEN.

WRITE_BL_LEN	Block length	Remark
0-8	reserved	
9	2^9 = 512 Bytes	
.....		
11	2^{11} = 2048 Bytes	
12-15	reserved	

Table 40: Data block length

- **WRITE_BL_PARTIAL**

Defines whether partial block sizes can be used in block write commands.

WRITE_BL_PARTIAL='0' means that only the WRITE_BL_LEN block size and its partial derivatives, in resolution of units of 512 bytes, can be used for block oriented data write.

WRITE_BL_PARTIAL='1' means that smaller blocks can be used as well. The minimum block size is one byte.

- **FILE_FORMAT_GRP**

Indicates the selected group of file formats. This field is read-only for ROM. The usage of this field is shown in Table 41 (see FILE_FORMAT).

- **COPY**

Defines if the contents is original (= '0') or has been copied ('1'). The COPY bit for OTP and MTP devices, sold to end consumers, is set to '1' which identifies the card contents as a copy. The COPY bit is an one time programmable bit.

- **PERM_WRITE_PROTECT**

Permanently protects the whole card content against overwriting or erasing (all write and erase

commands for this card are permanently disabled). The default value is '0', i.e. not permanently write protected.

- **TMP_WRITE_PROTECT**

Temporarily protects the whole card content from being overwritten or erased (all write and erase commands for this card are temporarily disabled). This bit can be set and reset. The default value is '0', i.e. not write protected.

- **FILE_FORMAT**

Indicates the file format on the card. This field is read-only for ROM. The following formats are defined:

FILE_ FORMAT_ GRP	FILE_FORMAT	Type
0	0	Hard disk-like file system with partition table
0	1	DOS FAT (floppy-like) with boot sector only (no partition table)
0	2	Universal File Format
0	3	Others / Unknown
1	0, 1, 2, 3	Reserved

Table 41: File formats

A more detailed description is given in SD Memory Card File System specification.

- **CRC**

The CRC field carries the check sum for the CSD contents. It is computed according to Chapter 7.2. The checksum has to be recalculated by the host for any CSD modification. The default corresponds to the initial CSD contents.

The following table lists the correspondence between the CSD entries and the command classes. A '+' entry indicates that the CSD field affects the commands of the related command class.

	Command classes							
CSD Field	**0**	**2**	**4**	**5**	**6**	**7**	**8**	**9**
CSD_STRUCTURE	+	+	+	+	+	+	+	+
TAAC		+	+	+	+	+	+	
NSAC		+	+	+	+	+	+	

Table 42: Cross reference of CSD fields vs. command classes

	Command classes							
CSD Field	**0**	**2**	**4**	**5**	**6**	**7**	**8**	**9**
TRAN_SPEED		+	+					
CCC	+	+	+	+	+	+	+	+
READ_BL_LEN		+						
WRITE_BLK_MISALIGN			+					
READ_BLK_MISALIGN		+						
DSR_IMP	+	+	+	+	+	+	+	+
C_SIZE_MANT		+	+	+	+	+	+	
C_SIZE_EXP		+	+	+	+	+	+	
VDD_R_CURR_MIN		+						
VDD_R_CURR_MAX		+						
VDD_W_CURR_MIN			+	+	+	+	+	
VDD_W_CURR_MAX			+	+	+	+	+	
ERASE_BLK_EN				+	+	+	+	
SECTOR_SIZE				+	+	+	+	
WP_GRP_SIZE					+	+	+	
WP_GRP_ENABLE					+	+	+	
R2W_FACTOR			+	+	+	+	+	
WRITE_BL_LEN			+	+	+	+	+	
WRITE_BL_PARTIAL			+	+	+	+	+	
FILE_FORMAT_GRP								
COPY	+	+	+	+	+	+	+	+
PERM_WRITE_PROTECT	+	+	+	+	+	+	+	+
TMP_WRITE_PROTECT	+	+	+	+	+	+	+	+
FILE_FORMAT								
CRC	+	+	+	+	+	+	+	+

Table 42: Cross reference of CSD fields vs. command classes

5.4 RCA Register

The writable 16-bit relative card address register carries the card address that is published by the card during the card identification. This address is used for the addressed host-card communication after the card identification procedure. The default value of the RCA register is 0x0000. The value 0x0000 is reserved to set all cards into the *Stand-by State* with CMD7.

5.5 DSR Register (Optional)

The 16-bit driver stage register is described in detail in Chapter 6.5. It can be optionally used to improve the bus performance for extended operating conditions (depending on parameters like bus length, transfer rate or number of cards). The CSD register carries the information about the DSR register usage. The default value of the DSR register is 0x404.

5.6 SCR Register

In addition to the CSD register there is another configuration register that named - SD CARD Configuration Register (SCR). SCR provides information on SD Memory Card's special features that were configured into the given card. The size of SCR register is 64 bit. This register shall be set in the factory by the SD Memory Card manufacturer.

The following table describes the SCR register content.

Description	Field	Width	Cell Type	SCR Slice
SCR Structure	SCR_STRUCTURE	4	R	[63:60]
SD Memory Card - Spec. Version	SD_SPEC	4	R	[59:56]
data_status_after erases	DATA_STAT_AFTER_ERASE	1	R	[55:55]
SD Security Support	SD_SECURITY	3	R	[54:52]
DAT Bus widths supported	SD_BUS_WIDTHS	4	R	[51:48]
reserved	-	16	R	[47:32]
reserved for manufacturer usage	-	32	R	[31:0]

Table 43: The SCR Fields

- **SCR_STRUCTURE**

Version number of the related SCR structure in the SD Memory Card Physical Layer Specification.

CSD_STRUCTURE	CSD structure version	Valid for SD Physical Layer Specification Version
0	SCR version No. 1.0	Version 1.0
1-15	reserved	

Table 44: SCR register structure version

- **SD_SPEC**

Describes the SD Memory Card Physical Layer Specification version supported by this card.

SPEC_VERS	Physical Layer Specification Version Number
0	Version 1.0
1-15	reserved

Table 45: SD Memory Card Physical Layer Specification Version

- **DATA_STAT_AFTER_ERASE**

Defines the data status after erase, whether it is '0' or '1' (the status is card vendor dependent).

- **SD_SECURITY**

Describes the security algorithm supported by the card.

SD_SECURITY	Supported algorithm
0	no security
1	security protocol 1.0
2	security protocol 2.0
3 .. 7	reserved

Table 46: SD Supported security algorithm

Security Protocol 1.0 relates to Security Specification Version 0.96.

Security Protocol 2.0 relates to Security Specification Version 1.0.

Note that it is mandatory for a Writable SD Memory Card (OTP or R/W) to support Security Protocol.

For ROM (Read Only) type of SD Memory Card the security feature is optional.

- **SD_BUS_WIDTHS**

Describes all the DAT bus widths that are supported by this card.

SD_BUS_WIDTHS	Supported Bus Widths
Bit 0	1 bit (DAT0)
Bit 1	reserved
Bit 2	4 bit (DAT0-3)
Bit 3 [MSB]	reserved

Table 47: SD Memory Card Supported Bus Widths

Since SD Memory Card shall support at least the two bus modes 1bit or 4bit width then any SD Card shall set at least bits 0 and 2 (SD_BUS_WIDTH="0101").

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

CONFIDENTIAL

6 SD Memory Card Hardware Interface

The SD Memory Card has six communication lines and three supply lines:

- CMD: Command is a bidirectional signal. The host and card drivers are operating in push pull mode.
- DAT0-3: Data lines are bidirectional signals. Host and card drivers are operating in push pull mode
- CLK: Clock is a host to card signal. CLK operates in push pull mode
- V_{DD}: V_{DD} is the power supply line for all cards.
- V_{SS1}, V_{SS2} are two ground lines.

In addition to those lines that are connected to the internal card circuitry there are two contacts of the Write Protect/Card Detect switch that are part of the socket. Those contacts are not mandatory but if they are exist they should be connected as given in the following figure.

When DAT3 is used for card detection, R_{DAT} for DAT3 should be unconnected and an another resistor should be connected to the ground.

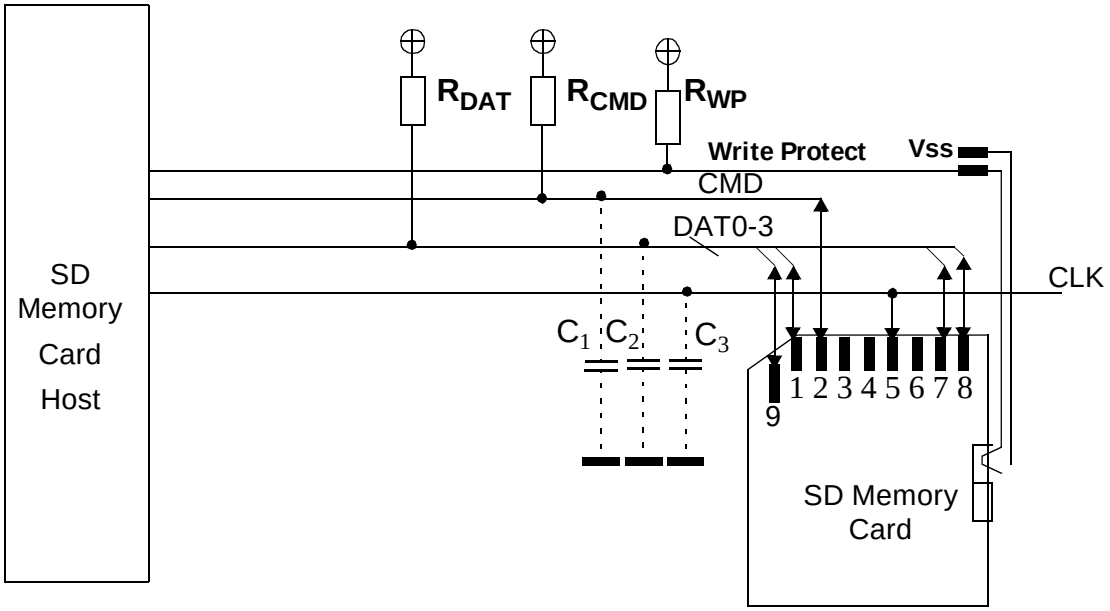


Figure 34: Bus circuitry diagram

R_{DAT} and R_{CMD} are pull-up resistors protecting the CMD and the DAT lines against bus floating when no card is inserted or when all card drivers are in an high-impedance mode.

The host shall pull-up all DAT0-3 lines by RDAT, even if the host uses SD Memory Card as 1 bit-mode-only in SD mode. Also, the host shall pull-up all "RSV" lines in SPI mode, eventhough they are not used.

R_{WP} is used for the Write Protect/Card Detection switch.

Refer to Chapter 6.6 for components values and conditions.

6.1 Hot insertion and removal

To guarantee the proper sequence of card pin connection during hot insertion, the use of either a

special hot-insertion capable card connector or an auto-detect loop on the host side (or some similar mechanism) is mandatory (see Chapter 8).

No card shall be damaged by inserting or removing a card into the SD Memory Card bus even when the power (V_{DD}) is up. Data transfer operations are protected by CRC codes, therefore any bit changes induced by card insertion and removal can be detected by the SD Memory Card bus master.

The inserted card must be properly reset also when CLK carries a clock frequency f_{PP}. Each card shall have power protection to prevent card (and host) damage. Data transfer failures induced by removal/insertion are detected by the bus master. They must be corrected by the application, which may repeat the issued command.

6.2 Card Detection (Insertion/Removal)

In order to be able to give feedback indication to the users, SD Memory Card system shall implement detection of card insertion or removal. One method is by sensing pin 1 of the card, and detecting the pull-up resistance on it. Detailed description of this and several other card detection options is given in Application Note “Card Detection Implementations”. In any case since it is not guaranty that all the MultiMediaCards in the market support the proposed card detection methods in SD Memory Card then the SD Memory Card host shall not ignore non-detected cards (empty sockets). If the host gets command from the user to operate a card it shall try to initialize a card in the socket even if no card insertion was previously reported - just in case that MultiMediaCards that do not support the card detection method is inside the socket.

6.3 Power protection (Insertion/Removal)

Cards shall be inserted/removed into/from the bus without damage. If one of the supply pins (V_{DD} or V_{SS}) is not connected properly, then the current is drawn through a data line to supply the card.

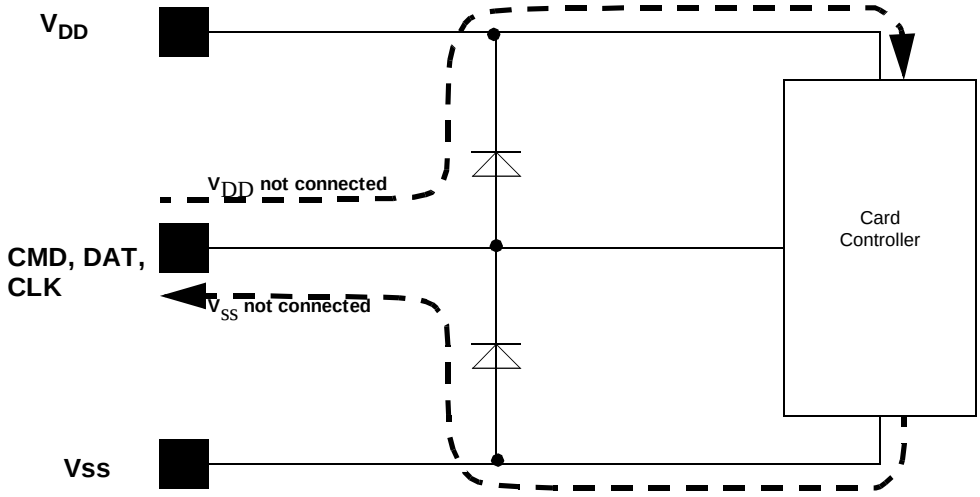


Figure 35: Improper power supply

Every card's output also shall be able to withstand shortcuts to either supply.

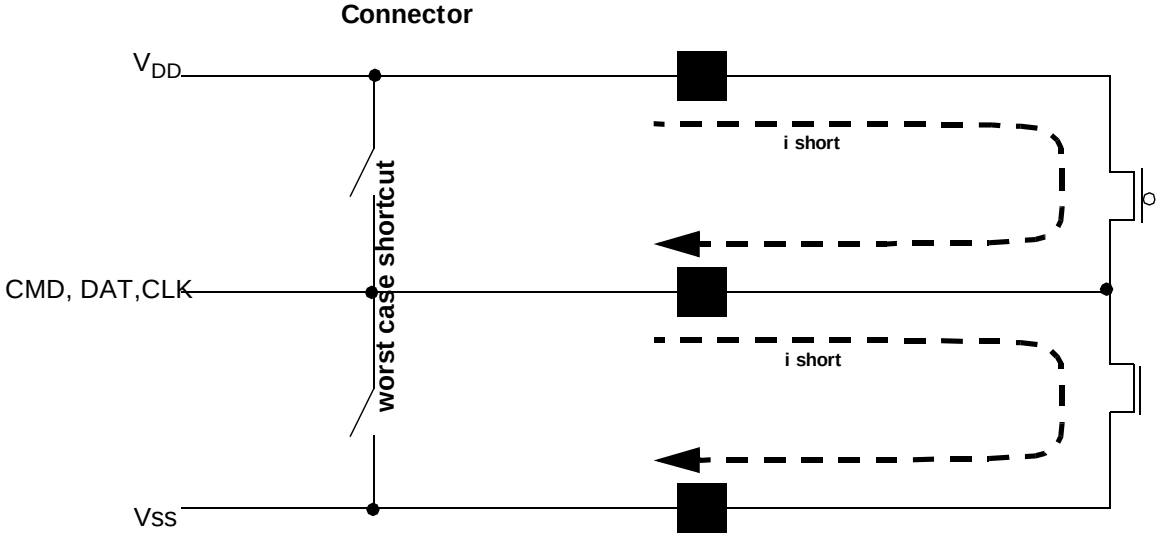


Figure 36: Short cut protection

If hot insertion feature is implemented in the host, than the host has to withstand an instant shortcut between V_{DD} and V_{SS} without damage.

6.4 Power up

The power up of the SD Memory Card bus is handled locally in each SD Memory Card and in the bus master.

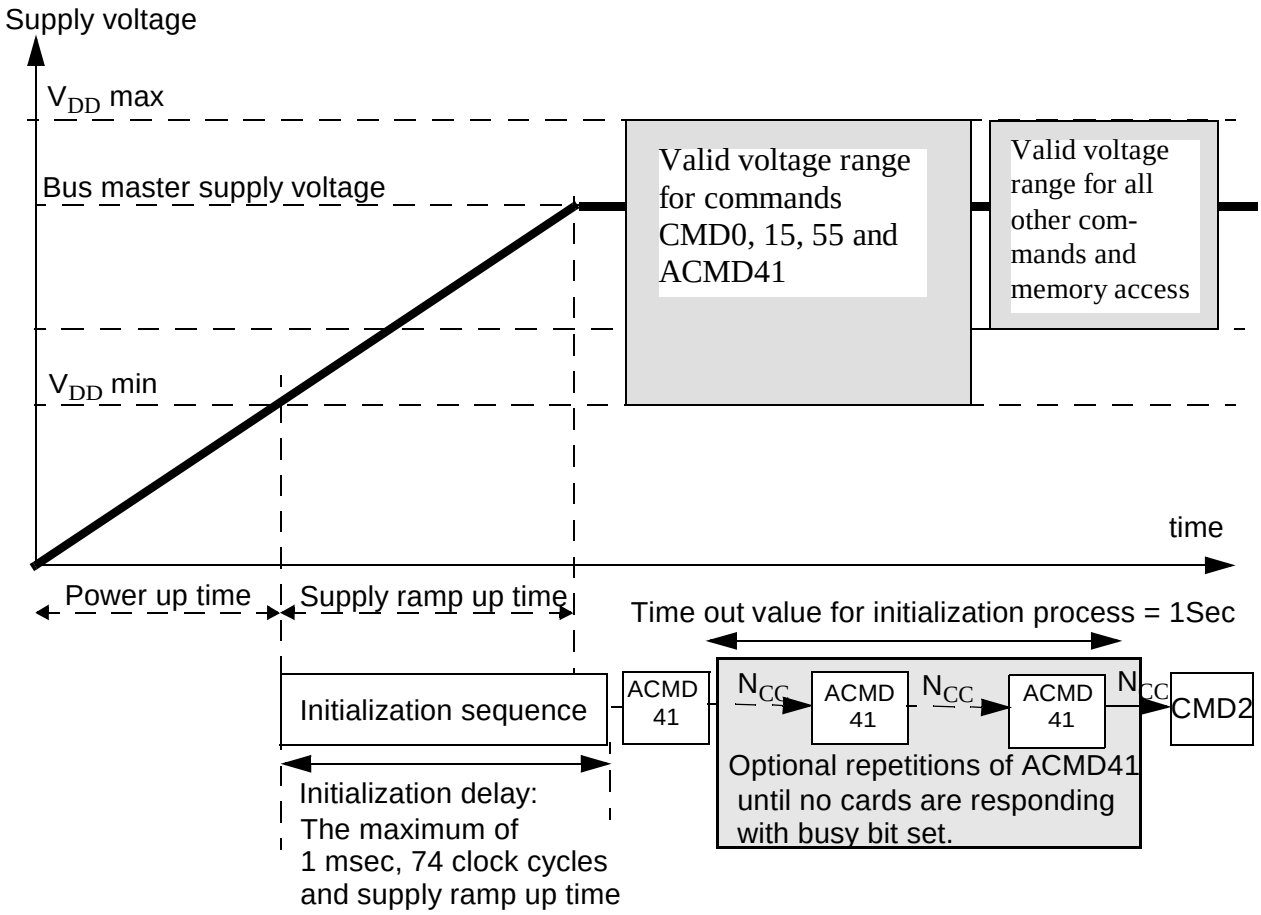


Figure 37: Power-up diagram

- After power up (including hot insertion, i.e. inserting a card when the bus is operating) the SD Memory Card enters the *idle state*. During this state the SD Memory Card ignores all bus transactions until ACMD41 is received (ACMD command type shall always precede with CMD55).
- ACMD41 is a special synchronization command used to negotiate the operation voltage range and to poll the cards until they are out of their power-up sequence. Besides the operation voltage profile of the cards, the response to ACMD41 contains a busy flag, indicating that the card is still working on its power-up procedure and is not ready for identification. This bit informs the host that the card is not ready. The host has to wait (and continue to poll the cards, each one on his turn) until this bit is cleared. The maximum period of power up procedure of single card shall not exceed 1 second.
- Getting individual cards, as well as the whole SD Memory Card system, out of *idle state* is up to the responsibility of the bus master. Since the power up time and the supply ramp up time depend on application parameters such as the maximum number of SD Memory Card s, the bus length and the power supply unit, the host must ensure that the power is built up to the operating level (the same level which will be specified in ACMD41) before ACMD41 is transmitted.
- After power up the host starts the clock and sends the initializing sequence on the CMD line. This sequence is a contiguous stream of logical '1's. The sequence length is the maximum of 1msec, 74 clocks or the supply-ramp-up-time; The additional 10 clocks (over the 64 clocks after what the card should be ready for communication) is provided to eliminate power-up synchronization problems.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

CONFIDENTIAL

- Every bus master shall have the capability to implement ACMD41 and CMD1. CMD1 will be used to ask MultiMediaCards to send their Operation Conditions. In any case the ACMD41 or the CMD1 shall be send separately to each card accessing it through its own CMD line.

6.5 Programmable card output driver (Optional)

The bus capacitance of each line of the SD Memory Card bus is the sum of the bus master capacitance, the bus capacitance itself and the capacitance of each inserted card. The sum of host and bus capacitance are fixed for one application, but may vary between different applications.The card load may vary in one application with each of the inserted cards.

In the following, programmable card output drivers for the push pull mode are described as an optional method for ensuring the defined maximum clock rate independently of the topology and of the number of inserted cards.

Both data and command driver stages in the push-pull mode have programmable peak current driving capabilities and programmable rise and fall times. The driver stage register (DSR) consists of two 8-bit latches. The contents of the latches is calculated from the required transfer speed of the interface and the bus load.

The CMD and DAT bus drivers consist of a predriver stage and a complementary driver transistor (Figure 38). The predriver stage output rise and fall time is set with the DSR1 register and determines the speed of the driver stage. The complementary driver transistor size is set with the DSR2 register and determines the current driving capabilities of the driver stage and also influences the peak current consumption of the bus driver. The proper combination of both allows the optimum bus performance.

Table 48 defines the DSR register contents:

DSR1	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
$t_{switch\text{-}on\ max}$	reserved				5ns	20ns	100ns	500ns
$t_{switch\text{-}on\ min}$					2ns	10ns	50ns	200ns

DSR2	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
$i_{peak\ min}$	reserved				100mA	20mA	5mA	1mA
$i_{peak\ max}$					200mA	50mA	10mA	2mA
$t_{rise\ typ}$					5ns	20ns	100ns	500ns

Table 48: DSR register contents

The time in DSR1 specifies the switch-on time of the output driver transistors. At the external interface, it is measurable as a delay time between the clock and driver stage output signal (e.g. for testing).

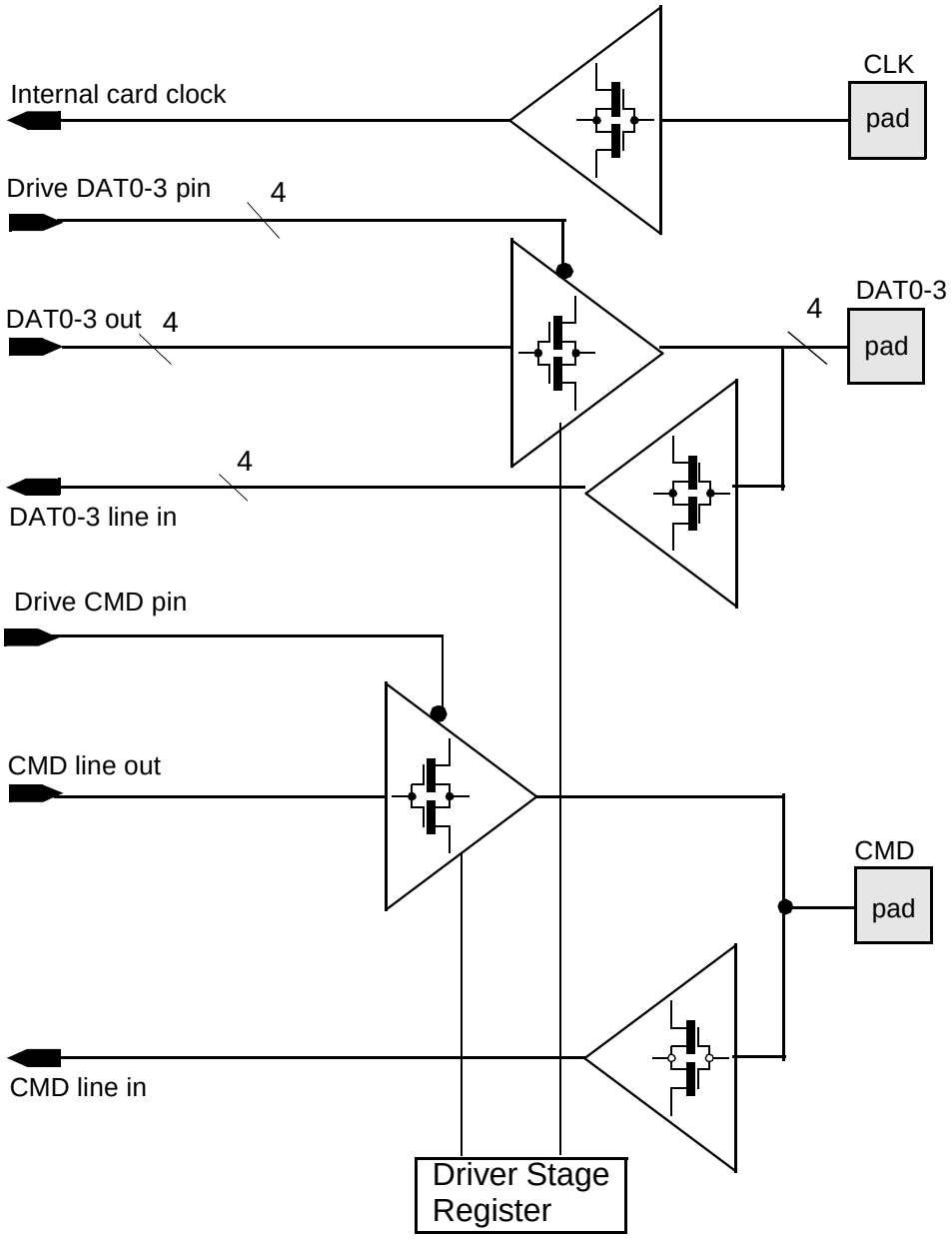


Figure 38: SD Memory Card bus driver

All data is valid for the specified operating range (voltage, temperature). Any combination of DSR1 and DSR2 bits may be programmed. DSR1 has to be programmed for the required clock frequency, where

$f_{clock} = (2\ t_{switch\text{-}on\ max})^{-1}$.

The DSR2 register must be programmed with the required driver size. Hints for the proper driver stage selection are part of future application notes (see Appendix).

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

6.6 Bus operating conditions

- **General**

Parameter	Symbol	Min	Max.	Unit	Remark
Peak voltage on all lines		-0.3	VDD+0.3	V	
All Inputs					
Input Leakage Current		-10	10	?A	
All Outputs					
Output Leakage Current		-10	10	?A	

Table 49: Bus Operating Cond. - General

- **Power supply voltage**

Parameter	Symbol	Min	Max.	Unit	Remark
Supply voltage	V_{DD}	**2.0**	**3.6**	**V**	**CMD0, 15, 55, ACMD41 commands**
Supply voltage specified in OCR register					**Except CMD0, 15, 55, ACMD41 commands**
Supply voltage differentials (V_{SS1}, V_{SS2})		**-0.3**	**0.3**	**V**	
Power up time			**250**	**mS**	**from 0v to V_{DD} Min.**

Table 50: Bus Operating Cond. - Power Supply Voltage

The current consumption of any card during the power-up period until the first command occure must not exceed 15 mA averaged over 1 Second. From first command until 'stby_state' (the state that the host may read CSD and verify the operating current consumptions) the maximum current consumtion may be 100mA averaged over 1 Sec. Note that a card that is delivered before fixing Physical Spec Ver 1.01, may not meet the above initialization current restrictions.

- **Bus signal line load**

The total capacitance C_L the CLK line of the SD Memory Card bus is the sum of the bus master capacitance C_{HOST}, the bus capacitance C_{BUS} itself and the capacitance C_{CARD} of each card connected to this line:

$C_L = C_{HOST} + C_{BUS} + N?C_{CARD}$

where N is the number of connected cards. Requiring the sum of the host and bus capacitances not to exceed 30 pF for up to 10 cards, and 40 pF for up to 30 cards, the following values must not be

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

exceeded:

Parameter	Symbol	Min	Max.	Unit	Remark
Pull-up resistance	R_{CMD} R_{DAT}	10	100	k?	to prevent bus floating
Bus signal line capacitance	C_L		250	pF	f_{PP} ? 5 MHz, 21 cards
Bus signal line capacitance	C_L		100	pF	f_{PP} ? 20 MHz, 7 cards
Single card capacitance	C_{CARD}		10	pF	
Maximum signal line inductance			16	nH	f_{PP} ? 20 MHz
Pull-up resistance inside card (pin1)	R_{DAT3}	10	90	k?	May be used for card detection

Table 51: Bus Operating Cond. - Signal Line's Load

Note that the total capacitance of CMD and DAT lines will be consist of C_{HOST}, C_{BUS} and one C_{CARD} only since they are connected separately to the SD Memory Card host.

6.7 Bus signal levels

As the bus can be supplied with a variable supply voltage, all signal levels are related to the supply voltage.

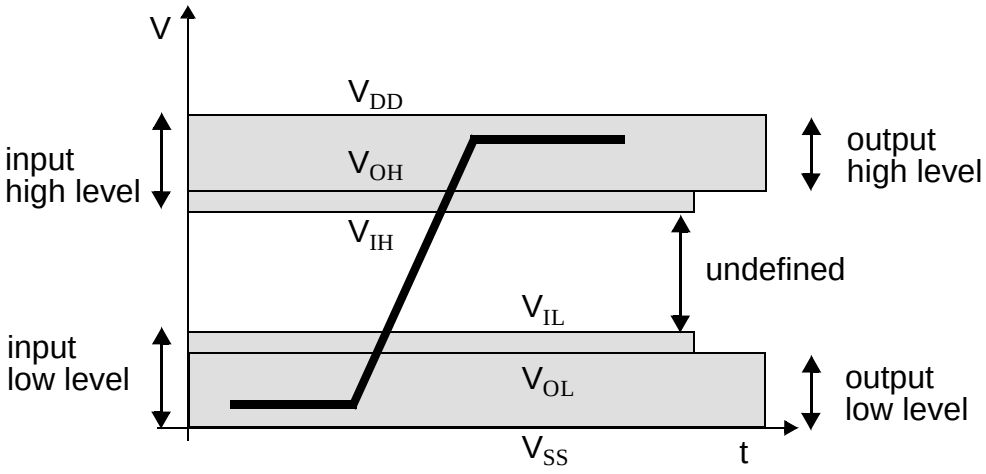


Figure 39: Bus signal levels

To meet the requirements of the JEDEC specification JESD8-1A, the card input and output voltages

shall be within the following specified ranges for any V_{DD} of the allowed voltage range:

Parameter	Symbol	Min	Max.	Unit	Conditions
Output HIGH voltage	V_{OH}	0.75?V_{DD}		V	I_{OH}=-100 ?A @V_{DD} min
Output LOW voltage	V_{OL}		0.125?V_{DD}	V	I_{OL}=100 ?A @V_{DD} min
Input HIGH voltage	V_{IH}	0.625?V_{DD}	V_{DD} + 0.3	V	
Input LOW voltage	V_{IL}	VSS-0.3	0.25?V_{DD}	V	

Table 52: Bus Signal Cond. - I/O Signal Voltages

6.8 Bus timing

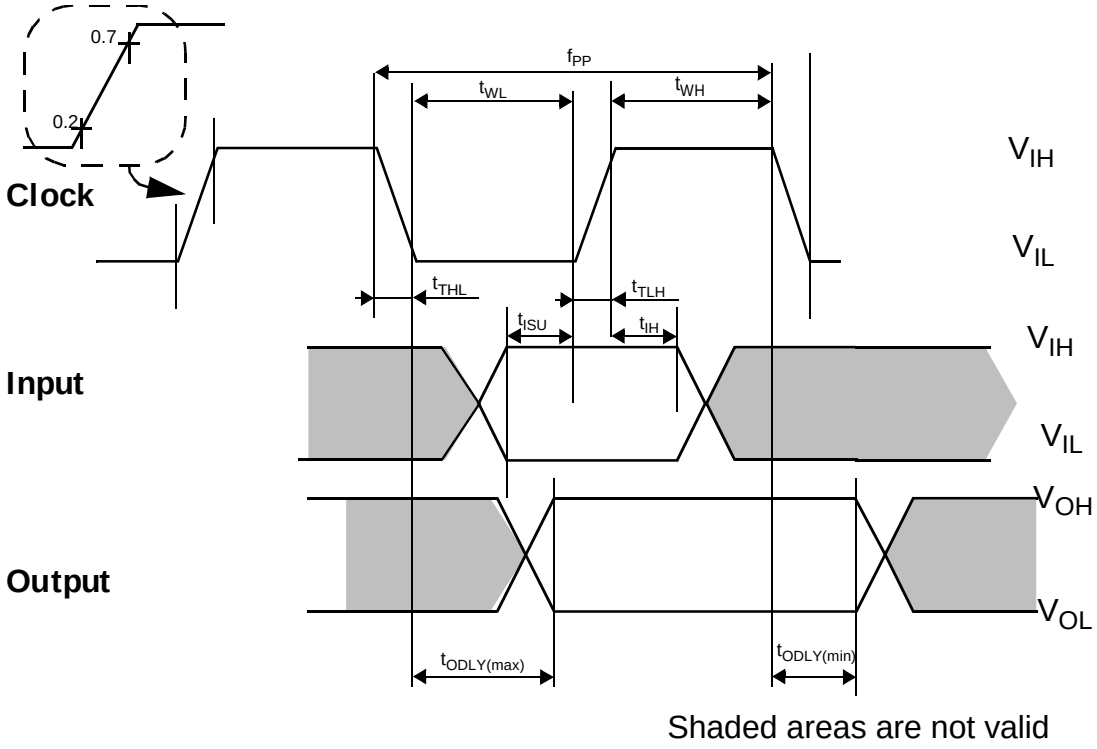


Figure 40: Timing diagram data input/output referenced to clock

Parameter	Symbol	Min	Max.	Unit	Remark
Clock CLK (All values are referred to min (V_{IH}) and max (V_{IL}),					
Clock frequency Data Transfer Mode	**f_{PP}**	**0**	**25**	**MHz**	**C_L ? 100 pF (7 cards)**

Parameter	Symbol	Min	Max.	Unit	Remark
Clock frequency Identification Mode (the low freq. is required for MultiMediaCard compatibility).	f_{OD}	**0(1)/ 100KHz**	**400**	**kHz**	**C_L ? 250 pF (21 cards)**
Clock low time	t_{WL}	**10**		**ns**	**C_L ? 100 pF (7 cards)**
Clock high time	t_{WH}	**10**		**ns**	**C_L ? 100 pF (7 cards)**
Clock rise time	t_{TLH}		**10**	**ns**	**C_L ? 100 pF (7 cards)**
Clock fall time	t_{THL}		**10**	**ns**	**C_L ? 100 pF (7 cards)**
Clock low time	t_{WL}	**50**		**ns**	**C_L ? 250 pF (21 cards)**
Clock high time	t_{WH}	**50**		**ns**	**C_L ? 250 pF (21 cards)**
Clock rise time	t_{TLH}		**50**	**ns**	**C_L ? 250 pF (21 cards)**
Clock fall time	t_{THL}		**50**	**ns**	**C_L ? 250 pF (21 cards)**
Inputs CMD, DAT (referenced to CLK)					
Input set-up time	t_{ISU}	**5**		**ns**	**C_L ? 25 pF (1 card)**
Input hold time	t_{IH}	**5**		**ns**	**C_L ? 25 pF (1 card)**
Outputs CMD, DAT (referenced to CLK)					
Output Delay time during Data Transfer Mode	t_{ODLY}	**0**	**14**	**ns**	**C_L ? 25 pF (1 card)**
Output Delay time during Identification Mode	t_{ODLY}	**0**	**50**	**ns**	**C_L ? 25 pF (1 card)**

(1) 0Hz means to stop the clock. The given minimum frequency range is for cases were continues clock is required (refer to Chapter 4.4 - Clock Control).

Table 53: Bus Timing - Parameters Values

6.9 SDLV Memory Card - Low Voltage SD Memory Cards

Next generation of SD Memory Cards will support operation at lower supply voltage.

Parameter	Symbol	Min	Max.	Unit	Remark
SDLV Memory Card Supply voltage	V_{DD}	**1.6**	**3.6**	**V**	**for Identification and regular operation**
Supply voltage differentials (V_{SS1}, V_{SS2})		**-0.2**	**0.2**	**V**	
Power up time			**250**	**mS**	**from 0v to V_{DD} Min.**

Table 54: SDLV Memory Card - Power Supply Voltage

The SDLV Memory Cards (low voltage cards) will support identification and operation voltage between 1.6V up to 3.6v. That means that the host may operate between 1.6 and 3.6 volts. Though it is strongly recommended that the host will be developed in such a way that the initialization will be done in the range of 2.0-3.6 volt. That will allow to communicate with regular SD Memory Cards at least for the initialization process and then to put them in inactive state in case that the host would prefer to work in a lower voltage range of SDLV Memory Card rather than SD Memory Card.

Although not mechanically different, new and old cards will not necessarily be interchangeable in a given system. The system definition allows an attempt to use old (high voltage) cards with new (low voltage) hosts. In case that the host supports initialization in the range of 2-3.6v the user will be informed that this is a wrong type of card.

In order to minimize customer dissatisfaction the SDLV Memory Cards will be labeled in such a way that the user will be able to distinguish between SDLV Memory Cards and regular SD Memory Cards (2.0-3.6v operation).

The marking method is to be defined **[T.B.D]**.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

7 SPI Mode

7.1 Introduction

The SPI mode consists of a secondary communication protocol which is offered by Flash-based SD Memory Cards. This mode is a subset of the SD Memory Card protocol, designed to communicate with a SPI channel, commonly found in Motorola's (and lately a few other vendors') microcontrollers. The interface is selected during the first reset command after power up (CMD0) and cannot be changed once the part is powered on.

The SPI standard defines the physical link only, and not the complete data transfer protocol. The SD Memory Card SPI implementation uses a subset of the SD Memory Card protocol and command set. The advantage of the SPI mode is the capability of using an off-the-shelf host, hence reducing the design-in effort to minimum. The disadvantage is the loss of performance of the SPI mode versus SD mode (e.g. Single data line and hardware CS signal per card).

7.2 SPI Bus Protocol

While the SD Memory Card channel is based on command and data bit streams which are initiated by a start bit and terminated by a stop bit, the SPI channel is byte oriented. Every command or data block is built of 8-bit bytes and is byte aligned to the CS signal (i.e. the length is a multiple of 8 clock cycles).

Similar to the SD Memory Card protocol, the SPI messages consist of command, response and data-block tokens. All communication between host and cards is controlled by the host (master). The host starts every bus transaction by asserting the CS signal low.

The response behavior in the SPI mode differs from the SD mode in the following three aspects:

- The selected card always responds to the command.
- An additional (8 bit) response structure is used
- When the card encounters a data retrieval problem, it will respond with an error response (which replaces the expected data block) rather than by a time-out as in the SD mode.

In addition to the command response, every data block sent to the card during write operations will be responded with a special data response token. A data block may be as big as one card write block (WRITE_BL_LEN) and as small as a single byte. Partial block read/write operations are enabled by card options specified in the CSD register.

7.2.1 Mode Selection

The SD Memory Card wakes up in the SD mode. It will enter SPI mode if the CS signal is asserted (negative) during the reception of the reset command (CMD0) and the card is in *idle_state*. If the card recognizes that the SD mode is required it will not respond to the command and remain in the SD mode. If SPI mode is required the card will switch to SPI and respond with the SPI mode R1 response.

The only way to return to the SD mode is by entering the power cycle. In SPI mode the SD Memory Card protocol state machine is not observed. All the SD Memory Card commands supported in SPI mode are always available.

During the initialization sequence, if the host gets Illegal Command indication for ACMD41 sent to the card, it may assume that the card is MultiMediaCard. In that case it should re-start the card as

MultiMediaCard using CMD0 and CMD1.

7.2.2 Bus Transfer Protection

Every SD Memory Card token transferred on the bus is protected by CRC bits. In SPI mode, the SD Memory Card offers a non protected mode which enables systems built with reliable data links to exclude the hardware or firmware required for implementing the CRC generation and verification functions.

In the non-protected mode the CRC bits of the command, response and data tokens are still required in the tokens. However, they are defined as ‘don’t care’ for the transmitter and ignored by the receiver.

The SPI interface is initialized in the non-protected mode. However, the RESET command (CMD0) which is used to switch the card to SPI mode, is received by the card while in SD mode and, therefore, must have a valid CRC field.

Since CMD0 has no arguments, the content of all the fields, including the CRC field, are constants and need not be calculated in run time. A valid reset command is:

0x40, 0x0, 0x0, 0x0, 0x0, 0x95

The host can turn the CRC option on and off using the CRC_ON_OFF command (CMD59).

7.2.3 Data Read

The SPI mode supports single block read and Multiple Block read operations (CMD17 or CMD18 in the SD Memory Card protocol). Upon reception of a valid read command the card will respond with a response token followed by a data token of the length defined in a previous SET_BLOCKLEN (CMD16) command (refer to Figure 41).

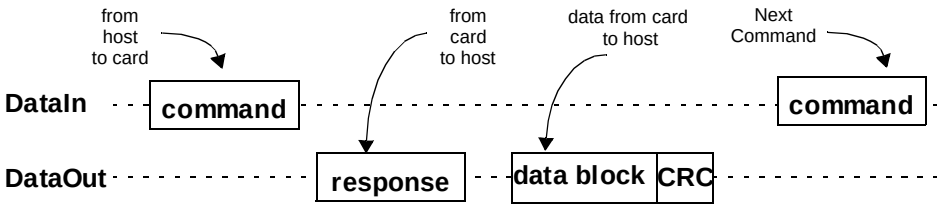


Figure 41: Single Block Read operation

A valid data block is suffixed with a 16 bit CRC generated by the standard CCITT polynomial $x^{16}+x^{12}+x^5+1$.

The maximum block length is given by READ_BL_LEN, defined in the CSD. If partial blocks are allowed (i.e. the CSD parameter READ_BL_PARTIAL equals 1), the block length can be any number between 1 and the maximum block size. Otherwise, the only valid block length for data read is given by READ_BL_LEN.

The start address can be any byte address in the valid address range of the card. Every block, however, must be contained in a single physical card sector.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

In case of a data retrieval error, the card will not transmit any data. Instead, a special data error token will be sent to the host. Figure 42 shows a data read operation which terminated with an error token rather than a data block.

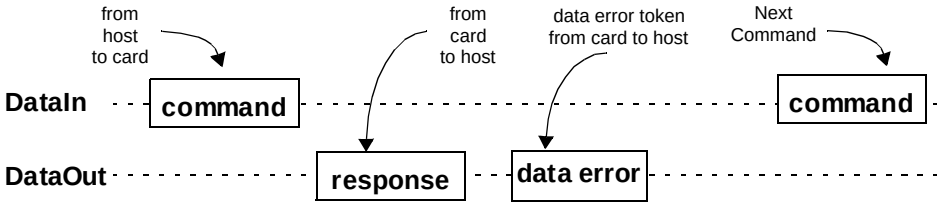


Figure 42: Read operation - data error

In case of Multiple block read operation every transferred block has its suffixed of 16 bit CRC.

Stop transmission command (CMD12) will actually stop the data transfer operation (the same as in SD Memory Card operation mode).

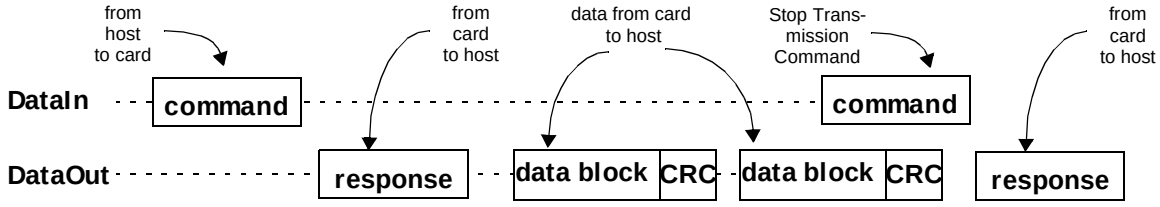


Figure 43: Multiple Block Read operation

7.2.4 Data Write

In SPI mode the SD Memory Card supports single block and Multiple block write commands. Upon reception of a valid write command (CMD24 or CMD25 in the SD Memory Card protocol), the card will respond with a response token and will wait for a data block to be sent from the host. CRC suffix, block length and start address restrictions are (with the exception of the CSD parameter WRITE_BL_PARTIAL controlling the partial block write option) identical to the read operation (see Figure 44).

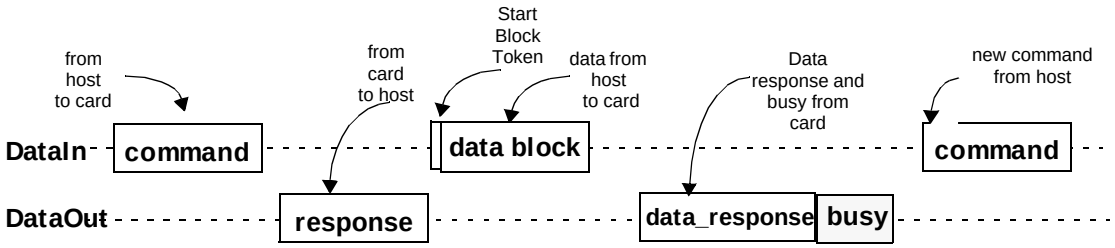


Figure 44: Single Block Write operation

Every data block has a prefix of 'Start Block' token (one byte).

After a data block has been received, the card will respond with a data-response token. If the data block has been received without errors, it will be programmed. As long as the card is busy programming, a continuous stream of busy tokens will be sent to the host (effectively holding the DataOut line low).

Once the programming operation is completed, the host must check the results of the programming using the SEND_STATUS command (CMD13). Some errors (e.g. address out of range, write protect violation etc.) are detected during programming only. The only validation check performed on the data block and communicated to the host via the data-response token is the CRC and general Write Error indication.

In Multiple Block write operation the stop transmission will be done by sending 'Stop Tran' token instead of 'Start Block' token at the beginning of the next block. In case of Write Error indication (on the data response) the host shall use SEND_NUM_WR_BLOCKS (ACMD22) in order to get the number of well written write blocks. The data tokens description is given in Chapter 7.3.3.

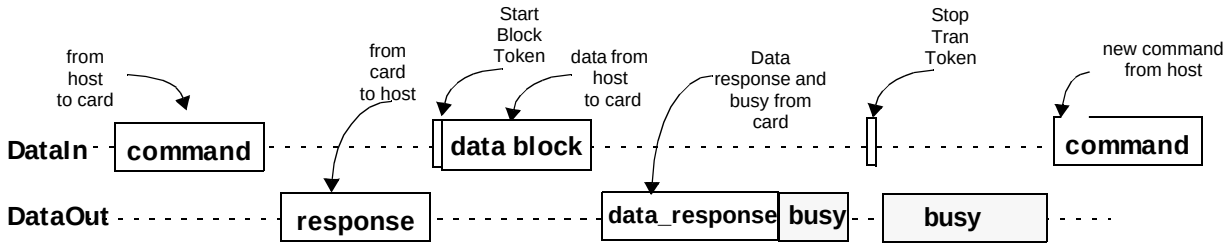


Figure 45: Multiple Block Write operation

While the card is busy, resetting the CS signal will not terminate the programming process. The card will release the DataOut line (tri-state) and continue with programming. If the card is reselected before the programming is finished, the DataOut line will be forced back to low and all commands will be rejected.

Resetting a card (using CMD0) will terminate any pending or active programming operation. This may destroy the data formats on the card. It is in the responsibility of the host to prevent it.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

7.2.5 Erase & Write Protect Management

The erase and write protect management procedures in the SPI mode are identical to those of the SD mode. While the card is erasing or changing the write protection bits of the predefined sector list, it will be in a busy state and hold the DataOut line low. Figure 46 illustrates a 'no data' bus transaction with and without busy signalling.

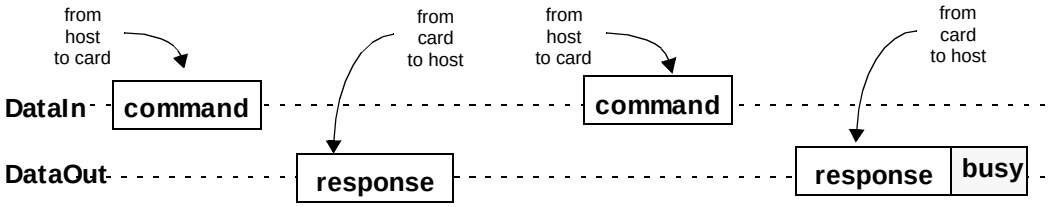


Figure 46: 'No data' operations

7.2.6 Read CID/CSD Registers

Unlike the SD Memory Card protocol (where the register contents is sent as a command response), reading the contents of the CSD and CID registers in SPI mode is a simple read-block transaction. The card will respond with a standard response token (see Figure 42) followed by a data block of 16 bytes suffixed with a 16 bit CRC.

The data time out for the CSD command cannot be set to the cards TAAC since this value is stored in the card's CSD. Therefore the standard response time-out value (N_{CR}) is used for read latency of the CSD register.

7.2.7 Reset Sequence

The SD Memory Card requires a defined reset sequence. After power on reset or CMD0 (software reset) the card enters an idle state. At this state the only valid host commands are ACMD41 (SD_SEND_OP_COND), CMD58 (READ_OCR) and CMD59 (CRC_ON_OFF). For the Thick (2.1mm) SD Memory Card - CMD1 (SEND_OP_COND) is also valid - that means that in SPI mode CMD1 and ACMD41 have the same behavior, though the usage of CMD41 is preferable since it allows easy distinguishing between SD Memory Card and MultiMediaCard. **For the Thin (1.4mm) SD Memory Card CMD1 (SEND_OP_COND) is illegal command during the initialization that is done after power on. After Power On, once the card accepted valid ACMD41, it will be able to accept also CMD1 even if used after re-initializing (CMD0) the card.** It was defined in such way in order to be able to distinguish between Thin SD Memory Card and MultiMediaCards (that supports CMD1 as well).

The host must poll the card (by repeatedly sending CMD1 or ACMD41) until the 'in-idle-state' bit in the card response indicates (by being set to 0) that the card completed its initialization processes and is ready for the next command.

In SPI mode, as opposed to SD mode, ACMD41 (or CMD1 as well, for 2.1mm-SD Memory Card) has no operands and does not return the contents of the OCR register. Instead, the host may use CMD58 (available in SPI mode only) to read the OCR register. Furthermore, it is in the responsibility

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

CONFIDENTIAL

of the host to refrain from accessing cards that do not support its voltage range.

The usage of CMD58 is not restricted to the initializing phase only, but can be issued at any time.

7.2.8 Error Conditions

Unlike the SD Memory Card protocol, in the SPI mode the card will always respond to a command. The response indicates acceptance or rejection of the command. A command may be rejected in any one of the following cases:

- It is sent while the card is in read operation (except CMD12 which is legal).
- It is sent wile the card is in Busy.
- Card is Locked and it is other than Class 0 or 7 commands.
- It is not supported (illegal opcode).
- CRC check failed.
- It contains an illegal operand.
- It was out of sequence during an erase sequence.

Note that in case the host sends command while the card sends data in read operation then the response with an illegal command indication may disturb the data transfer.

7.2.9 Memory Array Partitioning

Same as for SD mode.

7.2.10 Card Lock/unlock

Usage of card lock and unlock commands in SPI mode is identical to SD mode. In both cases the command is responded with a R1b response type. After the busy signal clears, the host should obtain the result of the operation by issuing a GET_STATUS command. Refer to Chapter 4.3.6 for details.

7.2.11 Application Specific commands

Identical to SD mode with the exception of the APP_CMD status bit (refer to Chapter 4.10.1) which is not available in SPI.

7.2.12 Copyright Protection commands

All the special Copyright Protection ACMDs and security functionality is the same as for SD mode.

7.3 SPI Mode Transaction Packets

7.3.1 Command Tokens

- **Command Format**

All the SD Memory Card commands are 6 bytes long. The command transmission always starts

with the left bit of the bitstring corresponding to the command codeword. All commands are protected by a CRC (see Chapter 7.2). The commands and arguments are listed in Table 57.

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'1'	x	x	x	'1'
Description	start bit	transmission bit	command index	argument	CRC7	end bit

Table 55: Command Format

• Command Classes

As in SD mode, the SPI commands are divided into several classes (See Table 56). Each class supports a set of card functions. A SD Memory Card will support the same set of optional command classes in both communication modes (there is only one command class table in the CSD register). The available command classes, and the supported command for a specific class, however, are different in the SD Memory Card and the SPI communication mode.

Note that except the classes that are not supported in SPI mode (class 1, 3 and 9), the mandatory required classes for the SD mode are the same for the SPI mode.

Card CMD Class (CCC)	**Class Description**	**Supported commands**																					
		0	**1**	**9**	**10**	**12**	**13**	**16**	**17**	**18**	**24**	**25**	**27**	**28**	**29**	**30**	**32**	**33**	**38**	**42**	**55**	**56**	**58**	**59**
class 0	Basic	+	+	+	+	+	+																+	+
class 1	Not supported in SPI																							
class 2	Block read							+	+	+														
class 3	Not supported in SPI																							
class 4	Block write										+	+	+											
class 5	Erase																+	+	+					
class 6	Write-protection (Optional)													+	+	+								
class 7	Lock Card (Optional)																			+				
class 8	Application specific																				+	+		
class 9	Not supported in SPI																							
class 10-11	Reserved																							

Table 56: Command classes in SPI mode

• Detailed Command Description

The following table provides a detailed description of the SPI bus commands. The responses are defined in Chapter 7.3.2. The Table 57 lists all SD Memory Card commands. A "yes" in the SPI mode colon indicates that the command is supported in SPI mode. With these restrictions, the command class description in the CSD is still valid. If a command does not require an argument, the value of this field should be set to zero. The reserved commands are reserved in SD mode as well.

The binary code of a command is defined by the mnemonic symbol. As an example, the content of

the **command index** field is (binary) '000000' for CMD0 and '100111' for CMD39.

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD0	Yes	None	R1	GO_IDLE_STATE	resets the SD Memory Card
CMD1	Yes[7]	None	R1	SEND_OP_ COND	Activates the card's initialization process (In Thin- 1.4mm SD Memory Card it is valid only if used after re-initializing a card - see note at Chapter 7.2.7)
CMD2	No				
CMD3	No				
CMD4	No				
CMD5	reserved				
CMD6	reserved				
CMD7	No				
CMD8	reserved				
CMD9	Yes	None	R1	SEND_CSD	asks the selected card to send its card-specific data (CSD)
CMD10	Yes	None	R1	SEND_CID	asks the selected card to send its card identification (CID)
CMD11	No				
CMD12	Yes	Non	R1b	STOP_ TRANSMISSION	forces the card to stop transmission in Multiple Block Read Operation
CMD13	Yes	None	R2	SEND_STATUS	asks the selected card to send its status register.
CMD14	reserved				
CMD15	No				
CMD16	Yes	[31:0] block length	R1	SET_BLOCKLEN	selects a block length (in bytes) for all following block commands (read and write).[1]
CMD17	Yes	[31:0] data address	R1	READ_SINGLE_ BLOCK	reads a block of the size selected by the SET_BLOCKLEN command.[2]
CMD18	Yes	[31:0] data address	R1	READ_MULTIPL E_BLOCK	continuously transfers data blocks from card to host until interrupted by a STOP_TRANSMISSION command.
CMD19	reserved				
CMD20	No				
CMD21 ... CMD23	reserved				
CMD24	Yes	[31:0] data address	R1	WRITE_BLOCK	writes a block of the size selected by the SET_BLOCKLEN command. [3]

Table 57: Commands and arguments

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD25	Yes	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	continuously writes blocks of data until 'Stop Tran' token is sent (instead 'Start Block').
CMD26	No				
CMD27	Yes	None	R1	PROGRAM_CSD	programming of the programmable bits of the CSD.
CMD28	Yes	[31:0] data address	R1b[4]	SET_WRITE_PROT	if the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	Yes	[31:0] data address	R1b	CLR_WRITE_PROT	if the card has write protection features, this command clears the write protection bit of the addressed group.
CMD30	Yes	[31:0] write protect data address	R1	SEND_WRITE_PROT	if the card has write protection features, this command asks the card to send the status of the write protection bits. [5]
CMD31	reserved				
CMD32	Yes	[31:0] data address	R1	ERASE_WR_BLK_START_ADDR	sets the address of the first write block to be erased.
CMD33	Yes	[31:0] data address	R1	ERASE_WR_BLK_END_ADDR	sets the address of the last write block of the continuous range to be erased.
CMD34 ... CMD37	reserved				
CMD38	Yes	[31:0] stuff bits	R1b	ERASE	erases all previously selected write blocks
CMD39	No				
CMD40	No				
CMD41	reserved				
CMD42	Yes	[31:0] stuff bits.	R1	LOCK_UNLOCK	Used to Set/Reset the Password or lock/unlock the card. A transferred data block includes all the command details - refer to Chapter 4.3.6. The size of the Data Block is defined with SET_BLOCK_LEN command.
CMD43 ... CMD54	reserved				

Table 57: Commands and arguments

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD55	Yes	[31:0] stuff bits	R1	APP_CMD	Defines to the card that the next command is an application specific command rather than a standard command
CMD56	Yes	[31:1] stuff bits. [0]: RD/WR_[6]	R1	GEN_CMD	Used either to transfer a Data Block to the card or to get a Data Block from the card for general purpose / application specific commands. The size of the Data Block shall be defined with SET_BLOCK_LEN command.
CMD57	Reserved				
CMD58	Yes	None	R3	READ_OCR	Reads the OCR register of a card.
CMD59	Yes	[31:1] stuff bits [0:0] CRC option	R1	CRC_ON_OFF	Turns the CRC option on or off. A '1' in the CRC option bit will turn the option on, a '0' will turn it off
CMD60 -63	Reserved For Manufacturer				

Table 57: Commands and arguments

1)The default block length is as specified in the CSD.
2)The data transferred must not cross a physical block boundary unless READ_BLK_MISALIGN is set in the CSD.
3)The data transferred must not cross a physical block boundary unless WRITE_BLK_MISALIGN is set in the CSD.
4)R1b: R1 response with an optional trailing busy signal.
5) 32 write protection bits (representing 32 write protect groups starting at the specified address) followed by 16 CRC bits are transferred in a payload format via the data line. The last (least significant) bit of the protection bits corresponds to the first addressed group. If the addresses of the last groups are outside the valid range, then the corresponding write protection bits shall be set to zero.
6) **RD/WR_**: "1" the Host shall get a block of data from the card.
"0" the host sends block of data to the card.
7) CMD1 is valid command for the Thin (1.4mm) SD Memory Card only if used after re-initializing a card (not after power on reset).

The following table describes all the application specific commands supported/reserved by the SD Memory Card. All the following commands shall be preceded with APP_CMD (CMD55).

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
ACMD6	No				
ACMD13	yes	[31:0] stuff bits	R2	SD_STATUS	Send the SD Memory Card status. The status fields are given in Table 24
ACMD17	reserved				

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
ACMD18	yes	--	--	--	Reserved for SD security applications[1]
ACMD19 to ACMD21	reserved				
ACMD22	yes	[31:0] stuff bits	R1	SEND_NUM_WR_ BLOCKS	Send the numbers of the well written (without errors) blocks. Responds with 32bit+CRC data block.
ACMD23	yes	[31:23] stuff bits [22:0]Number of blocks	R1	SET_WR_BLK_ ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for faster Multiple Block WR command). "1"=default (one wr block)(2).
ACMD24	reserved				
ACMD25	yes	--	--	--	Reserved for SD security applications[1]
ACMD26	yes	--	--	--	Reserved for SD security applications[1]
ACMD38	yes	--	--	--	Reserved for SD security applications[1]
ACMD39 to ACMD40	reserved				
ACMD41	yes	None	R1	SEND_OP_ COND	Activates the card's initialization process.
ACMD42	yes	[31:1] stuff bits [0]set_cd	R1	SET_CLR_CARD_ DETECT	Connect[1]/Disconnect[0] the 50KOhm pull-up resistor on CD/DAT3 (pin 1) of the card. The pull-up may be used for card detection.
ACMD43 ... ACMD49	yes	--	--	--	Reserved for SD security applications[1]
ACMD51	yes	[31:0] staff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

(1) Refer to "SD Memory Card Security Specification" for detailed explanation about the SD Security Features
(2) Command STOP_TRAN (CMD12) shall be used to stop the transmission in Write Multiple Block whether the pre-erase (ACMD23) feature is used or not.

Table 58: Application Specific Commands used/reserved by SD Memory Card - SPI Mode

7.3.2 Responses

There are several types of response tokens. As in the SD mode, all are transmitted MSB first:

- **Format R1**

This response token is sent by the card after every command with the exception of SEND_STATUS commands. It is one byte long, and the MSB is always set to zero. The other bits are error indica-

tions, an error being signaled by a '1'. The structure of the R1 format is given in Figure 47. The meaning of the flags is defined as following:

- **In idle state**: The card is in idle state and running the initializing process.
- **Erase reset:** An erase sequence was cleared before executing because an out of erase sequence command was received.
- **Illegal command**: An illegal command code was detected.
- **Communication CRC error**: The CRC check of the last command failed.
- **Erase sequence error**: An error in the sequence of erase commands occurred.
- **Address error**:A misaligned address, which did not match the block length, was used in the command.
- **Parameter error**: The command's argument (e.g. address, block length) was out of the allowed range for this card.

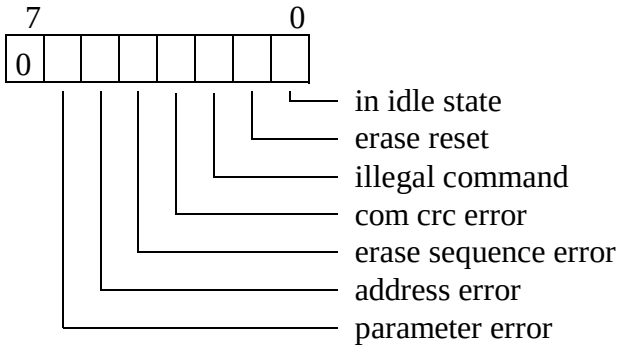


Figure 47: R1 Response Format

• Format R1b

This response token is identical to the R1 format with the optional addition of the busy signal. The busy signal token can be any number of bytes. A zero value indicates card is busy. A non-zero value indicates the card is ready for the next command.

• Format R2

This response token is two bytes long and sent as a response to the SEND_STATUS command. The format is given in Figure 48.

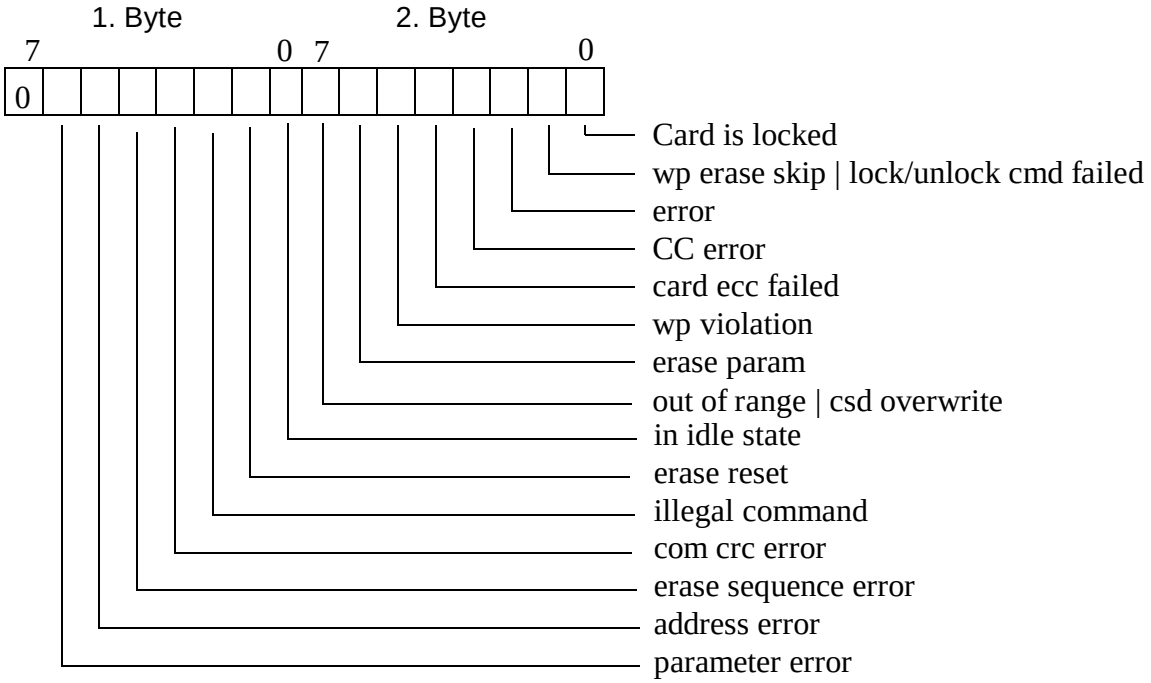


Figure 48: R2 response format

The first byte is identical to the response R1. The content of the second byte is described in the following:

- **Erase param**: An invalid selection, sectors or groups, for erase.
- **Write protect violation**: The command tried to write a write protected block.
- **Card ECC failed**: Card internal ECC was applied but failed to correct the data.
- **CC error**: Internal card controller error
- **Error**: A general or an unknown error occurred during the operation.
- **Write protect erase skip | lock/unlock command failed**: This status bit has two functions overloaded. It is set when the host attempts to erase a write protected sector or makes a sequence or password error during card lock/unlock operation.
- **Card is locked**: Set when the card is locked by the user. Reset when it is unlocked.

• Format R3

This response token is sent by the card when a READ_OCR command is received. The response length is 5 bytes (see Figure 49). The structure of the first (MSB) byte is identical to response type R1. The other four bytes contain the OCR register.

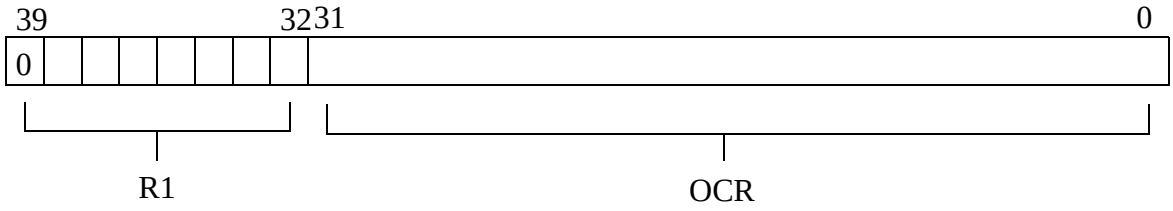
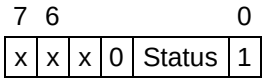


Figure 49: R3 Response Format

- **Data Response**

Every data block written to the card will be acknowledged by a data response token. It is one byte long and has the following format:



The meaning of the status bits is defined as follows:

'010' - Data accepted.

'101' - Data rejected due to a CRC error.

'110' - Data Rejected due to a Write Error

In case of any error (CRC or Write Error) during Write Multiple Block operation, the host shall stop the data transmission using CMD12. In case of Write Error (response '110') the host may send CMD13 (SEND_STATUS) in order to get the cause of the write problem. ACMD22 can be used to find the number of well written write blocks.

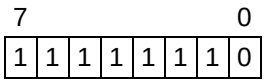
7.3.3 Data Tokens

Read and write commands have data transfers associated with them. Data is being transmitted or received via data tokens. All data bytes are transmitted MSB first.

Data tokens are 4 to 515 bytes long and have the following format:

For Single Block Read, Single Block Write and Multiple Block Read:

- First byte: Start Block

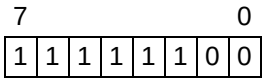


- Bytes 2-513 (depends on the data block length): User data
- Last two bytes: 16 bit CRC.

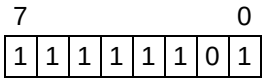
For Multiple Block Write operation:

- First byte of each block:

If data is to be transferred then - Start Block



If Stop transmission is requested - Stop Tran



Note that this format is used only for Multiple Block Write. In case of Multiple Block Read the stop transmission is done using STOP_TRAN Command (CMD12).

7.3.4 Data Error Token

If a read operation fails and the card cannot provide the required data, it will send a data error token instead. This token is one byte long and has the following format:

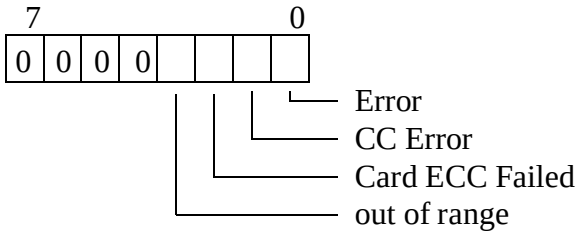


Figure 50: Data Error Token

The 4 least significant bits (LSB) are the same error bits as in the response format R2.

7.3.5 Clearing Status Bits

As described in the previous paragraphs, in SPI mode, status bits are reported to the host in three different formats: response R1, response R2 and data error token (the same bits may exist in multiple response types - e.g Card ECC failed)

As in the SD mode, error bits are cleared when read by the host, regardless of the response format. State indicators are either cleared by reading or in accordance with the card state.

The following table summarizes the set and clear conditions for the various status bits:

.

Identifier	Included in resp	Type[1]	Value	Description	Clear Condition[2]
Out of range	R2 DataErr	E R X	'0'= no error '1'= error	The command argument was out of the allowed range for this card.	C
Address error	R1 R2	E R X	'0'= no error '1'= error	A misaligned address which did not match the block length was used in the command.	C
Erase sequence error	R1 R2	E R	'0'= no error '1'= error	An error in the sequence of erase commands occurred.	C
Erase param	R2	E X	'0'= no error '1'= error	An error in the parameters of the erase command sequence	C
Parameter error	R1 R2	E R X	'0'= no error '1'= error	An error in the parameters of the command	C
WP violation	R2	E R X	'0'= not protected '1'= protected	Attempt to program a write protected block.	C
Com CRC error	R1 R2	E R	'0'= no error '1'= error	The CRC check of the previous command failed.	C
Illegal command	R1 R2	E R	'0'= no error '1'= error	Command not legal for the card state	C
Card ECC failed	R2 DataEr	E X	'0'= success '1'= failure	Card internal ECC was applied but failed to correct the data.	C
CC error	R2 dataEr	E R X	'0'= no error '1'= error	Internal card controller error	C
Error	R2 dataEr	E R X	'0'= no error '1'= error	A general or an unknown error occurred during the operation.	C
CID/ CSD_OVERWRITE	R2	E R X	'0'= no error '1'= error	can be either one of the following errors: - The CID register has been already written and can not be overwritten - The read only section of the CSD does not match the card content. - An attempt to reverse the copy (set as original) or permanent WP (unprotected) bits was made.	C
WP erase skip	R2	S X	'0'= not protected '1'= protected	Only partial address space was erased due to existing write protected blocks.	C
Lock/Unlock cmd failed	R2	X	'0'= no error '1'= error	Sequence or password error during card lock/unlock operation	C

Table 59: SPI mode status bits

Identifier	Included in resp	Type[1]	Value	Description	Clear Condition[2]
Card is locked	R2	S X	'0' = card is not locked '1' = card is locked	Card is locked by a user password and	A
Erase reset	R1 R2	S R	'0'= cleared '1'= set	An erase sequence was cleared before executing because an out of erase sequence command was received	C
In Idle state	R1 R2	S R	0 = Card is ready 1 = Card is in idle state	The card enters the idle state after power up or reset command. It will exit this state and become ready upon completion of its initialization procedures.	A

Table 59: SPI mode status bits

1) Type:

E: Error bit.

S: State bit.

R: Detected and set for the actual command response.

X: Detected and set during command execution. The host must poll the card by issuing the status command in order to read these bits.

2) Clear Condition:

A: According to the card current state.

C: Clear by read

7.4 Card Registers

In SPI mode only the OCR, CSD and CID registers are accessible. Their format is identical to the format in the SD mode. However, a few fields are irrelevant in SPI mode.

7.5 SPI Bus Timing Diagrams

All timing diagrams use the following schematics and abbreviations:

H	Signal is high (logical '1'
L	Signal is low (logical '0')
X	Don't care
Z	High impedance state (-> = 1)
*	Repeater
Busy	Busy Token
Command	Command token
Response	Response token
Data block	Data token

All timing values are defined in Table 60. The host must keep the clock running for at least N_{CR} clock cycles after receiving the card response. This restriction applies to both command and data response tokens.

7.5.1 Command / Response

- **Host Command to Card Response - Card is ready**

The following timing diagram describes the basic command response (no data) SPI transaction.

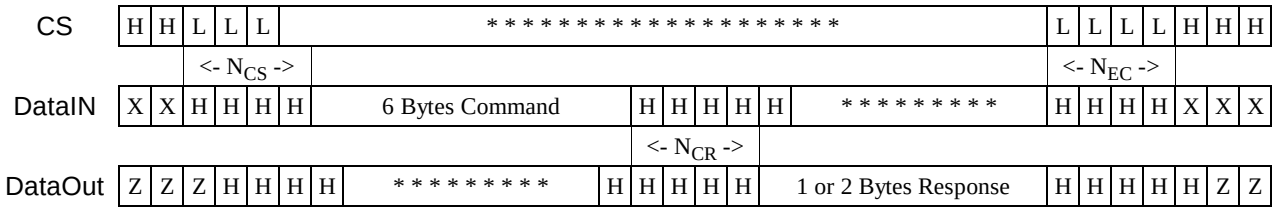


Figure 51: Basic command response

- **Host Command to Card Response - card is busy**

The following timing diagram describes the command response transaction for commands when the card responses which the R1b response type (e.g. SET_WRITE_PROT and ERASE). When the card is signaling busy, the host may deselect it (by raising the CS) at any time. The card will release the DataOut line one clock after the CS going high. To check if the card is still busy it needs to be reselected by asserting (set to low) the CS signal. The card will resume busy signal (pulling DataOut low) one clock after the falling edge of CS.

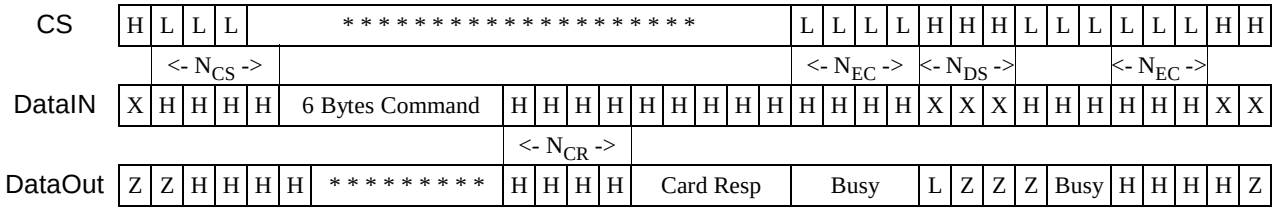


Figure 52: Command response with busy indication (R1b)

- **Card Response to Host Command**

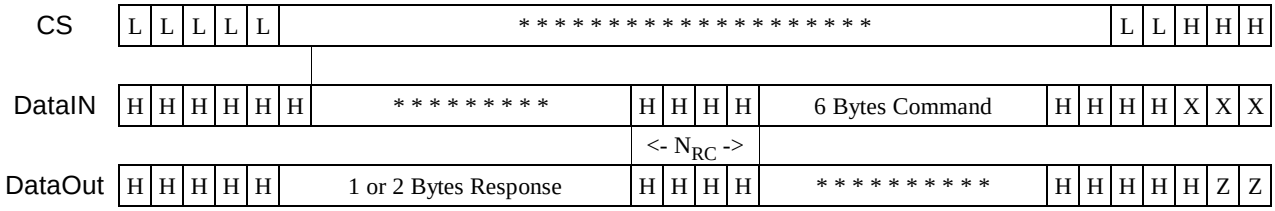


Figure 53: Timing between card response to new host command

7.5.2 Data read

- The following timing diagram describes all single block read operations with the exception of SEND_CSD and SEND_CID commands.

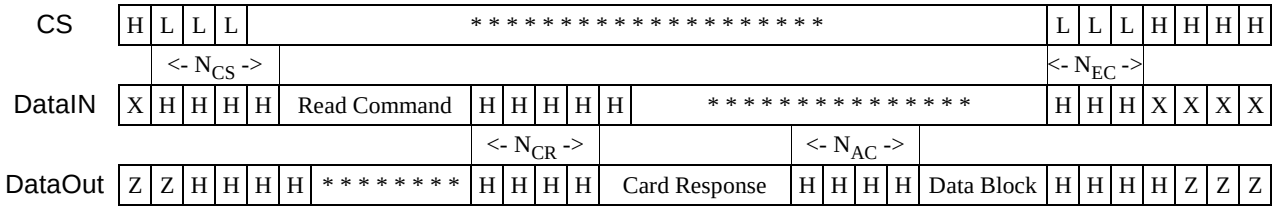


Figure 54: Read Single Block operations - bus timing

The following table describes Stop transmission operation in case of Multiple Block Read.

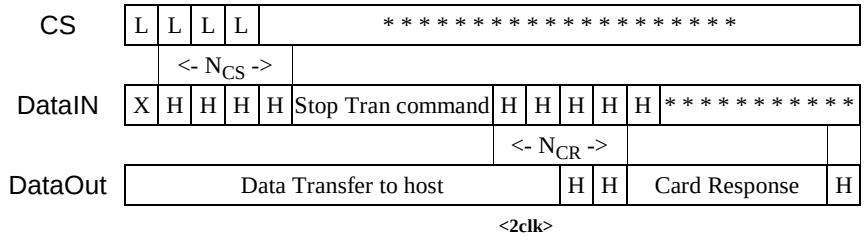


Figure 55: Stop Transmission in Read Multiple Block

- **Reading the CSD or CID register**

The following timing diagram describes the SEND_CSD and SEND_CID command bus transactions. The timeout values for the response and the data block are Ncr and Ncx respectively (Since the Nac is still unknown).

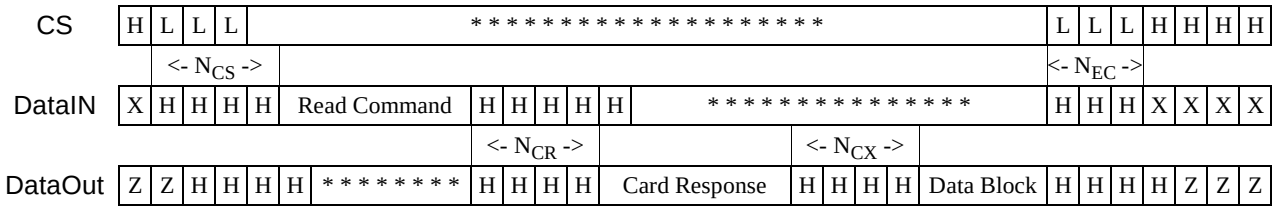


Figure 56: Read CSD / CID - bus timing

7.5.3 Data write

The host may deselect a card (by raising the CS) at any time during the card busy period (refer to the given timing diagram). The card will release the DataOut line one clock after the CS going high. To check if the card is still busy it needs to be reselected by asserting (set to low) the CS signal.

The card will resume busy signal (pulling DataOut low) one clock after the falling edge of CS.

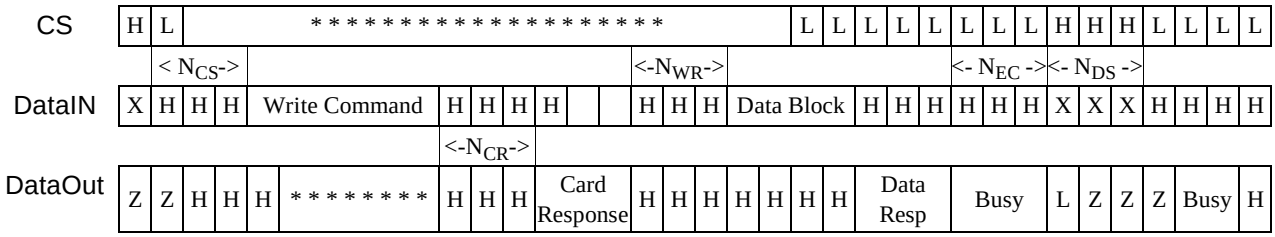
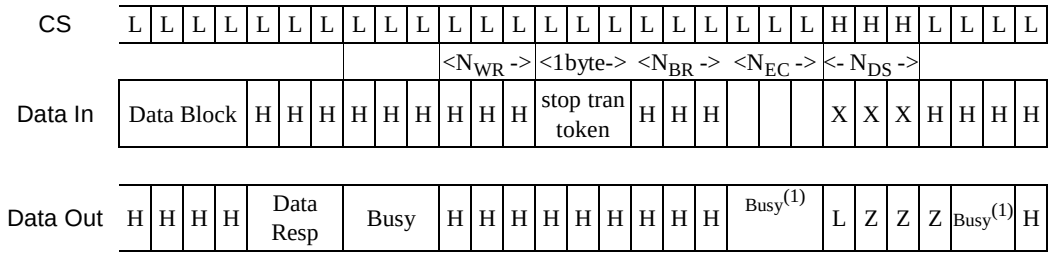


Figure 57: Write operation - bus timing

The following figure describes stop transmission operation in Multiple Block Write transfer.



(1) The Busy may appear within N_{BR} clocks after Stop Tran Token. If there is no Busy the host may continue to the next command.

Figure 58: Stop Transmission in Write Multiple Block

7.5.4 Timing Values

	Min	Max	Unit
N_{CS}	0	-	8 clock cycles
N_{CR}	1	8	8 clock cycles
N_{RC}	1	-	8 clock cycles
N_{AC}	1	spec. in the CSD	8 clock cycles
N_{WR}	1	-	8 clock cycles
N_{EC}	0	-	8 clock cycles
N_{DS}	0	-	8 clock cycles
N_{BR}	0	1	8 clock cycles
Ncx	0	8	8 clock cycles

Table 60: Timing values

7.6 SPI Electrical Interface

Identical to SD mode with the exception of the programmable card output drivers option which is not supported in SPI mode.

7.7 SPI Bus Operating Conditions

Identical to SD mode

7.8 Bus Timing

Identical to SD mode. The timing of the CS signal is the same as any other card input.

8 SD Memory Card mechanical specification

This chapter describes the mechanical and electromechanical features of the SD Memory Card, and furthermore the minimal recommendations to the SD Memory Card connector. All technical drafts follow DIN ISO standard.

The functions of the card package are:

- protecting the chip
- easy handling for the end user
- reliable electrical interconnection
- reliable write protect/card detection capability
- bearing textual information and image
- appealing appearance

The functions of the connector are:

- attaching and fixing the card
- electrical interconnecting the card to the system board
- write protect/card detect indication
- optional: switch on/off power supply
- protection against card inverse insertion

8.1 Card package

Every card package shall have the characteristics described in the following sections.

8.1.1 External signal contacts (ESC)

Number of ESC	9
distance from front edge	1.2 mm
ESC grid	2.5mm
contact dimensions	1.7mm x 4.0mm
electrical resistance	30 m? (worst case: 100 m?)
micro interrupts	< 0.1 ?s

Table 61: SD Memory Card Package - External Signal Contacts

8.1.2 Design and format

Dimensions SD Memory Card package	24mm x 32mm; (min. 23.9mm x 31.9mm; max.24.1mm x 32.1mm) other dimensions Figure 59 testing according to MIL STD 883, Meth 2016
thickness	'Inter Connect Area': 2.1mm +/- 0.15mm or 1.4mm+/-0.15mm for Thin SD Card. 'Substrate Area': Max 2.25mm or Max 1.55 for Thin SD Card - see Figure 61.
label or printable area	In 'Substrate Area' only - see Figure 61.
surface	plain (except contact area)
edges	smooth edges, see Figure 60, Figure 61
inverse insertion	protection on left corner (top view) see Figure 63
position of ESC contacts	along middle of shorter edge

Table 62: SD Memory Card Package - Dimensions

8.1.3 Reliability and durability

temperature	operation: -25°C / 85°C (Target spec) storage: -40°C (168h) / 85°C (500h) junction temperature: max. 95°C
moisture and corrosion	operation: 25°C / 95% rel. humidity storage: 40°C / 93% rel. hum./500h salt water spray: 3% NaCl/35C; 24h acc. MIL STD Method 1009
durability	10.000 mating cycles; Test procedure: tbd.
bending (note 1)	10N
torque (note 1)	0.15N.m or +/-2.5 deg .
drop test	1.5m free fall
UV light exposure	UV: 254nm, 15Ws/cm² according to ISO 7816-1
visual inspection shape and form (note 1)	no warpage; no mold skin; complete form; no cavities surface smoothness <= -0.1 mm/cm² within contour; no cracks; no pollution (fat, oil dust, etc.)
Minimum moving force of WP switch	40gf (Ensures that the WP switch will not slide while it is inserted to the connector).
WP Switch cycles	minimum 1000 Cycles (@ Slide force 0.4N to 5N)

Note (1): The SDA's recommended test methods for Torque, bending and Warpage are defined in seperate Application Notes document.

Table 63: Reliability and Durability

8.1.4 Electrical Static Discharge (ESD) Requirements

ESD testing should be conducted according to IEC61000-4-2

Required ESD parameters are:

(1) Human body model +- 4 KV 100 pf / 1.5 Kohm

(2) Machine model +- 0.25 KV 200 pf / 0 ohm

Contact Pads:

+/- 4kV, Human body model according to IEC61000-4-2

Non Contact Pads area:

+/-8kV (coupling plane discharge)
+/-15kV (air discharge)
Human body model according to IEC61000-4-2

The SDA's recommended test methods for the non-contact/air discharge tests are given in a separate Application Note document.

8.1.5 Quality assurance

The product traceability shall be ensured by an individual card identification number.

8.2 Mechanical form factor

The following 3 technical drawings define the card package of SD Memory Card with 2.1+/-0.15mm card thickness (the Thin SD Memory Card drawings are given in Chapter 8.4).

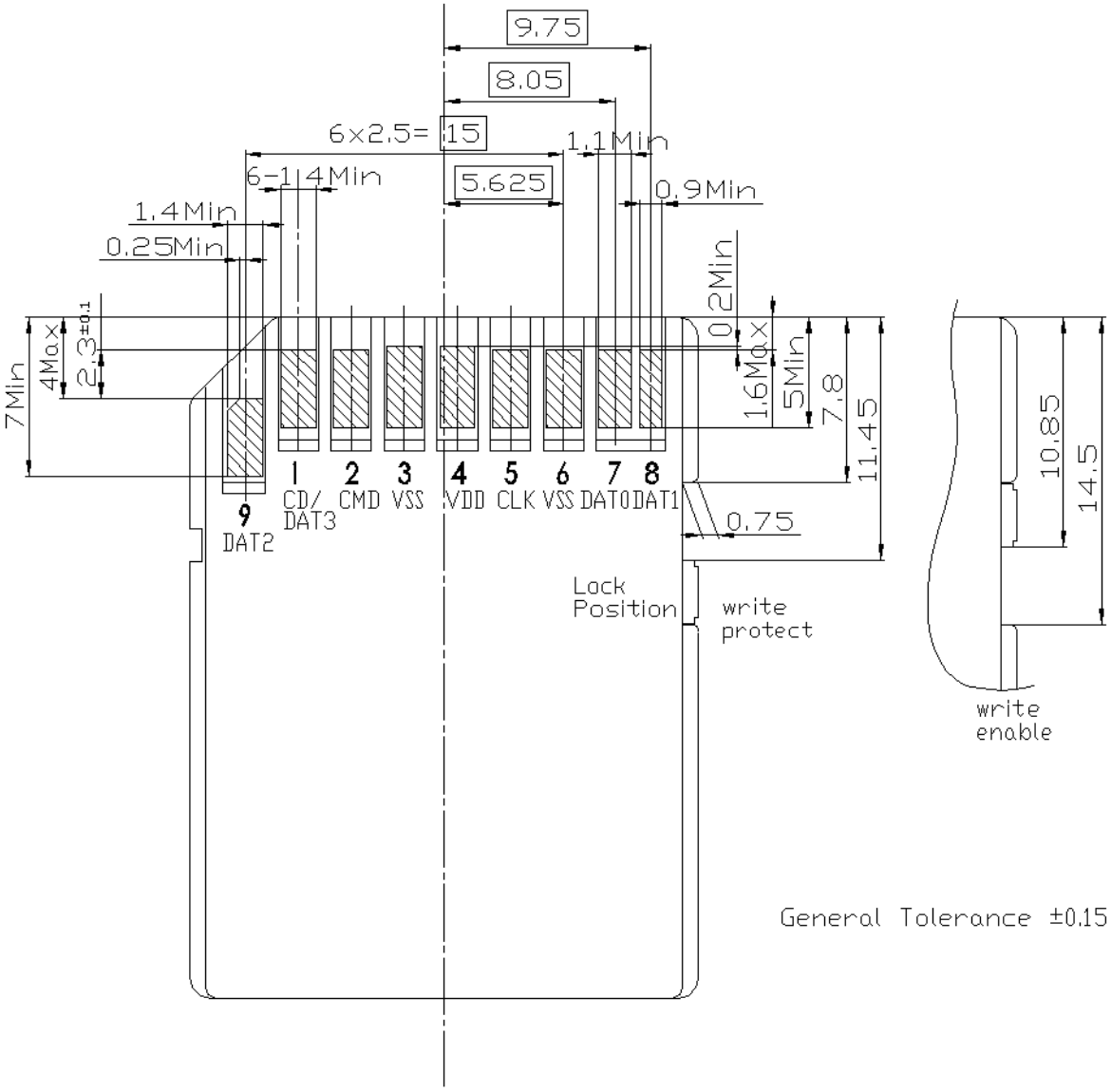


Figure 59: SD Memory Card - Mechanical Description (1 out of 3)

Notes for all the mechanical descriptions of the SD Memory Card (including Thin SD Memory Card):

1) The numbers enclosed by a square means that it is the distance between base lines. Those values are for information only (the given general tolerances are not related to them).

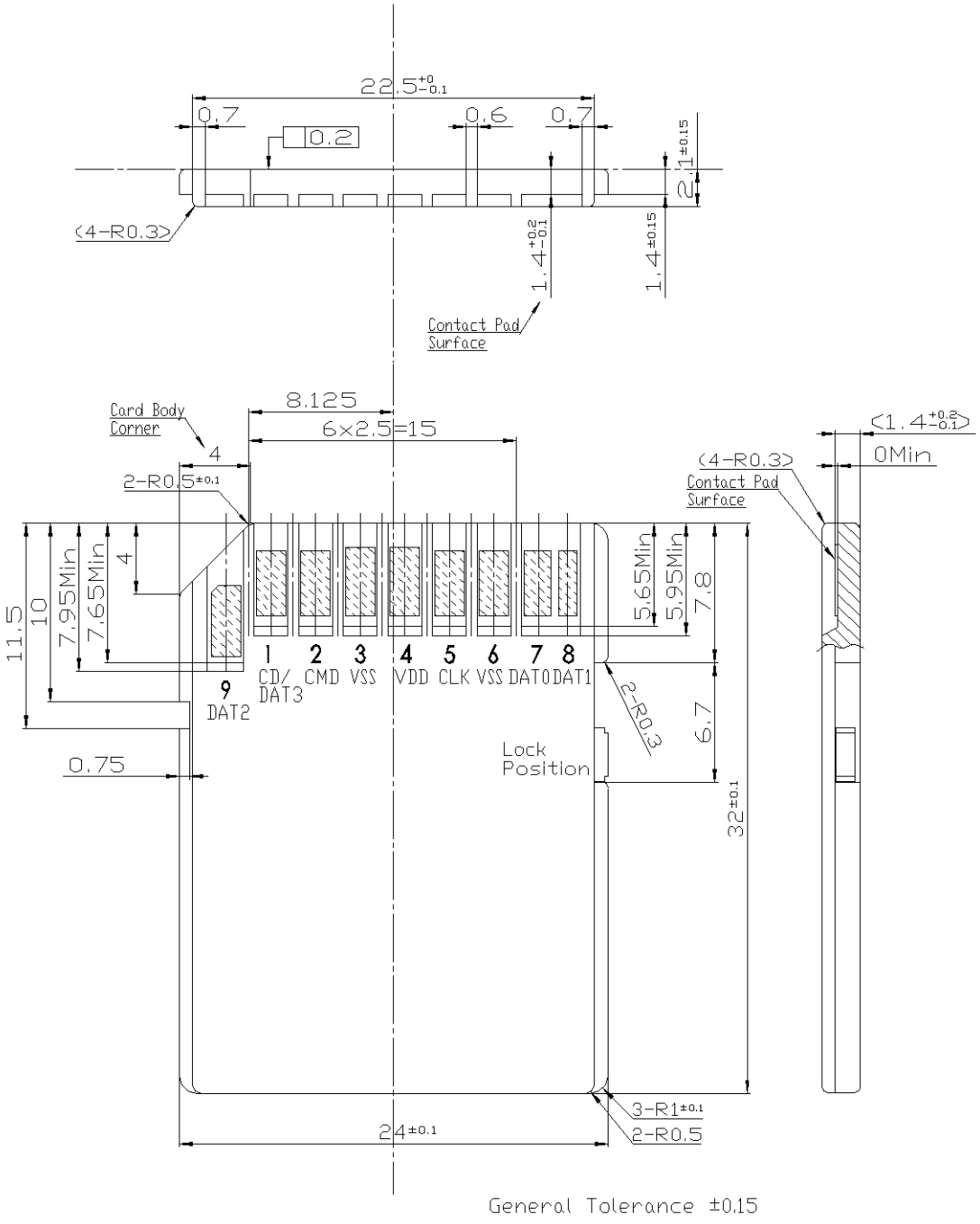
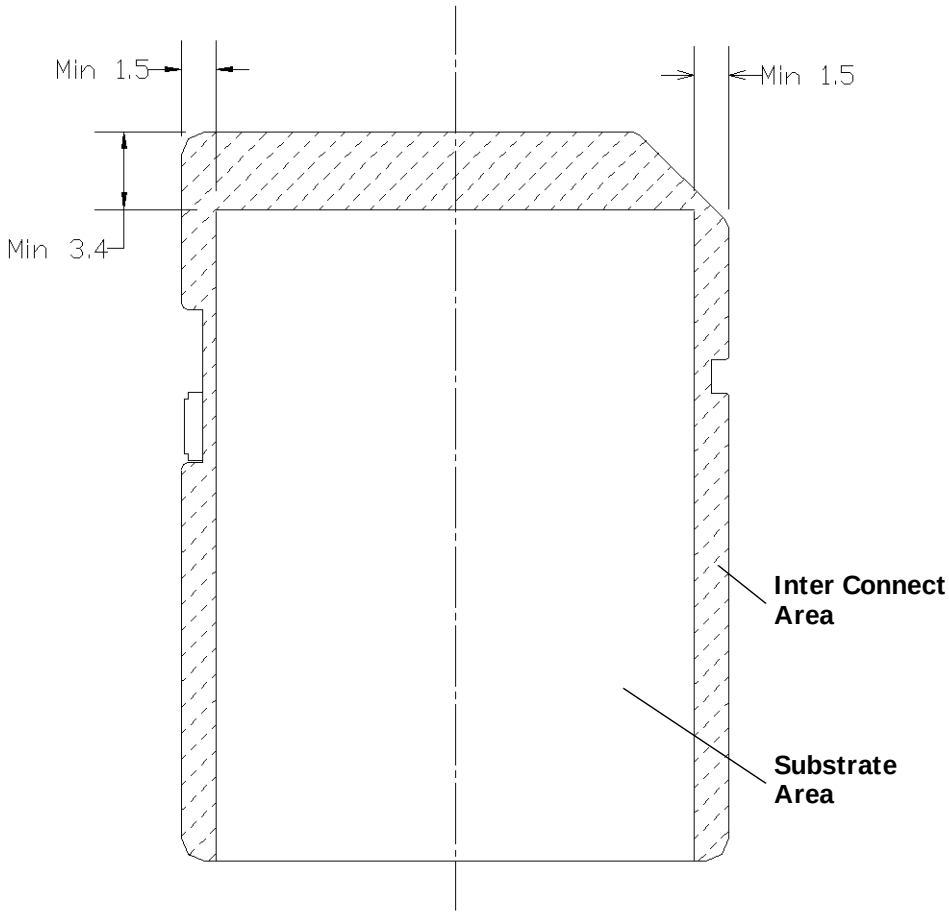


Figure 60: SD Memory Card - Mechanical Description (2 out of 3)



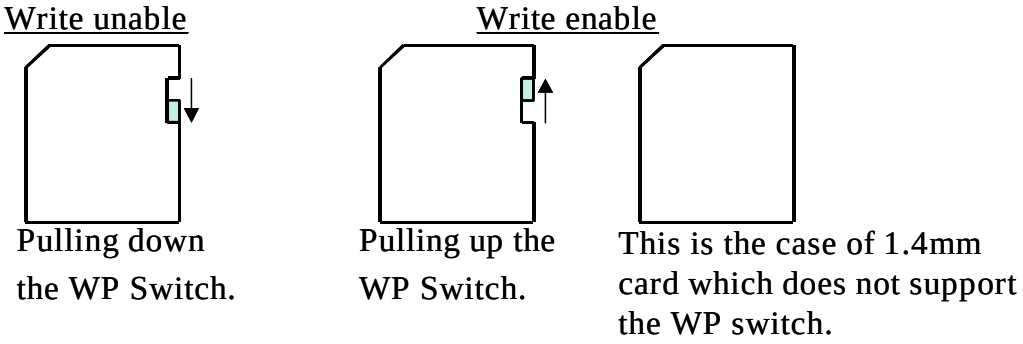
Note: Refer to Table 62 for the definition of 'substrate' and 'Inter Connect' areas.

Figure 61: SD Memory Card - Mechanical Description (3 out of 3)

Figure 62 describes the Write Protect switch position at all various cases and card types.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

- R/W card(WP switch is movable)



- ROM card(WP area is fixed)

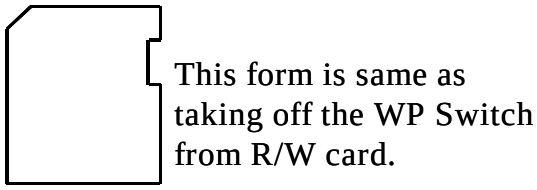


Figure 62: WP Switch definition for all cases and card types

8.3 System: card and connector

The description of the connector is out of the scope of this document. However, minimal recommendations to the connector comprise the ability to guarantee Write Protect and Card Detection, hot insertion and removal of the card, and to prevent inverse insertion.

8.3.1 Card hot insertion

To guarantee a reliable initialization during hot insertion, some measures shall be taken on the host side. For instance, a special hot-insertion capable card connector may be used to guarantee the proper sequence of card pin connection.

The card contacts are contacted in three steps:

1) Ground Vss (pin 3) and supply voltage Vdd (pin 4).
2) CLK, CMD, DAT0, DAT1, DAT2 and Vss (pin 6).
3) CD / DAT3 (pin 1).

Pins 3 and 4 should make first contact when inserting, and release last when extracting.

As another method, a switch could ensure that the power is switched on only after all card pads are contacted. Of course, any other similar mechanism is allowed.

8.3.2 Inverse insertion

Inverse insertion is prevented by the reclining corners of SD Memory Card and connector.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

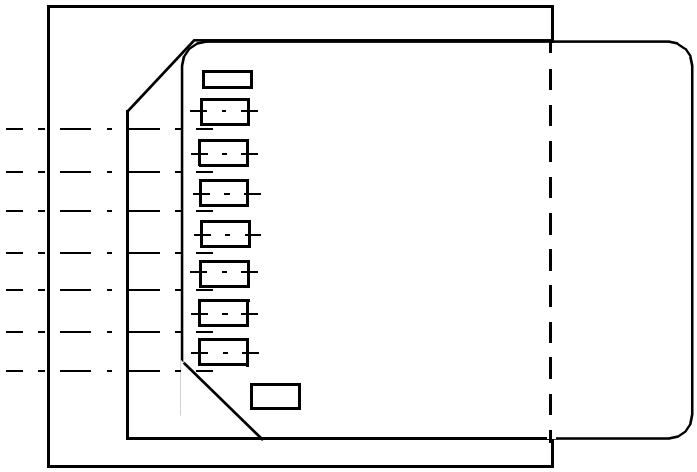


Figure 63: Inverse insertion

8.3.3 Card Orientation

For the benefit of unified terminology when discussing the three dimensional orientation of a card (e.g. for connector definition), the non contact-pads side (the side with the card label) is defined as the TOP side of the card and the contact-pads side of the card is defined as the BOTTOM side of the card.

8.4 Thin (1.4mm) SD Memory Card

SD Memory cards with mechanical dimensions that will be suitable for extra small applications will be available.

The "Thin SD Memory card" has very similar form factor as the SD Memory Card that is given in Chapter 8.2 above except it thickness. The thickness of the "Thin SD Memory Card" is 1.4mm+/-0.15mm.

Figure 64, 65 and 61 are the mechanical drawings of Thin SD Memory Card.

Note that even though the WP switch appears in the given diagram it was defined as an optional for Thin SD Memory Card.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

CONFIDENTIAL

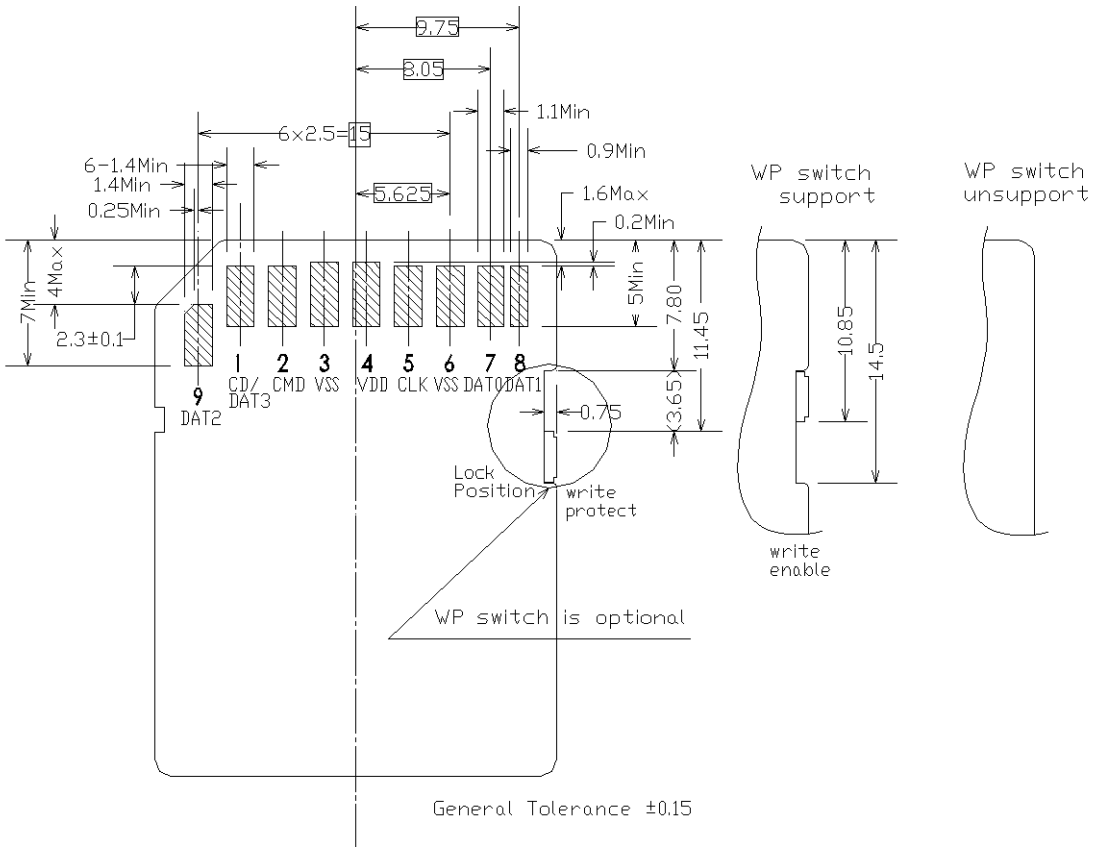


Figure 64: Mechanical Drawing of Thin SD Memory Card

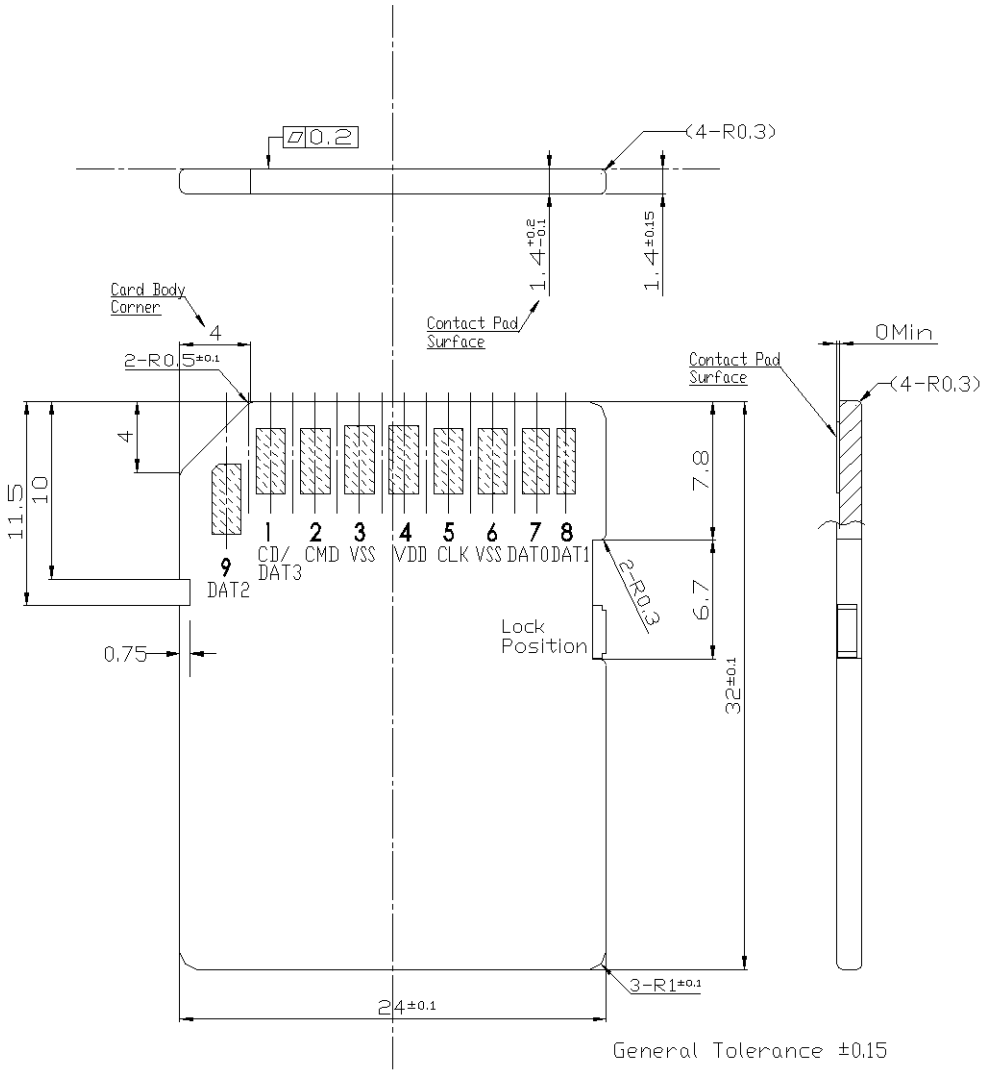


Figure 65: Mechanical Drawing of Thin SD Memory Card

9 Appendix

9.1 Power Supply Decoupling

The V_{SS1}, V_{SS2} and V_{DD} lines supply the card with operating voltage. For this, decoupling capacitors for buffering current peak are used. These capacitors are placed on the bus side corresponding to Figure 66.

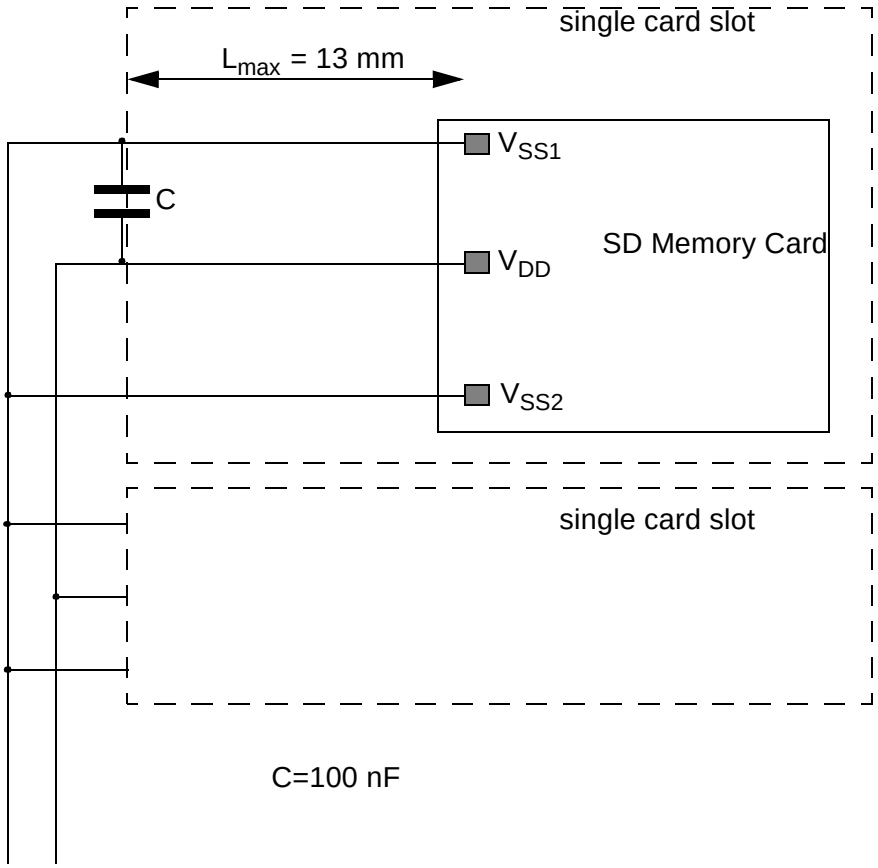


Figure 66: Power supply decoupling

The host controller includes a central buffer capacitor for V_{DD}. Its value is 1 ?F/slot

9.2 Connector

The connector described in this chapter serves as an example and is subject to further changes.

9.2.1 General

The connector housing which accommodates the card is formed of plastic. Inside are 9 contact springs for contacting the pads of the inserted card. As an option a Write Protect/Card Detection

switch shall be part of the housing in order to be able to detect the position of the Write Protect sliding tablet on the card. Testing procedures are performed according to DIN IEC 68.

9.2.2 Card Insertion and Removal

Insertion of the SD Memory Card is only possible with the contact area of the card and the contact area of the connector in the correct position to each other. This is ensured by the reclining corners of the card and the connector, respectively.

To guarantee a reliable initialization during hot insertion, some measures must be taken on the host side. One possible solution is shown in Figure 67. It is based on the idea of a defined sequence for card contact connection during the card insertion process. The card contacts are contacted in three steps:

1) Ground Vss (pin 3) and supply voltage Vdd (pin 4).
2) CLK, CMD, DAT0, DAT1, DAT2 and Vss (pin 6).
3) CD / DAT3 (pin 1).

The pins 3 and 4 should make first contact when inserting and release last when extracting.

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

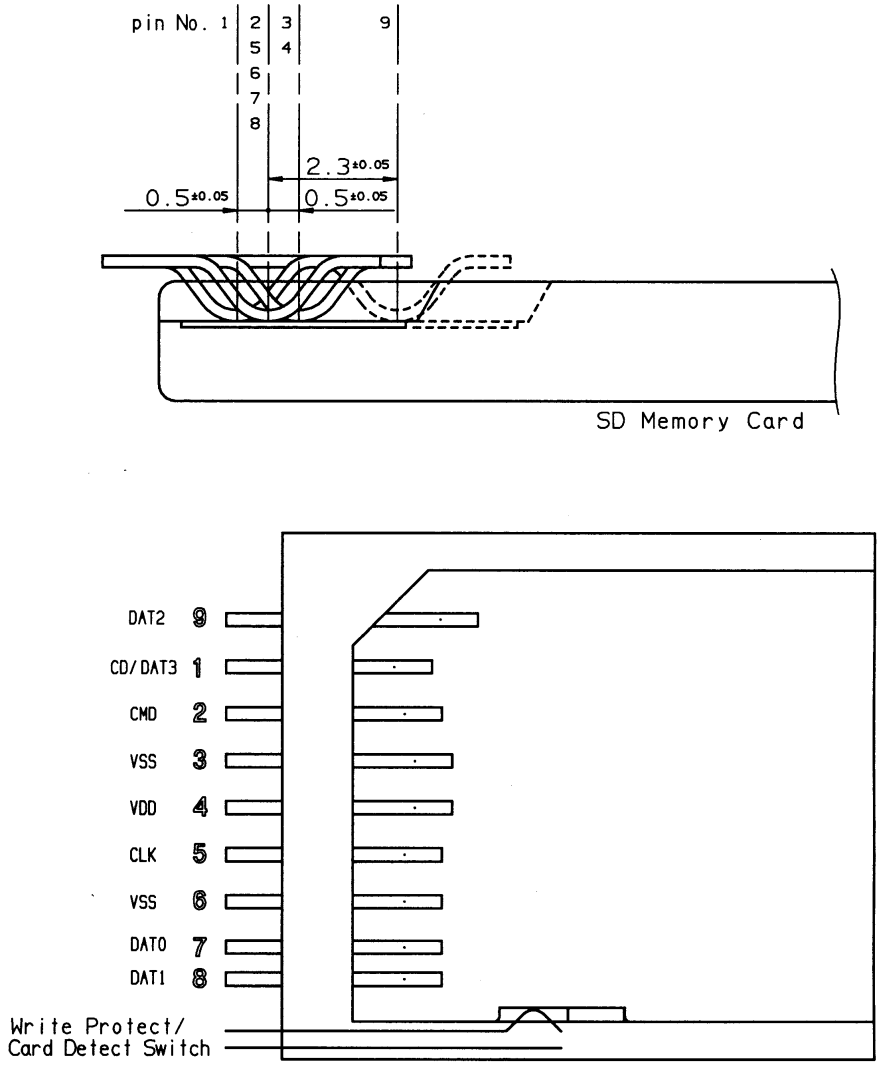


Figure 67: Modified SD Memory Card connector for hot insertion

9.2.3 Characteristics

The features described in the following must be considered when designing a SD Memory Card connector. The given values are typical examples.

- **Mechanical Characteristics**

  - Max. number of mating operations > 10000
  - Contact force 0.2 N (minimum contact force per one contact)

- Total pulling force	min.1 N	DIN IEC 512 part 7
- Total insertion force	max. 40 N	DIN IEC 512 part 7
- Vibration and High Frequency		
- Mechanical frequency range	10.....2000 Hz	DIN IEC 512 part 2 and 4
- Acceleration	2 g	
- Shock:		
- Acceleration	5 g	

- **Electrical Characteristics**

DIN IEC 512

- Contact resistance	100 m?
- Current carrying capacity at 25°C	0.5 A
- Insulation resistance	> 1000 M? ?> 100 M? after test
- Operating voltage	3.3 V
- Testing voltage	500 V
- Operating current	100 mA max.

- **Climatic Characteristics**

DIN IEC 512 part 6-9

- Operating temperature	-25°C.....90°C
- Storage temperature	-40°C.....90°C
- Humidity	95% max. non condensing

- This page was left empty intentionally -

10 Abbreviations and terms

block	a number of bytes, basic data transfer unit
broadcast	a command sent to all cards on the SD bus
CID	Card IDentification number register
CLK	clock signal
CMD	command line or SD bus command (if extended CMDXX)
CRC	Cyclic Redundancy Check
CSD	Card Specific Data register
DAT	data line
DSR	Driver Stage Register
ECC	Error Correction Code
Flash	a type of multiple time programmable non volatile memory
group	a number of sectors, composite erase and write protect unit
LOW, HIGH	binary interface states with defined assignment to a voltage level
NSAC	defines the worst case for the clock rate dependent factor of the data access time
MSB, LSB	the Most Significant Bit or Least Significant Bit
MTP	Multiple Time Programmable memory
OCR	Operation Conditions Register
open-drain	a logical interface operation mode. An external resistor or current source is used to pull the interface level to HIGH, the internal transistor pushes it to LOW
OTP	One Time Programmable memory
payload	net data
push-pull	a logical interface operation mode, a complementary pair of transistors is used to push the interface level to HIGH or LOW
RCA	Relative Card Address register
ROM	Read Only Memory
sector	a number of blocks, basic erase unit
stuff bit	filling bits to ensure fixed length frames for commands and responses
SPI	Serial Peripheral Interface
TAAC	defines the time dependent factor of the data access time
tag	marker used to select groups or sector to erase
TBD	To Be Determined (in the future)
three-state driver	a driver stage which has three output driver states: HIGH, LOW and high impedance (which means that the interface does not have any influence on the interface level)
token	code word representing a command
V_{DD}	+ power supply
V_{SS}	power supply ground

(c)2000 (c)2001 by SD Group (MEI,SanDisk,Toshiba)

SD Memory Card Specifications

Part 2
FILE SYSTEM SPECIFICATION

Version 1.01

April 15 2001

SD Group

Matsushita Electric Industrial Co., Ltd. (MEI)

SanDisk Corporation

Toshiba Corporation

CONFIDENTIAL

Copyright © 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

Conditions for publication:

- **Publisher and Copyright Holder:**
  SD Group (MEI, SanDisk, Toshiba)

- **Exemption:**
  None will be liable for any damage from use of this document.

Revision History

Date	Version	Changes compared to previous issue
March 22th, 2000	1.0	Base Version
April 15th, 2001	1.01	- One notation is added - The description of System ID field is modified - The description of Reserved field in the Extended FDC Descriptor is modified - The description of Reserved field in the Directory Entry is modified - Some computations are added in Annex B - Typo fixes and some clarification notes

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

This page is intentionally left blank.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

INDEX

Part2 FILE SYSTEM SPECIFICATION

1. General FS-1
1.1 Scope 1
1.2 Normative References 2
1.3 Definitions 3
1.3.1 byte 3
1.3.2 defective sector 3
1.3.3 descriptor 3
1.3.4 file 3
1.3.5 sector / block 3
1.3.6 partition 3
1.3.7 user 3
1.3.8 volume 3
1.4 Notations 4
1.4.1 Numerical notation 4
1.4.2 Arithmetic notation 4
1.4.3 Character strings 4
1.4.4 List of acronyms 4
1.5 Data types 5
1.5.1 Numerical values in one-byte fields 5
1.5.2 Numerical values in two-byte fields 5
1.5.3 Numerical values in four-byte fields 5
1.5.4 Pairs of 12-bit integers 5

2. Volume Structure FS-7
2.1 Arrangement of the Data Area 8
2.1.1 Physical Address 8
2.1.2 Physical Sector Number 8
2.1.3 Logical Sector Number 8
2.1.4 Partition Area and Regular Area 8
2.2 Arrangement of the User Area 9
2.2.1 Clusters 9
2.2.2 Status of clusters 9
2.3 Arrangement of the Partition Area 10
2.4 Arrangement of the System Area 12
2.4.1 System Area 12
2.4.2 Partition Boot Sector 12
2.4.3 File Allocation Table (FAT) 12
2.4.4 Root Directory 12

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

3. File Structure FS-13
3.1 Partition Boot Sector 13
3.2 File Allocation Table 17
3.3 File directories 18
3.3.1 Characteristics 18
3.3.2 Directory entry types 18
3.3.3 General definition of Directory entry fields 18
3.4 User Area 20

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

Annex

Annex FS-21
Annex A: File System Layout 21
Annex B: CHS Recommendation 22
Annex C: Sectors per Cluster and Boundary Unit Recommendation for Data Area 23
Annex D: Format Parameter Computations 25

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

List of Figures

Figure 2-1 : Example of Volume Structure for Data Area FS-7
Figure A-1 : Example of File System Layout 21

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

List of Tables

Table 2.3-1 : Master Boot Record and Partition Table FS-10
Table 2.3-2 : Partition Table 10
Table 3.1-1 : FDC Descriptor 13
Table 3.1-2 : Extended FDC Descriptor 13
Table 3.2-1 : FAT Entry Value 17
Table 3.3.3-1 : Directory Entry Field 18
Table B-1 : CHS Recommendation 22
Table B-2 : Example of Partition Table 22
Table C-1 : Sectors per Cluster and Boundary Unit Recommendation (Data Area) 23
Table C-2 : Maximum Data Area size and format parameters 23
Table D-1 : Extended FDC Descriptor (Regular Area) 26

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

This page is intentionally left blank.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

1. General

1.1 Scope

This part specifies the volume structure and file structure of the SD Memory Card (Secure Digital Memory Card). The SD Memory Card file system that defines the logical structure uses FAT file system based on ISO/IEC 9293 standard.

It also supports two areas: one for the Data Area that user can access without mutual authentication and one for the Protected Area that user can access after mutual authentication.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

1.2 Normative References

1) ISO/IEC646:1991

Information technology - ISO 7-bit code character set for information interchange

2) ISO/IEC9293:1994

Information technology - Volume and file structure of disk cartridges for information interchange

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

1.3 Definitions

1.3.1 byte

A string of binary digits operated upon as a unit.

1.3.2 defective sector

A sector that cannot be read or written.

1.3.3 descriptor

A recorded structure containing information about the volume or a file.

1.3.4 file

A named collection of information.

1.3.5 sector / block

A unit of data that can be accessed independently of other units on the SD Memory Card.

1.3.6 partition

An extent of sectors within a volume.

1.3.7 user

A person or other entity that causes the invocation of the services provided by an implementation.

1.3.8 volume

A sector address space as specified in the relevant standard for recording.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

1.4 Notations

1.4.1 Numerical notation

Numbers in decimal notation are represented by decimal digits, namely 0 to 9.

Numbers in hexadecimal notation are represented as a sequence of one or more hexadecimal digits namely 0 to 9 and A to F, prefixed by "0x".

ZERO represents a single bit with the value 0.

1.4.2 Arithmetic notation

The notation *ip(x)* shall mean the integer part of *x*.

The notation *ceil(x)* shall mean the minimum integer that is greater than *x*.

The notation *rem(x,y)* shall mean the reminder of the integer division of *x* by *y*.

1.4.3 Character strings

A value for a sequence of bytes may be specified by a quoted sequence of characters, encoded according to the ISO/IEC 646 standard.

1.4.4 List of acronyms

BP : Byte Position within a certain field, starting with 0 from the first byte of the field.
FAT : File Allocation Table.
FDC : Flexible Disk Cartridge.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

1.5 Data types

1.5.1 Numerical values in one-byte fields

A numerical value in a one-byte field shall be recorded as an 8-bit number in one-byte field.

1.5.2 Numerical values in two-byte fields

A numerical value in a two-byte field shall be recorded in the little endian representation. It shall be recorded according to ISO/IEC 9293.

1.5.3 Numerical values in four-byte fields

A numerical value in a four-byte field shall be recorded in the little endian representation. It shall be recorded according to ISO/IEC 9293.

1.5.4 Pairs of 12-bit integers

A pair of 12-bit numbers shall be recorded in three-byte field according to ISO/IEC 9293.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

This page is intentionally left blank.

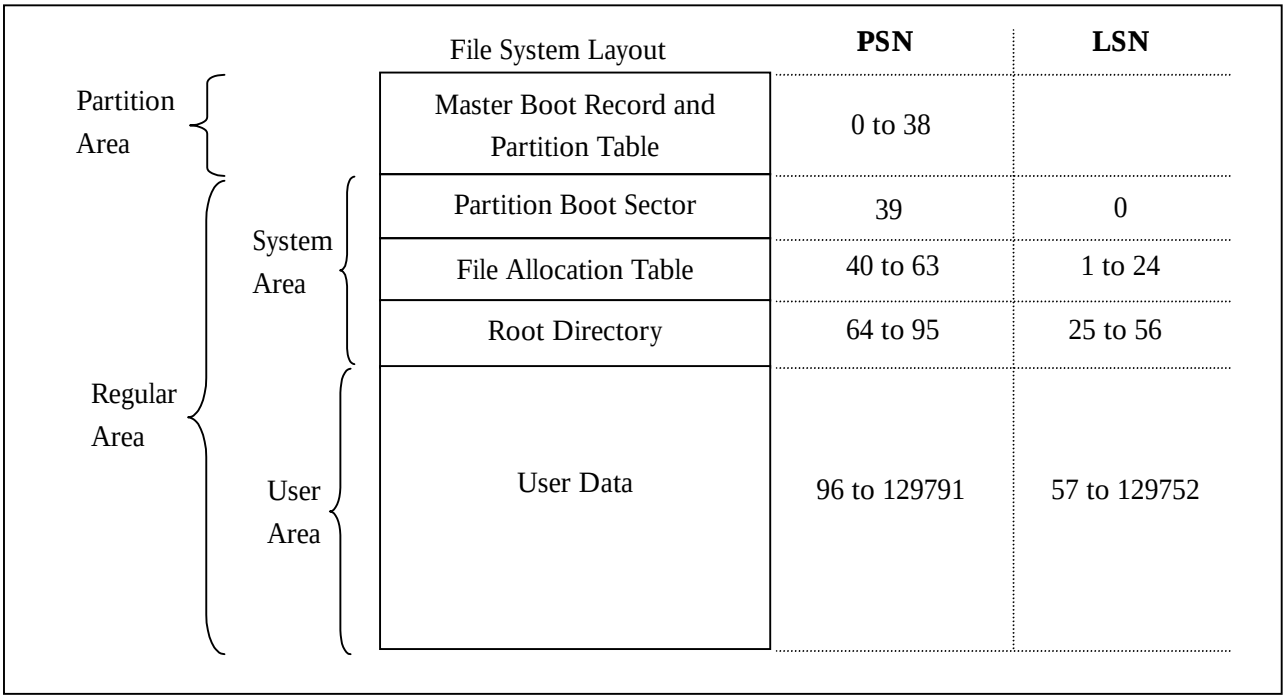
© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

2. Volume Structure

The volume structure of the SD Memory Card is specified in this section. It defines the logical structure of the Data Area. For the identification of the Data Area as a partition, the first sector has Master Boot Record and Partition Table. And the SD Memory Card file system uses the FAT file system (ISO/IEC 9293) and supports both FAT12 and FAT16 as the file system type.

◊ **Figure 2-1 : Example of Volume Structure for Data Area**



PSN : Physical Sector Number

LSN : Logical Sector Number

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

2.1 Arrangement of the Data Area

2.1.1 Physical Address

Each sector shall be identified by a Physical Address comprising the parameters of SD Memory Card's own.

2.1.2 Physical Sector Number

Each sector on a volume shall be identified by a Physical Sector Number. There shall be a one-to-one correspondence between Physical Address and Physical Sector Number. The Physical Sector Numbers shall be assigned in an ascending sequence, beginning with 0.

2.1.3 Logical Sector Number

Each sector on a partition shall be identified by a Logical Sector Number. The first sector of the partition shall be assigned 0 as Logical Sector Number. There shall be a one-to-one correspondence between Physical Sector Number.

2.1.4 Partition Area and Regular Area

The space on Data Area shall be divided into two parts: Partition Area and Regular Area. And the Regular Area shall be divided into System Area and User Area.
The Partition Area shall occupy sectors with the Physical Sector Numbers 0 to *NOM-1*, where *NOM* is the number of sectors in the Master Boot Record and Partition Table.
The Regular Area is a partition of the volume, and divided into System Area and User Area.
The System Area shall occupy sectors with the Physical Sector Numbers *NOM* to *NOM+SSA*-1, where *SSA* is the number of sectors in the System Area. The System Area shall contain Descriptors that specify the recording format of the Regular Area. No part of any file shall be contained in the System Area.
The User Area shall occupy sectors with the Physical Sector Numbers starting with *NOM+SSA*. The User Area shall contain files and directories, and be recorded user data.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

2.2 Arrangement of the User Area

2.2.1 Clusters

The User Area shall be organized into units of allocation called clusters. Each cluster shall consist of the same number of sectors. Each cluster shall be identified by a unique Cluster Number. Cluster Numbers shall be assigned integer number starting with 2.

2.2.2 Status of clusters

A status shall be assigned to each cluster, and shall be one of the following:

- allocated to a file
  The cluster is already allocated.
- available for allocation
  The cluster is prepared for allocate.
- defective
  The cluster is defective. This cluster cannot be allocated.

The status of each cluster shall be identified according to ISO/IEC 9293.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

2.3 Arrangement of the Partition Area

The first sector of the Data Area has a Master Boot Record that includes executable codes and Partition Table that includes the information to identify the partition.

◊ **Table 2.3-1 Master Boot Record and Partition Table**

BP	Length	Field Name	Contents
0	446	Master Boot Record	Not Restricted
446	16	Partition Table (partition1)	Refer to Table2.3-2
462	16	Partition Table (partition2)	All 0x00
478	16	Partition Table (partition3)	All 0x00
494	16	Partition Table (partition4)	All 0x00
510	2	Signature Word	0x55, 0xAA

(BP 0 to 445) Master Boot Record

The content of this field is not specified by this specification.

(BP 446 to 461) Partition Table (partition1)

This field shall specify the information of first partition in the volume. This partition means Regular Area that user can access without mutual authentication. It shall be recorded according to Table 2.3-2.

(BP 462 to 477) Partition Table (partition2)

This field shall be recorded as ZEROs, as a volume shall consist of single Regular Area.

(BP 478 to 493) Partition Table (partition3)

This field shall be recorded as ZEROs, as a volume shall consist of single Regular Area.

(BP 494 to 509) Partition Table (partition4)

This field shall be recorded as ZEROs, as a volume shall consist of single Regular Area.

(BP 510 and 511) Signature Word

This field shall be recorded as 0x55 (BP 510) and 0xAA (BP 511).

◊ **Table 2.3-2 Partition Table**

BP	Length	Field Name	Contents
0	1	Boot Indicator	0x00 or 0x80
1	1	Starting Head	Numeric Value
2	2	Starting Sector / Starting Cylinder	Numeric Value
4	1	System ID	0x01 or 0x04 or 0x06
5	1	Ending Head	Numeric Value
6	2	Ending Sector / Ending Cylinder	Numeric Value
8	4	Relative Sector	Numeric Value
12	4	Total Sector	Numeric Value

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

(BP 0) Boot Indicator
This field shall be recorded as 0x80 if SD Memory Card is used for boot. Otherwise, this field shall be recorded as 0x00.

(BP 1) Starting Head
This field shall specify the starting head of the partition.

(BP 2 and 3) Starting Sector / Starting Cylinder
This field shall specify the starting sector and cylinder of the partition. 6 bits (Bit 0 to Bit 5 in BP 2) in this field shall be used for starting sector. 10 bits (Bit 6 and Bit 7 in BP 2, Bit 0 to Bit 7 in BP 3) in this field shall be used for starting cylinder.

(BP 4) System ID
This field shall be determined only by partition length regardless of file system types (FAT12/FAT16). It shall be recorded as 0x01 if the partition size is less than 32680 sectors. And it shall be recorded as 0x04 if the one is less than 65536 sectors. Otherwise, it shall be recorded as 0x06.

(BP 5) Ending Head
This field shall specify the ending head of the partition.

(BP 6 and 7) Ending Sector / Ending Cylinder
This field shall specify the ending sector and cylinder of the partition. 6 bits (Bit 0 to Bit 5 in BP 6) in this field shall be used for ending sector. 10 bits (Bit 6 and Bit 7 in BP 6, Bit 0 to Bit 7 in BP 7) in this field shall be used for ending cylinder.

(BP 8 to 11) Relative Sector
This field shall specify the number of sectors existing before the starting sector of this partition.

(BP 12 to 15) Total Sector
This field shall specify the number of sectors on the partition.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

2.4 Arrangement of the System Area

2.4.1 System Area

The System Area shall contain the Partition Boot Sector, the Root Directory and the File Allocation Table (FAT) recorded twice.

2.4.2 Partition Boot Sector

The first sector of the System Area shall contain the Partition Boot Sector including the FDC Descriptor. The FDC Descriptor shall contain the parameters for the partition.

2.4.3 File Allocation Table (FAT)

The FAT shall contain a Format Identifier and some entries, each of which indicates cluster of the User Area. These entries shall be numbered consecutively starting with 2 and the Entry Number shall be equal to the Cluster Number of the corresponding cluster.
Each entry in the FAT shall indicate the status of the corresponding cluster. The FAT entries shall be used to identify the set of clusters that are allocated to each file.

2.4.4 Root Directory

The Root Directory shall be recorded in the System Area following the second occurrence of the FAT.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

3. File Structure

3.1 Partition Boot Sector

There is a Partition Boot Sector at the head of a partition and it contains an FDC Descriptor or an Extended FDC Descriptor. The FDC Descriptor and the Extended FDC Descriptor are compliant to ISO/IEC 9293. The Extended FDC is used for the default.

◊ **Table 3.1-1 FDC Descriptor**

BP	Length	Field Name	Contents
0	3	Jump Command	bytes
3	8	Creating System Identifier	a-characters
11	2	Sector Size	Numeric Value
13	1	Sectors per Cluster	Numeric Value
14	2	Reserved Sector Count	Numeric Value
16	1	Number of FATs	Numeric Value
17	2	Number of Root-directory Entries	Numeric Value
19	2	Total Sectors	Numeric Value
21	1	Medium Identifier	0xF8
22	2	Sectors per FAT	Numeric Value
24	2	Sectors per Track	Numeric Value
26	2	Number of Sides	Numeric Value
28	2	(Reserved for future standardization)	0x0000
30	480	(Reserved for system use)	Not Restricted
510	2	Signature Word	0x55, 0xAA

◊ **Table 3.1-2 Extended FDC Descriptor**

BP	Length	Field Name	Contents
0	3	Jump Command	bytes
3	8	Creating System Identifier	a-characters
11	2	Sector Size	Numeric Value
13	1	Sectors per Cluster	Numeric Value
14	2	Reserved Sector Count	Numeric Value
16	1	Number of FATs	Numeric Value
17	2	Number of Root-directory Entries	Numeric Value
19	2	Total Sectors	Numeric Value
21	1	Medium Identifier	0xF8
22	2	Sectors per FAT	Numeric Value
24	2	Sectors per Track	Numeric Value
26	2	Number of Sides	Numeric Value
28	4	Number of Hidden Sectors	Numeric Value
32	4	Total Sectors	Numeric Value
36	1	Physical Disk Number	0x80
37	1	Reserved	0x00
38	1	Extended Boot Record Signature	0x29
39	4	Volume ID Number	Numeric Value
43	11	Volume Label	d-characters
54	8	File System Type	d-characters
62	448	(Reserved for system use)	Not Restricted
510	2	Signature Word	0x55, 0xAA

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

(BP 0 to 2) Jump Command
This field shall specify the jump command to the boot program. It shall be recorded as 0xEB (BP 0), 0xXX (BP 1) and 0x90 (BP 2), or 0xE9 (BP 0), 0xXX (BP 1) and 0xXX (BP 2). 0xXX means that the value is not specified in this specification.

(BP 3 to 10) Creating System Identifier
This field shall specify identification for the system. This field shall be recorded using a-characters and according to ISO/IEC 9293 9.

(BP 11 and 12) Sector Size
This field shall specify the size of a sector in bytes. It shall be recorded as the number 512.

(BP 13) Sectors per Cluster
This field shall specify the number of sectors per cluster. It shall be recorded the following number: 1, 2, 4, 8, 16, 32 or 64. The Cluster Size shall be the multiple size of the erase block size determined by the physical layer. If the erase block size is larger than 32KB, the Cluster Size shall be 32KB and this field shall be recorded 64.

(BP 14 and 15) Reserved Sector Count
This field shall specify the number of sectors reserved for system use. It shall be recorded as the number 1.

(BP 16) Number of FATs
This field shall specify the number of FATs. It shall be recorded as the number 2.

(BP 17 and 18) Number of Root-directory Entries
This field shall specify the number of entries in the Root Directory. It shall be recorded as the number 512.

(BP 19 and 20) Total Sectors
This field shall specify the number of sectors on the partition. It shall be recorded according to ISO/IEC 9293 9.

(BP 21) Medium Identifier
This field shall be recorded as 0xF8 for this specification.

(BP 22 and 23) Sectors per FAT
This field shall specify the number of sectors that shall be occupied by each FAT. It shall be recorded according to ISO/IEC 9293 9.

(BP 24 and 25) Sectors per Track
This field shall specify the number of sectors in each track. This parameter depends on the SD Memory Card's parameter. It shall be recorded according to ISO/IEC 9293 9.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

(BP 26 and 27) Number of Sides
This field shall specify the number of sides that can be recorded. This parameter depends on the SD Memory Card's parameter. It shall be recorded according to ISO/IEC 9293 9.

(BP 28 and 29) Field reserved for future standardization
This field shall be reserved for future standardization. It shall contain only ZEROs.

(BP 30 to 509) Field reserved for system use
This field shall be reserved for system use. It shall be not specified in this specification.

(BP 510 and 511) Signature Word
This field shall be recorded as 0x55 (BP 510) and 0xAA (BP 511).

(Extended FDC Descriptor BP 28 to 31) Number of Hidden Sectors
This field shall specify the number of sectors existing before the starting sector of this partition.

(Extended FDC Descriptor BP 32 to 35) Total Sectors
This field shall specify the number of sectors on the partition if the field in BP 19 and 20 is recorded as ZEROs. It shall be recorded according to ISO/IEC 9293 9.

(Extended FDC Descriptor BP 36) Physical Disk Number
This field shall specify the BIOS physical disk number. This field shall be recorded as 0x80.

(Extended FDC Descriptor BP 37) Reserved
This field shall be reserved for future standardization. It shall be recorded as ZEROs. It shall be recorded according to ISO/IEC 9293 9. However, since a value other than 0x00 may be set on other devices, 0x00 shall not be expected at the time of operation.

(Extended FDC Descriptor BP 38) Extended Boot Record Signature
This field shall be used to identify the descriptor type in the Extended FDC Descriptor when either BP 19 or BP 20 is not recorded as ZEROs. This field shall be recorded as 0x29.

(Extended FDC Descriptor BP 39 to 42) Volume ID Number
This field shall specify the volume identification number. It shall be recorded according to ISO/IEC 9293 9.

(Extended FDC Descriptor BP 43 to 53) Volume Label
This field shall specify the volume label. It shall be recorded according to ISO/IEC 9293 9.

(Extended FDC Descriptor BP 54 to 61) File System Type
This field shall specify the type of the file system. It shall be recorded according to ISO/IEC 9293 9.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

(Extended FDC Descriptor BP 62 to 509) Field reserved for system use
This field shall be reserved for system use. It shall be not specified in this specification.

(Extended FDC Descriptor BP 510 and 511) Signature Word
This field shall be recorded as 0x55 (BP 510) and 0xAA (BP 511).

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

3.2 File Allocation Table

The File Allocation Table supports both the 12-bit FAT and the 16-bit FAT. The FAT structure is compliant to ISO/IEC 9293. The FAT type shall be determined by the number of clusters that depends on the parameter from the physical layer. If the cluster number is less than 4085, FAT12 shall be used. Otherwise, FAT16 shall be used. The first byte of the FAT shall specify the format identifier and be recorded 0xF8. In case of FAT12, the byte 2 and 3 shall be recorded as 0xFF each. In case of FAT16, the byte 2, 3 and 4 shall be recorded as 0xFF each. The sectors of the FAT may include unused area, because the number of clusters shall determine the FAT size in byte. This unused area shall be recorded as ZEROs.

◊ **Table 3.2-1 FAT Entry Value**

FAT Entry Value		Contents
FAT12	FAT16	
000	0000	Indicates that the corresponding cluster is not in use and may be allocated to a file or a directory.
002 to MAX	0002 to MAX	Indicates that the corresponding cluster is already allocated. The value of the entry is the cluster number of the next cluster following this corresponding cluster. Max shall be the Maximum Cluster Number.
MAX+1 to FF6	MAX+1 to FFF6	Shall be reserved for future standardization and shall not be used.
FF7	FFF7	Indicates that the corresponding cluster has a defective cluster.
FF8 to FFF	FFF8 to FFFF	The corresponding cluster is already allocated, and it is the final cluster of the file.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

3.3 File directories

3.3.1 Characteristics

A Directory is a Descriptor that shall contain a set of Directory entries each of which identifies a file, a Volume Label, another Directory or is unused. A Directory can contain the 65536 Directory entries.
The format of the file name for the Directory entries should supports 8.3 format. Although the Long File Name (LFN) can exist in the Directory entries, the SD Memory Card file system may ignore these entries, and refers to only the file name of 8.3 format that is stored with the LFN.
The character code in the Directory entry can be used the code which is permitted by the ISO/IEC 9293.

3.3.2 Directory entry types

Directory entries shall contain descriptive information about the files recorded on the partition. There are some types of these entries as below:

- File Entry

  A File Entry shall specify information of a file.

- Volume Label Entry

  A Volume Label Entry shall specify the volume label of the partition.

- Sub-directory Pointer Entry

  A Sub-directory Pointer Entry shall specify information of a directory.

- Sub-directory Identifier Entry

  A Sub-directory Identifier Entry shall identify a file as a Sub-directory.

- Sub-directory Parent Pointer Entry

  A Sub-directory Parent Pointer Entry shall specify information of its parent directory.

- Not-currently-in-use Entry

  A Not-currently-in-use Entry shall specify the entry is not used and able to allocate.

- Never-used Entry

  A Never-used Entry shall specify the end of the directory. It shall not appear before any other type of Directory entry.

These entries shall be recorded according to ISO/IEC 9293 11.

3.3.3 General definition of Directory entry fields

Table 3.3.3-1 indicates the structure of the Directory entry field.

◊ **Table 3.3.3-1 Directory Entry Field**

BP	Length	Field Name	Contents
0	8	Name	Depends on entry type
8	3	Name Extension	d-characters
11	1	Attributes	8 bits
12	10	Reserved Field	bytes
22	2	Time Recorded	Numeric Value
24	2	Date Recorded	Numeric Value
26	2	Starting Cluster Number	Numeric Value
28	4	File Length	Numeric Value

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

(BP 0 to 7) Name
The content and the description of this field shall depend on the entry type. It shall be recorded according to ISO/IEC 9293 11.

(BP 8 to 10) Name Extension
The content and the description of this field shall depend on the entry type. The content of this field shall be d-characters. It shall be recorded according to ISO/IEC 9293 11.

(BP 11) Attributes
This field shall specify the attributes of the entry. It shall be recorded according to ISO/IEC 9293 11.

(BP 12 to 21) Reserved Field
The content of this field shall depend on the entry type. If this entry is LFN entry, this field shall not be specified in this specification. Otherwise, it shall be recorded as ZEROs. However, since a value other than 0x00 may be set on other devices, 0x00 shall not be expected at the time of operation.

(BP 22 and 23) Time Recorded
This field shall contain a 16-bit integer representing a time. It shall be recorded according to ISO/IEC 9293 11.

(BP 24 and 25) Date Recorded
This field shall contain a 16-bit integer representing a date. It shall be recorded according to ISO/IEC 9293 11.

(BP 26 and 27) Starting Cluster Number
The content of this field shall depend on the entry type. It shall be recorded according to ISO/IEC 9293 11.

(BP 28 to 31) File Length
The content of this field shall depend on the entry type. It shall be recorded according to ISO/IEC 9293 11.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

3.4 User Area

The User Area shall be organized into clusters. Each cluster has a Cluster Number respectively. The first cluster in the User Area is corresponding to Cluster Number 2.

Although it is available to read/write by the sector, it is necessary to transact reading/writing with the unit whose minimum size is the same as that of the recommended reading/writing at the physical layer. Other than that, there are no special restrictions for the SD Memory Card file system.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

Annex

Annex A: File System Layout

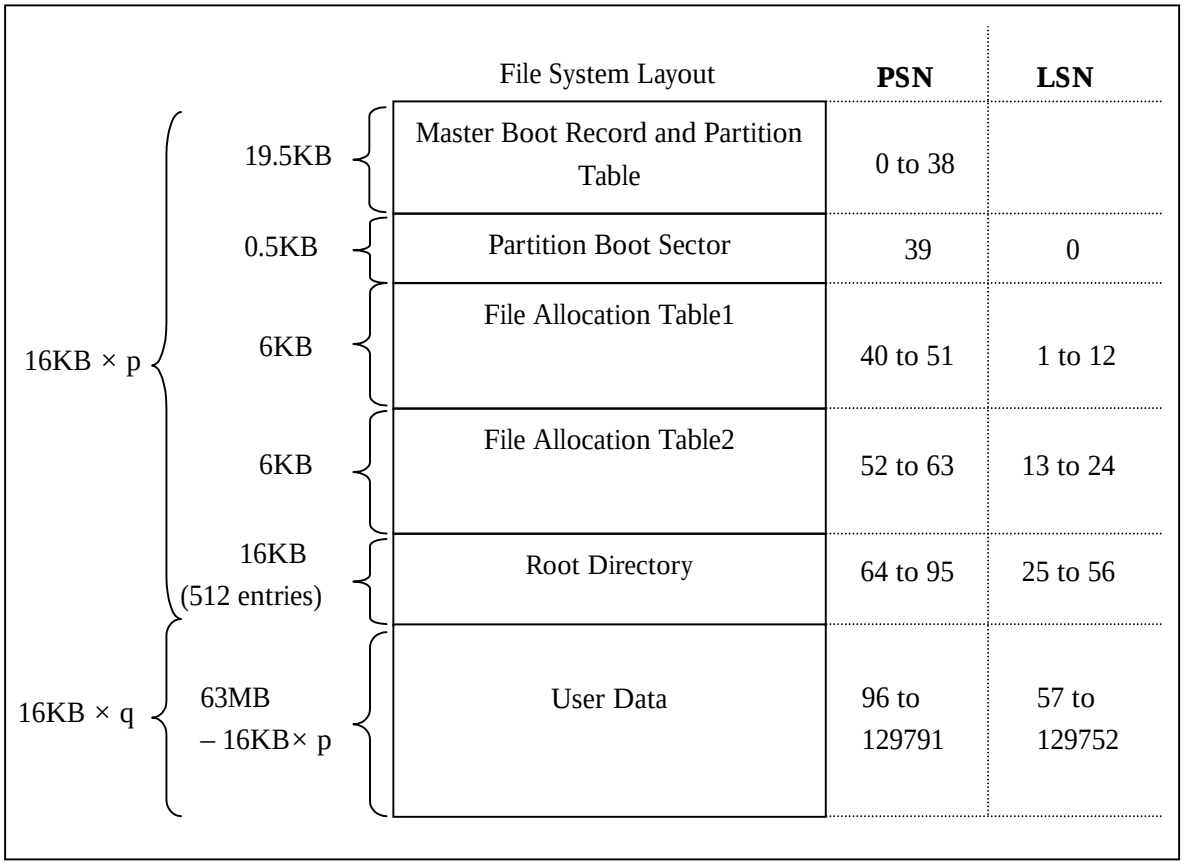
Reading/writing of the User Area should be done with the unit whose minimum size is the same as the recommended size of reading/writing at physical layer. Therefore, Cluster Size shall be determined considering the recommended size. And the head sector of the User Area must be started with the multiple offset of the Cluster Size.

The structure of the file system should be implemented as follows.

1. The combined size of Master Boot Record, Partition Table, Partition Boot Sector, File Allocation Table and Root Directory is a multiple size of the Cluster Size.
2. The number of the sectors before Partition Boot Sector adjusts the above size.
3. Master Boot Record and Partition Boot Sector belong to different cluster.
4. The first sector of the Master Boot Record and the first sector of the User Data are always placed on the cluster boundary.

The following is an example of a partition when the size of the cluster is 16KB.

◊ **Figure A-1 : Example of File System Layout**



p,q : a natural number

PSN : Physical Sector Number

LSN : Logical Sector Number

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

Annex B: CHS Recommendation

The following table shows the recommendation for CHS parameter. In this table, Card Capacity means the total size of Data Area and Protected Area.

◊ **Table B-1 CHS Recommendation**

Card Capacity	Number of Heads	Sectors per Track
~2MB	2	16
~16MB	2	32
~32MB	4	32
~128MB	8	32
~256MB	16	32
~504MB	16	63
~1008MB	32	63
~2016MB	64	63
~2048MB	128	63

Starting Head, Starting Sector, Starting Cylinder, Ending Head, Ending Sector, and Ending Cylinder shall be calculated with the above CHS parameter, Total Sector, and Relative Sector as described below.

$$\text{Starting Head} = ip\{ \frac{rem(\text{Relative Sector}, \text{Number of Heads} \times \text{Sectors per Track})}{\text{Sectors per Track}} \}$$

$$\text{Starting Sector} = rem(\text{Relative Sector}, \text{Sectors per Track}) + 1$$

$$\text{Starting Cylinder} = ip(\frac{\text{Relative Sector}}{\text{Number of Heads} \times \text{Sectors per Track}})$$

$$\text{Ending Head} = ip\{ \frac{rem(\text{Relative Sector} + \text{Total Sector} - 1, \text{Number of Heads} \times \text{Sectors per Track})}{\text{Sectors per Track}} \}$$

$$\text{Ending Sector} = rem(\text{Relative Sector} + \text{Total Sector} - 1, \text{Sectors per Track}) + 1$$

$$\text{Ending Cylinder} = ip(\frac{\text{Relative Sector} + \text{Total Sector} - 1}{\text{Number of Heads} \times \text{Sectors per Track}})$$

The following is an example of a Partition Table when the size of the Data Area 63MB.

◊ **Table B-2 Example of Partition Table**

BP	Length	Field Name	Contents
0	1	Boot Indicator	0x00
1	1	Starting Head	1
2	2	Starting Sector / Starting Cylinder	8 / 0
4	1	System ID	0x06
5	1	Ending Head	7
6	2	Ending Sector / Ending Cylinder	32 / 506
8	4	Relative Sector	39
12	4	Total Sector	129753

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

Annex C: Sectors per Cluster and Boundary Unit Recommendation for Data Area

The following table shows the recommendation for Sectors per Cluster and Boundary Unit of Data Area.

Number of sectors before the starting sector of User Data is multiple size of Boundary Unit. In this table, Card Capacity means the total size of Data Area size and Protected Area size.

◊ **Table C-1 Sectors per Cluster and Boundary Unit Recommendation (Data Area)**

Card Capacity	Sectors per Cluster	Boundary Unit
~8MB	16	16
~64MB	32	32
~256MB	32	64
~1024MB	32	128
~2048MB	64	128

Maximum Data Area size and format parameters are shown in following table.

◊ **Table C-2 Maximum Data Area size and format parameters**

Card Capacity	Sectors per Cluster	Max Data Area size(sector)	an example(values depend on Data Area size)				
			Clusters	FAT Sec	Hidden	FAT bits	User Data Offset
~4MB	16	8032	498	2	27	12	64
~8MB	16	16224	1010	3	25	12	64
~16MB	32	32448	1011	3	57	12	96
~32MB	32	64896	2025	6	51	12	96
~64MB	32	129792	4053	12	39	12	96
~128MB	32	259584	8106	32	95	16	192
~256MB	32	519168	16216	64	95	16	256
~512MB	32	1038336	32432	127	225	16	512
~1024MB	32	2076672	64872	254	227	16	768
~2048MB	64	4153344	64884	254	227	16	768

However,

Card Capacity…SD Card Capacity.

Sectors per Cluster…number of sectors per cluster. This parameter is defined from Card Capacity.

Max Data Area size…maximum number of sectors for Data Area.

Clusters…number of clusters in User Data. This parameter varies with the Data Area size.

FAT Sec…number of sectors per FAT. This parameter varies with the Data Area size.

Hidden…number of sectors existing before Partition Boot Sector. This parameter varies with the Data Area size.

FAT bits…If the area is formatted with FAT12, FAT bits is 12. And If the area is formatted with FAT16, FAT bits is 16. This parameter varies with the Data Area size.

User Data Offset…number of sectors existing before the starting sector of User Data.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

Sectors per Cluster is defined from Card Capacity. Clusters, FAT Sec, Hidden, FAT bits, and User Data Offset vary with the Data Area size (Use the parameters in Table C-1 for calculation).

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

Annex D: Format Parameter Computations

In this section, format parameter computations is proposed. Data Area should be formatted by the following steps.

1. Sectors per Cluster(SC) is determined from the area size.
2. Number of Root-directory Entries(RDE) is 512.
3. Sector Size(SS) is 512.
4. Reserved Sector Count(RSC) is 1.
5. Total Sectors(TS) is the number of all sectors of the area.
6. FAT bits(12[FAT12], 16[FAT16]) is determined by SC and TS.
7. Sectors per FAT(SF) is computed as following:

$$SF = ceil\{ \frac{TS/SC \times FAT\ bits}{SS \times 8} \}$$

8. Number of sectors in the system area(SSA) is computed as following:

$$SSA = RSC + 2 \times SF + ceil(\frac{32 \times RDE}{SS})$$

9. Number of Sectors in Master Boot Record(NOM) is computed as following:

$$NOM + SSA = BU \times n$$

   Here, n means the minimum natural number satisfying above expression. And BU means the Boundary Unit determined by Annex C.

10. If NOM isn't equal to BU, NOM is added BU.
11. Maximum Cluster Number(MAX) is computed as following:

$$MAX = ip(\frac{TS - NOM - SSA}{SC}) + 1$$

12. Sectors per FAT(SF') is recalculated as following:

$$SF' = ceil\{ \frac{[2 + (MAX - 1)] \times FAT\ bits}{SS \times 8} \}$$

    In this formula, 'MAX-1' means the number of clusters. And '2+(MAX-1)' means the number of FAT entries including two signature entries.

13. If SF' isn't equal to SF, SF' is used as SF. And recalculate from step 8.
14. If SF' is equal to SF, parameter computing is complete.

Example of a SD Memory Card including 63MB Data Area:

- TS=129792 Sectors

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

- SC=32 Sectors
- RDE=512 Entries
- SS=512 B
- RSC=1 Sectors
- FAT bits=12[FAT12]
- SF=12 Sectors
- SSA=57 Sectors
- NOM= 39 Sectors
- MAX=4054

Example of Extend FDC Descriptor for the Above Example:

◊ **Table D-1 Extended FDC Descriptor (Regular Area)**

BP	Length	Field Name	Contents
0	3	Jump Command	0xEB,0x00,0x90
3	8	Creating System Identifier	“SYSTEMID”
11	2	Sector Size	512
13	1	Sectors per Cluster	32
14	2	Reserved Sector Count	1
16	1	Number of FATs	2
17	2	Number of Root-directory Entries	512
19	2	Total Sectors	0
21	1	Medium Identifier	0xF8
22	2	Sectors per FAT	12
24	2	Sectors per Track	32
26	2	Number of Sides	8
28	4	Number of Hidden Sectors	39
32	4	Total Sectors	129753
36	1	Physical Disk Number	0x80
37	1	Reserved	0x00
38	1	Extended Boot Record Signature	0x29
39	4	Volume ID Number	0x01234567
43	11	Volume Label	“VOLUME1 ”
54	8	File System Type	“FAT12 ”
62	448	(Reserved for system use)	Not Restricted
510	2	Signature Word	0x55AA

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

SD Memory Card Specifications

Part 3
SECURITY SPECIFICATION

Version 1.01

April 15 2001

SD Group

Matsushita Electric Industrial Co., Ltd. (MEI)

SanDisk Corporation

Toshiba Corporation

CONFIDENTIAL

Copyright © 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

Conditions for publication:

- **Publisher and Copyright Holder:**
  SD Group (MEI, SanDisk, Toshiba)

- **Exemption:**
  None will be liable for any damage from use of this document.

Revision History

Date	Version	Changes compared to previous issue
March 22th, 2000	1.0	Base Version
April 15th, 2001	1.01	- The Supplementary Note (May 2000) for "SD Memory Card Specifications Part 3 Security Specification Version 1.0 (March 2000)" was incorporated into the spec. - MKB Supplementary Notes 1 (May 2000) for "SD Memory Card Specifications Part 3 Security Specification Version 1.0 (March 2000)" was incorporated into the spec. - Typo fixes and some clarification notes

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

This page is intentionally left blank

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

Part 3 SECURITY SPECIFICATION

1.General 1
1.1 Scope 1
1.2 References 1
2.Data Element 2
2.1 Media Identifier 2
3. Security Command set for copyright protection 3
3.1 Security Command List 3
3.2 Usage of Security command 11
3.3 SD Memory Card State Diagram on Authentication 12
3.4 Summarization of error responses 13
4. Random Number Generation (RNG) 14
5. File System 16
5.1 General 16
5.2 Master Boot Record and Partition Table 17
5.3 Partition Boot Sector 17
5.4 File Allocation Table 17
5.5 Root Directory 17
5.6 User Data 17
Annex 18
A. Test command Requirement 18
B. Sectors per Cluster and Boundary Unit Recommendation for Protected Area 19
C. Type of 16 MKBs on SD Memory Card 21

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

This page is intentionally left blank

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

1.General

1.1 Scope

The main objectives of the Security specification of SD Memory Card are:

- To protect the copyrighted data recorded on the SD Memory Card from unauthorized use (for reproduction and duplication).
- To give independent protection for different pieces of copyrighted data of different applications (electronic music distribution EMD, electronic books, etc.).

This document and *"Content Protection for Recordable Media Specification SD Memory Card Book"*, that is developed by 4C Entity, LLC (IBM, Intel, MEI, Toshiba), contain the information on the functions required of the SD Memory Card to achieve the above objectives.

This document especially contains the security specification that is depending on the implementation of the SD Memory Card, more concretely,

(A) Data Element (SD Memory Card Specific)
Media Identifier

(B) Security Command set of SD Memory Card

(C) Random Number Generation on SD Memory Card

(D) File system (volume structure) of Protected Area on SD Memory Card

The following technologies are offered by *Content Protection for Recordable Media Specification SD Memory Card*: *Book:*

- Content and key encryption algorithm (C2 encryption),
- Revocation scheme of the unauthorized accessing device (Media Key Block),
- Authentication and Key Exchange mechanism (AKE) between SD Memory Card and the accessing devices.
- Data Structure of Protected Area on SD Memory Card
- File System (directory and file format) of Protected Area on SD Memory Card
- Content Encryption Format

etc.

1.2 References

4C Entity, LLC, *Content Protection for Recordable Media Specification SD Memory Card Book*.
SD Group, SD Memory Card Specifications Part1: Physical Layer Specification
SD Group, SD Memory Card Specifications Part2: File System Specification

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

2.Data Element

This Section describes the SD Memory Card specific 'data element'.

SD Memory Card non-specific data element is described in *Content Protection for Recordable Media Specification SD Memory Card Book*.

2.1 Media Identifier

SD Memory Card shall contain a 64-bit Media Identifier (ID_{media}), a part of which is unique by each SD Memory Card. The Media Identifier logical format is shown in Table 2-1. As shown in Table 2-1, the least significant 56-bit (Byte"1" to Byte"7") of the Media Identifier is a SD Memory Card Specific part.

In Table2-1,

- The 4C Entity, LLC assigns each SD Memory Card Manufacturer a unique 1-byte value as the Manufacturer ID field. (The detail is defined in *Content Protection for Recordable Media Specification SD Memory Card Book*)
- The SD Group assigns each SD Memory Card Manufacturer a unique 2-byte value as the OEM/Application ID value
- Each SD Memory Card Manufacturer assigns a unique 5-byte value as the Serial Number, which consists of 1-byte Product Revision (PRV) value, and 4-byte Product serial number (PSN) value.

◊

◊ **Table 2.1 : Media Identifier for SD Memory Card**

Bit Byte	7	6	5	4	3	2	1	0
0	Manufacturer ID (MID: 1byte) assigned by 4C Entity, LLC							
1	OEM/Application ID (OID: 2byte) assigned by SD Group							
2								
3	Product Revision (PRV: 1byte)							
4	Product serial number (PSN: 4byte)							
5								
6								
7								

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

3. Security Command set for copyright protection

3.1 Security Command List

In order to support a new set of commands that will be 'behind' the MultiMediaCard standard, the new commands will be an Application Specific given commands and shall be preceded with APP_CMD (CMD55).
Note that all the commands do not use RCA (Relative Card Address). Therefore those commands shall be used after the card was selected (in '*tran_state*'). And note that all the following new Security related commands assume fixed block size of 512 Bytes per block. The Sector Size is able to change by SET_BLOCKLEN (CMD16) command, though it is not executable except for 512 Bytes Sector size.

◊ **Table 3.1 : Security Command List**

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
ACMD43	adtc	[31:24]Unit_Count: [23:16] MKB_ID: [15:0]Unit_Offset:	R1	GET_MKB	Reads Media Key Block from the System Area of SD Memory Card. -'Unit_Count' specifies the Number of units to read. (Here, a unit=512 byte (fixed).) - 'MKB_ID' specifies the application unique number. - 'Unit_Offset' specifies the start address(offset) to read. (See Note (13)(14))
ACMD44	adtc	[31:0] stuff bits	R1	GET_MID	Reads Media ID from the System Area of SD Memory Card. (See Note (13))
ACMD45	adtc	[31:0] stuff bits	R1	SET_CER_RN1	AKE Command: Writes random number RN1 as challenge1 in AKE process. (See Note (1)(2))
ACMD46	adtc	[31:0] stuff bits	R1	GET_CER_RN2	AKE Command: Reads random number RN2 as challenge2 in AKE process. (See Note (1))
ACMD47	adtc	[31:0] stuff bits	R1	SET_CER_RES2	AKE Command: Writes RES2 as response2 to RN2 in AKE process. (See Note (1))

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

ACMD48	adtc	[31:0] stuff bits	R1	GET_CER_ RES1	AKE Command: Reads RES1 as response1 to RN1 in AKE process. (See Note (1)(16))
ACMD18	adtc	[31:0] stuff bits	R1	SECURE_RE AD_MULTI_ BLOCK	Protected Area Access Command: Reads continuously transfer data blocks from Protected Area of SD Memory Card. (See Note (7)(14)(17)) The (essential) argument of this command as shown below is transferred securely in AKE command(ACMD45) (See Note (2)). - [31:24] 'Unit_Count' specifies the number of blocks to transfer. Block Size is fixed 512bytes. -[23] 'Reserve'.(This value shall be set to '0' for the future extension.) - [22:0] 'Unit_Address' specifies the start address to read. ([ ] shows bit position of the (essential) argument)

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

ACMD25	adtc	[31:0] stuff bits	R1	SECURE_WRITE_MULTI_BLOCK	Protected Area Access Command: Writes continuously transfer data blocks to Protected Area of SD Memory Card. (See Note (4)(7)(17)) The (essential) argument of this command as shown below is transferred securely in AKE command (ACMD45). (See Note (2)). - [31:24] 'Unit_Count' specifies the number of blocks to transfer. Block Size is fixed 512bytes. -[23] Mode specifies the following: - Mode = 0: This mode shall be used to write a data which should be shared by all applications, such as FAT associated data (e.g. Master boot record, Partition table, File Allocation Table and Root Directory). - Mode = 1: This mode shall be used to write a data which should be protected from other application such as content associated data(e.g. Title Key, CCI). - [22:0] 'Unit_Address' specifies the start address to write. ([ ] shows bit position of the (essential) argument)

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

ACMD38	ac	[31:0] stuff bits	R1b	SECURE_ERASE	Protected Area Access Command: Erase a specified region of the Protected Area of SD Memory Card. (See Note (4)(7)) The (essential) argument of this command as shown below is transferred securely in AKE command (ACMD45) (See Note (2)). - [31:24] 'Unit_Count' specifies the number of blocks to transfer. Block Size is fixed 512bytes. - [23:0] 'Unit_Address' specifies the start address to erase. ([ ] shows bit position of the (essential) argument)

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

ACMD49	ac	[31:0] stuff bits	R1b	CHANGE_SECURE_AREA	Protected Area Access Command: Change size of the Protected Area. (See Note (3)(9)(10)(15)(16)) The value of Unit_address [23:0] (discribed bellow) is given in units of 'MULT*BLOCK_LEN/512'-1. Error occurs in the case that the Protected Area size (in Bytes) is set to a value other than multiplies of MULT*BLOCK_LEN. Note that the minimum User Data Area size is MULT*BLOCK_LEN. In that case Unit_address[23:0]=MULT*BLOCK_LEN/512-1. (About MULT*BLOCK_LEN, see chapter5.3 of SD Memory Card Specifications Part1: Physical Specification). The (essential) argument of this command as shown below is transferred securely in AKE command (ACMD45) (See Note (2)). - [23:0]'Unit_Address' The Protected Area follows the User Data Area in such a way that the first unit address (unit=512 byte) of the Protected Area follows the last unit address of the User Data Area and the highest unit address of the Protected Area is the highest unit address of the memory area. Under those conditions, - [23:0]'Unit_Address' is an address, in units of 512byte, of the end of the User Data Area. - [31:24] stuff bits ([ ] shows bit position of the (essential) argument)

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

ACMD26	adtc	[31:0] stuff bits	R1	SECURE_WRITE_MKB	System Area Access Command: Overwrite the existing Media Key Block (MKB) on the System Area of SD Memory Card with new MKB. (See Note(4)(11)) The (essential) argument of this command as shown below is transferred securely in AKE command (ACMD45) (See Note (2)). -[31:24] 'Unit_Count' specifies the total number of units to be transferred (up to max of 128K bytes). Unit Size is fixed 512bytes. -[23:16] 'MKB_ID' -[15:0] reserved. (This value shall be set to '0' for the future extension.) ([ ] shows bit position of the (essential) argument)

Note:

(1) AKE Commands (ACMD45-48) are always executed in conjunction with either of "Protected Area Access Command" (ACMD18, ACMD25, ACMD26, ACMD38, ACMD49).

(2) In AKE command (ACMD45), challenge1 (random number RN1) is generated by encrypting an (essential) argument of the following "Protected Area Access Command" (shown in Table3.1). SD Memory Card gets the (essential) argument of the following "Protected Area Access Command" by decrypting received challenge1. Regarding the generation method of challenge1, please see 3.2.1 of *Content Protection for Recordable Media Specification SD Memory Card Book*.

(3) CHANGE_SECURE_AREA (ACMD49) is restricted to execute as follows:

-The use of this command in end-user application is prohibited.

-The use of this command is allowed only in special authorized applications or devices.
(e.g. manufacturer specific application which is used to make a custom SD Memory Card.)
and the following process shall be executed automatically by SD Memory Card itself:

- If the new Protected Area is larger than the former Protected Area, the region of the new Protected Area shall be erased.
- If the new Protected Area is smaller than the former Protected Area, the region of the former Protected Area shall be erased.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

(4) It is possible to send SECURE_ERASE (ACMD38) before SECURE_WRITE_MULTI_BLOCK (ACMD25) for high-speed purpose. In this case, AKE commands shall be done before each Protected Area Access command (ACMD38, ACMD25) is executed.

(5) The card will send "OUT_OF_RANGE" error when the MKB_ID number is bigger than the amount of MKBs saved in the card.

(6) The host shall send the stop transmission command described in the SD Memory Card Specifications Part1: Physical Layer Specification, in case that there was an error while Read or Write operation.

(7) In Protected Area Access Commands (ACMD18, ACMD25, ACMD38), Unit_Address starts at "0". And if data accessing is ”OUT_OF_RANGE”, the operation (write, read or erase) will be performed up to the 'end' of range and then indicates ”OUT_OF_RANGE”.

(8) Write Protect Group, Permanent Write Protection and Temporary Write Protection (see chapter4.3.5 and 4.11 of SD Memory Card Specifications Part1: Physical Specification) do not affect operations on Protected Area.

(9) CHANGE_SECURE_AREA (ACMD49) will set “WP_VIOLATON” error flag in Card Status (and this command will not be performed), in case that the card is Permanent Write Protected, Temporary Write Protected or if the requested Protected Area fall into Write Protected Group area.
In case that there is Write Protected area but the new requested Protected Area does not fall into the Write Protected Area then there will not be an error and the new Protected Area shall be defined.

(10) CHANGE_SECURE_AREA (ACMD49) will set “LOCK_UNLOCK_FAILED” error flag in Card Status (and this command will not be performed), in case that the card is LOCKED (with Password).
It is always the responsibility of the host to verify successful operation by sending SEND_STATUS (CMD13).

(11) As shown in the chapter 3.9.2 of *Content Protection for Recordable Media Specification SD Memory Card Book*, the data field of SECURE_WRITE_MKB (ACMD26) is begun with "Size of MKB" field, followed by "MKB" field , 0 or 4 bytes '0 padding' field, "Kmu" field, 1byte '0 padding' field and "RCC" field.
To simplify the calculation of RCC, further '0 padding' fields are added in the unit data. Those '0 padding' data has no meaning to RCC value. And it is allowed to send extended units after the last unit (that contains the RCC). In this case the content of the extended units shall be all "0".
Here, in "Size of MKB" field, byte length of MKB shall be stored by big-endian as well as *Content Protection for Recordable Media Specification Introduction and Common Cryptographic Elements* (which means that byte 0 is a most significant byte).
Following is SECURE_WRITE_MKB (ACMD26) data format and a drawing of explanation of the format.

*** 1st unit of 512 bytes contains the following data ***

8 Bytes	Size of MKB (= N bytes)
504 Bytes	'0 padding' (*1)

*** A succession of M units of 512 bytes each. (M is MKB size in units of 512 bytes) ***

N Bytes	MKB data
(M * 512 - N) Bytes	'0 padding' (*1)

((M * 512 - N) Bytes should be less than 512 bytes.)

*** (M+2)th unit of 512 bytes data ***

7 Bytes	Kmu
1 Byte	'0 Padding'

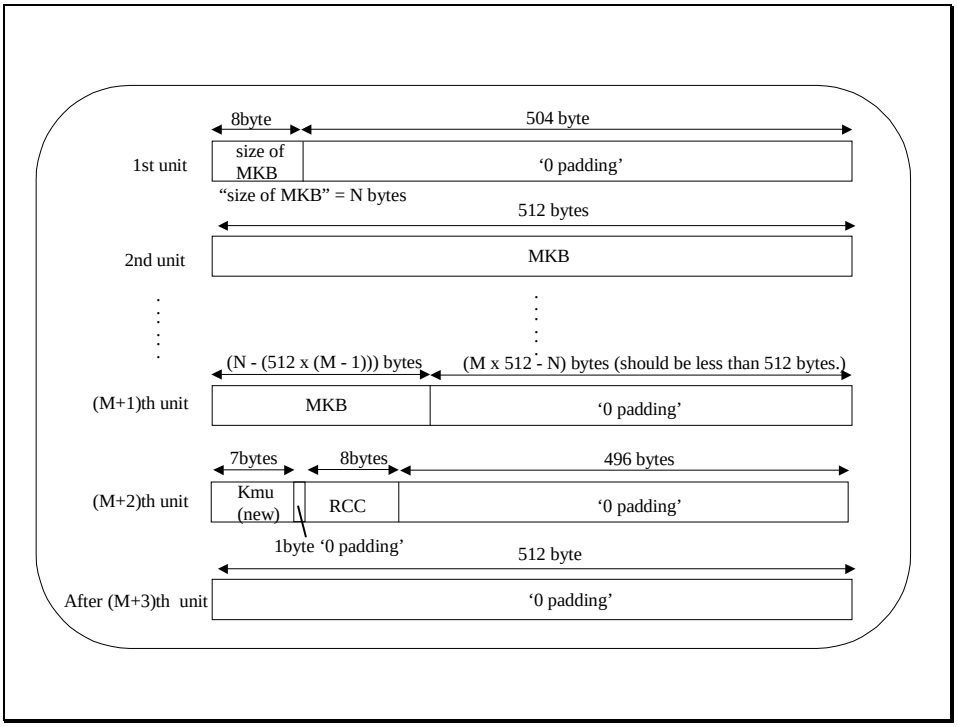
© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

8 Bytes	RCC
512 - 16 = 496 Bytes	'0 Padding' (*1)

*** After (M+3)th units are '0 padding' data ***

*1) To simplify the calculation of RCC remaining data in the unit should be "0".



◊ **Figure 3.1 : SECURE_WRITE_MKB data format**

(12) About 'Unit_Count' in the Security Read/Write/Erase, GET_MKB (ACMD43) and SECURE_WRITE_MKB (ACMD26), 'Unit_Count'=0 means 256 units.

(13) After GO_IDLE_STATE (CMD0), GET_MKB (ACMD43) and GET_MID (ACMD44) are required before the next authentication process.

(14) Wait time after GET_MKB (ACMD43) / SECURE_READ_MULTI_BLOCK (ACMD18) :
Those commands implement read operation with Stop Transmission by the card. Since there is no Busy indication for Reading, the following procedures shall be done by the host before continuing next command:
In SD mode: Confirm 'tran_state' or Wait 100us
In SPI mode: Wait 100us

(15) The host's time out after CHANGE_SECURE_AREA (ACMD49) shall consider the associated Erase operation (described in Note (3)), one block Read operation and two times Sector Write operation.

(16) After AKE sequence, the host is required to confirm AKE result by using SD_STATUS (ACMD13). (Especially in SPI mode, AKE_SEQ_ERROR error is not appeared.) After CHANGE_SECURE_AREA (ACMD49) execution, confirm user area size by reading CSD register and confirm Protected Area size by reading SD status. And initialize user and Protected Area.

(17) The Secure Write Mode and MKB_ID :

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

The host can't read the right data that was written by SECURE_WRITE_MULTI_BLOCK (ACMD25) with another MKB ID and Mode=1. In this case the data from the card is all '0' or '1' data that is specified as a DATA_STAT_AFTER_ERASE in SCR register.

3.2 Usage of Security command

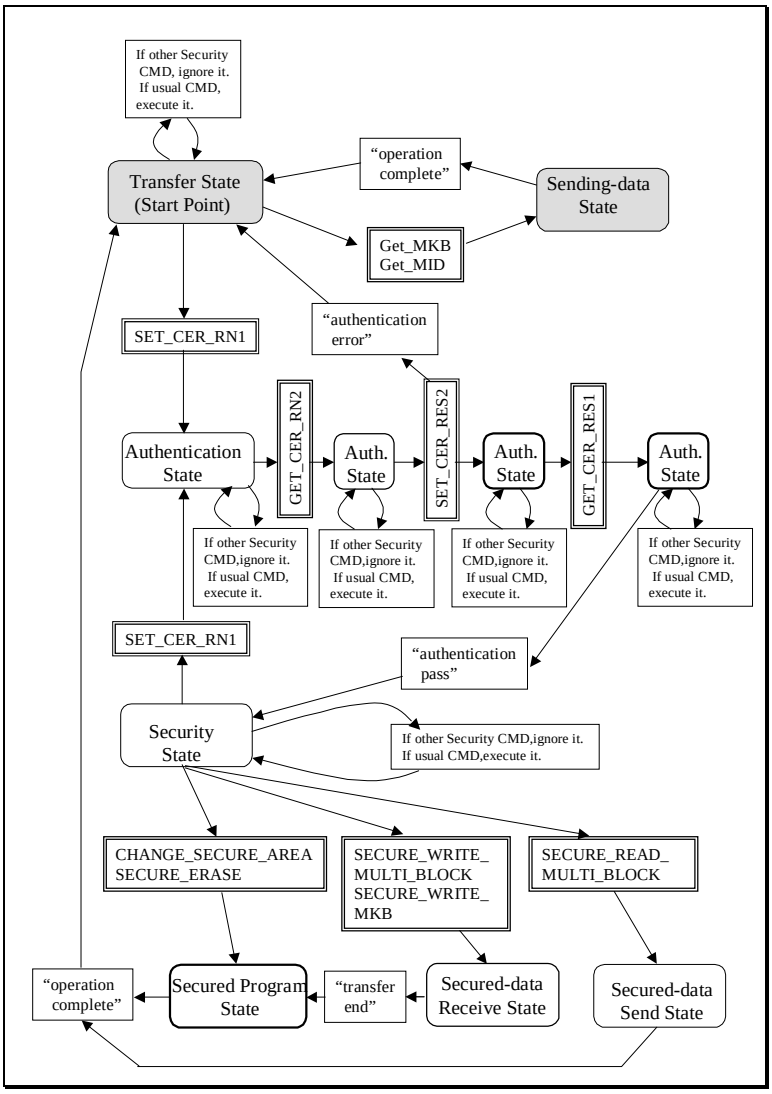
Regarding the usage of Security command, please refer to the chapter 3.3 of *Content Protection for Recordable Media Specification SD Memory Card Book*.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

3.3 SD Memory Card State Diagram on Authentication

Figure 3.2 shows the SD Memory Card State Diagram on Authentication.



◊ **Figure 3.2 : State Diagram**

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

3.4 Summarization of error responses

Next table summarizes the error responses while executing each command.

The error Type, Value and Clear condition are described in SD Memory Card Specifications Part1: Physical Layer Specification.

◊ **Table 3.2 : Error responses List**

CMD index	abbreviation	Error	Description
ACMD18	SECURE_READ_ MULTI_BLOCK	OUT_OF_RANGE	Address over.
		CARD_ECC_FAILED	Card internal ECC was applied but failed to correct the data.
		ERROR	A general or an unknown error occurred during the operation.
ACMD25	SECURE_WRITE _MULTI_BLOCK	OUT_OF_RANGE	Address over.
		ERROR	A general or an unknown error occurred during the operation.
ACMD26	SECURE_ WRITE_MKB	OUT_OF_RANGE	Address over.
		ERROR	A general or an unknown error occurred during the operation.
ACMD38	SECURE_ ERASE	OUT_OF_RANGE	Address over.
		ERROR	A general or an unknown error occurred during the operation.
ACMD43	GET_MKB	OUT_OF_RANGE	MKB_ID over / Unit_Count over / Unit_Offset over
		CARD_ECC_FAILED	Card internal ECC was applied but failed to correct the data.
		ERROR	Issued in secured mode. A general or an unknown error occurred during the operation.
ACMD44	GET_MID	CARD_ECC_FAILED	Card internal ECC was applied but failed to correct the data.
		ERROR	A general or an unknown error occurred during the operation.
ACMD45	SET_CER_RN1	-	There is no error.
ACMD46	GET_CER_RN2	ERROR	In case of unsuccessful save of the Random number in the Flash.
ACMD47	SET_CER_RES2	-	There is no error.
ACMD48	GET_CER_RES1	AKE_SEQ_ERROR	Error in the sequence of authentication process (SD mode only) In SPI mode, AKE_SEQ_ERROR error is not appeared. So the host shall make sure whether AKE process is completed or not by using ACMD13
ACMD49	CHANGE_ SECURE_AREA	OUT_OF_RANGE	Address over.
		WP_VIOLATION	In case that the card is Permanent Write Protected, Temp Write Protected or if requested Protected Area fall into Write Protected Group area.
		ERROR	A general or an unknown error occurred during the operation.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

4. Random Number Generation (RNG)

In AKE process, SD Memory Card and the accessing device can use the following random number generation scheme. In this scheme, each licensee assigns two 56-bit random numbers as Random Number Key (RNK) pair (c_1, c_2) and pre-stores (c_1, c_2) on Hidden Area of SD Memory Card.

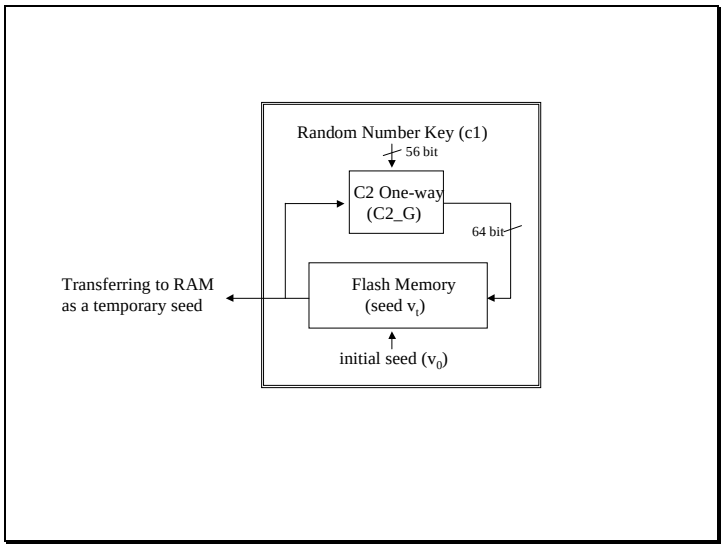
(1) Seed Generation

The 64-bit seed v_t is kept secret in RAM. More concretely, it shall be diffcult to access the seed v_t from outside of SD Memory Card.

The Media Unique Key (56-bit) is used for 56-bit (lsb) of the initial seed v_0 and 8-bit "0" is concatenated as the 8-bit (msb) of the initial seed v_0 when the Card is manufactured. Before shipment, the circuit of RNG freely runs. This makes temporary seed different from K_{mu}.

When the first AKE process is executed after the power of SD Memory Card is turned ON, the seed v_t stored in flash memory is transferred as a temporary seed to RAM. After that, the 64-bit seed (v_t) and 56-bit RNK (c_1) are input to C2 One-way function (C2_G), and 64-bit output (v_{t+1}) of C2_G is stored in flash memory as a next seed (v_{t+1}). Fig. 4.1 shows the procedure of seed generation.

Seed generation is executed only when the first AKE process is executed after the power of SD Memory Card is turned ON.



◊ **Figure 4.1 : Seed Generation**

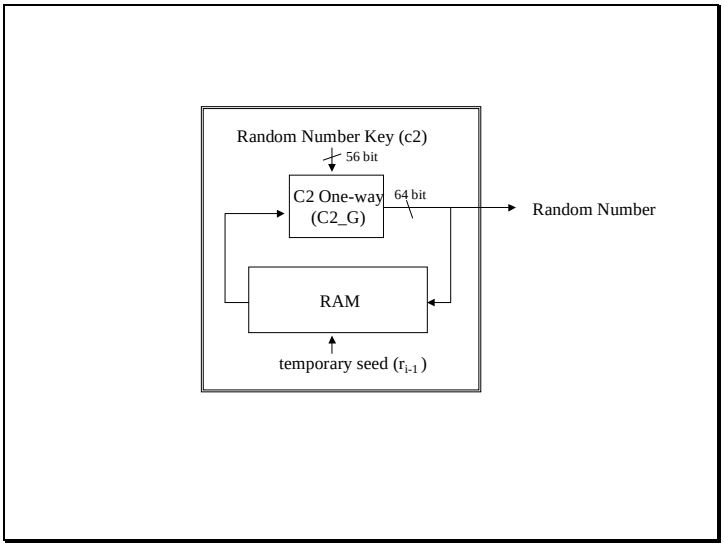
© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

(2) Random Number Generation:

When the first AKE process is executed after the power of SD Memory Card is turned ON, RAM receives the seed from flash memory as a temporary seed (r_{i-1}). After that, the 64-bit temporary seed (r_{i-1}) and 56-bit RNK (c_2) are input to C2 One-way function (C2_G), and 64-bit output (r_i) of C2_G is used as a 64-bit random number and stored in RAM as a next temporary seed (r_i).

Next, Random Number Generation is executed using new temporary seed (r_i). This process is executed repeatedly until the power of SD Memory Card is turned OFF. Fig. 4.2 shows the procedure of random number generation.



◊ **Figure 4.2 : Random Number Generation**

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

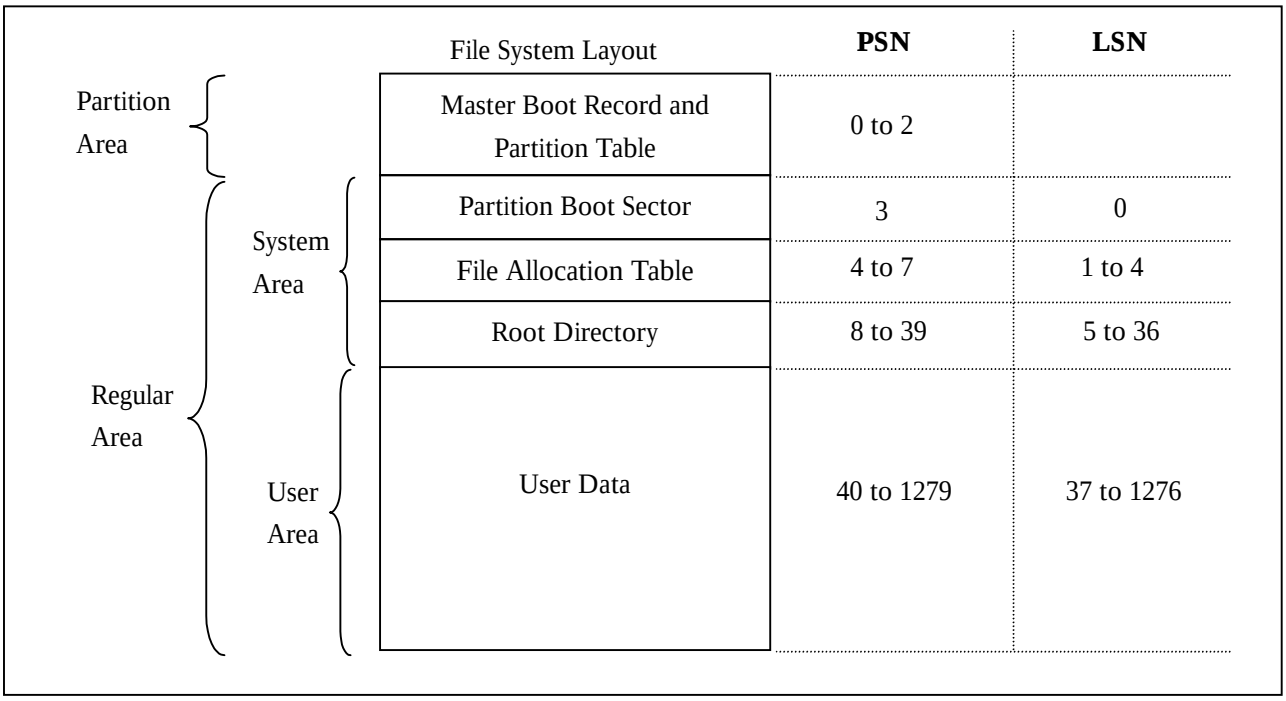
5. File System

5.1 General

This section defines volume structure regarding Protected Area in SD Memory Card. As well as User Data Area, File System in Protected Area uses ISO/IEC 9293-compliant FAT file system. Furthermore, following is a major point that File System in Protected Area differs from one in User Data Area (defined in SD Memory Card Specifications Part2: File System Specification):

- Cluster size of Protected Are is not necessarily a multiple of SD Memory Card erase block size.
- Since it does not require high-speed write for Protected Area, the starting address of User Data is not necessarily located on the boundary of SD Memory Card erase block.

Figure 5.1 shows the example of volume structure for Protected Area.



PSN : Physical Sector Number

LSN : Logical Sector Number

◊ **Figure 5.1 : Example of Volume Structure for Protected Area**

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

5.2 Master Boot Record and Partition Table

See SD Memory Card Specifications Part2: File System Specification as reference since this Master Boot Record and Partition Table are defined in the same way as Master Boot Record and Partition Table in Data Area.

5.3 Partition Boot Sector

Since each field in Partition Boot Sector is almost the same as the one in Partition Boot Sector of Data Area, see SD Memory Card Specifications Part2: File System Specification as reference. Fields that differ from the ones described in Specification above mentioned are explained as follows:

(BP13) Sectors per Cluster

This field shall specify the number of sectors per cluster. It shall be recorded the following number: 1, 2, 4, 8, 16, 32 or 64.

5.4 File Allocation Table

See SD Memory Card Specifications Part2: File System Specification as reference since this File Allocation Table is defined in the same way as File Allocation Table in User Data Area.

5.5 Root Directory

See SD Memory Card Specifications Part2: File System Specification as reference since this Root Directory is defined in the same way as Root Directory in User Data Area.

5.6 User Data

The minimum read/write access size is defined in the units of Sector.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

Annex

A. Test command Requirement

Each SD Memory card manufacturer can individually define the new test command for setting up and testing or analyzing the devices in an SD Memory Card. But such defined new test command shall comply the following security requirement. CMD60, 61, 62, 63 are reserved for manufacturer for test purpose.

Requirement:

(1) Can not read nor update the data stored in a Hidden Area

(2) Can not update the data stored in a System Area

(3) Can not read nor update the data stored in a Protected Area without successful authentication.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

B. Sectors per Cluster and Boundary Unit Recommendation for Protected Area

The following table shows the recommendation for Sectors per Cluster and Boundary Unit of Protected Area.

◊ **Table B-1 : Sectors per Cluster and Boundary Unit Recommendation (Protected Area)**

Protected Area size	Sectors per Cluster	Boundary Unit
~256KB	1	1
~1MB	2	2
~4MB	8	8
~1024MB	32	32
~2048MB	64	64

NOTE: The Table B-1 is not based on the Card Capacity.

Minimum Protected Area size and format parameters for Protected Area are shown in following table.

◊ **Table B-2 : Minimum Protected Area size and format parameters**

Card Capacity	Min Protected Area size(sector)	Sectors per Cluster	an example(values depend on Protected Area size)				
			Clusters	FAT Sec	Hidden	FAT bits	User Data Offset
~4MB	160	1	124	1	1	12	36
~8MB	160	1	124	1	1	12	36
~16MB	320	1	284	1	1	12	36
~32MB	640	2	301	1	3	12	38
~64MB	1280	2	620	2	3	12	40
~128MB	2560	8	314	1	13	12	48
~256MB	5120	8	634	2	11	12	48
~512MB	10240	32	317	1	61	12	96
~1024MB	20480	32	637	2	59	12	96
~2048MB	40960	32	1277	4	55	12	96

However,

Card Capacity…SD Memory Card Capacity.

Min Protected Area size…minimum number of sectors for Protected Area. This parameter is defined from the Card Capacity, using Table B-2.

Sectors per Cluster…number of sectors per cluster. This parameter is defined from the Protected Area size, using Table B-1.

Clusters…number of clusters in User Data. This parameter varies with the Protected Area size.

FAT Sec…number of sectors per FAT. This parameter varies with the Protected Area size.

Hidden…number of sectors existing before Partition Boot Sector. This parameter varies with the Protected Area size.

FAT bits…If the area is formatted with FAT12, FAT bits is 12. And If the area is formatted with FAT16, FAT bits is 16. This parameter varies with the Protected Area size.

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

User Data Offset…number of sectors existing before the starting sector of User Data.

Sectors per Cluster is defined from the Protected Area size. Clusters, FAT Sec, Hidden, FAT bits, and User Data Offset vary with the Protected Area size (Use the parameters in Table B-1 for calculation).

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

C. Type of 16 MKBs on SD Memory Card

This annex shows the type of 16 MKBs pre-stored in the SD Memory Card. As shown in Table C-1, first eight MKBs (MKB "#0" to MKB "#7") are read-only and not updateable by the "dynamic MKB update scheme". Here, the "dynamic MKB update scheme" is described in Chapter 3.9 of *Content Protection for Recordable Media Specification SD Memory Card Book*. Next seven MKBs (MKB "#8" to MKB "#14") are updateable MKB by using the "dynamic MKB update scheme". A last MKB (MKB "#15") is a master MKB which is used in a special authorized accessing device (e.g., a Kiosk), which is allowed to execute the "dynamic MKB update scheme". MKB "#0" is used in SD-Audio application. MKB "#8" is used in SD-Audio extension for Move and Preview, which is defined in SD Memory Card Specifications- Part4: Audio Specification MOVE, MIGRATE AND PREVIEW EXTENSION (from 1.0 to 1.1) and *Content Protection for Recordable Media Specification SD Memory Card Book* Appendix A Move Extension and Appendix C Preview Extension. MKB "#1" to MKB "#7" and MKB "#9" to MKB "#14" are reserved for other applications. Application which uses reserved MKB (MKB "#1" to MKB "#7" and MKB "#9" to MKB "#14") is added in the future.

◊ **TableC-1 : Type of 16MKBs**

Number of MKB	Type	Application
MKB0	Read only	SD-Audio
MKB 1	Read only	Reserved
MKB 2	Read only	Reserved
MKB 3	Read only	Reserved
MKB 4	Read only	Reserved
MKB 5	Read only	Reserved
MKB 6	Read only	Reserved
MKB 7	Read only	Reserved
MKB 8	Updateable	SD-Audio Extension
MKB 9	Updateable	Reserved
MKB 10	Updateable	Reserved
MKB 11	Updateable	Reserved
MKB 12	Updateable	Reserved
MKB 13	Updateable	Reserved
MKB 14	Updateable	Reserved
MKB 15	Master	For Update MKB

© 2000 © 2001 by SD Group (MEI, SanDisk, Toshiba)

CONFIDENTIAL

SD Memory Card Specifications

Part 4 AUDIO SPECIFICATIONS

MOVE, MIGRATE AND PREVIEW EXTENSION

(from AUDIO SPECIFICATION Version 1.0 to AUDIO SPECIFICATION Version 1.1)

Version 1.0 Draft

May 2000

Technical Committee

SD Association

CONFIDENTIAL

Conditions for publication

Publisher and Copyright Holder
SD Association

Exemption
None will be liable for any damages from use of this document.

©2000 SD Association

This page is intentionally left blank.

©2000 SD Association

CONFIDENTIAL

This page is intentionally left blank.

CONFIDENTIAL

INDEX

Part 4 AUDIO SPECIFICATIONS
MOVE, MIGRATE AND PREVIEW EXTENSION

1. General AM1-1
1.1 Scope 1
1.2 Normative References 2

2. Technical Elements AM2-1
2.1 Definitions 1
2.2 Notations 2
2.2.1 Numerical notation 2
2.2.2 Range 2
2.2.3 Bit fields 2
2.3 Terminology 3
2.3.1 AOB 3
2.3.2 Migrate 3
2.3.3 Move 3
2.3.4 POB 3
2.3.5 Preview 3
2.3.6 Secure Data 4
2.3.7 Secure Track Information (S_TKI) 4
2.3.8 Track 4
2.4 Abbreviations 5

3. Introduction AM3-1
3.1 Purpose of the Specifications 1
3.2 Directory and File Structure 3
3.3 Secure Track Information (S_TKI) 5
3.4 Move Content 6
3.5 Migrate Content 8
3.6 Preview Playback Content 9

4. Secure Data Structure AM4-1
4.1 Secure Track General Information (S_TKGI) 2
4.2 Secure Track Text Information Data Area (S_TKTXTI_DA) 12

©2000 SD Association

CONFIDENTIAL

List of Figures

Figure 1.1-1 : Structure of SD-Audio Version 1.1 .....AM1-1

Figure 3.1-1 : An Example of Move Content Flow.....AM3-1
Figure 3.1-2 : An Example of Migrate Content Flow 2
Figure 3.2-1 : Directory and File Configuration 3
Figure 3.3-1 : Relationship between S_TKI and Track..... 5
Figure 3.4-1 : An example of Directory Structure After Move 6

Figure 4-1 : Secure Track Information (S_TKI).....AM4-1
Figure 4.2-1 : Secure Track Information Data Area (S_TKTXTI_DA)..... 12

©2000 SD Association

CONFIDENTIAL

List of Tables

Table 4.1-1 : S_TKGI.....AM4-2
Table 4.1-2 : Character Code for ISRC 9
Table 4.2-1 : Text item's tag in S_TKTXTI_DA 13

©2000 SD Association

CONFIDENTIAL

This page is intentionally left blank.

©2000 SD Association

CONFIDENTIAL

1. General

1.1 Scope

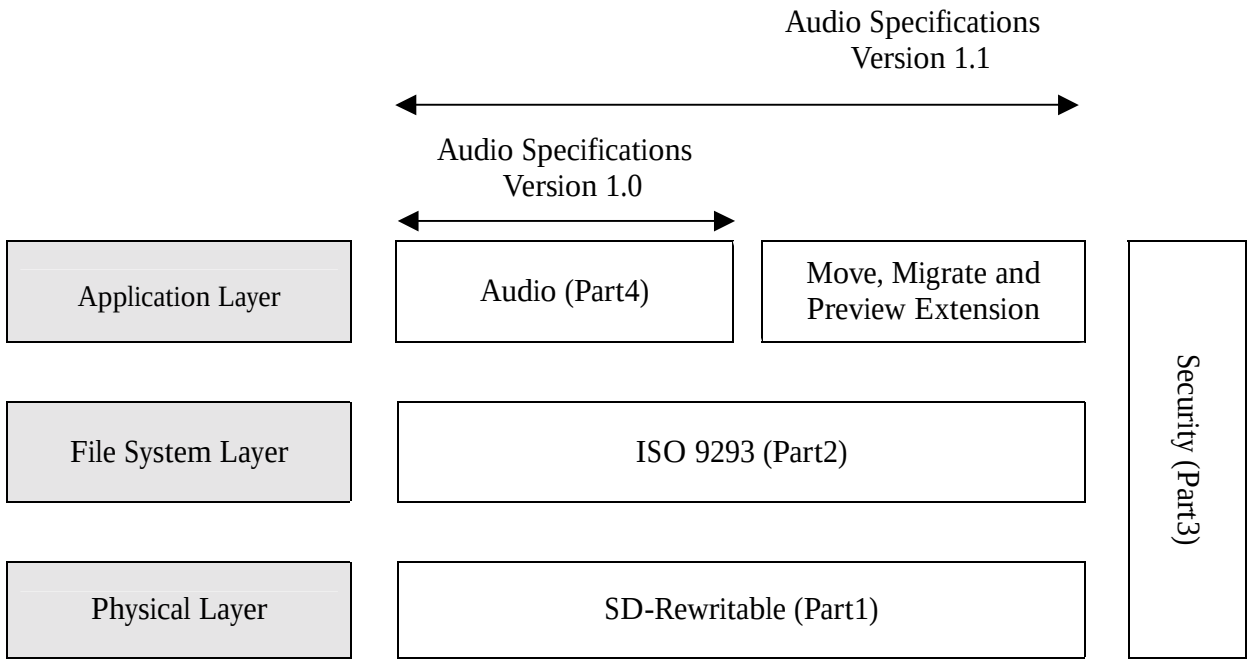
This specification defines the SD-Audio extension format for "Move", "Migrate" and "Preview. " Audio Specifications Version 1.1 consists of this Extension and Audio Specifications Version 1.0.

Audio Specifications Version 1.1 shall comply with **"SDMI Portable Device Specification – Part 1, Version 1.0."** The audio presentation data complies with MPEG2-AAC, MPEG Layer 3 (MP3), and Windows Media Audio (WMA). The picture presentation data basically complies with JPEG.

Devices compliant with Audio Specifications Version 1.1 are ensured to keep compatibility with Version 1.0 contents. Security mechanism for Version 1.1 contents is the same as for Version 1.0 contents. CPRM Extension, Appendix of **"CPRM for SD-Card",** defines the security of the device required for this Extension.

An SD Memory Card whose data is defined in this Extension shall comply with **"Part1. PHYSICAL SPECIFICATIONS," "Part2. FILE SYSTEM SPECIFICATIONS,"** and **"Part3. SECURITY SPECIFICATIONS"** of the SD Memory Card Specifications.

◊ **Figure 1.1-1 : Structure of SD-Audio Version 1.1**



©2000 SD Association

CONFIDENTIAL

1.2 Normative References

1) SD Memory Card Specifications
Part1. PHYSICAL SPECIFICATIONS Version 1.0, February 2000

2) SD Memory Card Specifications
Part2. FILE SYSTEM SPECIFICATIONS Version 1.0, February 2000

3) SD Memory Card Specifications
Part3. SECURITY SPECIFICATIONS Version 1.0, February 2000

4) SD Memory Card Specifications
Part4. AUDIO SPECIFICATIONS Version 1.0, February 2000

5) CPRM for SD-Card
Content Protection for Recordable Media Specification SD Memory Card Book, available soon

6) SDMI Document PDWG99070802
SDMI Portable Device Specification – Part 1, Version 1.0

7) ISO 3901: 1986
Documentation – International Standard Recording Code (ISRC)

8) ISO/IEC 646: 1983
Information processing - ISO 7-bit coded character set for information interchange

9) ISO 8859-1: 1987
Information processing – 8-bit single-byte coded graphic character sets – Part 1 Latin alphabet No.1

10) JIS X 0201-1997
7-bit and 8-bit coded character sets for information interchange

11) RIAJ document RIS506-1996
Music Shift JIS Kanji character set

12) ISO/IEC 13818-7: 1997
Information technology – Generic coding of moving pictures and associated audio information – Part 7 Advanced Audio Coding (AAC)

©2000 SD Association

CONFIDENTIAL

13) ISO/IEC 13818-7/COR1: 1998
Information technology – Generic coding of moving pictures and associated audio information – Part 7 Advanced Audio Coding (AAC) TECHNICAL CORRIGENDUM 1

14) ISO/IEC 11172-3: 1993
Information technology – Generic coding of moving pictures and associated audio information – Part 3 Audio

15) ISO/IEC 13818-3: 1995
Information technology – Generic coding of moving pictures and associated audio information – Part 3 Audio

16) ISO/IEC 9293: 1994
Information technology – Volume and file structure of disk cartridges for information interchange

17) ISO/IEC 10918-1: 1994
Information technology – Digital compression and coding of continuous-tone still pictures: requirements and guidelines

©2000 SD Association

CONFIDENTIAL

This page is intentionally left blank.

©2000 SD Association

CONFIDENTIAL

2. Technical Elements

2.1 Definitions

For the purpose of this specification, the following definitions apply:

RBN : Relative Byte Number (unsigned value) from the first byte of a defined field, starting with zero.

RBP : Relative Byte Position within a descriptor, starting with zero.

reserved : Reserved field. The term "reserved" indicates that a value may be used in the future for the SD Memory Card specifications. Unless otherwise specified within this specification, all reserved bits shall be set to zero.

ZERO : represents a single bit with the value 0.

ONE : represents a single bit with the value 1.

data_element[m..n] : The inclusive range of bits between bit m and bit n in the data_element.

©2000 SD Association

2.2 Notations

2.2.1 Numerical notation

Numbers in decimal notation are represented by decimal digits, namely 0 to 9. **Digit(s)** represents any digit from digit 0 to 9.

Numbers in Hexadecimal notation are represented as a sequence of one or more hexadecimal digits namely 0 to 9 and A to F, suffixed by "h."

Numbers in binary notation are represented as a sequence of one or more binary digits namely 0 to 1, suffixed by "b."

Multiple-byte numerical values in a description field shall be recorded in the **big endian** representation.
e.g.) The 32-bit hexadecimal number '12345678h' shall be recorded as '12h,' '34h,' '56h,' and '78h.'

1 Mbytes represents 10^6 bytes.

1 Gbytes represents 10^9 bytes.

1 kB represents 2^{10} bytes.

1 MB represents 2^{20} bytes.

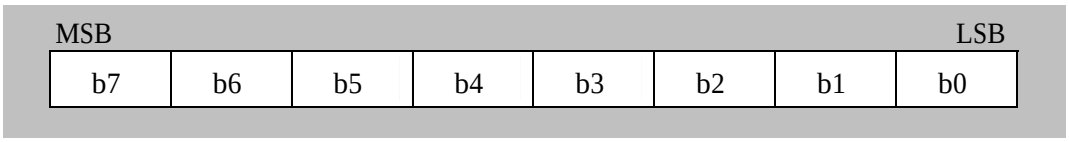
1 GB represents 2^{30} bytes.

2.2.2 Range

Constant_1..Constant_2 or [Constant1..Constant2] denotes the range from Constant_1 to Constant_2 both inclusive, in increments of 1.

2.2.3 Bit fields

Certain fields or parts of fields are interpreted as an array of bits.
This array of bits shall be referred to as a bit field. Bit positions within an ***n*** bit field are numbered such that the least significant bit is numbered 0 and the most significant bit is numbered ***n–1***.



©2000 SD Association

2.3 Terminology

2.3.1 AOB

Compressed audio data is treated as the unit of Audio Object (AOB) in the SD-Audio Specifications. Compressed audio formats operated by AOB are MPEG2-AAC (ISO/IEC 13818-7), MPEG layer 3 (ISO/IEC 11172-3, ISO/IEC 13818-3), and Windows Media Audio (Microsoft).

2.3.2 Migrate

This is a procedure to transfer SD-Audio contents from an SD Memory Card to a consumer's local storage devices (SDMI compliant application) under the following conditions:

- SD-Audio contents originate from the analog input or unprotected digital input of an SD-Audio Portable Device.
- SD-Audio contents are screened according to the SDMI screening rules.
- The contents shall be SDMI Protected Contents after they are transferred.

As for the details of the definition and requirement of Migrate, refer to section 5.7.3 of the **"SDMI Document PDWG99070802"**.

2.3.3 Move

This is a procedure to transfer SD-Audio contents from an SD Memory Card to consumer's local storage devices (SDMI compliant application) under the following conditions:

- SD-Audio contents originate from SDMI Protected Contents for Distribution.
- The contents shall be SDMI Protected Contents after they are transferred.

As for the details of the definition and requirement of Move, refer to section 3.19 and section 5.7.1 of the **"SDMI Document PDWG99070802"**.

2.3.4 POB

Picture Information handled in the SD-Audio Specification is called Picture Object (POB). POB only stores a single piece of picture information in a POB file.

2.3.5 Preview

This is a procedure to play contents under the limited or restricted conditions that are specified by content provider and/or distributor.

©2000 SD Association

2.3.6 Secure Data

This is the information on attributes and text information of the Presentation Data, which are given by a content provider and stored in an SD Memory Card securely. It consists of one logical structure, Secure Track Information (S_TKI).

2.3.7 Secure Track Information (S_TKI)

This is Track-related information securely stored in the SD Memory Card.

2.3.8 Track

In the case of Music application, it is the equivalent of a 'track.' In the case of Reading Book application, it is the equivalent of a 'chapter.' A Track consists of audio data (mandatory), time management data (mandatory), text data (optional), and picture data (optional).

©2000 SD Association

CONFIDENTIAL

2.4 Abbreviations

S_TKGI	Secure Track General Information
S_TKI	Secure Track Information
S_TKI_AOB_ATR	AOB Attribute of S_TKI
S_TKI_APP_ATR	Application Attribute of S_TKI
S_TKI_BLK_ATR	Block Attribute of S_TKI
S_TKI_FR_ID1	Free ID Area 1 of S_TKI
S_TKI_FR_ID2	Free ID Area 2 of S_TKI
S_TKI_FR_ID3	Free ID Area 3 of S_TKI
S_TKI_FR_ID4	Free ID Area 4 of S_TKI
S_TKI_ID	S_TKI Identifier
S_TKI_ISRC	ISRC of S_TKI
S_TKI_LNK_PTR	Link Pointer of S_TKI
S_TKI_PB_TM	S_TKI Playback Time
S_TKI_POB_ATR	POB Attribute of S_TKI
S_TKI_POB_SRP	POB Search Pointer of S_TKI
S_TKI_SZ	Size of S_TKI
S_TKI_TI1_ATR	Attribute of S_TKI Text Information 1
S_TKI_TI2_ATR	Attribute of S_TKI Text Information 2
S_TKIN	S_TKI Number
S_TKTXTI_ABM	Album Name
S_TKTXTI_ARR	Editor of Track
S_TKTXTI_ART	Artist Name
S_TKTXTI_CMP	Composer of Track
S_TKTXTI_CRD	Date of Track
S_TKTXTI_DA	Secure Track Text Information Data Area
S_TKTXTI_FR1	Free Item 1
S_TKTXTI_FR2	Free Item 2
S_TKTXTI_FR3	Free Item 3
S_TKTXTI_FR4	Free Item 4
S_TKTXTI_FR5	Free Item 5
S_TKTXTI_FR6	Free Item 6
S_TKTXTI_GNR	Genre of Track
S_TKTXTI_MSS	Artist's Message of Track
S_TKTXTI_PCM	Provider's Comment on Track
S_TKTXTI_PRD	Producer of Track
S_TKTXTI_RCD	Recording Company of Track
S_TKTXTI_SW	Song Writer of Track
S_TKTXTI_TTL	Title Name

©2000 SD Association

CONFIDENTIAL

S_TKTXTI_UCM	User's Comment on Track
S_TKTXTI_URL	URL of Track

©2000 SD Association

3. Introduction

3.1 Purpose of the Specifications

This Specification is designed to satisfy the following features:

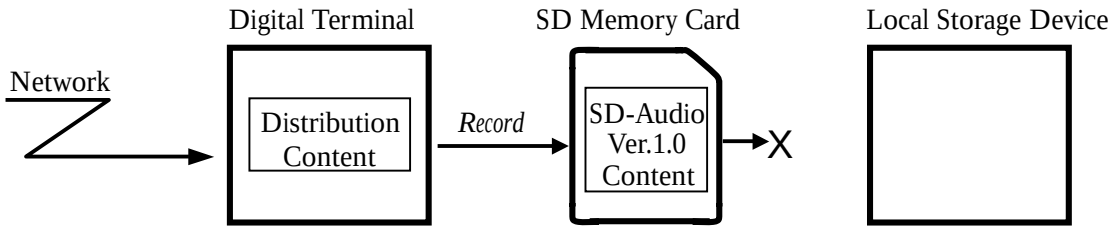
1) **Move**

A consumer who directly records an audio content at a Digital Terminal (e.g. KIOSK terminal, cellular phone, STB) onto the SD Memory Card in the SD-Audio format may want to transfer it from the SD Memory Card to his/her local storage device in order to make a space in the card for other contents.

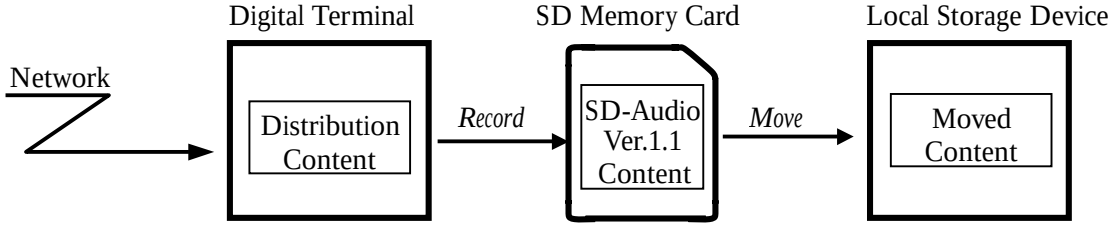
[Figure 3.1-1] illustrates the content flow of move. [Figure 3.1-1 (a)] illustrates a case where contents are recorded as SD-Audio Version 1.0 contents. Distributed content will be recorded into consumer's SD Memory Card at the Digital Terminal. A consumer can play the content on an SD-Audio 1.0 Player. However, the consumer cannot copy or move it to another local storage device since the SD-Audio Version 1.0 content is an SDMI-protected content for local use only and the SDMI specifications do not allow the consumer to copy or move it. [Figure 3.1-1 (b)] illustrates a case where contents are recorded as SD-Audio Version 1.1 contents. A content originating from the Digital Terminal can be moved to consumer's local storage device if the content is ensured to be the SDMI-protected content for distribution. As for the details of the SDMI Move rule, refer to **"SDMI Document PDWG99070802."**

◊ **Figure 3.1-1 : An Example of Move Content Flow**

(a) In the case of SD-Audio Version 1.0



(b) In the case of SD-Audio Version 1.1



©2000 SD Association

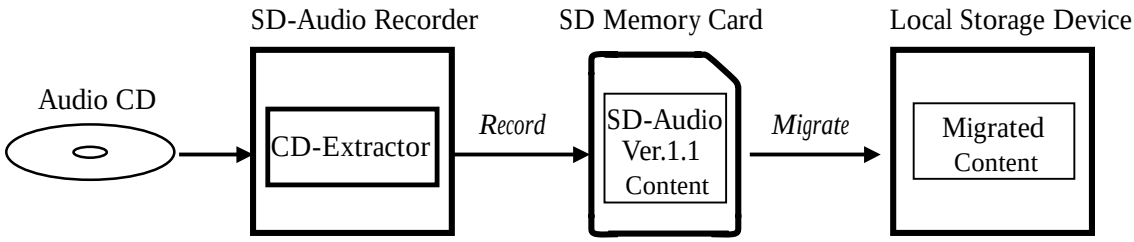
CONFIDENTIAL

2) **Migrate**

An SD-Audio recorder records audio data directly into the SD Memory Card from analog input or unprotected digital input. For example, the SD-Audio recorder with a CD Extraction feature can record audio contents from a CD to the SD Memory Card in the SD-Audio format. A consumer may want to transfer such contents from the SD Memory Card to his/her local storage device in order to make a space in the card for other contents.

[Figure 3.1-2] illustrates the flow of contents recorded by the SD-Audio recorder. The SD-Audio Recorder records an audio content as the SD-Audio content from a legacy Audio CD. The SD-Audio content can be migrated to the consumer's local storage device. As for the details of the SDMI Migrate rule, refer to **"SDMI Document PDWG99070802."**

◊ **Figure 3.1-2 : An Example of Migrate Content Flow**



3) **Preview Playback**

Content providers may distribute sample contents to consumers for promotion purpose via Digital Terminals. Such contents can be played under the limited or restricted conditions that are specified by content provider and/or distributor. The preview functionality of an SD-Audio 1.1 Player shall not be applied to other purposes such as pay-per-play.

4) **Secure Recording of Track Information**

A consumer acquires a content via the digital terminal such as digital KIOSK terminal and cellular phones, and records the content directly onto the SD Memory Card. The content may have content-specific information including song name, artist name, and content provider name. Such Track information is provided by the content provider and/or distributor. Therefore, it shall be guaranteed to be the same data as the one originally assigned.

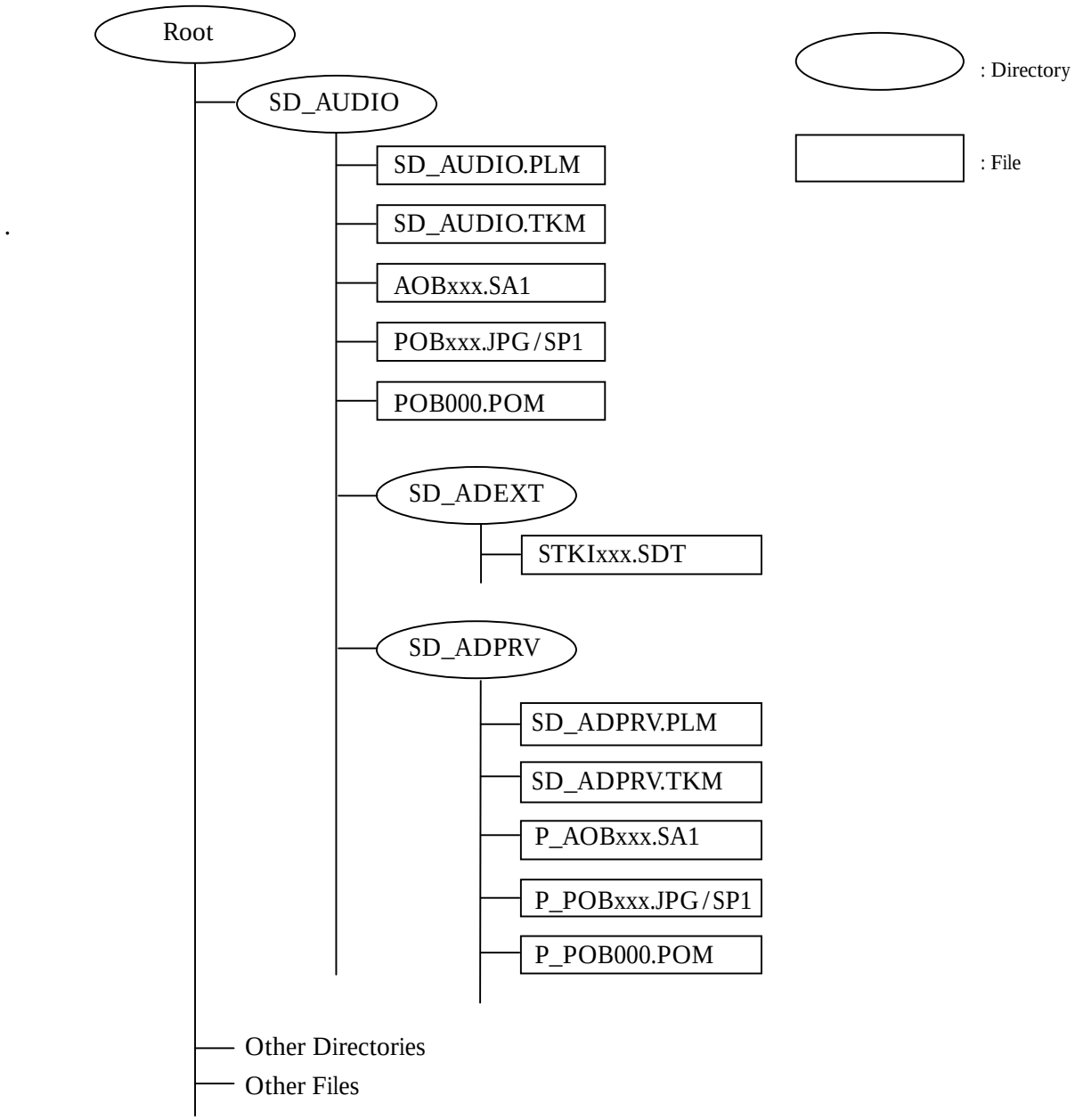
©2000 SD Association

CONFIDENTIAL

3.2 Directory and File Structure

[Figure 3.2-1] illustrates directories and files in which data compliant to this Specification is recorded. All files are placed under an SD_AUDIO directory, an SD_ADEXT directory, and an SD_ADPRV directory. The SD_AUDIO directory and all files under the SD_AUDIO directory comply with the SD-Audio specifications. As for details, refer to **"Part 4 AUDIO SPECIFICATIONS Version 1.0."**
The SD_ADEXT directory and all files under the SD_ADEXT directory are set to 'read-only' and 'hidden' attributes.
The SD_ADPRV directory and all files under the SD_ADPRV directory are set to 'read-only' and 'hidden' attributes.

◊ **Figure 3.2-1 : Directory and File Configuration**



.

Following file is to be created under the SD_ADEXT directory.

- **STKIxxx.SDT**

This file stores Secure Track Information (S_TKI) for related audio object data in the SD_AUDIO directory. This file shall be encrypted. "xxx" takes a value between '001' and '999.'

Following files are to be created under the SD_ADPRV directory.

- **SD_ADPRV.PLM**

This file stores the Playlist Manager for preview playback contents. The structure of this file is the same as that of SD_AUDIO.PLM in the SD-Audio Specifications. As for the details of SD_AUDIO.PLM, refer to **"Part4. AUDIO SPECIFICATIONS Version 1.0."**

- **SD_ADPRV.TKM**

This file stores the Track Manager for preview playback contents. The structure of this file is the same as that of SD_AUDIO.TKM in the SD-Audio Specifications. As for the details of SD_AUDIO.TKM, refer to **"Part4. AUDIO SPECIFICATIONS Version 1.0."**

- **P_AOBxxx.SA1**

This file stores encrypted audio object data for preview playback contents. "xxx" takes a value between '001' and '999.' The structure of this file is the same as that of AOBxxx.SA1 in the SD-Audio Specifications. As for the details of AOBxxx.SA1, refer to **"Part4. AUDIO SPECIFICATIONS Version 1.0."**

- **P_POBxxx.JPG and P_POBxxx.SP1**

These files store picture object data for preview playback contents. "xxx" takes a value between '001' and '999.' The structures of these files are the same as those of POBxxx.JPG and POBxxx.SP1 in the SD-Audio Specifications. As for the details of POBxxx.JPG and POBxxx.SP1, refer to **"Part4. AUDIO SPECIFICATIONS Version 1.0."**

- **P_POB000.POM**

This file stores management information regarding picture object files for preview playback contents. The structure of this file is the same as that of POB000.POM in the SD-Audio Specifications. As for the details of POB000.POM, refer to **"Part4. AUDIO SPECIFICATIONS Version 1.0."**

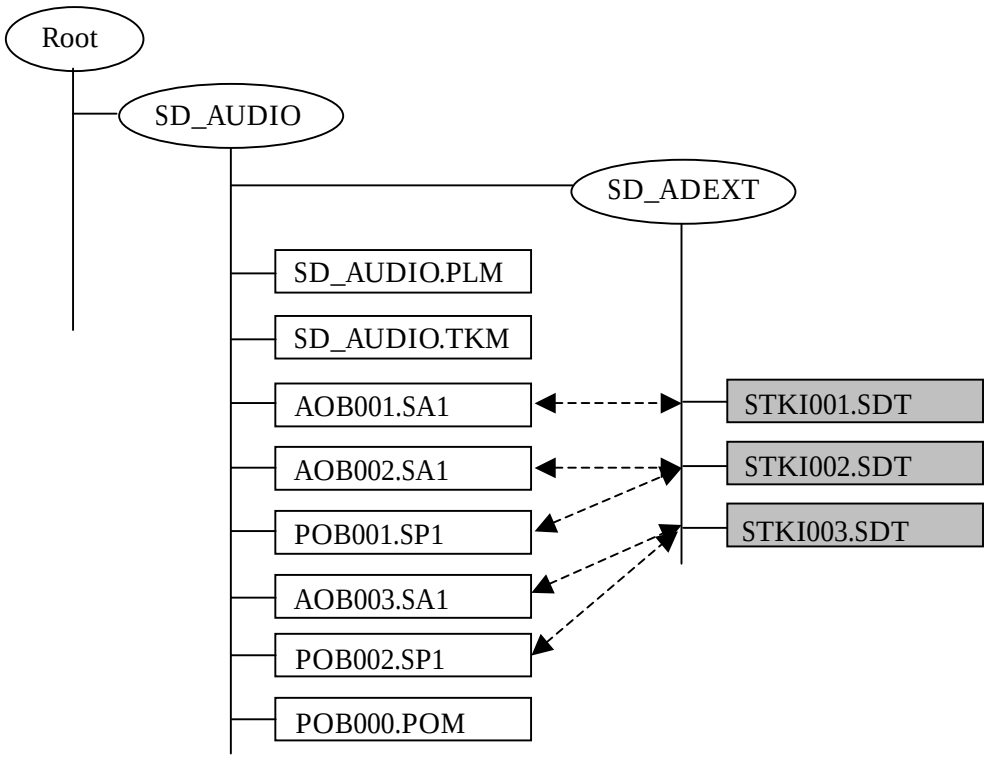
©2000 SD Association

3.3 Secure Track Information (S_TKI)

Secure Track Information (S_TKI) stores Track Information (TKI) for the corresponding Track in a secure way, except for information related to Time Search Table (TMSRT).

[Figure 3.3-1] illustrates the relationship between an S_TKI file and a related object data file. The STKIxxx.SDT file is linked to the audio object in an SD_AUDIO directory by its "xxx" in the file name, which corresponds to the TKI number. The STKIxxx.SDT file is linked to the picture object in the SD_AUDIO directory by S_TKI_POB_SRP in STKIxxx.SDT.

◊ **Figure 3.3-1 : Relationship between S_TKI and Track**



©2000 SD Association

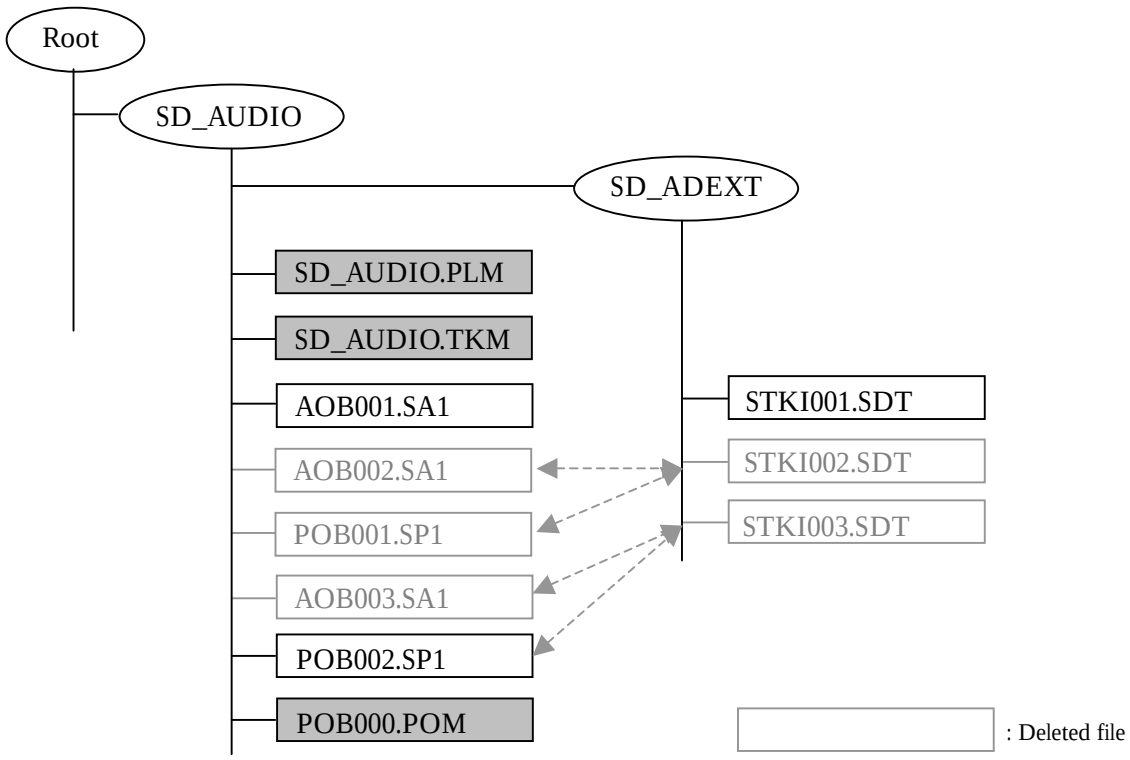
3.4 Move Content

All Tracks in the SD_AUDIO directory may have permission to move. No Tracks in the SD_ADPRV directory shall be permitted to move.
When a Track in the SD_AUDIO directory has permission to move, the Track which consists of Audio Object(s) (AOB(s)), Picture Object(s) (POB(s)), and Secure Track Information(s) (S_TKI(s)) can be moved from the SD Memory Card to the consumer's local storage device.

The Move permission information is stored in the Move Usage Rule as specified by the content provider and/or distributor in the Protected Area. As for details of the data structure, refer to **"CPRM for SD-Card."**

[Figure 3.4-1] shows an example of a directory structure after Track #2 and Track #3 in [Figure 3.3-1] are moved, assuming that a POB referred to by Track #2 is only referred to by Track#2 and that the POB referred to by Track #3 is referred to by other Tracks.

◊ **Figure 3.4-1 : An Example of Directory Structure After Move**



Track Information (TKI) of the moved Track shall be deleted from TKGI in SD_AUDIO.TKM. The moved Track shall be deleted from the Default Playlist in SD_AUDIO.PLM. POB Count Information for the corresponding POB shall be decremented in POB000.POM. An AOB file and an S_TKI file corresponding to the moved Track shall be deleted. The POB file whose reference counter is '0' shall be deleted.
In the local storage device, the moved content shall be SDMI Protected Content.

When the SD-Audio player compliant to this specification executes one of the following procedures for a Track, the player shall delete the Secure Track Information file (STKIxxx.SDT) corresponding to the Track after the procedure is completed.

- Deleting a Track
- Combining Tracks
- Dividing a Track

A Move content shall not be a Migrate content, and shall not be a Preview playback content.

©2000 SD Association

3.5 Migrate Content

In the User Data Area, Data structure of Migrate content shall comply with **"Part4. AUDIO SPECIFICATIONS Version 1.0."** Secure Track Information (S_TKI) is not used.

Track Information (TKI) of the migrated Track shall be deleted from TKGI in SD_AUDIO.TKM. The migrated Track shall be deleted from the Default Playlist in SD_AUDIO.PLM. An AOB file corresponding to the migrated Track shall be deleted.
In the local storage device, the migrated content shall be SDMI Protected Content.

Editing procedures of Tracks for Migrate contents shall comply with **"Part4. AUDIO SPECIFICATIONS Version 1.0."**
In case of combining two Tracks, if and only if both of them are Migrate contents, the combined Track can be a Migrate content.

The Migrate information is stored in the Protected Area. As for details of Migrate information in the Protected Area, refer to **"CPRM for SD-Card"**.

A Migrate content shall not be a Move content, and shall not be a Preview playback content.

©2000 SD Association

CONFIDENTIAL

3.6 Preview Playback Content

The SD-Audio player compliant to this specification shall play Preview playback contents under the SD_ADPRV directory only if the conditions set by the Preview Usage Rule are satisfied. Such Preview playback shall also be controlled as specified by the Preview Usage Rule. The Preview Usage Rule is stored in the Protected Area. As for details of the Preview Usage Rule, refer to **"CPRM for SD-Card"**.

The Preview playback Tracks in the Preview playback contents shall be played in accordance with the same playback procedure of the SD-Audio 1.0 contents except for the following procedures. The following procedures shall be prohibited for the preview playback contents.

- Combining Tracks
- Dividing a Track

A Preview playback content shall not be a Move content, and shall not be a Migrate content.

Note: When the conditions set by the Preview Usage Rule for a Preview playback Track are not satisfied anymore, the SD-Audio player compliant to this specification may delete the Track in accordance with the deletion procedure as specified in **"Part4. AUDIO SPECIFICATIONS Version 1.0."**

©2000 SD Association

CONFIDENTIAL

This page is intentionally left blank.

©2000 SD Association

CONFIDENTIAL

4. Secure Data Structure

Secure Track Information (S_TKI) is a certification data structure which is placed in the SD_ADEXT directory. Such information as attributes and text information of the object data is stored securely in S_TKI. S_TKI consists of two data structures, Secure Track General Information (S_TKGI) and Secure Track Text Information Data Area (S_TKTXTI_DA).

S_TKI describes information for a Track within SD_AUDIO directory.
S_TKI starts with Secure Track General Information (S_TKGI), followed by Secure Track Text information Data Area (S_TKTXTI_DA) as shown in [Figure 4-1].

One S_TKI is associated with one AOB as well as the corresponding TKI.
S_TKGI and S_TKTXTI_DA shall be encrypted.

◊ **Figure 4-1 : Secure Track Information (S_TKI)**

Secure Track Information (S_TKI)

Secure Track General Information (S_TKGI) (Mandatory)	Fixed Size (256B)
Secure Track Text Information Data Area (S_TKTXTI_DA) (Mandatory)	Fixed Size (256B)

©2000 SD Association

4.1 Secure Track General Information (S_TKGI)

As shown in [Table 4.1-1], following are described in Secure Track General Information (S_TKGI): S_TKI Identifier, S_TKI number, playback time of a Track, AOB attribute, POB attribute, text information attribute, ISRC, S_TKI application attributes, free ID areas, and POB Search Pointers.

◊ **Table 4.1-1 : S_TKGI** (Description order)

RBP		Contents	Number of bytes
0 to 1	S_TKI_ID	S_TKI Identifier	2 bytes
2 to 3	S_TKIN	S_TKI Number	2 bytes
4 to 5	S_TKI_BLK_ATR	Block Attribute of S_TKI	2 bytes
6 to 7	S_TKI_LNK_PTR	Link Pointer to next S_TKI	2 bytes
8 to 11	S_TKI_SZ	Size of S_TKI	4 bytes
12 to 15	S_TKI_PB_TM	Playback time of Track	4 bytes
16 to 19	S_TKI_AOB_ATR	Audio Attribute of S_TKI	4 bytes
20 to 23	reserved	reserved	4 bytes
24 to 25	S_TKI_POB_ATR	Picture Attribute of TKI	2 bytes
26 to 27	reserved	reserved	2 bytes
28 to 34	reserved	Reserved for copyright management Information	7 bytes
35	reserved	reserved	1 byte
36 to 37	S_TKI_TI1_ATR	Attribute of Text1	2 bytes
38 to 39	S_TKI_TI2_ATR	Attribute of Text2	2 bytes
40 to 43	reserved	reserved	4 bytes
44 to 53	S_TKI_ISRC	ISRC code	10 bytes
54 to 55	S_TKI_APP_ATR	S_TKI application attributes	2 bytes
56 to 67	reserved	reserved	12 bytes
68 to 83	S_TKI_FR_ID1	Free ID Area of S_TKI	bytes
84 to 99	S_TKI_FR_ID2	Free ID Area of S_TKI	bytes
100 to 115	S_TKI_FR_ID3	Free ID Area of S_TKI	bytes
116 to 131	S_TKI_FR_ID4	Free ID Area of S_TKI	16 bytes
132 to 175	reserved	reserved	44 bytes
176 to 255	S_TKI_POB_SRP	S_TKI_POB Search Pointers (4B*20)	80 bytes
Total			256 bytes

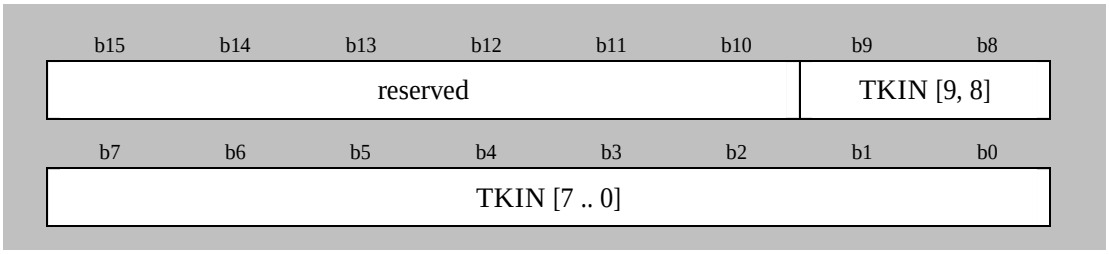
©2000 SD Association

(RBP 0 to 1) S_TKI _ID

Describes "A7" to identify S_TKI with character set code of ISO646 (a-characters).

(RBP 2 to 3) S_TKIN

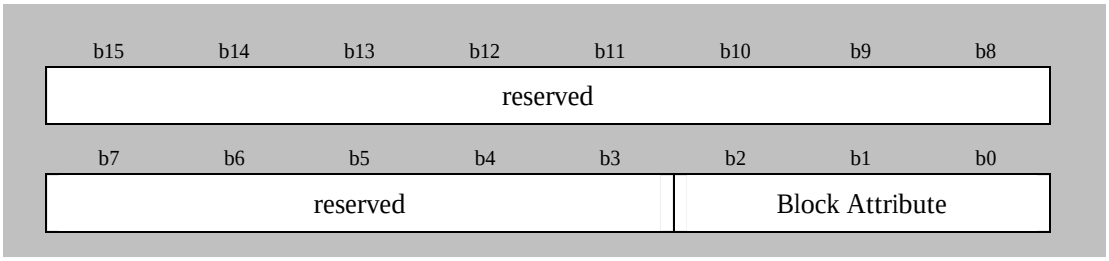
Describes a TKI number ranging from '1 to 999'.
This value is equal to the number in the S_TKI file name.



TKIN ... Describes the number between '1' and '999'.

(RBP 4 to 5) S_TKI_BLK_ATR

Describes whether or not the S_TKI is used. In a case where the S_TKI is used, it describes whether or not one Track consists of one S_TKI, and an S_TKI's position within the Track. In a case where the S_TKI is not used, it further describes whether or not the S_TKI has been used.



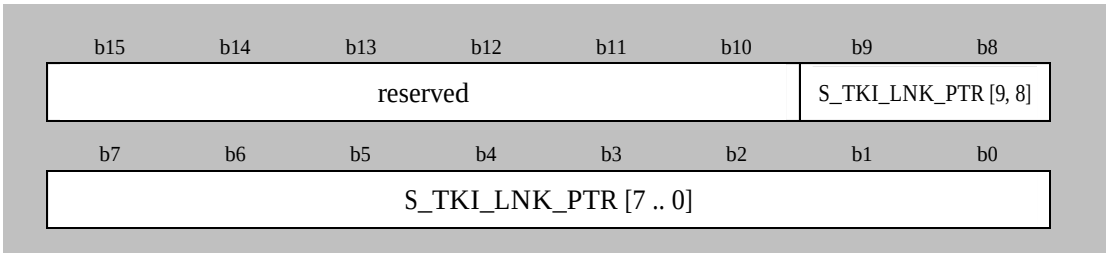
Block Attribute ... 000b : One Track consists of one S_TKI.
001b : One Track consists of multiple S_TKIs. This S_TKI is located in the head position.
010b : One Track consists of multiple S_TKIs. This S_TKI is located in the middle position.
011b : One Track consists of multiple S_TKIs. This S_TKI is located in the tail position.
Others : reserved

©2000 SD Association

CONFIDENTIAL

(RBP 6 to 7) S_TKI_LNK_PTR

Describes a S_TKI number when this S_TKI continues to other S_TKIs.



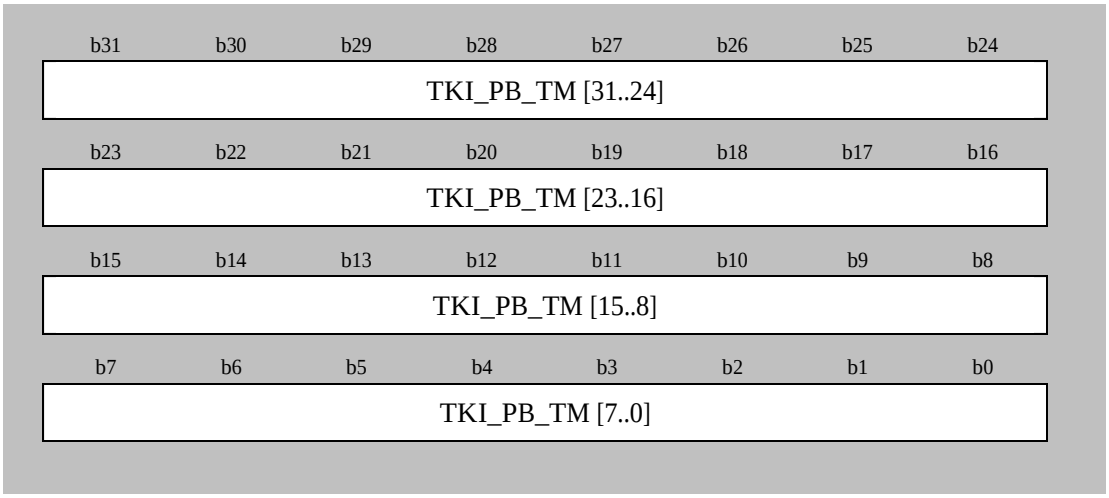
S_TKI_LNK_PTR ... Describes the number between '0' and '999'.
When the Block Attribute in S_TKI_BLK_ATR is '011b' and this S_TKI does not continue to other S_TKIs, enter '000h'.

(RBP 8 to 11) S_TKI_SZ

Describes the data size of the S_TKI in the number of bytes. In this specification, enter '512'.

(RBP 12 to 15) S_TKI_PB_TM

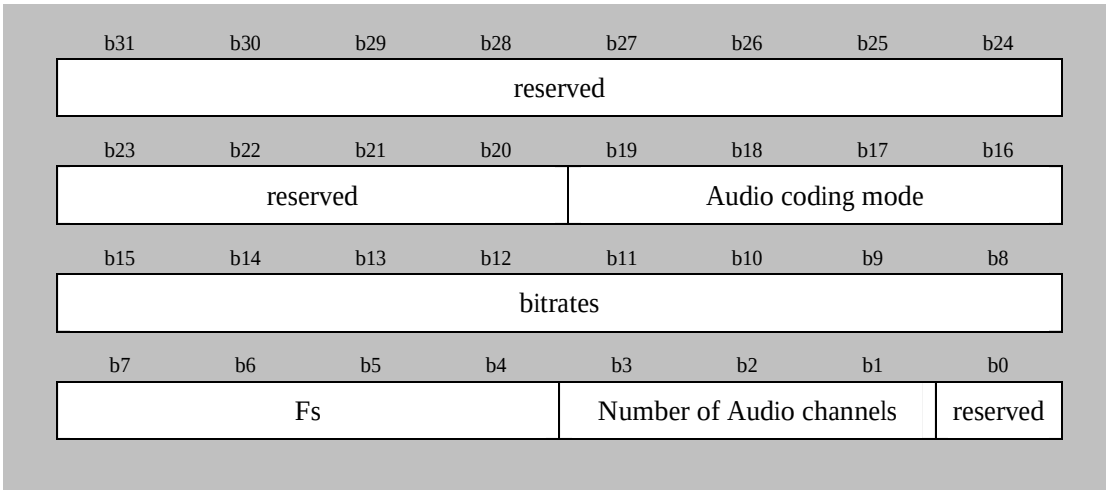
Describes a total playback time for the Track referred to by this S_TKI in msec.



©2000 SD Association

(RBP 16 to 19) S_TKI_AOB_ATR

Describes the audio attribute corresponding to this S_TKI.



Audio coding mode ... 0000b : MPEG-2 AAC LC profile (with ADTS header)
0001b : MPEG layer3 (MP3)
0010b : Windows Media Audio (WMA)
Others : reserved

bitrates (kbit/s/ch) ... Describes a number between '16' and '72' for AAC.
Describes a number between '16' and '96' for MPEG1 Layer3
Describes a number between '16' and '80' for MPEG2 Layer3 LSF
Describes a number between '8' and '80' for Windows Media Audio
11111111b : not specified (Variable bitrate).

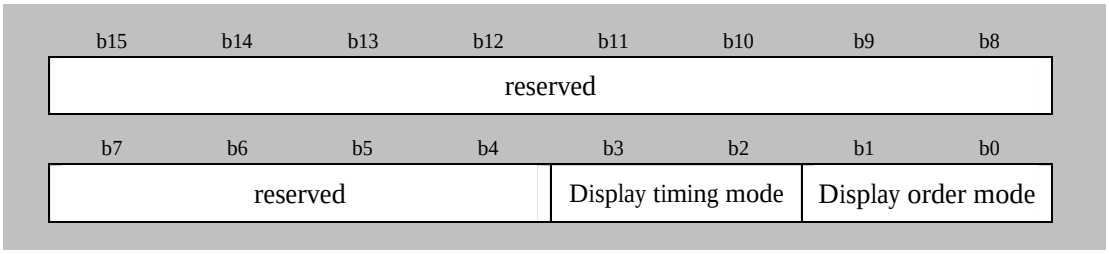
Fs ... 0000b : 48 kHz
0001b : 44.1 kHz
0010b : 32 kHz
0011b : 24 kHz
0100b : 22.05 kHz
0101b : 16 kHz
Others : reserved

Number of Audio Channels ... 000b : 1ch (mono)
001b : 2ch (stereo)
Others : reserved

©2000 SD Association

(RBP 24 to 25) S_TKI_POB_ATR

Describes the POB attribute corresponding to this S_TKI.

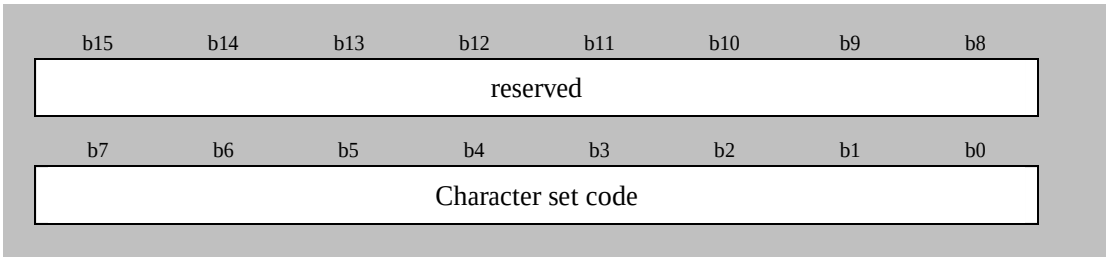


Display timing mode … 00b : Slideshow
01b : Browsable
Others : reserved

Display order mode … 00b : Sequential
01b : Random
10b : Shuffle
Others : reserved

(RBP 36 to 37) S_TKI_TI1_ATR

Describe the text attribute of the S_TKI.

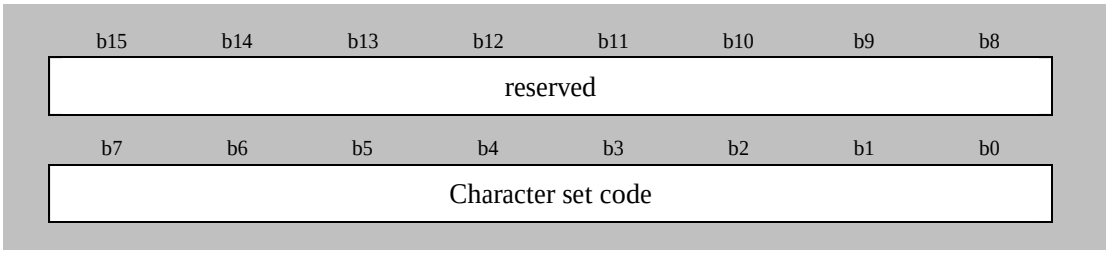


Character set code ... 00h : Text data whose character set code is S_TKI_TI1_ATR does not exist.
01h : ISO646
02h : JISX0201
03h : ISO 8859-1
Others : reserved

©2000 SD Association

(RBP 38 to 39) S_TKI_TI2 _ATR

Describes the text attribute of the S_TKI.

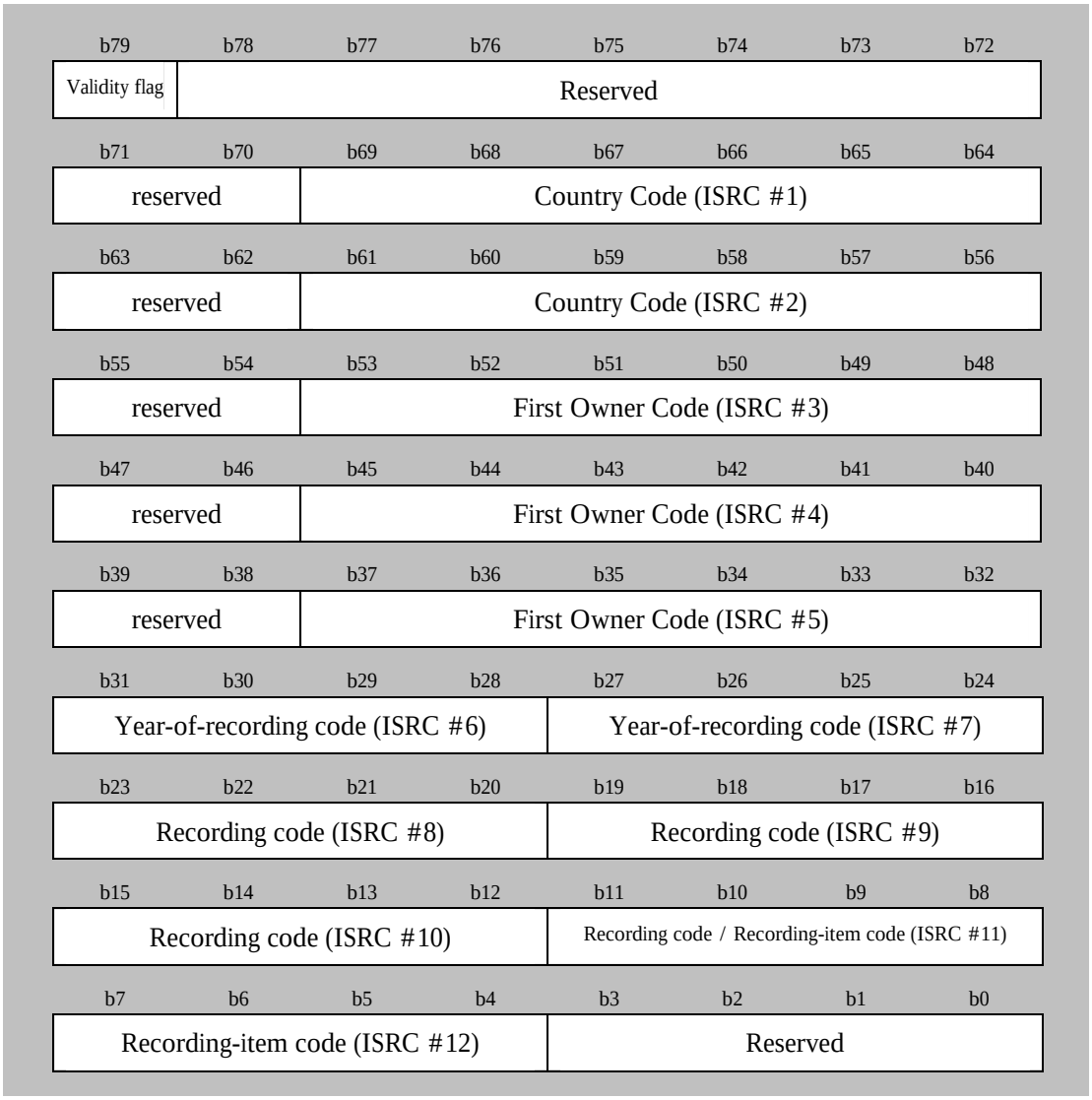


Character set code ... 00h : Text data whose character set code is S_TKI_TI2_ATR does not exist.
81h : Music Shift JIS Kanji
Others : reserved

©2000 SD Association

(RBP 44 to 53) S_TKI_ISRC

S_TKI_ISRC in S_TKGI shall be described in the following format. As for ISRC code, refer to ISO3901 : 1986 **"Documentation-International Standard Recording Code (ISRC). "**



Validity flag ... 0b : ISRC is invalid
1b : ISRC is valid

Note 1 : ISRC #1 to ISRC #5 shall be coded as shown in [Table 4.1-2].
Note 2 : ISRC #6 to ISRC #12 shall be described as the BCD code.

©2000 SD Association

CONFIDENTIAL

◊ **Table 4.1-2: Character Code for ISRC**

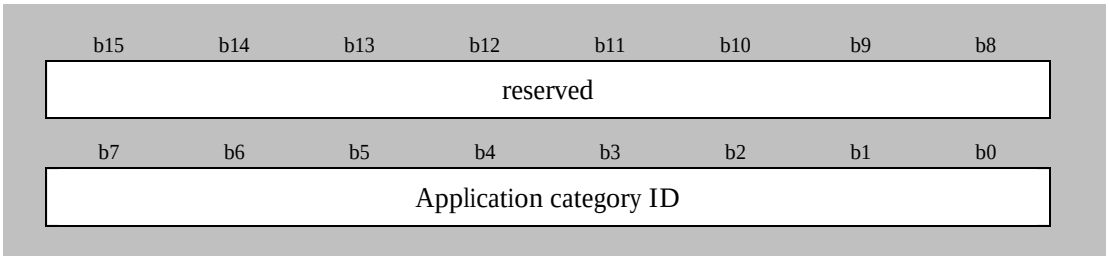
Character	Code	Character	Code	Character	Code
0	000000	C	010011	O	011111
1	000001	D	010100	P	100000
2	000010	E	010101	Q	100001
3	000011	F	010110	R	100010
4	000100	G	010111	S	100011
5	000101	H	011000	T	100100
6	000110	I	011001	U	100101
7	000111	J	011010	V	100110
8	001000	K	011011	W	100111
9	001001	L	011100	X	101000
A	010001	M	011101	Y	101001
B	010010	N	011110	Z	101010

©2000 SD Association

CONFIDENTIAL

(RBP 54 to 55) S_TKI_APP_ATR

Describes an application category ID for the Track.



Application category ID ... 01h : Music
02h : Karaoke
03h : Presentation
04h : Reading Book
Others : reserved

(RBP 68 to 83) S_TKI_FR_ID1

Describes the ID of the content distributor.

(RBP 84 to 99) S_TKI_FR_ID2

Describes the ID of the content distributor.

(RBP 100 to 115) S_TKI_FR_ID3

Describes the ID of the content distributor.

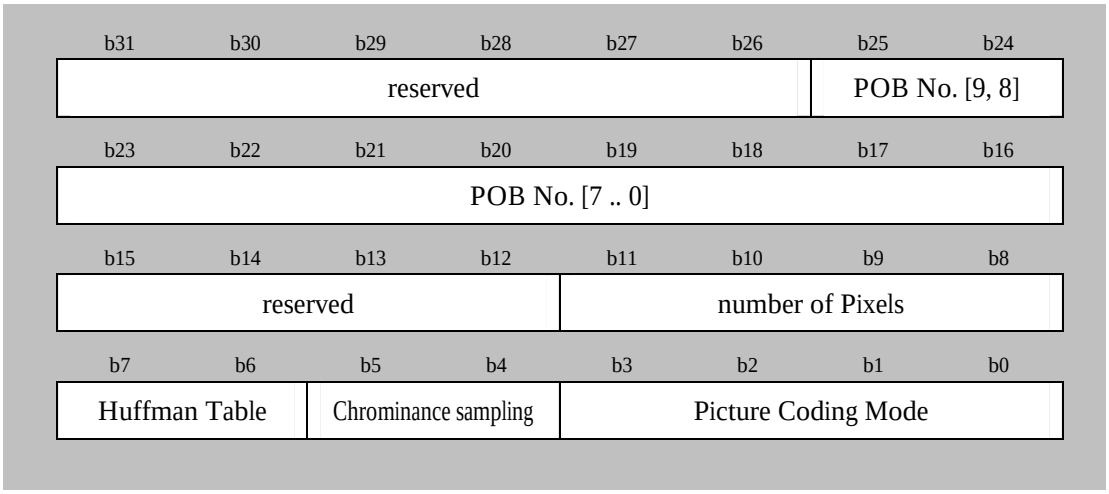
(RBP 116 to 131) S_TKI_FR_ID4

Describes the ID of the content distributor.

©2000 SD Association

(RBP 176 to 255) S_TKI_POB_SRP

Describes the POB Number, which is referred to by the Track and each POB's attribute.
When specifying a POB, use the head S_TKI_POB_SRP first. When not specifying a POB, enter '0'.



POB No. … Describes a number between '1' and '999'.
When no POB exists, enter '000h'.

number of Pixels … 0000b : 96x96 (recommended)
0001b : 640x480 (recommended)
0010b : 160x120 ~ 1800x1200 (optional)
Others : reserved

Huffman Table … 00b : Typical Table
Others : reserved

Chrominance sampling … 00b : 4:2:2
01b : 4:2:0
Others : reserved

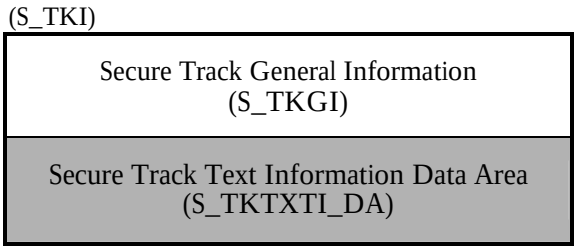
Picture Coding Mode … 0000b : JPEG
Others : reserved

©2000 SD Association

4.2 Secure Track Text Information Data Area (S_TKTXTI_DA)

S_TKTXTI_DA describes a text of S_TKI.
Even if the text does not exist, S_TKTXTI_DA is reserved.
The size of S_TKTXTI_DA is fixed at 256 bytes.

◊ **Figure 4.2-1 : Secure Track Text Information Data Area (S_TKTXTI_DA)**



Each item's text data is described in S_TKTXTI_DA.
In S_TKTXTI_DA, describes text data whose character set code is S_TKI_TI1_ATR, followed by the terminated code, followed by text data whose character set code is S_TKI_TI2_ATR, followed by the terminated code.
When only one type of text data whose character set code is S_TKI_TI1_ATR exists, describes text data followed by the terminated code.
When only one type of text data whose character set code is S_TKI_TI2_ATR exists, describes the terminated code, then text data followed by the terminated code. Each item starts with an item's tag, followed by text data.
It is not necessary to describe all tags. Only tags which contain text data need to be described with their text data.
Text item's tag is shown in [Table 4.2-1]. The description order of these tags is not required to accord with this table.

©2000 SD Association

CONFIDENTIAL

◊ **Table 4.2-1 : Text item's tag in S_TKTXTI_DA**

Text Name	Tag value		Contents
	ISO646 JISX0201 ISO8859-1	Music Shift JIS Kanji	
S_TKTXTI_TTL	01h	0001h	Title
S_TKTXTI_ART	02h	0002h	Artist
S_TKTXTI_ABM	03h	0003h	Album
S_TKTXTI_SW	04h	0004h	Songwriter
S_TKTXTI_CMP	05h	0005h	music writer
S_TKTXTI_ARR	06h	0006h	editor
S_TKTXTI_PRD	07h	0007h	producer
S_TKTXTI_RCD	08h	0008h	Recording company
S_TKTXTI_MSS	09h	0009h	artist's message
S_TKTXTI_UCM	0Ah	000Ah	user's comment
S_TKTXTI_PCM	0Bh	000Bh	provider's comment
S_TKTXTI_CRD	0Ch	000Ch	Date
S_TKTXTI_GNR	0Dh	000Dh	Genre
S_TKTXTI_URL	0Eh	000Eh	URL
S_TKTXTI_FR1	0Fh	000Fh	Free 1
S_TKTXTI_FR2	10h	0010h	Free 2
S_TKTXTI_FR3	11h	0011h	Free 3
S_TKTXTI_FR4	12h	0012h	Free 4
S_TKTXTI_FR5	13h	0013h	Free 5
S_TKTXTI_FR6	14h	0014h	Free 6

<Terminated code>
ISO646, JISX0201, ISO8859-1: '0x00'
Music Shift JIS Kanji: '0x0000'

Each text in 20 items can have variable size; however, the maximum size as a total S_TKTXTI_DA is not to exceed 256 (bytes.)

©2000 SD Association

CONFIDENTIAL

This page is intentionally left blank.

©2000 SD Association

SD Memory Card Specifications

Part 5
PICTURE SPECIFICATIONS

Version 1.0

September 2000

SD Association

CONFIDENTIAL

Conditions for publication

Publisher and Copyright Holder
SD Association

Exemption
None will be liable for any damages from use of this document.

©2000 SD Association.

Copyright Notice

The copyright of "QuickTime File Format Specification" belongs to Apple Computer Inc. QuickTime is a trademark of Apple Computer Inc. This SD-Picture specification is an independent publication and has not been authorized, sponsored, or otherwise approved by Apple Computer, Inc.

License Notice

Apple Computer, Inc. reserves all rights for licensing Apple's software and trademarks relating to QuickTime.

©2000 SD Association.

CONFIDENTIAL

This page is intentionally left blank.

©2000 SD Association.

CONFIDENTIAL

INDEX

Part 5 PICTURE SPECIFICATIONS

1. General..... 9
1.1. Scope..... 9
1.1.1. Relationship among SD Memory card specifications..... 9
1.2. Normative References..... 10
1.3. Terminology..... 12
1.4. Notations..... 12
1.5. Outline of this specification..... 13
1.5.1. Files to be specified in this specification..... 13
1.5.2. Directory and File name..... 14
1.5.3. Interoperability among the SD-Picture-Compatible Equipment..... 16
1.5.3.1. Scope of SD-Picture specification..... 16
1.5.3.2. Details of playback compatibility..... 18
2. Still Photographic Image File Format..... 23
3. Motion-JPEG Picture Format..... 24
3.1. Structure of Motion-JPEG picture file..... 24
3.2. Basic structure of compressed image data..... 24
3.2.1. Compressed image data in QuickTime file format..... 24
3.2.2. Compressed image data in AVI file format..... 25
4. Sound File Format..... 26
5. Composite File Format of Picture and Sound..... 27
5.1. Structure of Composite File Format..... 27
6. Other File Formats Related to Picture Files..... 28
6.1. Print Order File..... 28
Annex A Outline and Examples of QuickTime File Format..... 29
A.1 File Format Example for Motion-JPEG Picture..... 30
A.1.1 Movie data atoms..... 32
A.1.2 Movie atom..... 32
A.2 File Format Example for Still Photographic Image with Sound..... 33
A.2.1 Movie data atom..... 35
A.2.2 Movie atom..... 35
A.3 File Format Example for Motion-JPEG Picture with Sound..... 37
A.3.1 Movie data atom..... 39
A.3.2 Movie atom..... 40

©2000 SD Association.

CONFIDENTIAL

Annex B Outline of DPOF specification 41
B.1 Directory and File Name of DPOF specification 41
B.2 Functions 42

©2000 SD Association.

CONFIDENTIAL

List of Figures

Figure 1.1.1-1: Structure of SD-Picture Specifications 9
Figure 1.5.2-1: Directory and File Names of SD-Picture Files 15
Figure 3.2.1-1: The basic structure of compressed frame data (JPEG) 24
Figure 3.2.1-2: The basic structure of compressed frame data (JFIF) 25

Figure A- 1: The structure of atom 29
Figure A- 2: The structure example of a QuickTime file format 30
Figure A- 3: The overall structure example of QuickTime file format 31
Figure A- 4: The layout example of a movie data atom 32
Figure A- 5: The layout example of a movie atom 32
Figure A- 6: The structure example of a QuickTime file format 33
Figure A- 7: The overall structure example of QuickTime file format 34
Figure A- 8: The layout example of a movie data atom 35
Figure A- 9: The layout example of a movie atom 36
Figure A- 10: The structure example of a QuickTime file 37
Figure A- 11: The overall structure example of QuickTime file format 38
Figure A- 12: The layout example of a movie data atom 39
Figure A- 13: The layout example of a movie atom 40

Figure B- 1: Directory and File Name of DPOF specification 41

©2000 SD Association.

CONFIDENTIAL

List of Tables

Table 1.5.3-1: Scope of SD-Picture specification 16
Table 1.5.3-2: The relationship between a requirement level and playback compatibility 17
Table 1.5.3-3: The categories of SD-Picture-compatible equipment and the typical products 18
Table 1.5.3-4: Specifications according to categories of equipment 19
Table 1.5.3-5: The relationship between "categories of equipment", "function" in SD-Picture specification and "classification of function" in DCF standard 20
Table 1.5.3-6: The range of pixel number (DCF standard) 20
Table 1.5.3-7: The pixel ranges of "Equipment with playback function" in SD-Picture 20
Table 4-1: Sound file format and requirement level 26

Table A- 1: The field of atom 29
Table A- 2: Atoms contained in a QuickTime file 30

Table B- 1: Functions of DPOF specification 42

©2000 SD Association.

CONFIDENTIAL

1. General

1.1. Scope

This part specifies the application format of the SD-Picture files that are recorded in the SD memory card. The SD-Picture files consist of still photographic images, Motion-JPEG pictures and the files related to those picture files such as sound files, print order files. The card of which the data is defined in this specifications shall comply with **"Part1. PHYSICAL SPECIFICATIONS", "Part2. FILE SYSTEM SPECIFICATIONS "** and **"Part3. SECURITY SPECIFICATIONS "** of SD Memory Card specifications.

1.1.1. Relationship among SD Memory card specifications

Figure 1.1.1-1depicts the relationship between this part and the other parts of SD Memory Card Specifications.

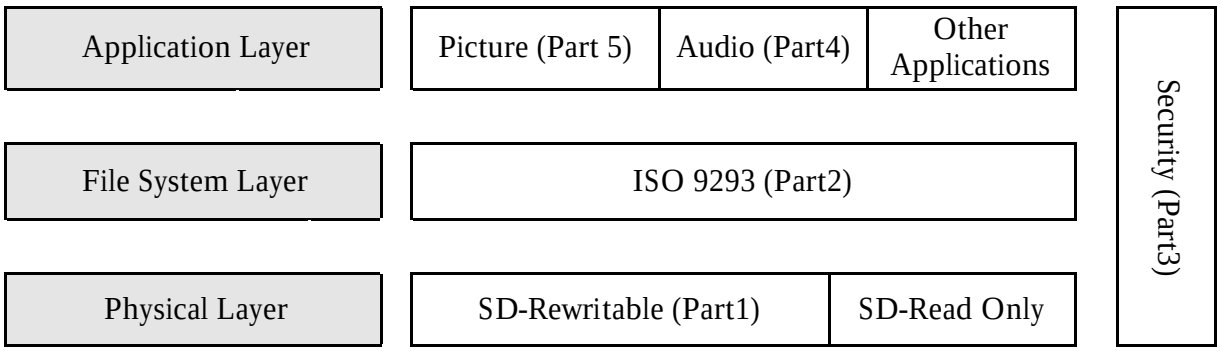


Figure 1.1.1-1: Structure of SD-Picture Specifications

©2000 SD Association.

CONFIDENTIAL

1.2. Normative References

1) JEIDA DCF Version 1.0: 1998

Design rule for Camera File System Version1.0, JEIDA-49-2-1998, Japan Electronic Industry Development Association.

The contact address for the distribution of DCF standard is shown at the following URL.

(http://www.jeida.or.jp/document/standard/index-e.html
or http://www.jeida.or.jp/document/standard/index.html)

2) JEIDA Exif Version 2.1: 1998

Digital Still Camera Image File Format Standard (Exchangeable image file format for Digital Still Camera: Exif) Version 2.1, JEIDA-49-1998, Japan Electronic Industry Development Association.

3) DPOF: 1998

Digital Print Order Format Specification Version 1.00, October 26, 1998, CANON INC., Eastman Kodak Company, Fuji Photo Film Co., Ltd.,and Matsushita Electric Industrial Co., Ltd.

The contact address for the distribution of DPOF specification is shown at the following URL.

(http://www.panasonic.co.jp/avc/video/dpof/index.htm)

4) ITU-T Recommendation T81: 1992, ISO/IEC 10918-1: 1994

Information technology - Digital compression and coding of continuous-tone still pictures - requirements and guidelines

5) SD Memory Card Specifications
Part 1: PHYSICAL SPECIFICATIONS

6) SD Memory Card Specifications
Part 2: FILESYSTEM SPECIFICATIONS

7) SD Memory Card Specifications
Part 3: SECURITY SPECIFICATIONS

8) SD Memory Card Specifications
Part4: AUDIO SPECIFICATIONS

©2000 SD Association.

CONFIDENTIAL

9) QuickTime File Format Specification, May 1996
(http://www.apple.com/quicktime/resources/qtfileformat.pdf)

10) SDMI Portable Device Specification Part1 Version 1.0,July 1999
(http://www.sdmi.org/public_doc
/pdwg99070802-Specification1.0.pdf)

©2000 SD Association.

1.3. Terminology

Some terminology in this specification is the same as DCF standard. The following list is shown for convenience.

DCF standard	Design rule for Camera File system Standard, Version 1.0 of the Japan Electronic Industry Development Association (JEIDA). ISO 12234-3 Photography - Electronic still picture imaging - Design rules for camera file system (DCF)
Exif standard	Digital Still Camera Image File Format Standard, Version 2.1 of the Japan Electronic Industry Development Association (JEIDA).
JPEG standard	ISO/IEC 10918-1 ITU-T Recommendation T81 information technology - Digital compression and coding of continuous-tone still images - Requirements and guide-lines
DPOF specification	Digital Print Order Format specification Version 1.00
Main image	The primary data of the image
Thumbnail	A small version of the main image, used for indexing
DCF image root directory	The directory directly under the root directory, created in accord with the DCF directory rules.
DCF directory	A directory under the DCF image root directory created in accord with the DCF directory rules, for storing images.
DCF directory name	A directory name assigned in accord with the DCF directory naming conventions.
DCF object	A group of files recorded in accord with DCF.
DCF file name	A file name assigned in accord with the DCF file naming conventions.
DCF basic file	An image file stored directly under a DCF directory, having a DCF file name and the extension "JPG" and having the DCF-stipulated data structure, based on the Exif standard.
DCF basic main image	An Exif primary image included in a DCF basic file.
DCF basic thumbnail	An Exif thumbnail image included in a DCF basic file.
DCF extended image file	An image file stored directly under a DCF directory, having an extension and data structure different form a DCF basic file
DCF thumbnail file	A compressed file for storing the thumbnail image of a DCF extend image file

1.4. Notations

Numbers in decimal notation are represented by decimal digits, namely 0 to 9. Digit(s) represents any digit from digit 0 to 9.

Numbers in Hexadecimal notation are represented as a sequence of one or more hexadecimal digits namely 0 to 9 and A to F, suffixed by "h".

©2000 SD Association.

1.5. Outline of this specification

1.5.1. Files to be specified in this specification

(1) Still Photographic Image File Format

Still photographic image files shall comply with JEIDA DCF standard as described in Chapter 2 of this specification.

(2) Motion-JPEG Picture File Format

Motion-JPEG picture files are permitted to record with QuickTime file format or AVI file format. Some additional specification is defined in Chapter 3 of this specification.

(3) Sound File Format

Sound files are permitted to record with Linear PCM sound data format or μlaw sound data format specified in JEIDA Exif 2.1 standard as described in Chapter 4 of this specification.

(4) Composite File Format of Picture and Sound

Composite files that include picture data and sound data to be reproduced simultaneously are permitted to record with QuickTime file format or AVI file format as described in Chapter 5 of this specification.

The composite file may be still photographic image with sound or Motion-JPEG picture with sound. A DCF object which consists of a DCF basic file and a WAVE file specified in Chapter 4 is also permitted as file format of still photographic images with sound.

(5) Other File Format Relating the Picture Files

Other files relating the picture files are recommended to comply with Chapter 6 of this specification.

(5-1) Print Order File

The file to specify the picture file(s) to be printed shall comply with DPOF specification as described in Chapter 6.1.

©2000 SD Association.

1.5.2. Directory and File name

The structure of directory and file names shall fully comply with a JEIDA DCF standard.

The DCF standard is distributed from Japan Electronic Industry Development Association.
(http://www.jeida.or.jp)

The contact address for the distribution of DCF standard is shown at the following URL.
(http://www.jeida.or.jp/document/standard/index-e.html
or http://www.jeida.or.jp/document/standard/index.html)

DCF extended image files such as Motion-JPEG picture files, and sound data files shall be stored under DCF directories.

An example of directory and file name of SD-Picture files is shown in Figure 1.5.2-1.

©2000 SD Association.

CONFIDENTIAL

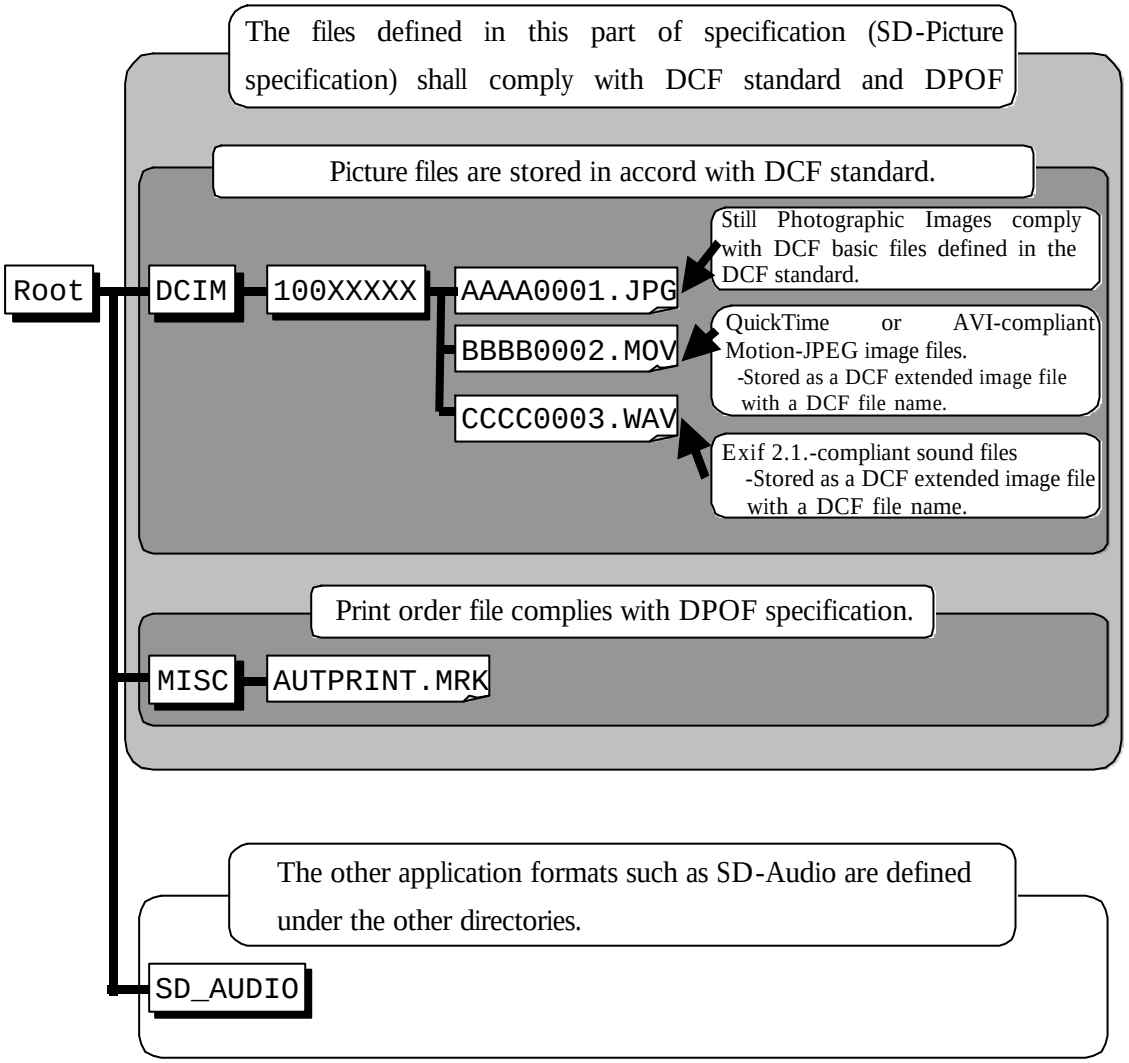


Figure 1.5.2-1: Directory and File Names of SD-Picture Files

©2000 SD Association.

CONFIDENTIAL

1.5.3. Interoperability among the SD-Picture-Compatible Equipment

1.5.3.1. Scope of SD-Picture specification

The SD-Picture specification defines picture file formats, mainly recorded by DSC (Digital Still Camera) , DVC (Digital Video Camera) and so on.

Types of recording data, file formats and requirement level of the file formats in SD-Picture specification are shown in Table 1.5.3-1.

Table 1.5.3-1: Scope of SD-Picture specification

Type of data	File format	Requirement level
Still photographic image	DCF basic file	Mandatory
Motion-JPEG picture	QuickTime and AVI	Option
Sound	WAVE File (compliant with Exif Ver.2.1)	Option
Still photographic image with sound	{DCF basic file + WAVE File } or { DCF basic file + QuickTime} or { DCF basic file + AVI}	Conditional Mandatory(When a still photographic image with sound is recorded, the DCF basic file shall be recorded simultaneously.)
Motion-JPEG picture with sound	QuickTime and AVI	Option
Print order file	DPOF	Conditional Mandatory(When a print order file is recorded, that shall be compliant with DPOF specification.)

©2000 SD Association.

The relationship between the requirement level in the SD-Picture specification and playback compatibility of equipment is shown in Table 1.5.3-2.

SD-Picture-compatible equipment shall play DCF basic files to ensure playback compatibility of still photographic images.

Table 1.5.3-2: The relationship between a requirement level and playback compatibility

Requirement level	Types of data	Playback compatibility	Remarks
Mandatory	Still Photographic image (DCF basic file)	Compatible	A DCF basic main image shall be played for playback compatibility. If a DCF basic main image is not played, a DCF basic thumbnail file shall be played.
Conditional Mandatory	-Still Photographic image with sound -Print order file	Compatible among products equipped with the same function	-For playback compatibility, a still photographic image shall be recorded with JPEG format compliant with DCF standard. -A print order file shall be compliant with DPOF specification.
Option	-Motion-JPEG picture -Still photographic images and so on except for JPEG image compliant with DCF standard -Sound	Not Compatible	Recording or reproduction is permitted.

Other files which are permitted in the DCF standard and are not described in Table 1.5.3-2, are not compatible among SD-Picture-compatible equipment.

The other files which are prohibited in the DCF standard, are prohibited in SD-Picture specification.

©2000 SD Association.

1.5.3.2. Details of playback compatibility

This chapter describes the details of the playback compatibility specified in the SD-Picture specification.

(1) Still Photographic Image Files

The still photographic image file in the SD-Picture specification shall be essentially treated in a same manner specified in DCF standard Ver.1.0.

Since detailed parts of the SD-Picture specification are different from the DCF standard, the different points are mentioned as follows.

If the SD-Picture specification differs from DCF standard, priority shall be normally given to SD-Picture specification in any case.

-- Classification of SD-Picture-compatible equipment --

In DCF standard, the functions of DCF-compatible equipment are classified into Writer, Reader1, and Reader2 functions.

In the SD-Picture specification, SD-Picture-compatible equipment are classified according to categories of equipment. The functions of playback and recording are specified.

The categories of SD-Picture-compatible equipment and the typical products are shown in Table 1.5.3-3.

Table 1.5.3-3: The categories of SD-Picture-compatible equipment and the typical products

Categories of equipment	Typical products
Equipment with recording and playback functions	DSC, DVC, scanner, etc.
Equipment with playback function	Printer, player, print service, etc.
Equipment with playback function of small-size image	Cellular phone, PDA, TV, etc.

©2000 SD Association.

-- Specifications regarding required functions of SD-Picture-compatible equipment --

According to categories of equipment, SD-Picture-compatible equipment shall have playback and recording functions shown in Table 1.5.3-4 to ensure playback compatibility for a still image.

Table 1.5.3-4: Specifications according to categories of equipment

<table>
<tr><th rowspan="2">Categories of equipment</th><th rowspan="2">Function</th><th colspan="2">Types of still image files</th></tr>
<tr><th>DCF standard-compliant JPEG files (=DCF basic file)</th><th>Image files except for JPEG files (TIFF, etc.)</th></tr>
<tr><td rowspan="2">Equipment with recording and playback functions (DSC, DVC, etc.)</td><td>Playback</td><td>-DCF basic files shall be played. Even though a DCF basic main image cannot be played, the corresponding DCF basic thumbnail shall be playable.
-DCF basic main image size that shall be playable is not defined.</td><td rowspan="5">Not specified
(1)Playback and recording of image files except for JPEG files follows the specifications for individual equipment.
(2)When an image file except for JPEG files is recorded, a DCF basic file which has the same pixel size or a thumbnail file of 160*120pixels may be recorded simultaneously.</td></tr>
<tr><td>Recording</td><td>DCF basic files shall be recorded.</td></tr>
<tr><td rowspan="2">Equipment with playback function (printer, etc.)</td><td>Playback</td><td>-DCF basic main image that is less than 2560*1920pixels shall be played.
-Playback of DCF basic thumbnail shall follow the specifications for individual equipment.</td></tr>
<tr><td>Recording</td><td>-It is not necessary to have the recording function.
-If the equipment can capture still images and record them, DCF basic files shall be recordable.</td></tr>
<tr><td rowspan="2">Equipment with playback function of small size image (cellular phone, etc.)</td><td>Playback</td><td>-DCF basic files shall be played. Even though a DCF basic main image cannot be reproduced, the corresponding DCF basic thumbnail shall be playable.
-DCF basic main image size that shall be playable is not defined.</td></tr>
<tr><td>Recording</td><td>-It is not necessary to have the recording function.
-If the equipment can capture still images and record them, DCF basic files shall be recordable.</td></tr>
</table>

-Playback of JPEG files except for DCF basic files shall follow the specifications for individual equipment.
-Recording of JPEG files, except for DCF basic files, under DCF directory is prohibited.

The categories of equipment and the functions shown in Table 1.5.3-4 have following relationship shown in Table 1.5.3-5 with the three classifications ("Writer","Reader1","Reader2") of the function in DCF standard.

©2000 SD Association.

Table 1.5.3-5: The relationship between "categories of equipment", "function" in SD-Picture specification and "classification of function" in DCF standard

"Categories of equipment" and "function" in the SD-Picture specification	"Classification of function" in the DCF standard
Recording function of "Equipment with recording and playback functions"	"Writer" (Writer specification (Ver.1.0 Chapter4))
Playback function of "Equipment with recording and playback functions" or "Equipment with playback function of small size image"	"Reader1" (Reader1 specification (Ver.1.0 Chapter5))
Playback function of "Equipment with playback function"	"Reader2" (Reader2 specification (Ver.1.0 Chapter6))

However, a supported pixel count specification is defined in DCF standard. The supported pixel count specification indicates the maximum and minimum pixel number of a DCF basic main image which shall be playable on a Reader2 device shown in Table 1.5.3-6. In the SD-Picture specification, the supported pixel count specification is shown in Table 1.5.3-7.
"Equipment with playback function of small size image" is not defined in DCF standard. This is unique to the SD-Picture specification.

Table 1.5.3-6: The range of pixel number (DCF standard)

	Pixel Ranges				
	min				max
Horizontal pixel range	160	<=	X	<=	1800
			and		
Vertical pixel range	120	<=	Y	<=	1200

(pixels)

Table 1.5.3-7: The pixel ranges of "Equipment with playback function" in SD-Picture

	Pixel Ranges				
	min				max
Horizontal pixel range	160	<=	X	<=	2560
			and		
Vertical pixel range	120	<=	Y	<=	1920

(pixels)

Based on the definition related to the requirements for still image files above, the playback compatibility is ensured to DCF basic files whose pixel is up to 2560*1920 pixels by using main images or thumbnails.

©2000 SD Association.

However, in the SD-Picture specification there is a possibility of increasing a number of required playback pixels for the equipment without thumbnail playback functionality. It is recommended that the equipment with/without playback function treat the main image which has as many number of pixels as possible to the specifications for individual equipment.

-- Playback priority for images in a DCF object --

If a DCF basic file and other files with DCF file name (for example, DCF extended image file) exist in the same DCF object, the playback priority shall follow the specifications for individual equipment. For example, when a still photographic image with sound (see Chapter5) are recorded with QuickTime format, a DCF basic file shall be recorded simultaneously (in this case, a QuickTime format file is the other file in a DCF object). The playback priority of a QuickTime format file and a DCF basic file shall follow the specifications for individual equipment.

-- Specification for recording other files except for JPEG files --

When a DCF basic file and other files which organize a DCF object are recorded, both of the files shall be recorded so that they can become the substantially same image after playback.

"The substantially same image" means that a remarkable difference is not bringing about to the image quality of both files. For example, when a compressed image file (ex. DCF basic file) and uncompressed image file (ex. TIFF) are recorded, images of the same number of pixels are recorded in order to maintain image quality. Furthermore, in the case of recording a still photographic image with sound using QuickTime format (see Chapter5), the number of pixels of a DCF basic file to be recorded simultaneously shall be equal to that of a QuickTime file in order to maintain image quality.

By establishing such a specification, compatibility between the equipment which offer a playback priority to a DCF basic file and the other equipment which offer the playback priority to the other file is intended.

Furthermore, all DCF objects, that is all DCF basic files, DCF extended image files, DCF thumbnail files and other files with DCF file names, are deleted, moved, and copied in object units same as the DCF standard.

(2) Print Order Files

It is recommended that SD-Picture-compatible equipment have the function of handling DPOF file. If SD-Picture-compatible equipment have a function of reproducing, recording, and/or editing the print order file, that print order file shall comply with DPOF specification and the other print order file not compliant with DPOF is prohibited.

©2000 SD Association.

(3) Other Files Specified in This Specification

SD-Picture-compatible equipment may have the function of reproducing, recording, and/or editing the other files specified in this specification, such as Motion-JPEG picture files, and sound files. It is recommended that the SD-Picture-compatible equipment handle those files that comply with this specification when that equipment have the function of handling the same kind of files.

©2000 SD Association.

CONFIDENTIAL

2. Still Photographic Image File Format

The still photographic image file is based on JEIDA DCF standard Ver.1.0.

SD-Picture specification defines the additional rules to JEIDA DCF standard Ver.1.0.
The aim of the additional rules are
(1) to establish the interoperability of DCF basic files, and
(2) to expand the application field of SD memory card.

One of the additional rules is that all the DCF basic files under DCF directories are required to be playable.
The other additional rules are described in Chapter 1.

©2000 SD Association.

3. Motion-JPEG Picture Format

3.1. Structure of Motion-JPEG picture file

The SD-Picture specification defines Motion-JPEG picture file format. QuickTime file format and AVI file format are permitted as Motion-JPEG picture file format in this specification.

3.2. Basic structure of compressed image data

3.2.1. Compressed image data in QuickTime file format

In the SD-Picture specification, Motion-JPEG picture data consists of a group of compressed frame data in conformance with JPEG baseline DCT format specified in ISO/IEC10918-1. Furthermore, the frame data is permitted to be in conformance with the JFIF (JPEG File Interchange Format).

The basic structure of compressed frame data (JPEG baseline DCT format) is shown in Figure 3.2.1-1, and the basic structure of compressed frame data (JFIF) is shown in Figure 3.2.1-2.

JPEG	Bytes
SOI (FFD8.h)	2
DQT (FFDB.h)	199
DHT (FFC4.h)	420
DRI (FFDD.h)	6
SOF (FFC0.h)	19
SOS (FFDA.h)	14
Compressed Data	variable
EOI (FFD9.h)	2

Figure 3.2.1-1: The basic structure of compressed frame data (JPEG)

©2000 SD Association.

JPEG with JFIF header	Bytes
SOI (FFD8.h)	2
APP0	
DQT (FFDB.h)	199
DHT (FFC4.h)	420
SOF (FFC0.h)	19
SOS (FFDA.h)	14
Compressed Data	variable
EOI (FFD9.h)	2

Figure 3.2.1-2: The basic structure of compressed frame data (JFIF)

DQT, DHT, DRI, and SOF may line up in any order.

3.2.2. Compressed image data in AVI file format

In the SD-Picture specification, Motion-JPEG picture data recorded in AVI file format is a group of compressed frame data in conformance with JPEG format. The details concerned with header structure of JPEG format are not specified.

©2000 SD Association.

4. Sound File Format

If SD-Picture-compatible equipment can record sound data, the data files are permitted to be in conformance with the WAVE File format specified in Exif Ver.2.1.
In the case where a WAVE file is recorded, the data streams shall be either of the following formats.

- Linear PCM
- u-Law PCM (conforming to ITU-T G.711)

Furthermore, the bit size shall be 8 bits or 16 bits. Only 8 bits shall be used for u-Law PCM in conformance with ITU-T G.711.
The recording of Exif-specific chunks for Exif-specific attributes is optional in the SD-Picture Specifications.

Although an SDMI (Secure Digital Music Initiative)-compliant product may accept analog input from a microphone embedded in this product, the SDMI Specifications restrict this input to band-limited (-3 dB at 100 Hz and -60 dB at 8 kHz) mono voice-grade. Therefore, input shall be restricted to monaural and band-limited (-3 dB at 100 Hz and -60 dB at 8 kHz) sound for the time being. In addition, a sampling frequency shall be either of 8.000 kHz or 11.025 kHz. Sound recording with a sampling frequency of 16 kHz or higher is prohibited.
These restrictions on sound data recording are applicable to still photographic images with sound and Motion-JPEG pictures with sound to be described in Chapter 5.

Table 4-1: Sound file format and requirement level

Function	Requirement level	Sample frequency	Bit size	Channels	Data structure
Playback and Record	Option	8kHz or 11.025kHz	8 or 16	1	Linear PCM
		8kHz	8	1	u-Law PCM(*)

(*) Only 8kHz shall be used for u-Law PCM in conformance with ITU-T G.711.

The greatest possible caution to the SDMI Specifications with regard to a product configuration should be exercised when an SDMI-compliant product capable of SD-Audio playback records the sound data defined in this chapter.

©2000 SD Association.

5. Composite File Format of Picture and Sound

5.1. Structure of Composite File Format

In this chapter, still photographic images with sound and Motion-JPEG pictures with sound are specified as composite file format of pictures and sound.

In SD-Picture specification, QuickTime file format, AVI file format and a DCF object which consists of a DCF basic file and a WAVE file specified in Chapter 4 are permitted as file format of still photographic images with sound.
The still photographic image in QuickTime file format is compressed image data in conformance with JPEG baseline DCT format specified in ISO/IEC10918-1, JFIF (JPEG File Interchange Format) or Exif standard.
In case of AVI file format, the still photographic image is in conformance with JPEG format. The details concerned with header structure of the JPEG format are not specified.
When a still photographic image with sound is recorded in QuickTime file format or AVI file format, a DCF basic file shall be recorded so as to form a DCF object with the QuickTime file or the AVI file to maintain the playback compatibility of still photographic image.
If a DCF basic file and a WAVE file are played simultaneously, the WAVE file included in a DCF object containing the DCF basic file shall be played first.

In SD-Picture specification, QuickTime file format and AVI file format are permitted as a file format of Motion-JPEG pictures with sound.
The Motion-JPEG picture data in QuickTime file format and AVI file format is the same as the one specified in Chapter 3.
When Motion-JPEG pictures with sound are recorded in QuickTime file format or AVI file format, the recording of a DCF basic file or a thumbnail file is optional.

The method of sound data encoding is the same as the one specified in Chapter 4 "Sound File Format".

©2000 SD Association.

6. Other File Formats Related to Picture Files

6.1. Print Order File

The print order file is a control file which directly specifies pictures and the number of sheets to be automatically printed out.

The print order file is fully compliant with DPOF specification.

©2000 SD Association.

CONFIDENTIAL

Annex A Outline and Examples of QuickTime File Format

As shown in "QuickTime File Format Specification", QuickTime file consists of so-called atoms, which have the structure shown in Figure A- 1. Table A- 1 shows the type and example value of each field of atom.

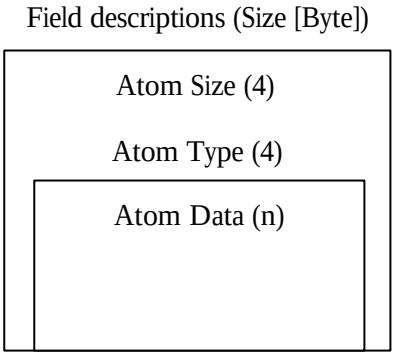


Figure A- 1: The structure of atom

Table A- 1: The field of atom

	Field type	Example of value
Atom Size	32bit-integer (Big Endian)	n+8 (n = size of atom data)
Atom Type	32 bits (typically four ASCII characters)	' mdat '
Atom Data	Data structure which may contain child atoms	Video data stream

In this chapter, examples of QuickTime file Format for a Motion-JPEG picture, a still photographic image with sound and a Motion-JPEG picture with sound in SD-Picture specification are shown.

©2000 SD Association.

Annex A
Outline and Examples of QuickTime File Format

A.1 File Format Example for Motion-JPEG Picture

This chapter shows an example of the file format of a Motion-JPEG picture in QuickTime file format.

Figure A- 2 shows the structure example of QuickTime file format for a Motion-JPEG picture in SD-Picture specification.

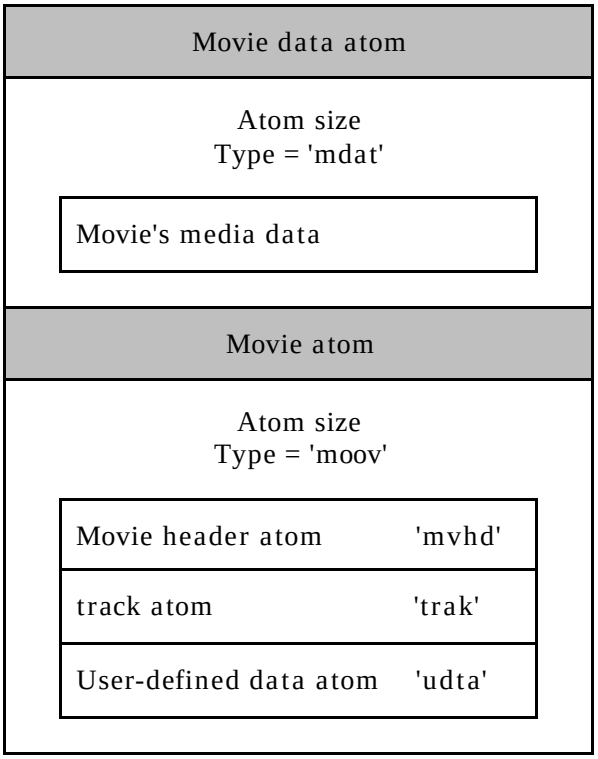


Figure A- 2: The structure example of a QuickTime file format (Motion-JPEG picture)

Table A- 2: Atoms contained in a QuickTime file

Name	Type	Description
movie data atom	mdat	a container for an atom header (atom size and type field) and a movie's media data
movie atom	moov	a container for the information that describes a movie's data

The overall structure example of QuickTime file format for a Motion-JPEG picture in SD-picture specification is shown in Figure A- 3.

©2000 SD Association.

CONFIDENTIAL

Annex A
Outline and Examples of QuickTime File Format

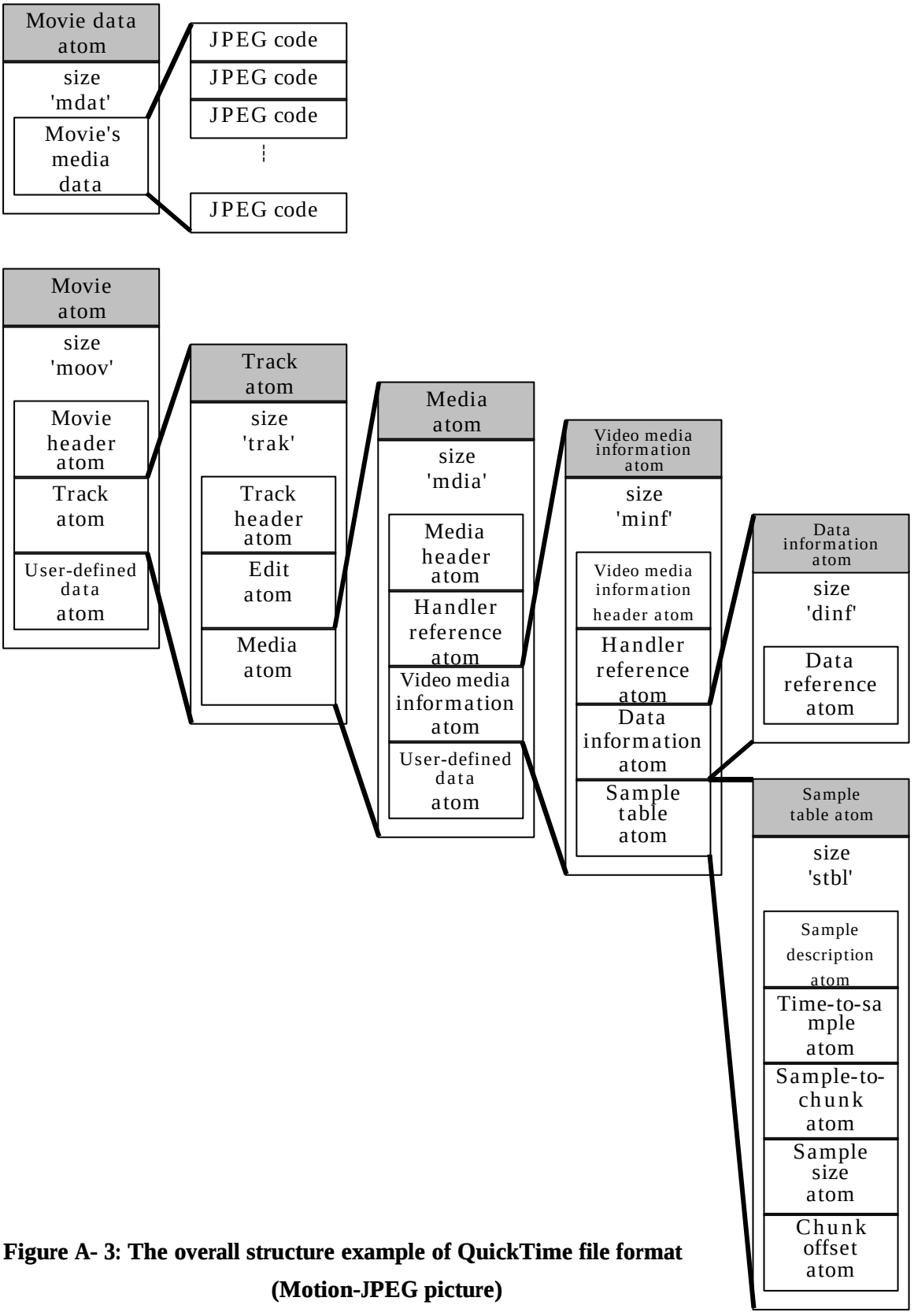


Figure A- 3: The overall structure example of QuickTime file format (Motion-JPEG picture)

©2000 SD Association.

CONFIDENTIAL

A.1.1 Movie data atoms

The movie data atom consists of an atom header (atom size and type field) and movie's media data. The layout example of a movie data atom is shown in Figure A- 4.

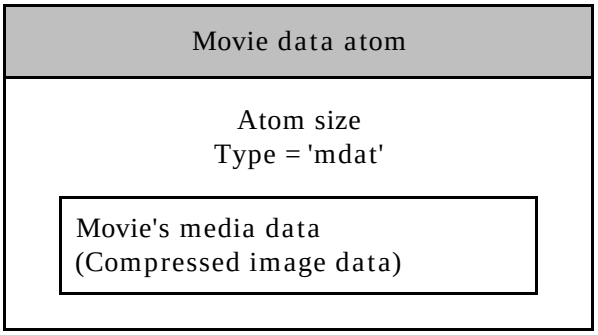


Figure A- 4: The layout example of a movie data atom

The compressed image data is a sequence of frame data. The data structure of each frame is in conformance with JPEG baseline DCT format specified in ISO/IEC10918-1 or JFIF (JPEG File Interchange Format) as described in Chapter 3.2.1. In a QuickTime file, the data format name (within the sample description) is 'jpeg', which declares the format of the sample data as Photo-JPEG images.

A.1.2 Movie atom

A movie atom has an atom type of 'moov'. This atom acts as a container for the information that describes a movie's data. This information, or meta-data, is stored in a number of different types of atoms.

Figure A- 5 shows the layout example of a movie atom.

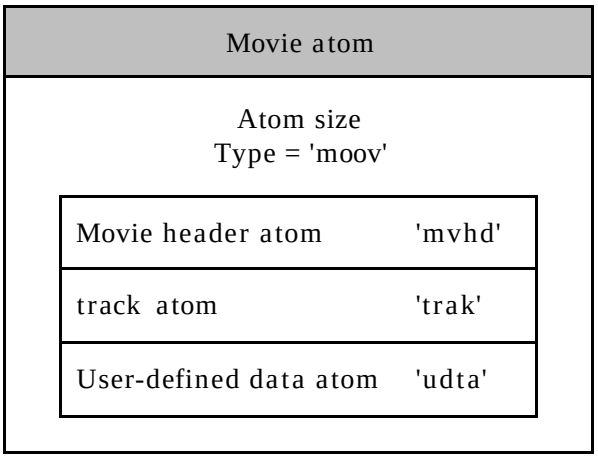


Figure A- 5: The layout example of a movie atom

©2000 SD Association.

A.2 File Format Example for Still Photographic Image with Sound

This chapter shows an example of the file format of a still photographic image with sound in QuickTime file format.

Figure A- 6 shows the structure example of a QuickTime file format for a still photographic image with sound.

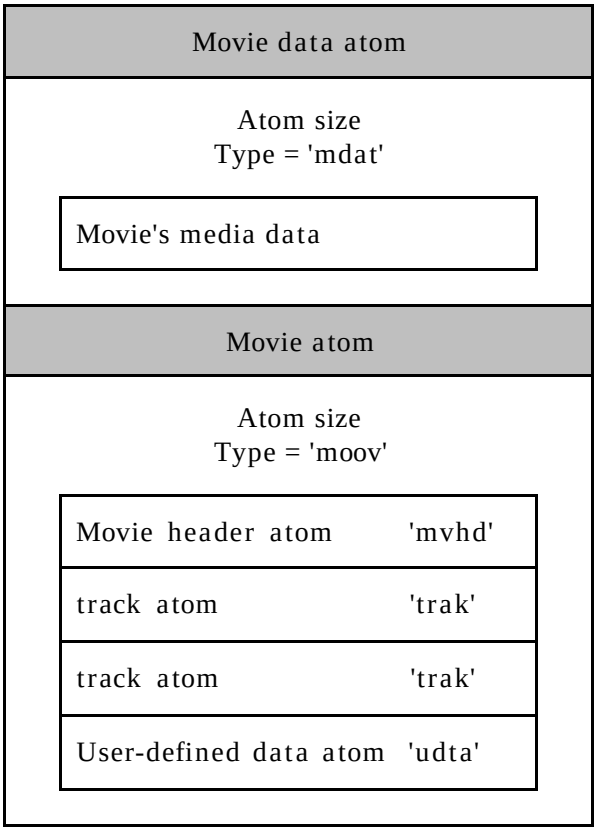


Figure A- 6: The structure example of a QuickTime file format (Still Photographic Image with Sound)

The basic structure of a still photographic image with sound in QuickTime file format is the same as that of Motion-JPEG picture file format shown in Chapter A.1. However, the difference between the two is that the movie data atom and the movie atom shown in Figure A- 6 include sound data and additional information for sound data playback.

The overall structure example of QuickTime file format for a still photographic image with sound in SD-picture specification is shown in Figure A- 7.

©2000 SD Association.

Annex A
Outline and Examples of QuickTime File Format

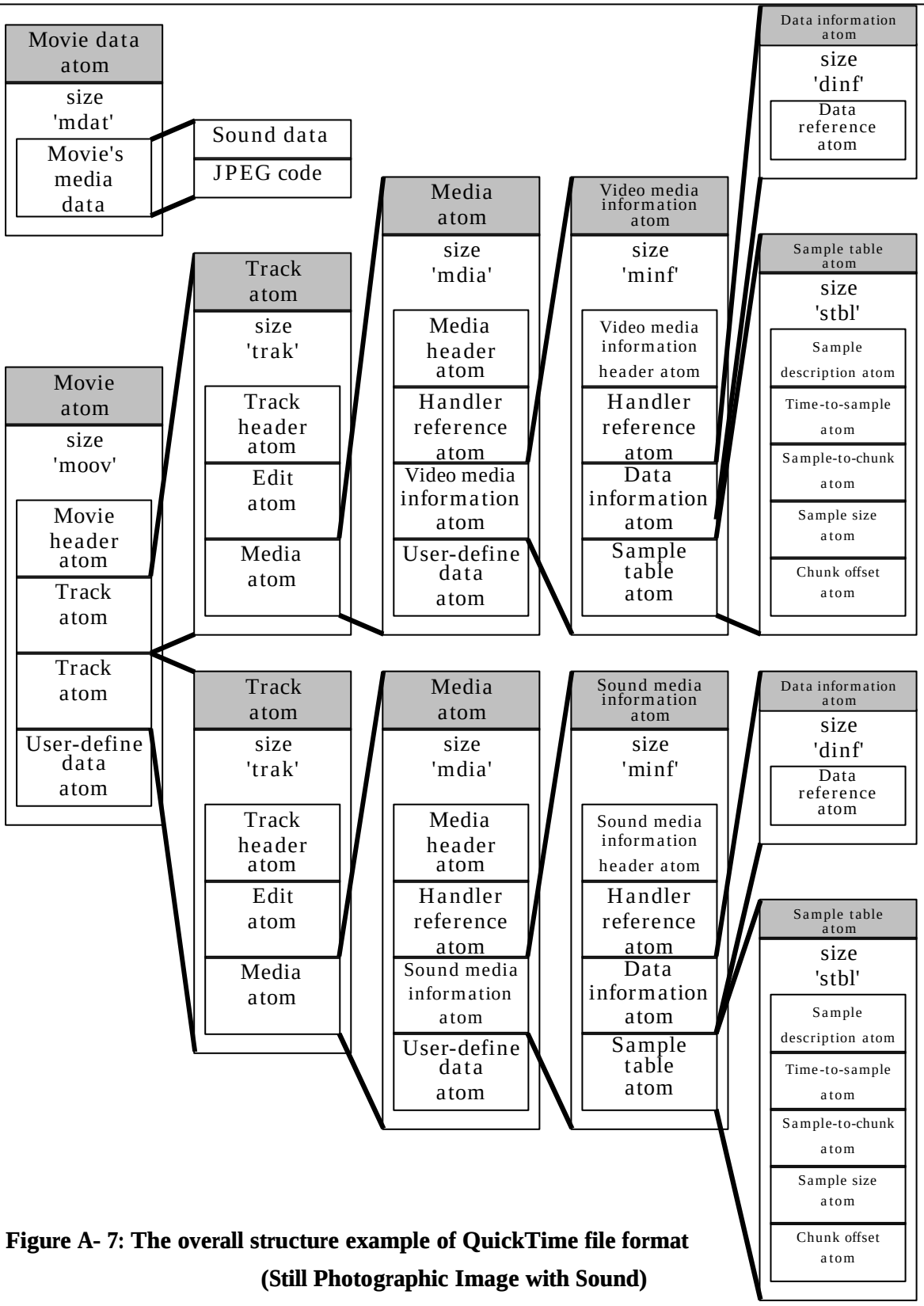


Figure A- 7: The overall structure example of QuickTime file format (Still Photographic Image with Sound)

©2000 SD Association.

CONFIDENTIAL

A.2.1 Movie data atom

A movie data atom is a container for an atom header (atom size and type field) and the movie's media data (sound data and compressed image data). The layout example of a movie data atom is shown in Figure A- 8.

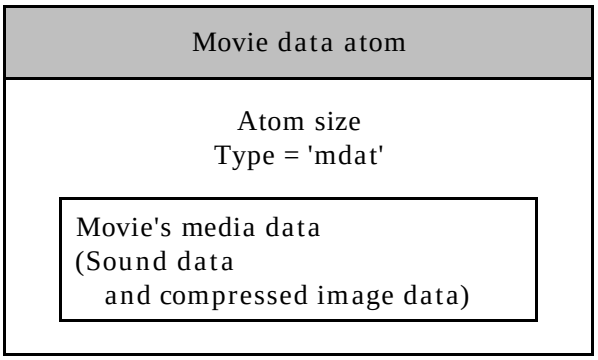


Figure A- 8: The layout example of a movie data atom

The sound data formats are as follows (cf. Chapter 4 & Chapter 5.1).

-Linear PCM (bit size : 8 bits or 16 bits; sampling frequency : 8kHz or 11.025kHz)

In a QuickTime file, the data format name (within the sample description) is 'twos' for 16-bit data and 'raw ' for 8-bit uncompressed audio data.

- u-Law PCM (bit size : 8 bits; sampling frequency : 8kHz)(*)

(*)conforming to ITU-T G.711

In a QuickTime file, the data format name (within the sample description) is 'ulaw', which declares the format of the sample data as U-Law 2:1 compressed audio.

The compressed image data is in conformance with JPEG baseline DCT format specified in ISO/IEC10918-1, JFIF (JPEG File Interchange Format) or Exif standard as described in Chapter 5.1.

A.2.2 Movie atom

This chapter shows an example of a movie atom in a still photographic image with sound in QuickTime file format.

A movie atom in a still photographic image with sound described in QuickTime file format includes a track atom for sound data as well as a track atom for still photographic image data.

A track atom for sound data has different structure from that for still photographic image data: the structure of a sound media information atom under a media atom and of a sample description atom under a sample table atom.

To be more specific, a sound media information atom includes a sound media information header

©2000 SD Association.

atom (‘smhd’), and a sample description table in a sample description atom has different structure.

The layout example of a movie atom in a still photographic image with sound in QuickTime file format is shown in Figure A- 9.

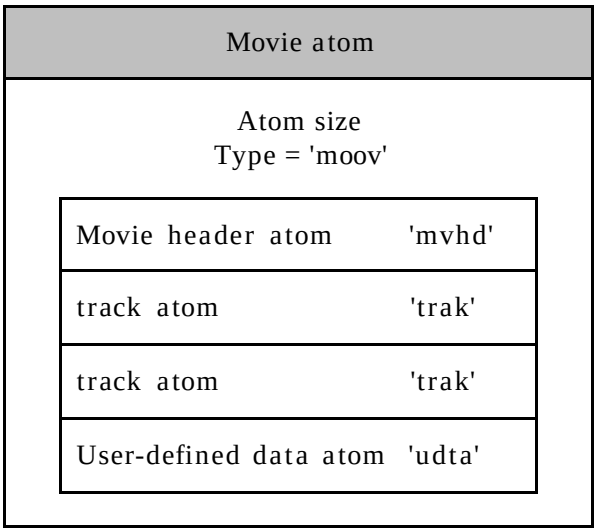


Figure A- 9: The layout example of a movie atom

©2000 SD Association.

A.3 File Format Example for Motion-JPEG Picture with Sound

This chapter shows an example of the file format of a Motion-JPEG picture with sound in QuickTime file format.

Figure A- 10 shows the structure example of a QuickTime file format for a Motion-JPEG picture with sound.

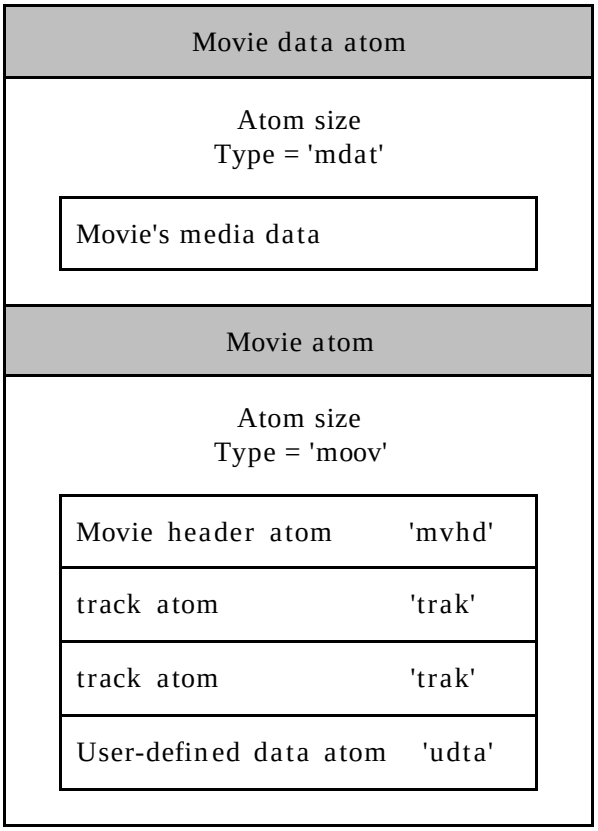


Figure A- 10: The structure example of a QuickTime file (Motion-JPEG picture with Sound)

The basic structure of a Motion-JPEG picture with sound in QuickTime file format is the same as that of Motion-JPEG picture file format shown in Chapter A.1. However, the difference between the two is that a movie data atom and a movie atom shown in Figure A- 10 include sound data and additional information for sound data playback.

The overall structure example of QuickTime file format for Motion-JPEG picture with sound in SD-picture specification is shown in Figure A- 11.

©2000 SD Association.

Annex A
Outline and Examples of QuickTime File Format

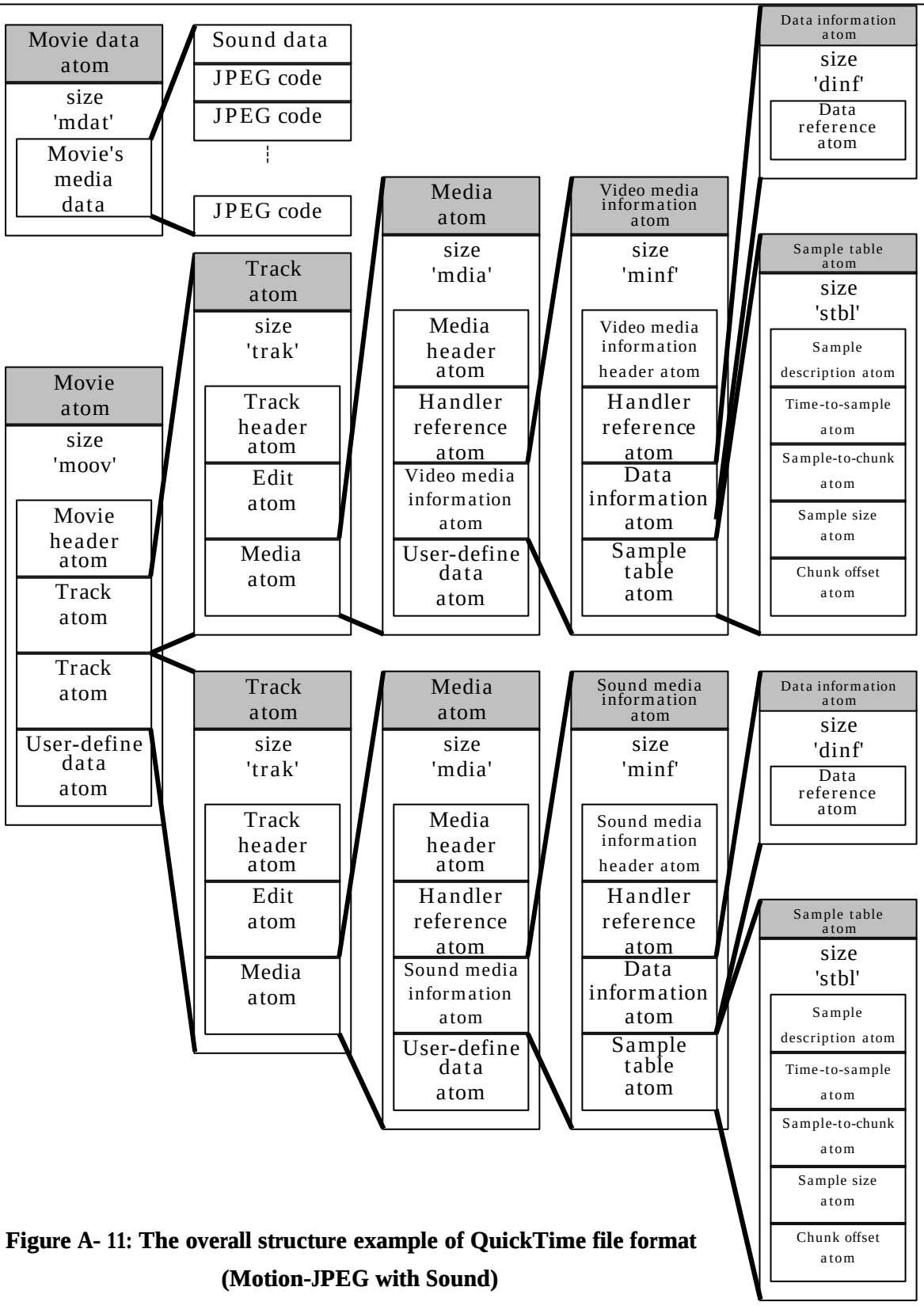


Figure A- 11: The overall structure example of QuickTime file format (Motion-JPEG with Sound)

©2000 SD Association.

CONFIDENTIAL

A.3.1 Movie data atom

A movie data atom is a container for an atom header (atom size and type field) and movie's media data(sound data and compressed image data). The layout example of a movie data atom is shown in Figure A- 12.

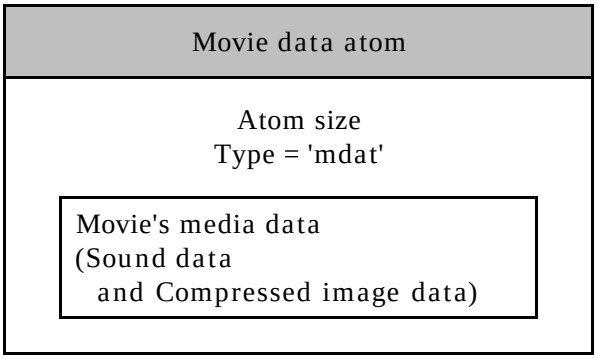


Figure A- 12: The layout example of a movie data atom

The sound data formats are as follows (cf. Chapter 4 & Chapter 5.1).

- Linear PCM (bit size : 8 bits or 16 bits; sampling frequency : 8kHz or 11.025kHz)
- u-Law PCM (bit size : 8 bits; sampling frequency : 8kHz)(*)

(*)conforming to ITU-T G.711

The compressed image data is in conformance with JPEG baseline DCT format specified in ISO/IEC10918-1 or JFIF (JPEG File Interchange Format) as described in Chapter 5.1.

©2000 SD Association.

A.3.2 Movie atom

This chapter shows an example of a movie atom included in a Motion-JPEG picture with sound.
A movie atom in a Motion-JPEG picture with sound includes track a atom for sound data in addition to a track atom for Motion-JPEG picture data.
The layout example of a movie atom in a Motion-JPEG picture with sound in QuickTime file format is shown in Figure A- 13.

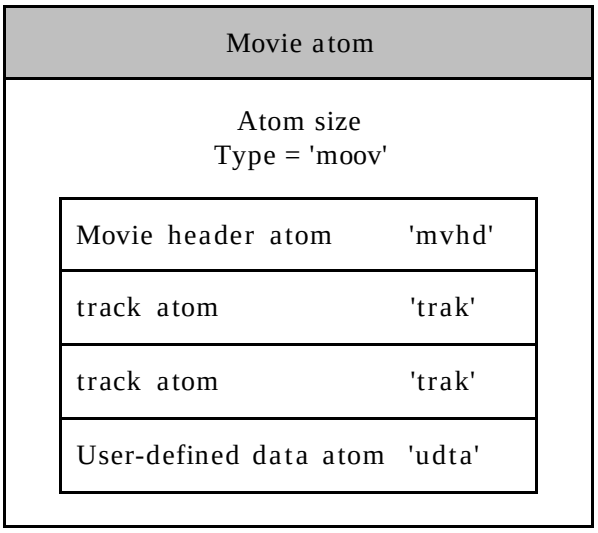


Figure A- 13: The layout example of a movie atom

©2000 SD Association.

Annex B Outline of DPOF specification

This chapter shows an outline of DPOF specification.

The contact address for the distribution of DPOF specification is shown at the following URL.
(http://www.panasonic.co.jp/avc/video/dpof/index.htm)
Send email to the following address for further information about DPOF specification and a license agreement.
(license@dpof.ved.mei.co.jp)

B.1 Directory and File Name of DPOF specification

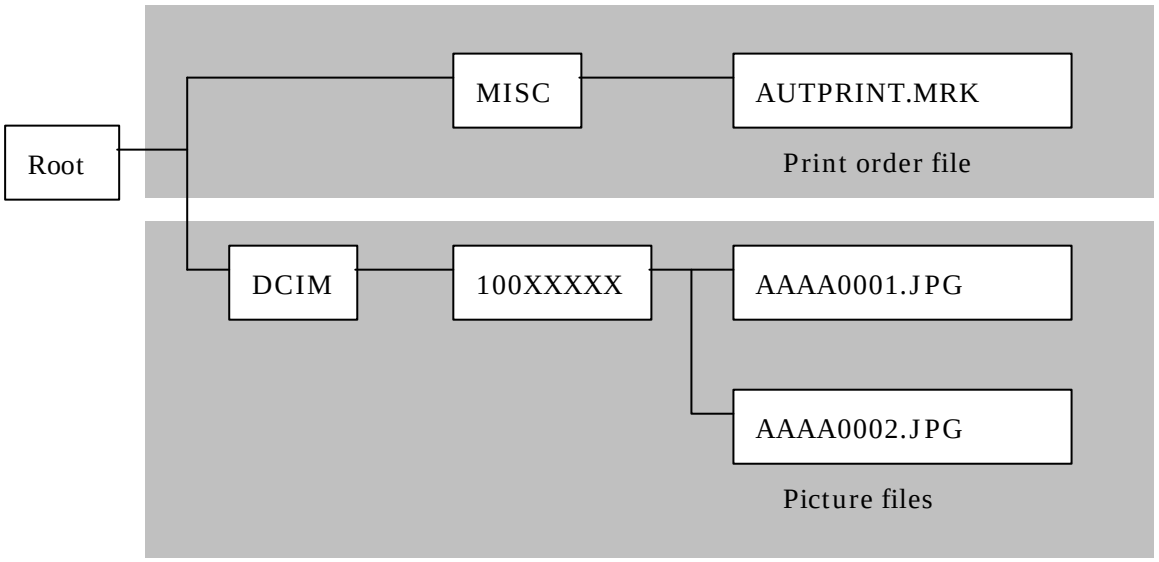


Figure B- 1: Directory and File Name of DPOF specification

©2000 SD Association.

Annex B
Outline of DPOF specification

B.2 Functions

Table B- 1: Functions of DPOF specification

Function		Content	Note
1	General Information	Version, product, date, time, etc.	
2	User Information	user name, address, telephone number, etc.	Option
3	Print Management	product-ID, quantity, size, index, etc.	
4	Image File	file format, image source file name, etc.	
5	Print Configuration	date, title, caption, trimming, rotation, etc.	Option

©2000 SD Association.

CONFIDENTIAL

SD Memory Card Specifications

Part 6
VOICE SPECIFICATION

Version 1.0

September 2000

SD Association

CONFIDENTIAL

Conditions for publication

Publisher and Copyright Holder
SD Association

Exemption
None will be liable for any damages from use of this document.

© 2000 SD Association

This page is intentionally left blank

© 2000 SD Association

CONFIDENTIAL

This page is intentionally left blank

© 2000 SD Association

CONFIDENTIAL

INDEX

Part 6 VOICE SPECIFICATION

1. General VC1-1
1.1 Scope 1
1.2 General Specifications of Presentation Data 2
1.3 Normative References 4

2. Technical Elements VC2-1
2.1 Definitions 1
2.2 Symbols 2
2.2.1 Relational operators 2
2.2.2 Mnemonics 2
2.3 Notations 3
2.3.1 Numerical notation 3
2.3.2 Range 3
2.3.3 Bit fields 3
2.4 Terminology 4
2.4.1 Default Playlist 4
2.4.2 MOB 4
2.4.3 Navigation Data 4
2.4.4 Playlist 4
2.4.5 Playlist Manager (PLMG) 4
2.4.6 POB 4
2.4.7 POB Manager (POBMG) 4
2.4.8 Presentation Data 5
2.4.9 Track 5
2.4.10 Track Manager (TKMG) 5
2.5 Abbreviations 6

3. Introduction VC3-1
3.1 Content Model 1
3.2 Directory and File Structure 2
3.3 Navigation Data Structure for SD-Voice 5
3.3.1 Structure of Playlist Manager 7
3.3.2 Structure of Track Manager 9
3.3.3 Playlist 11
3.3.3.1 Default Playlist Information (DPLI) 12
3.3.3.1.1 Text Information for Default Playlist 13
3.3.3.1.2 Still Picture for Default Playlist 13
3.3.3.1.3 TKI Reference Information 14

© 2000 SD Association

CONFIDENTIAL

3.3.3.2 Playlist Information (PLI) 15
3.3.3.2.1 TKI Reference Information 15
3.3.4 Track (TK) 16
3.3.4.1 Reference Information for Presentation Data 17
3.3.4.2 Attributes of the Presentation Data 17
3.4 Playback Procedure 18
3.4.1 Selecting a Playlist 18
3.4.2 Playback Order 19
3.4.3 Voice Playback 19
3.5 Editing Procedure of Tracks 20
3.5.1 Storing a Track by Dividing 20
3.5.2 Deleting a Track 22
3.5.3 Recording a Track 23
3.5.4 Changing Order of Tracks 25
3.5.5 Combining Tracks 26
3.5.6 Dividing a Track 27
3.6 Presentation Data Structure 29
3.6.1 Structure of Voice Object (MOB) 29
3.6.2 Structure of Picture Object (POB) 33
3.6.3 Time Search Table(TMSRT) 34
3.6.4 Block information Table(BIT) 35
3.7 System Model 36
3.7.1 Playback Operation 36
3.7.2 Special Playback Operation 36
3.7.3 Recording Operation 37
3.7.4 Editing Operation 37
3.7.4.1 Editing Tracks 37
3.7.4.2 Dividing Tracks 38
3.8 Conditions for Editing 41
3.8.1 Conditions for Combining 41
3.8.2 Examples of Combine 42
3.8.3 Category of MOBs 43
3.8.3.1 MOB_TYPE1 43
3.8.3.2 MOB_TYPE2 43

4. Navigation Data Structure VC4-1
4.1 Playlist Manager (PLMG) 3
4.1.1 Playlist Manager Information(PLMGI) 4
4.1.2 Default Playlist Information (DPLI) 9
4.1.2.1 Default Playlist General Information (DPLGI) 10
4.1.2.2 Default Playlist Track Search Pointer (DPL_TK_SRP) 17

© 2000 SD Association

CONFIDENTIAL

4.1.3 Playlist Information (PLI) 18
4.1.3.1 Playlist General Information (PLGI) 19
4.1.3.2 Playlist TrackSearch Pointer (PL_TK_SRP) 26
4.2 Track Manager (TKMG) 27
4.2.1 Track Information (TKI) 28
4.2.1.1 Track General Information (TKGI) 29
4.2.1.2 Track Text Information Data Area (TKTXI_DA) 45
4.2.1.3 Time Search Table (TMSRT) 47
4.2.1.3.1 Time Search Table Header (TMSRT_H) 47
4.2.1.3.2 TMSRT_entry 48
4.3 POB Manager (POBMG) 49
4.3.1 POB Manager Information (POBMGI) 50
4.3.2 POB Count Information (POBCI) 51

5. Presentation Data Structure VC5-1
5.1 Voice Object (MOB) 2
5.1.1 G.726 ADPCM 2
5.1.2 AMR 3
5.1.3 Structure of MOB 4
5.1.4 Structure of MOB_FRAME Header 6
5.2 Picture Object (POB) 8
5.2.1 POB Header (POB_H) 10

© 2000 SD Association

CONFIDENTIAL

Annex

Annex A (Normative): Definition of Character Table VCX-1
A.1 ISO646 character table 1
A.2 JISX0201 character table 2
A.3 ISO8859-1 character table 3

© 2000 SD Association

CONFIDENTIAL

List of Figure

Figure 1.1-1 : Structure of SD-Voice Specifications....VC1-1

Figure 3.1-1 : The hierarchical structure to present the contents of SD-Voice....VC3-1
Figure 3.2-1 : Directory and File Configuration....2
Figure 3.2-2 : Playlist Manager (PLMG)....3
Figure 3.2-3 : Track Manager (TKMG)....4
Figure 3.2-4 : POB Manager (POBMG)....4
Figure 3.3-1 : Navigation Structure for voice message data....5
Figure 3.3-2 : Navigation Structure for picture data....6
Figure 3.3.1-1 : Playlist Manager (PLMG)....7
Figure 3.3.1-2 : Default Playlist Information (DPLI)....7
Figure 3.3.1-3 : Playlist Information (PLI)....8
Figure 3.3.2-1 : Track Manager (TKMG)....9
Figure 3.3.2-2 : Track Information (TKI)....9
Figure 3.3.3-1 : Default PlayList....11
Figure 3.3.3-2 : Playlist....11
Figure 3.3.3.1-1 : Default Playlist Information....12
Figure 3.3.3.2-1 : Playlist Information (PLI)....15
Figure 3.3.4-1 : Track Contents....16
Figure 3.3.4-2 : Track Information Structure....16
Figure 3.3.4.1-1 : TKI Numbers and File Numbers....17
Figure 3.4.1-1 : Selecting a Playlist....18
Figure 3.4.2-1 : Playback Order of Tracks....19
Figure 3.4.3-1 : TKI Reference Information....19
Figure 3.5-1 : Example of DPL_TK_ATR, TKI and file structure....20
Figure 3.5.2-1 : Deleting a Track....22
Figure 3.5.3-1 : Recording a Track D....24
Figure 3.5.4-1 : Changing order of Tracks between Track C and Track E....25
Figure 3.5.5-1 : Combining Track C and Track E....26
Figure 3.5.6-1 : Dividing Track C....28
Figure 3.6.1-1 : Structure of MOB....30
Figure 3.6.1-2 : Structure of MOB_FRAME Header (MOBFH)....31
Figure 3.6.3-1 : An example of TMSRT....34
Figure 3.6.4-1 : BIT....35
Figure 3.7-1 : System Model....36
Figure 3.7.4.2-1 : Example of MOB Split (FNs_Middle_TMSRTE=20)....38
Figure 3.7.4.2-2 : Example of dividing TMSRT for Figure 3.7.4.2-1....39
Figure 3.7.4.2-3 : Example of dividing BIT for Figure 3.7.4.2-1....40

© 2000 SD Association

CONFIDENTIAL

Figure 3.8.1-1 : Combine MOBs 41
Figure 3.8.2-1 : Combines MOB_TYPE1 and MOB_TYPE1 42
Figure 3.8.2-2 : Combines MOB_TYPE1 and MOB_TYPE2 42
Figure 3.8.2-3 : Combines MOB_TYPE1, MOB_TYPE2, MOB_TYPE2 and MOB_TYPE1 42
Figure 3.8.2-4 : Combines MOB_TYPE1, MOB_TYPE2 and MOB_TYPE1 42
Figure 3.8.2-5 : Combines MOB_TYPE2, MOB_TYPE2 and MOB_TYPE1 43
Figure 3.8.3.1-1 : Structure of MOB_TYPE1 43
Figure 3.8.3.2-1 : Structure of MOB_TYPE2 43

Figure 4-1 : Navigation Data Structure VC4-1
Figure 4.1-1 : Playlist Manager (PLMG) 3
Figure 4.1.2-1 : Default Playlist Information (DPLI) 9
Figure 4.1.3-1 : Playlist Information (PLI) 18
Figure 4.2-1 : Track Manager (TKMG) 27
Figure 4.2.1-1 : Track Information (TKI) 28
Figure 4.2.1.1.-1 : relationship for entry of BIT 37
Figure 4.2.1.2-1 : Track Text Information Data Area (TKTXTI_DA) 45
Figure 4.2.1.3-1 : TMSRT 47
Figure 4.3-1 : POB Manager (POBMG) 49

Figure 5-1 : Presentation Data Structure VC5-1
Figure 5.1.3-1 : Structure of MOB 5
Figure 5.1.4-1 : Structure of MOB_FRAME Header (MOBFH) 6
Figure 5.2-1 : Structure of POB which has reference pointer to JPEG 9

© 2000 SD Association

CONFIDENTIAL

List of Table

Table 3.3.3.1.1-1 : Text Attribute of Default Playlist VC3-13
Table 3.3.3.1.2-1 : Attribute of Still Picture for Default Playlist 13
Table 3.3.3.1.2-2 : Display Order Mode 14
Table 3.3.3.1.2-3 : Display Timing Mode 14
Table 3.3.4.2-1 : TKI_MOB_ATR Information 17
Table 3.3.4.2-2 : TKI_POB_ATR Information 17
Table 3.5.1-1 : DPL_TK_ATR attribute 21
Table 3.6.1-1 : Number of encoded samples per MOB_CODEC_FRAME 29
Table 3.6.1-2 : Number of MOB_CODEC_FRAMEs 29
Table 3.6.1-3 : Number of MOB_ELEMENTs 30
Table 3.6.1-4 : Format of MOB_FRAME Fixed Length Header (MOBFLH) 31
Table 3.6.1-5 : Structure of MOB_FRAME Variable Length Header (MOBVLH) 32
Table 3.6.2-1 : JPEG format 33

Table 4.1.1-1 : PLMGI VC4-4
Table 4.1.2.1-1 : DPLGI 10
Table 4.1.2.2-1 : DPL_TK_SRP 17
Table 4.1.3.1-1 : PLGI 19
Table 4.1.3.2-1 : PL_TK_SRP 26
Table 4.2.1.1-1 : TKGI 29
Table 4.2.1.1-2 : BIT 38
Table 4.2.1.1-3 : Relation between sampling frequency and FNs_Middle_TMSRTE 41
Table 4.2.1.2-1 : Text item's tag in TKTXTI_DA 46
Table 4.2.1.3.1-1 : TMSRT_H 47
Table 4.2.1.3.2-1 : TMSRT_entry 48
Table 4.3.1-1 : POBMGI 50
Table 4.3.2-1 : POBCI 51

Table 5.1.1-1 : Restrictions for G.726 ADPCM format VC5-2
Table 5.1.2-1 : Restrictions for AMR format 3
Table 5.1.3-1 : Number of encoded samples per MOB_CODEC_FRAME 4
Table 5.1.3-2 : Number of MOB_CODEC_FRAMEs 4
Table 5.1.3-3 : Number of MOB_FRAMEs in an MOB_ELEMENT 5

©2000 SD Association

CONFIDENTIAL

Table 5.1.4-1 : Format of MOB_FRAME Fixed Length Header (MOBFLH) 6
Table 5.1.4-2 : Structure of MOB Variable Length Header (MOBVLH) 7
Table 5.2-1 : JPEG format 9
Table 5.2.1-1 : POB Header (POB_H) 10

Table A.1-1 : ISO646 character table VCX-1
Table A.2-1 : JISX0201 character table 2
Table A.3-1: ISO8859-1 character table 3

©2000 SD Association

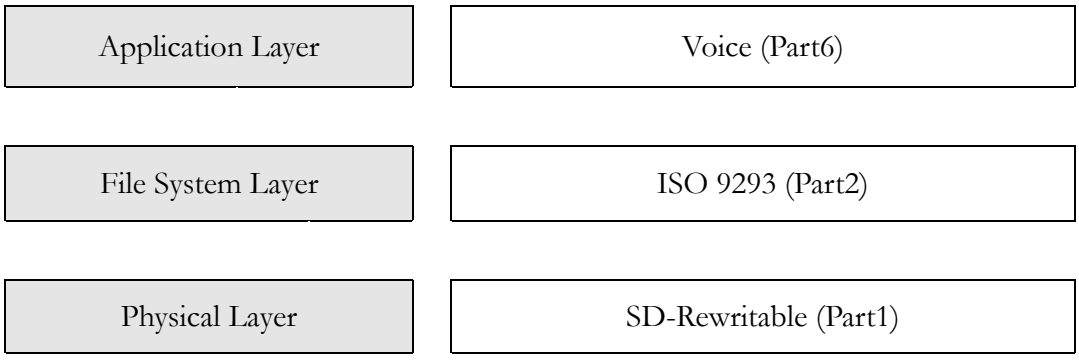
CONFIDENTIAL

1. General

1.1 Scope

This Part defines the SD-Voice which is recorded in the SD Memory Card. The voice message presentation data complies with ITU-T G.726 ADPCM and 3GPP AMR, and also vendor dependent codecs are available. The card of which the data is defined in this specification shall comply with **"Part1. PHYSICAL SPECIFICATION"** and **"Part2. FILE SYSTEM** of SD Memory Card specifications.

◊ **Figure 1.1-1 : Structure of SD-Voice Specification**



© 2000 SD Association

CONFIDENTIAL

1.2 General Specifications of Presentation Data

Hierarchical structure for contents classification

Hierarchy name	Maximum number
Playlist	99 per "SD_VCyyy" directory (001 ≤ yyy ≤ 999)
Track	999 per card / 99 per Playlist

Presentation data

Data	General specification	
Structure	ITU-T G.726 ADPCM 3GPP AMR / Interface Format 1 Vendor dependent	
Voice Message	Number of streams	:1
	Coding mode	: ITU-T G.726 ADPCM : AMR [Interface Format 1] : Vendor dependent codecs : Others (reserved for option)
	Voice message data basics	
	Sampling frequency	: 8kHz (G.726, AMR)
	Bitrates	: G.726 : 40~16kbit/s : AMR : VBR (max 13.15kbit/s)
	Number of channels	: 1
Picture(option)	Compression	: JPEG
	Number of pixels	: 96x96, 640x480 (recommend) : 160x120 ~ 1800x1200(option)
	Chrominance sampling	: 4:2:2, 4:2:0
Text(option)	Character set code	: ISO646 / JISX0201 / ISO8859-1 / Shift JIS

© 2000 SD Association

CONFIDENTIAL

1.3 Normative References

1) SD Memory Card Specifications
Part1. PHYSICAL SPECIFICATION Version 1.0, February 2000

2) SD Memory Card Specifications
Part2. FILE SYSTEM SPECIFICATION Version 1.0, February 2000

3) SD Memory Card Specifications
Part4. AUDIO SPECIFICATION Version 1.0, February 2000

4) ITU-T Recommendation G.726: 1990
40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation

5) 3GPP TS26.071: 1999
AMR Speech Codec; General Description

6) 3GPP TS26.101: 2000
AMR Speech Codec; Frame Structure

7) ISO/IEC 646: 1983
Information processing – ISO 7-bit coded character set for information interchange

8) ISO 8859-1: 1987
Information processing – 8-bit single-byte coded graphic character sets – Part 1
Latin alphabet No.1

9) ISO/IEC9293:1994
Information technology - Volume and file structure of disk cartridges for information interchange

10) JIS X 0201-1997
7-bit and 8-bit coded character sets for information interchange

11) JIS X 0208-1997
Code of the Japanese graphic character set for information interchange

12) JEIDA Exif Version 2.1: 1998
Digital Still Camera Image File Format Standard

13) JEIDA DCF Version 1.0: 1998

© 2000 SD Association

CONFIDENTIAL

Design rule for Camera File System

14) ISO/IEC 10918-1: 1994
Information technology - Digital compression and coding of continuous-tone still pictures: requirements and guidelines

15) IEEE Registration Authority
Organizationally Unique Identifiers (OUI) / Company_id

© 2000 SD Association

CONFIDENTIAL

2. Technical Elements

2.1 Definitions

For the purpose of this specification, the following definitions apply:

RBN : Relative Byte number (unsigned value) from the first byte of a defined field, starting with zero.

RBP : Relative Byte Position within a descriptor, starting with zero.

reserved : Reserved field. The term "reserved" indicates that the value may be used in the future for SD Memory Card specifications. Unless other wise specified within this specification, all reserved bits shall be set to zero.

RLBN : Relative Logical Block number (unsigned value) with assigned Logical Block as '0'.

ZERO : represents a single bit with the value 0.

ONE : represents a single bit with the value 1.

data_element[m..n] : The inclusive range of bits between bit m and bit n in the data_element.

© 2000 SD Association

CONFIDENTIAL

2.2 Symbols

2.2.1 Relational operators

≤ or <= Less than or equal to

2.2.2 Mnemonics

CH :channel

© 2000 SD Association

CONFIDENTIAL

2.3 Notations

2.3.1 Numerical notation

Numbers in decimal notation are represented by decimal digits, namely 0 to 9. **Digit(s)** represents any digit from digit 0 to 9.

Numbers in Hexadecimal notation are represented as a sequence of one or more hexadecimal digits namely 0 to 9 and A to F, suffixed by "h".

Numbers in binary notation are represented as a sequence of one or more binary digits namely 0 to 1, suffixed by "b".

Multiple-byte numerical values in a description field shall be recorded in the **big endian** representation. e.g. the 32-bit hexadecimal number '12345678h' shall be recorded as '12h', '34h', '56h', '78h'.

1 Mbytes represents 10^6 bytes

1 Gbytes represents 10^9 bytes

1 kB represents 2^{10} bytes

1 MB represents 2^{20} bytes

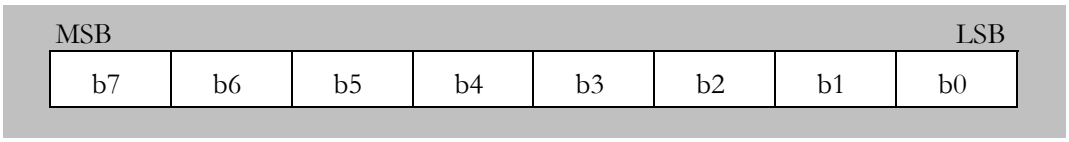
1 GB represents 2^{30} bytes

2.3.2 Range

Constant_1..Constant_2 or [Constant1..Constant2] denotes the range from and including Constant_1 up to and including Constant_2, in increments of 1.

2.3.3 Bit fields

Certain fields or parts of fields are interpreted as an array of bits.
This array of bits shall be referred to as a bit field. Bit positions within an **n** bit field are numbered such that the least significant bit is numbered 0 and the most significant bit is numbered **n–1**.



© 2000 SD Association

CONFIDENTIAL

2.4 Terminology

2.4.1 Default Playlist

The Default Playlist defines the playback order of all Tracks stored in the "SD_VCyyy" directory (001 ≤ yyy ≤ 999). A maximum of 999 Track playback orders can be defined. One Default playlist exists in the "SD_VCyyy" directory.

2.4.2 MOB

Compressed voice message data is treated as the unit of Voice Object (MOB) in the SD-Voice Specification. Compressed voice message format operated by MOB are ITU-T G.726 ADPCM, and 3GPP AMR.

2.4.3 Navigation Data

This is the information on the attribution and the control data of the Presentation Data. It consists of three logical structures, Playlist Manager (PLMG) Track Manager (TKMG), and POB Manager (POBMG).

2.4.4 Playlist

The Playlist defines the playback order of Tracks. A maximum of 99 Tracks playback orders can be defined. A maximum of 99 Playlists can be defined in the "SD_VCyyy" directory (001 ≤ yyy ≤ 999).

2.4.5 Playlist Manager (PLMG)

Playlist Manager describes information about Default Playlist and Playlists within the "SD_VCyyy" directory (001 ≤ yyy ≤ 999). It consists of Playlist Manager Information (PLMGI), Default Playlist Information (DPLI), and set of Playlist Information (PLI)

2.4.6 POB

Picture Information handled in SD-Voice Specification is called Picture Object (POB). POB only stores a single piece of Picture Information in a POB file.

2.4.7 POB Manager (POBMG)

POB Manager describes POB management information. It consists of POB Manager Information (POBMGI) and set of POB Count Information (POBCI).

2.4.8 Presentation Data

© 2000 SD Association

CONFIDENTIAL

Presentation Data consists of Voice Object (MOB) and Picture Object (POB), which are all operated from Navigation Data.

2.4.9 Track

A Track is a unit of Message. The Track consists of Voice Object (mandatory), time management data (mandatory), text data (optional) and Picture Object (optional).

2.4.10 Track Manager (TKMG)

Track Manager describes information about Tracks stored in "SD_VCyyy" directory (001 ≤ yyy ≤ 999). It consists of set of Track Information (TKI). TKI number ranges '1' to '999'.

© 2000 SD Association

CONFIDENTIAL

2.5 Abbreviations

BIT	Block Information Table
DATA_OFFSET	Size of invalid area at the head of MOB file
DPL_TK_ATR	Default Playlist Track attribute
DPL_TK_SRP	Default Playlist Track Search Pointer
DPL_TKN	Default Playlist Track number
DPLGI	Default Playlist General Information
DPLI	Default Playlist Information
DPLI_APP_ATR	Default Playlist application attribute
DPLI_ID	DPLI Identifier
DPLI_POB_ATR	DPLI_POB attribute
DPLI_POB_SRP	DPLI_POB search pointer
DPLI_PB_TM	Playback Time of Default Playlist
DPLI_PLTI	Default Playlist Text Information
DPLI_PLTI1_ATR	Attribute of Default Playlist Text Information 1
DPLI_PLTI2_ATR	Attribute of Default Playlist Text Information 2
DPLI_RSM_PL	Resume information of Default Playlist
DPLI_SUBDIR_NM1	Sub-directory name for 1 byte characters
DPLI_SUBDIR_NM2	Sub-directory name for 2 byte characters
DPLI_TK_Ns	Number of Tracks in DPLI
DPLI_TKIN	TKI number in Default Playlist
FNs_1st_TMSRTE	Number of MOB_FRAMEs in the first MOB_ELEMENT
FNs_Last_TMSRTE	Number of MOB_FRAMEs in the last MOB_ELEMENT
FNs_Middle_TMRTE	Number of MOB_FRAMEs in MOB_ELEMENT
MOB_BLOCK	MOB Block
MOB_CODEC_FRAME	MOB codec frame
MOB_ELEMENT	MOB element
MOB_FRAME	MOB frame
MOBFLH	MOB_FRAME Fixed Length Header
MOBVLH	MOB_FRAME Variable Length Header
POB_Ns	Number of POBs
POB_RCN	Reference count number of POB
POBCI	POB Count Information
POBMG	POB Manager
POBMGI	POB Manager Information
POBMGI_ID	POBMGI Identifier
PL_TK_SRP	Playlist Track Search Pointer
PL_TKIN	TKI number in Playlist

© 2000 SD Association

CONFIDENTIAL

PLMG	Playlist Manager
PLMG_AP_PL	Auto Play Playlist
PLMG_APP_ATR	PLMG application attribute
PLMG_ID	PLMG Identifier
PLMG_PL_Ns	Number of Playlists
PLMG_RSM_PL	Playlist is used to be played back at last time
PLMG_SZ	Size of PLMG
PLMGI	Playlist Manager Information
PLGI	Playlist General Information
PLI	Playlist Information
PLI_APP_ATR	Playlist application attribute
PLI_ID	PLI Identifier
PLI_POB_ATR	POB Attribute of PLI
PLI_POB_SRP	POB Search Pointer of PLI
PLI_PB_TM	Playback Time of Playlist
PLI_RSM_PL	Resume information of Playlist
PLI_TK_Ns	Number of Tracks in PLI
PLI_PLTI	Playlist Text Information
PLI_PLTI1_ATR	Attribute of Playlist Text Information 1
PLI_PLTI2_ATR	Attribute of Playlist Text Information 2
SDV_ID	SD-Voice Identifier
SZ_DATA	Data length of MOB_BLOCK
TIME_LENGTH	Presentation time duration of MOB_ELEMENT
TKGI	Track General Information
TKI	Track Information
TKI_BLK_ATR	Block attribute of TKI
TKI_CMPNY_ID	Company Identifier
TKI_ID	TKI Identifier
TKI_MOB_ATR	MOB Attribute of TKI
TKI_POB_ATR	POB Attribute of TKI
TKI_POB_SRP	POB search pointer of TKI
TKI_LNK_PTR	Link pointer to next TKI
TKI_Ns	Number of TKIs
TKI_PB_TM	TKI Playback Time
TKI_RCD_TM	TKI Recording Time
TKI_SZ	Size of TKI
TKI_TI1_ATR	Attribute of TKI Text Information 1
TKI_TI2_ATR	Attribute of TKI Text Information 2
TKI_TMSRT_SA	Start Address of TMSRT
TKIN	TKI Number

© 2000 SD Association

CONFIDENTIAL

TKMG	Track Manager
TKMGI	Track Manager Information
TKMGI_ID	TKMGI Identifier
TKTXTI	Track Text Information
TKTXTI_ABM	Album Name
TKTXTI_ARR	Editor of Track
TKTXTI_ART	Artist Name
TKTXTI_CMP	Composer of Track
TKTXTI_CRD	Date of Track
TKTXTI_DA	Track Text Information Data Area
TKTXTI_FR1	Free item 1
TKTXTI_FR2	Free item 2
TKTXTI_FR3	Free item 3
TKTXTI_FR4	Free item 4
TKTXTI_FR5	Free item 5
TKTXTI_FR6	Free item 6
TKTXTI_GNR	Genre of Track
TKTXTI_MSS	Artist's Message of Track
TKTXTI_PCM	Provider's comment of Track
TKTXTI_PRD	Producer of Track
TKTXTI_RCD	Recording Company of Track
TKTXTI_SW	Song writer of Track
TKTXTI_TTL	Title Name
TKTXTI_UCM	User's comment of Track
TKTXTI_URL	URL of Track
TMSRT	Time Search Table
TMSRT_entry	TMSRT entry
TMSRT_ENT	TMSRT entry
TMSRT_ID	TMSRT Identifier
TMSRTE_Ns	Number of TMSRT_entry
VERN	Version number

© 2000 SD Association

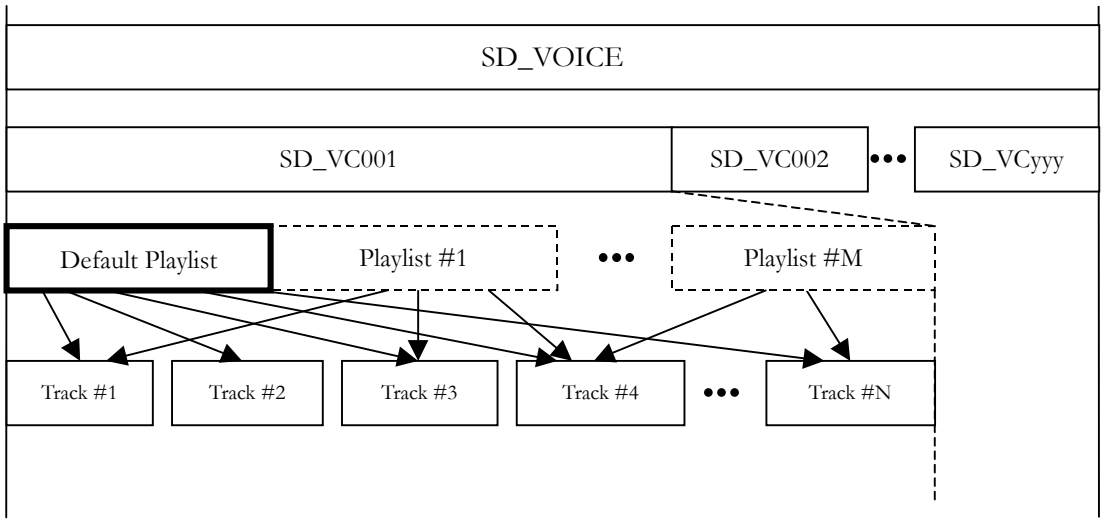
CONFIDENTIAL

3. Introduction

3.1 Content Model

Recorded content which complies with SD-Voice Specifications is accessed via a content model which consists of Playlists and Messages as shown in [Figure 3.1-1]:

◊ **Figure 3.1-1: The hierarchical structure to present the contents of SD-Voice**



(yyy ≤ 999, M ≤ 99, N ≤ 999)

- **Playlist**

A Playlist defines the playback order of Tracks in an "SD_VCyyy" directory (001≤yyy≤999). There are two types of Playlist. One is Default Playlist which specifies playback order of all the Tracks, and the other is User-defined Playlist (optional). Using a User-defined, one can specify desirable playback order of Tracks selected among all the Tracks.

- **Track**

A Track is a unit of Message.
A Track consists of Voice Object (MOB: mandatory), Time Search Table (TMSRT: mandatory), Text Data (optional) and Picture Object (POB: optional).

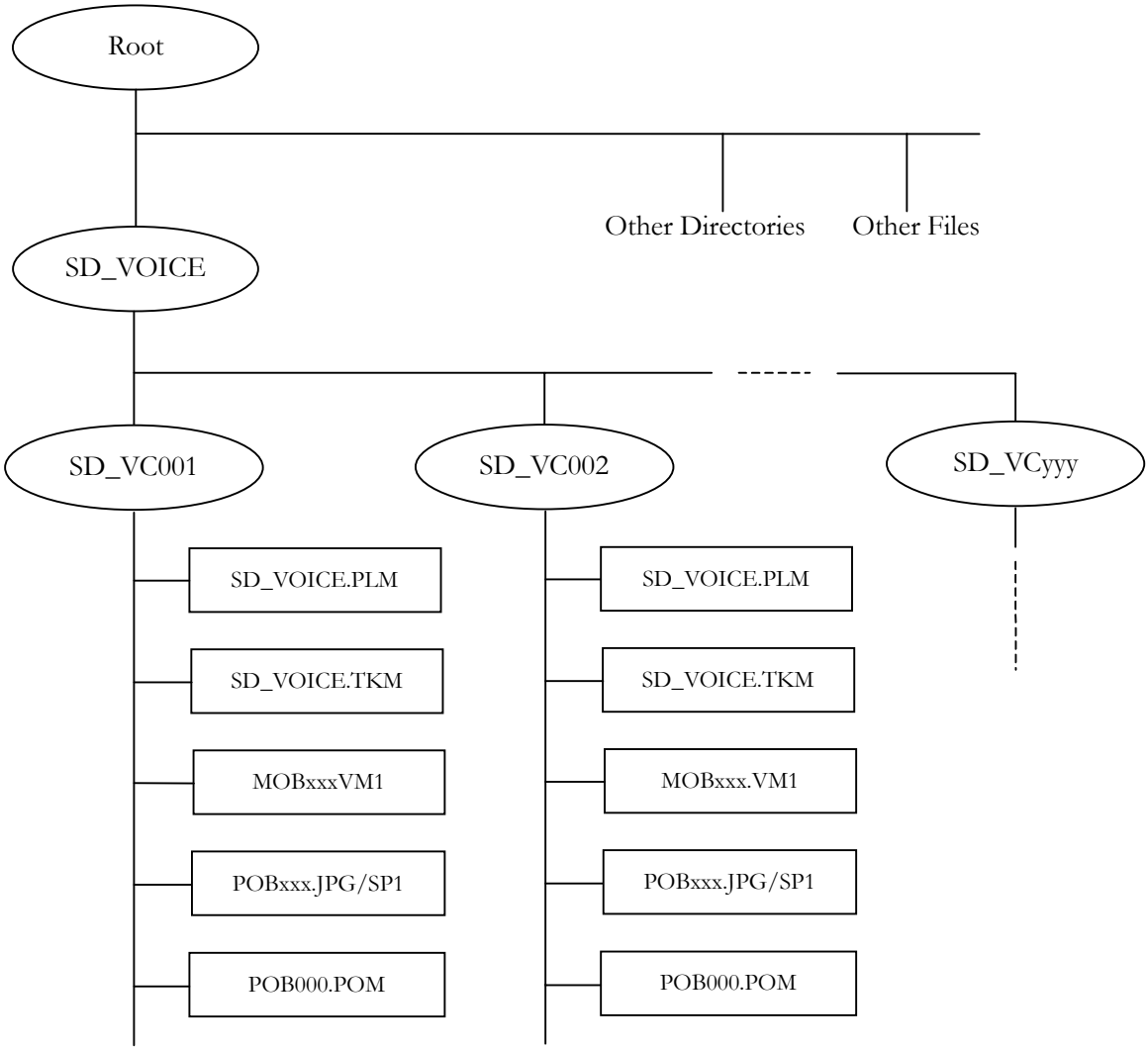
© 2000 SD Association

CONFIDENTIAL

3.2 Directory and File Structure

[Figure 3.2-1] illustrates the directory and files where the data compliant to this Specifications are recorded.
All files are placed under SD_VOICE directory.
From the viewpoint of content protection, the following condition is recommended;
SD_VOICE directory and all files under SD_VOICE directory are set to 'read-only', and 'hidden' attribute.

◊ **Figure 3.2-1: Directory and File Configuration**



(yyy ≤ 999)

- **SD_VOICE (Directory)**

This directory corresponds to the root directory in the SD-Voice. SD_VCyyy directory (001≤yyy≤999) can be created as subdirectory.

- **SD_VCyyy (Directory)**

© 2000 SD Association

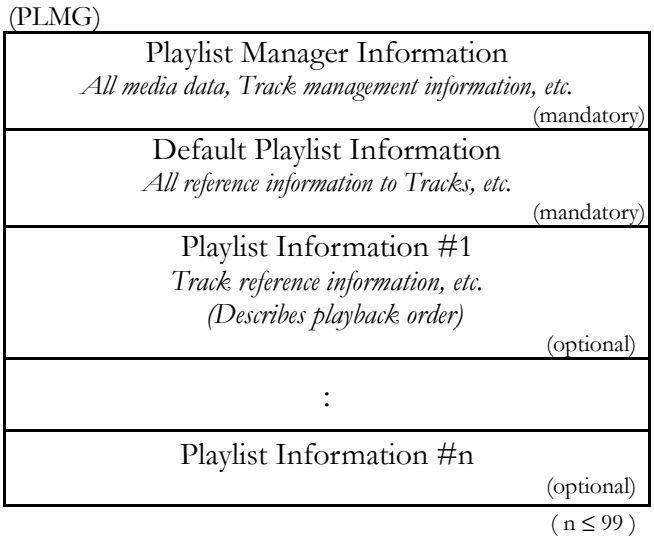
CONFIDENTIAL

This directory is created under the SD_VOICE directory. All files compliant with the SD-Voice specification are placed under this directory, and their configurations are similar to those under the SD-Voice specification.
"yyy" takes a value between "001" and "999" and shall be unique in "SD_VCyyy" directory.

- **SD_VOICE.PLM (File)**

Only one SD_VOICE.PLM file exists in an "SD_VCyyy" directory. All management information regarding SD-Voice Playlists is described under the Playlist Manager (PLMG) as shown in [Figure 3.2-2].

◊ **Figure 3.2-2 : Playlist Manager (PLMG)**



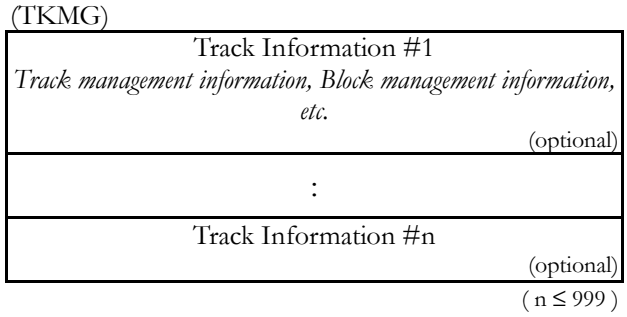
- **SD_VOICE.TKM (File)**

Only one SD_VOICE.TKM file exists in an "SD_VCyyy" directory. The management information for all Tracks and other information are described under the Track Manager (TKMG) as shown in [Figure 3.2-3].

© 2000 SD Association

CONFIDENTIAL

◊ **Figure 3.2-3 : Track Manager (TKMG)**



- **MOBxxx.VM1 (File)**

An MOBxxx.VM1 file stores non-encrypted Voice Object (MOB). "xxx" takes a value between '001' and '999'.

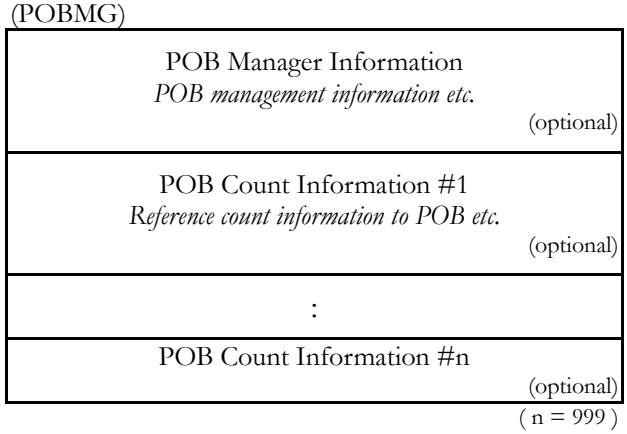
- **POBxxx.JPG (File) and POBxxx.SP1 (File)**

A POBxxx.JPG file stores JPEG compressed Picture Object (POB). In a POBxxx.SP1 file, there is a reference pointer to a JPEG file. "xxx" takes a value between '001' and '999'.

- **POB000.POM (File)**

Only one POB000.POM file exists in an "SD_VCyyy" directory when POBxxx.JPG/SP1 exists. The management information regarding the Picture Object files (POBxxx.JPG, POBxxx.SP1) are described under the POB Manager (POBMG) as shown in [Figure 3.2-4].

◊ **Figure 3.2-4 : POB Manager (POBMG)**



© 2000 SD Association

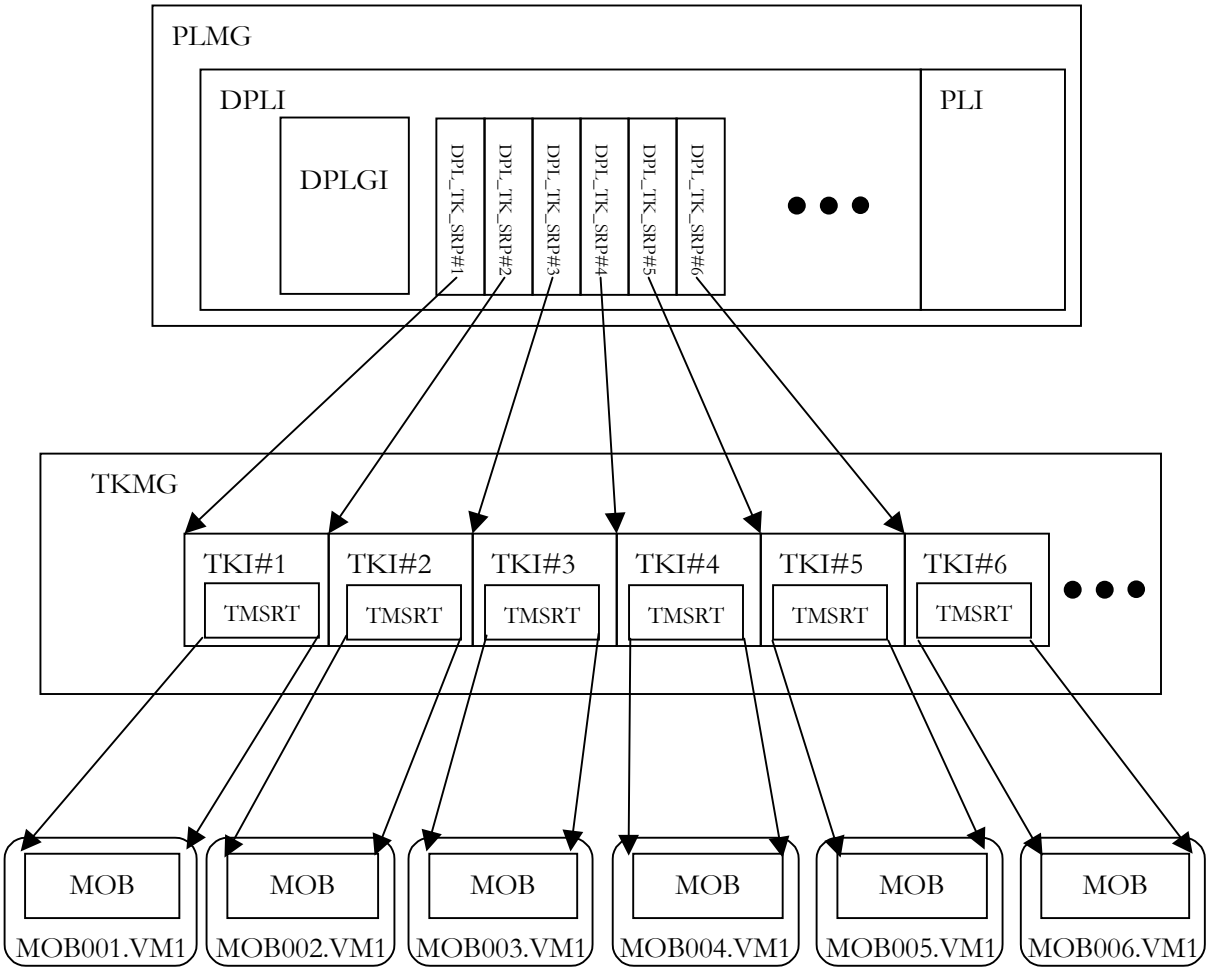
CONFIDENTIAL

3.3 Navigation Data Structure for SD-Voice

1) Navigation Data Structure for voice message data

[Figure 3.3-1] illustrates an example of navigation data structure for voice message data. Default Playlist Information (DPLI) and Playlist Information (PLI) refer to Track Information (TKI) through DPLI_TK_SRP and PLI_TK_SRP, respectively. An MOB is further refferenced using TMSRT in each TKI. An MOB file is linked to a TKI by its "xxx" in the file name, which corresponds to the TKI number.

◊ **Figure 3.3-1 : Navigation Data Structure for voice message data**



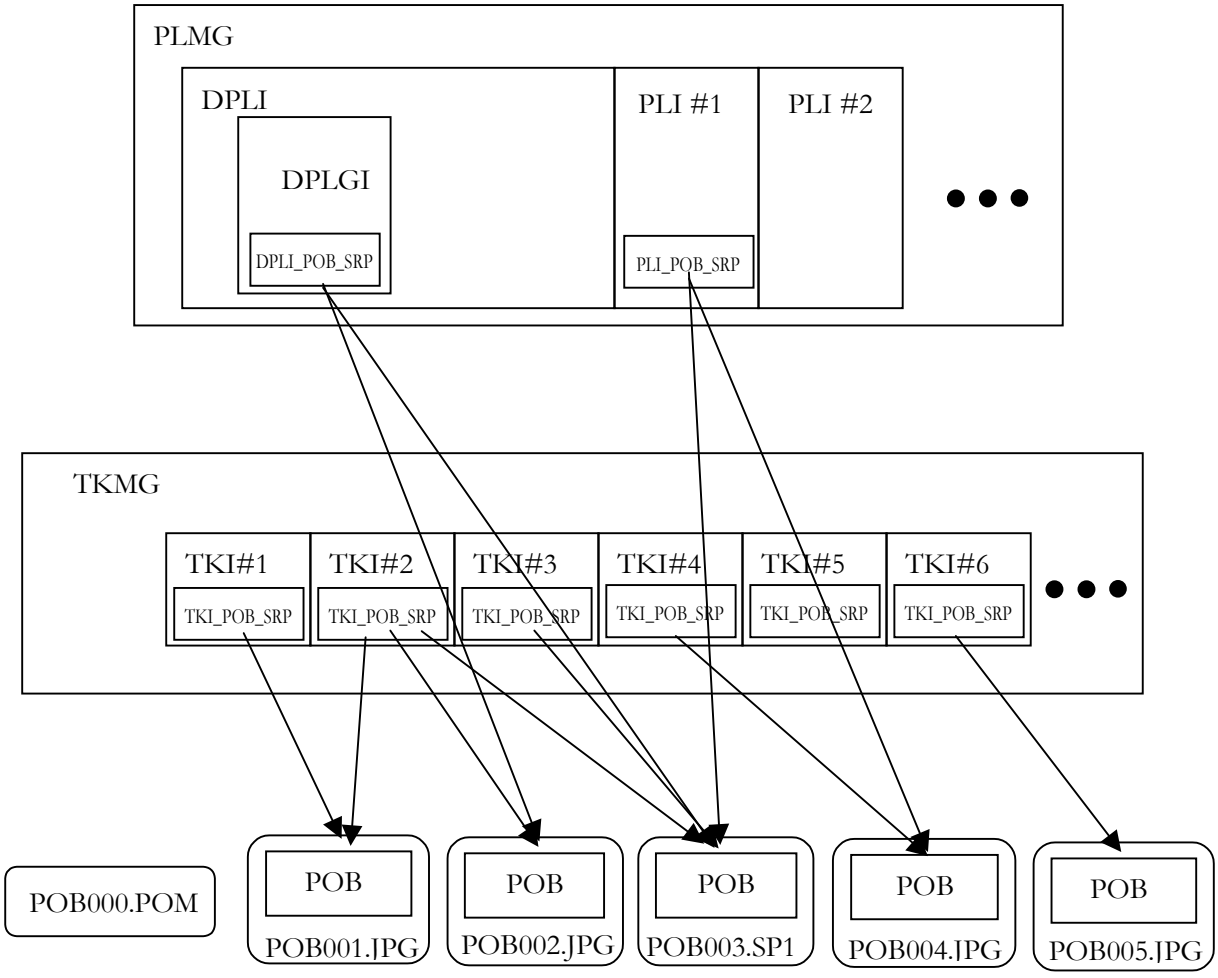
© 2000 SD Association

CONFIDENTIAL

2) Navigation Data Structure for picture data

[Figure 3.3-2] illustrates navigation data structure for picture data. A Picture Data Object may be referrenced by DPLI_POB_SRP, PLI_POB_SRP and TKI_POB_SRP which belongs to the Default Playlist Information (DPLI), Playlist Information (PLI) and Track Information(TKI), respectively.

◊ **Figure3.3-2 : Navigation Structure for picture data**



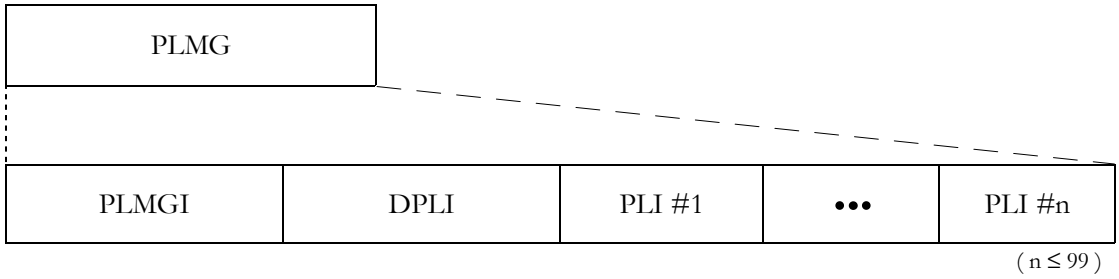
© 2000 SD Association

CONFIDENTIAL

3.3.1 Structure of Playlist Manager

The Playlist Manager (PLMG) consists of three elements: Playlist Manager Information (PLMGI), Default Playlist Information (DPLI) and Playlist Information (PLI).

◊ **Figure 3.3.1-1 : Playlist Manager (PLMG)**



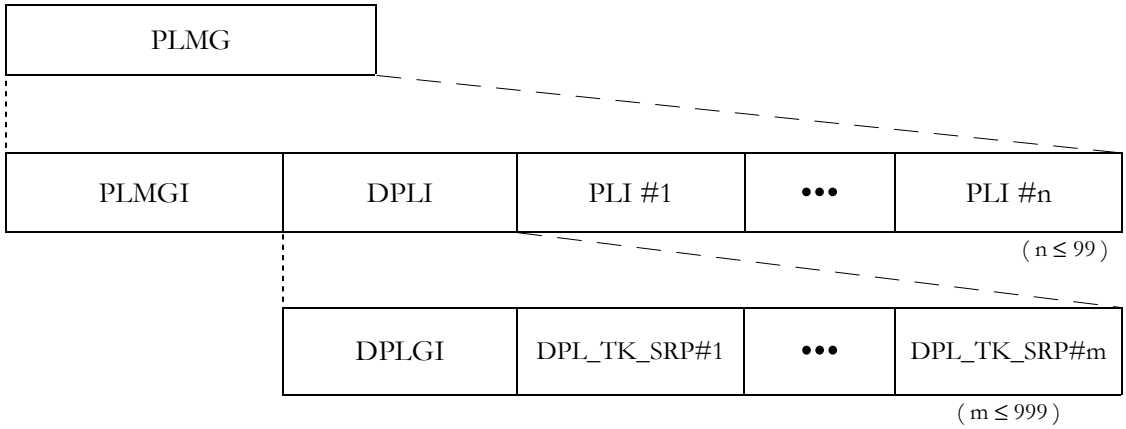
- **Playlist Manager Information (PLMGI)**

Describes SD-Voice information such as SD-Voice ID, number of Playlists, number of Tracks in the Default Playlist, which corresponds to the total number of Tracks, etc.

- **Default Playlist Information (DPLI)**

The Default Playlist Information (DPLI) consists of Default Playlist General Information (DPLGI) and Default Playlist Track Search Pointer (DPL_TK_SRP).

◊ **Figure 3.3.1-2 : Default Playlist Information (DPLI)**



© 2000 SD Association

CONFIDENTIAL

- **Default Playlist General Information (DPLGI)**

This contains the mangement information for the Default Playlist. The DPLGI describes information such as the number of Tracks referenced from the Default Playlist (equivalent to the number of Tracks stored in the "SD_VCyyy" directory), the total playback time of all valid Tracks in the "SD_VCyyy" directory and the text data. The DPLGI also manages the list of slideshows played back on the Default Playlist.

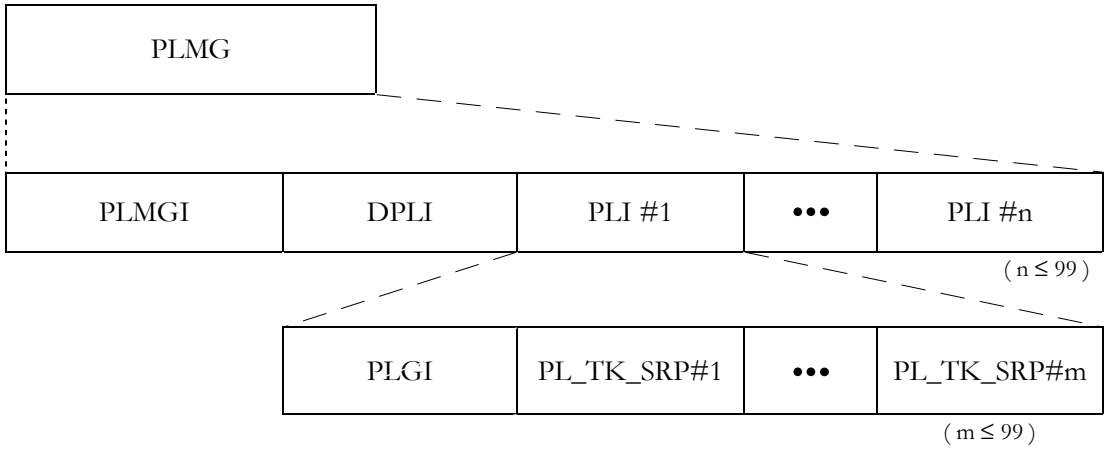
- **Default Playlist Track Search Pointer (DPL_TK_SRP)**

Describes the playback information for the Tracks in the order of Default Playlist playback. In each DPL_TK_SRP, an entry number into Track Information (TKI) and a flag indicating the state of the TKI, such as whether the Track is being used or unused, are described.

- **Playlist Information (PLI)**

The Playlist Information (PLI) consists of the Playlist General Information (PLGI) and the Playlist Track Search Pointers (PL_TK_SRP).

◊ **Figure 3.3.1-3 : Playlist Information (PLI)**



- **Playlist General Information (PLGI)**

Describes the number of Tracks in the Playlist, and total playback time of the Playlist. The PLGI also manages the list of slideshows played back in the Playlist.

- **Playlist Track Search Pointer (PL_TK_SRP)**

The playback information (TKI numbers) referenced from the Playlist is described.

© 2000 SD Association

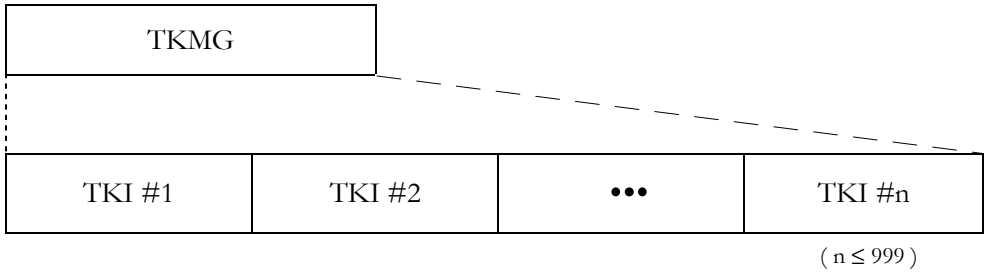
CONFIDENTIAL

3.3.2 Structure of Track Manager

- **Track Manager (TKMG)**

The Track Manager (TKMG) consists of the Track Information (TKI).

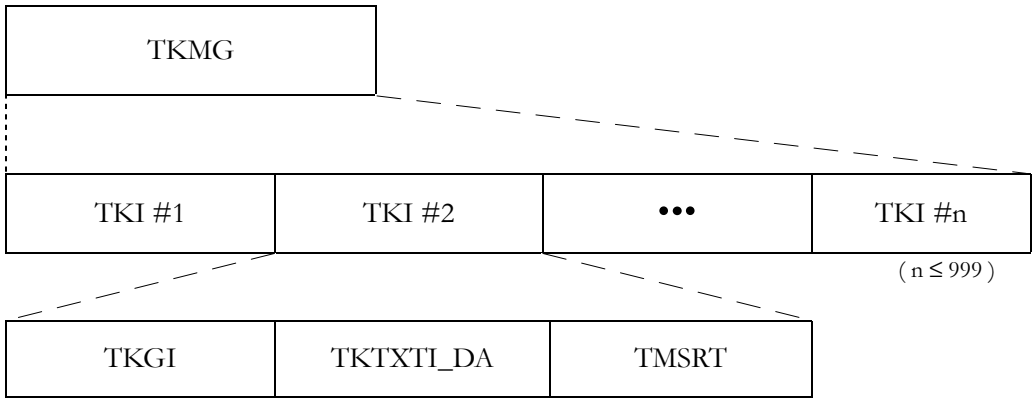
◊ **Figure 3.3.2-1 : Track Manager (TKMG)**



- **Track Information (TKI)**

Track Information (TKI) consists of Track General Information (TKGI), Track Text Information Data Area (TKTXTI_DA) and Time Search Table (TMSRT). TKI has a fixed size of 1024 bytes. TKGI and TKTXTI_DA have a fixed size of 512 bytes in total.

◊ **Figure 3.3.2-2 : Track Information (TKI)**



- **Track General Information (TKGI)**

Describes Track management information, slideshow picture list and reference information for Time Search Table (TMSRT).

- **Track Text Information Data Area (TKTXTI_DA)**

© 2000 SD Association

CONFIDENTIAL

Describes the text data.

- **Time Search Table (TMSRT)**

Describes the data sizes which corresponds to every two seconds of playback. This information is used when MOB is accessed for fast forward and rewind. The table size is fixed to 512 bytes. If it exceeds 512 bytes, a new TKI and an MOB file must be created in order to use its TMSRT.

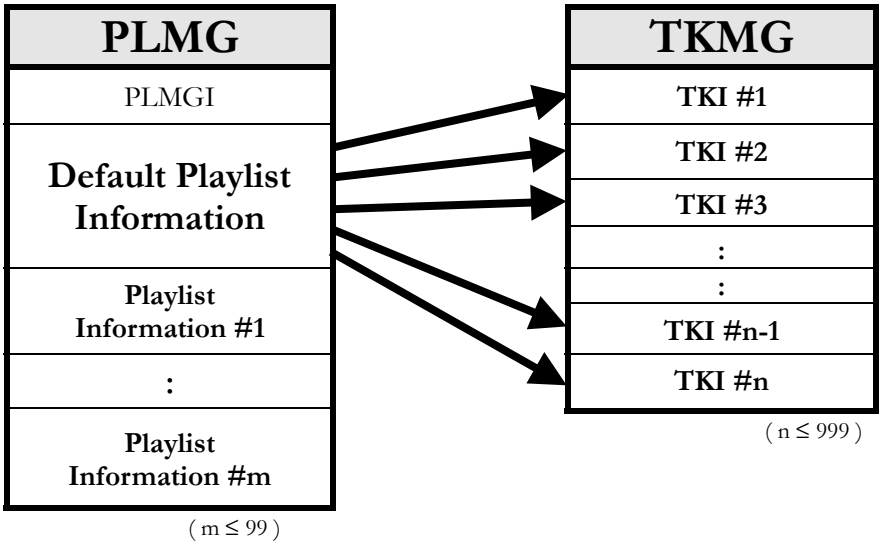
© 2000 SD Association

CONFIDENTIAL

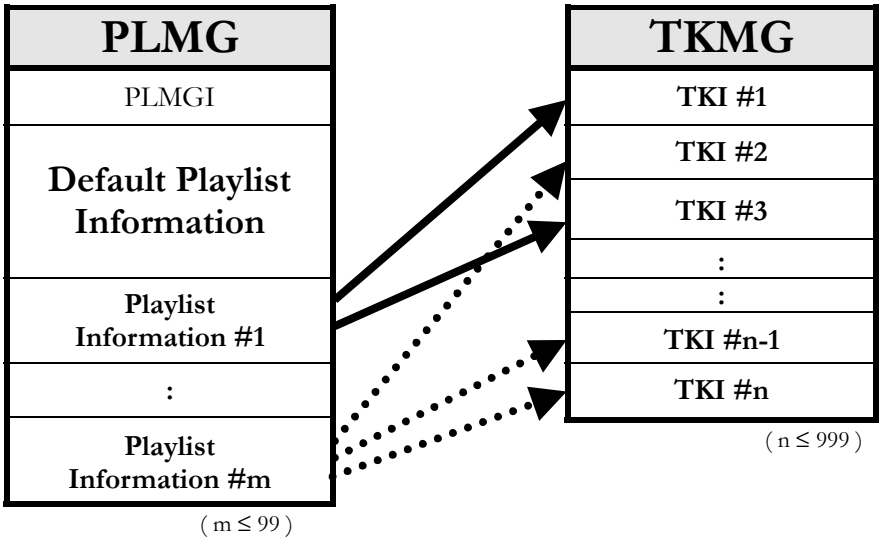
3.3.3 Playlist

Playlist defines a playback order of the associated Track. Playlist manages all Tracks stored in the "SD_VCyyy" directory and is categorized to the Default Playlist and optional user definable Playlists. Management information are described in the Default Playlist Information (DPLI) and the Playlist Information (PLI) of the PLMG within the SD_VOICE.PLM file.

◊ **Figure 3.3.3-1 : Default Playlist**



◊ **Figure 3.3.3-2 : Playlist**



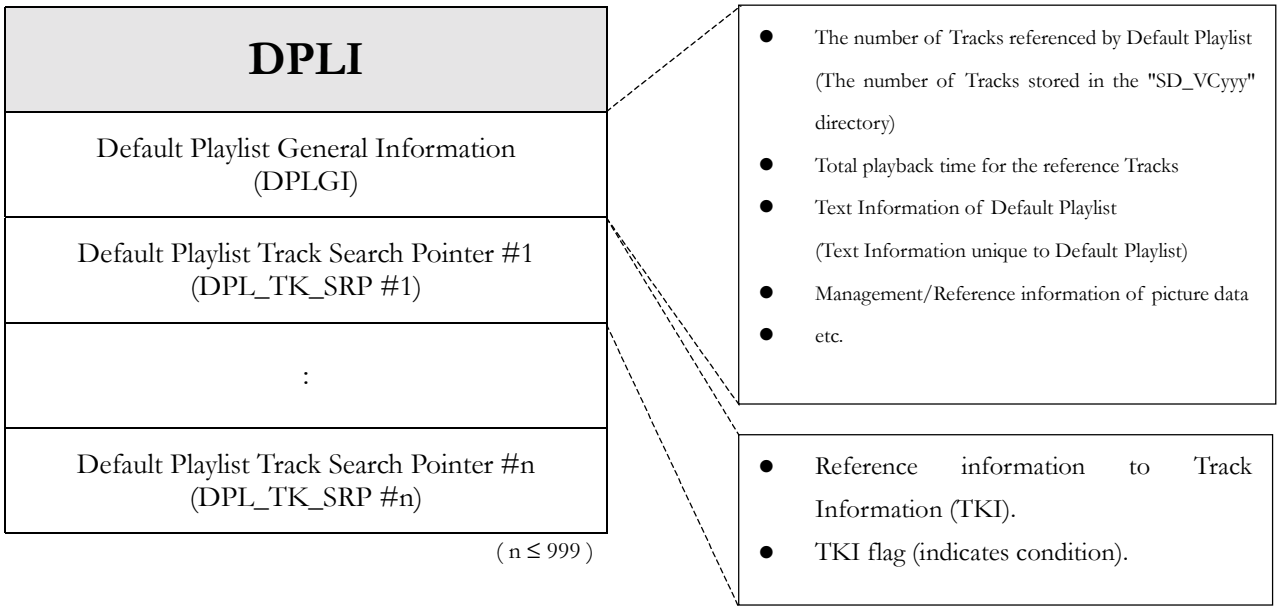
© 2000 SD Association

CONFIDENTIAL

3.3.3.1 Default Playlist Information (DPLI)

The Default Playlist contains reference information and playback orders for all the Tracks stored in the "SD_VCyyy" directory. The Tracks are stored in the same order as they were recorded. New Tracks are to be registered at the tail end of the Default Playlist.

◊ **Figure 3.3.3.1-1: Default Playlist Information**



© 2000 SD Association

CONFIDENTIAL

3.3.3.1.1 Text Information for Default Playlist

The Default Playlist may store distinct text information. Two types of text information may be held depending on the Character Set Code. For example, this information is used to distinguish the SD Memory Card from the other SD Memory Cards.

- **Attribute of the Text Information for Default Playlist**

Attributes of the Text Information for Default Playlist are described in DPLI_PLTI1_ATR and DPLI_PLTI2_ATR.

◊ **Table 3.3.3.1.1-1: Text Attribute of Default Playlist**

DPLI_PLTI1_ATR	ISO646, JISX0201, ISO8859-1
DPLI_PLTI2_ATR	Shift JIS

3.3.3.1.2 Still Picture for Default Playlist

The Default Playlist may hold information regarding the still pictures to be referenced.

- **Attribute of Still Picture for Default Playlist**

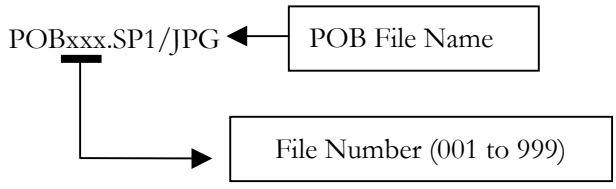
Attribute of the Still Picture for Default Playlist is described in the DPLI_POB_SRP.

◊ **Table 3.3.3.1.2-2: Attribute of Still Picture for Default Playlist**

Number of pixels	96x96, 640x480(recommended) 160x120 ~ 1800x1200(optional)
Chrominance sampling	4:2:2, 4:2:0
Picture Coding Mode	JPEG

- **Reference Information of the Still Picture for Default Playlist**

Information of the Still Picture referenced in the Default Playlist is described in DPLI_POB_SRP.



A POB is stored in a separate file named POBxxx.SP1/JPG, where "xxx" denotes the reference number to be stored in DPLI_POB_SRP in Playlists and has a value between '001' and '999'.
The file may store the picture data itself or the directory path referencing to the actual picture file. The reference path is limited to two directory layers.

© 2000 SD Association

CONFIDENTIAL

- **Display Order Mode**

Display Order Mode may be designated as shown below.

◊ **Table 3.3.3.1.2-2: Display Order Mode**

Sequential	All picture data are to be presented in the order when they are recorded in the DPLI_POB_SRP.
Random	Display order of each picture is selected randomly.
Shuffle	Similar to Random mode except no picture data is presented twice until all the picture data are presented

- **Display Timing Mode**

Display timing mode may be designated as shown below.

◊ **Table 3.3.3.1.2-3: Display Timing mode**

Slideshow	Synchronizes display timing of pictures and voice message presentation so that every display duration of each picture becomes the same. The duration is calculated by the following formula. Duration = (duration of voice message presentation) / (number of picture presentations)
Browsable	User may browse the picture display during the voice message presentation.

3.3.3.1.3 TKI Reference Information

The TKI entry number (between '1' and '999') describes the TKI references in the DPL_TK_SRP.

The TKI management flag shows the TKI's state. Each TKI manages the start, middle and the end of the Track as well as the state of usage.

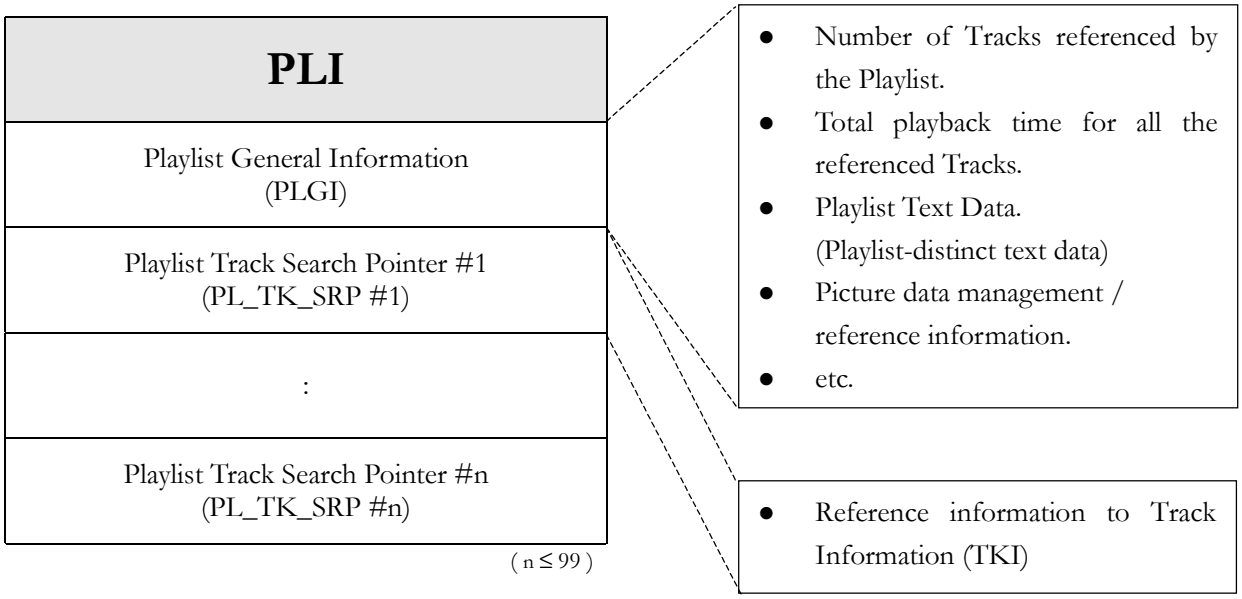
© 2000 SD Association

CONFIDENTIAL

3.3.3.2 Playlist Information (PLI)

User may edit a Playlist, which defines a playback sequence comprising maximum of 99 Tracks. Maximum of 99 Playlists may be created under the "SD_VCyyy" directory. The management information for Playlists is described in Playlist Information (PLI) which stores the reference information to the registered Track Information (TKI). If a single Track consists of multiple TKIs, only the first TKI is referenced.

◊ **Figure 3.3.3.2-1 : Playlist Information (PLI)**



For the description of text data, refer to **3.3.3.1 Default Playlist Information**.

3.3.3.2.1 TKI Reference Information

PL_TK_SRP in Playlist Information (PLI) describes reference information to the Track Information (TKI). TKI entry number between '1' and '999' is described as the reference information.

© 2000 SD Association

CONFIDENTIAL

3.3.4 Track (TK)

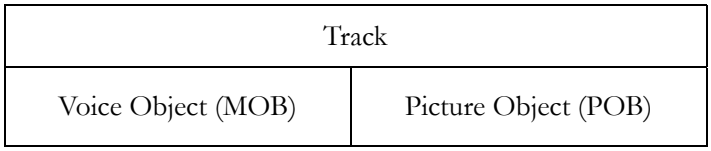
A Track is a unit of Message.

A Track consists of Voice Object (MOB: mandatory), Time Search Table (TMSRT: mandatory), Text Data (optional) and Picture Object (POB: optional).

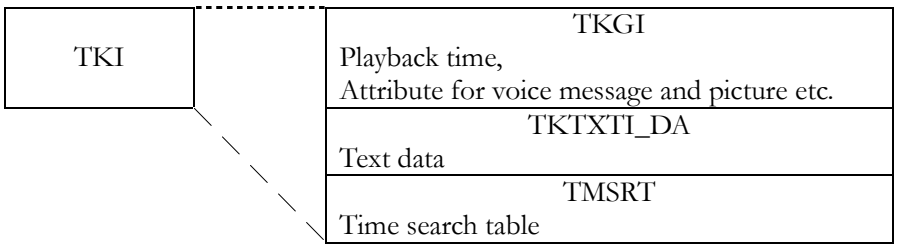
In Presentation layer, Track content consists of Voice Object (MOB) and Picture Object (POB) as shown in[Figure 3.3.4-1].

In navigation layer, the information associated with theTrack is stored in the Track Information (TKI). A TKI stores all information necessary to play back the Track Content, and such information includes links to Voice Object (MOB), Picture Object (POB) and Time Search Table (TMSRT) as shown in [Figure 3.3.4-2].

◊ **Figure 3.3.4-1 : Track Contents**



◊ **Figure 3.3.4-2: Track Information Structure**



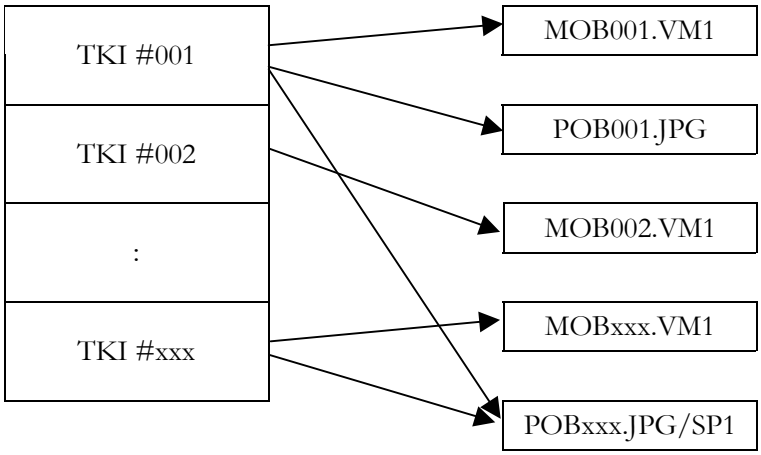
© 2000 SD Association

CONFIDENTIAL

3.3.4.1 Reference Information for Presentation Data

The reference information for an MOB uses the same number as of the TKIN in TKGI.
The reference information for POBs is described in a list in the TKGI. The POB number may be different from the number of TKIN.

◊ **Figure3.3.4.1-1 : TKI Numbers and File Numbers**



3.3.4.2 Attributes of the Presentation Data

- **MOB Attribute Information**

The attribute information for an MOB is described in TKI_MOB_ATR within TKGI.

◊ **Table 3.3.4.2-1 : TKI_MOB_ATR Information (Example of G.726)**

Voice Coding Mode	G.726
Bit Rate per channel	40, 32, 24, 16 kbit/s
Sampling Frequency	8 kHz
Channels	1 ch

- **POB Attribute Information**

The attribute information for a POB is described in TKI_POB_SRP within TKGI.

◊ **Table 3.3.4.2-2 : TKI_POB_SRP Information**

Number of pixels	96x96, 640x480(recommended) 160x120 ~ 1800x1200(optional)
Chrominance sampling	4:2:2, 4:2:0
Picture Coding Mode	JPEG

© 2000 SD Association

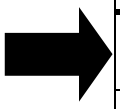
CONFIDENTIAL

3.4 Playback Procedure

3.4.1 Selecting a Playlist

A Playlist is selected for playback by the player.

◊ **Figure 3.4.1-1 : Selecting a Playlist**



Media-registered Playlist	Playlist Name (Sample)
Default Playlist (M)	*Meeting '00/01/23*
Playlist 1(O)	*Agenda A*
Playlist 2 (O)	*Agenda B*
Playlist 3 (O)	*Comments from Mr. C*
:	:

(M:Mandatory; O:Optional)

- The Default Playlist management information (DPLI) is described in the Playlist Manager (PLMG). Following the DPLI, the Playlist management information is described to a maximum of 99. A Playlist number is the order that one is described in the PLMG.
- At the Initial Access, a Playlist is to be selected in the following precedence:

1) Playlist, which is described in PLMG_RSM_PL in PLMG, is selected.
   (Playlist which was played back at the last time is described in PLMG_RSM_PL)
2) Playlist, which is described in PLMG_AP_PL in PLMG, is selected
   (Playlist which is played back at initial access is described in PLMG_AP_PL,)
3) DPLI is selected.

Note: When the Player does not have recording function, it is not necessary to read PLMG_RSM_PL and PLMG_AP_PL.

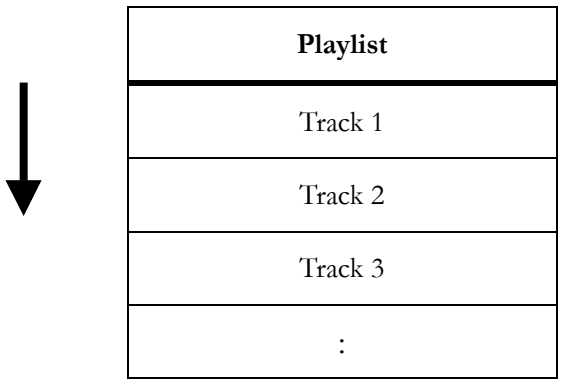
© 2000 SD Association

CONFIDENTIAL

3.4.2 Playback Order

Once a Playlist is selected, Tracks are played back. The reference information (DPL_TK_SRP and PL_TK_SRP) for Track Information (TKI) is described in the order of playback in DPLI and PLI.

◊ **Figure 3.4.2-1 : Playback Order of Tracks**

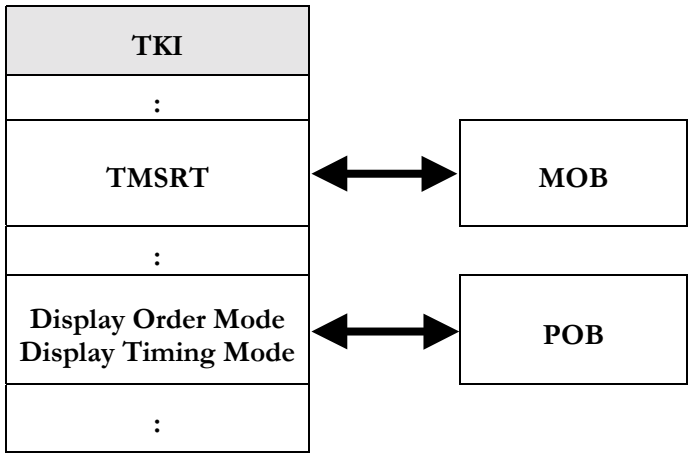


- Tracks are numbered in the order described in DPL_TK_SRP and PL_TK_SRP. (The maximum value in the Default Playlist is 999.)
- Only when a DPL_TK_SRP flag indicates the start of a Track in DPLI, the TKI is processed as a Track.

3.4.3 Voice Playback

Voice playback is conducted using the MOB and the TMSRT in the TKI. The POB is presented based on the Display Timing Mode and Display Order Mode.
As for the operation based on the Voice Object (MOB) and TMSRT, refer to **3.7.1 Playback Operation**.

◊ **Figure 3.4.3-1 : TKI Reference Information**



© 2000 SD Association

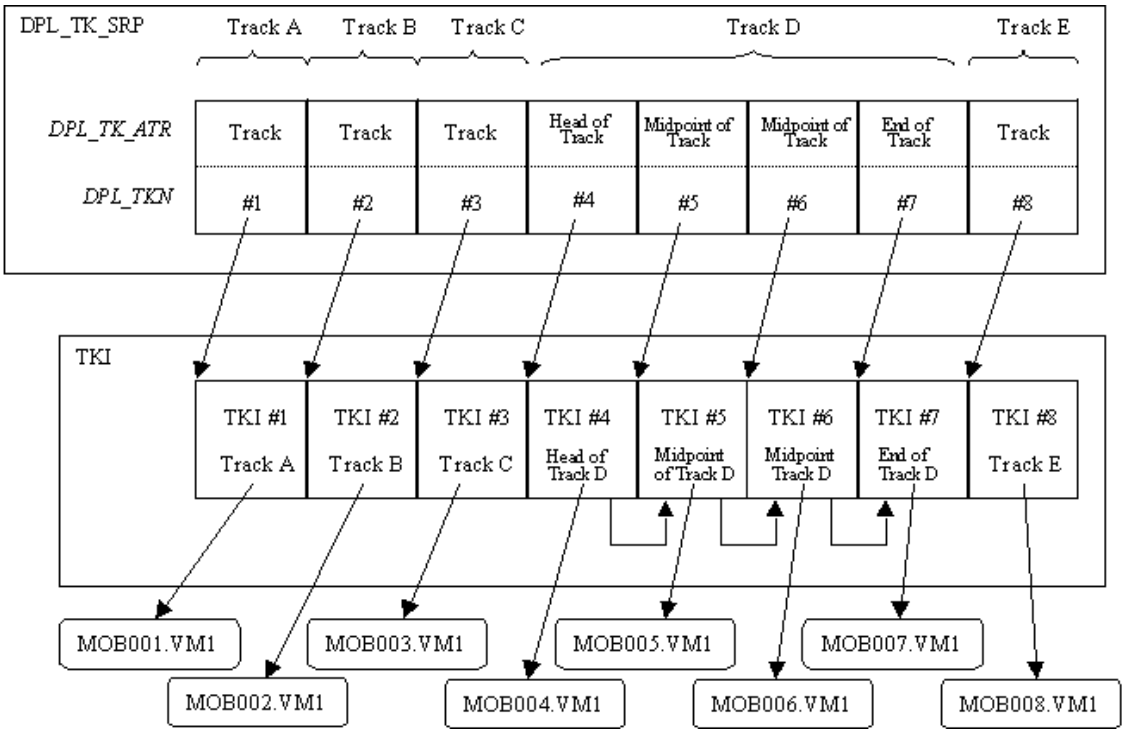
CONFIDENTIAL

3.5 Editing Procedure of Tracks

Track Information (TKI) and file management information are described in Default Playlist Information (DPLI) in Playlist Manager. DPL_TK_ATR in DPLI indicates the condition of TKIs. DPL_TKN indicates numbers for TKIs and files.

Each operation flow of DPLI, TKI and file, mainly during editing operations, is explained below.

◊ **Figure 3.5-1: Example of DPL_TK_ATR, TKI and file structure**



3.5.1 Storing a Track by Dividing

As shown in [Figure 3.5-1], assume that each of Track A, B, C and E corresponds to one TKI. If a Track becomes longer than capacity of one unit, it may use multiple TKIs as shown in Track D.

The length of Time Search Table (TMSRT) of a TKI is fixed and limited to 512 bytes. Therefore, if TMSRT of a Track requires more than 512 bytes, it is necessary to use multiple TMSRTs described in multiple TKIs.

When a Track corresponds to multiple TKIs, DPL_TK_SRPs for these TKIs shall be described sequentially; and furthermore, status information of each TKI shall be described in each DPL_TK_ATR.

© 2000 SD Association

CONFIDENTIAL

◊ **Table 3.5.1-1: DPL_TK_ATR attribute**

Track	One Track is in one TKI
Head	One Track lies over multiple TKIs. Current TKI is head.
Midpoint	One Track lies over multiple TKIs. Current TKI is midpoint.
End	One Track lies over multiple TKIs. Current TKI is end.
Unused	TKI is not used yet. TKI area is available. It occurs when deleting.
Un-alloc	TKI is not used yet. TKI area is not available, and is at initial state.

In an example of Track D shown in [Figure 3.5-1], Track is divided and stored in multiple TKIs starting at TKI#4 and ending at TKI#7. Each TKI has a link to the next TKI. Each state that indicates head of the Track, midpoint of the Track, or end of the Track is described in DPL_TK_ATR.

Note:

The attribute information, which describes the TKI's state, is also defined at TKI_BLK_ATR in TKGI. These two information shall be equal to each other.

When a Track is composed of two or more TKIs as in the case of Track D, only the first TKI is effective to TKI_POB_SRP.

© 2000 SD Association

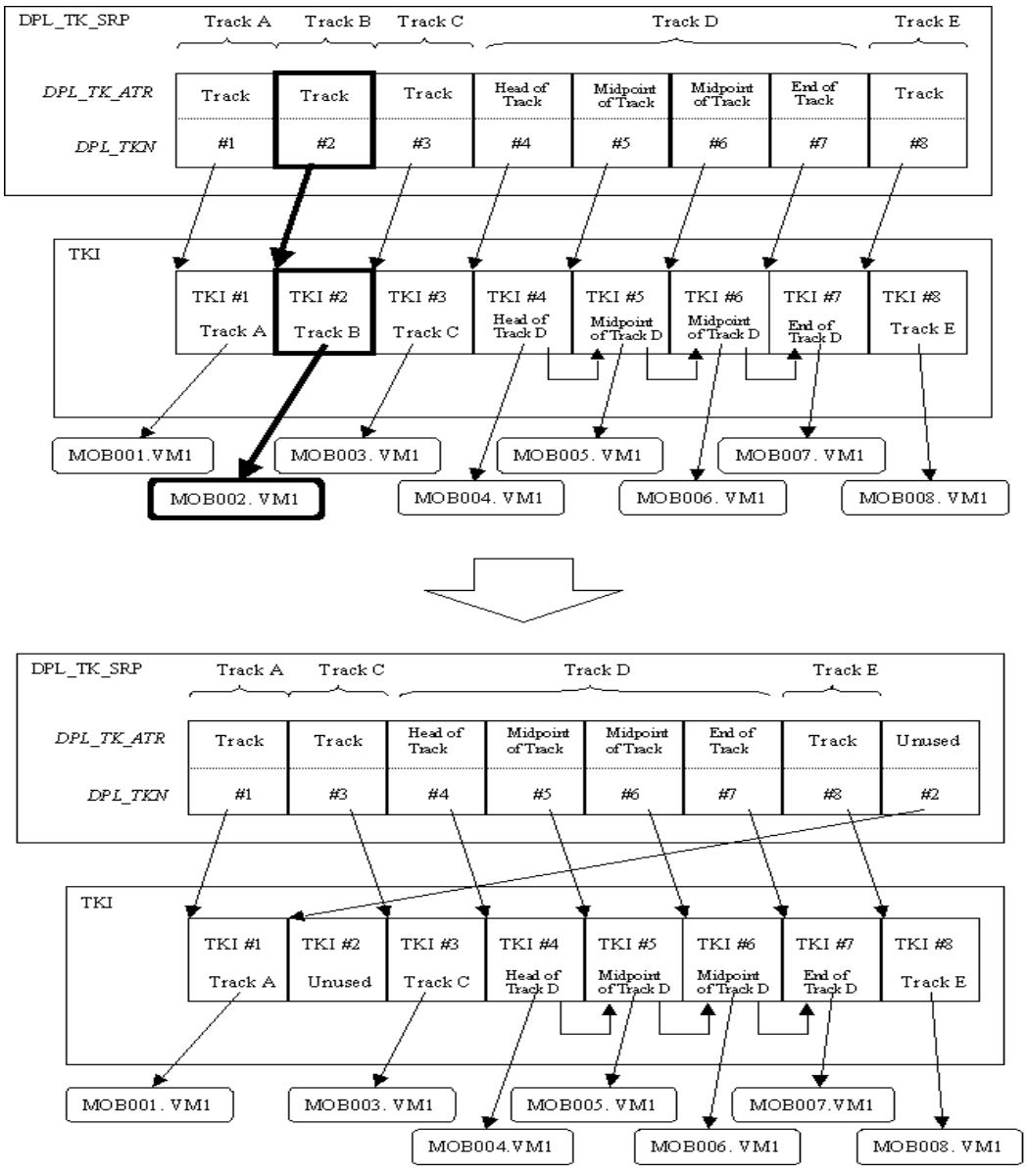
CONFIDENTIAL

3.5.2 Deleting a Track

When user deletes a Track from a Playlist, delete only reference pointer to TKI in corresponding Playlist. When a Track is actually deleted from a SD Memory Card, it is necessary to delete corresponding Track from the Default Playlist. Deleting is executed according to the following procedure:

1. Delete entry of DPL_TK_SRP for user-specified Track. Deleting manner is as follows: Set DPL_TK_ATR to "unused", then move it to the end of DPL.
2. Set TKI to "unused" state, and make DPL_TKN reference to the TKI.
3. Delete MOB file with the number which DPL_TKN indicates. When POB file is specified in TKI_POB_SRP, the reference pointer of TKI_POB_SRP is deleted. However, POB file need not be deleted.
4. If a Track corresponds to multiple TKIs and files, delete multiple corresponding TKIs and files as well.

◊ **Figure 3.5.2-1: Deleting a Track**



© 2000 SD Association

3.5.3 Recording a Track

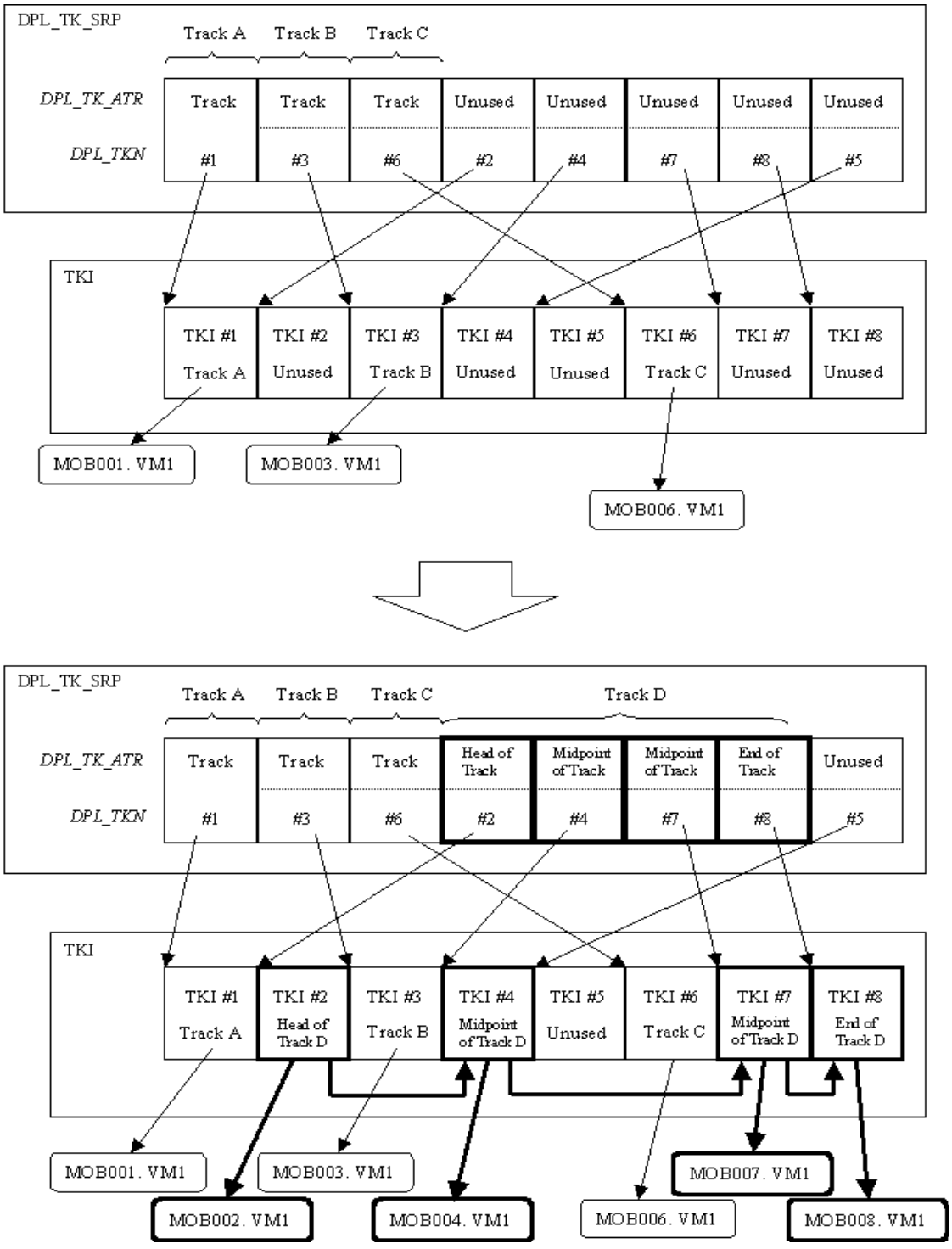
When user records a message on each track by Track, information regarding the new Track is added only to the Default Playlist. Actual operation is executed according to the following procedure:

1. When user starts recording a message, an unused entry is retrieved from DPL_TK_SRP in DPL.
2. Determine TKI entry and MOB file number from the unused entry.
3. Record message data in the MOB file.
4. Write Time Search Table of the message to TMSRT in TKI.
5. When Time Search Information exceeds the maximum size of Time Search, find an "unused" DPL_TK_SRP, and determine TKI and MOB file for it.
6. Data in previous TKI except TMSRT are copied to the new TKI, and set a link from previous TKI.
7. Repeat an operation from procedure 3 to 6. When recording is completed, the flags indicating head of the Track, midpoint of the Track, and end of the Track are properly described in DPL_TK_ATR.

As for MOB file structure and TMSRT structure, refer to **3.6 Presentation Data Structure.**

© 2000 SD Association

◊ **Figure 3.5.3-1: Recording a Track D**



© 2000 SD Association

3.5.4 Changing Order of Tracks

When user changes the order of Tracks in Default Playlist, changing operation is executed according to the following procedure:

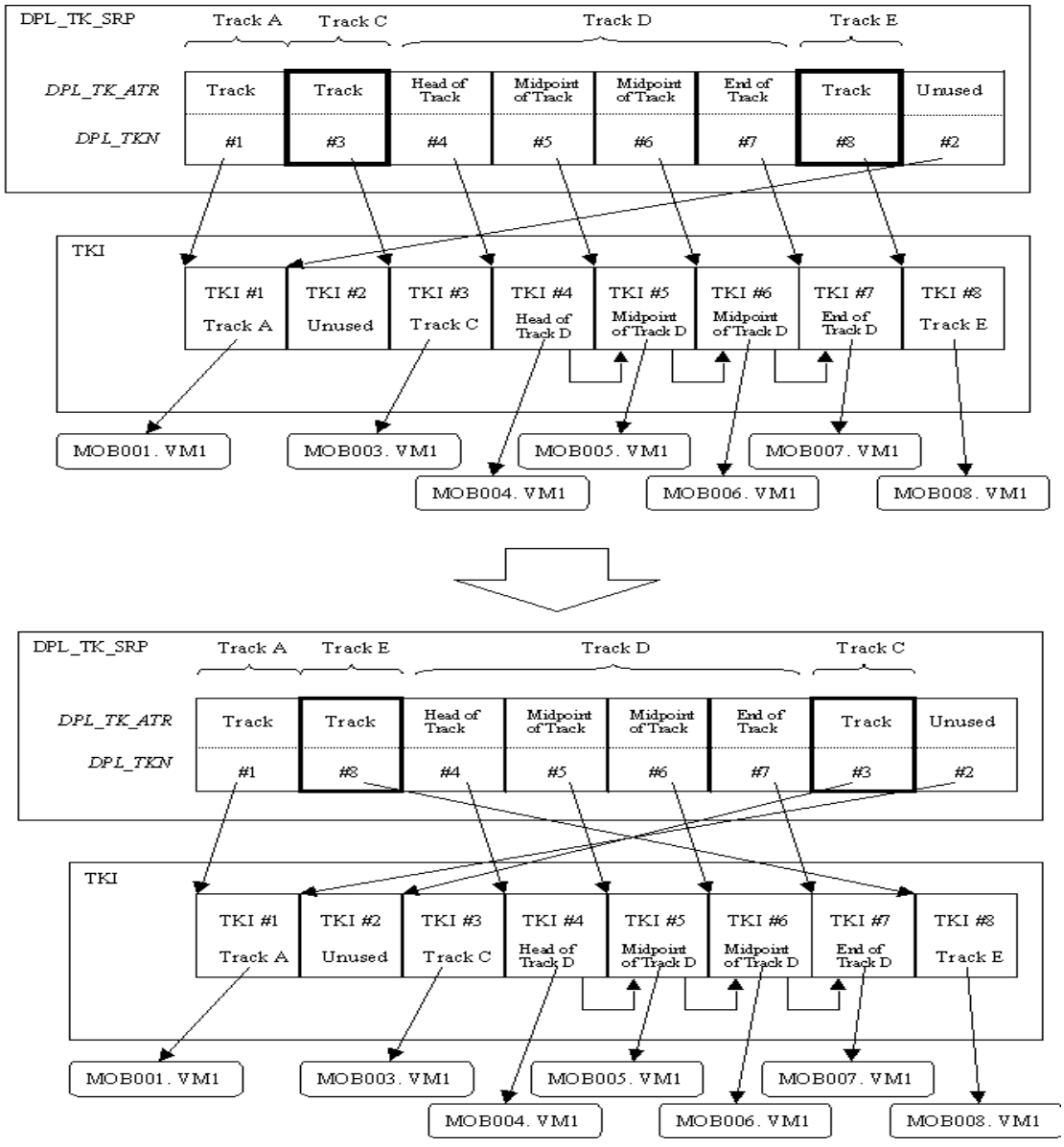
1. Select two Tracks to be exchanged.
2. Exchange the entries of DPL_TK_SRPs.
3. If a Track consists of multiple TKIs and multiple MOB files, keep the order of DPL_TK_SRP entries, which correspond to a Track, become sequential.

Note: Even when changing order of Tracks in Default Playlist, order of Tracks in Playlist referring corresponding Tracks does not have any side effect.

When user changes order of Tracks in Playlist, it is executed simply by changing reference pointer.

In this case, order of Tracks in the Default Playlist does not have any side effect

◊ **Figure 3.5.4-1: Changing order of Tracks between Track C and Track E**



© 2000 SD Association

3.5.5 Combining Tracks

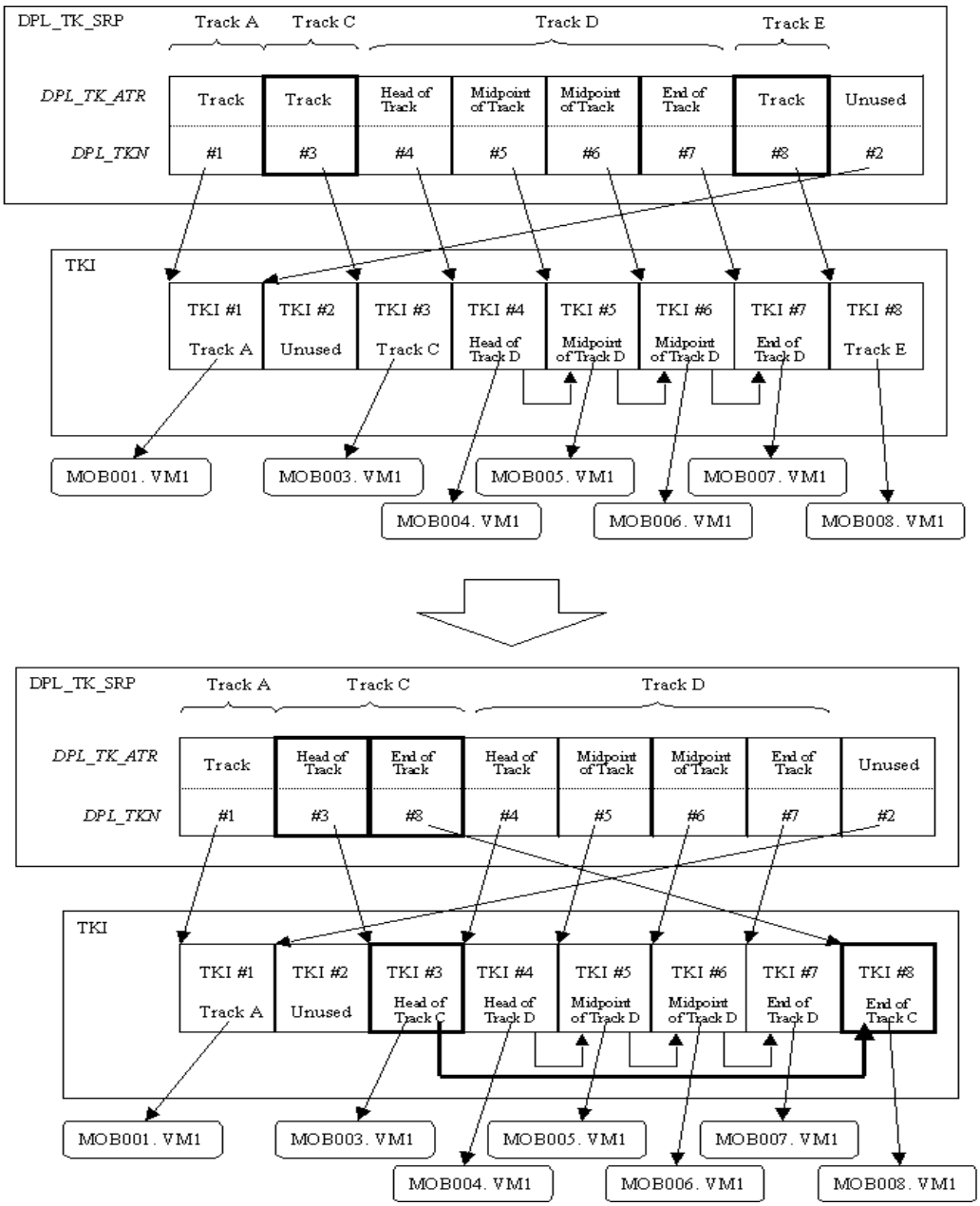
Processing steps of the Default Playlist and TKMG to combine two Tracks into one Track are executed according to following procedure.

1. Locate DPL_TK_SRP of user-specified 2 Tracks sequentially
2. Rewrite DPL_TK_ATRs into head of the Track and end of the Track respectively, in order to describe the state at which Tracks are stored as one Track.
3. Change a link in the preceding TKI to the succeeding TKI number of the combined Track.

As for conditions for combining, refer to **3.8 Conditions for Editing**.

Note: The POB in the next Track may be merged with the POB in the previous Track.

◊ **Figure 3.5.5-1: Combining Track C and Track E**



© 2000 SD Association

3.5.6 Dividing a Track

Processing steps of Default Playlist and TKMG to divide one Track in order to create two Tracks are executed according to the following procedure:

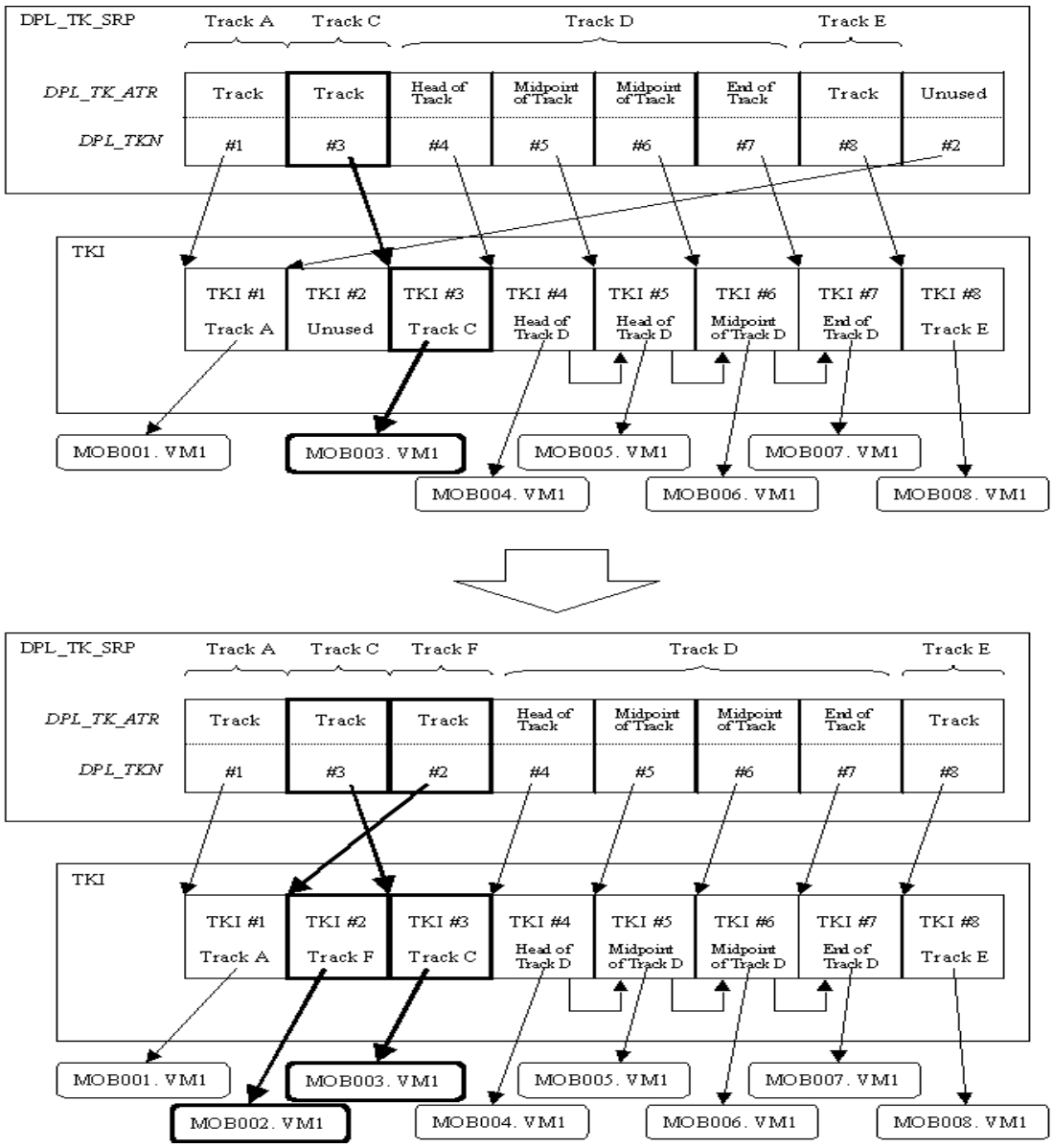
1. Find "unused" entry among DPLs, and then find "unused" TKI.
2. Move "unused" DPL_TK_SRP entry to immediately after the Track is divided.
3. Find TKI based on DPL_TK_SRP entry to be divided, and copy Track data except Time Search Table to the "unused" TKI
4. Divide Time Search Table of TKI and MOB file, and then allocate them to "unused" TKI and MOB file.
5. Change "unused" DPL_TK_ATR, TKI to DPL_TK_ATR, TKI to "busy" states.

As for dividing Time Search Table and MOB file, refer to **3.6 Presentation Data Structure**.

Note: As a result of this operation, meaning of TKI in TKMG changes. However, reference pointer in Playlist referring to TKI for a Track to be divided does not need to add TKI for a newly created Track. All of POB information and Text information in the newly created TKI may be copied from the previous TKI.

© 2000 SD Association

◊ **Figure 3.5.6-1: Dividing Track C**



© 2000 SD Association

3.6 Presentation Data Structure

3.6.1 Structure of Voice Object (MOB)

Voice Object (MOB) has a hierarchical structure as shown in [Figure 3.6.1-1].

- MOB_CODEC_FRAME :

  The minimum component of an MOB. An MOB_CODEC_FRAME has the same structure as a codec frame and its structure is dependent on the applied codec. In the case where sample-based codec like ITU-T G.726 ADPCM is applied, an MOB_CODEC_FRAME consists of a predetermined number of encoded samples as shown in [Table3.6.1-1].

◊ **Table 3.6.1-1 : Number of encoded samples per MOB_CODEC_FRAME**

Sampling Frequency	Number of encoded samples per MOB_CODEC_FRAME
	G.726
8kHz	40

- MOB_FRAME :

  The minimum unit for editing operations on an MOB. An MOB_FRAME consists of a header section and a given number of consecutive MOB_CODEC_FRAMEs. The structure of the header section is dependent on the applied codec. The number of MOB_CODEC_FRAMEs is fixed for each codec as shown in [Table 3.6.1-2].

◊ **Table 3.6.1-2 : Number of MOB_CODEC_FRAMEs**

Sampling Frequency	Number of MOB_CODEC_FRAMEs	
	G.726	AMR
8kHz	20	5

In the case where the number of total bits in an MOB_FRAME is not a multiple of 8 bits, a certain number of '0's are padded to the end of the MOB_FRAME in order to adjust the number of total bits to a multiple of 8 bits (Byte alignment).

- MOB_ELEMENT :

  An MOB_ELEMENT is an aggregate of multiple contiguous MOB_FRAMEs managed by Time Search Table (TMSRT). It usually stores a fixed number of MOB_FRAMEs, but some MOB_ELEMENTs may have less than its fixed number of frames due to dividing the MOB, etc.

© 2000 SD Association

◊ **Table 3.6.1-3 : Number of MOB_ELEMENTs**

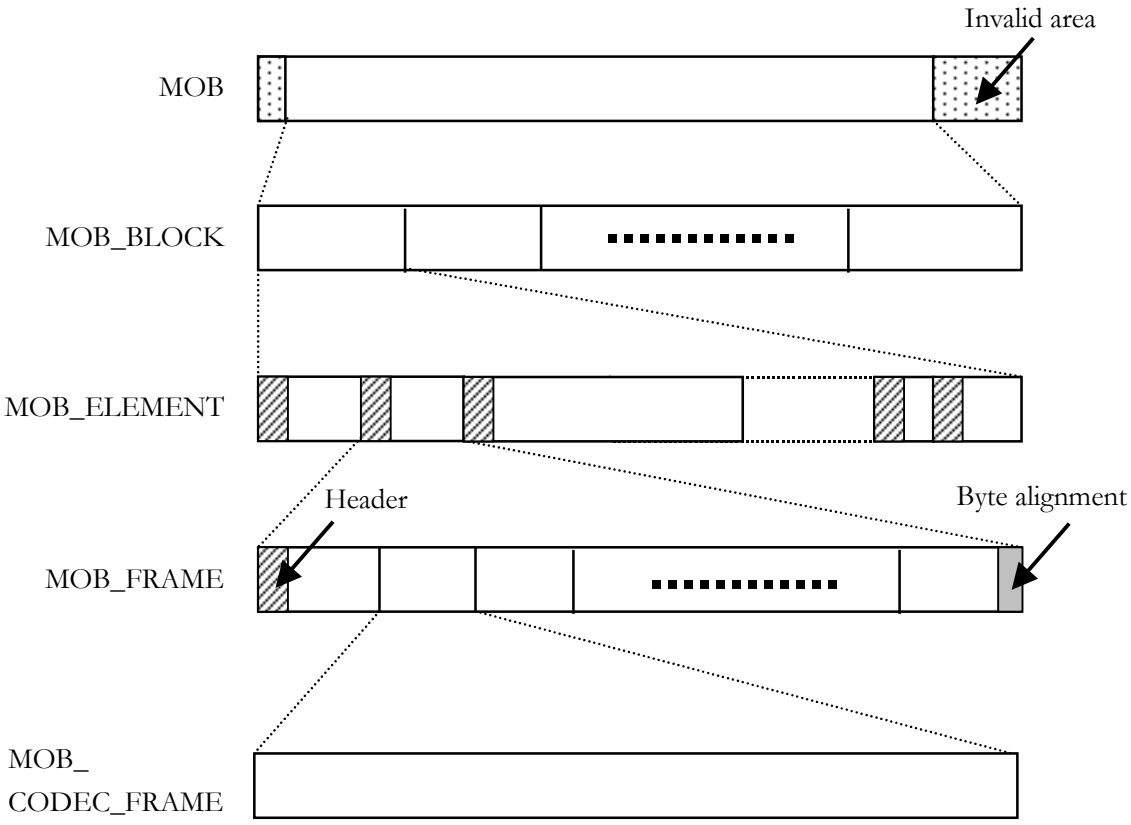
Sampling Frequency	Number of MOB_FRAMEs	
	G.726	AMR
8kHz	10*N	10*N

N indicates the presentation time duration of an MOB_ELEMNT in second.

- MOB_BLOCK :
  An MOB_BLOCK is an area which stores contiguous valid MOB_ELEMENTs in a Voice Object (MOB) managed by TKI are stored. Only one MOB_BLOCK exists in an MOB file.

- MOB :
  TKI manages MOB.

◊ **Figure 3.6.1-1 : Structure of MOB**



© 2000 SD Association

○ Structure of MOB_FRAME Header

(1) MOB_FRAME Header

An MOB_FRAME Header consists of one MOB_FRAME Fixed Length Header (MOBFLH) and one MOB_FRAME Variable Length Header (MOBVLH).

◊ **Figure 3.6.1-2 : Structure of MOB_FRAME Header (MOBFH)**

MOB_FRAME Header (MOBFH)

MOB_FRAME Fixed Length Header (MOBFLH)	4 bytes
MOB_FRAME Variable Length Header (MOBVLH)	0 to 31 bytes

(2) MOB_FRAME Fixed Length Header

MOB_FRAME Fixed Length Header (MOBFLH) is a mandatory part of the MOB_FRAME Header. The format of MOBFLH, 4-byte in length, is shown in [Table 3.6.1-4].

◊ **Table 3.6.1-4 : Format of MOB_FRAME Fixed Length Header (MOBFLH)**

	No. of bits	Contents
syncword	12 bits	'1111 1111 1111'
frame length	13 bits	Size of MOB_FRAME in bytes
MOBVLH present	1 bit	'0' no MOBVLH present '1' MOBVLH present
MOBVLH length	5 bits	Size of MOBVLH in bytes
	1 bit	reserved
Total	32 bits	

(3) MOB_FRAME Variable Length Header

If an MOBVLH present bit in MOB_FRAME Fixed Length Header (MOBFLH) is set to '1', the applied codec requires additional specifications described in this field. The number of header bits and header structure are dependent on the applied codec. For the codecs specified by the coding mode in the TKI_MOB_ATR filed, the structure of MOB_FRAME Variable Length Header (MOBVLH) is defined for each codec as shown in [Table 3.6.1-5]. If a specified coding mode is vendor dependent ('1111'), the MOBVLH filed shall be vendor dependent as well. In the case where the number of total bits of MOB_FRAME Variable Length Header is not a multiple of 8 bits, a certain number of '0's are padded to the end of the MOB_FRAME Variable Length Header in order to adjust the number of total bits to a multiple of 8 bits (Byte alignment).

© 2000 SD Association

CONFIDENTIAL

(4) Structure of MOB_FRAME Variable Length Header

For the codecs specified by the coding mode in the TKI_MOB_ATR filed, the structure of MOB_FRAME Variable Length Header (MOBVLH) is defined for each codec as shown in [Table 3.6.1-5].

◊ **Table 3.6.1-5 : Structure of MOB_FRAME Variable Length Header (MOBVLH)**

Codec	MOBVLH present	MOBVLH length (bytes)	Structure
G.726	'0'	0	no MOBVLH required
AMR	'0'	0	no MOBVLH required

New codecs which require variable length header might be applied in this specification. MOB_FRAME Variable Length Header (MOBVLH) is provided for such future extensions.

© 2000 SD Association

CONFIDENTIAL

3.6.2 Structure of Picture Object (POB)

Picture Information managed in SD-Voice Specification is called Picture Object (POB). A POB is stored in a POB file.

There are two types of POB files:

(1) Reference pointer to JPEG file

File name is "POBxxx.SP1" where POB_ATR is set to '1'.

In the "POBxxx.SP1", there is a reference pointer to a JPEG file. JPEG file name may contain a directory name, and the directory name may have a maximum of 2 hierarchical directories.

Directory name is described in absolute path (e.g. DirectoryName¥DirectoryName¥FileName).

This file name and JPEG file format shall comply with DCF specifications.

(2) JPEG picture

File name is "POBxxx.JPG".

It is JPEG picture of Exif V2.1 specifications (subset)

Following formats shown in [Table 3.6.2-1] are JPEG picture formats to be operated on SD-Voice.

◊ **Table 3.6.2-1: JPEG format**

File format		Exif V2.1
Compression		JPEG
Thumbnail data format	Number of Pixels	96x96, 160x120
	Aspect ratio	4:3
	Chrominance sampling	4:2:2
	Compression ratio	Not specificated
	Huffman table	Typical Table in JPEG standard (ISO/IEC 10918-1)
Main picture data format	Number of Pixels	96x96(recommended) 640x480(recommended) 160x120 ~ 1800x1200(optional)
	Aspect ratio	4:3(recommended) Others(optional)
	Chrominance sampling	4:2:2, 4:2:0
	Compression ratio	Not specificated
	Huffman table	Typical Table in JPEG standard (ISO/IEC 10918-1)
	Color space	sRGB

As shown in [Table 3.6.2-1], thumbnail data and main picture data are restricted in the file format of Exif V2.1 specifications. This restriction is the same as JPEG file format in DCF specifications.

© 2000 SD Association

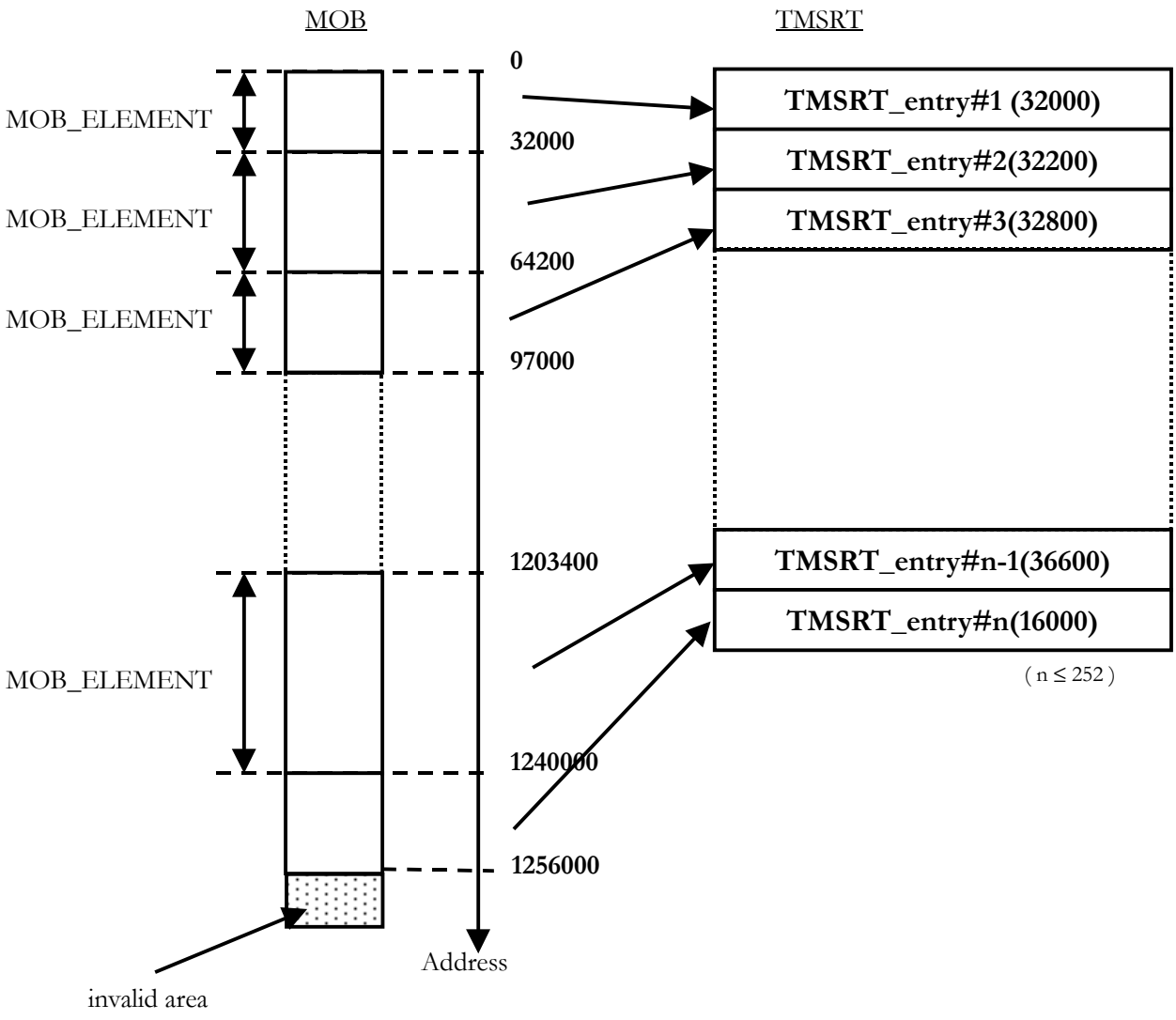
CONFIDENTIAL

3.6.3 Time Search Table (TMSRT)

TMSRT is information that indicates the size of MOB_ELEMENT in an MOB file and is contained in TKI. TMSRT_entry is a component of TMSRT.

[Figure 3.6.3-1] shows an example of TMSRT.

◊ **Figure 3.6.3-1 : an example of TMSRT**



© 2000 SD Association

CONFIDENTIAL

3.6.4 Block Information Table (BIT)

Block Information Table (BIT) manages MOB_BLOCKs in MOB and consists of the following entries.

(1) Data_Offset: Size of invalid area at the head of MOB file

(2) SZ_Data: Size of MOB_BLOCK

(3) TMSRTE_Ns: Number of TMSRT_entries in the MOB_BLOCK.

(4) FNs_1st_TMSRTE: Number of frames in the first TMSRT_entry.

(5) FNs_Last_TMSRTE: Number of frames in the last TMSRT_entry.

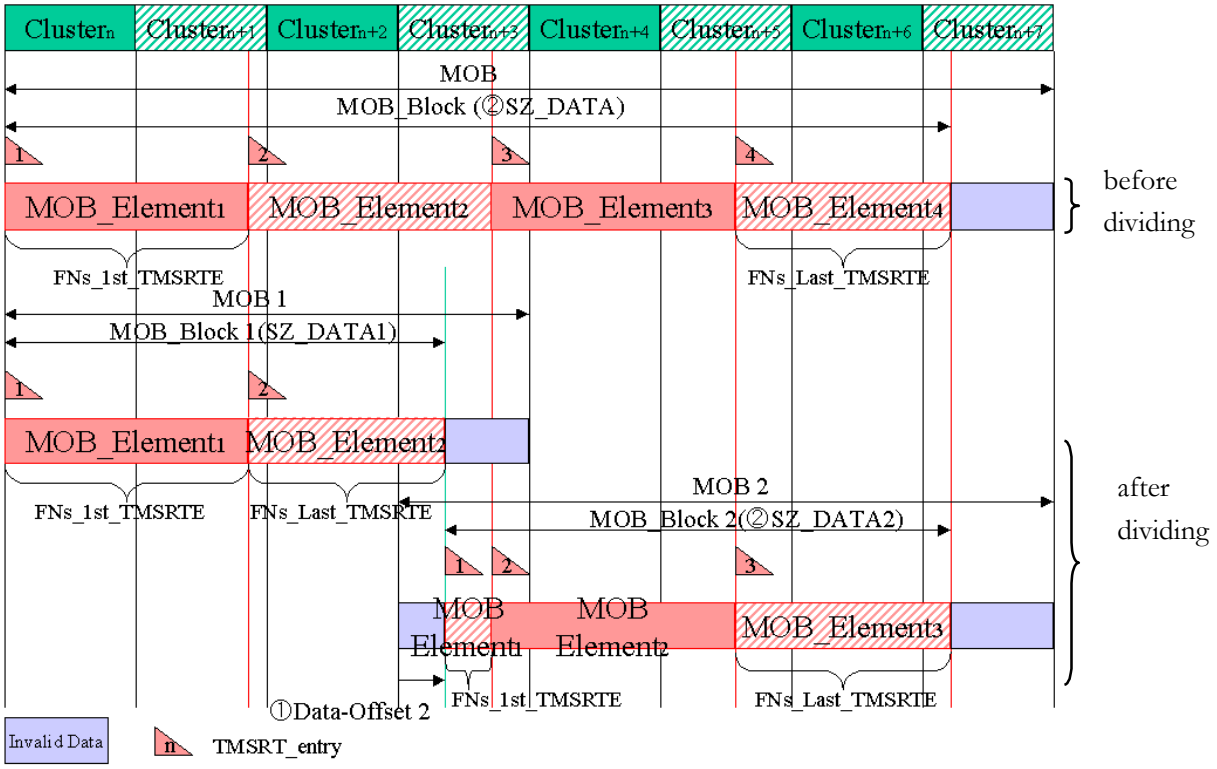
(6) FNs_Middle_TMSRTE: Number of frames in the middle TMSRT_entry.

(7) TIME_LENGTH: Presentation time duration of MOB_ELEMENT in msec.

As for details of "Cluster", refer to **Part2: FILE SYSTEM SPECIFICATION**.

[Figure3.6.4-1] shows the relationship of entry between MOB and BIT.

◊ **Figure 3.6.4-1 : BIT**

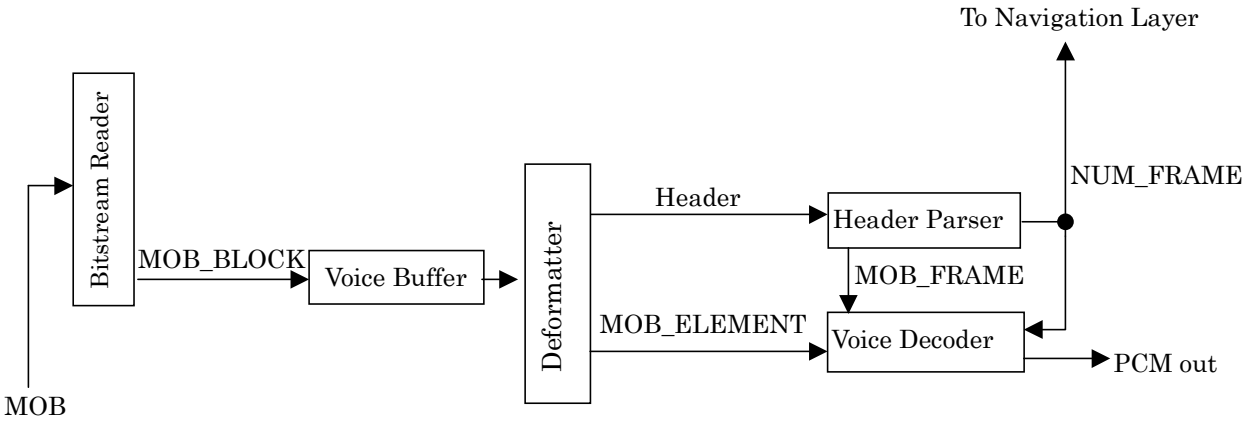


© 2000 SD Association

CONFIDENTIAL

3.7 System Model

◊ **Figure 3.7-1 : System Model**



NUM_FRAME : Number Of MOB_FRAME Per MOB_ELEMENT

Bitstream Reader	:	reads the Voice Object (MOB) from the storage medium according to the Block Information Table (BIT).
Voice Buffer	:	is the temporal memory (typically FIFO) to store the Voice Stream (MOB_BLOCK)
Deformatter	:	deformats the Voice Stream (MOB_BLOCK) into the Header and the MOB_ELEMENT.
Header Parser	:	parses the Header into the MOB_FRAME information.
Voice Decoder	:	decodes the MOB_FRAME into the PCM data according to the Header information.

3.7.1 Playback Operation

The procedure of normal playback is as follows:

(1) Obtain Data_Offset, SZ_DATA, and FNs_1st_TMSRT from BIT.

(2) Submit the number of FNs_1st_TMSRTE of MOB_FRAMEs from Data_Offset into Decoder.

(3) Increment the elapsed playback time by (playback time of FNs_1st_TMSRTE * 1 MOB_FRAME).

(4) Subsequently, increment the elapsed playback time by playback time of each MOB_FRAME whenever an MOB_FRAME is submit into Decoder.

(5) Repeat step 4 until the number of data put into Decoder becomes equal to SZ_DATA and, at the same time, the number of frames put into Decoder after the last TMSRT_entry becomes equal to FNs_Last_TMSRTE.

3.7.2 Special Playback Operation

An example of special playback, "skip-playback" is described herein after. The detailed procedure is as follows:

(1) Obtain Data_Offset, SZ_DATA and FNs_1st_TMSRTE from BIT.

(2) Skip data reading amount of the Data_Offset counting from the head of MOB and put MOB_FRAMEs corresponding to "skip-playback" time into Decoder.

© 2000 SD Association

CONFIDENTIAL

(3) Increment the elapsed playback time by the equivalent of skip playback time.
(4) Detect the location of a new entry frame after the increment within the TMSRT entry.
(5) Detect the TMSRT_entry which contains the location incremented by the number of MOB_FRAMEs equivalent to "skip-intervals" from the TMSRT.
(6) Check MOB_ELEMENT in the TMSRT_entry and detect the location of the frame just after "skip-interval" within the MOB_ELEMENT.
(7) Put MOB_FRAMEs corresponding to "skip-playback" time into Decoder from the detected location of the frame as indicated above.
(8) Repeat step 3 to 7 until the end of "skip-playback".

3.7.3 Recording Operation

(1) TMSRT_entry is incremented by one for every number (FNs_Middle_TMSRTE) of frames composing MOB_ELEMENT.
(2) Since the size of TMSRT is 512 bytes, and one TMSRT_entry occupies 2 bytes, 252 entries can be stored in it.
(3) TMSRT_entry is added for every period of time corresponding to the number of frames composing MOB_ELEMENT. When the total number of TMSRT_entries reaches 252, a new TKI (TMSRT) is created and at the same time the original BIT is replicated. TMSRT_entry is added to the new TMSRT for every period of time corresponding to the number of frames composing MOB_ELEMENT.
(4) Step 3 is repeated until the end of input signal.

3.7.4 Editing Operation

3.7.4.1 Editing Tracks

When two MOBs are combined, BIT and TMSRT are not modified and each TKI for the two MOBs is linked each other. Refer to **3.5.5 Combining Tracks**.

© 2000 SD Association

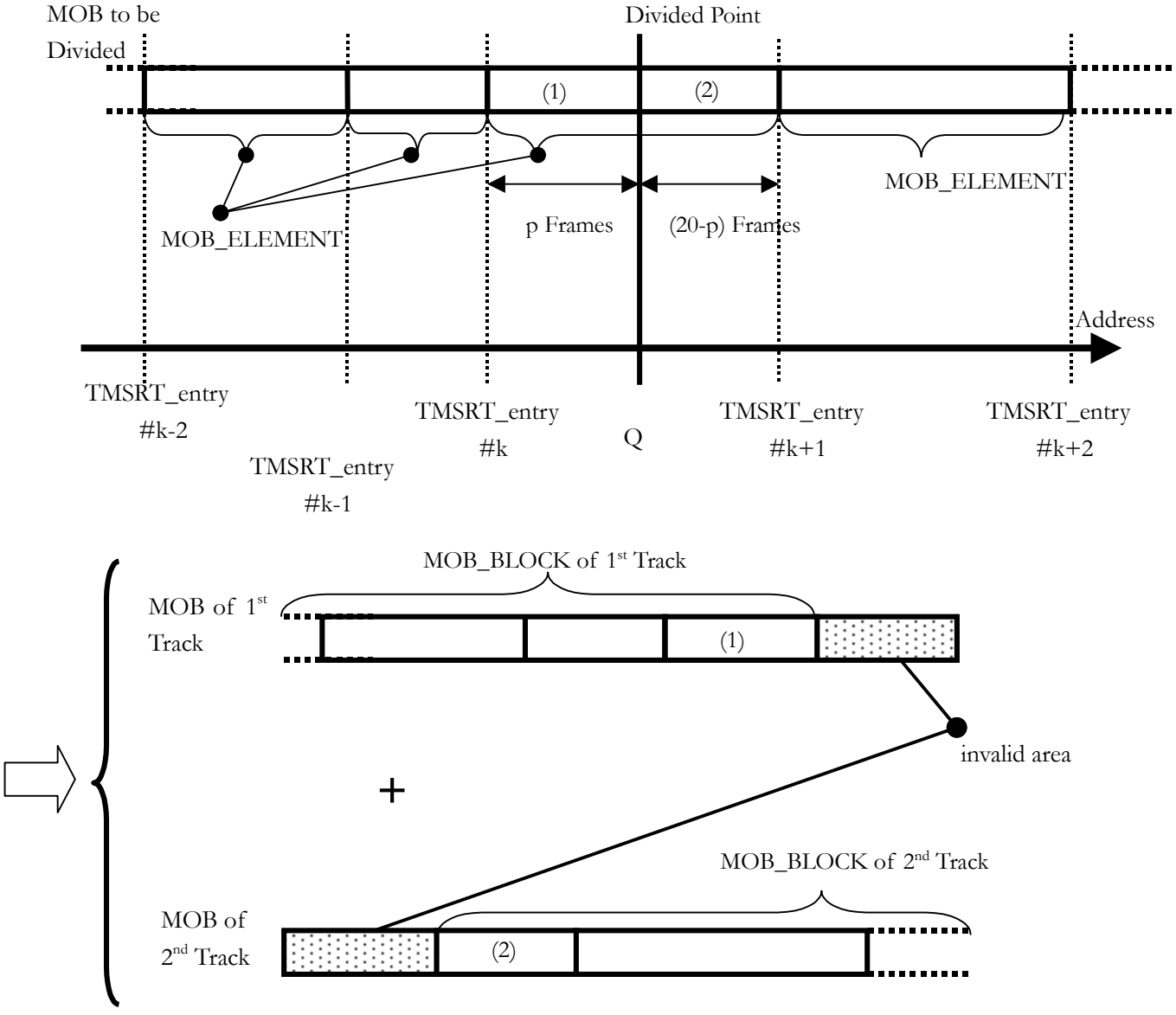
CONFIDENTIAL

3.7.4.2 Dividing Tracks

This section describes a divide of Tracks. One example is that an MOB is divided into two at address Q between the *k*th TMSRT_entry (TMSRT_entry #k) and the *k+1*th TMSRT_entry (TMSRT_entry #k+1) in the MOB which is to be divided. The first part and the last part of the MOB before the divide are called 1st Track and 2nd Track, respectively, even after the divide for convenience of explanation.

MOB is divided into two pieces of MOB, and each of them contains one MOB_BLOCK. [Figure 3.7.4.2-1] shows an example of MOB_BLOCK after the divide containing multiple TMSRT_entries. This is the case where MOB is divided at the *p*th frame from the head of MOB_ELEMENT containing address Q.

◊ **Figure 3.7.4.2-1 : Example of MOB Divide (FNs_Middle_TMSRTE=20)**



© 2000 SD Association

CONFIDENTIAL

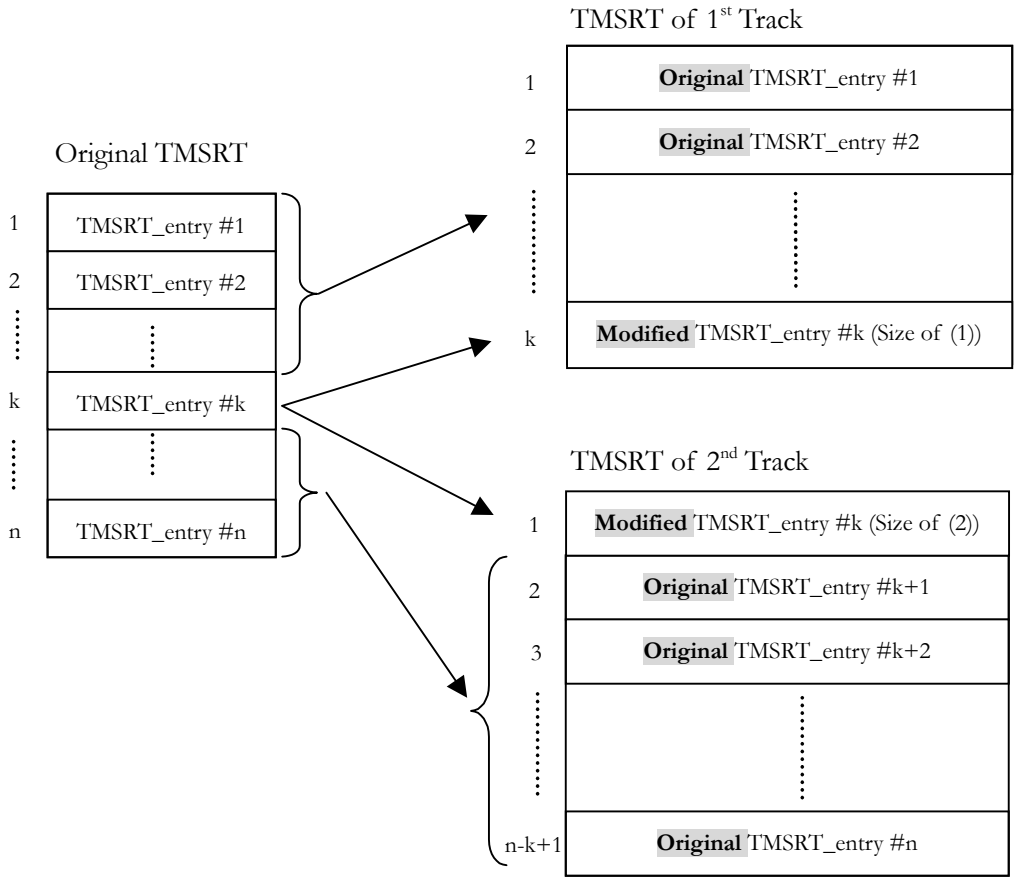
Changes of TMSRT and BIT due to the divide shown in [Figure3.7.4.2-1] are as follows:

(i) TMSRT

TMSRT of the 1st Track contains the initial entry to the *k*th entry of TMSRT of MOB before the divide. TMSRT of the 2nd Track contains the *k+1*th entry to the last entry (TMSRT_entry #n) of the TMSRT of the MOB before the divide. In addition, a single TMSRT_entry is newly added at the head.

Note: When a Track is divided along TMSRT_entry boundary, a single TMSRT_entry is not added at the head of the 2nd Track's TMSRT.

◊ **Figure 3.7.4.2-2 : Example of dividing TMSRT for Figure 3.7.4.2-1**



© 2000 SD Association

CONFIDENTIAL

(ii) BIT

Changes in BIT of the 1st Track are as follows:

SZ_DATA: data length to the divided point Q

TMSRTE_Ns: k

FNs_Last_TMSRTE: p (frames)

Changes in BIT of the 2nd Track are as follows:

Data_Offset: R

SZ_DATA: original SZ_DATA - data length to the divided point Q

TMSRTE_Ns: n-k+1 (k of the 1st Track plus n-k+1 of the 2nd Track equals n+1, the original value.)

FNs_1st_TMSRTE: 20-p (frames)

FNs_Last_TMSRTE: 12 (equal to the original one)

Note: If either of MOB_BLOCKs after divided does not contain TMSRT_entry, FNs_1st_TMSRTE in the corresponding BIT is the number of MOB_FRAMEs that is included in this MOB, and TMSRTE_Ns is 1.

◊ **Figure 3.7.4.2-3 : Example of dividing BIT for Figure 3.7.4.2-1**

Original BIT

Data_Offset	0
SZ_DATA	52428
TMSRTE_Ns	n
FNs_1st_TMSRTE	20
FNs_Last_TMSRTE	12
FNs_Middle_TMSRTE	20

BIT of 1st Track

Data Offset	0
SZ DATA	**Q**
TMSRTE Ns	**k**
FNs 1st TMSRTE	20
FNs Last TMSRTE	**p**
FNs Middle TMSRTE	20

BIT of 2nd Track

Data Offset	**R**
SZ DATA	**52428-Q**
TMSRTE Ns	**n-k+1**
FNs 1st TMSRTE	**20-p**
FNs Last TMSRTE	12
FNs Middle TMSRTE	20

© 2000 SD Association

CONFIDENTIAL

3.8 Conditions for Editing

3.8.1 Conditions for Combining

The conditions for combining of MOBs are as follows.

(1) Voice attributes (Voice Coding Mode, bitrate, Sampling frequency, and the number of channels) shall be equal between preceding and subsequent MOBs.

(2) Combination points shall satisfy the following conditions.

When MOB is divided into two categories as described below:

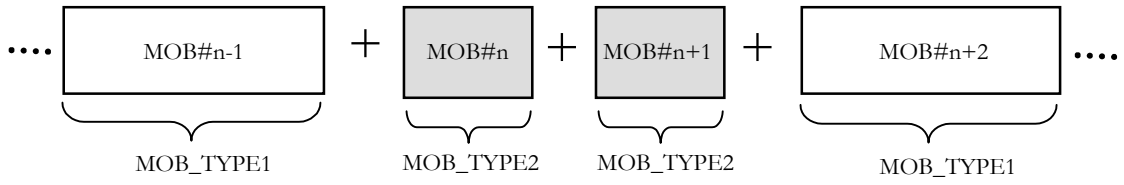
MOB_TYPE1 … An MOB which has at least one MOB_ELEMENT containing as many frames as FNs_Middle_TMSRTE indicates.

MOB_TYPE2 … An MOB which does not have MOB_ELEMENT containing as many frames as FNs_Middle_TMSRTE indicates

(An MOB composed only by MOB_ELEMENT which contains the number of frames under FNs_Middle_TMSRTE beyond 1.)

up to two MOB_TYPE2s are allowed to be combined. In addition, if MOBs are combined before or behind such MOB_TYPE2s, those MOBs shall always be MOB_TYPE1s.

◊ **Figure 3.8.1-1 : Combine MOBs**



The minimum composition of one Track shall be as follows:

1 MOB = 1 MOB_FRAME

If, as a result of combination, there are consecutive MOB_TYPE2s which contain frames less than the number of frames FNs_Middle_TMSRTE indicates (MOB_TYPE2s containing the number of frames under FNs_Middle_TMSRTE beyond 1), the maximum number of such MOBs is two.

© 2000 SD Association

CONFIDENTIAL

3.8.2 Examples of Combine

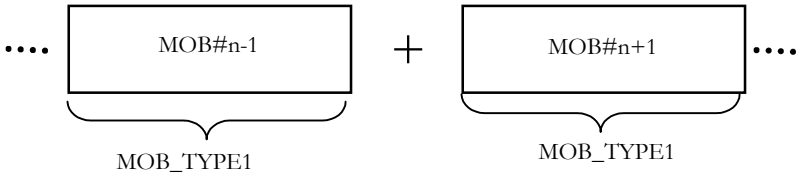
In the following cases, combining MOBs shall be allowed.

(case1) The end of the first half is MOB_TYPE1.

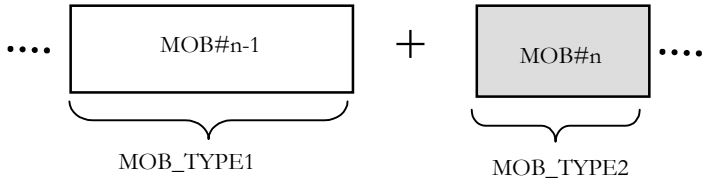
(1-1) The head of the latter half is MOB_TYPE1, as shown in [Figure 3.8.2-1].

(1-2) The head of the latter half is MOB_TYPE2, as shown in [Figure 3.8.2-2].

◊ **Figure 3.8.2-1 : Combines MOB_TYPE1 and MOB_TYPE1**



◊ **Figure 3.8.2-2 : Combines MOB_TYPE1 and MOB_TYPE2**

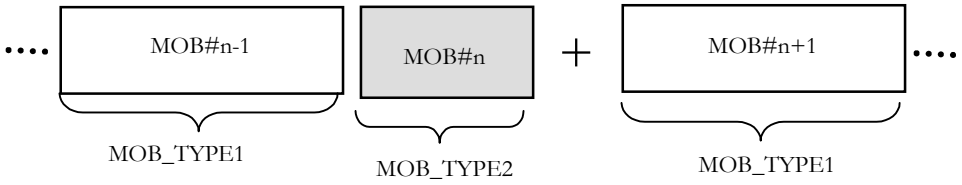


(case2) The end of the first half is MOB_TYPE2, and MOB just before that is MOB_TYPE1.

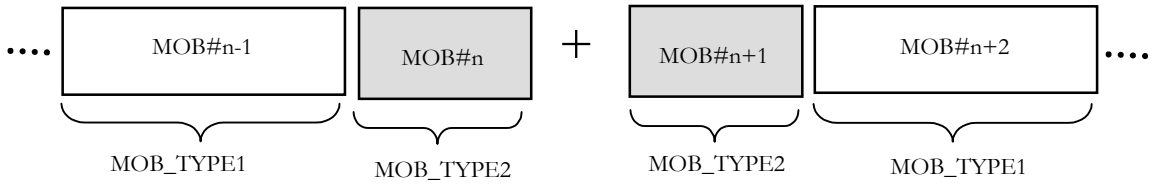
(2-1) The head of the latter half is MOB_TYPE1, as shown in [Figure 3.8.2-3].

(2-2) The head of the latter half is MOB_TYPE2, and then the next MOB is MOB_TYPE1, as shown in [Figure 3.8.2-4].

◊ **Figure 3.8.2-3: Combines MOB_TYPE1, MOB_TYPE2 and MOB_TYPE1**



◊ **Figure 3.8.2-4 : Combines MOB_TYPE1, MOB_TYPE2, MOB_TYPE2 and MOB_TYPE1**

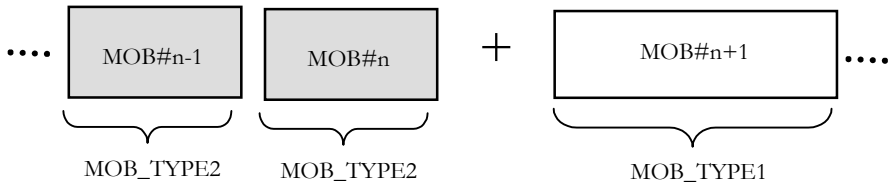


© 2000 SD Association

CONFIDENTIAL

(case3) The end of the first half consists of MOB_TYPE2+MOB_TYPE2.
The head of the latter half is MOB_TYPE1, as shown in [Figure 3.8.2-5].

◊ **Figure 3.8.2-5 : Combine MOB_TYPE2, MOB_TYPE2 and MOB_TYPE1**

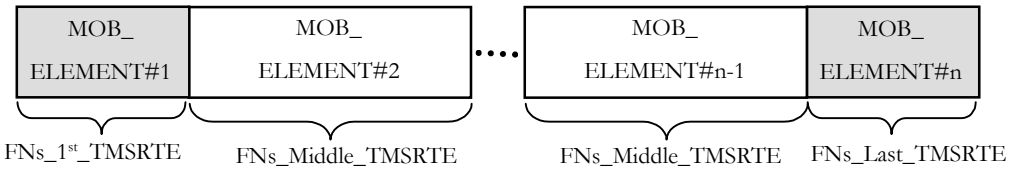


3.8.3 Category of MOBs

MOBs are categorized to MOB_TYPE1 and MOB_TYPE2.

3.8.3.1 MOB_TYPE1

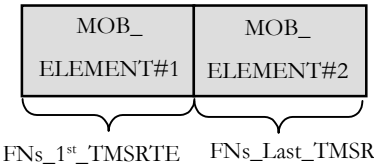
◊ **Figure 3.8.3.1-1 : Structure of MOB_TYPE1**



- When MOB consists of one MOB_ELEMENT,
  FNs_1st_TMSRTE = FNs_Middle_TMSRTE.
- When MOB consists of two MOB_ELEMENTs,
  FNs_1st_TMSRTE = FNs_Middle_TMSRTE
  or FNs_Last_TMSRTE = FNs_Middle_TMSRTE
- When MOB consists of three MOB_ELEMENTs or more,
  the MOB is always MOB_TYPE1.

3.8.3.2 MOB_TYPE2

◊ **Figure 3.8.3.2-1 : Structure of MOB_TYPE2**



- When MOB consists of one MOB_ELEMENT,

© 2000 SD Association

CONFIDENTIAL

$1 \leq FNs_1^{st}_TMSRTE < FNs_Middle_TMSRTE$

- When MOB consists of two MOB_ELEMENTs,

  $1 \leq FNs_1^{st}_TMSRTE < FNs_Middle_TMSRTE$

  and $1 \leq FNs_Last_TMSRTE < FNs_Middle_TMSRTE$

As for details of MOB, MOB_FRAME, and MOB_ELEMENT, refer to **3.6 Presentation Data Structure**.

© 2000 SD Association

CONFIDENTIAL

4. Navigation Data Structure

Navigation Data is the information on attributes and playback control for the Presentation Data. Navigation Data consists of three logical structures, Playlist Manager (PLMG), Track Manager (TKMG), and POB Manager (POBMG). PLMG contains Default Playlist Information (DPLI) and Playlist Information (PLI). All information including text and graphic data about Playlist is described in PLMG. TKMG contains Track Information (TKI), in which reference information and Management information for all Tracks are stored. POBMG contains POB Count Information (POBCI) which indicates by how many DPLIs, PLIs, and TKIs the POB file is referenced.

◊ **Figure 4-1 : Navigation Data Structure**

Playlist Manager (PLMG)

Playlist Manager Information (PLMGI) (Mandatory)	Fixed Size(2560B)
Default Playlist Information (DPLI) (Mandatory)	
Playlist Information #1 (PLI #1) (Optional)	Fixed Size(512B)
:	:
Playlist Information #n (PLI #n) (Optional)	Fixed Size(512B)

(n ≤ 99)

Track Manager (TKMG)

Track Information #1 (TKI #1) (Optional)	Fixed Size(1024B)
:	:
Track Information #m (TKI #m) (Optional)	Fixed Size(1024B)

(m ≤ 999)

© 2000 SD Association

CONFIDENTIAL

POB Manager (POBMG)

POB Manager Information (POBMGI) (Optional)	Fixed Size(2048B)
POB Count Information #1 (POBCI #1) (Optional)	
:	
POB Count Information #n (POBCI #n) (Optional)	
reserved (42 bytes)	

(n = 999)

© 2000 SD Association

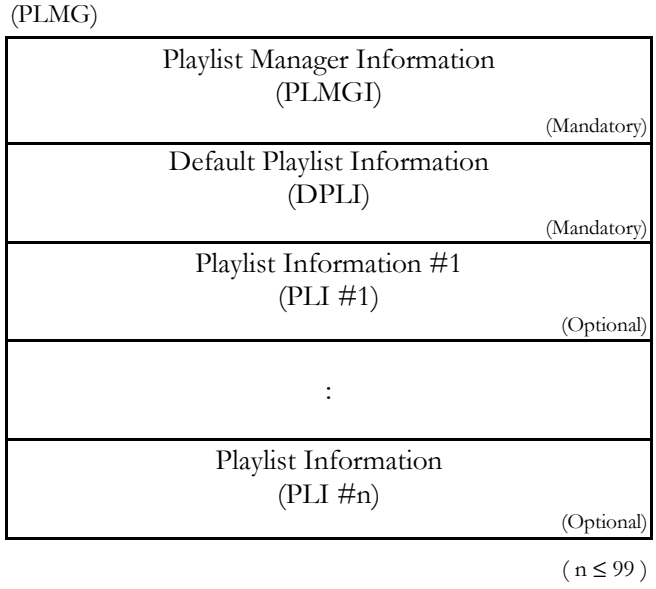
CONFIDENTIAL

4.1 Playlist Manager (PLMG)

PLMG describes information about Default Playlist and Playlists within the "SD_VCyyy" directory (001 ≤ yyy ≤ 999).

The PLMG starts with Playlist Manager Information (PLMGI), followed by Default Playlist Information (DPLI), and set of Playlist Information (PLI), as shown in [Figure 4.1-1].

◊ Figure 4.1-1 : Playlist Manager (PLMG)



© 2000 SD Association

CONFIDENTIAL

4.1.1 Playlist Manager Information(PLMGI)

As shown in [Table 4.1.1-1], PLMG Identifier, SD-Voice Identifier, Version number, number of Playlists, Track number played back first, Track number played back at last time, application attribute, and number of Tracks are described in Playlist Manager Information (PLMGI).

◊**Table 4.1.1-1 : PLMGI** (Description order)

RBP		Contents	Number of bytes
0 to 1	PLMG_ID	PLMG Identifier	2 bytes
2 to 3	reserved	reserved	2 bytes
4 to 11	SDV_ID	SD-Voice Identifier	8 bytes
12 to 13	VERN	Version number of **Part 6: VOICE SPECIFICATION**	2 bytes
14 to 15	PLMG_PL_Ns	Number of Playlists	2 bytes
16 to 19	PLMG_AP_PL (optional)	Track number played back first	4 bytes
20 to 27	PLMG_RSM_PL (optional)	Track number played back at last time	8 bytes
28 to 29	PLMG_APP_ATR	PLMG application attributes	2 bytes
30 to 31	reserved	reserved	2 bytes
32 to 33	TK_Ns	Number of Tracks	2 bytes
34 to 35	reserved	reserved	2 bytes
Total			36 bytes

(RBP 0 to 1) PLMG_ID

Describes "M1" to identify PLMGI with character set code of ISO646 (a-characters).

(RBP 4 to 11) SDV_ID

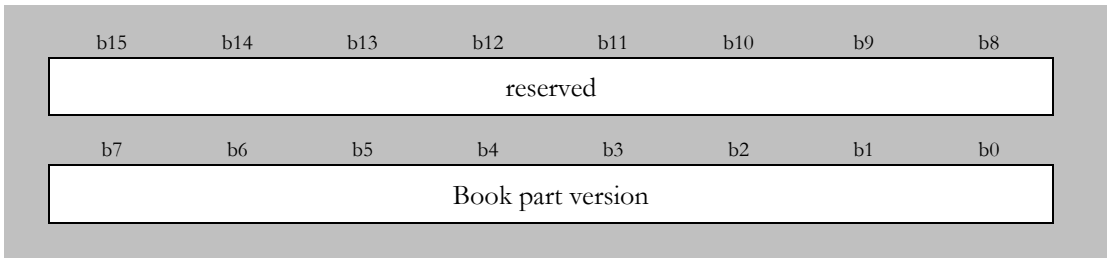
Describes "SD-VOICE" to identify SD-VOICE SPECIFICATION with character set code of ISO646 (a-characters).

© 2000 SD Association

CONFIDENTIAL

(RBP 12 to 13) VERN

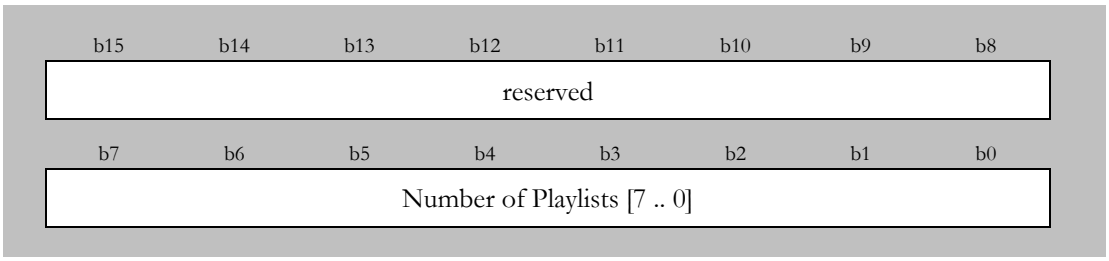
Describes the version number of the Part6. Voice specifications (SD Memory Card Specifications for VOICE).



Book part version ... 09h : version0.9 (for test use only)
10h : version1.0
Others : reserved

(RBP 14 to 15) PLMG_PL_Ns

Describes number of PLIs operated in PLMG.



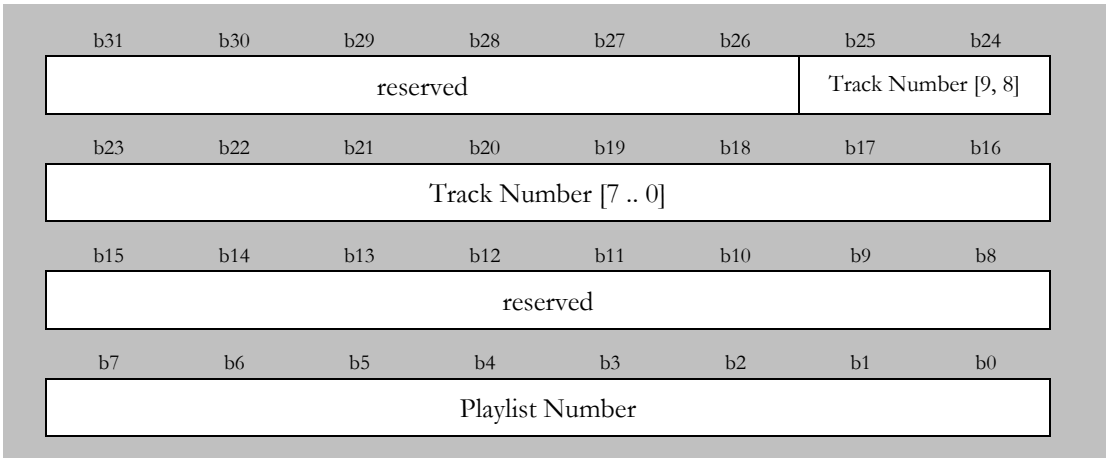
Number of Playlists ... Describes the number between '0' and '99'.

© 2000 SD Association

CONFIDENTIAL

(RBP 16 to 19) PLMG_AP_PL (optional)

Describes entry number of Playlist and entry number of DPL_TK_SRP or PL_TK_SRP which are played back at the initial access (such as after loading card, or in power up).
When PLMG_AP_PL is not specified, this field shall be set to '0'.



Playlist Number ... Describes the Playlist number between '0' and '99' which is played back at the initial access.
When Default Playlist is played back, enter '0'.

Track Number ... Desicribes the Track number which is played back at the initial access.
When Playlist Number is between '1' and '99', describes the number between '1' and '99'. It is equal to the entry number of PL_TK_SRP.
When Playlist Number is '0', describes the number between '1' and '999'. It is equal to the entry number of DPL_TK_SRP.

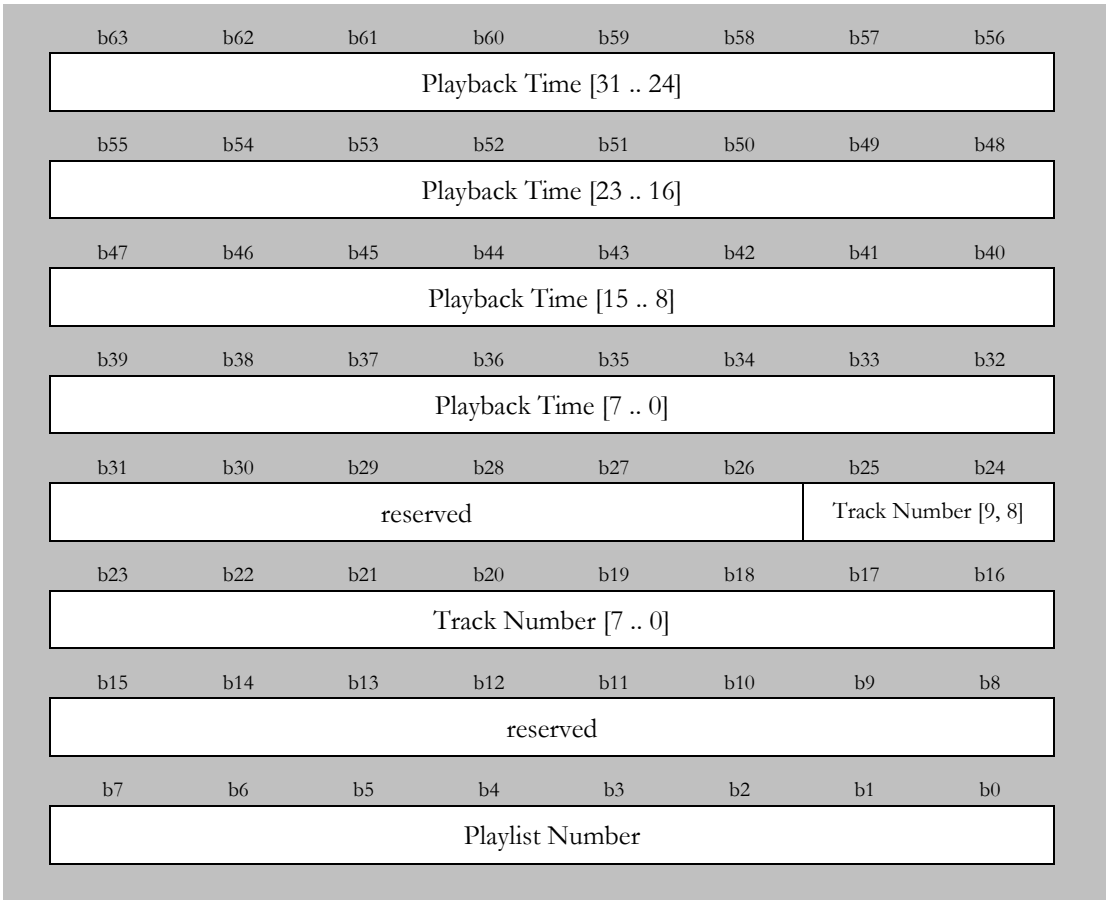
© 2000 SD Association

CONFIDENTIAL

(RBP 20 to 27) PLMG_RSM_PL (optional)

Describes the entry number of Playlist and entry number of DPL_TK_SRP or PL_TK_SRP that is played back last time.

When PLMG_RSM_PL is not specified, this field shall be set to '0'.



Playlist Number ... Describes the Playlist number between '0' and '99' which is played back last time.
When Default Playlist is played back, enter '0'.

Track Number ... Describes the Track number which is played back last time.
When Playlist Number is between '1' and '99', describes the number between '1' and '99'. It is equal to the entry number of PL_TK_SRP.
When Playlist Number is '0', describes the number between '1' and '999'. It is equal to the entry number of DPL_TK_SRP.

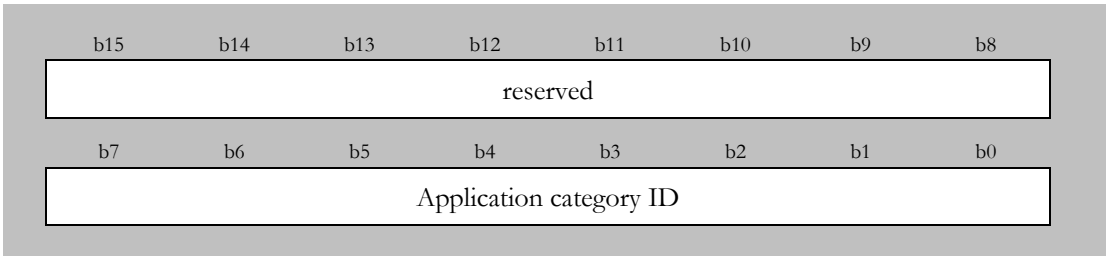
Playback Time ... Describes the Playback time of the Track which is played back at last time indicated in PLMG_RSM_PL in msec.

© 2000 SD Association

CONFIDENTIAL

(RBP 28 to 29) PLMG_APP_ATR

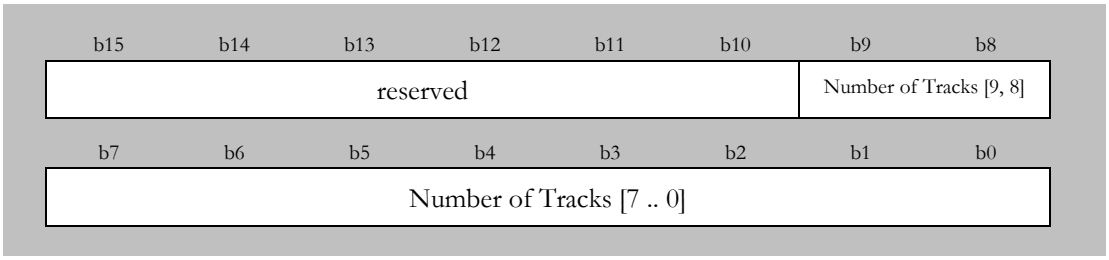
Application category ID area for this "SD_VCyyy" directory.



Application category ID ... 10h : Voice Recording
Others : reserved

(RBP 32 to 33) TK_Ns

Describes total number of Tracks.



Number of Tracks ... Describes the number between '0' and '999'.

© 2000 SD Association

CONFIDENTIAL

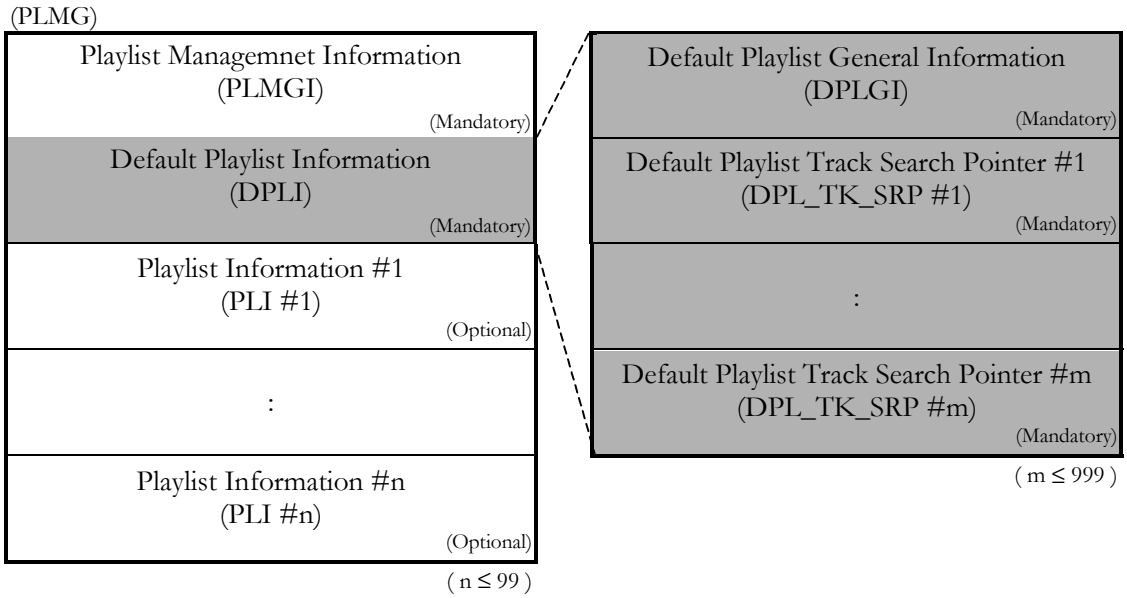
4.1.2 Default Playlist Information (DPLI)

DPLI can manage all Tracks stored in "SD_VCyyy" directory.

DPLI starts with Default Playlist General Information (DPLGI), followed by set of Default Playlist Track Search Pointer (DPL_TK_SRP), as shown in [Figure 4.1.2-1].

Only one DPLI exists in "SD_VCyyy" directory, and it can manage up to 999 Tracks (999 Tracks can exist in "SD_VCyyy" directory).

◊ Figure 4.1.2-1 : Default Playlist Information (DPLI)



© 2000 SD Association

CONFIDENTIAL

4.1.2.1 Default Playlist General Information (DPLGI)

As shown in [Table 4.1.2.1-1], DPLI Identifier, number of Tracks, total playback time of Tracks, application attribute, text information attribute, sub-directory name, text information, POB attribute, POB Search Pointers and Track number played back at last time are described in Default Playlist General Information (DPLGI).

◊**Table 4.1.2.1-1 : DPLGI** (Description order)

RBP		Contents	Number of bytes
0 to 1	DPLI_ID	DPLI Identifier	2 bytes
2 to 3	reserved	reserved	2 bytes
4 to 5	DPLI_TKI_Ns	Number of TKIs referred from Default Playlist	2 bytes
6 to 7	reserved	reserved	2 bytes
8 to 11	DPLI_PB_TM	Total playback time for Tracks referred from Default Playlist	4 bytes
12 to 13	DPLI_APP_ATR	DPLI application attributes	2 bytes
14 to 15	reserved	reserved	2 bytes
16 to 17	DPLI_PLTI1_ATR	Default Playlist Text Information Attribute 1	2 bytes
18 to 19	DPLI_PLTI2_ATR	Default Playlist Text Information Attribute 2	2 bytes
20 to 51	DPLI_SUBDIR_NM1	Sub-directory name for 1 byte characters	32 bytes
52 to 83	DPLI_SUBDIR_NM2	Sub-directory name for 2 byte characters	32 bytes
84 to 219	DPLI_PLTI	Default Playlist Text Information	136 bytes
220 to 221	DPLI_POB_ATR	Default Playlist POB Attribute	2 bytes
222 to 223	reserved	reserved	2 bytes
224 to 463	DPLI_POB_SRP	DPLI_POB Search Pointer (4B * 60)	240 bytes
464 to 469	DPLI_RSM_PL (optional)	Track number played back at last time	6 bytes
470 to 475	reserved	reserved	6 bytes
Total			476 bytes

(RBP 0 to 1) DPLI _ID

Describes "M2" to identify DPLI with character set code of ISO646 (a-characters).

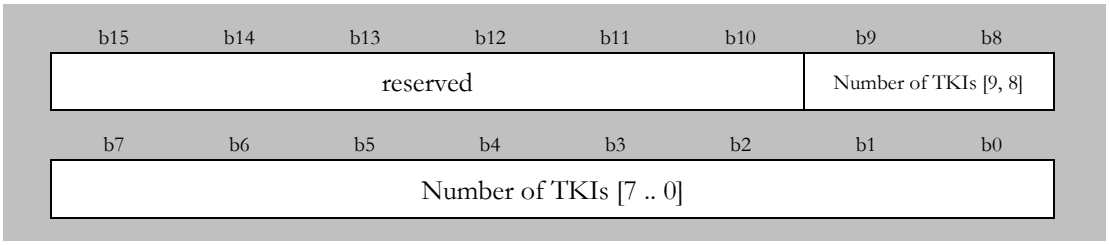
© 2000 SD Association

CONFIDENTIAL

(RBP 4 to 5) DPLI_TKI_Ns

Describes number of TKIs referenced from DPLI.

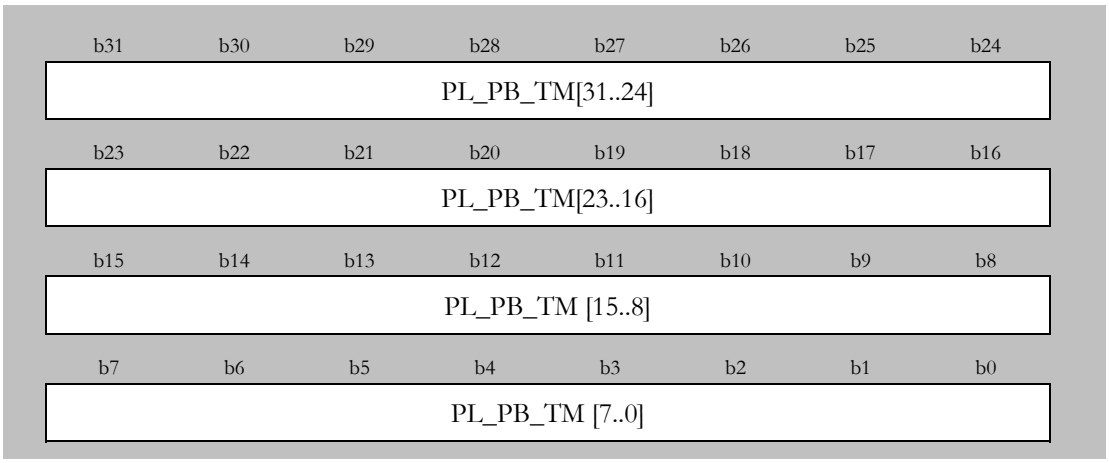
It may be more than actual Track numbers, since a Track may consist of multiple TKIs.



Number of TKIs ... Describes the number between '0' and '999'.

(RBP 8 to 11) DPLI_PB_TM

Describes total playback time for all Tracks referenced from Default Playlist in msec.

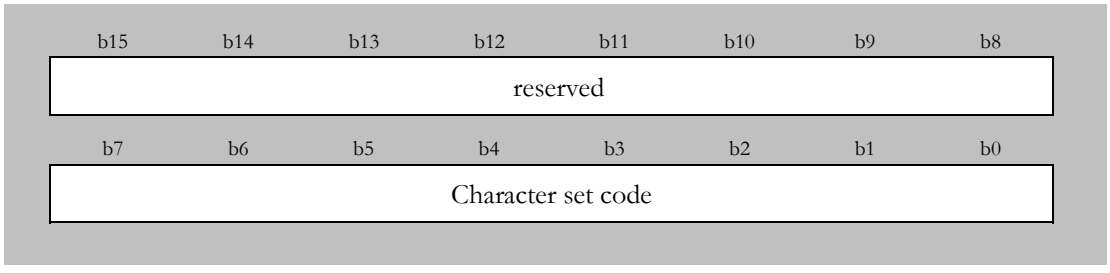


(RBP 12 to 13) DPLI_APP_ATR

Application attribute ID area for Default Playlist.

(RBP 16 to 17) DPLI_PLTI1_ATR

Describes the text attribute of Default Playlist.



Character set code ... 00h : Text data whose character set code is DPLI_PLTI1_ATR does not exist.

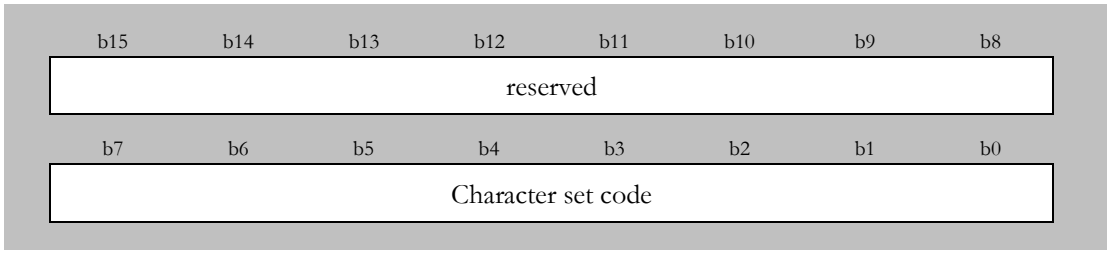
© 2000 SD Association

CONFIDENTIAL

01h : ISO646
02h : JISX0201
03h : ISO 8859-1
Others : reserved

(RBP 18 to 19) DPLI_PLTI2_ATR

Describes the text attribute of Default Playlist.



Character set code ... 00h : Text data whose character set code is DPLI_PLTI2_ATR does not exist.
82h : Shift JIS
Others : reserved

(RBP 20 to 51) DPLI_SUBDIR_NM1

Stores the 1 byte character text for "SD_VCyyy" directory nickname.
Character set code is specified in DPLI_PLTI1_ATR.

(RBP 52 to 83) DPLI_SUBDIR_NM2

Stores the text data for "SD_VCyyy" directory nickname.
Character set code is specified in DPLI_PLTI2_ATR.

© 2000 SD Association

CONFIDENTIAL

(RBP 84 to 219) DPLI_PLTI

Stores the text data.

When text data does not exist, this area shall be set to '0'.

When only one type of text data whose character set code is DPLI_PLTI1_ATR exists, describes text data followed by terminated code.

When only one type of text data whose character set code is DPLI_PLTI2_ATR exists, describes terminated code first, then text data followed by terminated code.

When two types text data exist, describes text data whose character set code is DPLI_PLTI1_ATR first, followed by terminated code, followed by text data whose character set code is DPLI_PLTI2_ATR, followed by terminated code.

<Terminated code>

ISO646 , JISX0201, ISO8859-1: '0x00'

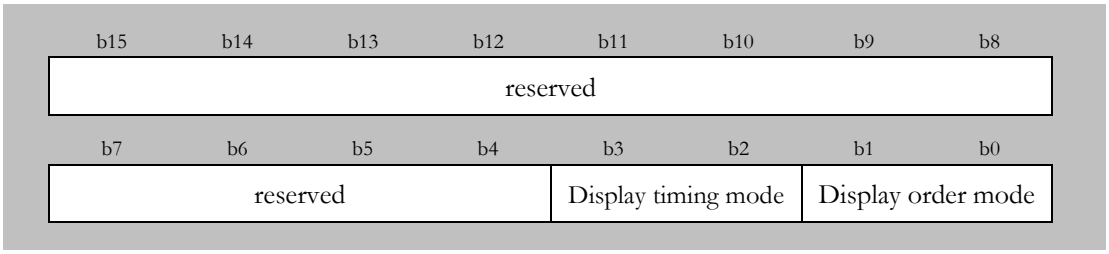
Shift JIS: '0x0000'

© 2000 SD Association

CONFIDENTIAL

(RBP 220 to 221) DPLI_POB_ATR

Describes the POB attribute of Default Playlist.



Display timing mode … 00b : Slideshow
01b : Browsable
Others : reserved

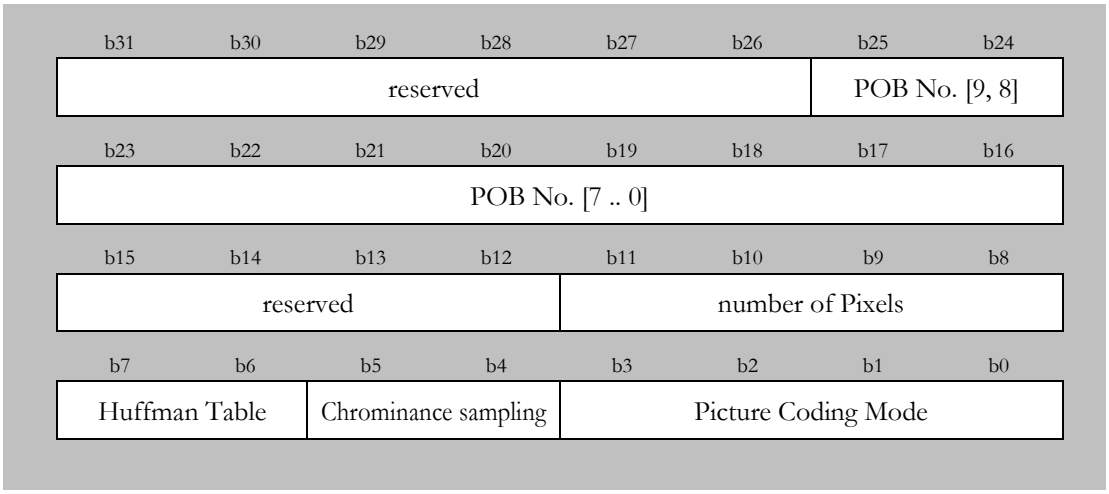
Display order mode … 00b : Sequential
01b : Random
10b : Shuffle
Others : reserved

© 2000 SD Association

CONFIDENTIAL

(RBP 224 to 463) DPLI_POB_SRP

Describes the POB Search Pointer of Default Playlist and each POB's attribute.
When specifying a POB, use the head DPLI_POB_SRP first. When not specifying a POB, enter '0'.



POB No. … Describes the number between '1' and '999'.
When no POB exists, enter '000h'.

number of Pixels … 0000b : 96x96 (recommended)
0001b : 640x480 (recommended)
0010b : 160x120 ~ 1800x1200 (optional)
Others : reserved

Huffman Table … 00b : Typical Table
Others : reserved

Chrominance sampling … 00b : 4:2:2
01b : 4:2:0
Others : reserved

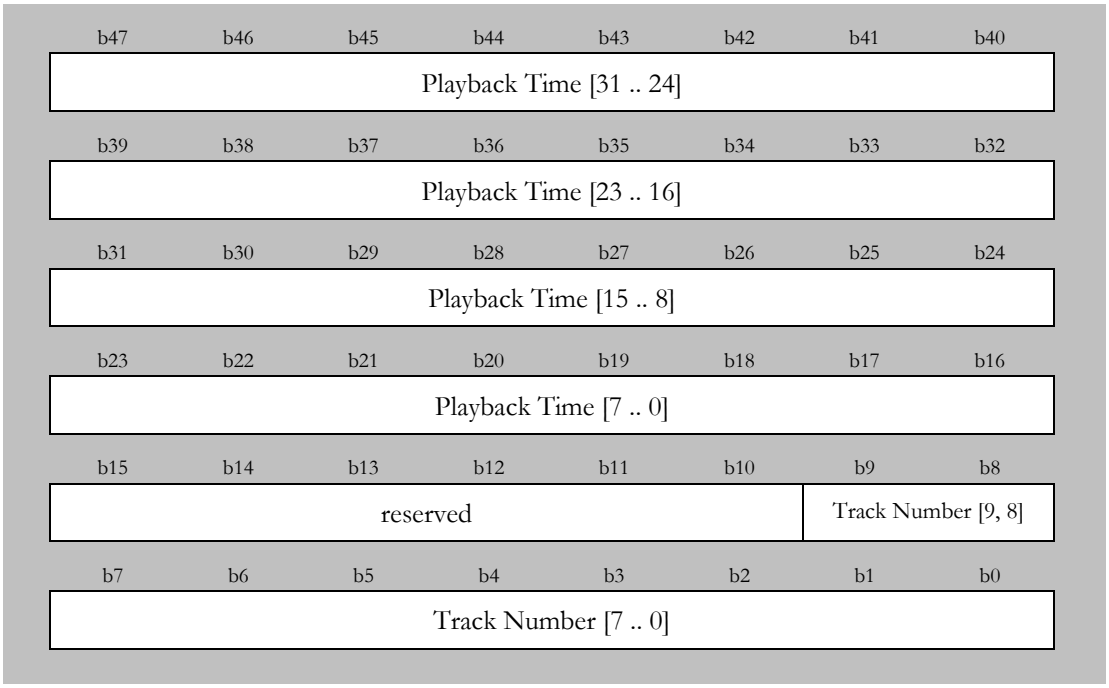
Picture Coding Mode … 0000b : JPEG
Others : reserved

© 2000 SD Association

CONFIDENTIAL

(RBP 464 to 469) DPLI_RSM_PL (optional)

Describes the entry number of DPL_TK_SRP and resume point that is played back last time.
When DPLI_RSM_PL is not specified, this field shall be set to '0'.



Track Number ... Describes the Track number between '1' and '999' which is played back at last time. It is equal to the entry number of DPL_TK_SRP.

Playback Time ... Describes the playback time of the Track which is played back at last time indicated in DPLI_RSM_PL in msec.

© 2000 SD Association

CONFIDENTIAL

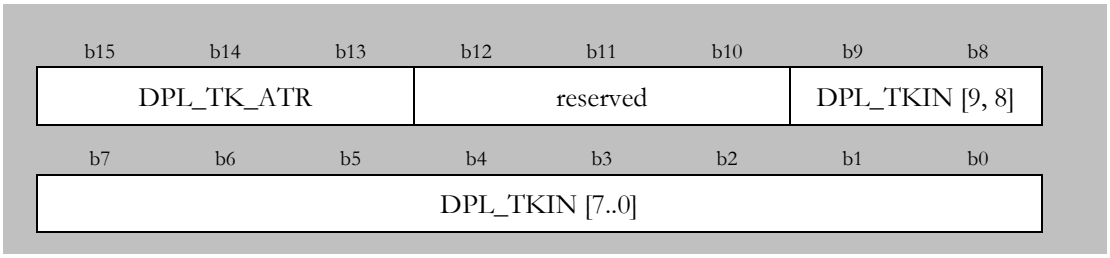
4.1.2.2 Default Playlist Track Search Pointer (DPL_TK_SRP)

Reference Information to Track is described in Default Playlist Track Search Pointer (DPL_TK_SRP). DPL_TK_SRPs, from the head to the number shown in DPL_TKI_Ns are valid, but 999 DPL_TK_SRPs are always registered regardless of the existence of corresponding Tracks.

◊Table 4.1.2.2-1 : DPL_TK_SRP

RBP		Contents	Number of bytes
0 to 1997	DPL_TK_SRP	Reference information to TKI	2 bytes * 999
1998 to 2047	reserved	Reserved	50 bytes
Total			2048 bytes

(RBP 0 to 1997) DPL_TK_SRP



(1) DPL_TK_ATR

Describes whether or not the TKI is used In a case where the TKI is used, describes whether or not one Track consists of one TKI, and TKI's position within the Track. In a case where the TKI is not used, it further describes whether or not the TKI has been used. In a case where the corresponding TKI has not been used yet (e.g. at the initial state), DPL_TK_SRP shall be set to '101b'.
Note : Same information is defined at TKI_BLK_ATR in TKGI. These shall be equal to each other.

DPL_TK_ATR ... 000b : One Track consists of one TKI.
001b : One Track consists of multiple TKIs. This TKI locates head position.
010b : One Track consists of multiple TKIs. This TKI locates middle position.
011b : One Track consists of multiple TKIs. This TKI locates tail position.
100b : TKI is not used. This TKI was deleted.
101b : TKI is not used. Corresponding TKI has not been allocated.
Others : reserved

(2) DPL_TKIN

Describes referred TKI number ranging '1 to 999'.
A number same as a DPL_TKIN number in other DPL_TK_SRPs shall not be described. The number shall be unique. Even if corresponding TKI has not been used yet, DPL TKIN shall be set to TKI number.

© 2000 SD Association

CONFIDENTIAL

4.1.3 Playlist Information (PLI)

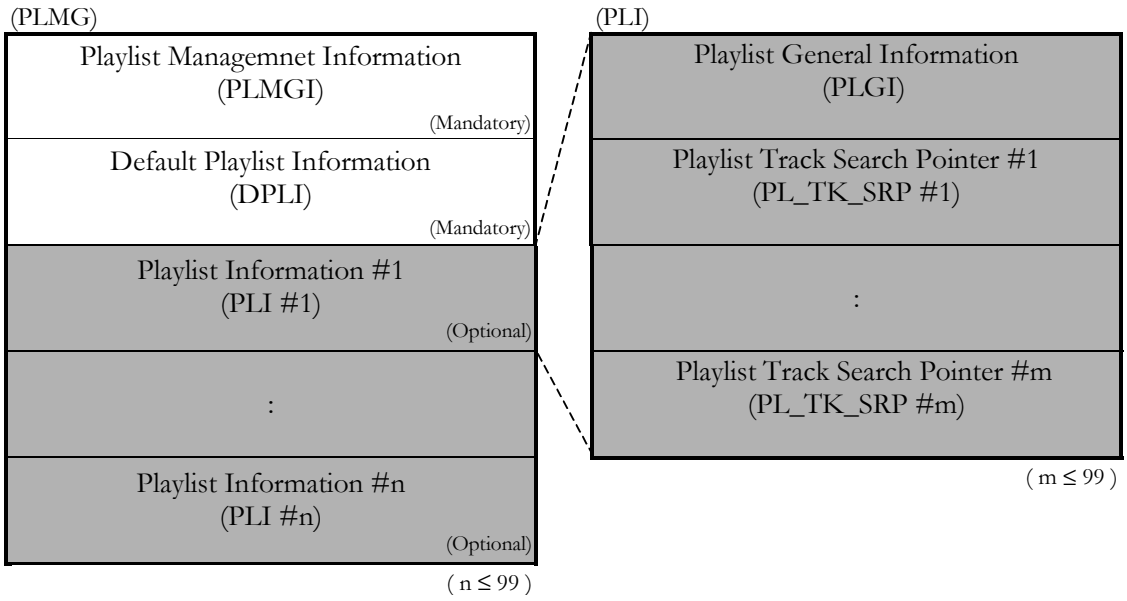
PLI can manage Tracks stored in "SD_VCyyy" directory.

PLI starts with Playlist General Information (PLGI), followed by set of Playlist Track Search Pointer (PL_TK_SRP). Playlist General Information (PLGI) manages additional information such as text, picture and so on. Track Search Pointer (PL_TK_SRP) is a reference information to Track. '0' to '99' Playlist exist in media, and each Playlist manages between '1' and '99' Tracks.

Playlist Information number (PLIN) is numbered for each Playlist from '1' to '999' in the ascending order.

PLIs, from the head to the number shown in PLMG_PL_Ns, are valid.

◊ **Figure 4.1.3-1 : Playlist Information (PLI)**



© 2000 SD Association

CONFIDENTIAL

4.1.3.1 Playlist General Information (PLGI)

As shown in [Table 4.1.3.1-1] , PLI Identifier, number of Tracks, total playback time of Tracks, application attribute, text information attribute, text information, POB attribute, POB Search pointers and Track number played back at last time are described in Playlist General Information (PLGI).

◊Table 4.1.3.1-1 : PLGI (Description order)

RBP		Contents	Number of bytes
0 to 1	PLI_ID	PLI Identifier	2 bytes
2 to 3	PLIN	PLI Number	2 bytes
4 to 5	PLI_TK_Ns	Number of Tracks referred from Playlist	2 bytes
6 to 7	reserved	reserved	2 bytes
8 to 11	PLI_PB_TM	Total playback time of Tracks referred from Playlist	4 bytes
12 to 13	PLI_APP_ATR	Playlist Application Attributes	2 bytes
14 to 15	reserved	reserved	2 bytes
16 to 17	PLI_PLTI1_ATR	Playlist Text Attribute 1	2 bytes
18 to 19	PLI_PLTI2_ATR	Playlist Text Attribute 2	2 bytes
20 to 219	PLI_PLTI	Playlist Text Information	200 bytes
220 to 221	PLI_POB_ATR	PLI_POB attribute	2 bytes
222 to 223	reserved	reserved	2 bytes
224 to 303	PLI_POB_SRP	PLI_POB Search Pointers (4B*20)	80 bytes
304 to 309	PLI_RSM_PL (optional)	Track number played back at last time	6 bytes
310 to 311	reserved	reserved	2 bytes
Total			312 bytes

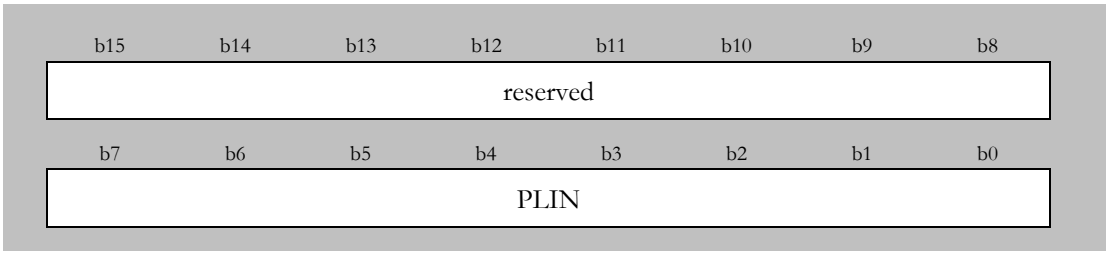
(RBP 0 to 1) PLI _ID

Describes "M3" to identify PLI with character set code of ISO646 (a-characters).

(RBP 2 to 3) PLIN

Describes PLI number ranging '1' to '99'.

This value shall be unique in PLGI, and shall be equal to a relative position in ascending order.



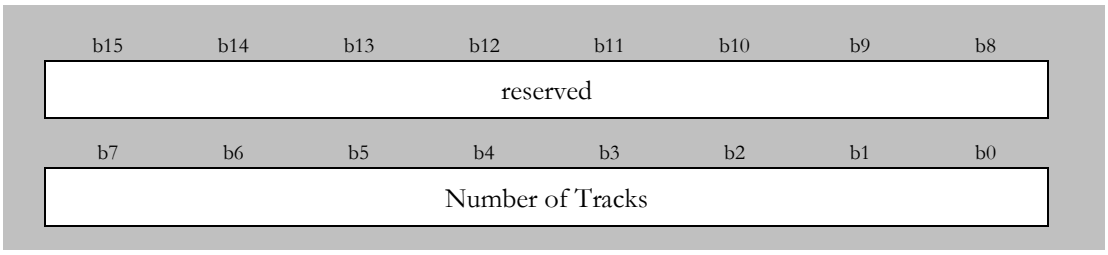
© 2000 SD Association

CONFIDENTIAL

PLIN ... Describes the number between '1' and '99'.

(RBP 4 to 5) PLI_TK_Ns

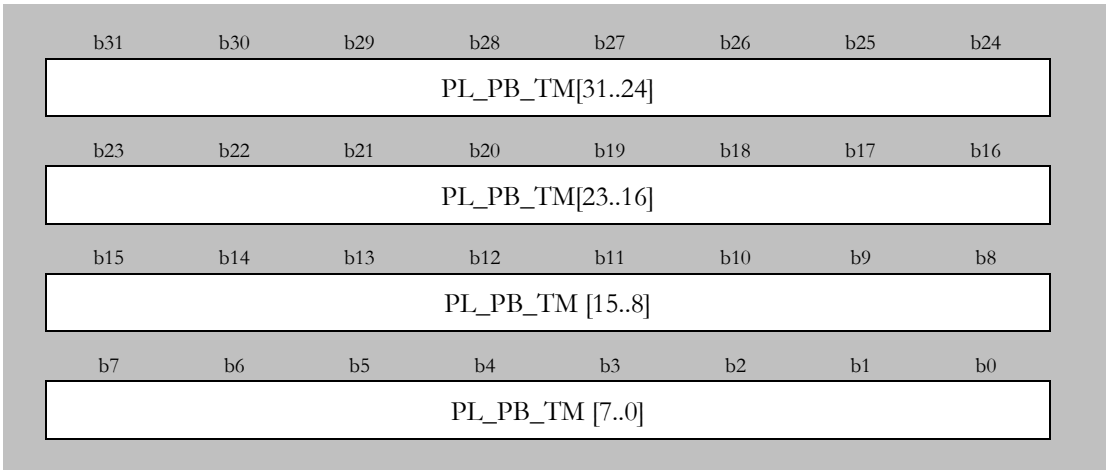
Describes number of Tracks referenced from PLI.



Number of Tracks ... Describes the number between '1' and '99'.

(RBP 8 to 11) PLI_PB_TM

Describes total playback time of all Tracks referenced from Playlist in msec.

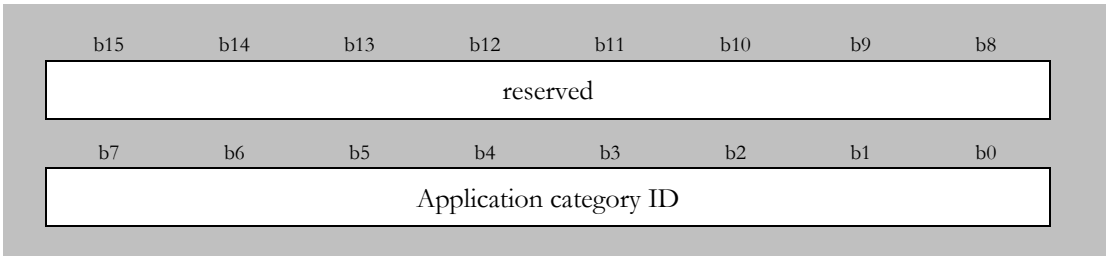


© 2000 SD Association

CONFIDENTIAL

(RBP 12 to 13) PLI_APP_ATR

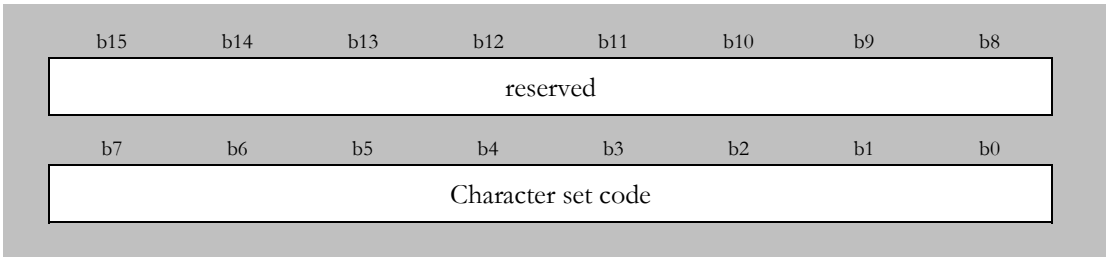
Application attribute ID area for Playlist.



Application category ID ... 10h : Voice Recording
Others : reserved

(RBP 16 to 17) PLI_PLTI1_ATR

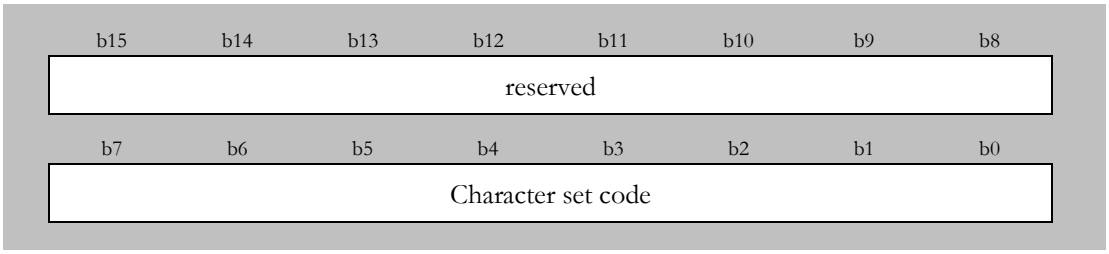
Describes the text attribute of Playlist.



Character set code … 00h : Text data whose character set code is PLI_PLTI1_ATR does not exist.
01h : ISO646
02h : JISX0201
03h : ISO 8859-1
Others : reserved

(RBP 18 to 19) PL_PLTI2_ATR

Describes the text attribute of Playlist.



Character set code … 00h : Text data whose character set code is PLI_PLTI2_ATR does not exist.
82h : Shift JIS

© 2000 SD Association

CONFIDENTIAL

Others : reserved

(RBP 20 to 219) PLI_PLTI

Stores the text data.

When text data does not exist, this area shall be set to '0'.

When only one type of text data whose character set code is PLI_PLTI1_ATR exists, describes text data followed by terminated code.

When only one type of text data whose character set code is PLI_PLTI2_ATR exists, describes terminated code, then text data followed by terminated code.

When two types text data exist, describes text data whose character set code is PLI_PLTI1_ATR first, followed by terminated code, followed by text data whose character set code is PLI_PLTI2_ATR, followed by terminated code.

<Terminated code>

ISO646 , JISX0201, ISO8859-1: '0x00'

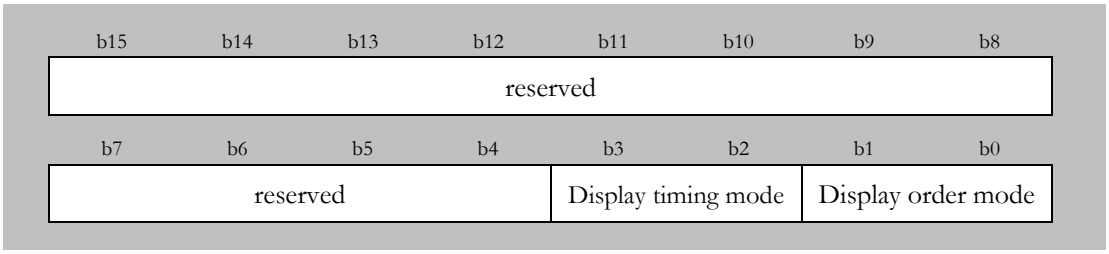
Shift JIS: '0x0000'

© 2000 SD Association

CONFIDENTIAL

(RBP 220 to 221) PLI_POB_ATR

Describes the POB attribute of Playlist.



Display timing mode … 00b : Slideshow
01b : Browsable
Others : reserved

Display order mode … 00b : Sequential
01b : Random
10b : Shuffle
Others : reserved

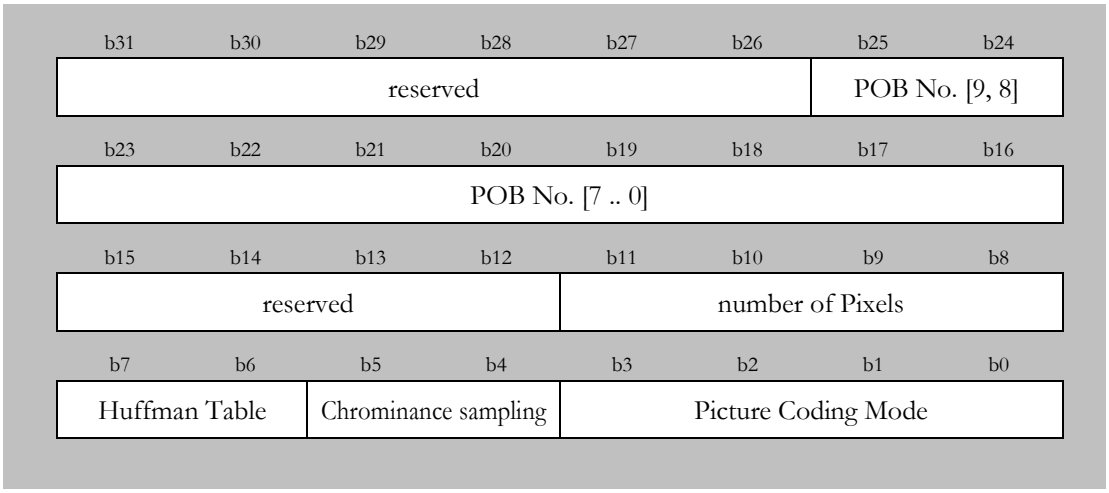
© 2000 SD Association

CONFIDENTIAL

(RBP 224 to 303) PLI_POB_SRP

Describes the POB Search Pointer of Playlist and each POB's attribute.

When specifying a POB, use the head PLI_POB_SRP first. When not specifying a POB, enter '0'.



POB No. … Describes the number between '1' and '999'.
When no POB exists, enter '000h'.

number of Pixels … 0000b : 96x96 (recommended)
0001b : 640x480 (recommended)
0010b : 160x120 ~ 1800x1200 (optional)
Others : reserved

Huffman Table … 00b : Typical Table
Others : reserved

Chrominance sampling … 00b : 4:2:2
01b : 4:2:0
Others : reserved

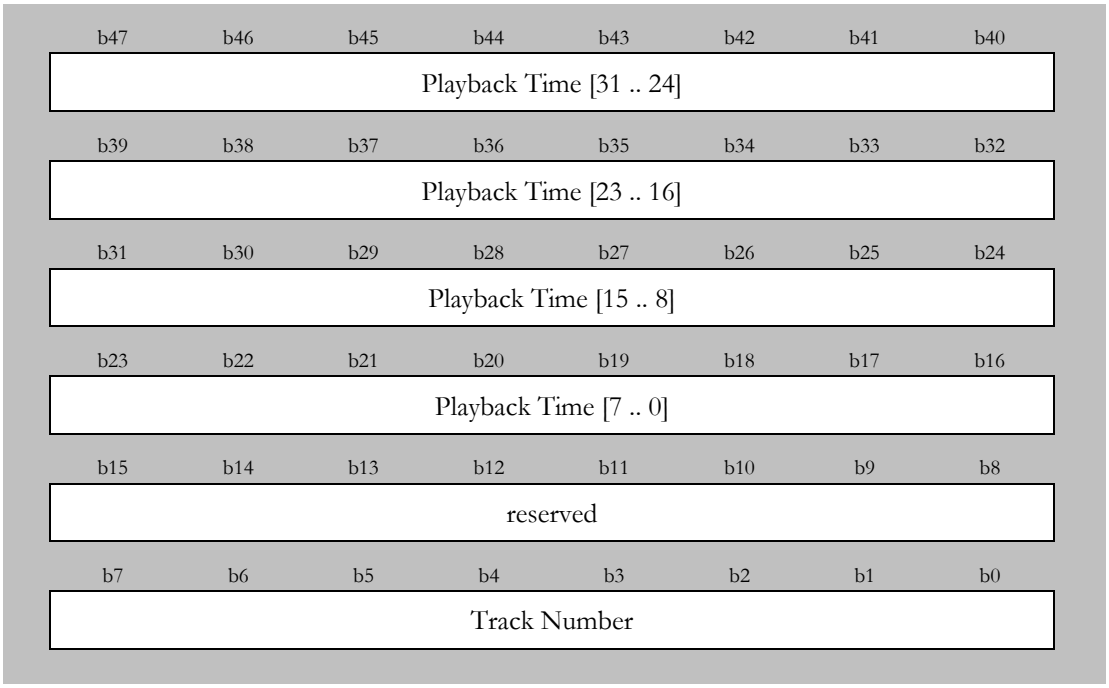
Picture Coding Mode … 0000b : JPEG
Others : reserved

© 2000 SD Association

CONFIDENTIAL

(RBP 304 to 309) PLI_RSM_PL (optional)

Describes the entry number of PL_TK_SRP and resume point that is played back last time.
When PLI_RSM_PL is not specified, this field shall be set to '0'.



Track Number ... Describes the Track number between '1' and '99' which is played back at last time. It is equal to the entry number of PL_TK_SRP.
When the Playlist is played through entirely, enter 'FFh'.

Playback Time ... Describes the playback time of the Track which is played back at last time indicated in PLI_RSM_PL in msec.

© 2000 SD Association

CONFIDENTIAL

4.1.3.2 Playlist TrackSearch Pointer (PL_TK_SRP)

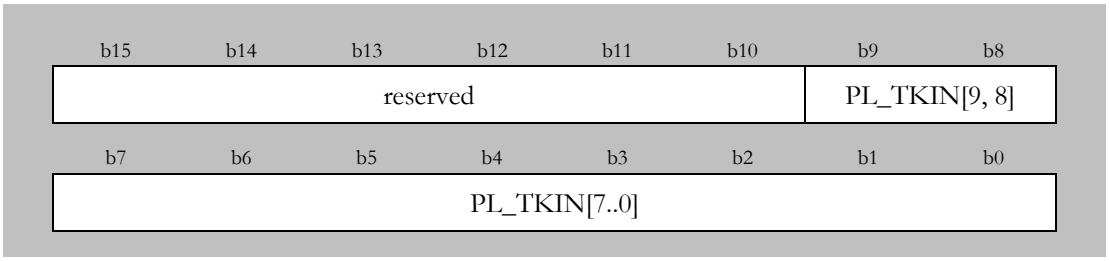
Reference Information to Track is described in Playlist Track Search Pointer (PL_TK_SRP). PL_TK_SRP, from the head to the number shown in PLI_TK_Ns, are valid, but 99 PL_TK_SRPs are always registered regardless of the existence of corresponding Tracks. When PL_TK_SRP is not used, this area shall be set to '0'.

◊ Table 4.1.3.2-1 : PL_TK_SRP

RBP		Contents	Number of bytes
0 to 197	PL_TK_SRP	Reference Information to Track	2 bytes * 99
198 to 199	reserved	reserved	2 bytes
Total			200 bytes

(RBP 0 to 197) PL_TK_SRP

Describes TKI number ranging '1' to '999' which is referred by Playlist.
When Track consists of multiple TKIs, a TKI number which corresponds to the head of Track shall be described.



PL_TKIN ... Describes referred TKI number between '1' and '999'.
A number same as a TKI number used in other PL_TK_SRPs can be described.

© 2000 SD Association

CONFIDENTIAL

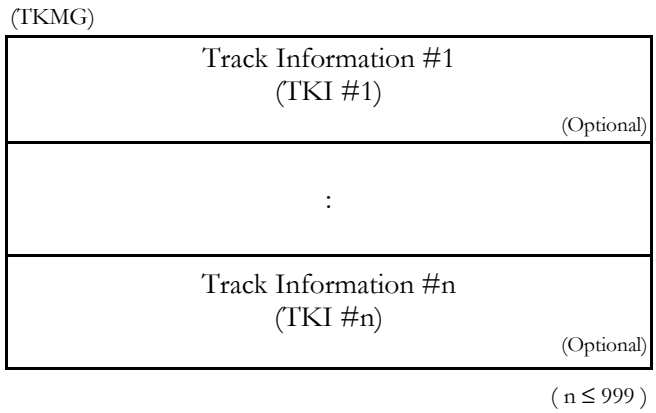
4.2 Track Manager (TKMG)

TKMG describes information about Tracks stored in "SD_VCyyy" directory.

TKMG contains set of Track Information (TKI), as shown in [Figure 4.2-1].

Track Information number (TKIN) is numbered for each TKI from 1 to 999 in the ascending order.

◊ **Figure 4.2-1 : Track Manager (TKMG)**



© 2000 SD Association

CONFIDENTIAL

4.2.1 Track Information (TKI)

TKI describes information for Track.

TKI starts with Track General Information (TKGI), followed by Track Text Information Data Area (TKTXTI_DA), followed by Time Search Table (TMSRT), as shown in [Figure 4.2.1-1].

TKI manages MOB Block, and MOB file.

TKI consists of fixed size (1024 bytes) data area.

TKI is used for as follows:

Case 1 :

One Track consists of one TKI. In this case, all information of the Track is described in one TKI.

Case 2:

One Track consists of multiple TKIs.

When one Track has long playing time, it may require more than 252 TMSRT entries (Refer to **4.2.1.3.2 TMSRT_entry**). In such case TKI Link Pointer (TKI_LNK_PTR) in the TKGI is set, then describes time search information in the TMSRT of next TKI.

Further more, TKI_LNK_PTR and time search information are different from those of previous TKI's. But other information is same as previous TKI's.

And MOB file is divided because TKI is separated.

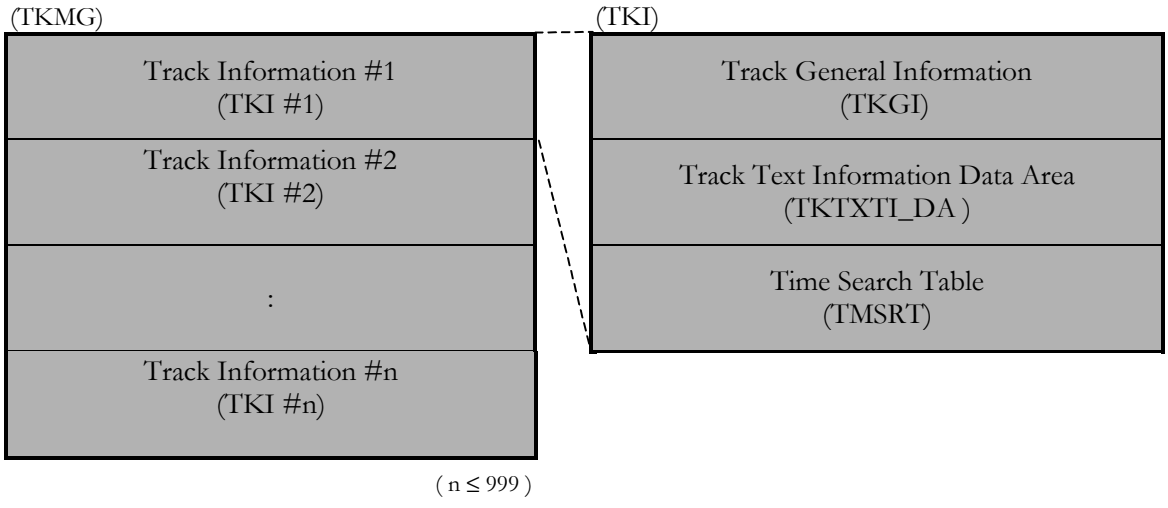
Case 3 :

One Track consists of multiple TKIs.

Even in the case that multiple Tracks are merged into one Track, TKIs themselves are not merged. Instead, TKI continuous flag (TKI_BLK_ATR) and TKI_LNK_PTR are changed appropriately.

In this case, multiple MOB files that are referenced from TKIs are played back seamlessly.

◊ **Figure 4.2.1-1 : Track Information (TKI)**



© 2000 SD Association

CONFIDENTIAL

4.2.1.1 Track General Information (TKGI)

As shown in [Table 4.2.1.1-1], TKI Identifier, TKI number, block attribute of TKI, link pointer to next TKI, size of TKI, playback time of Tracks, MOB attribute, company identifier, POB attribute, text information attribute, start address of TMSRT, TKI application attribute, Block Information Table, recording time, and POB Search Pointers are described in Track General Information (TKGI).

◊**Table 4.2.1.1-1 : TKGI** (Description order)

RBP		Contents	Number of bytes
0 to 1	TKI_ID	TKI Identifier	2 bytes
2 to 3	TKIN	TKI Number	2 bytes
4 to 5	TKI_BLK_ATR	Block Attribute of TKI	2 bytes
6 to 7	TKI_LNK_PTR	Link Pointer to next TKI	2 bytes
8 to 11	TKI_SZ	Size of TKI	4 bytes
12 to 15	TKI_PB_TM	Playback time of Tracks	4 bytes
16 to 19	TKI_MOB_ATR	Voice Message Attribute of TKI	4 bytes
20 to 23	TKI_CMPNY_ID	Company Identifier	4 bytes
24 to 25	TKI_POB_ATR	Picture Attribute of TKI	2 bytes
26 to 27	reserved	reserved	2 bytes
28 to 34	reserved	Reserved for copyright management Information	7 bytes
35	reserved	reserved	1 byte
36 to 37	TKI_TI1_ATR	Attribute of Text1	2 bytes
38 to 39	TKI_TI2_ATR	Attribute of Text2	2 bytes
40 to 43	TKI_TMSRT_SA	Start Address of TMSRT	4 bytes
44 to 53	reserved	reserved	10 bytes
54 to 55	TKI_APP_ATR	TKI application attributes	2 bytes
56 to 59	reserved	reserved	4 bytes
60 to 79	BIT	Block Information Table	20 bytes
80 to 87	TKI_RCD_TM	Recording Time	8 bytes
88 to 175	reserved	reserved	88 bytes
176 to 255	TKI_POB_SRP	TKI_POB Search Pointers (4B*20)	80 bytes
Total			256 bytes

(RBP 0 to 1) TKI _ID

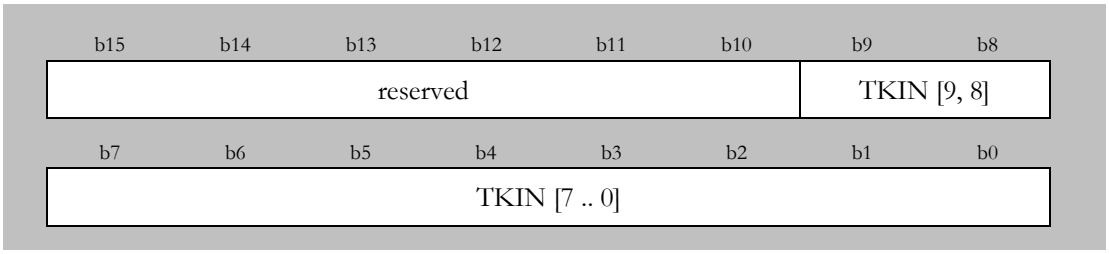
Describes "M4" to identify TKI with character set code of ISO646 (a-characters).

© 2000 SD Association

CONFIDENTIAL

(RBP 2 to 3) TKIN

Describes a TKI number ranging '1' to '999'.
This value is equal to the number in the TKI file name.



TKIN ... Describes the number between '1' and '999'.

(RBP 4 to 5) TKI_BLK_ATR

Describes whether or not the TKI is used. In the case where that TKI is used, describes whether or not one Track consists of one TKI, and a TKI's position within the Track. In the case where that TKI is not used, it further describes whether or not TKI has been used.
Note: Same information is defined at DPL_TK_ATR in DPL_TK_SRP. These shall be equal to each other.

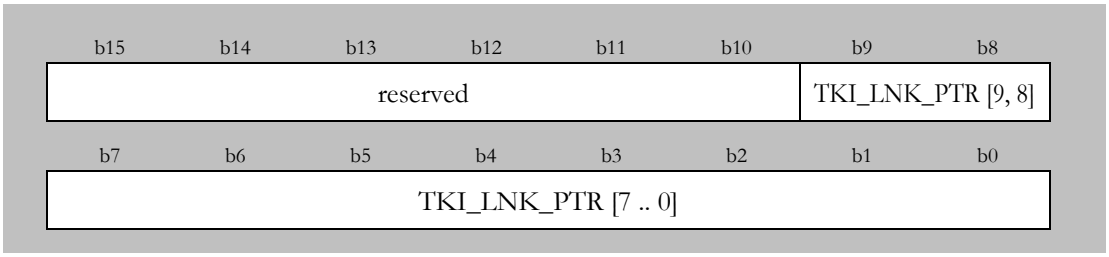
Block Attribute ... 000b : One Track consists of one TKI.
001b : One Track consists of multiple TKIs. This TKI locates head position.
010b : One Track consists of multiple TKIs. This TKI locates middle position.
011b : One Track consists of multiple TKIs. This TKI locates tail position.
100b : TKI is not used. This TKI was deleted.
101b : TKI is not used. Corresponding TKI has not been allocated.
Others : reserved

© 2000 SD Association

CONFIDENTIAL

(RBP 6 to 7) TKI_LNK_PTR

Describes a TKI number when this TKI continues to other TKI.



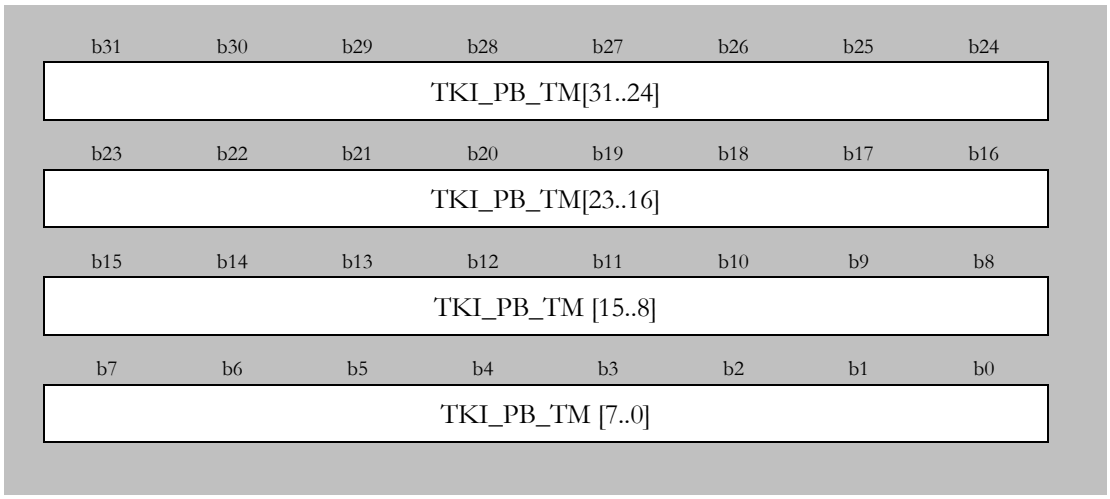
TKI_LNK_PTR ... Describes the number between '0' and '999'.
When the Block Attribute in TKI_BLK_ATR is '011b' and this TKI doesn't continue to other TKI, enter '000h'.

(RBP 8 to 11) TKI_SZ

Describes data size of the TKI in the number of bytes. In this specification, enter '1024'.

(RBP 12 to 15) TKI_PB_TM

Describes a total playback time for the Track referenced from this TKI in msec.
When one Track consists of multiple TKIs, total playback time of the Track is described only in the top TKI.
The playback time of this TKI shall be described in the second TKI or later.
'0' indicates that the duration of this TKI is not registered.



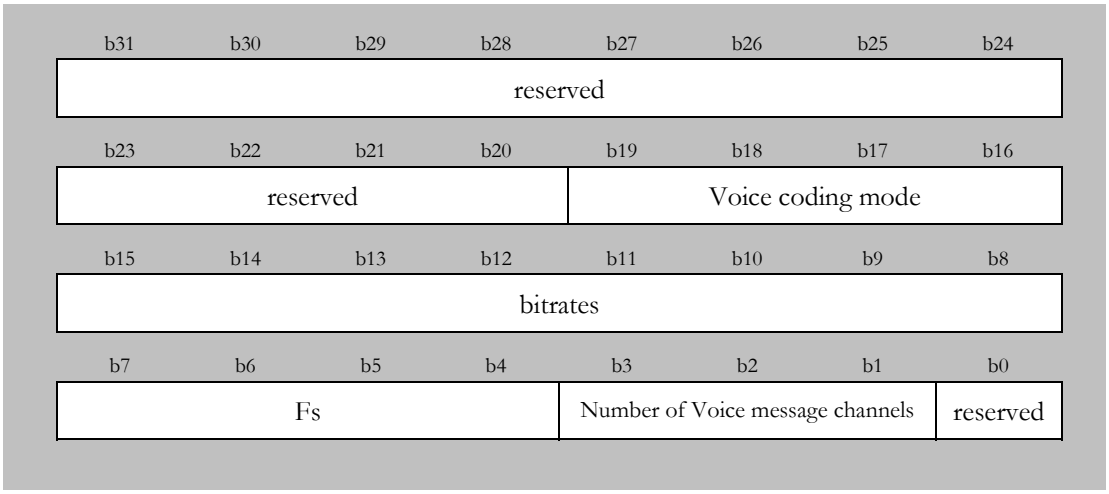
© 2000 SD Association

CONFIDENTIAL

(RBP 16 to 19) TKI_MOB_ATR

Describes the voice message attribute of corresponding to this TKI.

When a "Voice coding mode" (b16 to b19) is '1111b', no limitation will be set on using from b0 to b15 (Vendor Dependent).



Voice coding mode ... 1000b : ITU-T G.726
1001b : AMR
1111b: Vender Dependent
Others : reserved

bitrates (kbit/s/ch) ... Describes the number between '16' and '40' for ITU-T G.726
11111111b for AMR
11111111b : not specified (Variable bitrate).

Fs ... 0000b : 48 kHz
0001b : 44.1 kHz
0010b : 32 kHz
0011b : 24 kHz
0100b : 22.05 kHz
0101b : 16 kHz
0110b : 12 kHz
0111b : 11.025 kHz
1000b : 8 kHz
Others : reserved

Number of Voice Message Channels ... 000b : 1ch (mono)
001b : 2ch (stereo)
Others : reserved

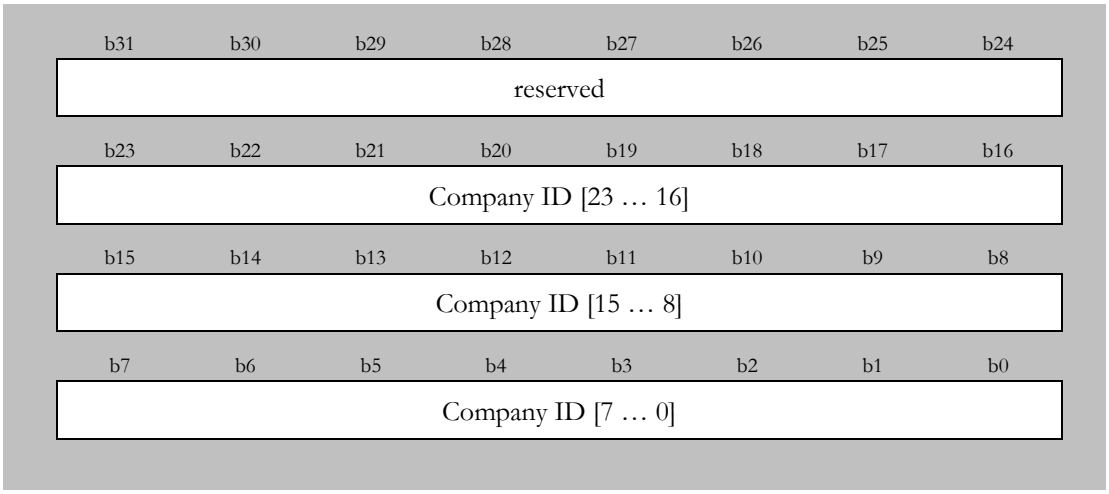
© 2000 SD Association

CONFIDENTIAL

(RBP 20 to 23) TKI_CMPNY_ID

When a "Voice coding mode" (b16 to b19) in a TKI_MOB_ATR is '1111b', describe a Company ID. When the "Voice coding mode" (b16 to b19) is anything other than '1111b', enter '0'.

As for details of the Company ID, refer to IEEE Registration Authority Committee (RAC).

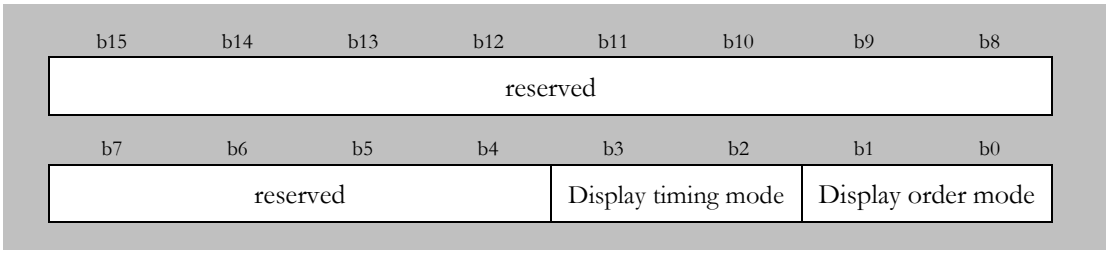


© 2000 SD Association

CONFIDENTIAL

(RBP 24 to 25) TKI_POB_ATR

Describes the POB attribute corresponding to this TKI.

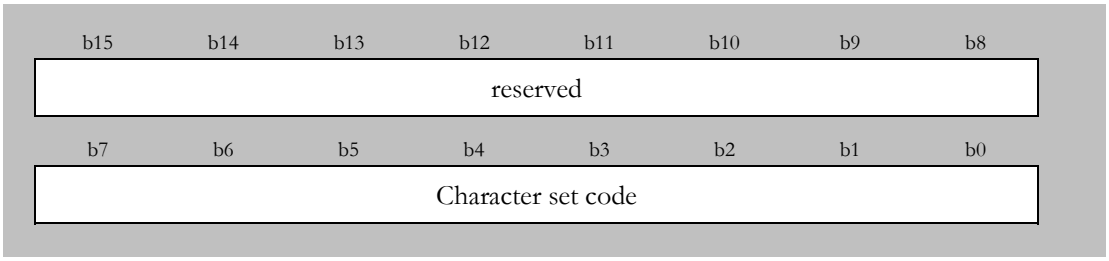


Display timing mode … 00b : Slideshow
01b : Browsable
Others : reserved

Display order mode … 00b : Sequential
01b : Random
10b : Shuffle
Others : reserved

(RBP 36 to 37) TKI_TI1_ATR

Describe the text attribute of the TKI.



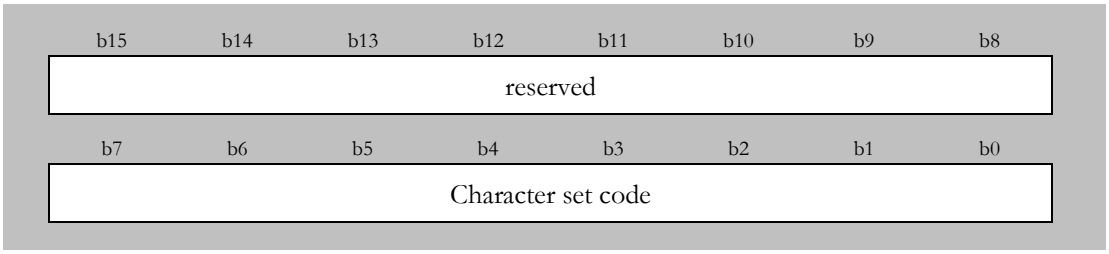
Character set code ... 00h : Text data whose character set code is TKI_TI1_ATR does not exist.
01h : ISO646
02h : JISX0201
03h : ISO 8859-1
Others : reserved

© 2000 SD Association

CONFIDENTIAL

(RBP 38 to 39) TKI_TI2 _ATR

Describes the text attribute of the TKI.



Character set code ... 00h : Text data whose character set code is TKI_TI2_ATR does not exist.
82h : Shift JIS
Others : reserved

(RBP 40 to 43) TKI_TMSRT_SA

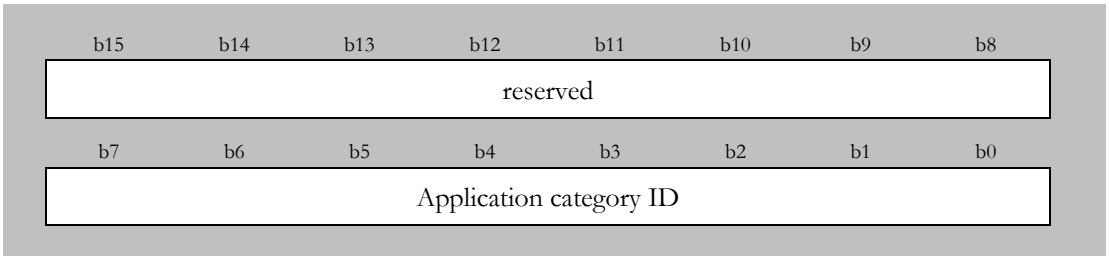
Describes start address TMSRT with relative logical byte number of this TKI. In this specification, enter '512'.

© 2000 SD Association

CONFIDENTIAL

(RBP 54 to 55) TKI_APP_ATR

Describes an application category ID for Track.



Application category ID ... 10h: Voice Recording
Others : reserved

© 2000 SD Association

CONFIDENTIAL

Block Information Table (BIT)

Block Information Table (BIT) is a table for the management of information about each MOB_BLOCK, and its structure is shown in [Figure 4.2.1.1-1].

Block Information Table (BIT) consists of the following entries.

(1) Data_Offset: Size of invalid area at the head of MOB file

(2) SZ_Data: Size of MOB_BLOCK

(3) TMSRTE_Ns: Number of TMSRT_entries in the MOB_BLOCK.

(4) FNs_1st_TMSRTE: Number of frames in the first TMSRT_entry.

(5) FNs_Last_TMSRTE: Number of frames in the last TMSRT_entry.

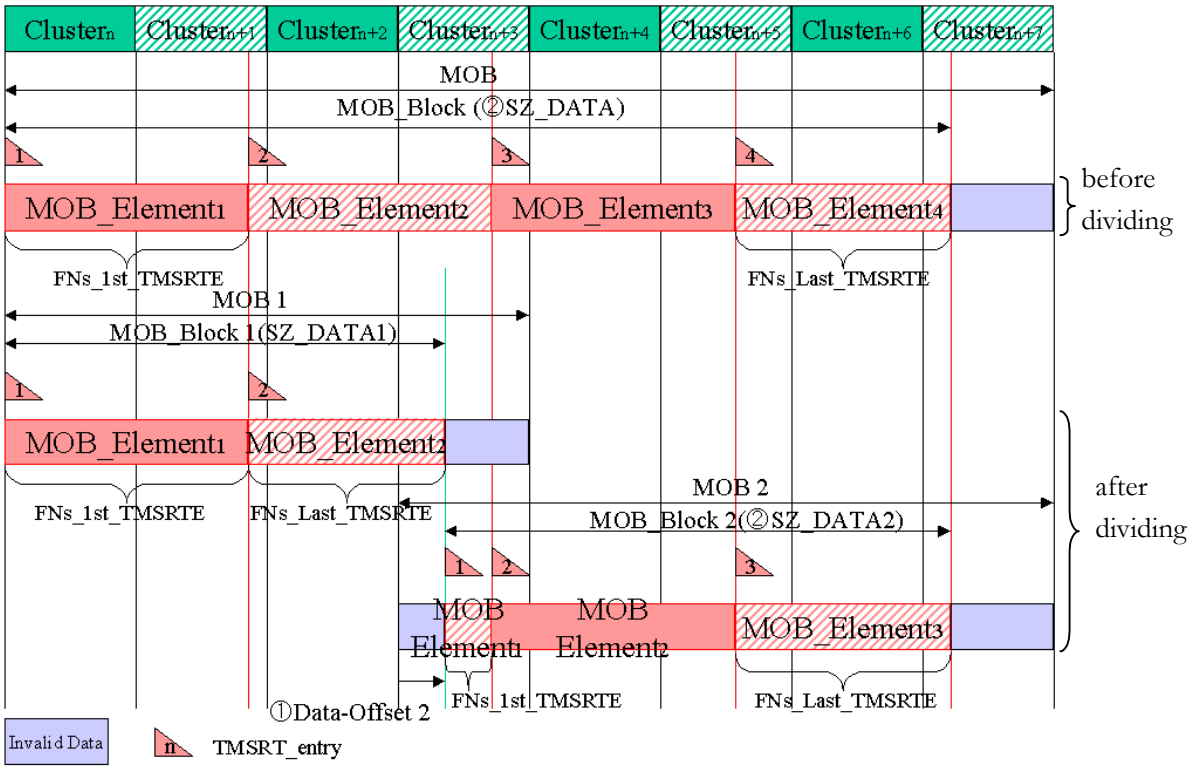
(6) FNs_Middle_TMSRTE: Number of frames in the middle TMSRT_entry.

(7) TIME_LENGTH: Presentation time duration of MOB_ELEMENT in msec.

As for details of "Cluster", refer to **Part2: FILE SYSTEM SPECIFICATION**.

As for the playback operation based on the BIT and TMSRT , refer to **3.7.1 Playback Operation** and **3.7.2 Special Playback Operation**.

◊ **Figure 4.2.1.1.-1 : relationship for entry of BIT**



The details structure of BIT is shown in [Table 4.2.1.1-2].

© 2000 SD Association

CONFIDENTIAL

◊ **Table 4.2.1.1-2: BIT**

(Description order)

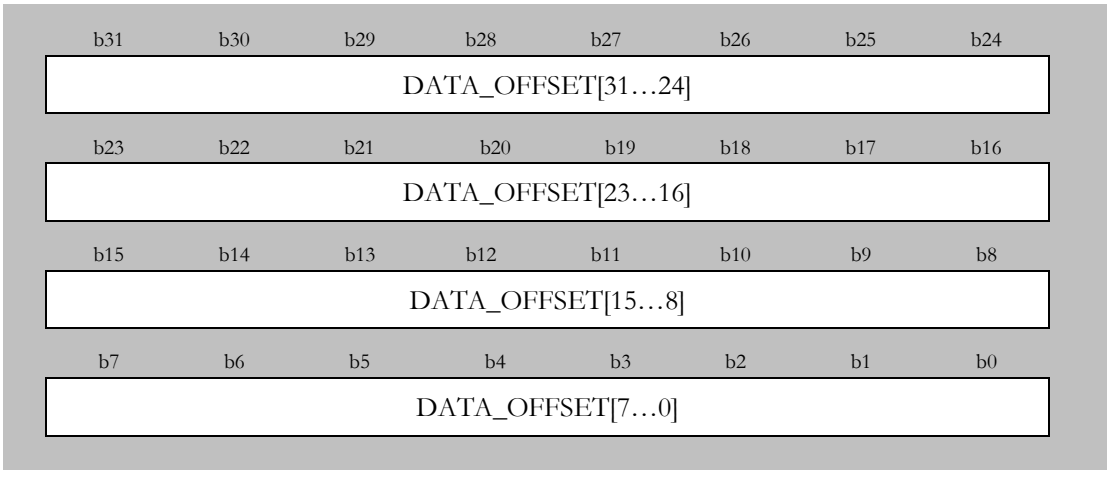
RBP	Field Name	Contents	Numbers of bytes
60 to 63	DATA_OFFSET	Size of invalid area at the head of MOB file	4 bytes
64 to 67	SZ_DATA	Data Length of MOB_BLOCK (Byte)	4 bytes
68 to 71	TMSRTE_Ns	Number of TMSRT_entry	4 bytes
72 to 73	FNs_1st_TMSRTE	Number of MOB_FRAMEs in the first MOB_ELEMENT	2 bytes
74 to 75	FNs_Last_TMSRTE	Number of MOB_FRAMEs in the last MOB_ELEMENT	2 bytes
76 to 77	FNs_Middle_TMSRTE	Number of MOB_FRAMEs in MOB_ELEMENT	2 bytes
78 to 79	TIME_LENGTH	Time length of MOB_ELEMENT	2 bytes
Total			20 bytes

(RBP 60 to 63) DATA_OFFSET

Describes the head address (in bytes) of each MOB_BLOCK.

DATA_OFFSET shall range from '0' to '(Size of Cluster) –1'.

Refer to **Part2: FILE SYSTEM SPECIFICATION** for details of the Cluster.

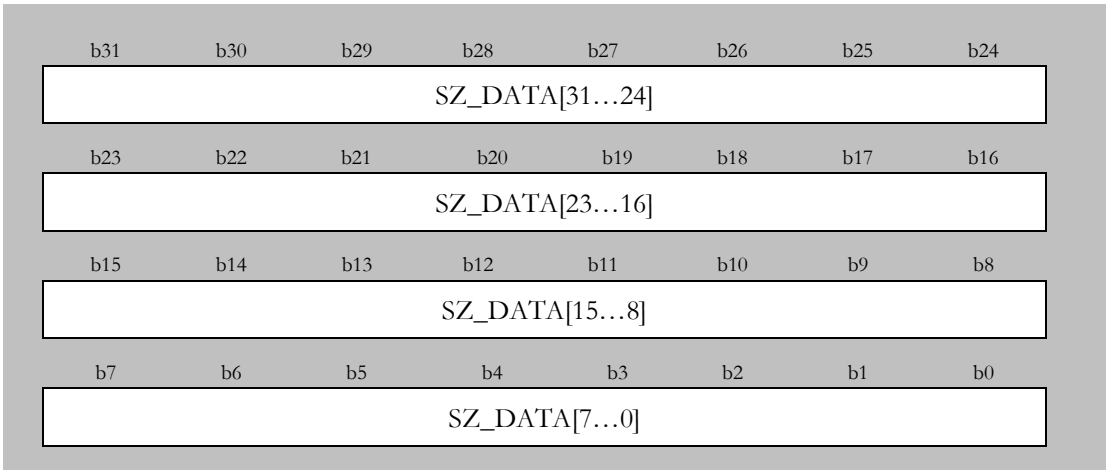


© 2000 SD Association

CONFIDENTIAL

(RBP 64 to 67) SZ_DATA

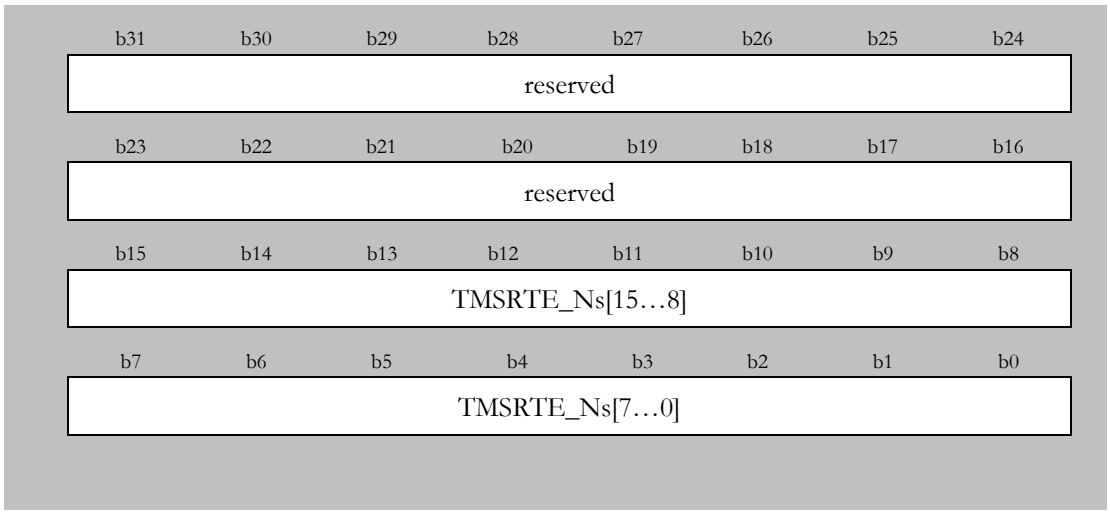
Describes the data length (in bytes) of each MOB_BLOCK.



(RBP 68 to 71) TMSRTE_Ns

Describes the total number of TMSRT_entry contained in each MOB_BLOCK.

When TKI is used, enter any number between '1' and '252'.



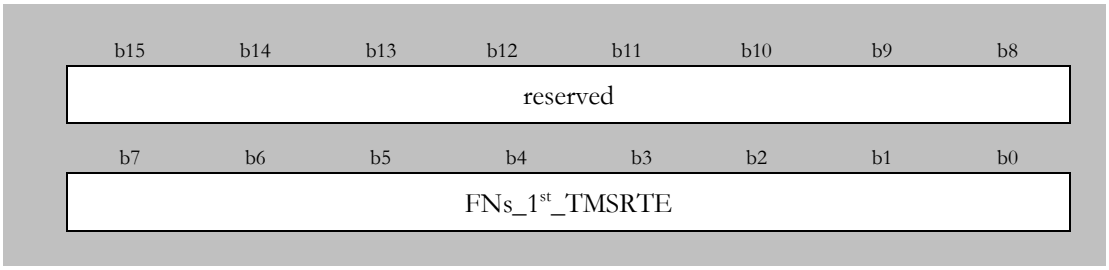
© 2000 SD Association

CONFIDENTIAL

(RBP 72 to 73) FNs_1st_TMSRTE

When one or more TMSRT_entry are contained in the MOB_BLOCK, FNs_1st_TMSRTE describes the number of MOB_FRAMEs between DATA_OFFSET and the first TMSRT_entry.

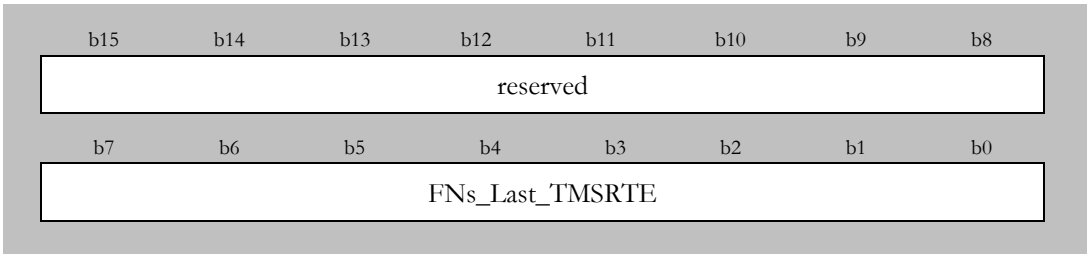
When there is no TMSRT_entry, FNs_1st_TMSRTE shall be set to '0'.



(RBP 74 to 75) FNs_Last_TMSRTE

Describes the number of MOB_FRAMEs that are contained in the last MOB_ELEMENT within MOB_BLOCK.

When only one TMSRT_entry is contained in the MOB_BLOCK, FNs_Last_TMSRTE shall be set to '0'.

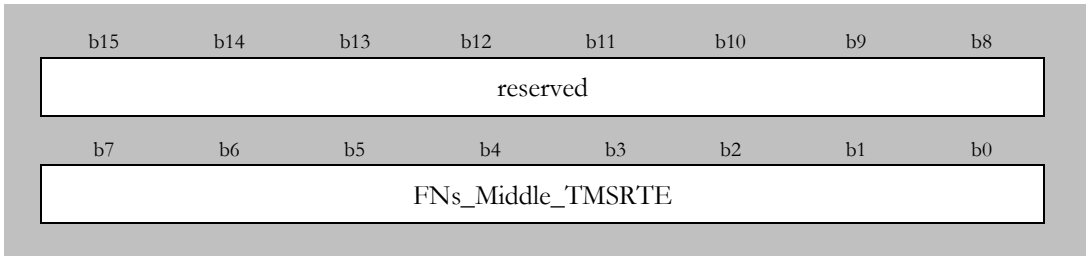


© 2000 SD Association

CONFIDENTIAL

(RBP 76 to 77) FNs_Middle_TMSRTE

Describes the number of MOB_FRAMEs that compose MOB_ELEMENT.
The value depends on a sampling frequency of MOB as shown in [Table 4.2.1.1-3].
Even if two or less TMSRT_entries are contained in the MOB_BLOCK, FNs_Middle_TMSRTE shall be set to the value as shown in [Table 4.2.1.1-3].



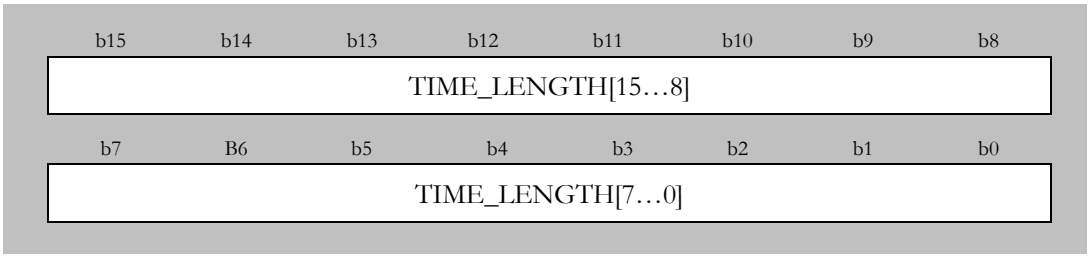
◊ **Table 4.2.1.1-3 : Relation between sampling frequency and FNs_Middle_TMSRTE**

Sampling Frequency	FNs_Middle_TMSRTE	
	G.726	AMR
16kHz	N. A.	N. A.
8kHz	10*N	10*N

N means TIME_LENGTH/1000, which describes presentation time duration of MOB_ELEMENT in second.
In case of coding mode of MOB is G.726 or AMR, value 2 (seconds) should be used as N.

(RBP 78 to 79) TIME_LENGTH

Describes presentation time duration of MOB_ELEMENT in msec.
When TIME_LENGTH equals to 1000, it corresponds to 1 second.
In case of coding mode of MOB is G.726, AMR, and the bitrate is within bitrates defined as mandatory, TIME_LENGTH shall be set to '2000'.



© 2000 SD Association

CONFIDENTIAL

(RBP 80 to 87) TKI_RCD_TM

Describes the Recording date of corresponding MOB's recording time, in year, month, date, hour, minute and second in the BCD format.

In the Recording time, it is recommended to describe a value at the time when an MOB is created.

b63	b62	b61	b60	b59	b58	b57	b56
reserved							
b55	b54	b53	b52	b51	b50	b49	b48
Year (thousand's)				Year (hundred's)			
b47	b46	b45	b44	b43	b42	b41	b40
Year (ten's)				Year (units)			
b39	b38	b37	b36	b35	b34	b33	b32
Month (ten's)				Month (units)			
b31	b30	b29	b28	b27	b26	b25	b24
Date (ten's)				Date (units)			
b23	b22	b21	b20	b19	b18	b17	b16
Hour (ten's)				Hour (units)			
b15	b14	b13	b12	b11	b10	b9	b8
Minute (ten's)				Minute (units)			
b7	b6	b5	b4	b3	b2	b1	b0
Second (ten's)				Second (units)			

Year (thousand's) ... Describes any number between '1' and '2'.

Year (hundred's) ... Describes any number between '0' and '9'.

Year (ten's) ... Describes any number between '0' and '9'.

Year (units) ... Describes any number between '0' and '9'.

Month (ten's) ... Describes any number between '0' and '1'.

Month (units) ... Describes any number between '0' and '9'.

Data (ten's) ... Describes any number between '0' and '3'.

Date (units) ... Describes any number between '0' and '9'.

© 2000 SD Association

CONFIDENTIAL

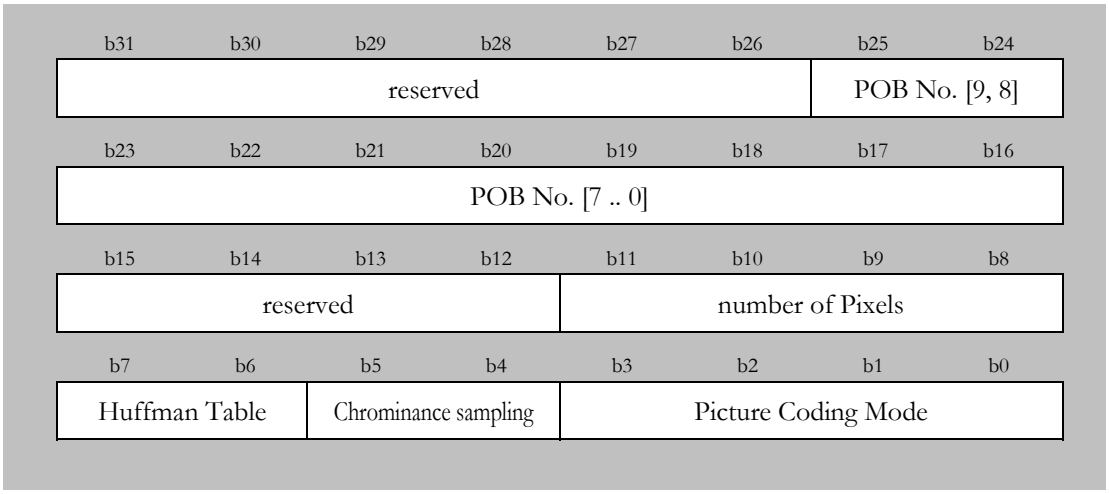
Hour (ten's)	...	Describes any number between '0' and '2'.
Hour (units)	...	Describes any number between '0' and '9'.
Minute (ten's)	...	Describes any number between '0' and '5'.
Minute (units)	...	Describes any number between '0' and '9'.
Second (ten's)	...	Describes any number between '0' and '5'.
Second (units)	...	Describes any number between '0' and '9'.

© 2000 SD Association

CONFIDENTIAL

(RBP 176 to 255) TKI_POB_SRP

Describes the POB No. which is referenced from Track and each POB's attribute.
When specifying a POB, use the head TKI_POB_SRP first. When not specifying a POB, enter '0'.
When a Track consists of multiple TKIs, only TKI_POB_SRP of the head TKI is effective.



POB No. … Describes the number between '1' and '999'.
When no POB exists, enter '000h'.

number of Pixels … 0000b : 96x96 (recommended)
0001b : 640x480 (recommended)
0010b : 160x120 ~ 1800x1200 (optional)
Others : reserved

Huffman Table … 00b : Typical Table
Others : reserved

Chrominance sampling … 00b : 4:2:2
01b : 4:2:0
Others : reserved

Picture Coding Mode … 0000b : JPEG
Others : reserved

© 2000 SD Association

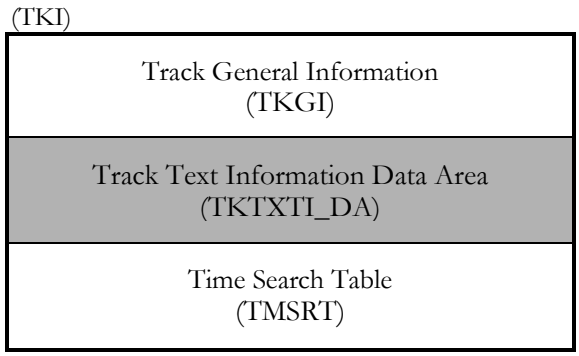
CONFIDENTIAL

4.2.1.2 Track Text Information Data Area (TKTXI_DA)

TKTXTI_DA describes text of TKI.

Even in the case that text does not exist, TKTXTI_DA is reserved. The size of TKTXTI_DA is fixed at 256 bytes.

◊ Figure 4.2.1.2-1 : Track Text Information Data Area (TKTXTI_DA)



Each item's text data is described in TKTXTI_DA.

In TKTXTI_DA, describes text data whose character set code is TKI_TI1_ATR, followed by terminated code, followed by text data whose character set code is TKI_TI2_ATR, followed by terminated code.

When only one type of text data whose character set code is TKI_TI1_ATR exists, describes text data followed by terminated code.

When only one type of text data whose character set code is TKI_TI2_ATR exists, describes terminated code, then text data followed by terminated code. Each item starts with item's tag, followed by text data.

It is not necessary to describe all tags, and only tags which have text data are needed to be described with their text data.

Text item's tag is shown in [Table 4.2.1.2-1]. The description order of these tags is not required to accord with this table.

When a Track consists of multiple TKIs, only a text described in the head TKTXTI_DA is effective.

© 2000 SD Association

CONFIDENTIAL

◊**Table 4.2.1.2-1 : Text item's tag in TKTXTI_DA**

Text Name	Tag value		Contents
	ISO646 JISX0201 ISO8859-1	Shift JIS	
TKTXTI_TTL	01h	0001h	Title
TKTXTI_ART	02h	0002h	Artist
TKTXTI_ABM	03h	0003h	Album
TKTXTI_SW	04h	0004h	Songwriter
TKTXTI_CMP	05h	0005h	music writer
TKTXTI_ARR	06h	0006h	editor
TKTXTI_PRD	07h	0007h	producer
TKTXTI_RCD	08h	0008h	Recording company
TKTXTI_MSS	09h	0009h	artist's message
TKTXTI_UCM	0Ah	000Ah	user's comment
TKTXTI_PCM	0Bh	000Bh	provider's comment
TKTXTI_CRD	0Ch	000Ch	Date
TKTXTI_GNR	0Dh	000Dh	Genre
TKTXTI_URL	0Eh	000Eh	URL
TKTXTI_FR1	0Fh	000Fh	Free 1
TKTXTI_FR2	10h	0010h	Free 2
TKTXTI_FR3	11h	0011h	Free 3
TKTXTI_FR4	12h	0012h	Free 4
TKTXTI_FR5	13h	0013h	Free 5
TKTXTI_FR6	14h	0014h	Free 6

<Terminated code>
ISO646 , JISX0201, ISO8859-1: '0x00'
Shift JIS: '0x0000'

Each text in 20 items can have variable size; however, the maximum size as a total TKTXTI_DA is not to exceed 256 (bytes).

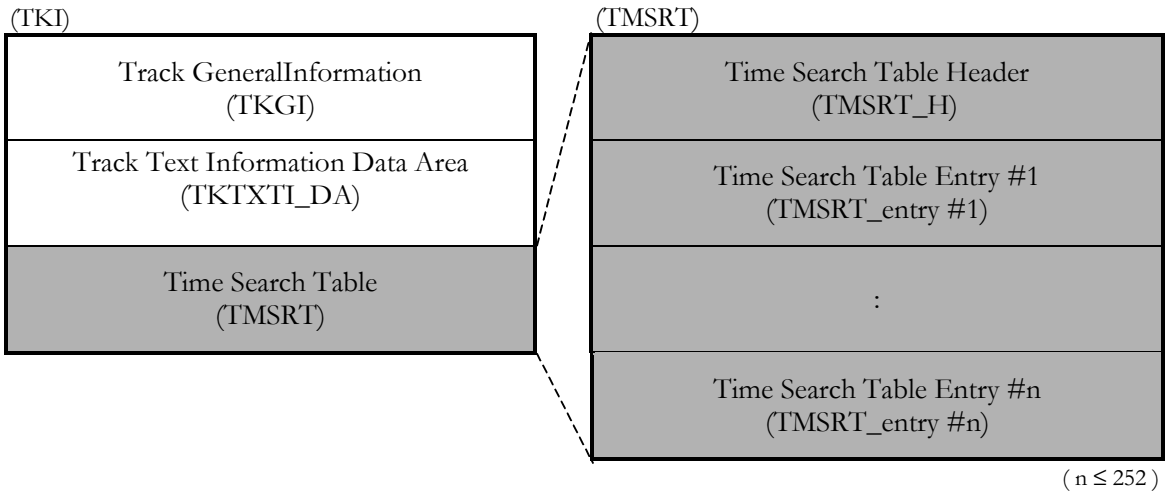
© 2000 SD Association

CONFIDENTIAL

4.2.1.3 Time Search Table (TMSRT)

TMSRT consists of TMSRT_elements that are management information of each MOB_ELEMENT.
TMSRT starts with Time Search Table Header (TMSRT_H), followed by Time Search Table entry (TMSRT_entry), as shown in [Figure 4.2.1.3-1].
Total size of TMSRT is fixed at 512 bytes.

◊ **Figure 4.2.1.3-1 TMSRT**



4.2.1.3.1 Time Search Table Header (TMSRT_H)

This is at the head of TMSRT File and keeps information concerning the entire TMSRT.

◊ **Table 4.2.1.3.1-1 : TMSRT_H**

(Description order)

RBP	Field Name	Contents	Numbers of bytes
0 to 1	TMSRT_ID	TMSRT Identifier	2 bytes
2 to 3	reserved	reserved	2 bytes
4 to 7	TMSRTE_Ns	Total Number of TMSRT _entry	4 bytes
Total			8 bytes

(RBP 0 to 1) TMSRT_ID

Describes "M5" to identify TMSRT with character set code of ISO646 (a-characters).

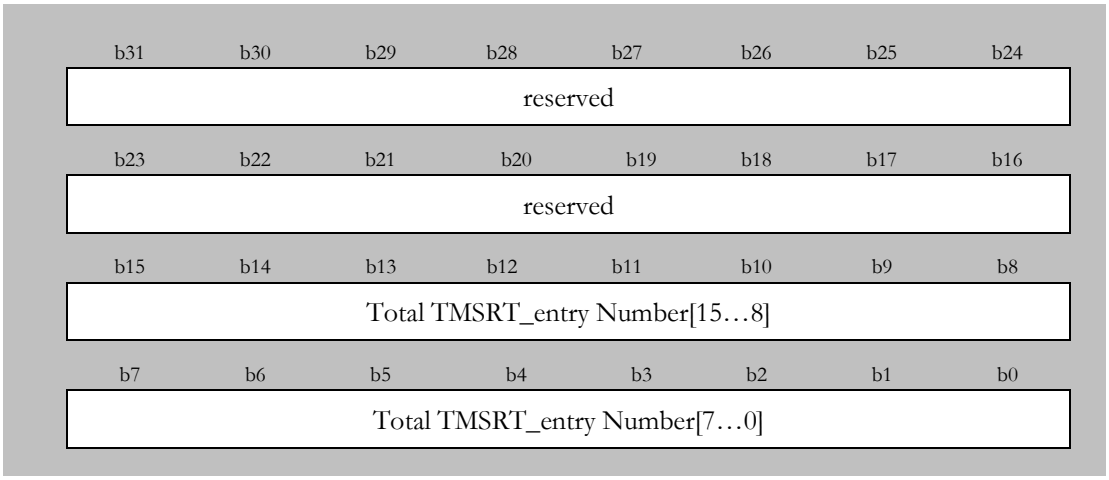
© 2000 SD Association

CONFIDENTIAL

(RBP 4 to 7) TMSRTE_Ns

Describes the total number of TMSRT_entry in the TMSRT.

When TKI is used, enter any number between '1' and '252'.



4.2.1.3.2 TMSRT_entry

TMSRT_entry describes the data size of each MOB_ELEMENT.

◊ Table 4.2.1.3.2-1 : TMSRT_entry

(Description order)

RBP	Field Name	Contents	Numbers of bytes
0 to 503	TMSRT_ENT	Data size of MOB_ELEMENT	2 bytes * 252
Total			504 bytes

(RBP 0 to 503) TMSRT_ENT

Describes the data size of MOB_ELEMENT in byte.

© 2000 SD Association

CONFIDENTIAL

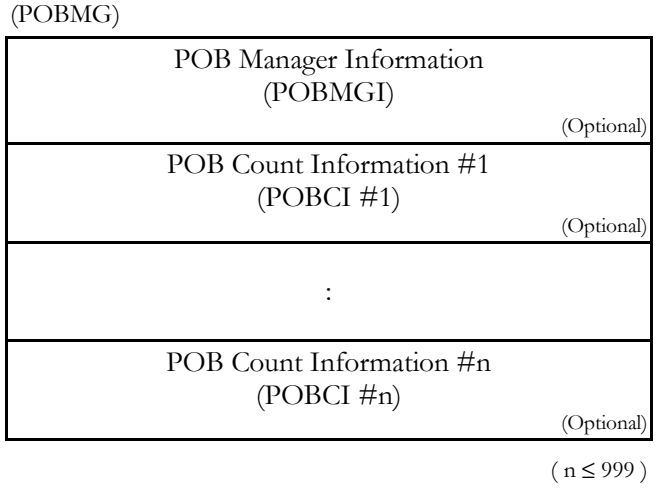
4.3 POB Manager (POBMG)

POBMG describes POB management information.

The POBMG starts with POB Manager Information (POBMGI), followed by set of POB Count Information (POBCI), as shown in [Figure 4.3-1].

A POBCI number (relative position of POBCI) corresponds to a number assigned to the POB file name.

◊ **Figure 4.3-1 : POB Manager (POBMG)**



© 2000 SD Association

CONFIDENTIAL

4.3.1 POB Manager Information (POBMGI)

As shown in [Table 4.3.1-1] , POBMGI Identifier, Number of POBs are described in POB Manager Information (POBMGI).

◊Table 4.3.1-1 : POBMGI (Description order)

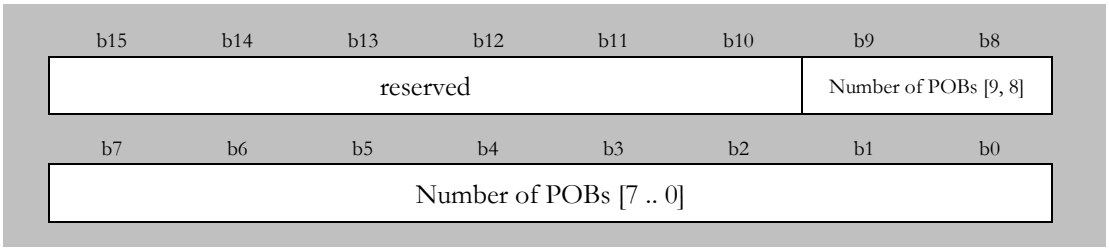
RBP		Contents	Number of bytes
0 to 1	POBMGI_ID	POBMGI Identifier	2 bytes
2 to 3	reserved	reserved	2 bytes
4 to 5	POB_Ns	Number of POBs	2 bytes
6 to 7	reserved	reserved	2 bytes
Total			8 bytes

(RBP 0 to 1) POBMGI _ID

Describes "M6" to identify POBMGI with character set code of ISO646 (a-characters).

(RBP 4 to 5) POB_Ns

Describes number of POBs in the "SD_VCyyy" directory.



Number of POBs ... Describes the number between '0' and '999'.

© 2000 SD Association

CONFIDENTIAL

4.3.2 POB Count Information (POBCI)

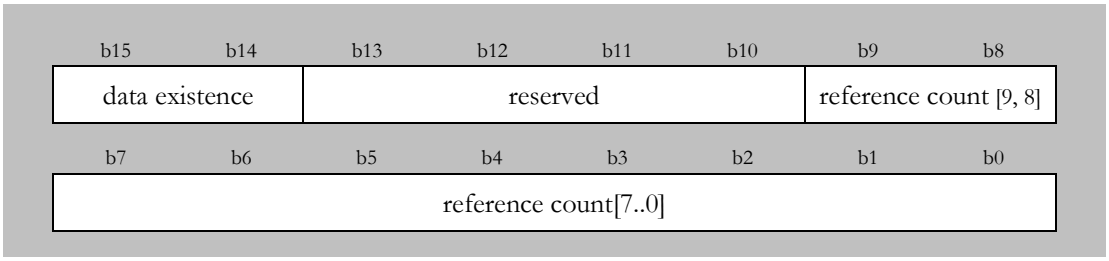
Describes whether the POB is referenced from Default Playlist, Playlist, and Track to each POB.
When POB is referenced, describes the total referenced number.
A POBCI number (relative position of POBCI) corresponds to a number assigned to the POB file name.

◊ **Table 4.3.2-1 : POBCI**

RBP		Contents	Number of bytes
0 to 1997	POB_RCN	Reference count number of POB	2 bytes * 999
1998 to 2039	reserved	reserved	42 bytes
Total			2040 bytes

(RBP 0 to 1997) POB_RCN

Describes reference count number ranging '0' to '999' which is referenced from Default Playlist, Playlist and Track. When this POB is not referenced any Default Playlist, Playlist, and Track, describes whether POB is exist or not.



reference count … Describes the reference count number between '0' and '999'.
When this POB is not referenced, enter '000h'.

data existence 00b : This POB is not exist.
01b : This POB is exist.
Others : reserved
When reference count is any number between '1' and '999', enter '01b'.

© 2000 SD Association

CONFIDENTIAL

This page is intentionally left blank

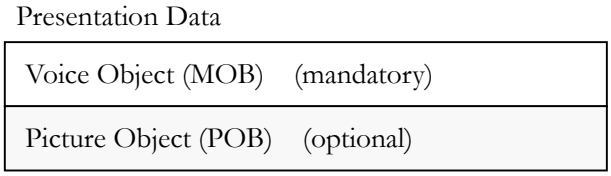
© 2000 SD Association

CONFIDENTIAL

5. Presentation Data Structure

Presentation Data consists of Voice Object (MOB) and Picture Object (POB) as shown in [Figure 5-1], which are all operated from Navigation Data.

◊ **Figure 5-1 : Presentation Data Structure**



© 2000 SD Association

CONFIDENTIAL

5.1 Voice Object (MOB)

Compressed voice message data is treated as the unit of Voice Object (MOB) in the SD-Voice Specifications. Compressed voice message formats operated by MOB are G.726 ADPCM (ITU-T G.726), AMR (3GPP TS 26.071).

5.1.1 G.726 ADPCM

G.726 ADPCM format is based on ITU-T Recommendation G.726 – 40, 32, 24, 16kbit/s Adaptive Differential Pulse Code Modulation (ADPCM).

The following restrictions shown in [Table 5.1.1-1] are applied to G.726 ADPCM operated in SD-Voice.

◊ **Table 5.1.1-1 Restrictions on G.726 ADPCM format**

bitrate per channel	between 16kbit/s(min.) and 40kbit/s(max.)
sampling_frequency	8kHz

© 2000 SD Association

CONFIDENTIAL

5.1.2 AMR

AMR format is based on 3rd Generation Partnership Project (3GPP) 3G TS 26.071: "AMR Speech Codec; General Description" and 3G TS 26.101: "AMR Speech Codec; Frame Structure".
The following restrictions shown in [Table 5.1.2-1] are applied to AMR format treated in SD-Voice.

◊ **Table 5.1.2-1 : Restrictions on AMR format**

format	AMR Interface Format 1
bitrate per channel	Variable bitrate up to 13.15kbit/s
sampling_frequency	8kHz

© 2000 SD Association

CONFIDENTIAL

5.1.3 Structure of MOB

Voice Object (MOB), which is managed by TKI, has a hierarchical structure as shown in [Figure 5.1.3-1]. An MOB consists of one MOB_BLOCK, which consists of multiple MOB_ELEMENTs. Also an MOB_ELEMENT consists of multiple MOB_FRAMEs and an MOB_FRAME consists of multiple MOB_CODEC_FRAMEs.

(1) MOB_CODEC_FRAME:

The minimum component of an MOB. An MOB_CODEC_FRAME has the same structure as a codec frame and its structure is dependent on the applied codec. In the case where sample-based codec like ITU-T G.726 ADPCM is applied, an MOB_CODEC_FRAME consists of a predetermined number of encoded samples as shown in [Table5.1.3-1].

◊ **Table 5.1.3-1 : Number of encoded samples per MOB_CODEC_FRAME**

Sampling Frequency	Number of encoded samples per MOB_CODEC_FRAME
	G.726
8kHz	40

(2) MOB_FRAME:

The minimum unit for editing operations on an MOB. An MOB_FRAME consists of a header section and a given number of consecutive MOB_CODEC_FRAMEs. The structure of the header section is dependent on the applied codec and is specified in **5.1.4 Structure of MOB_FRAME Header**. The number of MOB_CODEC_FRAMEs is fixed for each codec as shown in [Table 5.1.3-2].

◊ **Table 5.1.3-2 : Number of MOB_CODEC_FRAMEs**

Sampling Frequency	Number of MOB_CODEC_FRAMEs	
	G.726	AMR
8kHz	20	5

In the case where the number of total bits in an MOB_FRAME is not a multiple of 8 bits, a certain number of '0's are padded to the end of the MOB_FRAME in order to adjust the number of total bits to a multiple of 8 bits (Byte alignment).

(3) MOB_ELEMENT:

An MOB_ELEMENT is an aggregate of multiple contiguous MOB_FRAMEs managed by Time Search Table (TMSRT). It usually stores a fixed number of MOB_FRAMEs as described in [Table 5.1.3-3], but some MOB_ELEMENTs may have less than its fixed number of frames due to splitting the MOB etc.

© 2000 SD Association

CONFIDENTIAL

◊ **Table 5.1.3-3 : Number of MOB_FRAMEs in an MOB_ELEMENT**

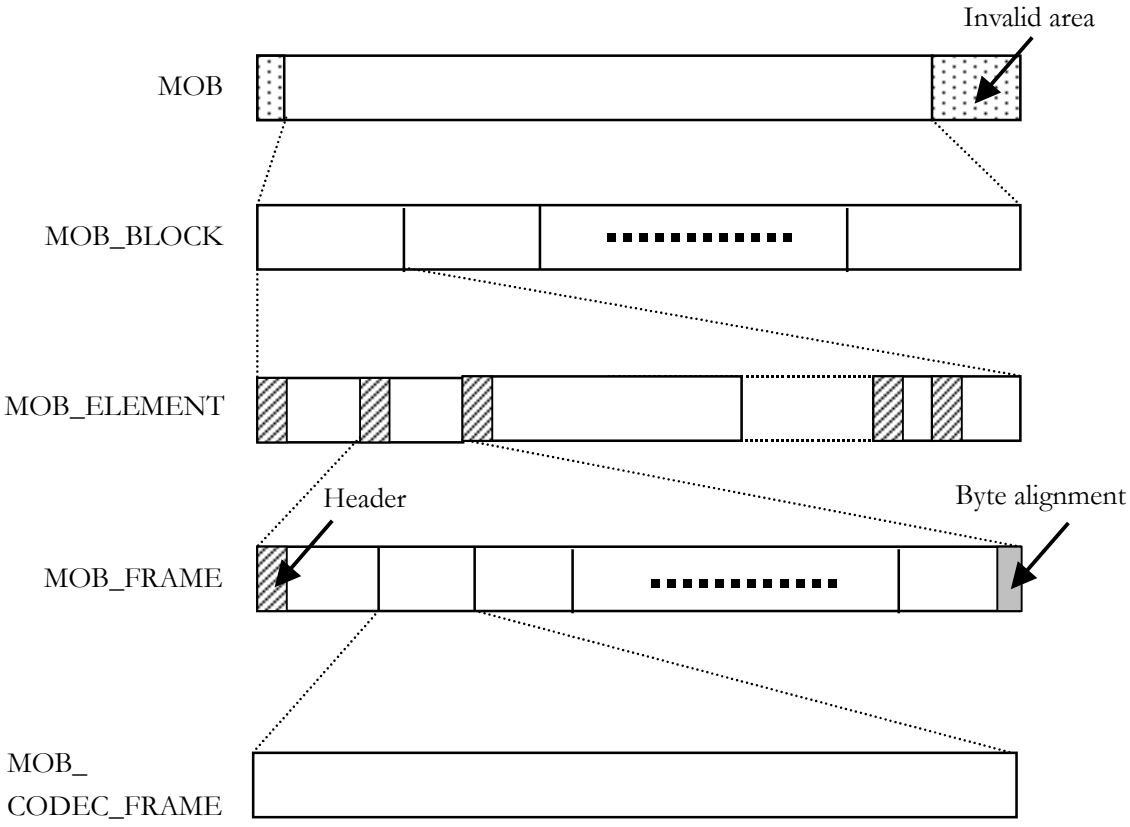
Sampling Frequency	Number of MOB_FRAMEs	
	G.726	AMR
8kHz	10*N	10*N

N indicates the presentation time duration of an MOB_ELEMNT in second.

(4) MOB_BLOCK:

An MOB_BLOCK is an area which stores contiguous valid MOB_ELEMENTs in a Voice Object (MOB) managed by TKI are stored. Only one MOB_BLOCK exists in an MOB file.

◊ **Figure 5.1.3-1 : Structure of MOB**



© 2000 SD Association

CONFIDENTIAL

5.1.4 Structure of MOB_FRAME Header

(1) MOB_FRAME Header (MOBFH)
An MOB_FRAME Header consists of one MOB_FRAME Fixed Length Header (MOBFLH) and one MOB_FRAME Variable Length Header (MOBVLH).

◊ **Figure 5.1.4-1 : Structure of MOB_FRAME Header (MOBFH)**

MOB_FRAME Header (MOBFH)

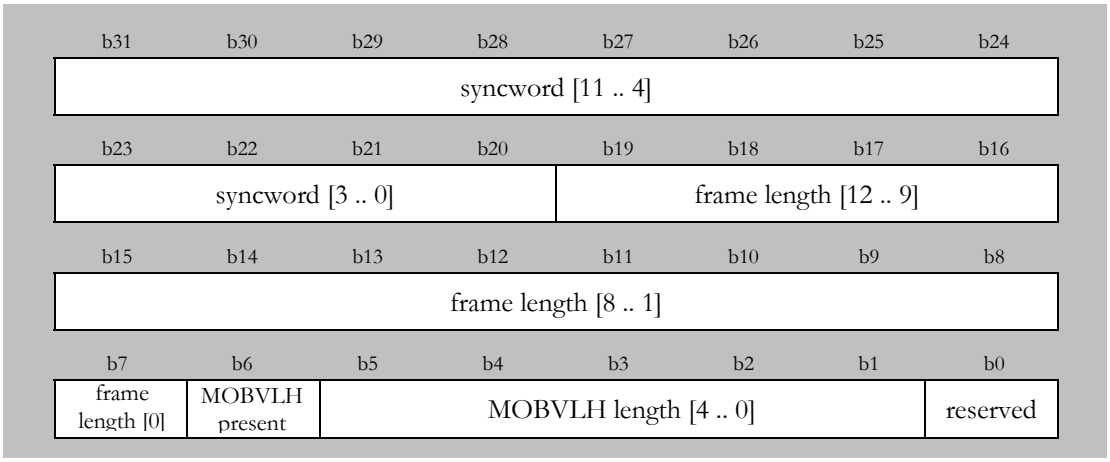
MOB_FRAME Fixed Length Header (MOBFLH)	4 bytes
MOB_FRAME Variable Length Header (MOBVLH)	0 to 31 bytes

(2) MOB_FRAME Fixed Length Header (MOBFLH)

MOB_FRAME Fixed Length Header (MOBFLH) is a mandatory part of the MOB_FRAME Header. The format of MOBFLH, 4-byte in length, is shown in [Table 5.1.4-1].

◊ **Table 5.1.4-1 : Format of MOB_FRAME Fixed Length Header (MOBFLH)**

	No. of bits	Contents
syncword	12 bits	'1111 1111 1111'
frame length	13 bits	Size of MOB_FRAME in bytes
MOBVLH present	1 bit	'0' no MOBVLH present '1' MOBVLH present
MOBVLH length	5 bits	Size of MOBVLH in bytes
	1 bit	reserved
Total	32 bits	



© 2000 SD Association

CONFIDENTIAL

(3) MOB_FRAME Variable Length Header (MOBVLH)

If an MOBVLH present bit in MOB_FRAME Fixed Length Header (MOBFLH) is set to '1', the applied codec requires additional specifications described in this field. The number of header bits and header structure are dependent on the applied codec. For the codecs specified by the coding mode in the TKI_MOB_ATR filed, the structure of MOB_FRAME Variable Length Header (MOBVLH) is defined for each codec as shown in [Table 5.1.4-2]. If a specified coding mode is vendor dependent ('1111'), the MOBVLH filed shall be vendor dependent as well. In the case where the number of total bits of MOB_FRAME Variable Length Header is not a multiple of 8 bits, a certain number of '0's are padded to the end of the MOB_FRAME Variable Length Header in order to adjust the number of total bits to a multiple of 8 bits (Byte alignment).

(4) Structure of MOB_FRAME Variable Length Header (MOBVLH)

For the codecs specified by the coding mode in the TKI_MOB_ATR filed, the structure of MOB_FRAME Variable Length Header (MOBVLH) is defined for each codec as shown in [Table 5.1.4-2].

◊ **Table 5.1.4-2 : Structure of MOB_FRAME Variable Length Header (MOBVLH)**

Codec	MOBVLH present	MOBVLH length (bytes)	Structure
G.726	'0'	0	no MOBVLH required
AMR	'0'	0	no MOBVLH required

New codecs which require variable length header might be applied in this specification. MOB_FRAME Variable Length Header (MOBVLH) is provided for such future extensions.

© 2000 SD Association

CONFIDENTIAL

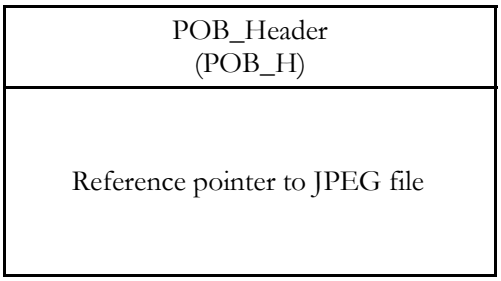
5.2 Picture Object (POB)

Picture Information handled in SD-Voice Specification is called Picture Object (POB). POB stores only a single piece of Picture Information in a POB file.
As indicated below, there are two types of POB files:

(1) Reference pointer to JPEG file

POB consists of POB Header (POB_H) and reference pointer to non-encrypted JPEG file as shown in [Figure 5.2-1].
File name is "POBxxx.SP1" where POB_ATR is set to '1.'
In the "POBxxx.SP1", there is a reference pointer to a JPEG file with character set code of ISO646. JPEG file name may contain a directory name, and the directory name may have a maximum of 2 hierarchical directories.
Directory name is described in absolute path (e.g. DirectoryName¥DirectoryName¥FileName).
This file name and JPEG file format shall be compliant with DCF specifications.

◊ **Figure 5.2-1 : Structure of POB which has reference pointer to JPEG**



(2) JPEG picture

POB consists of only JPEG picture, and does not have POB Header (POB_H).
File name is "POBxxx.JPG".
It is JPEG picture of Exif V2.1 specifications (subset)

Following formats shown in [Table 5.2-1] are JPEG picture formats to be operated on SD-Voice specifications.

© 2000 SD Association

CONFIDENTIAL

◊ **Table 5.2-1: JPEG format**

File format		Exif V2.1
Compression		JPEG
Thumbnail data format	Number of Pixels	96x96, 160x120
	Aspect ratio	4:3
	Chrominance sampling	4:2:2
	Compression ratio	Not specificated
	Huffman table	Typical Table in JPEG standard (ISO/IEC 10918-1)
Main picture data format	Number of Pixels	96x96(recommended) 640x480(recommended) 160x120 ~ 1800x1200(optional)
	Aspect ratio	4:3(recommended) Others(optional)
	Chrominance sampling	4:2:2, 4:2:0
	Compression ratio	Not specificated
	Huffman table	Typical Table in JPEG standard (ISO/IEC 10918-1)
	Color space	sRGB

As shown in [Table 5.2-1], thumbnail data and main picture data are restricted in the file format of Exif V2.1 specifications. This restriction is the same as JPEG file format of DCF specifications.

© 2000 SD Association

CONFIDENTIAL

5.2.1 POB Header (POB_H)

◊ **Table 5.2.1-1 : POB Header (POB_H)**

(Description order)

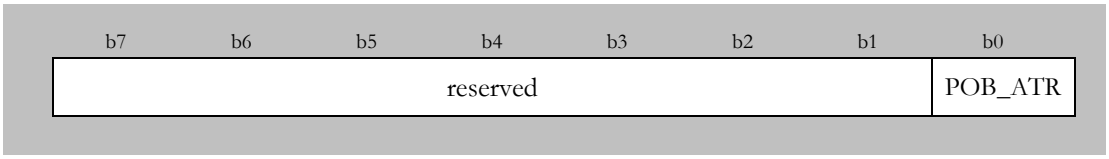
RBP	Field Name	Contents	Numbers of bytes
0 to 1	POB_ID	POB Identifier	2 bytes
2	reserved	reserved	1 byte
3	POB_ATR	POB Attribute	1 byte
4 to 7	POB_SZ	Size of POB	4 bytes
Total			8 bytes

(RBP 0 to 1) POB_ID

Describes ID that can uniquely identify POB.
This value shall be 'FFE0h'.

(RBP 3) POB_ATR

Describes whether picture data is stored in the POB file or not.
If picture data is not stored, it indicates that the POB file refers to another file.



POB_ATR ... 0b : Picture data is stored in the POB.
1b : Reference information is described in the POB.

(RBP 4 to 7) POB_SZ

Describes POB data size in bytes.

© 2000 SD Association

CONFIDENTIAL

Annex A (Normative): Definition of Character Table

A.1 ISO646 character table

ISO646 character table, which is compliant with SD-Voice format is shown in [Table A.1-1].

◊ **Table A.1-1 : ISO646 character table**

HEX	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0			sp	0	@	P	`	p								
+1			!	1	A	Q	a	q								
+2			"	2	B	R	b	r								
+3			#	3	C	S	c	s								
+4			$	4	D	T	d	t								
+5			%	5	E	U	e	u								
+6			&	6	F	V	f	v								
+7			'	7	G	W	g	w								
+8			(	8	H	X	h	x								
+9			)	9	I	Y	i	y								
+A			*	:	J	Z	j	z								
+B			+	;	K	[	k	{								
+C			,	<	L	\	l	|								
+D			-	=	M	]	m	}								
+E			.	>	N	^	n	~								
+F			/	?	O	_	o									

Reserved sp SPACE

© 2000 SD Association

CONFIDENTIAL

A.2 JISX0201 character table

JISX0201 character table, which is compliant with SD-Voice format is shown in [Table A.2-1].

◊ **Table A.2-1 : JISX0201 character table**

HEX	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0			sp	0	@	P	`	p				ー	タ	ミ		
+1			!	1	A	Q	a	q			。	ア	チ	ム		
+2			"	2	B	R	b	r			「	イ	ツ	メ		
+3			#	3	C	S	c	s			」	ウ	テ	モ		
+4			$	4	D	T	d	t			、	エ	ト	ヤ		
+5			%	5	E	U	e	u			・	オ	ナ	ユ		
+6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
+7			'	7	G	W	g	w			ァ	キ	ヌ	ラ		
+8			(	8	H	X	h	x			ィ	ク	ネ	リ		
+9			)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
+A			*	:	J	Z	j	z			ェ	コ	ハ	レ		
+B			+	;	K	[	k	{			ォ	サ	ヒ	ロ		
+C			,	<	L	¥	l	|			ャ	シ	フ	ワ		
+D			-	=	M	]	m	}			ュ	ス	ヘ	ン		
+E			.	>	N	^	n	~			ョ	セ	ホ	゛		
+F			/	?	O	_	o				ッ	ソ	マ	゜		

Reserved　　sp SPACE

© 2000 SD Association

CONFIDENTIAL

A.3 ISO8859-1 character table

ISO8859-1 character table, which is compliant with SD-Voice format is shown in [Table A.3-1].

◊ **Table A.3-1: ISO8859-1 character table**

HEX	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
+0			sp	0	@	P	`	p				°	À	Ð	à	ð
+1			!	1	A	Q	a	q			¡	±	Á	Ñ	á	ñ
+2			"	2	B	R	b	r			¢	²	Â	Ò	â	ò
+3			#	3	C	S	c	s			£	³	Ã	Ó	ã	ó
+4			$	4	D	T	d	t			¤	´	Ä	Ô	ä	ô
+5			%	5	E	U	e	u			¥	µ	Å	Õ	å	Õ
+6			&	6	F	V	f	v			¦	¶	Æ	Ö	æ	Ö
+7			'	7	G	W	g	w			§	·	Ç	×	ç	÷
+8			(	8	H	X	h	x			¨	¸	È	Ø	è	ø
+9			)	9	I	Y	i	y			©	¹	É	Ù	é	ù
+A			*	:	J	Z	j	z			ª	º	Ê	Ú	ê	ú
+B			+	;	K	[	k	{			«	»	Ë	Û	ë	û
+C			,	<	L	\	l	|			¬	¼	Ì	Ü	ì	ü
+D			-	=	M	]	m	}			-	½	Í	Ý	í	ý
+E			.	>	N	^	n	~			®	¾	Î	Þ	î	þ
+F			/	?	O	_	o				¯	¿	Ï	ß	ï	ÿ

Reserved sp SPACE

© 2000 SD Association

CONFIDENTIAL

This page is intentionally left blank

© 2000 SD Association

CONFIDENTIAL

SD Memory Card Specifications

Part 7
pDOCUMENT SPECIFICATION

Version 1.0

October 2000

Technical Committee
SD Association

CONFIDENTIAL

Conditions for publication

Publisher and Copyright Holder
SD Association

Exemption

None will be liable for any damages from use of this document.

© 2000 SD Association

Copy right Notice

TIFF is a trademark of Adobe System Incorporated.

© 2000 SD Association

CONFIDENTIAL

This page is intentionally left blank

© 2000 SD Association

CONFIDENTIAL

INDEX

Part 7 pDOCUMENT SPECIFICATION

Conditions for publication i
1. General 1
1.1 Scope 1
1.2 General Specifications of Presentation Data 2
1.3 Normative References 3
2. Technical Elements 5
2.1 Definitions 5
2.2 Symbols 6
2.3 Notations 6
2.3.1 Numerical notation 6
2.3.2 Bit fields 6
2.4 Terminology 7
2.5 Abbreviation 8
3. Overview 9
3.1 Content Model 9
3.2 Directory and File Structure 11
3.2.1 Plain Text 11
3.2.2 Raster Image 13
3.2.3 Job 13
3.2.4 Navigation Data Structure for SD-pDocument 15
3.2.5 Navigation Data Structure for Plain Text Data 16
3.2.6 Navigation Data Structure for Raster Image data 19
3.3 Editing Procedure (Informative) 22
3.3.1 Initializing Procedure 22
3.3.2 Plain text Editing Procedure 22
4. Navigation Data Structure 26
4.1 Navigation Data Structure for Plain Text Data 26
4.1.1 Plain Text Manager (PTMG) 26
4.1.2 Plain Text Manager Information(PTMGI) 26
4.1.3 Plain Text Information (PTI) 27
4.2 Navigation Data Structure for Raster Image Data 32
4.2.1 Raster Image Manager (RIMG) 32
4.2.2 Raster Image Manager Information(RIMGI) 33
4.2.3 Raster Image Information (RII) 34
5. Raster Image Data Structure 42
5.1 Monochrome Binary Image 43
5.1.1 Required TIFF Fields 43
5.1.2 Recommended TIFF Fields 45
5.1.3 Other TIFF Fields 46
5.1.4 File Structure 46
5.2 Color Image 48
5.2.1 Required TIFF Fields 48
5.2.2 Recommended TIFF Fields 51
5.2.3 Other TIFF Fields 51
5.2.4 File Structure 51
5.2.5 JPEG Stream 51
5.2.6 Relation from JFIF 51
5.2.7 Relation from DCF(Exif) 52

© 2000 SD Association

CONFIDENTIAL

6. JOB File 53
6.1 Navigation Data Structure for Job File 54
6.2 Structure of Job File 55
6.3 Size of Job File 57
6.4 Job File Header 58
6.5 Job Information (JI) 59
Annex A. Functional Requirements for SD_pDocument specification I
Annex B. Character Set Code II
Annex C. Language Tag III

© 2000 SD Association

CONFIDENTIAL

List of Figures

◊ Figure 1.1-1Structure of SD-pDocument Specifications 1
◊ Figure 2.5-1 Three Media under SD-pDocument..... 9
◊ Figure 3.1-1 The hierarchical structure to present the contents of Plain Text 9
◊ Figure 3.1-2 The hierarchical structure to present the contents of Rater Image..... 9
◊ Figure 3.1-3 The hierarchical structure to present Job data 10
◊ Figure 3.2-1 Directory and File Configuration 11
◊ Figure 3.2-2 Plain Text Manager (PTMG)..... 12
◊ Figure 3.2-3 Raster Image Manager (RIMG) 13
◊ Figure 3.2-4 Job File 14
◊ Figure 3.2-5 Structure of Manager 15
◊ Figure 3.2-6 Navigation Structure for Plain Text Data..... 16
◊ Figure 3.2-7 Plain Text Manager 17
◊ Figure 3.2-8 Navigation Structure for Raster Image data..... 19
◊ Figure 3.2-9 Raster Image Manager..... 20
◊ Figure 3.3-1 Adding a file 22
◊ Figure 3.3-2 Deleting a file..... 24
◊ Figure 4.1-1 Plain Text Manager(PTMG)..... 26
◊ Figure 4.2-1 Raster Image Manager(RIMG) 32
◊ Figure 5.1-1 TIFF file structure..... 47
◊ Figure 6.1-1 Navigation Structure for Job File..... 54
◊ Figure 6.2-1 Structure of Job File..... 55
◊ Figure 6.3-1 Size of Job File 57
◊ Figure 6.5-1 Long Side Binding : left/top..... 62
◊ Figure 6.5-2 Long Side Binding : right/bottom 62
◊ Figure 6.5-3 Short Side Binding : top/right..... 62
◊ Figure 6.5-4 Short Side Binding : bottom/left 62

© 2000 SD Association

CONFIDENTIAL

List of Tables

◊ TABLE 1-1 Hierarchical structure for contents classification 2
◊ Table 1-2 Presentation Data 2
◊ Table 3-1 Specifications of naming convention of Plane Text File 12
◊ Table 3-2 Specifications of naming convention of Raster Image File 13
◊ Table 4-1 PTMGI 26
◊ Table 4-2 PTI 27
◊ Table 4-3 RIMGI 33
◊ Table 4-4 RII 34
◊ Table 4-5 Item's Tags in Comment Field of RII 38
◊ Table 4-6 Example of contents stated with Tags in Comment field 39
◊ Table 6-1 Job File Header 58
◊ Table 6-2 JI 59
◊ Table A-1 Classifications I
◊ Table A-2 Categories of products I

© 2000 SD Association

CONFIDENTIAL

1. General

1.1 Scope

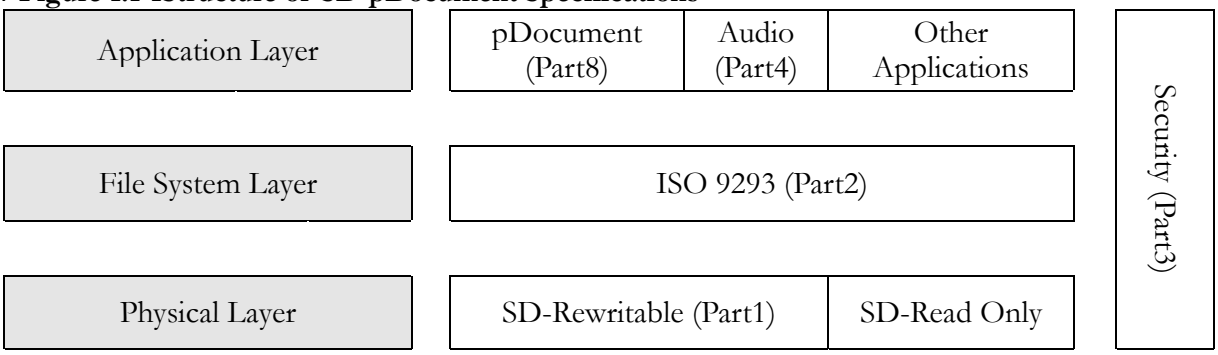
This part specifies the application format of the SD-pDocument files that are recorded in the SD Memory Card. The SD-pDocument files include multi page raster image data and plain text data that will be printed on printing materials or displayed as slide show. The SD-pDocument specification also defines JOB FILE that produces automatic printing when a SD Memory Card is inserted into an equipment like a printer.

The following features are not supported in this version of the SD-pDocument specification. These features will be supported in later version of the SD-pDocument specification if necessary.

1) CPRM
2) Complex printing conditions
3) Job type extension (ex. Fax sending)

This specification does NOT define to be used by CPRM guarded. The card of which the data is defined in this specifications shall comply with **"Part1. PHYSICAL SPECIFICATIONS", "Part2. FILE SYSTEM SPECIFICATIONS "** and **"Part3. SECURITY SPECIFICATIONS "** of SD Memory Card specifications.

◊ **Figure 1.1-1Structure of SD-pDocument Specifications**



© 2000 SD Association

CONFIDENTIAL

1.2 General Specifications of Presentation Data

◊ TABLE 1-1 Hierarchical structure for contents classification

Hierarchy name	Maximum number
Plane Text Manager	1
Plane Text Information	65,535*
Plane Text File	65,535*
Raster Image Manager	1
Raster Image Information	65,535*
Raster Image File	65,535*
Job File	1
Job Information	255

*In FAT File System, maximum total number of logical files is limited by total number of clusters (ex. On a FAT12 64MB SD memory card, total number of clusters is 4053.)

◊ Table 1-2 Presentation Data

Data	General specification	
TEXT	Text Level 0	: US-ASCII
	Text Level 1	: National Character code set , for example: Japanese: JISX0201, Shift-JIS European languages: ISO8859 Latin (code set or encoding name must be accepted by IANA)
	Text Level 2	ISO10646 / UTF-8
RASTER IMAGE	Raster Level 0	: monochrome binary image
	Raster Level 1	: continuous tone (gray scale) / color

© 2000 SD Association

CONFIDENTIAL

1.3 Normative References

1) ISO/IEC 10646-1:1993
Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -
Part 1: Architecture and Basic Multilingual Plane (available in English only)
2) ISO 8859-1: 1987
Information processing – 8-bit single-byte coded graphic character sets – Part 1
Latin alphabet No.1
3) JIS X 0201-1997
7-bit and 8-bit coded character sets for information interchange (ISO/IEC 646)
4) JIS X 0208:1997
7-bit and 8-bit coded 2 byte character sets for information interchange
5) ISO/IEC 10646-1:1993/Amd 2:1996
UCS Transformation Format 8 (UTF-8)
6) Shift-JIS IANA MIB enum: 17
7) JEIDA DCF Version 1.0:1998
Design rule for Camera File System
Japan Electronic Industry Development Association.
http://www.jeida.or.jp
8) JEIDA Exif Version 1.0:1998
Digital Still Camera Image File Format Standard
(Exchangeable image file format for Digital Still Camera: Exif)
Japan Electronic Industry Development Association.
9) [T.43] ITU-T Recommendation T.43, Colour and gray-scale image representations using lossless coding scheme for facsimile, February 1997
10) [T.81] ITU-T Recommendation T.81, Information technology - Digital compression and coding of continuous-tone still images – Requirements and guidelines, September 1992
11) [T.82] ITU-T Recommendation T.82, Information technology - Coded representation of picture and audio information - Progressive bi-level image compression, March 1995
12) [T.85] ITU-T Recommendation T.85, Application profile for Recommendation T.82 - Progressive bi-level image compression (JBIG coding scheme) for facsimile apparatus, August 1995
13) [TIFF] Tag Image File Format, revision 6.0.
Adobe Developers Association, June 3, 1992
ftp://ftp.adobe.com/pub/adobe/devrelations/devtechnotes/pdffiles/tiff6.pdf
The TIFF 6.0 specification dated June 3, 1992 specification (c)
1986-1988, 1992 Adobe Systems Incorporated. All Rights Reserved.
14) [TIFF-FX] Parsons, G. and J. Rafferty, "Tag Image File Format (TIFF) - F Profile for Facsimile", RFC 2306, March 1998.

© 2000 SD Association

CONFIDENTIAL

15) [TIFF-F0] TIFF Class F specification, Apr 28, 1990,
ftp://ftp.faximum.com/pub/documents/tiff_f.txt
16) [TIFF-REG] Parsons, G., Rafferty J. and S. Zilles, "Tag Image File
17) Format (TIFF) - image/tiff MIME Sub-type Registration", RFC 2302, March 1998.
18) [TTN1] Adobe PageMaker 6.0 TIFF Technical Notes, Sept. 14, 1995,
http://www.adobe.com/supportservice/devrelations/PDFS/TN/TIFFPM6.pdf
19) [TTN2] Draft TIFF Technical Note 2, Replacement TIFF/JPEG specification, March 17, 1995,
ftp://ftp.sgi.com/graphics/tiff/TTN2.draft.txt
20) [RFC2301] File Format for Internet FAX Buckley, et. Al., August 1999.
21) ISO/IEC 10918-1: 1994 Digital compression and coding of continuous tone still pictures: requirements and guidelines
22) [sRGB] Default RGB color space IEC/3WD 61966-2.1: 1998
http://w3.hike.te.chiba-u.ac.jp/IEC/100/PT61966
23) [JFIF]
JPEG File Interchange Format Version 1.02 : September 1, 1992
http://www.w3.org/Graphics/JPEG/jfif.txt
24) ISO8601
ISO (International Organization for Standardization). Representations of dates and times, 1988-06-15. Available at: http://www.iso.ch/markete/8601.pdf
25) ISO 8601 Draft Revision
ISO (International Organization for Standardization). Representations of dates and times, draft revision, 1998.
26) [T.4] ITU-T Recommendation T.4, Standardization of group 3 facsimile apparatus for document transmission, October 1997.
27) [T.42]ITU-T Recommendation T.42, Continuous-tone colour representation method for facsimile, February 1996
28) [T.6] ITU-T Recommendation T.6, Facsimile coding schemes and coding control functions for group 4 facsimile apparatus, November 1988
29) IANA : Internet Assigned Numbers Authority http://www.iana.org

© 2000 SD Association

CONFIDENTIAL

2. Technical Elements

2.1 Definitions

For the purpose of this specification, the following definitions apply:

RBP	:	Relative Byte Position within a descriptor, starting with zero.
Reserved	:	Reserved field. The term "reserved" indicates that the value may be used in the future for SD Memory Card specifications. Unless other wise specified within this specification, all reserved bits shall be set to zero.

© 2000 SD Association

CONFIDENTIAL

2.2 Symbols

Relational operators

== or =	Equal to

Arithmetic operators

+	Addition
-	Subtraction
*	Multiplication
/	Division

2.3 Notations

2.3.1 Numerical notation

Numbers in decimal notation are represented by decimal digits, namely 0 to 9. Digit(s) represents any digit from digit 0 to 9.

Numbers in Hexadecimal notation are represented as a sequence of one or more hexadecimal digits namely 0 to 9 and A to F, suffixed by "h" or prefixed by "0x".

Multiple-byte numerical values in a description field MUST be recorded in the **BIG ENDIAN** representation. e.g.
the 16-bit hexadecimal number '1234h' shall be recorded as '12h', '34h'.

2.3.2 Bit fields

Certain fields or parts of fields are interpreted as an array of bits.

This array of bits shall be referred to as a bit field. Bit positions within an ***n*** bit field are numbered such that the least significant bit is numbered 0 and the most significant bit is numbered ***n–1***.

© 2000 SD Association

CONFIDENTIAL

2.4 Terminology

SD-pDocument Data

Plain Text data and multi page Raster Image data stored in SD Memory Card. Encoding information of the data is explicitly described.

Plain Text Data

Unformatted text data.

Raster Image Data

Multi page image data based on TIFF.

Job File

A file including Job information for invoking Jobs automatically. (ex. Print , Display)

File Object

A file of SD-pDocument Data.

Used

A data structure is "used" when it represents certain data. Overwriting this data structure is not permitted.

Unused

A data structure is "unused" when it does not represent any data. Overwriting this data structure is permitted.

Reader/Writer

Any devices access SD Memory Card.
Readers reproduce SD-pDocument Data on SD memory cards.
(Readers may modify or delete Job information in JOB FILE.)
Writers store or delete SD-pDocument Data or Job information from SD memory cards.

© 2000 SD Association

CONFIDENTIAL

2.5 Abbreviation

PTMG	Plain Text Manager
PTMGI	Plain Text Manager Information
PTI	Plain Text Information
PTMG_PTI_Ns	Number of Plain Text Information exist
PTMG_PTI_NUSED	Number of used Plain Text Information
PTMG_PTI_FUN	First used Plain Text Information Number
P_PTI_SRN	Previous Plain Text Information Search Number
N_PTI_SRN	Next Plain Text Information Search Number
RIMG	Raster Image Manager
RIMGI	Raster Image Manager Information
RII	Raster Image Information
RIMG_RII_Ns	Number of Raster Image Information exist
RIMG_RII_NUSED	Number of used Raster Image Information
RIMG_RII_FUN	First used Raster Image t Information Number
P_RII_SRN	Previous Raster Image Information Search Number
N_RII_SRN	Next Raster Image Information Search Number
CHAR_CODE	Character Set Code
FOB	File Object
FL_SRN	File Search Number
SFL_SRN	File Search Number of Source File
TPAGE	Total Page
SPAGE	Start Page
EPAGE	End page
JOBINF	Job Information
JI	Job Information
JI_Ns	number of Job Information exist
JI_NUSED	Number of used Job Information
JI_FUN	First used Job Information Number
SFL_SRN	Source File Search Number
P_JI_SRN	Previous Job Information Search Number
N_JI_SRN	Next Job Information Search Number
UI	Used Information
TIFF	Tag Image File Format
IFD	Image File Directory
JFIF	JPEG File Interchange Format
IANA	Internet Assigned Numbers Authority
JPEG	Joint Photographic Experts Group
JBIG	Joint Bi-level Image Experts Group
MH	Modified Huffman [T.4]
MR	Modified Read [T.4]
MMR	Modified MR [T.6]

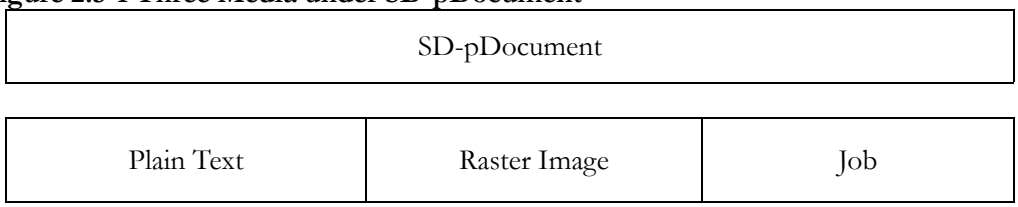
© 2000 SD Association

CONFIDENTIAL

3. Overview

This specification manages to store Plain Text data and/or multi page Raster Image data , and to print or display the data.

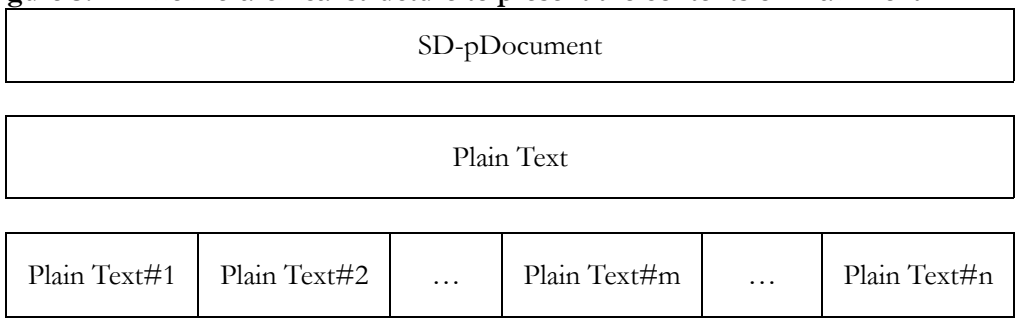
◊ **Figure 2.5-1 Three Media under SD-pDocument**



3.1 Content Model

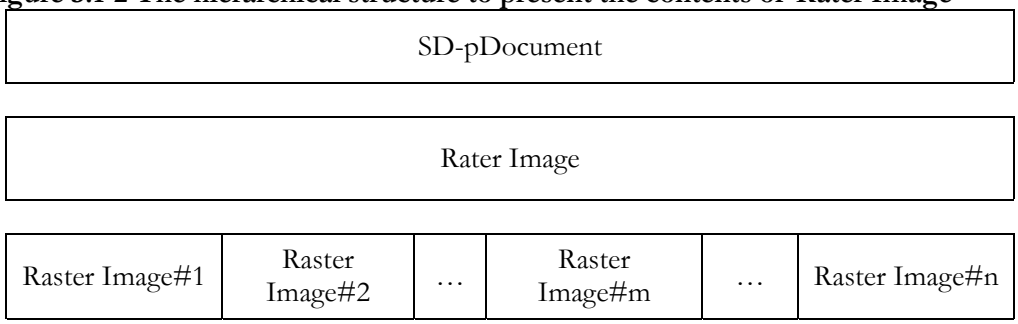
- Plain Text

◊ **Figure 3.1-1 The hierarchical structure to present the contents of Plain Text**



- Raster Image

◊ **Figure 3.1-2 The hierarchical structure to present the contents of Rater Image**

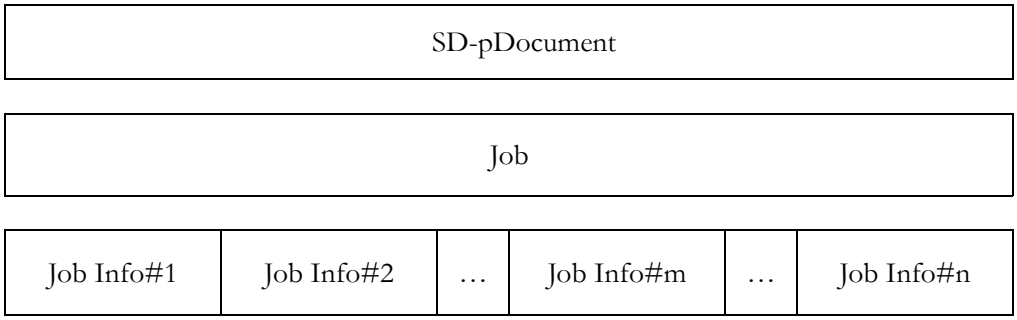


© 2000 SD Association

CONFIDENTIAL

- Job

◊ **Figure 3.1-3 The hierarchical structure to present Job data**



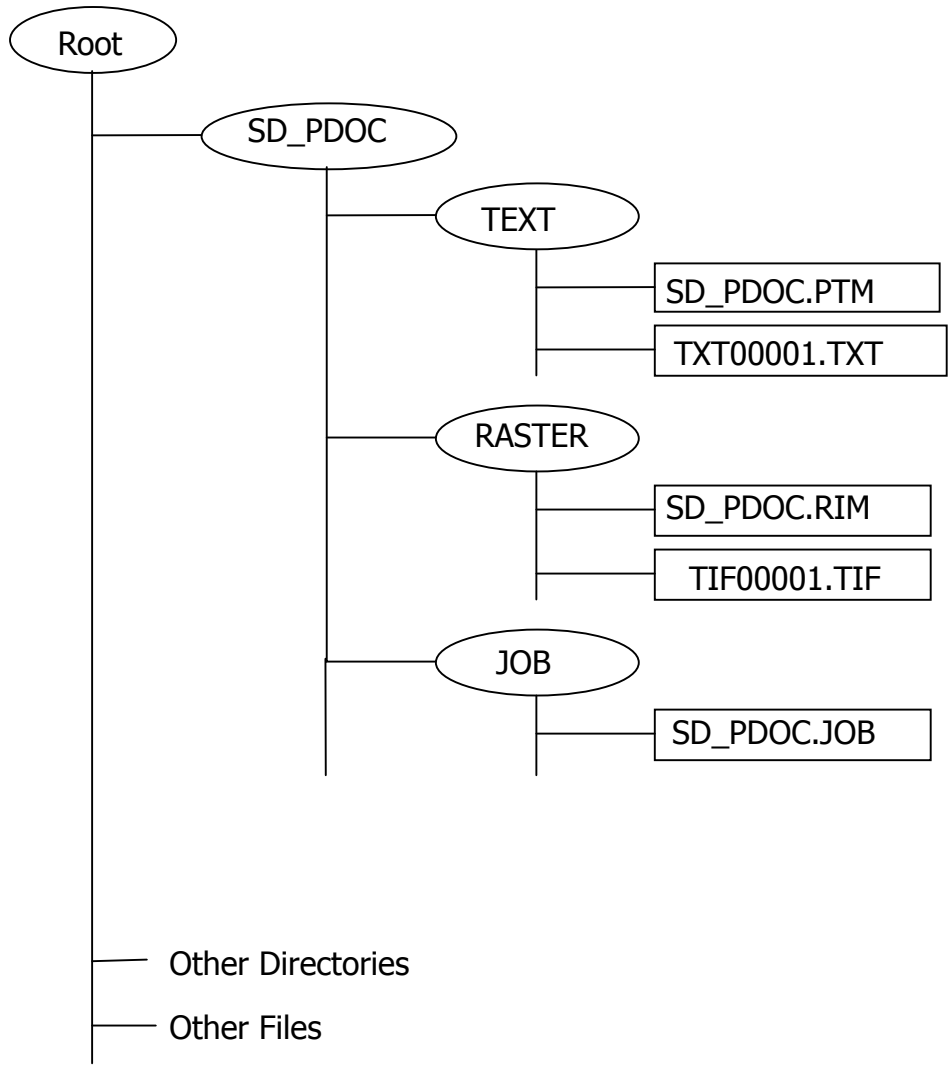
© 2000 SD Association

CONFIDENTIAL

3.2 Directory and File Structure

[Figure 3.2-1] illustrates the directory and files where the data which comply with this Specifications are recorded. All files are placed under SD_PDOC directory.

◊ **Figure 3.2-1 Directory and File Configuration**



- **SD_PDOC (Directory)**
  All files that are based on SD-pDocument specifications are placed under this directory.

3.2.1 Plain Text

- **TEXT (Directory)**
  TEXT is a directory containing Plain Text Manager (PTMG) and text files.

- **SD_PDOC.PTM (File)**
  The management information for all of plain text files is described under the Plain Text Manager as shown in [Figure 3.2-2] (see 4.1 for details)

© 2000 SD Association

CONFIDENTIAL

If the Plain Text Manager file does not present, a Writer shall create it when the Writer writes Plain Text Information.

◊ **Figure 3.2-2 Plain Text Manager (PTMG)**

PTMG

Plain Text Manager
Plain Text Manager Information
Plain Text Information #1
Plain Text Information #2
. .
Plain Text Information #n

(n ≤ 65,535)

- **TXTxxxxx.TXT (File)**

Plain Text File
The file name shall be 8 characters as base name and 4 characters as extension (including period).
The first three characters shall be "TXT".
The next five characters shall be a decimal number between "00001" and "65535". The number "00000" shall not be used.
The extension of the file name shall be ".TXT".
The specifications of the naming convention of Plain Text File are shown in [Table 3-1]

◊ **Table 3-1 Specifications of naming convention of Plane Text File**

	Fixed string part			Hexadecimal number part (File Number)					Alternative string part			
Specifications	T	X	T	0	0	0	0	1	.	T	X	T
				≀								
				6	5	5	3	5				
Example	T	X	T	0	1	2	3	4	.	T	X	T

© 2000 SD Association

CONFIDENTIAL

3.2.2 Raster Image

- **RASTER (Directory)**
  A directory to store a RASTER IMAGE Manager (RIMG) and TIFF files
- **SD_PDOC.RIM (File)**
  The management information for all of TIFF files is described under the Raster Image Manager as shown in [Figure 3.2-3]. (see section 3.2.6.1 for details)
  If the Raster Image Manager file does not present, a Writer shall create it when the Writer writes Raster Image Information.

◊ Figure 3.2-3 Raster Image Manager (RIMG)

RIMG

Raster Image Manager
Raster Image Manager Information
Raster Image Information #1
Raster Image Information #2
. .
Raster Image Information #n

(n ≤ 65,535)

- **TIFxxxxx.TIF (File)**
  TIFF File (See Section 5. for detail)
  The file name shall be 8 characters as base name and 4 characters as extension (including period).
  The first three characters shall be "TIF".
  The next five characters shall be a decimal number between "00001" and "65535". The number "00000" shall not be used.
  The extension of the file name shall be ".TIF".
  The specifications of the naming convention of Raster Image File are shown in [Table 3-2]

◊ Table 3-2 Specifications of naming convention of Raster Image File

	Fixed string part			Hexadecimal number part (File Number)					Alternative string part			
Specifications	T	I	F	0	0	0	0	1	.	T	I	F
				≀								
				6	5	5	3	5				
Example	T	I	F	0	1	2	3	4	.	T	I	F

3.2.3 Job

- **Job (Directory)**
  A directory to store a JOB FILE

© 2000 SD Association

CONFIDENTIAL

- **SD_PDOC.JOB(File)**
  Job File.
  The job information for all of jobs is described in the Job File as shown in [Figure 3.2-4]. (See Section 6. for detail)

◊ **Figure 3.2-4 Job File**

Job File
Job File Header
Job Information #1
Job Information #2
. .
Job Information #n

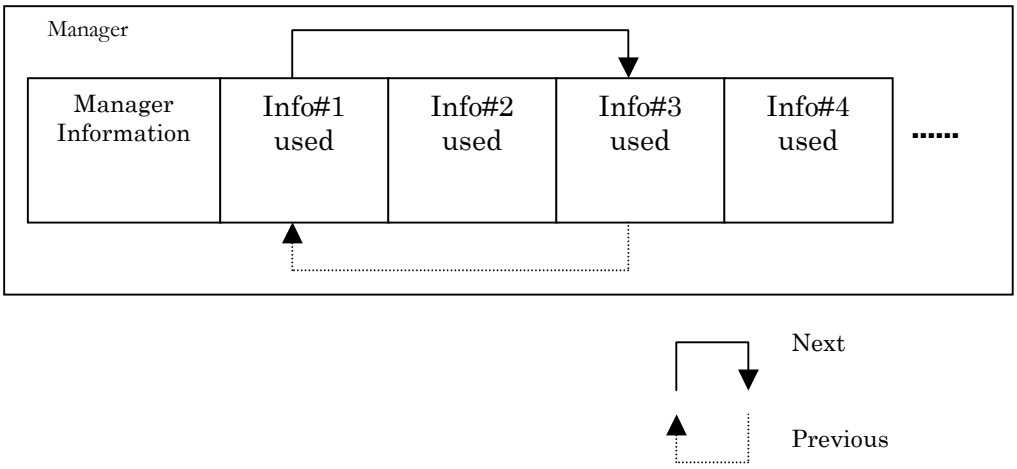
(n ≤ 255)

© 2000 SD Association

3.2.4 Navigation Data Structure for SD-pDocument

Manager in this specification has the following concept:

◊ **Figure 3.2-5 Structure of Manager**



Each manager contains one Manager Information (fixed length) and 0 or more Information (fixed length)

Directory Initialization operation will create Manager containing one Manager Information (no information yet).

When files are add into the directory, Writer read Manager Information from Manager inside directory. If total number of Information blocks and total number of used Information blocks are same, all Information blocks in the Manager is used and a new Information block has to be created. If total number of Information blocks is less than total number of used Information blocks, then Writer should search for the first unused Information block and uses it. The number part of a file is related directly to Information block number. (ie. PTI#00002 represent file TXT00002.txt) Total number of information blocks in Manager is not less than total number of 'used' Information block.

Information block which is created once present permanently unless Manager is deleted. (Total number of Information will grow up).

Note) For easy implementation to Reader/Writer, Unused Information block will not be deleted. So, Garbage Collection of Information block in Manager will not be done. But the Garbage Collection is not prohibited.

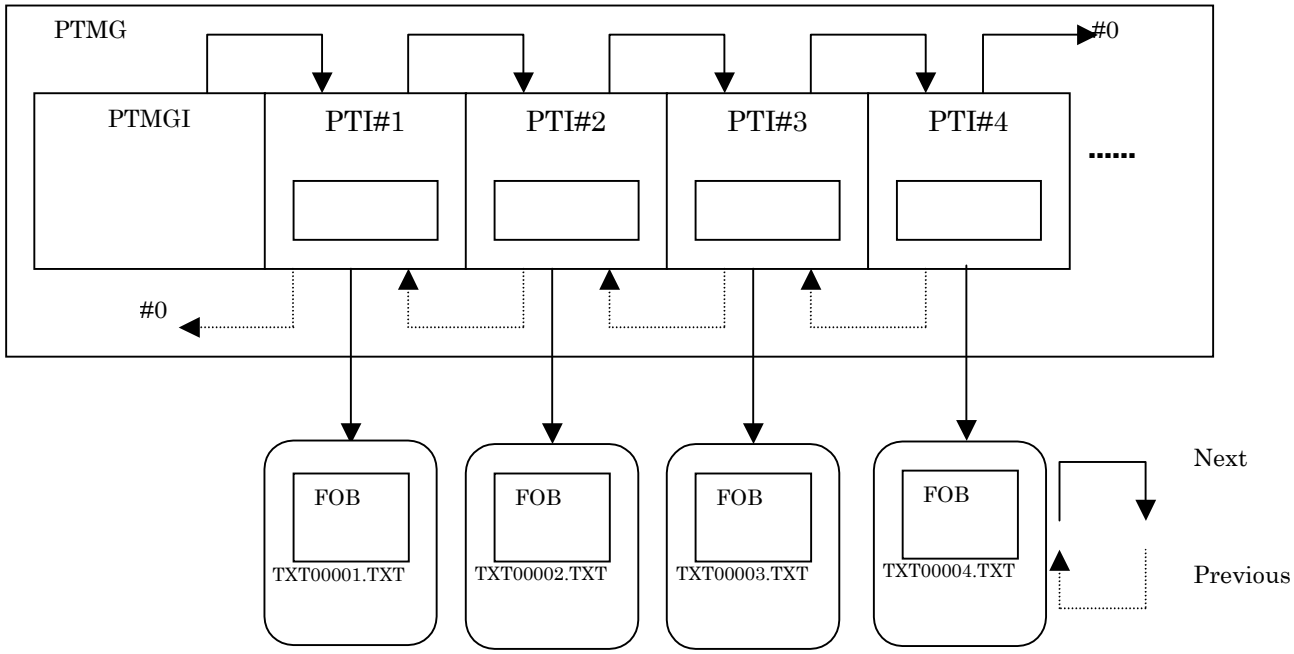
© 2000 SD Association

CONFIDENTIAL

3.2.5 Navigation Data Structure for Plain Text Data

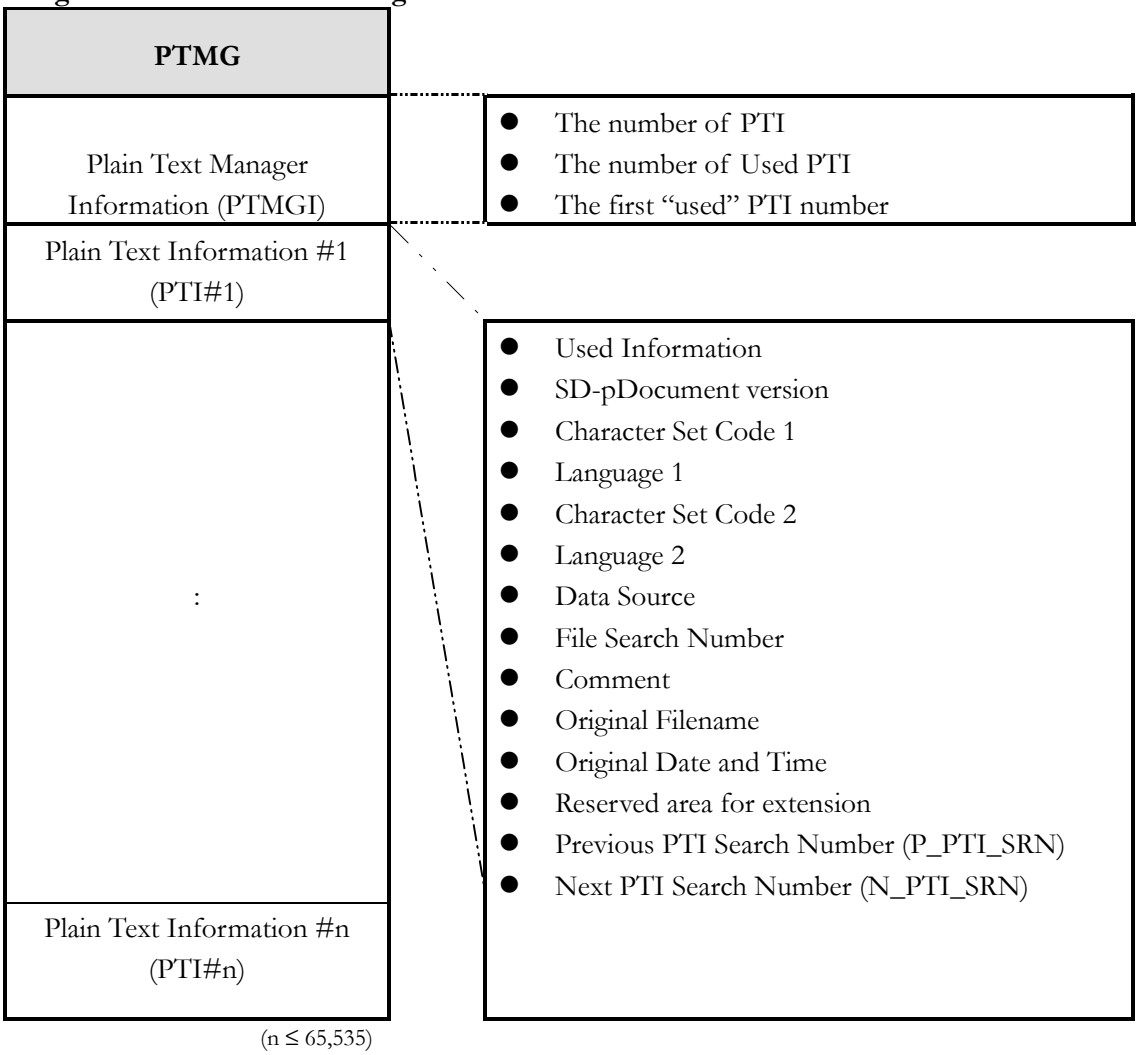
The navigation structure for data in TEXT directory is as shown in [Figure 3.2-6].

◊ **Figure 3.2-6 Navigation Structure for Plain Text Data**



3.2.5.1Structure of Plain Text Manager (PTMG)
Plain Text Manager (PTMG) consists of 2 elements: the Plain Text Manager Information (PTMGI) and the Plain Text Information (PTI).

◊ **Figure 3.2-7 Plain Text Manager**



(n ≤ 65,535)

- **Plain Text Manager Information (PTMGI)**
  PTMGI describes PTMG information such as the number of Plain Text Informations exist, the number of plain text files stored in TEXT directory and the first used PTI number.
- **Plain Text Information (PTI)**
  PTI consists of:
- **Used Information**
  This information describes if PTI is used or not.
  If a PTI points a certain Plain Text file, the PTI is 'used'.
  If a PTI does not point a valid Plain Text file, the PTI is 'unused'.
- **Version**
  the version of SD-pDocument supported
- **Character Set Code 1**
  Character code Information of the Plain Text data

© 2000 SD Association

CONFIDENTIAL

- **Language 1**
  A language used in the Plain Text data
- **Character Set Code 2**
  Character code Information of character fields in Plain Text Manager
- **Language 2**
  A language used in the character fields in Plain Text Manager
- **Data Source**
  Source of Plane Text Data (ex. Sent e-mail, received e-mail, address book)
- **File Search Number**
  File number of the Plain Text File
- **Comment**
  Additional information about the Plain Text Data
- **Original File Name**
  A file name of the Plane Text Data before renamed TXTxxxxx.TXT
- **Original Date and Time**
  Date and time at the original file created
- **Reserved area for Extension**
- **Previous PTI Search Number (P_PTI_SRN)**
- **Next PTI Search Number (N_PTI_SRN)**

© 2000 SD Association

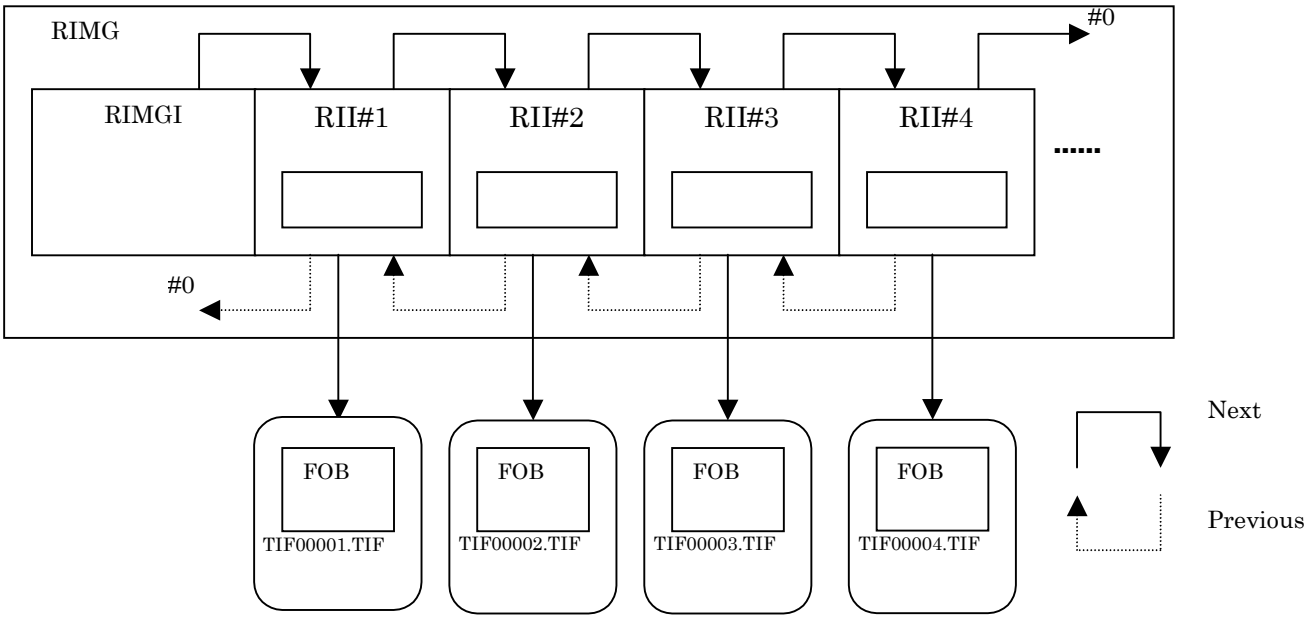
CONFIDENTIAL

3.2.6 Navigation Data Structure for Raster Image data

The navigation structure for data in RASTER directory is as shown in [Figure 3.2-8].

.

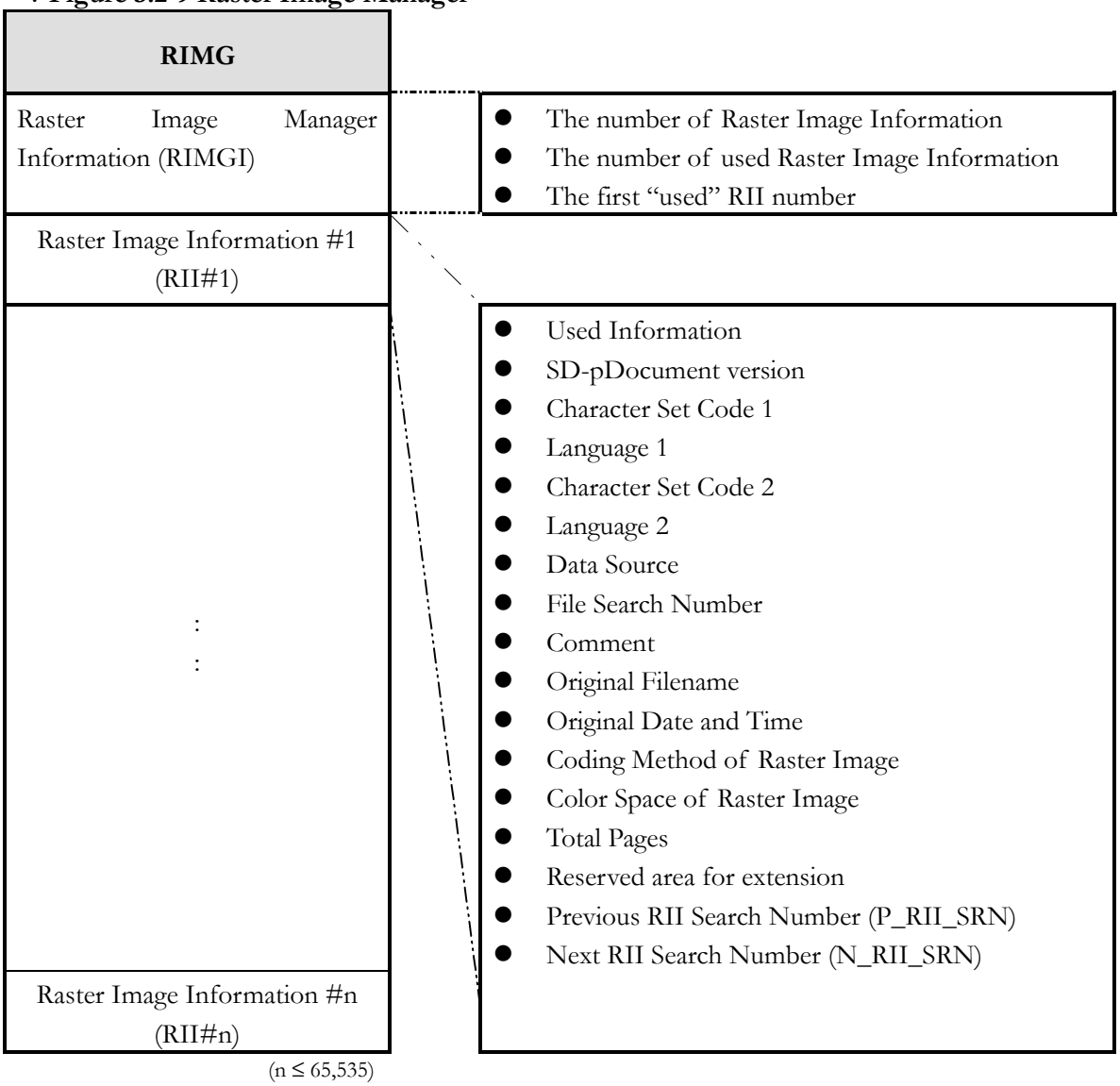
◊ **Figure 3.2-8 Navigation Structure for Raster Image data**



3.2.6.1 Structure of Raster Image Manager

Raster Image Manager (RIMG) consists of 2 elements: the Raster Image Manager Information (RIMGI) and the Raster Image Information (RII).

◊ **Figure 3.2-9 Raster Image Manager**



- **Raster Image Manager Information (RIMGI)**
  RIMGI describes RIMG information such as number of TIFF files stored in RASTER directory, existing number of Raster Image Information, and the first used RII number.

- **Raster Image Information (RII)**
  RII consists of:

- **Used Information**
  This information describes if RII is used or not.
  If a RII points a certain Raster Image file, the RII is 'used'.
  If a RII does not point a valid Raster Image file, the RII is 'unused'.
- **Version**

the version of SD-pDocument supported

- **Character Set Code 1**
  Character code Information of character fields in Tags of the Raster Image Data
- **Language 1**
  A language used in the character fields in Tags of the Raster Image Data
- **Character Set Code 2**
  Character code Information of character fields in Raster Image Manager
- **Language 2**
  A language used in the character fields in Raster Image Manager
- **Data Source**
  Source of Raster Image Data (ex. PC, Image Scanner, FAX)
- **File Search Number**
  File number of the Raster Image File
- **Comment**
  Additional information about the Raster Image Data
- **Original File Name**
  A file name of the Raster Image Data before renamed TIFxxxxx.TIF
- **Original Date and Time**
  Date and time at the original file created
- **Coding Methods of Raster Image**
  Coding Methods of the Raster Image Data
- **Color Space of Raster Image**
  Color space of the Raster Image Data
- **Total Pages**
  Total Pages of the Raster Image Data
- **Reserved area for Extension**
- **Previous RII Search Number (P_RII_SRN)**
- **Next RII Search Number (N_RII_SRN)**

© 2000 SD Association

CONFIDENTIAL

3.3 Editing Procedure (Informative)

3.3.1 Initializing Procedure

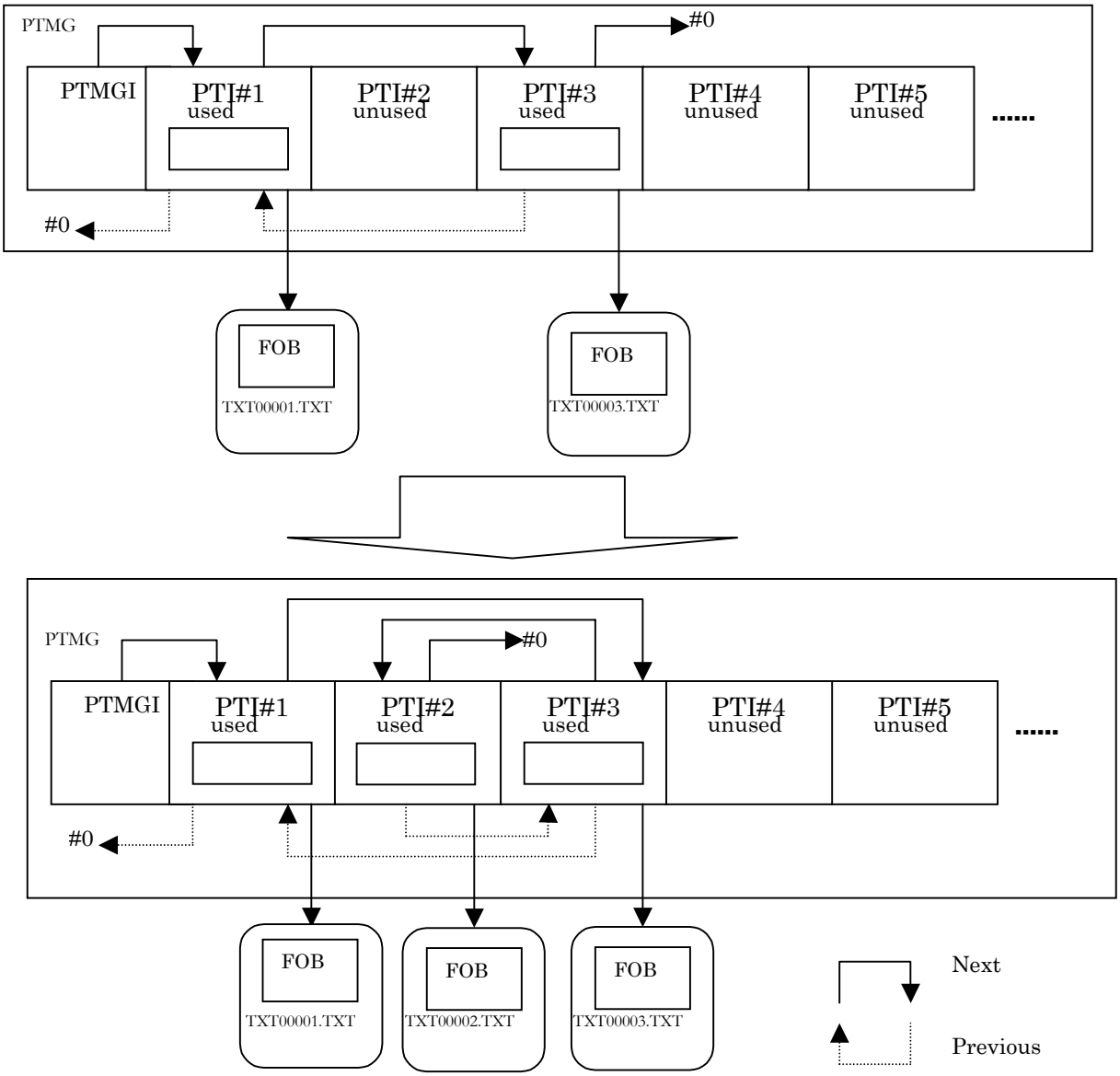
If TEXT or RASTER directory is empty, initializing process to make Manager files is necessary.

3.3.2 Plain text Editing Procedure

The following describes about Plain Text data, it is the same about Raster Image data.

3.3.2.1 Addition

◊ **Figure 3.3-1 Adding a file**



1. Search the first "unused" PTI (PTI#2)
2. Change PTI#2 to "used".
3. Renew P_PTI_SRN PTI and N_PTI_SRN of PTI#2
4. Renew N_PTI_SRN in PTI#3
5. Renew PTMGI

© 2000 SD Association

CONFIDENTIAL

6. Store a TEXT file
   Write a plain text file into the SD memory card , then name it "TXT00002.TXT"
7. Renew other information fields of PTI#2

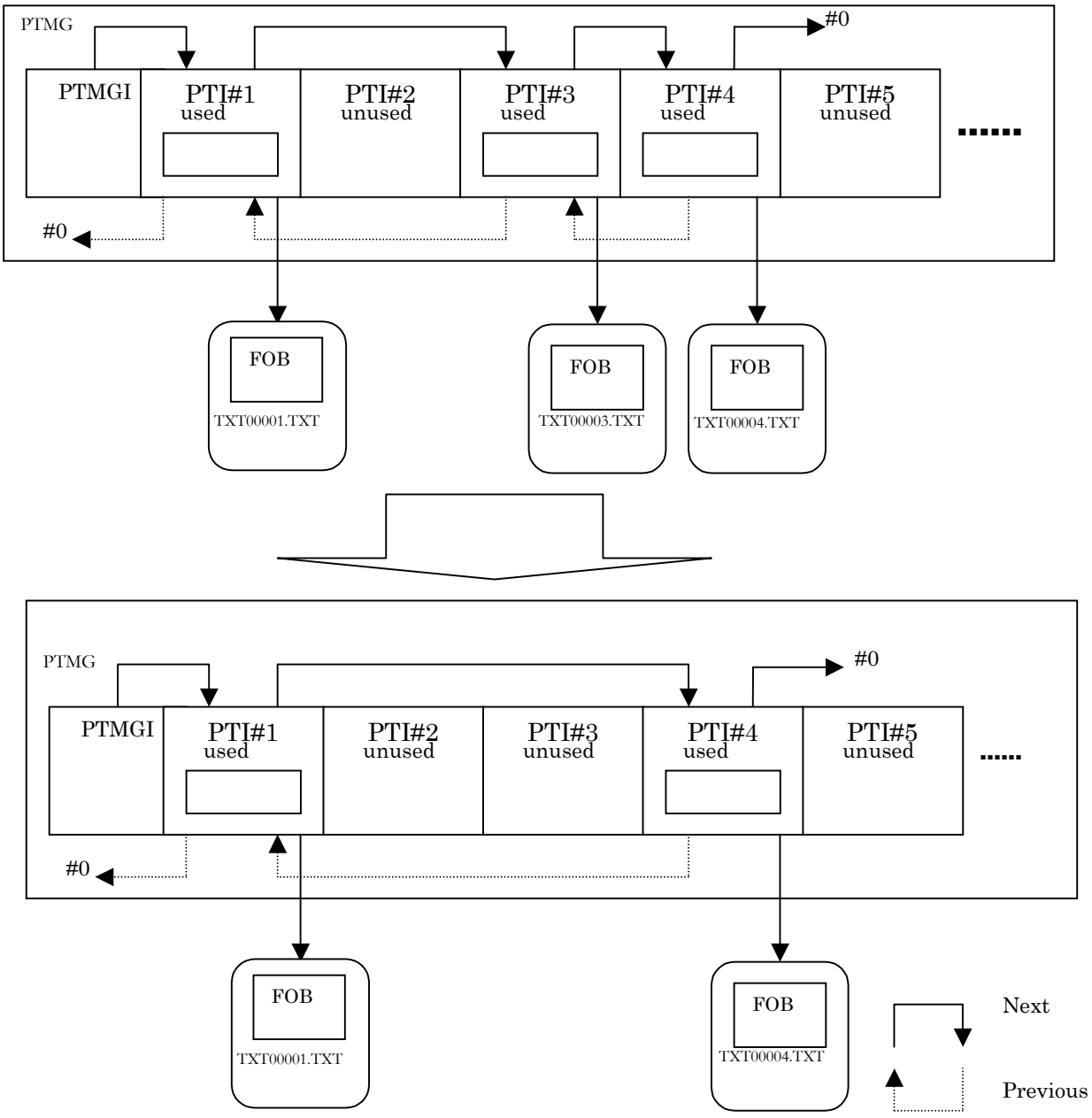
Exclusive access control to PTMGI is necessary from procedure 1 to 5.
Exclusive access control to PTI#2 is necessary in procedure 7.

© 2000 SD Association

CONFIDENTIAL

3.3.2.2 Deletion

◊ **Figure 3.3-2 Deleting a file**



Delete TXT00003.TXT

1. Change PTI chain list structure for deleting PTI#3 (modify PTI#1 and #4)
2. Renew PTI#3 "unused"
3. Renew PTMGI

Exclusive access control to PTMGI is necessary from procedure 1 to 3.

© 2000 SD Association

CONFIDENTIAL

3.3.2.3 Order modification

Order modification is achieved by changing the list structure of PTIs. (If order of the first PTI is modified, PTMG_PTI_FUN[see section 4.1.2] in PTMGI must be renewed.)

But the order modification is prohibited, so it shall be guaranteed that the order of PTIs is sorted by addition of PTI.

4. Navigation Data Structure

4.1 Navigation Data Structure for Plain Text Data

Plain Text Manager (PTMG) is a navigation data for Plain Text data.

4.1.1Plain Text Manager (PTMG)

PTMG describes information about Plain Text Files within TEXT directory.
The PTMG starts with Plain Text Manager Information (PTMGI), followed by set of Plain Text Information (PTI), as shown in [Figure 4.1-1].

◊ **Figure 4.1-1 Plain Text Manager(PTMG)**

Plain Text Manager (PTMG)

Plain Text Manager Information (PTMGI) (Mandatory)	Fixed Size(8B)
Plain Text Information #1 (PTI #1) (Optional)	Fixed Size(512B)
Plain Text Information #2 (PTI #2) (Optional)	Fixed Size(512B)
:	:
Plain Text Information #n (PTI #n) (Optional)	Fixed Size(512B)

(n ≤ 65,535)

4.1.2 Plain Text Manager Information(PTMGI)

As shown in [Table 4-1], Number of Plain Text Information, Number of used Plain Text Information, and First "used" PTI number are described in Plain Text Manager Information (PTMGI).

◊ **Table 4-1 PTMGI** (Description order)

RBP		Contents	Number of bytes
0 to 1	PTMG_PTI_Ns	Number of PTIs exist	2 bytes
2 to 3	PTMG_PTI_NUSED	Number of used PTIs	2 bytes
4 to 5	PTMG_PTI_FUN	First "used" PTI number	2 bytes
6 to 7	Reserved	Reserved	2 bytes
Total			8 bytes

(RBP 0 to 1) PTMG_PTI_Ns
Describes number of PTIs exist in PTMG.

(RBP 2 to 3) PTMG_PTI_NUSED
Describes number of used PTIs in PTMG.

© 2000 SD Association

CONFIDENTIAL

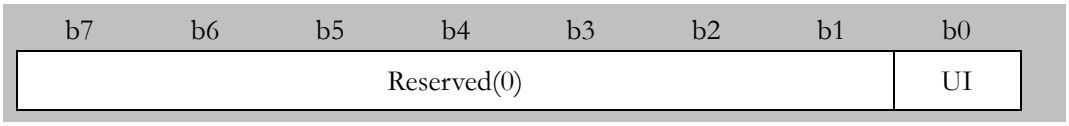
(RBP 4 to 5) PTMG_PTI_FUN
[Table 4-2] Describes the entry number of the first "used" Plain Text Information.
If 0, there are NO "used" PTIs in PTMG.

4.1.3 Plain Text Information (PTI)

◊ **Table 4-2 PTI** (Description order)

RBP		Contents	Number of bytes
0	UI	Used Information	1 byte
1	Reserved	Enter 00h	1 byte
2	Reserved	Enter 00h	1 byte
3	VERN	SD-pDocument Version No.	1 byte
4	EXTENSION	Reserved for Extension	1 byte
5	SRC	Data Source	1 byte
6 to 7	FL_SRN	File Search Number	2 bytes
8 to 9	CHAR_CODE1	Character Set Code 1	2 bytes
10 to 17	LANG1	Language 1	8 bytes
18 to 19	CHAR_CODE2	Character Set Code 2	2 bytes
20 to 27	LANG2	Language 2	8 bytes
28 to 155	COMMENT	Comment	128 bytes
156 to 283	ORG_FNAME	Original Filename	128 bytes
284 to 315	ORG_DATETIME	Original Date and Time	32 bytes
316 to 317	P_PTI_SRN	Previous PTI Search Number	2 bytes
318 to 319	N_PTI_SRN	Next PTI Search Number	2 bytes
320 to 511	Reserved	Enter 00h	192 bytes
Total			512 bytes

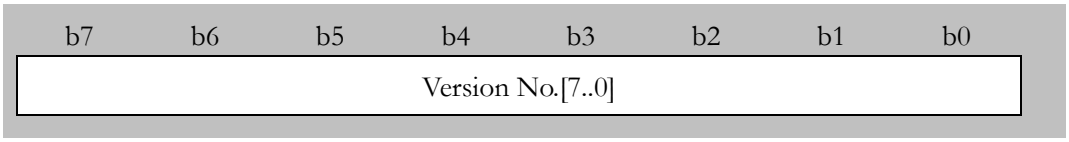
(RPB 0) UI
Describes Used Information.



Used Information ... When the PTI is not in use, enter 0.
When the PTI is in use, enter 1.

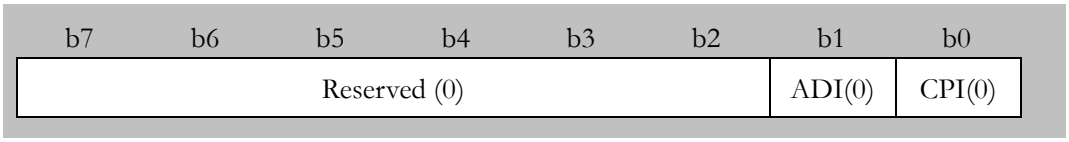
CONFIDENTIAL

(RPB3) VERN
Describes version No. of SD-pDocument specification supported by the writer.



Version number ... 09h : version 0.9
10h : version 1.0
Others : reserved

(RPB 4) EXTENSION
Describes extension function field for future specifications of SD-pDocument.
In this version, ALL bit fields MUST BE 0.



CPI ... Describes if the Plain Text data is guarded by CPRM
0 : not guarded
1 : guarded

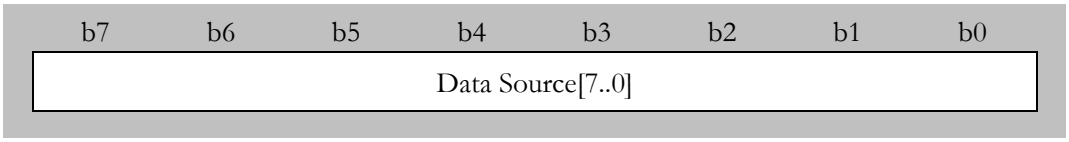
ADI ... Describes if there is additional information of the PTI
0 : there is no additional information.
1 : additional information presents.

If CPI was set, an appliance (reader) that support this specification can NOT manage the Plain Text data.
If ADI was set, an appliance (reader) that support this specification will be able to manage the Plain Text data without the additional information.

© 2000 SD Association

CONFIDENTIAL

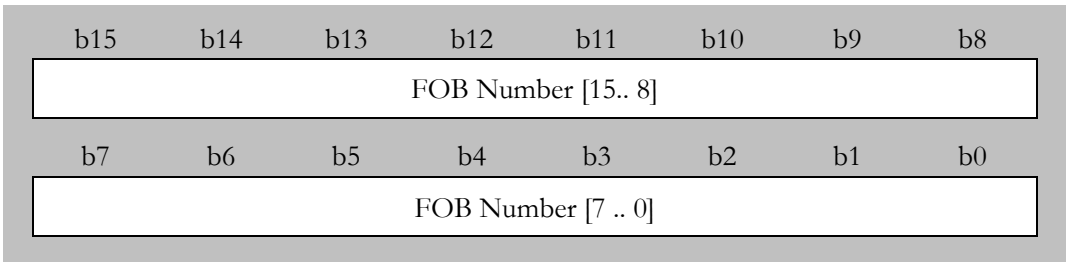
(RPB 5) SRC
Describes what is an original data of the Plain Text.



Data Source ... 00h : unknown
01h : e-mail
02h : personal information (ex. address book)
03h : distributed content

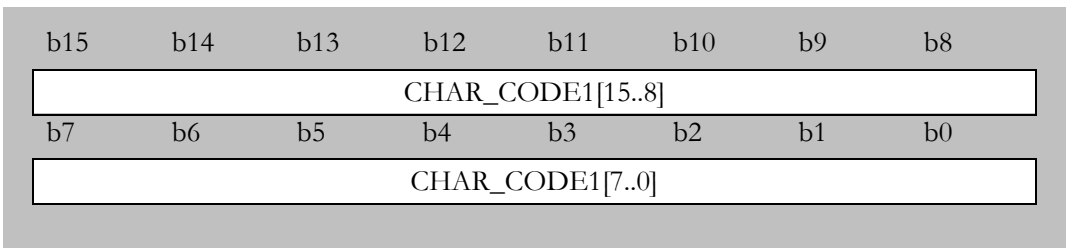
Others : reserved

(RBP 6 to 7) FL_SRN
Describes the entry number of referenced FOB.
FL_SRN must be equal to :
((absolute byte position of this PTI) – (byte size of PTMGI))/ (byte size of PTI) + 1.



FOB Number ... Describes the number between '1' and '65,535'.

(RBP 8 to 9) CHAR_CODE1
Describes character set code of the Plain Text file referenced by this PTI..



Character Set code 1 … IANA Character Sets
See ANNEX B

© 2000 SD Association

CONFIDENTIAL

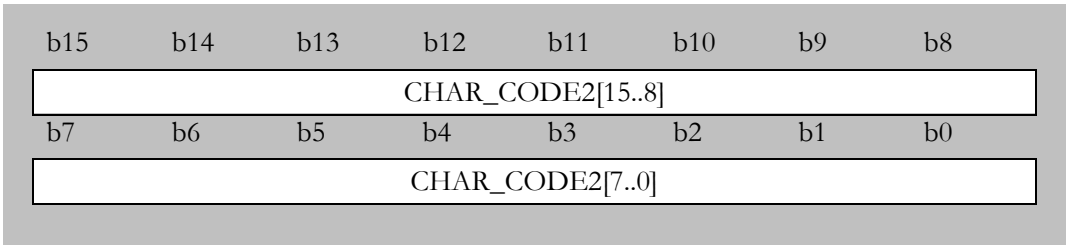
(RBP 10 to 17) LANG1

Describes a language used in the Plain Text file referenced by this PTI.
This field must be terminated by NULL character.

LANG1 … Language TAG (ex. "en-us")
See ANNEX C

(RBP 18 to 19) CHAR_CODE2

Describes character set code of the character field of this PTI



Character Set code 2 … IANA Character Sets
See ANNEX B

(RBP 20 to 27) LANG2

Describes a language used in of the character field of this PTI..
This field must be terminated by NULL character.

LANG2 … Language TAG (ex. "en-us")
See ANNEX C

(RBP 28 to 155) COMMENT

Describes comment for the plain text data referenced by this PTI.

This field must be terminated by NULL character.

*NOTE: A NULL character must be included in each character fields in this specification. If a field length is N bytes, actual maximum character length is (N – size of a NULL character) bytes.

(RBP 156 to 283) ORG_FNAME

Describes the original file name referenced to by this PTI before conversion to TXTxxxxx.TXT.

This field must be terminated by NULL character.

(RBP 284 to 315) ORG_DATETIME

Describes the date and time of the original file referenced to by this PTI before conversion to TXTxxxxx.TXT.

This field must be terminated by NULL character.

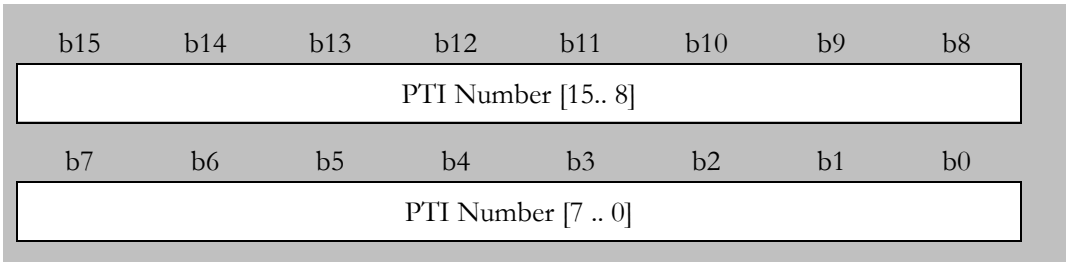
Format of date and time is based on ISO8601

yyyy-mm-ddThh:mm:ss+hh:ss (ex. "1998-09-15T21:15:00+09:00")

© 2000 SD Association

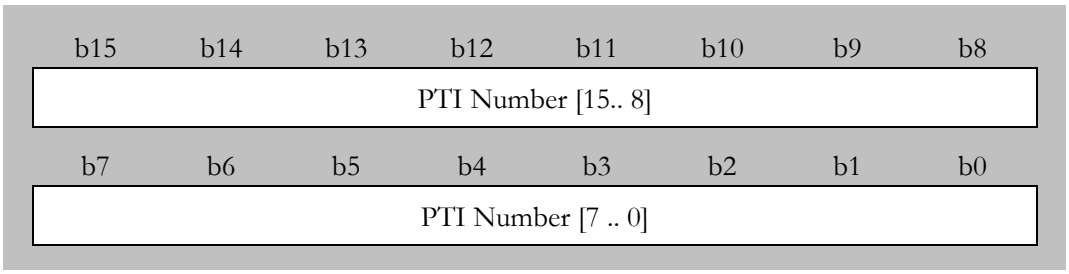
CONFIDENTIAL

(RBP 316 to 317) P_PTI_SRN
Describes the entry number of previous PTI.



Previous PTI No. ... Describes the number between '1' and '65,535'.
When no PTI is referenced, enter 0.

(RBP 318 to 319) N_PTI_SRN
Describes the entry number of next PTI.



Next PTI No. ... Describes the number between '1' and '65,535'.
When no PTI is referenced, enter 0.

© 2000 SD Association

CONFIDENTIAL

4.2 Navigation Data Structure for Raster Image Data

Raster Image Manager (RIMG) is a navigation Data for Raster Image data.

4.2.1 Raster Image Manager (RIMG)

RIMG describes information about multi page Raster Image Files within RASTER directory.
The RIMG starts with Raster Image Manager Information (RIMGI), followed by set of Raster Image Information (RII), as shown in [Figure 4.2-1].

◊ **Figure 4.2-1 Raster Image Manager(RIMG)**

Raster Image Manager (RIMG)

Raster Image Manager Information (RIMGI) (Mandatory)	Fixed Size(8B)
RasterImage Information #1 (RII #1) (Optional)	Fixed Size(512B)
Raster Image Information #2 (RII #2) (Optional)	Fixed Size(512B)
:	:
Raster ImageInformation #n (RII #n) (Optional)	Fixed Size(512B)

(n ≤ 65,535)

4.2.2 Raster Image Manager Information(RIMGI)

As shown in [Table 4-3], Number of Raster Image Information, Number of used Raster Image Information, and First "used" RII number are described in Raster Image Manager Information RIMGI).

◊ **Table 4-3 RIMGI** (Description order)

RBP		Contents	Number of bytes
0 to 1	RIMG_RII_Ns	Number of RIIs exist	2 bytes
2 to 3	RIMG_RII_NUSED	Number of used RIIs	2 bytes
4 to 5	RIMG_RII_FUN	First "used" RII number	2 bytes
6 to 7	Reserved	Reserved	2 bytes
Total			8 bytes

(RBP 0 to 1) RIMG_RII_Ns
Describes number of RIIs exist in RIMG.

(RBP 2 to 3) RIMG_RII_NUSED
Describes number of used RIIs in RIMG.

(RBP 4 to 5) RIMG_RII_FUN
[Table 4-2] Describes the entry number of the first "used" Raster Image Information.
If 0, there are NO "used" RIIs in RIMG.

© 2000 SD Association

CONFIDENTIAL

4.2.3 Raster Image Information (RII)

◊ **Table 4-4 RII** (Description order)

RBP		Contents	Number of bytes
0	UI	Used Information	1 byte
1	Reserved	Enter 00h	1 byte
2	Reserved	Enter 00h	1 byte
3	VERN	SD-pDocument Version No.	1 byte
4	EXTENSION	Reserved for Extension	1 byte
5	SRC	Data Source	1 byte
6 to 7	FL_SRN	File Search Number	2 bytes
8 to 9	CHAR_CODE1	Character Set Code 1	2 bytes
10 to 17	LANG1	Language 1	8 bytes
18 to 19	CHAR_CODE2	Character Set Code 2	2 bytes
20 to 27	LANG2	Language 2	8 bytes
28 to 155	COMMENT	Comment	128 bytes
156 to 283	ORG_FNAME	Original Filename	128 bytes
284 to 315	ORG_DATETIME	Original Date and Time	32 bytes
316 to 317	CODING	Coding Method of Raster Image	2 bytes
318	COLOR	Color Space of Raster Image	1 byte
319	reserved		1 byte
320 to 321	TPAGE	Total Pages	2 bytes
322 to 323	P_RII_SRN	Previous RII Search Number	2 bytes
324 to 325	N_RII_SRN	Next RII Search Number	2 bytes
326 to 511	Reserved	Enter 00h	186 bytes
Total			512 bytes

© 2000 SD Association

CONFIDENTIAL

(RPB 0) UI
Describes Used Information.

b7	b6	b5	b4	b3	b2	b1	b0
Reserved(0)							UI

Used Information ... When the RII is not in use, enter 0.
When the RII is in use, enter 1.

(RPB 3) VERN
Describes version No. of SD-pDocument specification supported by the writer.

b7	b6	b5	b4	b3	b2	b1	b0
Version No.[7..0]							

Version number ... 09h : version 0.9
10h : version 1.0
Others : reserved

(RPB 4) EXTENSION
Describes extension function field for future specifications of SD-pDocument.
In this version, ALL bit fields MUST BE 0.

b7	b6	b5	b4	b3	b2	b1	b0
Reserved (0)						ADI(0)	CPI(0)

CPI ... Describes if the Raster Image data is guarded by CPRM
0 : not guarded
1 : guarded

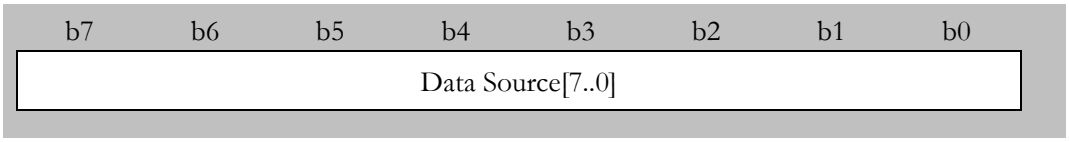
ADI ... Describes if there is additional information of the PTI
0 : there is no additional information.
1 : additional information presents.

If CPI was set, an appliance (reader) that support this specification can NOT manage the Raster Image data.
If ADI was set, an appliance (reader) that support this specification will be able to manage the Raster Image data without the additional information.

© 2000 SD Association

CONFIDENTIAL

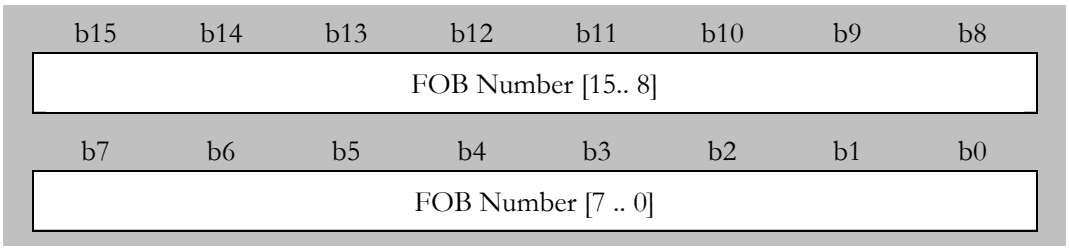
(RPB 5) SRC
Describes what is an original data of the Raster Image data.



Data Source ... 00h : unknown
01h : e-mail
02h : data generated by Computer
03h : data from KIOSK terminal
05h : FAX data
06h : data scanned by Image Scanner

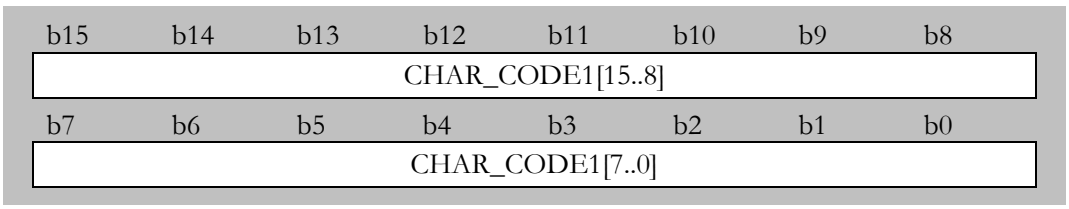
Others : reserved

(RBP 6 to 7) FL_SRN
Describes the entry number of referenced FOB.
FL_SRN must be equal to :
((absolute byte position of this RII) – (byte size of RIMGI))/ (byte size of RII) + 1.



FOB Number ... Describes the number between '1' and '65,535'.

(RBP 8 to 9) CHAR_CODE1
Describes character set code of characters in character fields in TAG of TIFF file referenced by this RII.



Character Set code 1 … IANA Character Sets
See ANNEX B

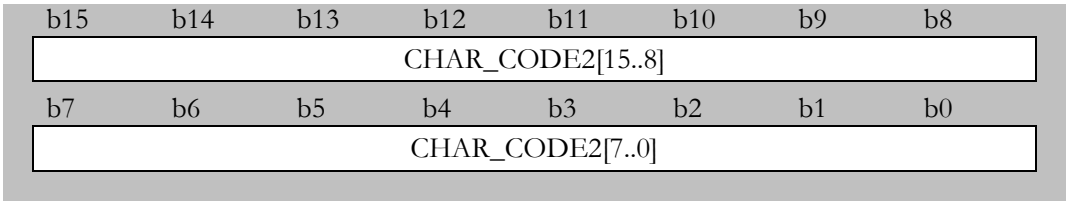
© 2000 SD Association

CONFIDENTIAL

(RBP 10 to 17) LANG1
Describes a language used in characters in character fields in TAG of TIFF file referenced by this RII.
This field must be terminated by NULL character.

LANG1 … Language TAG (ex. “en-us”)
See ANNEX C

(RBP 18 to 19) CHAR_CODE2
Describes character set code of the character field of this RII.



Character Set code 2 … IANA Character Sets
See ANNEX B

(RBP 20 to 27) LANG2
Describes a language used in the character fields of this RII.
This field must be terminated by NULL character.

LANG2 … Language TAG (ex. “en-us”)
See ANNEX C

© 2000 SD Association

CONFIDENTIAL

(RBP 28 to 155) COMMENT

Describes comment for the plain text data referenced by this RII.

(ex. FAX sender information)

This field must be terminated by NULL character.

Each item starts with Tag, followed by text data.

It is not necessary to describe all Tags, and only Tags which have text data are needed to be described with their text data.

Tags are shown in [Table 4-5]. The description order of these Tags is not required to accord with this table.

Each text in 20 items have variable size, however, the total size included Tags and a termination code is not to exceed 128 bytes.

◊ **Table 4-5 Item's Tags in Comment Field of RII**

Tag Name	Tag Value		Contents
	ISO646 JISX0201 IOS8859 (1 byte)	Shift-JIS (2 bytes)	
SENDER _INF	01h	0001h	Information of a sender who send the Raster Image
RECEIVER_INF	02h	0002h	Information of a receiver who send the Raster Image
START_TIME	03h	0003h	Date and Time when the data communication started
USAGE_TIME	04h	0004h	Time period of the data communication
RESULT	05h	0005h	Result of the data communication
PAGES	06h	0006h	Total pages of Raster Image in the data communication
USER_NAME	07h	0007h	User name
APPL_SOFT	08h	0008h	Application software name which made original data of the Raster Image
WRITER_INFO	09h	0009h	Writer information which wrote the Raster Image data
URL	0ah	000ah	URL
TITLE	0bh	000bh	TITLE of the Raster Image
FREE1	0ch	000ch	Free 1
FREE2	0dh	000dh	Free 2
FREE3	0eh	000eh	Free 3
FREE4	0fh	000fh	Free 4
FREE5	10h	0010h	Free 5
FREE6	11h	0011h	Free 6
FREE7	12h	0012h	Free 7
FREE8	13h	0013h	Free 8
FREE9	14h	0014h	Free 9

© 2000 SD Association

CONFIDENTIAL

<Terminated code>
US-ASCII(ISO646), ISO8859, JIS X 0201 : '0x00'
Shift-JIS : '0x0000'

◊ **Table 4-6 Example of contents stated with Tags in Comment field**

SENDER _INF	Phone number of a FAX machine which sent the Rater Image.
RECEIVER_INF	Phone number of a FAX machine which will receive the Rater Image.
START_TIME	Date and Time when FAX communication stated
USAGE_TIME	Time period of FAX communication
RESULT	Result of FAX communication
PAGES	Total pages of Raster Image in FAX communication
USER_NAME	User name in PC, when the writer was PC
APP_SOFT	Word processing software name which made original data of the Raster Image
WRITER_INFO	Name of FAX machine, when writer was FAX machine
URL	URL, when the Raster Image was downloaded from the URL
TITLE	A title name of the Raster Image which was scanned by a image scanner

(RBP 156 to 283) ORG_FNAME

Describes the original file name referenced to by this RII before conversion to TIFxxxxx.TIF.
This field must be terminated by NULL character.

(RBP 284 to 315) ORG_DATETIME

Describes the date and time of the original file referenced to by this PTI before conversion to TIFxxxxx.TIF.
This field must be terminated by NULL character.
Format of date and time is based on ISO8601
yyyy-mm-ddThh:mm:ss+hh:ss (ex. "1998-09-15T21:15:00+09:00")

© 2000 SD Association

CONFIDENTIAL

(RBP 316 to 317) CODING

Describes coding methods of pages of the Raster Image.
A plurality of coding methods can be used because coding method might be different in each page.

b15	b14	b13	b12	b11	b10	b9	b8
Reserved(0)	Reserved(0)	Reserved(0)	Reserved(0)	Reserved(0)	Reserved(0)	Reserved(0)	JPEG
Color or Gray scale image							

b7	b6	b5	b4	b3	b2	b1	b0
Reserved(0)	Reserved(0)	Reserved(0)	JBIG	MMR	MR	MH	RAW
Monochrome Binary image							

JPEG … 0 : There are no JPEG coded pages in the Raster Image.
1 : JPEG coding is used in the Raster Image.

JBIG … 0 : There are no JBIG coded pages in the Raster Image.
1 : JBIG coding is used in the Raster Image.

MMR … 0 : There are no MMR coded pages in the Raster Image.
1 : MMR coding is used in the Raster Image.

MR 0 : There are no MR coded pages in the Raster Image.
1 : MR coding is used in the Raster Image.

MH … 0 : There are no MH coded pages in the Raster Image.
1 : MH coding is used in the Raster Image.

RAW … 0 : There are no RAW image pages in the Raster Image.
1 : RAW image page is in the Raster Image.

If all bits are 0, that indicates the coding methods are unknown.

(RBP 318) COLOR

Describes color space of pages of the Raster Image.
A plurality of color space can be used because color space might be different in each page.

b7	b6	b5	b4	b3	b2	b1	b0
Reserved(0)	Reserved(0)	Reserved(0)	Reserved(0)	Reserved(0)	Gray	La*b*	YCbCr
Color or Gray scale image							

Color space : YCbCr … 0 : YCbCr is not used in all pages.
1 : YCbCr is used in one or more pages.

Color space : La*b* … 0 : La*b* is not used in all pages.
1 : La*b* is used in one or more pages.

If all bits are 0, that indicates the color spaces are unknown or all pages are monochrome binary image.

© 2000 SD Association

CONFIDENTIAL

(RBP 320 to 321) TPAGE
Describes total number of pages of the TIFF file referenced by this RII.

b15	b14	b13	b12	b11	b10	b9	b8
Total Number of pages[15.. 8]							
b7	b6	b5	b4	b3	b2	b1	b0
Total Number of pages[7 .. 0]							

Total number of pages ... Describes the number between '1' and '999'.
When the total number is unknown, enter 0.

(RBP 322 to 323) P_RII_SRN
Describes the entry number of previous RII.

b15	b14	b13	b12	b11	b10	b9	b8
RII Number [15.. 8]							
b7	b6	b5	b4	b3	b2	b1	b0
RII Number [7 .. 0]							

Previous RII No. ... Describes the number between '1' and '65,535'.
When no RII is referenced, enter 0.

(RBP 324 to 325) N_RII_SRN
Describes the entry number of next RII.

b15	b14	b13	b12	b11	b10	b9	b8
RII Number [15.. 8]							
b7	b6	b5	b4	b3	b2	b1	b0
RII Number [7 .. 0]							

Next RII No. ... Describes the number between '1' and '65,535'.
When no RII is referenced, enter 0.

© 2000 SD Association

CONFIDENTIAL

5. Raster Image Data Structure

This section describes the tag fields required or recommended by this specification. The pattern for the description of TIFF fields in this specification is:

FieldName(TagValueInDecimal) = allowable values **TYPE**
Count = (omitted if =1)
Explanation of this field.

The following sub sections describe monochrome (black and white binary) image and color (or gray scale) image in each sub section. Multi page TIFF file can include monochrome binary pages, gray scale pages and color pages.

© 2000 SD Association

CONFIDENTIAL

5.1 Monochrome Binary Image

This section prescribes TIFF fields and their values necessary for handling especially black-and-white binary images by TIFF.

5.1.1 Required TIFF Fields

This section lists mandatory TIFF fields in pages of the monochrome binary image to be supported by Rater Image Data. When there are several values for one field, the Writer may select one value but the Reader must support all the field values.

ImageWidth(256) **SHORT or LONG**
ImageLength(257) **SHORT or LONG**

Number of pixels (columns) per scanline (line) and total number of scanlines in a page
The maximum number is not prescribed. However, it is preferable to set the total number of scanlines by the resolution specified with the X-Resolution and Y-Resolution fields according to the output paper size (standardized by JIS or ISO) for the purpose of print output or number of pixels of display devices (VGA, SVGA, XGA, etc.) for display output.
These fields shall always be specified because no default is set.
It is recommended that pixels in 4mm area of edges of print paper are white data, because many laser printers can not print on edge area of print paper.
If size of an image is smaller than paper size, It is recommended that the image is placed on center of the paper in ImageWidth direction and on top of paper in ImageLength direction.
If size of an image is larger than paper size, It is desired that all image area is printed by resolution conversion.
If it is possible to print normally by rotation, rotation processing of an image is desired.
If a device does not support resolution conversion or rotation processing function and can not print normally, it is desired that the device notify the point to a user of the device.
These Tag values do not exceed 65,535.

StripOffsets(273) **SHORT or LONG**

Count = 1
Byte offset of a strip from top of the TIFF file to the strip
This field shall always be specified because no default is set.
Number of strips must be 1. (Single strip : COUNT must be 1)

PageNumber(297) **SHORT**

Count = 2
Field 1 indicates a page number (0 for the first page) and field 2 indicates the total number of pages in the document. If the value of field 2 is 0, the total number of pages is unknown.
This field shall always be specified because no default is set.
Total number of pages is set in filed 2 of the first page. It is not necessary that total number of pages is set in filed 2 of PageNumber of all the pages.
If total number of pages is not set and Reader wants to know the number, the reader can know it by counting pages.
It is recommended that Writer set total number of pages in field 2 of the first page.

RowsPerStrip(278) **SHORT or LONG**

Number of scanlines for each TIFF strip, excluding the last one
For a single-strip image, this is the same as the value of the ImageLength field. A single-strip image shall be use.
Default = $2^{32} - 1$ or $2^{16} - 1$ (A single strip contains all scanlines.)
The default permits a single-strip image but the value of the ImageLength field should be set.

© 2000 SD Association

CONFIDENTIAL

StripByteCounts(279) **SHORT or LONG**

Count = 1
Number of bytes indicating the compressed volume of image data contained in each strip
Count must be 1 because of single strip.
This field shall always be specified because no default is set.

BitsPerSample(258) = 1 **SHORT**

Number of bits per image sample (binary data only)
This field shall be set 1 because of monochrome binary image.

Compression(259) = 1, 3, 4, (9) **SHORT**

Compression method
1 = No compression (Raw)
3 = One-dimensional coding (MH) or two-dimensional coding (MR)
4= two-dimensional coding (MMR)
Raw and MH are mandatory. MR and MMR are recommended strongly.
The Writer shall support at least one of these.
The Reader shall support Raw and MH (+ MR, MMR).
When the Compression method is MH, the **T4Options** field shall be 0 or 4.
(T4Options : bit0:0=1 dimension coding bit2:1 = byte align EOLs)
When the Compression method is MR, the **T4Options** field shall be 1 or 5.
When the Compression method is MMR, the **T6Options** field shall be 0.
JBIG(9) is an option.

FillOrder(266) = 1, 2 **SHORT**

1 = MSB First
2 = LSB First
Bit order in byte
When using MH, MR and MMR compression (Compression = 3 or 4), the Writer can select LSB First (Field value = 2).
Default = 1 (When using the default, this field may be omitted. If the FillOrder field is not detected, the Reader shall assume 1 in this field.)

NewSubFileType(254) = (Bit 1=1) **LONG**

General instruction for the type of data contained in this image file directory (IFD)
If an image is a page of a multi page document, bit 1 is 1.
Default = 0 (This value cannot be ignored because value 2 must be specified.)

PhotometricInterpretation(262) = 0 **SHORT**

Field indicating the color space of image data
0 = Pixel value 1 means black.
This field must be 0.
This field shall always be specified because no default is set.
The standard for image printing is adopted that Pixel value 0 means white. For display, pixel value 0 means black (PhotometricInterpretation = 1) but this is not supported because data conversion to invert data is easy.

ResolutionUnit(296) = 2 **SHORT**

Unit of resolution (2 = inch)

Default = 2 (This field may be omitted. If the ResolutionUnit field is not detected, the Reader shall

© 2000 SD Association

CONFIDENTIAL

assume 2 in this field.)

SamplesPerPixel(277) = 1 **SHORT**

Number of components per pixel (1 for black-and-white)
This field must be 1.

XResolution(282) = 200,300 **RATIONAL**
YResolution(283) = 200,300 **RATIONAL**

The horizontal and vertical resolutions of an image are prescribed in pixels/inch.
For the Raster Image Data to support a square for the pixel shape, the horizontal and vertical resolutions shall be of the same value.
In consideration of implementation into established FAX machines, 200x200dpi is mandatory. For image quality, it is recommended strongly that Reader supports 300x300dpi if the Reader is designed newly.
Resolution other than them or aspect ration other than 1 are options. In case of these option, The reproduced image will be distorted if Reader does not support resolution conversion function.
It is allowable that the FAX resolutions (204x196 and 408x391) are processed as 200x200 or 400x400dpi.

T4Options(292) = (Bit 0 = 0 or 1, Bit 1 = 0, Bit 2 = 0, 1) **LONG**

This field shall be specified when the Compression field value is 3.
Set Bit 0 to 0 to specify MH compression. Set Bit 0 to 1 to specify MR compression.
Bit 1 shall be 0.
Bit 2 = 1 indicates that the end-of-line (EOL) code is byte-aligned and Bit 2 = 0 indicates that it is not.
The default applies when all bits are 0 (EOL code is not byte-aligned).

T6Options(293) = 0 **LONG**

Bit 0: unused
Bit 1: 1:allowed raw mode in code data 0:not allowed
This field shall be specified when the Compression field value is 4.
All bits except bit 1 must be set to 0.

5.1.2 Recommended TIFF Fields

This section lists recommended TIFF fields that may be used. The Reader may not interpret these fields but must not assume an error.

DateTime(306) **ASCII**

Image creation date and time in 24-hour format
This field shall be terminated by NULL character.
No default
Format yyyy-mm-ddThh:mm:ss+hh:ss(ex. 1998-09-15T21:15:00+09:00)[ISO8601] shall be used by Writer.
For lower compatibility, yyyy:mm:dd hh:mm:ss shall be supported by Reader.

DocumentName(269) **ASCII**

Name of a scanned document. This is not a baseline TIFF field but a TIFF extended field.
No default
This field terminated by NULL character.

ImageDescription(270) **ASCII**

String indicating the contents of an image

© 2000 SD Association

CONFIDENTIAL

No default
This field terminated by NULL character.

Orientation(274) = 1 **SHORT**

1: Line 0 indicates the apparent beginning of the image and column 0 indicates the apparent left side of the image. The origin is at the top left of the image. Like TIFF6.0 baseline, only the value of 1 is mandatory.
Default = 1 (Even for the default, it is recommended to include this field.)
The Writer may omit the Orientation field but the Reader must set 1.
This specifications does not prescribe whether the machine rotates, clips, or reduces an image due to a difference of the origin.

Software(305) **ASCII**

Name and release number of a software package used to create image data
This field terminated by NULL character.
No default

5.1.3 Other TIFF Fields

It is permitted to add other TIFF fields which do not contradict the fields described above.
But, Reader may ignore the other TIFF fields.

5.1.4 File Structure

In consideration of network system, there is the advantage that prescribes data arrangement as follows. It is mandatory that Writer makes the following data arrangement, but it is desired that Reader supports the free data arrangement also.
IFD and image data shall be arranged appropriately in a file according to the following requirements:

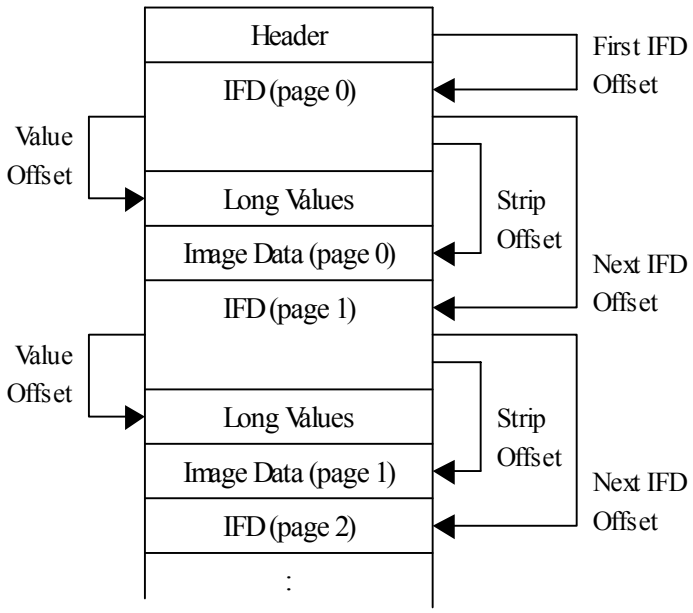
(1) IFD shall exist for each page of a multi page document.
(2) IFD shall be generated in the file in the same order as page generation in the document.
(3) IFD shall precede image data with offset.
(4) Image data shall be generated in the file in the same order as page generation in the document.
(5) IFD, value data, and image data with offset shall precede the next image IFD.
(6) One strip shall contain image data for each page.

If the requirement of (6) is satisfied, the StripOffsets field will include a pointer to image data. The field entry by IFD has two exceptions and contains an offset to the field value placed outside the IFD. The two exceptions are values related to the X-Resolution and Y-Resolution fields. Both fields are of the RATIONAL type and require two 4-byte numbers. These long field values shall be placed immediately before the IFD including their offsets and before the image data indicated by the IFD.
Because of these requirements, the IFD to the page shall come first in the file after the TIFF header, then the long values of the X-Resolution and Y-Resolution fields. Then the image data of page 1 comes next, followed by the IFD of page 2. See the next page for this sequence. The IFD value is 1 for page 1, 2 for page 2, and so on. The PageNumber field is required.

© 2000 SD Association

CONFIDENTIAL

◊ **Figure 5.1-1 TIFF file structure**



The above file structure may reduce memory requests when implemented. This structure also supports a stream of processing when a file is received but not totally yet.

The first two bytes of an 8-byte image file header indicate a byte sequence in the header. The byte sequence of little endian (0x949) or big endian (0x4D4D) can be used for a 16-bit or 32-bit integer. The next two bytes contain the value "42" in the sequence conforming to the first two bytes of the header to identify the file as a TIFF one. The last four bytes give an offset indicating the first IFD. All offsets in a TIFF file are related to the beginning of the file. An IFD may be placed at any file position after the header but must start with a word (2-byte) boundary.

© 2000 SD Association

CONFIDENTIAL

5.2 Color Image

This section prescribes TIFF fields and their values necessary for handling color (multivalued) images, including black-and-white gray scale images.

5.2.1 Required TIFF Fields

This section lists mandatory TIFF fields to be supported.

ImageWidth(256) ImageLength(257) **SHORT or LONG**
Same as Monochrome binary image specification

StripOffsets(273) **SHORT or LONG**
Same as Monochrome binary image specification

PageNumber(297) **SHORT**
Same as Monochrome binary image specification

RowsPerStrip(278) **SHORT or LONG**
Same as Monochrome binary image specification

StripByteCounts(279) **SHORT or LONG**
Same as Monochrome binary image specification

NewSubFileType(254) = (Bit 1=1) **LONG**
Same as Monochrome binary image specification

BitsPerSample(258) = 8 **SHORT**
Count = SamplesPerPixel
8 bits/sample shall be set.

Compression(259) = 7 **SHORT**
Baseline JPEG compression (7) shall be used.
JPEG stream must be based on ISO10918-1.

FillOrder(266) = 1, 2 **SHORT**
1 = MSB First
2 = LSB First
1 or 2 shall be set. 1 is desired.

PhotometricInterpretation(262) = 1, 2, 6, (10) **SHORT**
0: min value is white (gray scale image)
1: min value is black (gray scale image)
6: YCbCr (color image)
10: ITULab (color image)
If image data are gray scale images (BitsPerSample = 8 and SamplesPerPixel = 1) , PhotometricInterpretation must be 0 or 1.
If PhotometricInterpretation is 1 , the gray scale space will be equal to Y of YCrCb which is based on sRGB color space. If PhotometricInterpretation is 0, it will be inverted Y of YCrCb based on sRGB.
A device-independent color space is preferable for exchanging color images. So YCbCr is obtained by conversion based on sRGB.
YCbCr is mandatory and ITULab is an option.
When the PhotometricInterpretation value is set to 10, Color space is ITULAB by CIE L*a*b* encoding

© 2000 SD Association

CONFIDENTIAL

defined in ITU-T Recommendation T.42. This encoding maps the minimum sample value to 0 and the maximum sample value to 2^n-1, i.e., the maximum value of BitsPerSample value n. The Decode field determines conversion from unsigned ITULAB coding sample values to signed CIE L*, a*, and b* values. Conversion from signed ITULAb coded sample values to signed CIE L*a*b* values is defined by DECODE field.

ResolutionUnit(296) = 2 **SHORT**

Same as Monochrome binary image specification

SamplesPerPixel(277) = 1, 3 **SHORT**

Number of samples per pixel
1: Y or L* component only (Gray scale image)
3: YCbCr or L*a*b* components
This value is coded according to the PhotometricInterpretation field.

XResolution(282) = 200,300 **RATIONAL**
YResolution(283) = 200,300 **RATIONAL**

Same as Monochrome binary image specification

The JPEG compression standard permits sub-sampling where the Y brightness (L* lightness) component of an image is compared with the Cb and Cr (a* and b*) chroma components. The two extended fields (ChromaSubSampling and ChromaPositioning) explained below define sub-sampling. The fields have the same functions as YCbCrSubSampling and YCbCrPositioning of [TIFF] but were renamed to reflect their applications to other color spaces.

ChromaSubSampling(530) **SHORT**

Count = 2
When the PhotometricInterpretation value is 6 or 10, this field specifies the sub-sampling factor related to the chroma components of an image. The two subfields (ChromaSubsampleHoriz and ChromaSubsampleVert) of this field specify the horizontal and vertical sub-sampling factors.

SHORT 0: ChromaSubsampleHoriz = 1, 2
1: Lightness sample equal to the chroma sample in the horizontal direction
2: Lightness sample double the chroma sample in the horizontal direction

SHORT 1: ChromaSubsampleVert = 1, 2
1: Lightness sample equal to the chroma sample in the vertical direction
2: Lightness sample double the chroma sample in the vertical direction

The default of ChromaSubSampling is (2,2) . This field can be omitted if default value is used.

ChromaPositioning(531) = 1 **SHORT**

When PhotometricInterpretation = 6 or 10, this field specifies spatial positioning of the chroma component compared with the lightness component.
1: centered
2: cosited
If Reader does not support 2 (cosited), It is permitted that Reader manage it as 1 (centered).
If this field was omitted, Reader manage this field as 1.

Decode(433) **RATIONAL**

© 2000 SD Association

CONFIDENTIAL

Count = 2 * SamplesPerPixel
Describes how to map image sample values into the range of values appropriate for the current color space. In general, the values are taken in pairs and specify the minimum and maximum output value for each color component. For the base color fax profile, Decode has a count of 6 values.

If PhotometricInterpretation is YCbCr, this field must be omitted. (Reader shall ignore this field.)
If PhotometricInterpretation is ITULAB and Decode values are default value, this fields may be omitted.
Decode maps the unsigned ITULAB-encoded sample values (Lsample, asample, bsample) to signed L*a*b* values, as follows:

$L^* = Decode[0] + Lsample \times (Decode[1] - Decode[0]) / (2^n - 1)$
$a^* = Decode[2] + asample \times (Decode[3] - Decode[2]) / (2^n - 1)$
$b^* = Decode[4] + bsample \times (Decode[5] - Decode[4]) / (2^n - 1)$

where Decode[0], Decode[2] and Decode[4] are the minimum values for L*, a* and b*; Decode[1], Decode[3] and Decode[5] are the maximum values for L*, a* and b*; and n is the BitsPerSample, either 8 or 12. For example, when n=8, L*=Decode[0] when Lsample=0 and L*=Decode[1] when Lsample=255.

$minimum = -(range \times offset) / 2^n - 1$
$maximum = minimum + range$

The Decode field default values depend on the color space. For the ITULAB color space encoding, the default values correspond to the base range and offset, as specified in ITU-T Rec. T.42 [T.42]. The following table gives the base range and offset values for BitsPerSample=8 and 12, and the corresponding default minimum and maximum default values for the Decode field, calculated using the equations above when PhotometricInterpetation=10.

Refer to ITU-T Rec. T.42 [T.42] to calculate the range and offset, and hence the minimum and maximum values, for other BitsPerSample values.

		ITU-T Rec. T.42 base values		Decode default values	
BitsPer -Sample	Component	Range	Offset	Min	Max
8	L*	100	0	0	100
	a*	170	128	-21760/255	21590/255
	b*	200	96	-19200/255	31800/255

For example, when PhotometricInterpretation=10 and BitsPerSample=8, the default value for Decode is (0, 100, -21760/255, 21590/255, -19200/255, 31800/255).]

YCbCrCoefficients(529) **RATIONAL**

Count = 3
If these field present, default values (CCIR601) must be set.
(299/1000, 587/1000, 114/1000)
These fields may be omitted.
Reader ignores these fields except YCbCr. (It is recommended that Writer omits these fields.)

© 2000 SD Association

CONFIDENTIAL

Standard		*LumaRed*	*LumaGreen*	*LumaBlue*
CCIR	Recommendation601-1	299 / 1000	587 / 1000	114 / 1000
CCIR	Recommendation709 2	125 / 10000	7154 / 10000	721 / 10000

Y = (*LumaRed* * R + *LumaGreen* * G + *LumaBlue* * B)
C b = (B - Y) / (2 - 2 * *LumaBlue*) + offset
C r = (R - Y) / (2 - 2 * *LumaRed*)+offset
R, G, and B may be computed from YC b C r as follows:
R = C r * (2 - 2 * *LumaRed*) + Y
G = (Y - *LumaBlue* * B - *LumaRed* * R) / *LumaGreen*
B = C b * (2 - 2 * *LumaBlue*) + Y
*offset is 128 if BitsPerSamples = 8.

5.2.2 Recommended TIFF Fields

ReferenceBlackWhite(532) **RATIONAL**

Count=6
Specifies a pair of headroom and footroom image data values (codes) for each pixel component. The first component code within a pair is associated with ReferenceBlack, and the second is associated with ReferenceWhite. The ordering of pairs is the same as those for pixel components of the PhotometricInterpretation type. ReferenceBlackWhite can be applied to images with a PhotometricInterpretation value of YCbCr.
The default values are (0/1, 255/1, 128/1, 255/1, 128/1, 255/1).
If Writer writes these fields, it must write default values. Writer may omit these fields. Reader recognizes default values only.

PlanerConfiguration(284) **SHORT**

Count=1
If SamplesPerPixel is 1, PlanerConfiguration is irrelevant, and need not be included.
It is recommended that this field is omitted.
If SamplesPerPixels is 3 and this field presents, the field value must be 1.

The other Recommended TIFF Fields are same as Monochrome binary image specification.

5.2.3 Other TIFF Fields

Same as Monochrome binary image specification

5.2.4 File Structure

Same as Monochrome binary image specification

5.2.5 JPEG Stream

If PhotometricInterpretation is ITULab, JPEG stream shall be based on ITU-T Rec. T.42 [T.42] ANNEX-E.
If PhotometricInterpretation is YCbCr, JPEG stream shall be based on JFIF [JFIF] except APP0 marker.

5.2.6 Relation from JFIF

Only if Writer selects YCbCr as PhotometricInterpretation, JFIF stream can be use as TIFF JPEG data. But contents in APP0 marker of JFIF shall not be inconsistent with TAGs of TIFF.
Reader may ignore APP0 marker. So, TIFF Tags shall reflect information in APP0 maker which relates TIFF Tags if JPEG data in TIFF includes APP0 used in JFIF. Reader may ignore APP0 marker.

© 2000 SD Association

CONFIDENTIAL

5.2.7 Relation from DCF(Exif)

If Writer selects YCbCr as PhotometricInterpretation, Exif [Exif] data can be use as TIFF JPEG data. But contents in APP1 marker of Exif shall not be inconsistent with TAGs of TIFF. Reader may ignore APP1 marker.

© 2000 SD Association

CONFIDENTIAL

6. JOB File

If JOB file does not present, Writer shall create JOB FILE when the Writer writes a job information..

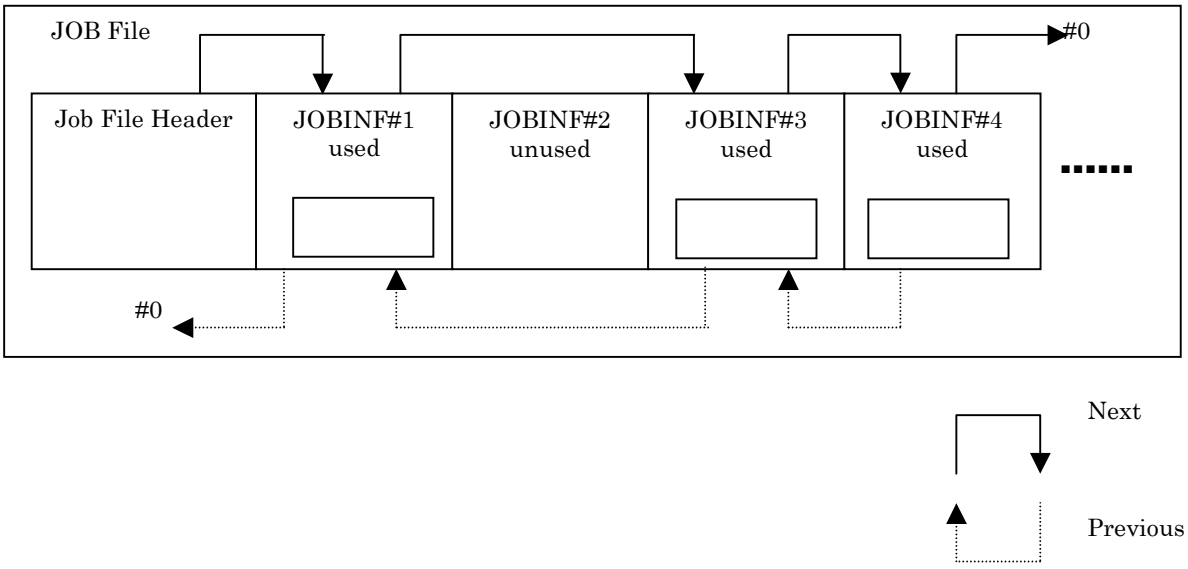
© 2000 SD Association

CONFIDENTIAL

6.1 Navigation Data Structure for Job File

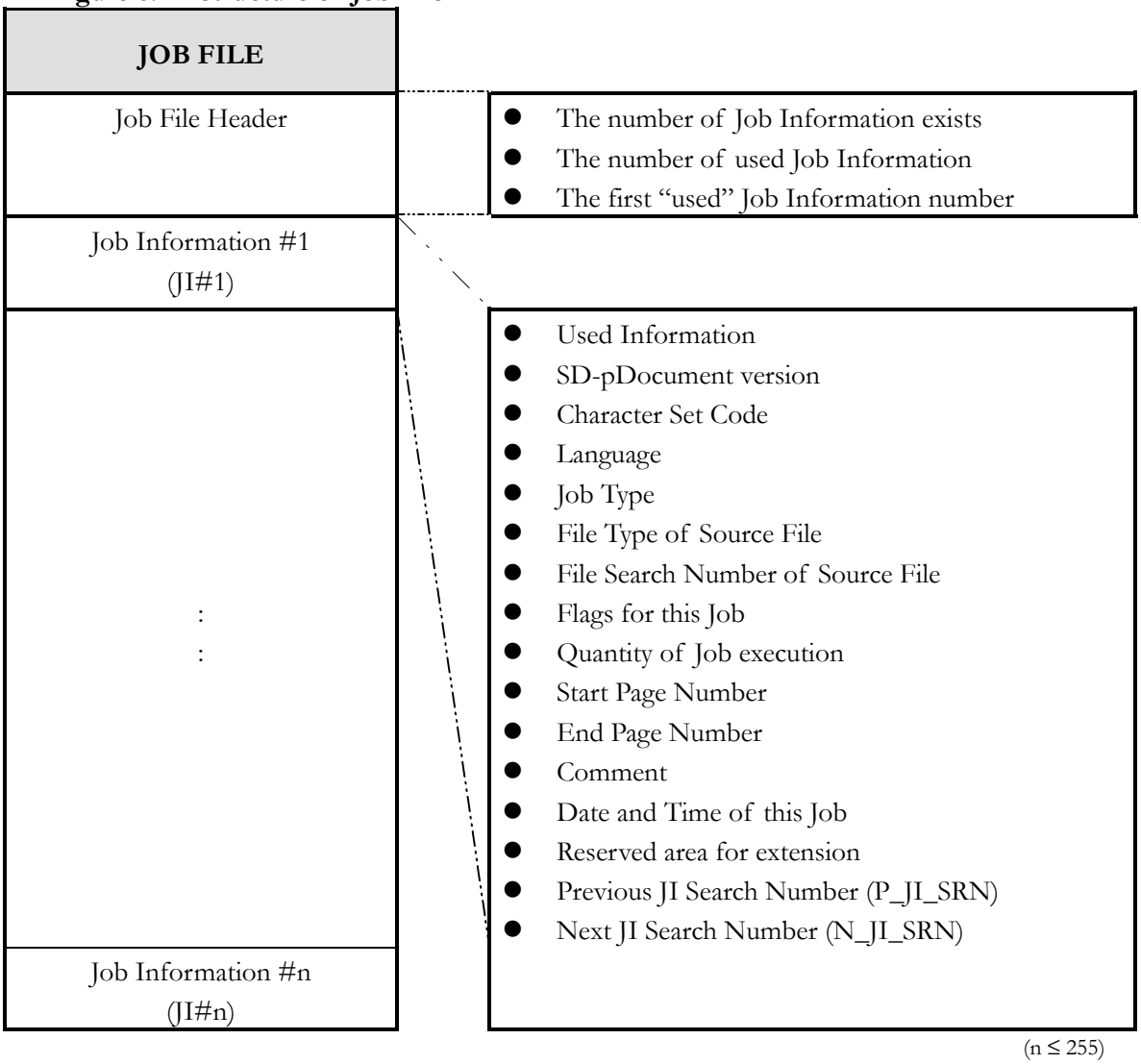
Data Structure for Job File is as shown in [Figure 6.1-1].

◊ **Figure 6.1-1 Navigation Structure for Job File**



6.2 Structure of Job File
Job File consists of 2 elements: the Job File Header and the Job Information (JOBINF).

◊ **Figure 6.2-1 Structure of Job File**



(n ≤ 255)

- **Job File Header**
  Job File Header describes JI information such as number of Jobs stored in the Job File, the number of JI exist, and the first used JI number.

- **Job Information (JI)**
  JI consists of:
- **Used Information**
  This information describes if JI is used or not.
  If information in a JI is valid, the JI is 'used'.
  If information in a JI in invalid the JI is 'unused'.
- **Version**
  the version of SD-pDocument supported
- **Character Set Code**

© 2000 SD Association

CONFIDENTIAL

Character code information of character fields in JI

- **Language**
  A language used in the character fields in JI
- **Job Type**
  Executed Job type (ex. Print)
- **File Type of Source File**
  Plain Text or Raster Image file
- **File Search Number of Source File**
  File number :1 to 65535
- **Flags of this Job**
- **Quantity of Job execution**
- **Start Page Number for Job execution**
- **End Page Number for Job execution**
- **Comment**
  Information about Job
- **Date and Time of this Job**
  Date and time created Job
- **Reserved area for Extension**
- **Previous JI search Number (P_JI_SRN)**
- **Next JI Search Number (N_JI_SRN)**

© 2000 SD Association

CONFIDENTIAL

6.3 Size of Job File

Structure of Job file is as shown in [Figure 6.3-1].

◊ **Figure 6.3-1 Size of Job File**

Job File

Job File Header (Mandatory)	Fixed Size(8B)
Job Information #1 (JI #1) (Optional)	Fixed Size(256B)
Job Information #2 (JI #2) (Optional)	Fixed Size(256B)
:	:
Job Information #n (JI #n) (Optional)	Fixed Size(256B)

(n ≤ 255)

6.4 Job File Header

As shown in [Table 6-1], Number of Job Information, Number of used Job Information, and First "used" JI number are described in Job File Header.

◊ **Table 6-1 Job File Header** (Description order)

RBP		Contents	Number of bytes
0 to 1	JI_Ns	Number of JIs exist	2 bytes
2 to 3	JI_NUSED	Number of used JIs	2 bytes
4 to 5	JI_FUN	First "used" JI number	2 bytes
6 to 7	Reserved	Reserved	2 bytes
Total			8 bytes

(RBP 0 to 1) JI_Ns
Describes number of JIs exist in Job File.

(RBP 2 to 3) JI_NUSED
Describes number of used JIs in Job File.

(RBP 4 to 5) JI_FUN
Describes the entry number of the first "used" Job Information.
If 0, there are NO "used" JIs in Job File.

© 2000 SD Association

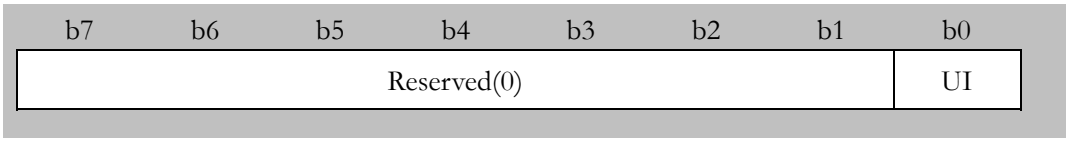
CONFIDENTIAL

6.5 Job Information (JI)

◊ **Table 6-2 JI** (Description order)

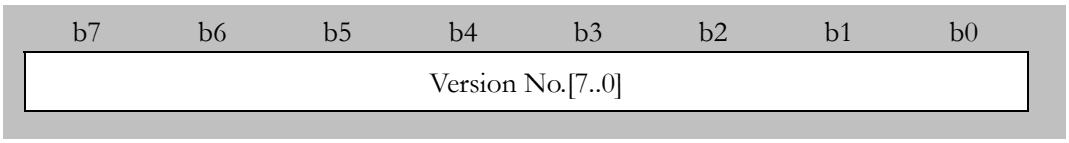
RBP		Contents	Number of bytes
0	UI	Used Information	1 byte
1	Reserved	Enter 00h	1 byte
2	Reserved	Enter 00h	1 byte
3	VERN	SD-pDocument Version No.	1 byte
4 to 5	CHAR_CODE	Character Set Code	2 bytes
6 to 13	LANG	Language	8 bytes
14	JTYPE	Job Type	1 byte
15	FTYPE	File Type of Source File	1 byte
16 to 17	JFLAG	Flags of this Job	2 bytes
18 to 19	SFL_SRN	File Search Number of Source File	2 bytes
20 to 21	REP_N	Quantity of Job Execution	2 bytes
22 to 23	S_PAGE	Start Page Number	2 bytes
24 to 25	E_PAGE	End Page Number	2 bytes
26 to 153	COMMENT	Comment	128 bytes
154 to 185	DATETIME	Original Date and Time	32 bytes
186 to 187	P_JI_SRN	Previous JI Search Number	2 bytes
188 to 189	N_JI_SRN	Next JI Search Number	2 bytes
190 to 255	reserved	Enter all 00h	66 bytes
Total			256bytes

(RPB 0) UI
Describes Used Information.



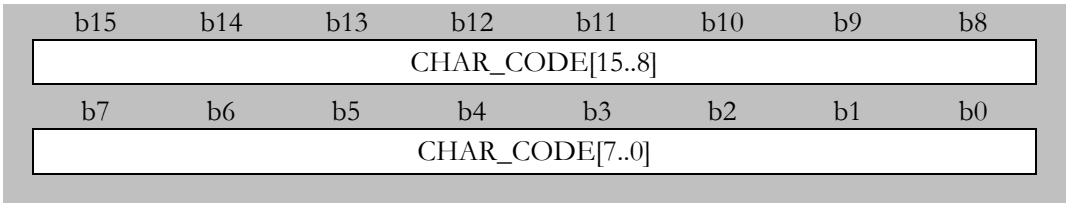
Used Information ... When the JI is not in use, enter 0.
When the JI is in use, enter 1.

(RPB 3) VERN
Describes version No. of SD-pDocument specification supported by the writer which had wrote this Job.



Version number ... 09h : version 0.9
10h : version 1.0
Others : reserved

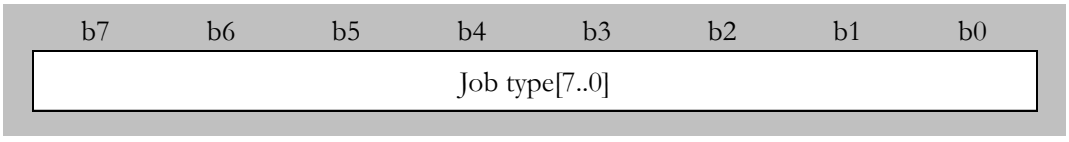
(RBP 4 to 5) CHAR_CODE
Describes character set code of the character field of this JI.



Character Set code … IANA Character Sets
See ANNEX B

(RBP 6 to 13) LANG
Describes TAG of a language used in the character fields of the JI.
This field must be terminated by NULL character.
LANG … Language TAG (ex. "en-us")
See ANNEX C

(RPB 14) JTYPE
Describes type of this Job.



Data Source ... 00h : Print
01h : Display

Others : reserved

© 2000 SD Association

CONFIDENTIAL

(RPB 15) FTYPE
Describes type of the source file (FOB) of this Job.

b7	b6	b5	b4	b3	b2	b1	b0
File type[7..0]							

Source file type ... 00h : Plain TEXT (The file name is TXTxxxxx.TXT)
01h : Raster Image (The file name is TIFxxxxx.TIF)

Others : reserved

(RBP 16 to 17) JFLAG
Describes job options by bit fields of which mean is depend on JTYPE and FTYPE.

	b15	b14	b13	b12	b11	b10	b9	b8
	Reserved(0)							

	b7	b6	b5	b4	b3	b2	b1	b0
	Reserved(0)		CONT	Reserved(0)	DEL	Print Face[2..0]		
JTYPE	Print			Print				
FTYPE	Plain Text			Plain Text or Raster Image				

Print Face … 000 : Single side print
001 : Both side print (Long Side Binding : left/top)[Figure 6.5-1]
010 : Both side print (Long Side Binding : right/bottom)[Figure 6.5-2]
011 : Both side print (Short Side Binding : top/right) [Figure 6.5-3]
100 : Both side print (Short Side Binding : bottom/left)[Figure 6.5-4]

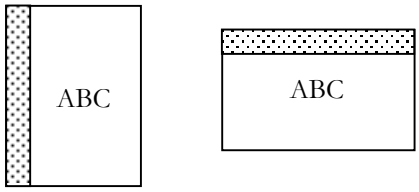
DEL … 1: Delete both of JI and Source File of this Job automatically,
after the job have been finished normally.
0: Delete only JI of this Job automatically,
after the job have been finished normally.

CONT … 0 : print on a new sheet.
1 : print on the same sheet that was printed by the previous print job, if the job printed Plain Text data.
This bit is valid if present and previous FTYPEs are 'Plain TEXT'.

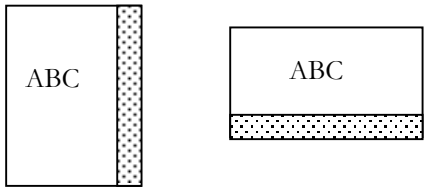
© 2000 SD Association

CONFIDENTIAL

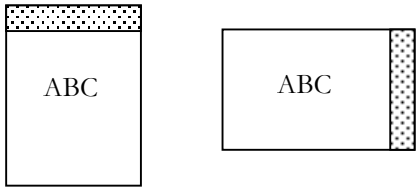
◊ **Figure 6.5-1 Long Side Binding : left/top**



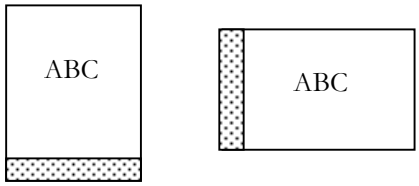
◊ **Figure 6.5-2 Long Side Binding : right/bottom**



◊ **Figure 6.5-3 Short Side Binding : top/right**



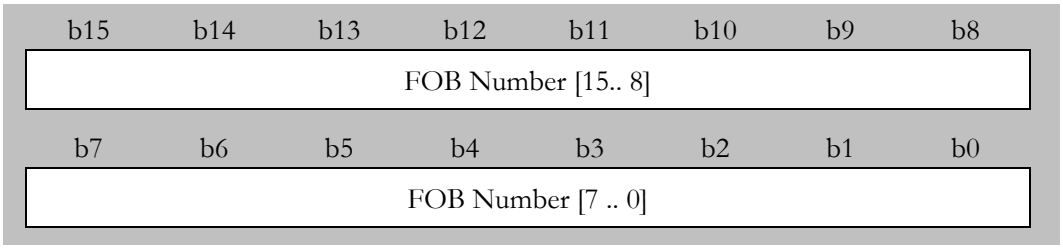
◊ **Figure 6.5-4 Short Side Binding : bottom/left**



© 2000 SD Association

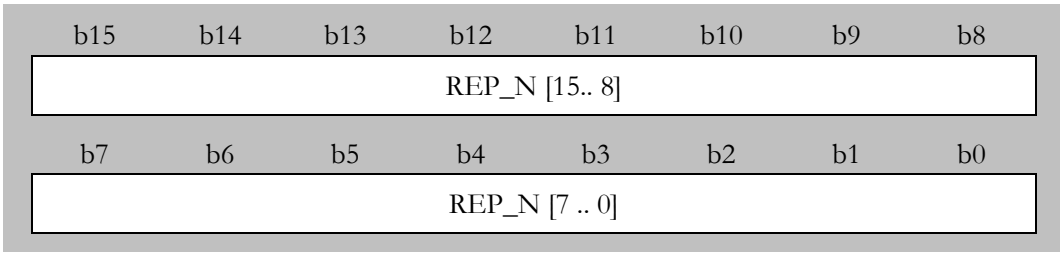
CONFIDENTIAL

(RBP 18 to 19) SFL_SRN
Describes the entry number of referenced FOB which is managed by this Job.
By using FTYPE and SFR_SRN, the file name referenced FOB will be generated.



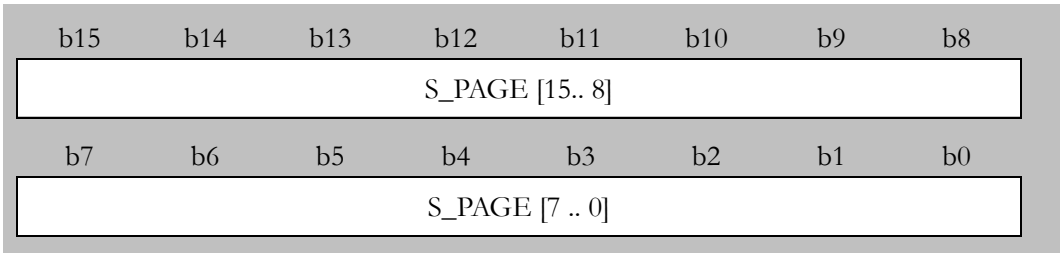
FOB Number ... Describes the number between '1' and '65,535'.
When no FOB is referenced, enter 0.

(RBP 20 to 21) REP_N
Describes the Quantity of Job execution. (ex: number of Copies)
REP_N is NOT valid, if JTYPE is 'Display'.



REP_N ... Describes the number between '1' and '999'.

(RBP 22 to 23) S_PAGE
Describes the start page of JOB. S_PAGE is valid on Raster Image FOB.
SPAGE is Not valid, if FTYPE is 'Plain Text'.



S_PAGE ... Describes the number between '1' and '999'.

© 2000 SD Association

CONFIDENTIAL

(RBP 24 to 25) E_PAGE
Describes the end page of JOB. E_PAGE is valid on Raster Image FOB.
EPAGE is Not valid, if FTYPE is 'Plain Text'.

b15	b14	b13	b12	b11	b10	b9	b8
E_PAGE [15.. 8]							
b7	b6	b5	b4	b3	b2	b1	b0
E_PAGE [7 .. 0]							

E_PAGE ... Describes the number between '1' and '999'.
'999' means the last page of FOB also.

(RBP 26 to 153) COMMENT
Describes comment of this Job.
This field must be terminated by NULL character.

(RBP 154 to 185) DATETIME
Describes the date and time at which this Job had been generated.
This field must be terminated by NULL character.
Format of date and time is based on ISO8601
yyyy-mm-ddThh:mm:ss+hh:ss (ex. "1998-09-15T21:15:00+09:00")

(RBP 186 to 187) P_JI_SRN
Describes the entry number of previous JI.

b15	b14	b13	b12	b11	b10	b9	b8
JI Number [15.. 8]							
b7	b6	b5	b4	b3	b2	b1	b0
JI Number [7 .. 0]							

Previous JI No. ... Describes the number between '1' and '255'.
When no JI is referenced, enter 0.

© 2000 SD Association

CONFIDENTIAL

(RBP 188 to 189) N_JI_SRN
Describes the entry number of next JI.

b15	b14	b13	b12	b11	b10	b9	b8
JI Number [15.. 8]							
b7	b6	b5	b4	b3	b2	b1	b0
JI Number [7 .. 0]							

Next JI No. ... Describes the number between '1' and '255'.
When no JI is referenced, enter 0.

© 2000 SD Association

CONFIDENTIAL

Annex A. Functional Requirements for SD_pDocument specification

Classified Functions

Functions of the products compatible with SD_pDocument can be classified into two systems, Plain TEXT and TIFF(Raster Image data)

◊ **Table A-1 Classifications**

Classifications		Functions
TEXT	Reader	Display or Print Plain TEXT files stored in an SD memory card
	Writer	Record Plain TEXT files and JOB into an SD memory card
TIFF	Reader	Display or Print Plain TEXT files stored in an SD memory card
	Writer	Record Plain TEXT files and JOB into an SD memory card

Functional categories for SD_pDocument products

The products compatible with SD_pDocument can be classified into two categories as follows.

◊ **Table A-2** Categories of products

No.	Category	Function	File Type			Examples
			TEXT	TIFF	JOB	
1	Category 1 TEXT	Reader	md*1	-	md	Small-size dot printer mobile phone
		Writer	md		op	mobile phone PDA
2	Category 2 TIFF	Reader		md	md	Fax Printer,Copy machine, TV, Projecter, Video printer, PC
		Writer		md	op	Fax, Copy machine, Image scanner, TV, PC

md: Mandatory
op: Optional

Notes:

1. Reader shall support JOB file.
2. It is recommended that Reader which reproduces color TIFF file manage monochrome TIFF files also.
3. It is recommended that Reader which reproduces only monochrome image manage color TIFF files by conversion from color to monochrome.
4. It is recommended that Reader which reproduce color image manage monochrome TIFF.

*1 Mandate of Character code is US-ASCII, but it is strongly recommended to support regional character code for the region where the products will be used.
(ex. Kanji code is strongly recommended for products used in Japanese market.)

© 2000 SD Association

CONFIDENTIAL

Annex B. Character Set Code

Numbers of character sets or encoding names are defined by IANA.
The following numbers can be used as character set codes in this specification.

Character Set code … Assigned MIB enum by IANA : Character Sets name

3 : US-ASCII
4 : ISO8859-1
5 : ISO8859-2
6 : ISO8859-3
7 : ISO8859-4
8 : ISO8859-5
9 : ISO8859-6
10 : ISO8859-7
11 : ISO8859-8
12 : ISO8859-9
15 : JIS X 0201
17 : Shift-JIS
106 : UTF-8 [RFC2279]
2001 : ISO-8859-1-Windows-3.1-Latin-1
2002 : ISO-8859-2-Windows-Latin-2
2003 : ISO-8859-9-Windows-Latin-5

Notes: Euro (currency symbol) is often assigned 0x80.

© 2000 SD Association

CONFIDENTIAL

Annex C. Language Tag

Language Tags defined in RFC1766 "Tags for the Identification of Languages" are used in this specification.

For example:

"en-us"	American English
"en"	English
"fr"	French
"du"	German
"ja"	Japanese
"ko"	Korean
"zh"	Chinese

© 2000 SD Association

CONFIDENTIAL

This page is intentionally left blank

© 2000 SD Association

SD Memory Card Specifications

Part 8 VIDEO SPECIFICATIONS

Version 1.0 Draft e

February 2001

Technical Committee

SD Association

Conditions for publication

Publisher and Copyright Holder

SD Association

Exemption

None will be liable for any damages from use of this document.

ã 2001 SD Association

This page is intentionally left blank.

This page is intentionally left blank.

CONFIDENTIAL

INDEX

Part 8 VIDEO SPECIFICATIONS

1. General VI1-1
1.1 Scope 1
1.2 General Specification of Presentation Data 2
1.2.1 Hierarchical structure of SD-Video specification 2
1.2.2 Profile of SD-Video specification 3
1.3 Normative References 5
2 Technical Elements VI2-1
2.1 Definitions 1
2.2 Symbols 2
2.2.1 Relational operators 2
2.2.2 Arithmetic operators 2
2.3 Notations 3
2.3.1 Numerical notation 3
2.3.2 Range 3
2.3.3 Bit fields 4
2.4 Terminology 5
2.4.1 SD-Video Directory 5
2.4.2 Management Data Directory 5
2.4.3 Program Directory 5
2.4.4 Management Data File 5
2.4.5 Program Manager File 5
2.4.6 Program Attribute 5
2.4.7 Playlist Manager File 5
2.4.8 Playlist Information 5
2.4.9 Program Files 5
2.4.10 Program Information File 6
2.4.11 Media Object 6
2.4.12 Object ID 6
2.4.13 Movie 6
2.4.14 Separately-Recorded Audio 6
2.4.15 Replaced Audio 6
2.4.16 Media Object Data File 6
2.4.17 Media Object Information File 6
2.4.18 Still Picture Data File 7
2.4.19 Text Data File 7
2.4.20 Scene Description Data File 7
2.4.21 Program Number 7

ã 2001 SD Association

CONFIDENTIAL

2.4.22 File Number 7
2.4.23 Time Search Table 7
2.4.24 Time Search Entry 7
2.4.25 Time Search Entry Information 7
2.4.26 Entry Frame 7
2.4.27 Entry Pack 7
2.4.28 Representation Picture 8
2.4.29 Resume Marker 8
2.4.30 Marker 8
2.4.31 Media Object Data Unit 8
2.4.32 Pack 8
2.4.33 stream_id 8
2.4.34 stream_type 8
2.5 Abbreviations 9
3 Introduction VI3-1
3.1 Total Data Structure 1
3.1.1 Program 1
3.1.2 Playlist 4
3.1.3 Organization of Directories and Files 5
3.1.4 Structure of Media Object Data File 8
3.2 Navigation Structure 10
3.2.1 Random Access Methods to Media Objects 10
3.2.2 Resuming Last Playback 17
3.2.3 Separately-Recorded Audio and Replaced Audio 18
3.2.4 Subtitles, Credits, and Other Texts Synchronized with Video and Audio Streams 20
4 Directory and File Naming Convention VI4-1
4.1 SD-Video Directory 1
4.2 Management Data Directory 2
4.3 Program Directory 3
4.4 Program Files 4
4.4.1 Program Information File 4
4.4.2 Media Object Data File 5
4.4.3 Media Object Information File 5
4.4.4 Advanced Streaming Format File 6
4.4.5 Still Picture Data File 7
4.4.6 Text Data File 7
4.4.7 Scene Description Data File 8
4.5 Method for generating file name with hexadecimal number 9
5 Navigation Data Structure VI5-1
5.1 Miscellaneous definitions 1
5.1.1 Object ID 1

ã 2001 SD Association

CONFIDENTIAL

5.1.2 Data Types 2
5.2 Considerations 4
5.2.1 Reserved Fields 4
5.2.2 Version 4
5.2.3 DataSize 4
5.3 Management Data File 5
5.4 Program Manager File 7
5.4.1 Program Manager (PRG_MGR) 7
5.4.2 Program Attribute (PRG_ATTR) 9
5.5 Program Information File 12
5.5.1 Program Information (PRG_INFO) 12
5.6 Playlist Manager File 16
5.6.1 Playlist Manager (PLST_MGR) 16
5.6.2 Playlist Information (PLST_INFO) 17
5.7 Media Object Information File 20
5.7.1 Media Object Information (MOI) 20
6 Scene Description Data Structure VI6-1
7. Media Object Data File VI-71
7.1 Definition of Media Object Data File 1
7.2 Media Object Data Unit 3
7.3 Access Unit 4
7.4 Pack 5
7.4.1 Structure of pack 5
7.4.2 Video Pack (V_PCK) 7
7.4.3 Audio Pack (A_PCK) 10
7.4.4 Text Pack (T_PCK) 13
7.4.5 Program stream map Pack (P_PCK) 15
7.5 Player Reference Model 17
Annex VIX-1
Annex A. Mandatory Item of SD-Video Specification 1
Annex A.1. MGR_DATA 1
Annex A.2. PRG_MGR 2
Annex A.3. PRG_INFO 3
Annex A.4. PLST_MGR 4
Annex A.5. MOI 5
Annex A.6. Media Object Data File 7
Annex A.7. Advanced Streaming Format File 7
Annex A.8. Program Title 7
Annex A.9. Player Functions 8
Annex A.10. Restrictions of Maximum Data Size 10
Annex B Operation Example 11

ã 2001 SD Association

CONFIDENTIAL

Annex B.1. Recording Procedure 11
Annex B.2. Editing Procedure 15
Annex B.3. Playback Procedure 18
Annex B.4. Audio-Replacement Procedure 29

ã 2001 SD Association

CONFIDENTIAL

List of Figure

Figure 1.1-1 : Position of SD-Video Specifications.....VI1-1
Figure 3.1-1: Structure of Program.....VI3-1
Figure 3.1-2: Scene Description.....3
Figure 3.1-3: Organization of Directories and Files for SD-Video.....5
Figure 3.1-4: Structure of Media Object Data File.....9
Figure 3.2-1: Mapping Program Section to Set of Media Objects.....11
Figure 3.2-2: Procedure to Obtain Target Frame Data by Time Search Table of Type 1 or 2..13
Figure 3.2-3: Procedure to Obtain Target Frame Data by Time Search Table of Type 3.....15
Figure 3.2-4: Fast Forward/Reverse Play Using MODU Information.....16
Figure 3.2-5: Separately-Recorded Audios and Replaced Audios.....18
Figure 5.1-1 : Example of Object ID.....VI5-2
Figure 5.7-1 : Table of MODU Information 24
Figure 5.7-2 : Structure of Time Search Table in case of having MODU Information 25
Figure 5.7-3 : Structure of Time Search Table in case of movie interleaved video and audio or text.....26
Figure 5.7-4 : Structure of Time Search Table in case of no MODU Information 27
Figure 5.7-5 : Structure of Time Search Table in case of variable pack size audio.....28
Figure 5.7-6 : Structure of Time Search Table in case of fixed pack size audio.....29
Figure 7.1-1 : Structure of pack (in case of no padding packet).....VI7-1
Figure 7.2-1 : Structure of MODU.....3
Figure 7.3-1 : Definitions of Video Access Unit.....4
Figure 7.4-1 : Structure of Pack (in case of no padding Packet).....5
Figure 7.4-2 : Structure of pack length adjustment 5
Figure 7.4-3 : Structure of Video Pack.....7
Figure 7.4-4 : Structure of Audio Pack 10
Figure 7.4-5 : Structure of Text pPack.....13
Figure 7.4-6 : Structure of Program stream map Pack.....15
Figure 7.5-1 : Player Reference model.....18

Figure Annex B-1 : When a blank card is inserted, create directories and files VIX-11
Figure Annex B-2 : When Rec is Started, create new MOD file and record AV data.....12
Figure Annex B-3 : When Rec. button is released, renew MOI parameters.....12
Figure Annex B-4 : Then, renew Program Information File in Program nnn+1 directory.....13
Figure Annex B-5 : Then, renew PRG_MGR and PRG_ATTR for Program nnn+1 13
Figure Annex B-6 : When Rec. mode is released, renew NumPro in PRG_MGR 14
Figure Annex B-7 : New Playlist creation 15
Figure Annex B-8 : Playlist Editing.....16

ã 2001 SD Association

CONFIDENTIAL

Figure Annex B-9 : Partial Deletion..... 17
Figure Annex B-10 : Playlist Playback: Compare StartPos of the first PLST_INFO with the MoDuration in the first MO_INFO in PRGnnn.PGI..... 18
Figure Annex B-11 : Then, Compare the residue of Start Position with the next MO Durations until it becomes smaller than the next MODuration..... 19
Figure Annex B-12 : Then, Compare the residue of Start Position with the next TstInterval until it becomes smaller than the next TstInterval..... 20
Figure Annex B-13 : Stop when the residue of End Position become minus. 21
Figure Annex B-14 : Compare Starting Time with MoDurations in Program nnn. 22
Figure Annex B-15: Then, Compare the residue of Starting Time with the next TstInterval until it becomes smaller than the next TstInterval..... 23
Figure Annex B-16 : Continue Playback until the residue of End Position becomes minus..... 24
Figure Annex B-17 : Fast Forward Playback 26
Figure Annex B-18 : Decode every Entry Frame 27
Figure Annex B- 19 : Decode every Entry Frame backward 28
Figure Annex B-20 : Find stating Media Object and ending Media Object..... 29
Figure Annex B-21 : Set b1 of MoAttr in MO_INFO and create AUDmmm.MOD and AUDmmm.MOI..... 29
Figure Annex B-22 : Record Audio-Replacement data in AUDmmm.MOD up to AUDnnn.MOD. 30

ã 2001 SD Association

CONFIDENTIAL

List of Table

Table 1.2-1 : Hierarchical structure for contents classification.....VI1-2
Table 1.2-2 : Presentation data (Mobile Video Profile).....3
Table 1.2-3 : Mobile Video Profile Specification.....4
Table 4.3-1 : Specification of naming convention of Program Directory2.....VI4-3
Table 4.4-1 : Program Files under Program Directory 4
Table 4.4-2 : Specification of naming convention of Program Information File.....4
Table 4.4-3 : Specification of naming convention of Media Object Data File.....5
Table 4.4-4 : Specification of naming convention of Media Object Information File.....6
Table 4.4-5 : Specification of naming convention of Advanced Streaming Format File.....6
Table 4.4-6 : Specification of naming convention of Still Picture Data File.....7
Table 4.4-7 : Specification of naming convention of Text Data File 7
Table 4.4-8 : Specification of naming convention of Scene Description Data File.....8
Table 5.1-1 : OBJPOS VI5-3
Table 5.3-1 : MGR_DATA.....5
Table 5.4-1 : PRG_MGR 7
Table 5.4-2 : PRG_ATTR.....9
Table 5.5-1 : PRG_INFO 12
Table 5.5-2 : MO_INFO 13
Table 5.5-3 : PRG_EXT.....15
Table 5.6-1 : PLST_MGR.....16
Table 5.6-2 : PLST_INFO 17
Table 5.6-3 : PRGSECTION.....19
Table 5.7-1 : MOI.....20
Table 5.7-2 : CREATION_TIME.....21
Table 5.7-3 : STRM_ATTR 21
Table 5.7-4 : MODU_INFO 23
Table 5.7-5 : TSE_INFO1.....24
Table 5.7-6 : TSE_INFO2.....26
Table 7.1-1 : List of Pack types.....VI7-1
Table 7.1-2 : stream_id 2
Table 7.1-3 : stream_type.....2
Table 7.3-1 : Definitions of Access Unit.....4
Table 7.4-1 : Pack length adjustment method.....5
Table 7.4-2 : Pack header.....6
Table 7.4-3 : System header in V_PCK.....8
Table 7.4-4 : Video Packet 9
Table 7.4-5 : System header in A_PCK.....11

ã 2001 SD Association

CONFIDENTIAL

Table 7.4-6 : Audio Packet 12
Table 7.4-7 : Text Packet 14
Table 7.4-8 : Program stream map Packet 15
Table 7.4-9: (a) Video portion of Program stream map Packet 16
Table 7.4-10: (b) Audio portion of Program stream map Packet 16
Table 7.4-11: (c) Text portion of Program stream map Packet 16
Table 7.5-1 : Input Buffer definitions 17

Table Annex A-1 : MGR_DATA VIX-1
Table Annex A-2 : PRG_MGR 2
Table Annex A- 3: PRG_INFO 3
Table Annex A-4 : PLST_MGR 4
Table Annex A-5 : Header of MOI 5
Table Annex A-6 : Tail of MOI in case of TstType==1 5
Table Annex A-7 : Tail of MOI in case of TstType==2 5
Table Annex A-8 : Tail of MOI in case of TstType==3 6
Table Annex A- 9 : Functions supported by players 8
Table Annex A- 10 : Restrictions of Maximum Number of Objects 10
Table Annex A- 11: Restrictions of Maximum Data Size of Information Tables 10

ã 2001 SD Association

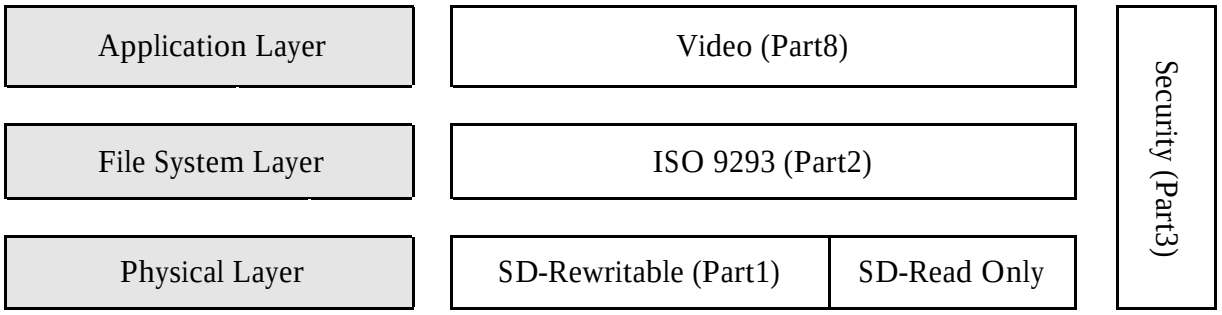
CONFIDENTIAL

1. General

1.1 Scope

SD-Video Specifications 1.0 is the Part 8 of SD Memory Card Specifications Family. In the SD-Video Specifications, the video presentation data complies with MPEG-4 Standards depending on the application. The SD Memory card applying these specifications shall comply with **"Part1. PHYSICAL SPECIFICATIONS", "Part2. FILE SYSTEM SPECIFICATIONS"** and **"Part3. SECURITY SPECIFICATIONS"** of SD Memory Card Specifications.

Figure 1.1-1 : Position of SD-Video Specifications



ã 2001 SD Association

CONFIDENTIAL

1.2 General Specification of Presentation Data

1.2.1 Hierarchical structure of SD-Video specification

SD-Video specification consist of the following information hierarchy. Playlist Manager is the highest layer which specifies contents to be played back according to user intention. Program Manager specifies all Programs recorded in an SD-Video card and is referred by Playlist Manager. Program is a directory which contains AV streams as Media Object. Media Object is a combination of Media Object Data File and Media Object Information File. Media Object Data File contains AV stream, and Media Object Information File contains its timing information. The duration of Media Object is user definable. An example of the length is the AV stream recorded after Rec. button was pushed down until it is released. Media Object Data File consists of Media Object Data Units. Media Object Data Unit includes one or more GOV of MPEG-4 video stream. Media Object Data Unit is a sequence of Packs. Pack includes one Packet.

Table 1.2-1 : Hierarchical structure for contents classification

Hierarchy name	Maximum number
Playlist Manager (User defined, n Programs)	1 per card
Program Manager (TOC, all Programs)	1 per card
Program (m Media Object Data Files)	99 per card
Program Information File	1 per Program
Media Object (AV contents & timing information)	99 per Program
Media Object Data File (n Packets)	max 3 per Media Object (Note 1)
Media Object Information File	1 for each Media Object Data File
Media Object Data Unit (one or more GOVs)	any number per Media Object Data File
Pack (one Packet)	any number per Media Object Data Unit
Packet	1 per Pack (Note 2)

Note 1: Media Object can contain up to three different Media Object Data Files. They are Movie Data File, Separately-Recorded Audio Data File and Replaced Audio Data File. Each of them can co-exist with other Data Files.

Note 2: One Padding Packet can coexist with a Packet in a Pack.

ã 2001 SD Association

CONFIDENTIAL

1.2.2 Profile of SD-Video specification

As there are two main application areas envisaged, SD-Video will have two Profiles for Mobile use and Entertainment use. In the SD-Video Specifications 1.0, Mobile Video Profile is specified for mobile applications such as cellular video phones, or internet applications, where MPEG-4 streams are used. For mobile phone application, MPEG-4 video simple profile is mainly used together with AMR voice coding. For the MPEG-4 stream, the structure and the timing of AV objects may be described in Scene Description Data written in SMIL.
A profile for the Entertainment use is planned to be specified in a latter version of SD-Video Specifications.

Table 1.2-2 : Presentation data (Mobile Video Profile)

Data	General specification
Video	MPEG-4 Video [Simple profile/Level 1 restricted, Level 1, Level 2, Level 3, Core profile/Level 1] Number of streams : 1 to 4
Audio	Coding mode : AMR*, MPEG-2 AAC(LC), G.726, Windows Media Audio Number of streams : max. 1
	AMR data basics Sampling frequency :8kHz Mode : Frame Type Index 0-8 and 15 Number of channels :1
	AAC data basics Sampling frequency :16/22.05/24/32/44.1/48 kHz Bitrates per channel :16~72kbps/channel Number of channels :max. 2
Picture	Coding mode : JPEG Number of Pixels : 96x96, 160x120 (recommended) 160x120 – 1800x1200 (optional) Chrominance sampling : 4:2:2, 4:2:0
Text	Character set code : ISO646, JISX0201, ISO8859-1, JISX0208
Scene Description	SMIL 2.0 Basic Profile

* AMR Interface Format 2 (AMR IF2) shall be used. AMR IF2 is defined in 3G TS.26.101.

Reference : 3G TS 26.101 V.3.1.0 (2000-03), 3GPP; TSG-SA Codec WG;
Mandatory Speech Codec speech processing functions;
AMR Speech Codec Frame Structure (Release 1999)

ã 2001 SD Association

CONFIDENTIAL

In Mobile Video Profile, MPEG-4 Video Simple profile/Level 1 restricted is mandatory for players. Simple profile/Level 1 restricted is a restricted Simple profile/Level 1 with the following restrictions.

maximum picture parameters = 176 pels/line, 144 lines/frame and 15frames/second.
where, Max. number of objects = 1, vop_fcode =1, intra_dc_vlc_threshold = 0,
AC prediction can be used only with a constant Quantization Parameter value across a VOP.

This specification will be published as "MPEG-4 Video Simple profile/Level 0" in ISO/IEC 14496-2:2000/AMD2.

Simple profile/Level 1, Level 2 and Level 3 are recommended for players. "Recommended" means players should implement recommended functions as much as possible. If full implementation is difficult, Conforming equipment shall safely process recommended functions without aborting them. Core profile/Level 1 is optional. Regarding audio, at least one coding format among AMR, G.726, AAC and Windows Media Audio shall be playable by players. JPEG still picture is optional. All text character sets are optional. SMIL scene description is optional.

Table 1.2-3 : Mobile Video Profile Specification

Data	item	in Card	for Player
Video	MPEG-4 Simple profile/Level 1 restricted	at least one profile/level shall exist	mandatory
	MPEG-4 Simple profile/Level 1, 2, 3		recommended
	MPEG-4 Core profile/Level 1		optional
Audio	AMR (IF2)	Optional	At least one coding format shall be playable
	G.726	Optional	
	AAC(LC)	Optional	
	Windows Media Audio	Optional	
Picture	JPEG	optional	optional
Text	ISO646, JISX0201, ISO8859-1, JISX0208	optional	optional
Scene Description	SMIL	optional	optional

ã 2001 SD Association

CONFIDENTIAL

1.3 Normative References

1) ISO/IEC 646: 1983(ASCII)
Information processing – ISO 7-bit coded character set for information interchange

2) ISO 8859-1: 1987
Information processing – 8-bit single-byte coded graphic character sets – Part 1 Latin alphabet No.1

3) JIS X 0201: 1997 (Kana)
7-bit and 8-bit coded character sets for information interchange

4) JIS X 0208: 1997 (Shift JIS Kanji)
7-bit and 8-bit 2-byte coded Kanji character sets for information interchange

5) ISO/IEC 13818: 1994 (MPEG-2)
Information technology – Generic coding of moving pictures and associated audio information

6) ISO/IEC 14496: 1999 (MPEG-4)
Information technology – Generic coding of audio-visual objects

7) W3C Working Draft 21 September 2000 (SMIL)
Synchronized Multimedia Integration Language (SMIL 2.0) Specification

8) 3GPP TSG-SA Codec WG SPEC (AMR)
3GPP TS 26.071: AMR Speech Codec; General Description
3GPP TS 26.101: AMR Speech Codec Frame Structure
3GPP TS 26.104: ANSI-C code for the floating-point AMR speech codec

9) SD Memory Card Specifications
Part 1: PHYSICAL SPECIFICATIONS

10) SD Memory Card Specifications
Part 2: FILESYSTEM SPECIFICATIONS

11) SD Memory Card Specifications
Part 3: SECURITY SPECIFICATIONS

12) ISO/IEC 10918-1:1994 (JPEG)
Information technology - Digital compression and coding of continuous-tone still pictures: requirements and guidelines (JPEG)

13) ITU-T Recommendation G.726 (1990)
General Aspects of Digital Transmission Systems, Terminal Equipment - 40, 32, 24,16kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)

ã 2001 SD Association

CONFIDENTIAL

14) Advanced Streaming Format

Advanced Streaming Format is a file format specified by Microsoft Corporation.
This naming will be changed to “Windows Media Container” by Microsoft Corporation.

15) Windows Media Audio

Windows Media Audio is an audio coding format specified by Microsoft Corporation.

ã 2001 SD Association

CONFIDENTIAL

2 Technical Elements

2.1 Definitions

In the SD-Video Specifications, the following definitions shall be applied :

mandatory Mandatory functions must be implemented. If any mandatory function is not implemented in the equipment, the equipment can not be said to conform to SD-Video Specification.

optional Optional functions do not have to be implemented in the equipment which claims to conform to SD-Video Specification. Equipment which conforms to SD-Video Specification, may implement optional functions without losing the conformity to SD-Video Specification.

Inter-operability between the equipment implementing the optional functions should be assured by the manufacturers who implement them.

recommended Recommended functions should be implemented as much as possible but may not be implemented in full specs. Conforming equipment shall safely process recommended functions without aborting them.

reserved Reserved field. The term "reserved" indicates that the value may be used in the future for SD-Video specification. Unless otherwise specified within this specification, all reserved bits shall be set to zero.

ZERO represents a single bit with the value 0.

ONE represents a single bit with the value 1.

data_element [m..n] The inclusive range of bits between bit m and bit n in the data_element.

ã 2001 SD Association

CONFIDENTIAL

2.2 Symbols

2.2.1 Relational operators

== or =	Equal to

2.2.2 Arithmetic operators

+	Addition
-	Subtract
*	Multiplication
/	Division
round(x)	rounds off the argument x to the nearest integer
floor(x)	rounds off the argument x to the nearest integer that is not larger than x

ã 2001 SD Association

CONFIDENTIAL

2.3 Notations

2.3.1 Numerical notation

Numbers in decimal notation are represented by decimal digits, namely 0 to 9. **Digit(s)** represents any digit from digit 0 to 9.

Numbers in hexadecimal notation are represented as a sequence of one or more hexadecimal digits namely 0 to 9 and A to F, suffixed by "h".

Numbers in binary notation are represented as a sequence of one or more binary digits namely 0 to 1, suffixed by "b".

Multiple-byte numerical values in a description field shall be recorded in the **big endian** representation. e.g.) the 32-bit hexadecimal number '12345678h' shall be recorded as '12h', '34h', '56h', '78h'.

1 kB represents 2^{10} bytes

1 MB represents 2^{20} bytes

1 GB represents 2^{30} bytes

2.3.2 Range

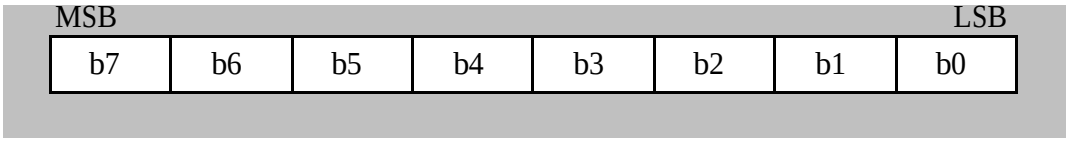
Constant_1..Constant_2 or [Constant1..Constant2] denotes the range from and including Constant_1 up to and including Constant_2, in increments of 1.

ã 2001 SD Association

2.3.3 Bit fields

Certain fields or parts of fields are interpreted as an array of bits.

This array of bits shall be referred to as a bit field. Bit positions within an ***n*** bit field are numbered such that the least significant bit is numbered 0 and the most significant bit is numbered ***n–1***.



2.4 Terminology

2.4.1 SD-Video Directory

This is a directory with the name “SD_VIDEO” directly created under the root directory. All data files managed by SD-Video are created under this directory.

2.4.2 Management Data Directory

This is a directory with the name “MGR_INFO” directly created under SD-Video Directory. This directory contains Management Data File (MGR_DATA), Program Manager File (PRG_MGR) and Playlist Manager File (PLST_MGR).

2.4.3 Program Directory

This is a directory that contains Program Files and created for each Program directly under SD-Video Directory.

2.4.4 Management Data File

This is a file with the name “MGR_DATA” created in Management Data Directory. This file defines global characteristics of SD-Video management data, contains version information, text information and so forth.

2.4.5 Program Manager File

This is a file with the name “PRG_MGR” created in Management Data Directory. This file defines characteristics of all Programs in SD-Video, contains total playback duration of Programs, the number of Programs, each Program Attributes and so forth.

2.4.6 Program Attribute

This is a structure which describes the attribute of Program and is contained in Program Manager File.

2.4.7 Playlist Manager File

This is a file with the name “PLST_MGR” created in Management Data Directory. This file defines characteristics of all Playlists of SD-Video, contains the number of Playlists, each Playlist Information and so forth.

2.4.8 Playlist Information

This is a structure which describes the information of Playlist and is contained in Playlist Manager File.

2.4.9 Program Files

Program Files are created under Program Directory. There are seven kinds of Program Files, which are Program Information File, Media Object Data File, Media Object Information File, Still Picture

ã 2001 SD Association

CONFIDENTIAL

Data File, Text Data File, Scene Description Data File and Advanced Streaming Format File. Advanced Streaming Format File can be used alternatively with Media Object Data File and Media Object Information File.

2.4.10 Program Information File

This is a file with the name "PRGxxx.PGI" created in Program Directory. This file defines characteristics of the Program, contains Profile, the number of Media Objects, the information of each Media Object and so forth.

2.4.11 Media Object

Media Object is a conceptual unit composed of video and audio streams to be played back together and their time information. Media Object shall include at least one video stream. Media Object may be divided into up to three exclusive portions to realize dynamic stream switching during its playback. The portions are Movie, Separately-Recorded Audio, and Replaced Audio.

2.4.12 Object ID

This is a 32-bit ID to identify an object such as Program, Playlist, Media Object and so forth.

2.4.13 Movie

Movie is the mandatory portion of Media Object and shall contain all video streams and their time information composing the Media Object. Audio streams of the same Media Object may be also contained in this portion.

2.4.14 Separately-Recorded Audio

Separately-Recorded Audio is an optional portion of Media Object that consists of audio streams with their time information, which should be played back superseding audio streams in Movie portion of the Media Object.

2.4.15 Replaced Audio

Replaced Audio is an optional portion of Media Object that consists of audio streams with their time information, which should be played back superseding audio streams in the other portions (Movie and Separately-Recorded Audio) of the Media Object.

2.4.16 Media Object Data File

Media Object Data File (MOD file) is one of the Program Files, and forms a part of the Program. A Program can consist of more than one Media Object Data File.

2.4.17 Media Object Information File

Media Object Information File is one of the Program Files, and is the information of Media Object Data File that has the same File Number. The number of Media Object Information Files is the same as the number of Media Object Data Files in the Program.

ã 2001 SD Association

CONFIDENTIAL

2.4.18 Still Picture Data File

Still Picture Data File is one of the Program Files, and contains a still image. The format is JPEG. This file is referred from only Scene Description Data File.

2.4.19 Text Data File

Text Data File is one of the Program Files and contains a plain text. This file is referred from only Scene Description Data File.

2.4.20 Scene Description Data File

Scene Description Data File is one of the Program Files and contains the scene described by the scene description language, SMIL.

2.4.21 Program Number

Program Number is the number assigned to the Program.

2.4.22 File Number

File Number is the number assigned to the Program File.

2.4.23 Time Search Table

This is a table of Time Search Entry.

2.4.24 Time Search Entry

This is a structure which contains information of Time Search Entry.

2.4.25 Time Search Entry Information

This is a helper to find entry points of MOD file when the decoder application plays in the middle of the stream. Time Search Entry Information is created at regular intervals, and its index can be calculated by dividing the entry time offset by the interval value. Time Search Entry Information has an offset value of the entry point in MOD file.

2.4.26 Entry Frame

This is a frame that the decoder application can start to play from, and it is typical case that the frame is I-Frame, but in case that all frames are P-Frame and the like, Entry Frame may be P-Frame. What type of frame is chosen as an Entry Frame is dependent to the recorder application. In case of no I-frame contents, each macroblock should be I-macroblock at least once in 132 frames.

2.4.27 Entry Pack

This is a pack that the decoder application can start to play from. An Entry Frame and an Entry Pack match one-to-one.

ã 2001 SD Association

CONFIDENTIAL

2.4.28 Representation Picture

This is a picture that represents an object such as Program or Playlist. Represent Picture in Management Data File is a picture that represents SD-Video.

2.4.29 Resume Marker

This is information about the position where the playback was stopped last time.

2.4.30 Marker

This is information about the position, which is marked in Program. A Program can have more than one Marker.

2.4.31 Media Object Data Unit

This is a basic unit of a Media Object, in case that I-Frames exist in video stream, Media Object Data Unit should start with the I-Frame of video stream. In case that I-Frames don't exist, it depends on the recorder application how to divide the stream into Media Object Data Unit.

2.4.32 Pack

Pack is a unit of Program Stream which contains a pack header and one Packet. One Padding Packet can coexist with a Packet in a Pack.

2.4.33 stream_id

Stream_id is an 8-bit field that indicates the coding elementary stream number of the stream.

2.4.34 stream_type

Stream_type is an 8-bit field specifying the type of the stream.

ã 2001 SD Association

CONFIDENTIAL

2.5 Abbreviations

AAC	Advanced Audio Coding
AMR	Adaptive Multi Rate
A_PCK	Audio Pack
ASF	Advanced Streaming Format
ATTR	Attribute
AUD	Audio
AV	Audio Video
B-Frame	Bi-directionally predictive-coded Frame
B-VOP	Bi-directionally predictive-coded Video Object Plane
DEMUX	De-Multiplexer
DTS	Decoding Time Stamp
DVD	Digital Versatile Disc
EXT	Extra
GOV	Group of Video Object Plane
I-Frame	Intra coded Frame
I-VOP	Intra coded Video Object Plane
ID	Identifier
IEC	International Electrotechnical Commission
IF	Interface Format
INFO	Information
INT	Integer
ISO	International Organization for Standardization
JIS	Japanese Industrial Standard
JPG	JPEG (Joint Photographic coding Experts Group)
LC	Low Complexity
MGR_DATA	Management Data
MGR_INFO	Management Information
MO	Media Object
MOD	Media Object Data
MODU	Media Object Data Unit
MOI	Media Object Information
MOV	Movie
MPEG	Motion Picture Expert Group
Num	Number
OBJ	Object
OBJPOS	Object Position
P-Frame	Predictive-coded Frame
P-VOP	Predictive-coded Video Object Plane
Pb	Play back

ã 2001 SD Association

PCK	Pack
PCM	Pulse Code Modulation
PGI	Program Information
PIC	Picture
PKT	Packet
PLST_INFO	Play List Information
PLST_MGR	Play List Manager
POS	Position
P_PCK	Program stream map Pack
PRG	Program
PRG_ATTR	Program Attribute
PRG_INFO	Program Information
PRG_MGR	Program Manager
PRGSECTION	Program Section
PSM	Program Stream Map
PTS	Presentation Time Stamp
QCIF	Quarter Common Intermediate Format
SCN	Scene
SCR	System Clock Reference
SD	Secure Digital
SEP	Separate
SF1	Secure ASF
SM1	Secure MOD
SML	SMIL (Synchronized Multimedia Integration Language)
SP1	Secure JPG
SS1	Secure SML
ST1	Secure TXT
STRM	Stream
TBD	To Be defined
Tbl	table
TOC	table of contents
T_PCK	Text Pack
TSE	Time Search Entry
Tst	Time Search Table
TXT	Text
UINT	Unsigned Integer
V_PCK	Video Pack
VOP	Video Object Plane

ã 2001 SD Association

CONFIDENTIAL

3 Introduction

3.1 Total Data Structure

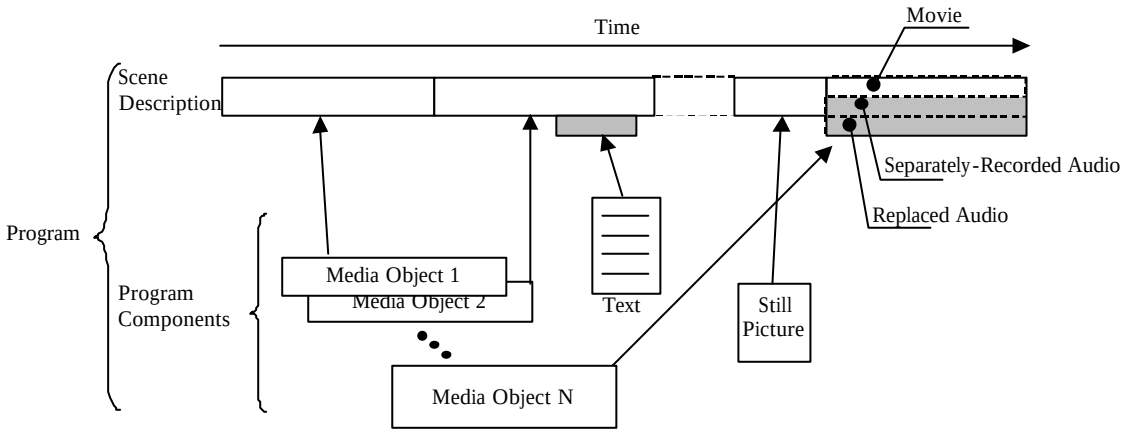
In this section, the total data structure of SD-Video contents is outlined. First of all, the important concepts on which all SD-Video contents are based are expounded; Program and Playlist.

3.1.1 Program

Program is a group of AV streams, still pictures, texts and scene description, which is intended to constitute one independent movie by the producer. The concept of Program can be easily understood in association with a TV program or a movie.

The management information of all Programs in an SD card is stored in **Program Manager File** and **Program Information Files**. Program Manager File contains summary information of all Programs including their attributes. Program Information File is assigned to each Program to contain the list of its components and attributes, which are essential for player devices to play the Program. Program Information File, components, and scene description of each Program are stored in the directory exclusively assigned to the Program (see Figure 3.1-1).

Figure 3.1-1: Structure of Program



3.1.1.1 Program Components

Program is composed by one or more Program Components. Program Components are classified into three types; Media Objects, Still Pictures, and Texts. Moreover, Program shall be composed of at least one Media Object.

ã 2001 SD Association

CONFIDENTIAL

- **Media Object** consists of video and audio streams to be played back together and their time information. Media Object shall include at least one video stream. Media Object may be divided into up to three exclusive portions to realize dynamic stream switching on its playback. The portions are Movie, Separately-Recorded Audio, and Replaced Audio, which are described below. Any stream cannot be included in two or three portions simultaneously. Media Object can contain one text stream in every portion to provide subtitles, credits, and so forth.
  - **Movie** is the mandatory portion of Media Object and shall contain all video streams and their time information composing the Media Object. Audio streams of the same Media Object may be also contained in this portion, that is, Media Object consisting of only Movie portion may be formed.
  - **Separately-Recorded Audio** is an optional portion of Media Object and shall contain audio streams with their time information. This portion shall not include any video streams. All streams in this portion shall be played back superseding all related streams (other than video streams) in Movie portion of the Media Object. Typically, this portion is formed when video signals and audio signals are recorded by separate devices.
  - **Replaced Audio** is also an optional portion of Media Object and shall contain audio streams with their time information. This portion shall not include any video streams. All streams in this portion should be played back superseding all related streams (other than video streams) in other portions (Movie and Separately-Recorded Audio) of the Media Object. Typically, this portion is created to modify or edit audio streams of the Media Object.
- **Still Picture** is a JPEG still image and may be used to insert a slide show of still images into Program. The duration of Still Picture is specified in Scene Description Data Files.
- **Text** is a plain text and may be used as credits giving the viewer some information, which are superimposed on the other components.

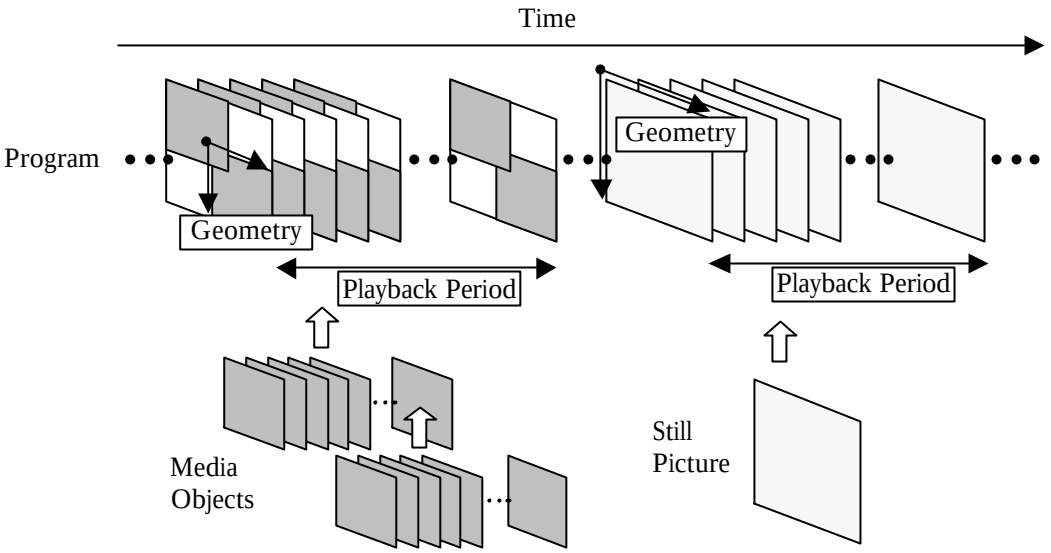
3.1.1.2 Scene Description

Scene Description tells the player devices how to playback Program by combining the Program Components. It describes the geometries (positions and sizes) and the playback periods of Program Components that compose a Program (Figure 3.1-2). There are two ways to give Scene Description to Program; one is to use **Program Information File** and the other is to make **Scene Description Data Files** that may be accepted by devices supporting Mobile Video Profile. Both files shall be placed in the directory assigned to the Program. In the latter way, Scene Description for Program may be composed of multiple Scene Description Files.

ã 2001 SD Association

CONFIDENTIAL

Figure 3.1-2: Scene Description



Program Information File holds the information of Programs and their Program Components. If no Scene Description Data File is specified for a certain Program, the player devices shall play Media Objects that are listed in Program Information File. The playback order and durations of the Media Objects are also specified in Program Information File. Therefore, by summing up the playback durations of the Media Objects in the playback order, which is the list order of Program Information File, player devices can decide the playback periods of the Media Objects.

By making Scene Description Data Files, it becomes possible to play multiple Media Objects at the same time in their own sub-frames and to utilize Still Pictures and Texts for extending the Program's power of expression. Scene Description shall be written in SMIL to describe the playback periods and the geometries of Program Components explicitly. Though it may be described in two or more Scene Description Data Files, only the file that has the name 'SCN001.SML' shall be used to provide the initial Scene Description of the Program. The others shall be referred to directly or indirectly by the initial description.

ã 2001 SD Association

CONFIDENTIAL

3.1.2 Playlist

Playlist is an ordered list of the sections of Programs. It describes only a playback order of the sections of Programs in an SD card. By making a Playlist, the viewer of SD-Video contents can change the playback sections and their order that have been decided by the producer. It is also possible for the viewer to virtually create a new movie only by combining the sections of Programs already existing in the SD card.

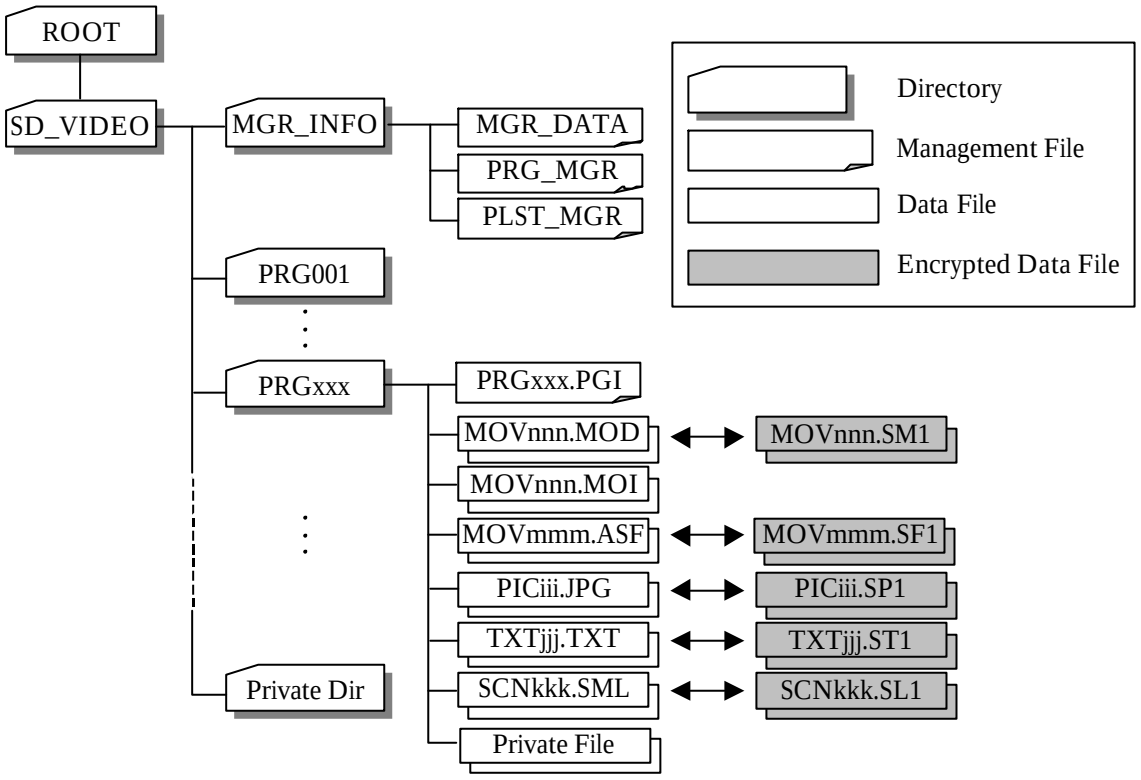
The management information of all Playlists in an SD card is stored in **Playlist Manager File**. Of course, the management information of the Programs referred by Playlists is also required on playback time.

ã 2001 SD Association

3.1.3 Organization of Directories and Files

All files of SD-Video are stored in **SD-Video Directory** whose name is 'SD_VIDEO'. The files related to global information of all Programs and Playlists on the SD card are placed in **Management Data Directory**, which is the directory with the name 'MGR_INFO' under SD-Video Directory. The files containing Program Components composing a certain Program are stored in its **Program Directory** that is exclusively assigned to the Program. The organization of directories and files is illustrated in Figure 3.1-3.

Figure 3.1-3: Organization of Directories and Files for SD-Video



In Management Data Directory, there exist Management Data File, Program Manager File, and Playlist Manager File. The role of each file can be outlined as follows.

- **Management Data File** provides the global information that is used and updated by SD-Video devices on the initial or final access to the SD card. It includes the version number of this specification, the reference to the position where the suspended playback should be resumed, and so forth. The file name shall be 'MGR_DATA'.
- **Program Manager File** provides the summary information of all the Programs recorded in SD-Video Directory. For every Program, it keeps the playback duration, the text information, the attributes and so forth. The file name shall be 'PRG_MGR'.

ã 2001 SD Association

CONFIDENTIAL

- **Playlist Manager File** provides the management information of all the Playlists composed by the sections of the Programs recorded in the SD-Video Directory. For every Playlist, it keeps the playback duration, the list of the sections of Programs that compose the Playlist, and so forth. The file name shall be 'PLST_MGR'.

The Program Components and the Scene Description of a certain Program are recorded in files stored in its Program Directory. The name of the Program Directory shall have the three-character prefix 'PRG' followed by three-digit Program Number in hexadecimal notation. The Program Number shall be exclusively assigned to each Program and used to identify the Program.

In the Program Directory, the below files that contain the management information, Program Components, and Scene Description of the Program may exist.

- **Program Information File** contains the management information of the Program, such as video and audio attributes, the list of Program Components, and so forth. The file name shall have the three-character prefix 'PRG' followed by three-digit Program Number in hexadecimal notation and the three-character extension 'PGI' preceded by a period. This file shall exist in the Program Directory of each valid Program.
- **Media Object Data File** contains video and audio streams of Media Object. The time information of the streams shall be stored in Media Object Information File, which shall coexist with the Media Object Data File. Media Object Data Files are classified into three types according to their contents; Movie Data Files, Separately-Recorded Audio Data Files, and Replaced Audio Data Files.
  - **Movie Data File** shall contain the Movie portion of Media Object, that is, it shall include all video streams of the Media Object. Its name shall have the three-character prefix 'MOV' followed by three-digit File Number in hexadecimal notation and the three-character extension 'MOD' preceded by a period. If the contents of this file are encrypted, the extension of the file name shall be changed to 'SM1'.
  - **Separately-Recorded Audio Data File** shall contain the Separately-Recorded Audio portion of Media Object. Its name shall be the same as the Movie Data File containing the Movie portion of the same Media Object except that the prefix shall be 'SEP'.
  - **Replaced Audio Data File** shall contain the Replaced Audio portion of Media Object. Its name shall be the same as the Movie Data File containing the Movie portion of the same Media Object except that the prefix shall be 'AUD'.
- **Media Object Information File** contains the time information of the streams stored in the corresponding Media Object Data File, which is used to play the streams in the Media Object Data File by player devices. This shall coexist with the corresponding Media Object Data File. Its name shall be the same as the corresponding Media Object Data File except that the extension shall be 'MOI'.

ã 2001 SD Association

CONFIDENTIAL

- **Advanced Streaming Format File** contains the data of Media Object but differs from Media Object Data File in two points. First, it is formatted in Advanced Streaming Format, which is designed by Microsoft to store synchronized multimedia data. Second, it contains both of stream data and the time information. This may be used as a substitute for a pair of Movie Data File and Media Object Information File. This shall be named in the same way as Movie Data File except that the name shall have the three-character extension 'ASF. This type of files cannot be used as containers of Media Objects that include Separately-Recorded Audio or Replaced Audio portions. If the contents are encrypted, the extension shall be changed to 'SF1'.
- **Still Picture Data File** contains the data of Still Picture. Its name shall have the three-character prefix 'PIC' followed by three-digit File Number in hexadecimal notation and the three-character extension 'JPG' preceded by a period. If the contents of this file are encrypted, the extension of the file name shall be changed to 'SP1'.
- **Text Data File** contains the data of Text. Its name shall have the three-character prefix 'TXT' followed by three-digit File Number in hexadecimal notation and the three-character extension 'TXT' preceded by a period. If the contents of this file are encrypted, the extension of the file name shall be changed to 'ST1'.
- **Scene Description Data File** contains the SMIL document that specifies Scene Description of the Program. Its name shall have the three-character prefix 'SCN' followed by three-digit File Number in hexadecimal notation and the three-character extension 'SML' preceded by a period. If the contents of this file are encrypted, the extension of the file name shall be changed to 'SL1'. Scene Description Data Files are mainly used for multiple object description and are optional for the players conforming to Mobile Video Profile.
- **Private Files** may be placed only in a Program Directory, in a manufacture-defined Private Directory under SD-Video Directory, or in SD-Video Root Directory, and shall not be placed under MGR_INFO directory. Private File name and Private Directory name must not coincide with the SD-Video Directory and File Naming Convention described in Chapter 4, including the suffix of "SXn". Where, X is any character and n is any digit.

The specification of the data file encryption will be described in CPRM for Video Specifications in the future. Until then, encrypted files shall not exist in SD-Video Directory.

ã 2001 SD Association

CONFIDENTIAL

3.1.4 Structure of Media Object Data File

Media Object Data File consists of just one Program Stream specified in ISO/IEC 13818-1. The Program Stream may be composed of one or more streams interleaved with each other. Each of them shall be one of the followings.

- Video stream
  MPEG-4 Video is available.
- Audio stream
  MPEG-2 AAC, AMR, ITU-T Rec. G.726, and Windows Media Audio are available.
- Text stream

Each stream shall be divided into one or more **Packets**. Each Packet is packed in a **Pack** and the Packs of the stream are interleaved with Packs of the other streams to compose the Program Stream. Each Packet of the same stream is identified by **stream_id** field of the Packet header (Figure 3.1-4). The stream type, i.e., coding type and media type of the stream, shall be given to player devices by **Program Stream Map** that is the special stream with stream_id=0xBC also stored in Program Stream. In Program Stream Map, the stream type of each stream is specified in the **stream_type** field connected with the elementary_stream_id field that has the same value as stream_id.

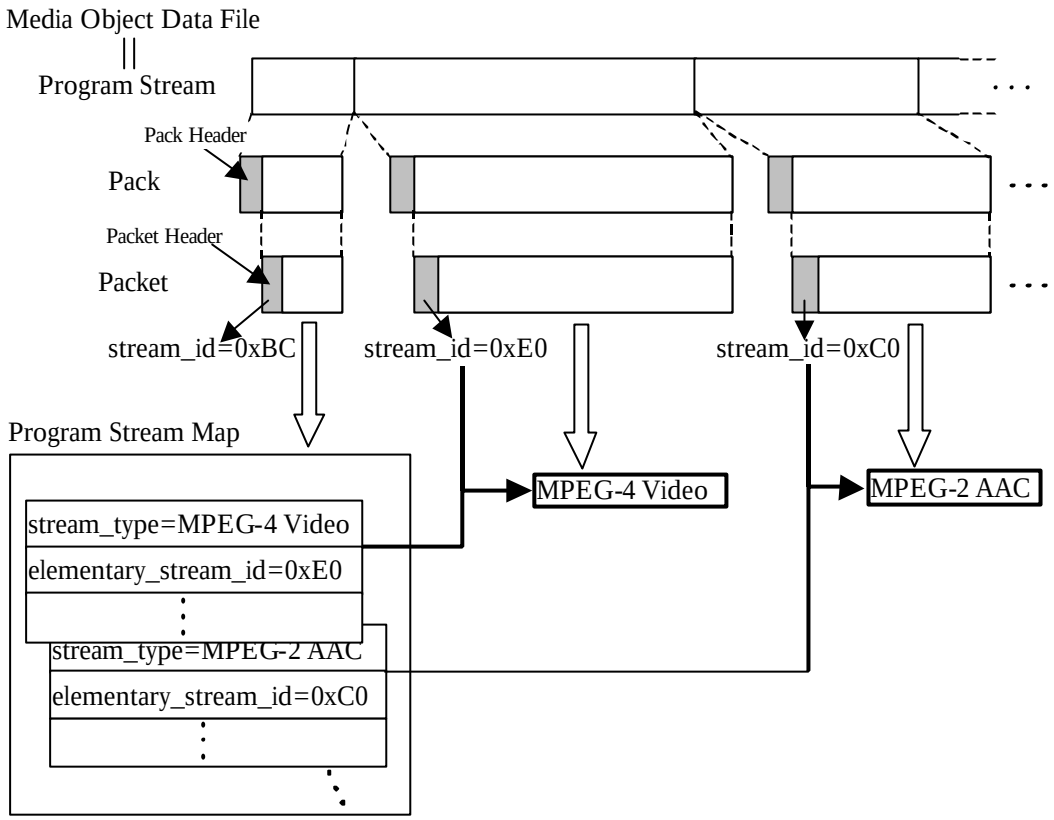
In Figure 3.1-4, the second Pack of the Program Stream has Packet of stream_id=0xE0, which shall be assigned to video streams, and the Program Stream Map indicates that the stream with stream_id=0xE0 is an MPEG-4 video stream. Thus, the player device sends the payload of the second Packet to its MPEG-4 video decoder. In the same way, the third Packet is found to be a Packet of an MPEG-2 AAC audio stream.

In this way, each stream contained in Media Object Data File is reconstituted and identified by utilizing stream_id in the packet headers and strem_type in the Program Stream Map.

ã 2001 SD Association

CONFIDENTIAL

Figure 3.1-4: Structure of Media Object Data File



3.2 Navigation Structure

In this section, the navigation data structure and mechanism for SD-Video media data is outlined.

3.2.1 Random Access Methods to Media Objects

To realize virtual editing of Programs, that is, making and editing Playlists, it is essential to provide reference methods to arbitrary sections of Media Objects. Moreover, the player devices may be required by their users to provide various non-sequential play functions; such as skipping to some section of Program, fast forward/reverse play, and so on.

In this section, random access methods to Media Objects that enables the SD-Video devices to access arbitrary sections of Programs according to their presentation times is expounded.

3.2.1.1 Reference to Program Sections and Accessing Corresponding Program Components

Program may be composed of multiple Program Components according to Scene Descriptions. Therefore, to access a section of Program, it is necessary to find the corresponding Program Components whose beginning or ending time is within the Program Section.

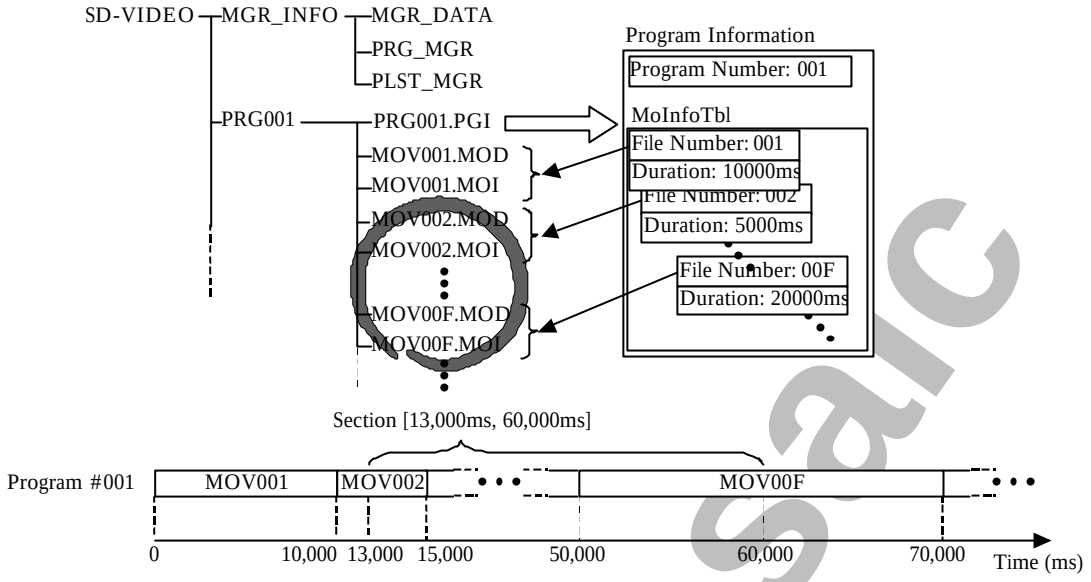
In case Scene Descriptions of Program are defined in **Scene Description Data Files**, the playback periods of the Program Components are explicitly described in SMIL. The Scene Description Data File specifies when the playback of each Program Component starts and how long the playback continues. Once one decides which Scene Description Data File is used to playback the Program, it is easy to find the Program Components corresponding to the designated Program Section.

Without Scene Description Data Files, only Media Objects can compose a Program. The playback durations and orders of the Media Objects are specified in Program Information in **Program Information File**. Program Information specifies the File Numbers and the playback durations of Media Objects that belong to the corresponding Program. By summing the playback durations of Media Objects in order of their appearance in the Program Information, player devices can decide which Media Objects correspond to the Program Section.

ã 2001 SD Association

CONFIDENTIAL

Figure 3.2-1: Mapping Program Section to Set of Media Objects



An example procedure mapping Program Section to the set of the corresponding Media Objects is shown in Figure 3.2-1. In Figure 3.2-1, the designated Program Section ranges from 13,000 ms to 60,000 ms in Program '001'. As Media Object '002' is played back in the period between 10,000 ms and 15,000 ms in Program '001', it is found to be the first one in the Program Section. Similarly, Media Object '00F' is found to be the last one because its playback period ranges from 50,000 ms to 70,000 ms. Finally, the Program Section is mapped to the set of Media Objects '002' to '00F'.

3.2.1.2 Time Search Table

By the previously mentioned mechanism, SD-Video devices are now able to obtain the Program Components corresponding to a Program Section. Accessing frames at any time within the playback periods of Still Pictures or Texts requires no additional procedures because all frames in the playback periods of them are the same. On the other hand, for Media Objects, an additional mechanism to access the data of a certain frame during the playback period of Media Object is required. In this section, Time Search Table is introduced as a data structure to realize random access to a certain frame data in Media Object Data File.

Time Search Table is stored in **Media Object Information File** of the associated Media Object Data File. It consists of Time Search Entries, each of which associates the time offset of a frame with the byte offset of its coded data from the beginning of the Media Object Data File. They are arranged in the sequence of frames at regular intervals that are specified in Media Object Information File. Player devices find the byte offset of the target frame data by consulting the Time Search Entry nearest to the target frame. That is, player devices can directly access only the frames where the Time Search Entries have been arranged.

Alphamosaic

There are three types of Time Search Table, one of which is selected according to the characteristics of the contents of the Media Object Data File. It is also permitted to prepare no Time Search Table for them. These types are the followings.

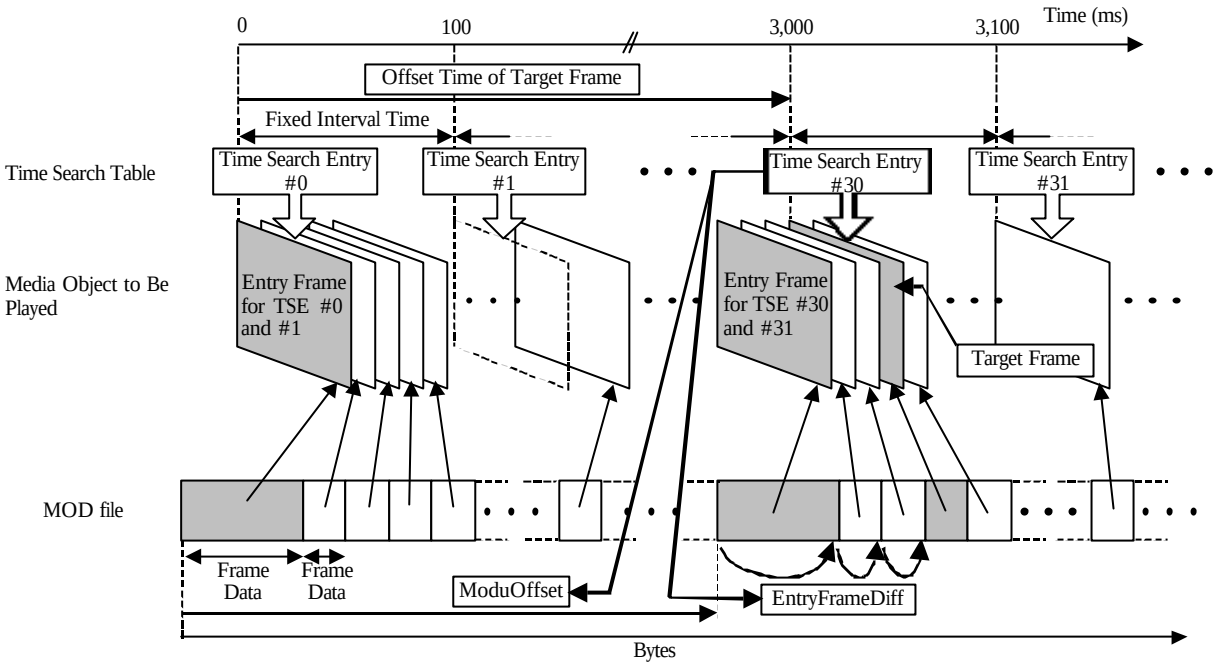
- **Type 1 (Time Search Tables with MODU Information)**
  This type may be used to enable player devices to access certain frame data in Media Object Data Files containing variable-size frame data. This type of Time Search Tables has references to MODUs described in the next section, which are utilized for skipping predictive-coded frames such as B-VOPs or P-VOPs in MPEG-4 video streams to realize fast forward/reverse play. Therefore, it is effective to use this type of Time Search Tables for Media Object Data Files containing MPEG-4 video streams including I-VOPs.
- **Type 2 (Time Search Tables without MODU Information)**
  This type is almost the same as the previous type except that it has no references to MODUs. Though these Time Search Tables may be also used for Media Object Data Files containing variable-size frame data, it is especially effective to use them in case where the first frames of MODUs have no special meanings. In audio streams with variable-data-size frames, every frame is coded independently from other frames. On the other hand, MPEG-4 video streams may be composed of only predictive-coded frames (P-VOPs). These are the typical cases where there is no need to distinguish the data of intra-coded frames from those of predictive-coded frames, and consequently also the typical cases where the effectiveness of this type of Time Search Tables is shown.
- **Type 3 (Time Search Tables for fixed Pack size Media Object Data Files)**
  This type may be used for Media Object Data Files where stream data are divided into the fixed size Packs. Such Media Object Data Files are usually composed of audio streams whose frames have the same size. There is no need of Time Search Entries that have the byte offsets of frame data from the beginning of the Media Object Data File. The byte offset of the target frame data is calculated from the fixed interval time between adjacent frames, the fixed Pack size, and the fixed number of frames in a Pack.

The former two types require the same procedure to obtain the target frames. In case of Figure 3.2-1, Program Section ranging from 13,000 ms to 60,000 ms in Program ‘001’ begins at 3,000 ms from the beginning of Media Object ‘002’. To access the first frame of the Program Section (the target frame in this example), player devices should follow the following procedure (see Figure 3.2-2).

ã 2001 SD Association

CONFIDENTIAL

Figure 3.2-2: Procedure to Obtain Target Frame Data by Time Search Table of Type 1 or 2



1. Player devices search the Time Search Table of the Media Object Data File for the Time Search Entry that points to the nearest time to the playback time of the target frame. The index of the Time Search Entry, ***n***, is obtained by the following formula,

$$n = \text{round}\left(\frac{OffsetTime}{\text{TstInterva l}}\right)$$

   where **TstInterval** is the intervals of Time Search Entries and ***OffsetTime*** is the time offset of the target frame. The function round(x) rounds off the argument x to the nearest integer. **TstInterval** shall be specified in Media Object Information File. In this example, ***n***=30 is obtained using **TstInterval**=100 ms and ***OffsetTime***=3,000 ms.
2. The ***n***-th Time Search Entry provides the byte offset of its Entry Frame data, which is stored in the **ModuOffset** field. The Entry Frame is usually the intra-coded frame before and nearest to the target frame, and used to decode the succeeding predictive-coded frames including the target when intra-coded and predictive-coded frames coexist in the Media Object Data File. In other cases, the Entry Frame may be identical with the target frame. By skipping data to the byte offset, player devices can obtain the data of the Entry Frame.
3. Finally, after decoding and skipping some frames from the Entry Frame (if the target frame is not the Entry Frame), the target frame is obtained. The number of frames skipped to the target frame is specified in **EntryFrameDiff** field of the Time Search Entry.

ã 2001 SD Association

CONFIDENTIAL

By the other type of Time Search Table, Type 3, player devices obtain the byte offset of the target frame data using the fixed parameters commonly used in the Media Object Data File. The procedure is as follows (Figure 3.2-3).

1. First, the number of frames before the target frame, ***OffsetFrame***, is obtained by dividing the time offset of the target frame ***OffsetTime*** (3,000 ms in Figure 3.2-3) by the fixed playback duration of one frame, which is given at **FrameTimeNmr** field and **FrameTimeDnm** field in Media Object Information File.

$$OffsetFrame = \text{round}\left(\frac{OffsetTime * \text{FrameTimeD nm}}{\text{FrameTimeN mr}}\right)$$

2. The byte offset of the Pack including the target frame data, ***OffsetBytes***, is calculated with the number of skipped Packs, the Pack size, and the number of frames coded in a Pack, which are given in **NumSkippedPack**, **PackSize**, and **NumFrame** fields respectively in Media Object Information File, in addition to ***OffsetFrame*** previously obtained.

$$OffsetBytes = \text{PackSize} \times \left(\text{NumSkipped Pack} + \text{floor}\left(\frac{OffsetFrame}{\text{NumFrame}}\right)\right)$$

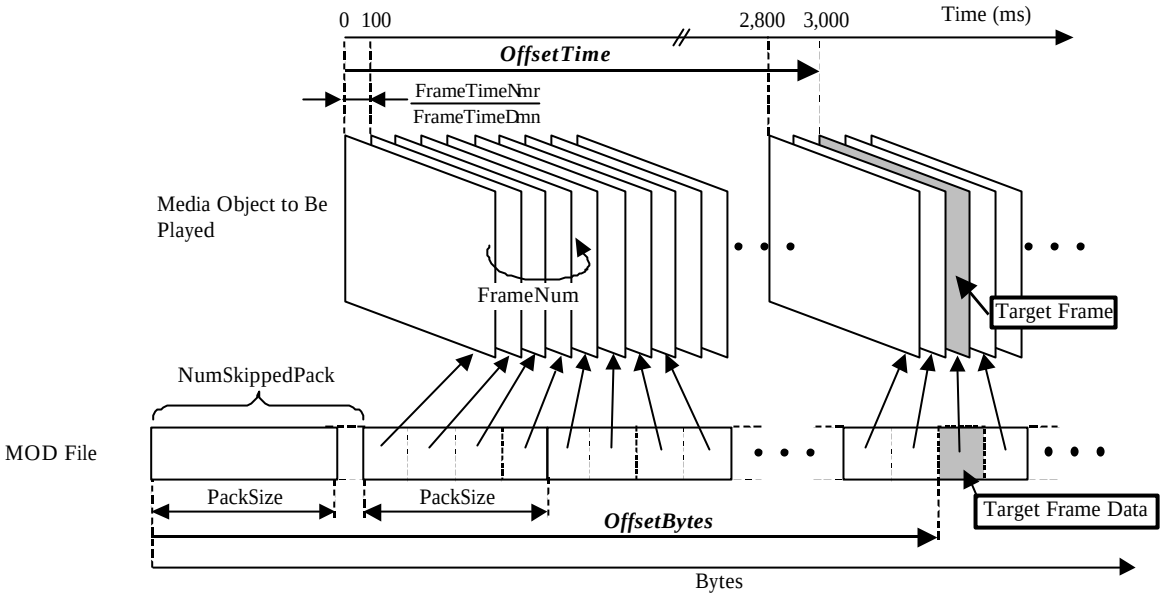
   The function floor(x) rounds off the argument x to the nearest integer that is not larger than x.
3. Finally, the byte offset of the target frame data from the beginning of the Pack is calculated with the playback duration of a frame, the number of frames in a Pack, and the Pack size.

This example is made on the condition that four frames are packed into one Pack, which is allowed only for audio frames. As this type of Time Search Tables is adopted only for Media Object Data Files composed of only audio streams, the above procedure is commonly used for Time Search Tables of Type 3.

ã 2001 SD Association

CONFIDENTIAL

Figure 3.2-3: Procedure to Obtain Target Frame Data by Time Search Table of Type 3



3.2.1.3 MODU Information

To realize fast forward/reverse play of Media Objects, it is essential for player devices to know the next frame to be played without processing the intermediate frames to be skipped. Nevertheless, it is difficult to know the distance between two distant frames in byte streams without special information because of the following factors.

- Media Objects are usually composed of frames with variable data sizes. Therefore, it is impossible to guess the total data size of the frames between the current frame and the next frame to be played.
- Media Objects may contain predictive-coded frames. As these frames shall be decoded with their reference frames, they are not suitable for the frames jumped into in fast play.

A solution to the above problem is to prepare the list of the intra-coded frames to be played back in fast forward/reverse play. First, AV streams in Media Object Data File are logically divided into MODUs (Media Object Data Units) that are the basic units starting with the data of an intra-coded frame. That is, MODUs correspond to GOVs of MPEG-4 Video. By playing the first frames of the MODUs and skipping the other frames according to the data size and the playback duration of each MODU, player devices realize fast forward/reverse play of the Media Object.

Moreover, on the condition that every macroblock should be refreshed before it is coded 132 times as predictive macroblocks, MPEG-4 Video streams without intra-coded frames can be also divided into MODUs and, of course, their starting frames can be predictive-coded frames (P-VOPs). In this case, though the video frames may be corrupted for a while at the start of playback, it will get better and better gradually. However, fast forward/reverse play of this type of video streams is not available yet.

ã 2001 SD Association

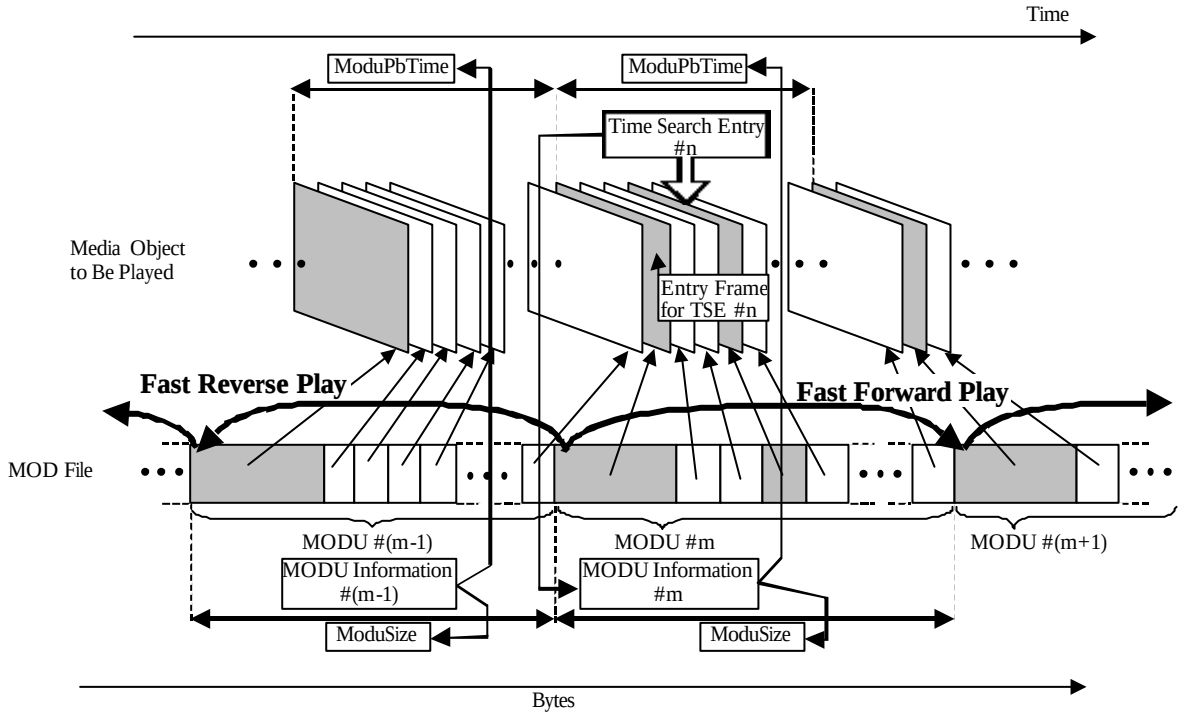
CONFIDENTIAL

In other cases, MODUs are formed to describe the locations of Time Search Entries of Type 2. The location of each Time Search Entry is described with the byte offset of the Entry Pack of the nearest MODU and the number of frames between the Entry Frame of the MODU and the Time Search Entry. These MODUs are used on creating Time Search Table and the MODU Information is not preserved anywhere.

The data size and the playback duration of each MODU are stored in MODU Information, which is stored with Time Search Table of Type1 in Media Object Information File. MODU Information consists of the following fields; **EntrySize** (the data size of the first intra-coded frame), **ModuPbTime** (the total playback duration of the concerned MODU), and **ModuSize** (the total data size of the concerned MODU). MODU Informations are stored in tabular form in playback order. By tracing the MODU Informations sequentially forward/backward, player devices can realize fast forward/reverse play (Figure 3.2-4). The speed of fast play can be controlled to some extent by changing the number of MODUs to be played back according to ModuPbTime field.

Additionally, in case that fast forward/reverse play begins at an intermediate frame of Media Object Data File using Time Search Table, every Time Search Entry may have the reference to the MODU, the first frame of which is the Entry Frame of the Time Search Entry (in fact, the reference to the corresponding MODU Information is stored here). This type of Time Search Table is described as Type 1 in the previous section.

Figure 3.2-4: Fast Forward/Reverse Play Using MODU Information



3.2.2 Resuming Last Playback

Even if the user of a player device stops the playback of some Program or Playlist and then turns off the device, the device and the other devices can resume the last playback at the same position where the user has stopped. The position may be preserved in **ResumeMarker** field in **Management Data File** and be designated by Object ID of the Program or Playlist and the time offset from the beginning of the Program or Playlist in milliseconds. If the devices do not support resuming function, they shall set ResumeMarker field to zero at the time when they are turned off. Thus, whenever player devices are turned off, ResumeMarker shall be updated with the correct Resume Marker or zero.

The way to start playing the Program or Playlist at the position designated by Resume Marker is described in Section 3.2.1.

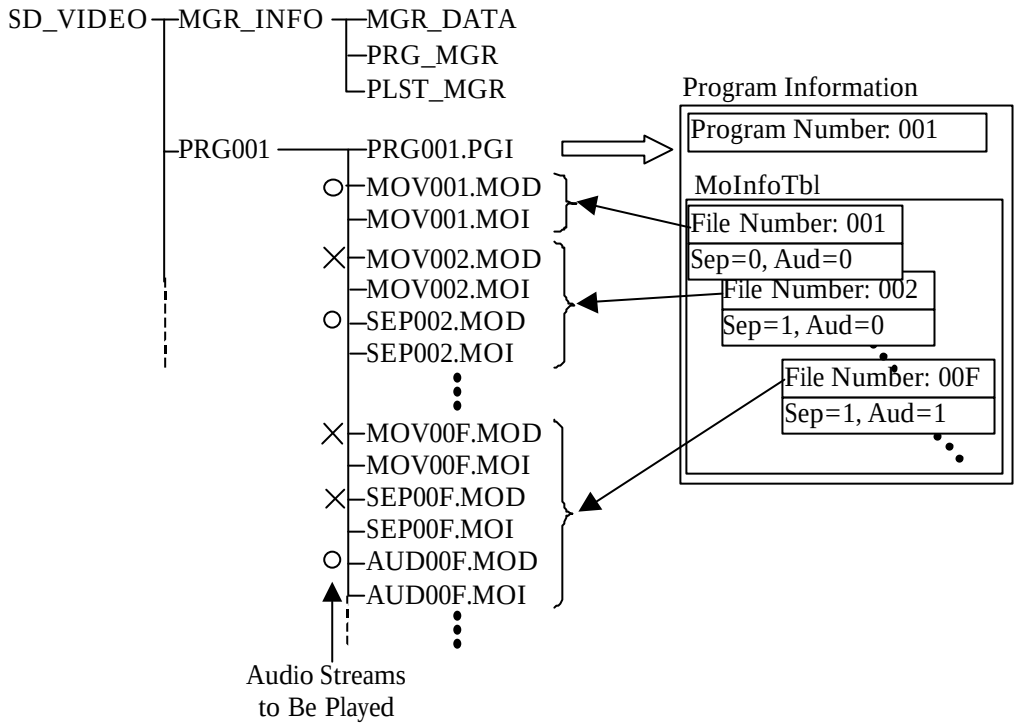
ã 2001 SD Association

CONFIDENTIAL

3.2.3 Separately-Recorded Audio and Replaced Audio

In this Section, the way to switch the audio streams among Movie, Separately-Recorded Audio, and Replaced Audio portions of Media Object is described. The Separately-Recorded Audio Data File and Replaced Audio Data File are stored in the same directory and have the same File Number as the Movie Data File containing the video streams of the same Media Object.

Figure 3.2-5: Separately-Recorded Audios and Replaced Audios



In case that no Scene Description Data Files exist, it is also required to set the flags, named **Sep** and **Aud**, telling player devices the existence of Separately-Recorded Audio and Replaced Audio portion in the concerning Media Object. Flags Sep and Aud is stored in **MoAttr** field of the Media Object's entry of **MoInfoTbl** that is a part of Program Information (Figure 3.2-5). MoInfoTbl has one or more entries each of which describes File Number, the playback duration, and the attributes of a certain Media Object composing the Program associated with the Program Information. Only one Separately-Recorded Audio and one Replaced Audio portion can be included in Media Object. The audio streams of Separately-Recorded Audio portion supersede the audio streams of Movie portion of the same Media Object. Moreover, the audio streams of Replaced Audio portion supersede any other audio streams of the Media Object including those of Separately-Recorded Audio portion. Namely, even if both of Separately-Recorded Audio and Replaced Audio are included in a certain Media Object, it is not the Separately-Recorded Audio but the Replaced Audio to be played back. In case of the example in Figure 3.2-5, the audio streams in Media Object Data Files preceded by a circle shall be played and the others preceded by a cross shall not.

ã 2001 SD Association

CONFIDENTIAL

On the other hand, that is, in case of using Scene Description Data Files instead of Program Information to describe how to play Program, it shall be specified in the Scene Description Data Files which audio streams should be played with the video stream or the other Program Components; audio streams in Movie Data File, Separately-Recorded Audio Data File, or Replaced Audio Data File. In these cases, in fact, there is no distinction between Separately-Recorded Audio Data File and Replaced Audio Data File in the point that both of them are Media Object Data Files containing a set of audio streams of Media Object. The only difference between them exists in the first three characters of their file names.

ã 2001 SD Association

CONFIDENTIAL

3.2.4 Subtitles, Credits, and Other Texts Synchronized with Video and Audio Streams

Text data that are used for subtitles, credits, and so forth can be included in Media Objects to be played back synchronously with video and audio streams. Text stream is interleaved with the other streams in the Media Object Data File and interpreted in the coding system specified in **StrmAttr** field of **Media Object Information File** associated with the Media Object Data File. The playback periods of Text stream is decided in the same way as the other streams contained in the Media Object Data File, that is, by using time stamps stored in the headers of Packets into which all of them are divided when being written into Media Object Data File.

ã 2001 SD Association

CONFIDENTIAL

4 Directory and File Naming Convention

This chapter describes the naming convention of directories and files in SD-Video.

4.1 SD-Video Directory

A directory with the name “SD_VIDEO” directly under the root directory is called SD-Video Directory. All data files managed by SD-Video are created under this directory.

ã 2001 SD Association

4.2 Management Data Directory

A directory with the name “MGR_INFO” directly under SD-Video Directory is called Management Data Directory. This directory contains Management Data File (MGR_DATA), Program Manager File (PRG_MGR) and Playlist Manager File (PLST_MGR).

ã 2001 SD Association

CONFIDENTIAL

4.3 Program Directory

Directories that contain Program Files are called Program Directory. They are created directly under SD-Video Directory. The naming convention of Program Directory is as follows.

- The directory name shall be 6 characters in length.
- The first three characters shall be “PRG”.
- The last three characters shall be a hexadecimal number between “001” and “FFF”. The number “000” shall not be used.
- Extension of file name shall not be used.

The specification of the naming convention of Program Directory are shown in Table 4.3-1.

Table 4.3-1 : Specification of naming convention of Program Directory2

	Fixed string part			Hexadecimal number part (Program Number)		
Specification	P	R	G	0	0	1
				~		
				F	F	F
Example	P	R	G	0	1	F

ã 2001 SD Association

CONFIDENTIAL

4.4 Program Files

The following Program Files are created under Program Directory, and constitute a Program.

Table 4.4-1 : Program Files under Program Directory

File	Contents
Program Information File	Information of the Program
Media Object Data File	AV Data of Media Object
Media Object Information File	Time information of Media Object Data File
Advanced Streaming Format File	Option file format of Media Object
Still Picture Data File	JPEG
Text Data File	Plain text
Scene Description Data File	SMIL

4.4.1 Program Information File

Program Information File contains the management information of the Program, such as video and audio attributes, the list of Program Components, and so forth. The naming convention of Program Information File is as follows.

- The file name shall be 6 characters as base name plus 4 characters as extension (including period).
- The first three characters shall be “PRG”.
- The next three characters shall be a hexadecimal number between “001” and “FFF”. The number “000” shall not be used. This number shall be equal to Program Number.
- The extension of the file name shall be “.PGI”.

The specification of the naming convention of Program Information File are shown in Table 4.4-2.

Table 4.4-2 : Specification of naming convention of Program Information File

	Alternative string part			Hexadecimal number part (File Number)			Alternative string part			
Specification	P	R	G	0 ~ F	0 ~ F	1 ~ F	.	P	G	I
Example	P	R	G	0	1	F	.	P	G	I

4.4.2 Media Object Data File

Media Object Data File contains presentation data such as video, audio and so forth. The naming convention of Media Object Data File is as follows.

- The file name shall be 6 characters as base name plus 4 characters as extension (including period).
- The first three characters shall be one of “MOV”, “SEP” and “AUD”. “MOV” is used for Movie, “SEP” is used for Separately-Recorded Audio, and “AUD” is used for Replaced Audio. “SEP” file and “AUD” file shall be attached to “MOV” file with the same File Number.
- The next three characters shall be a hexadecimal number between “001” and “FFF”. The number “000” shall not be used.
- The extension of the file name shall be “.MOD”, but if the file is encrypted it shall be “.SM1”.

The specification of the naming convention of Media Object Data File are shown in Table 4.4-3.

Table 4.4-3 : Specification of naming convention of Media Object Data File

	Alternative string part			Hexadecimal number part (File Number)			Alternative string part			
Specification	M	O	V	0	0	1	.	M	O	D
	or			≀			or			
	S	E	P							
	or									
	A	U	D	F	F	F	.	S	M	1
Example	M	O	V	0	1	F	.	M	O	D

4.4.3 Media Object Information File

Media Object Information File contains the time information of Media Object Data File that has the same hexadecimal number. The naming convention of Media Object Information File is as follows.

- The file name shall be 6 characters as base name plus 4 characters as extension (including period).
- The first three characters shall be one of “MOV”, “SEP” and “AUD”. “MOV” is used for Movie, “SEP” is used for Separately-Recorded Audio, and “AUD” is used for Replaced Audio. “SEP” file and “AUD” file shall be attached to “MOV” file with the same File Number.
- The next three characters shall be a hexadecimal number between “001” and “FFF”. The number “000” shall not be used.
- The extension of the file name shall be “.MOI”.

The specification of the naming convention of Media Object Information File are shown in Table 4.4-4.

ã 2001 SD Association

CONFIDENTIAL

Table 4.4-4 : Specification of naming convention of Media Object Information File

	Alternative string part			Hexadecimal number part (File Number)			Fixed string part			
Specification	M	O	V	0	0	1	.	M	O	I
	or			~						
	S	E	P							
	or									
	A	U	D	F	F	F				
Example	M	O	V	0	1	F	.	M	O	I

4.4.4 Advanced Streaming Format File

Advanced Streaming Format File is designed to store synchronized multimedia data. The naming convention of Advanced Streaming Format File is as follows.

- The file name shall be 6 characters as base name plus 4 characters as extension (including period).
- The first three characters shall be "MOV". Separately-Recorded Audio and Replaced Audio are not supported in case of Advanced Streaming Format.
- The next three characters shall be a hexadecimal number between "001" and "FFF". The number "000" shall not be used. The number space is shared with Media Object Data File, and the File Numbers of any Advanced Streaming Format File shall not be equal to the File Number of any Media Object Data File.
- The extension of the file name shall be ".ASF", but if the file is encrypted it shall be ".SF1".

The specification of the naming convention of Advanced Streaming Format File are shown in Table 4.4-5.

Table 4.4-5 : Specification of naming convention of Advanced Streaming Format File

	Alternative string part			Hexadecimal number part (File Number)			Alternative string part			
Specification	M	O	V	0	0	1	.	A	S	F
				~			or			
				F	F	F	.	S	F	1
Example	M	O	V	0	1	F	.	A	S	F

ã 2001 SD Association

CONFIDENTIAL

4.4.5 Still Picture Data File

Still Picture Data File contains still image, and is referred only from Scene Description Data File. The naming convention of Still Picture Data File is as follows.

- The file name shall be 6 characters as base name plus 4 characters as extension (including period).
- The first three characters shall be “PIC”.
- The next three characters shall be a hexadecimal number between “001” and “FFF”. The number “000” shall not be used.
- The extension of the file name shall be “.JPG”, and if the file is encrypted it shall be “.SP1”.

The specification of the naming convention of Still Picture Data File are shown in Table 4.4-6.

Table 4.4-6 : Specification of naming convention of Still Picture Data File

	Fixed string part			Hexadecimal number part (File Number)			Alternative string part			
Specification	P	I	C	0	0	1	.	J	P	G
				২			or			
				F	F	F	.	S	P	1
Example	P	I	C	0	1	F	.	J	P	G

4.4.6 Text Data File

Text Data File contains plain text, and is referred only from Scene Description Data File. The naming convention of Text Data File is as follows.

- The file name shall be 6 characters as base name plus 4 characters as extension (including period).
- The first three characters shall be “TXT”.
- The next three characters shall be a hexadecimal number between “001” and “FFF”. The number “000” shall not be used.
- The extension of the file name shall be “.TXT”, and if the file is encrypted it shall be “.ST1”.

The specification of the naming convention of Text Data File are shown in Table 4.4-7.

Table 4.4-7 : Specification of naming convention of Text Data File

	Fixed string part			Hexadecimal number part (File Number)			Alternative string part			
Specification	T	X	T	0	0	1	.	T	X	T
				২			or			
				F	F	F	.	S	T	1
Example	T	X	T	0	1	F	.	T	X	T

ã 2001 SD Association

CONFIDENTIAL

4.4.7 Scene Description Data File

Scene Description Data File contains a scene description for playback of Media Objects. The naming convention of Scene Description Data File is as follows.

- The file name shall be 6 characters as base name plus 4 characters as extension (including period).
- The first three characters shall be “SCN”.
- The next three characters shall be a hexadecimal number between “001” and “FFF”. The number “000” shall not be used.
- The extension of the file name shall be “.SML”, and if the file is encrypted it shall be “.SL1”.

The specification of the naming convention of Scene Description Data File are shown in Table 4.4-8.

Table 4.4-8 : Specification of naming convention of Scene Description Data File

	Fixed string part			Hexadecimal number part (File Number)			Alternative string part			
Specification	S	C	N	0	0	1	.	S	M	L
				~			or			
				F	F	F	.	S	L	1
Example	S	C	N	0	1	F	.	S	M	L

4.5 Method for generating file name with hexadecimal number

The names of Program Directory and Program Files have hexadecimal number part. The number of the part is assigned sequentially according to each file type when new file/directory is created. For example, MOV001.MOD and PIC001.JPG can coexist in the same Program Directory, but MOV001.MOD and MOV001.SM1 cannot coexist in the same Program Directory.

An example of the number assignment rules is as follows.

(1) Find the maximum file number / directory number of the same type within the directory in which new file/directory is created.
(2) Create a file/directory with the number greater than the maximum number by one.
(3) If the number has reached the maximum limit, creating new file/directory is failed. Or it may search for deleted number for the reuse.

ã 2001 SD Association

CONFIDENTIAL

This page is intentionally left blank

ã 2001 SD Association

CONFIDENTIAL

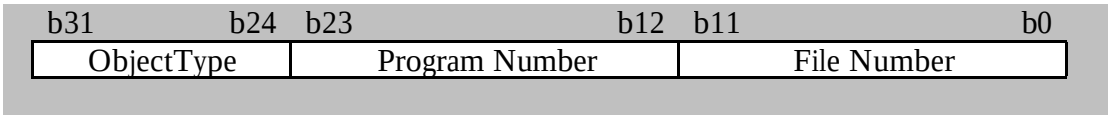
5 Navigation Data Structure

Navigation Data is used to playback AV streams, and it consists of Program Manager, Program Information, Playlist Manager and Media Object Information.

5.1 Miscellaneous definitions

5.1.1 Object ID

Navigation Data Structure contains the field, Object ID. Object ID is used for identification of objects such as Program, Playlist, Media Object. Object ID is a 32-bit value, which concerns with its own file/directory name. The format of Object ID is as follows.



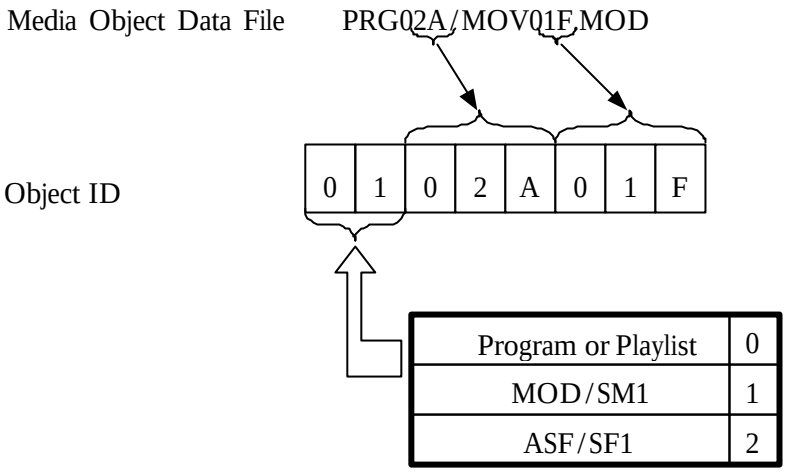
ObjectType	0: Program or Playlist 1: Media Object Data File 2: Advanced Streaming Format File Others: Reserved
Program Number	Hexadecimal number of the Program that the object belongs to. If the object is Program, it is its own number, and if the object is Playlist, it is the fixed number, '000h'.
File Number	Hexadecimal file number of Media Object, and so forth. If the object is Program, it is the fixed number, 000h. If the object is Playlist, it is its own number of Playlist.

An example of the correspondence between an object and its Object ID is shown in Figure 5.1-1.

ã 2001 SD Association

CONFIDENTIAL

Figure 5.1-1 : Example of Object ID



5.1.2 Data Types

5.1.2.1 UINT8

The UINT8 type specifies an unsigned 8-bit data item.

5.1.2.2 UINT16

The UINT16 type specifies an unsigned 16-bit data item.

5.1.2.3 UINT32

The UINT32 type specifies an unsigned 32-bit data item.

5.1.2.4 OBJECTID

The OBJECTID specifies an unsigned 32-bit value that maintains Object ID.

5.1.2.5 OBJPOS

The OBJPOS is Object Position Structure that specifies a position in an Object.

ã 2001 SD Association

CONFIDENTIAL

Table 5.1-1 : OBJPOS

Field Type	Field Name	Contents	Field Size (Byte)
OBJECTID	ObjID	Object ID	4
UINT32	OffsetPos	Time offset in millisecond	4
Total			8

ObjID

Describes Object ID of the object that is specified the position by this structure.

OffsetPos

Describes time offset in the object in millisecond.

ã 2001 SD Association

5.2 Considerations

5.2.1 Reserved Fields

There are some reserved fields in the data structures explained later. These fields are for future extension, and shall have a value of zero.

5.2.2 Version

There are also some version fields in the data structures explained later, which are MGR_DATA, PRG_MGR, PRG_ATTR, PRG_INFO, PLST_MGR and PLST_INFO. Version fields have major version part and minor version part. Major version part indicates the major version of its data structure, which is increased when the structure cannot keep the compatibility with the older version. Minor version part indicates the minor version of its data structure, which is increased when the structure can keep the compatibility with the older version. In the first version of SD-Video, major version shall be 1 and minor version shall be 0.
Player shall check the version to know whether to be able to access the data structure. If the major version of the player is older than the major version of the data, the player shall stop accessing to the data, and if the major version of the player is newer or equal to the major version of the data, the player shall be able to access the data.

5.2.3 DataSize

DataSize fields indicate the size of structure that they are contained. But in changing the data structure in the future, there is a possibility of increasing the size of the data structure and the value of DataSize field, and placing the new field at newly created area. This is for keeping the compatibility with older version, older player does not know new field but can access the data except it.
For the reasons mentioned above, Recorder or Player shall access the data considering that the value of DataSize field may be not equal to the value calculated by this specification.

ã 2001 SD Association

CONFIDENTIAL

5.3 Management Data File

Management Data File defines global characteristics of SD-Video management data, and its file name shall be “MGR_DATA”. This file shall not include any other data.

Table 5.3-1 : MGR_DATA

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	DataType	Data type identifier	2
UINT32	DataSize	Size of Data	4
UINT16	Version	Version number of SD-Video	2
OBJPOS	ResumeMarker	Marker position for resumed playback	8
UINT8	Reserved[36]	Reserved	36
UINT16	TextInfoAttr1	Attribute of text information 1	2
UINT16	TextInfoAttr2	Attribute of text information 2	2
UINT8	TextInfo[200]	Text information for SD-Video	200
Total			256

DataType

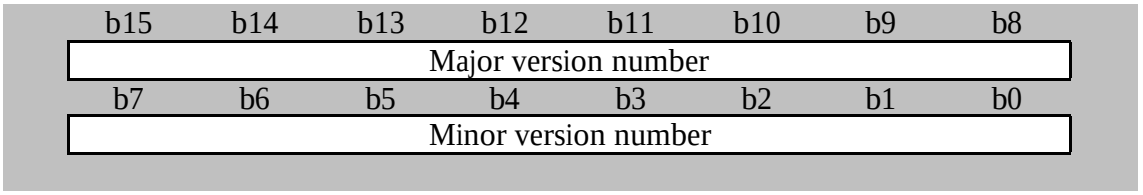
This field shall be “V1” with character set code of ISO/IEC 646:1983(ASCII).

DataSize

Describes the size of MGR_DATA. The value of this field shall be 256.

Version

Describes the version number of MGR_DATA.



In the first version, major version number shall be 1 and minor shall be 0.

ResumeMarker

Describe the information about the position where the playback has been stopped last time. Object ID field of ResumeMarker specifies Program or Playlist, and OffsetPos field specifies the offset time in millisecond where the playback has been stopped. The Object specified by Object ID field shall exist. OffsetPos shall be greater than or equal to zero and less than the playback duration of specified Program or Playlist, too. In case not supported, fill with all zeros.

TextInfoAttr1

Describe the text attribute of the first sub field of TextInfo in the MGR_DATA. The value is as follows. In case not supported, the value of this field should be 0000h.

0001h	ISO/IEC 646:1983(ASCII)
0002h	JIS X 0201
0003h	ISO 8859-1
Others	Reserved

ã 2001 SD Association

CONFIDENTIAL

TextInfoAttr2

Describe the text attribute of the second sub field of TextInfo in the MGR_DATA. The value is as follows. In case not supported, the value of this field should be 0000h.

0001h	Shift JIS Kanji
Others	Reserved

TextInfo

Describe the text information of SD Memory Card for SD-Video. This field is divided into two sub fields by termination code '00h'. The first sub field is used for the text information of SD Memory Card in the character set specified by TextInfoAttr1. The second sub field is coded in another character set specified by TextInfoAttr2. The part after the second field is filled with '00h'. In other words, the maximum length of the sum of both sub fields is 198 bytes because at least two termination codes are needed.

If the both text information do not exist, this field shall be filled with '00h'. If only the first field exists, the part after the first sub field shall be filled with '00h'. If only the second field exists, the first byte shall be '00h' and the second field follows after.

ã 2001 SD Association

5.4 Program Manager File

Program Manager File contains Program Manager, and its file name shall be “PRG_MGR”. This file shall not include any other data.

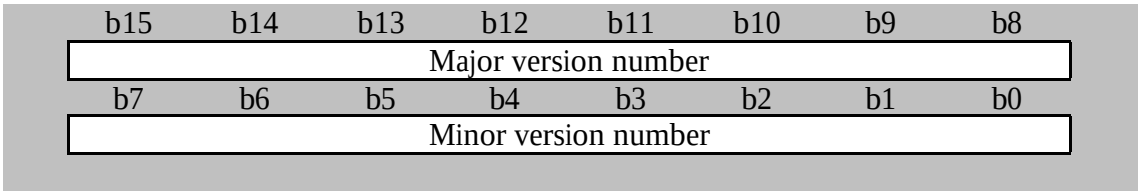
5.4.1 Program Manager (PRG_MGR)

Program Manager defines characteristics of all Programs in SD-Video.

Table 5.4-1 : PRG_MGR

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	DataType	Data type identifier	2
UINT32	DataSize	Size of Data	4
UINT16	Version	Version number of PRG_MGR	2
UINT32	PlaybackDuration	Total playback duration of all Programs in millisecond	4
UINT8	Reserved[26]	Reserved	26
UINT16	NumPrg	Number of Program	2
PRG_ATTR	PrgAttrTbl[NumPrg]	Table of Program Attribute	280 * NumPrg
Total			Variable

DataType
This field shall be “V2” with character set code of ISO/IEC 646:1983(ASCII).
DataSize
Describes the size of PRG_MGR.
Version
Describes the version number of PRG_MGR.



In the first version, major version number shall be 1 and minor version shall be 0.
PlaybackDuration
Describes total playback duration of all Programs in millisecond. The value shall be equal to the sum of each PlaybackDuration field of PRG_ATTR.
NumPrg
Describes the number of Programs, which is equal to the number of Program Attribute entries. The maximum value of NumPrg is 99.
PrgAttrTbl
Describes the Program Attribute of each Program in tabular form. The size of Program Attribute shall be 280 bytes. All Programs that are defined by Program Attributes shall exist in SD

ã 2001 SD Association

CONFIDENTIAL

Memory Card, and no Program that is not defined by Program Attributes shall exist.

ã 2001 SD Association

5.4.2 Program Attribute (PRG_ATTR)

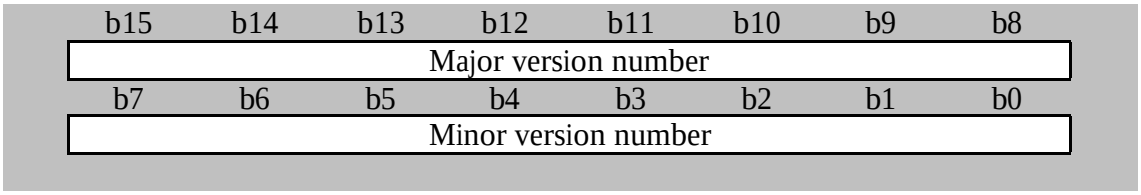
Program Attribute defines characteristics of a Program.

Table 5.4-2 : PRG_ATTR

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	Version	Version number of PRG_ATTR	2
OBJECTID	PrgID	Object ID of Program	4
UINT32	PlaybackDuration	Playback duration of Program in millisecond	4
UINT16	Attribute	Attribute of Program	2
UINT16	Profile	Profile of Program	2
UINT16	VideoAttr	Attribute of video stream in Program	2
UINT16	AudioAttr	Attribute of audio stream in Program	2
UINT8	Reserved[4]	Reserved	4
UINT32	RepOffset	Offset of Representation Picture in millisecond	4
UINT16	TextInfoAttr1	Attribute of text information 1	2
UINT16	TextInfoAttr2	Attribute of text information 2	2
UINT8	TextInfo[200]	Text information of Program	200
UINT16	NumMoInfo	Number of Media Objects referred by Program	2
UINT8	NumMarker	Number of marker positions	1
UINT16	NumExtraData	Number of extra data of Program	2
UINT8	Reserved[45]	Reserved	45
Total			280

Version

Describes the version number of PRG_ATTR.



In the first version, major version number shall be 1 and minor version shall be 0.

PrgID

Describes Object ID of the Program.

PlaybackDuration

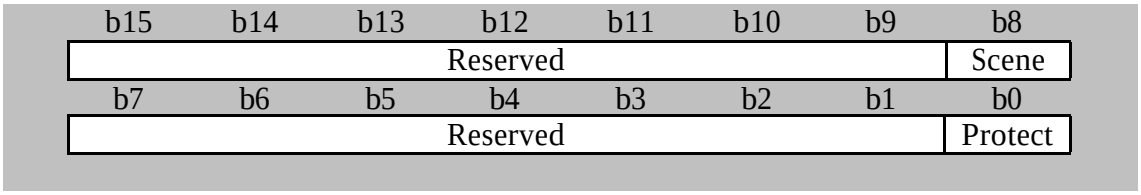
Describes the playback duration of the Program in millisecond. The value shall be equal to the sum of MoDuration fields of MoInfoTbl in PRG_INFO.

ã 2001 SD Association

CONFIDENTIAL

Attribute

Describes the attribute of the Program. The value shall be equal to the Attribute field in PRG_INFO.



Scene	0: No Scene Description File. MoInfoTbl field specifies playback of the Program. 1: Use Scene Description File. The playback of Program starts from SCN001.SML.
Protect	0: The Program is not write protected. 1: The Program is write protected.

Profile

Describes the profile of the Program. All Media Objects referred by the Program shall be in conformity with this profile. The value is as follows.The value shall be equal to the Profile field in PRG_INFO.

0001h	MV (Mobile Video) Profile
Others	Reserved

VideoAttr

Describes attributes of video streams that this Program contains. The value is as follows. If all video streams do not have the same attributes, this field shall be set FFFFh. The value shall be equal to the VideoAttr field in PRG_INFO.

0000h	No video
0100h	MPEG-4 video [Simple profile/Level 1] restricted
0110h	MPEG-4 video [Simple profile/Level 1]
0120h	MPEG-4 video [Simple profile/Level 2]
0130h	MPEG-4 video [Simple profile/Level 3]
0112h	MPEG-4 video [Core profile/Level 1]
FFFFh	More than one attributes exist.
Others	Reserved

AudioAttr

Describes attributes of audio streams that this Program contains. The value is as follows. If all audio streams do not have the same attributes, this field shall be set FFFFh. The value shall be equal to the AudioAttr field in PRG_INFO.

0000h	No Audio
0101h	MPEG-2 AAC (LC)
0111h	GSM_AMR(modified IF2)
0121h	G.726
0131h	Windows Media Audio
FFFFh	More than one attributes exist.
Others	Reserved

ã 2001 SD Association

CONFIDENTIAL

RepOffset

Descries the Representation Picture of the Program. The Representation Picture is defined by time offset of the picture from the beginning of Program in millisecond. RepOffset shall be greater than or equal to zero, and shall be less than the playback duration of the Program, too. In case not supported, the value of this field shall be all zero.

TextInfoAttr1

Describe the text attribute of the first sub field of TextInfo in the PRG_ATTR. The value is as follows. In case not supported, the value of this field shall be 0000h.

0001h	ISO/IEC 646:1983(ASCII)
0002h	JIS X 0201
0003h	ISO 8859-1
Others	Reserved

TextInfoAttr2

Describe the text attribute of the second sub field of TextInfo in the PRG_ATTR. The value is as follows. In case not supported, the value of this field shall be 0000h.

0001h	Shift JIS Kanji
Others	Reserved

TextInfo

Describe the text information of the Program. This field is divided into two sub fields by termination code '00h'. The first sub field is used for the text information of the Program in the character set specified by TextInfoAttr1. The second sub field is coded with another character set specified by TextInfoAttr2. The part after the second field shall be filled with '00h'. In other words, the maximum length of the sum of both sub fields is 198 bytes because at least two termination codes are needed.

If the both text information do not exist, this field shall be filled with '00h'. If only the first field exists, the part after the first sub field shall be filled with '00h'. If only the second field exists, the first byte shall be '00h' and the second field follows after.

NumMoInfo

Describes the number of entries of MoInfoTbl and it is equal to the number of Media Objects referred by the Program. The maximum value of NumMoInfo is 99. The value shall be equal to the NumMoInfo field in PRG_INFO.

NumMarker

Describes the number of entries of MarkerTbl and it is equal to the number of Markers marked in the Program. The maximum value of NumMarker is 99. The value shall be equal to the NumMarker field in PRG_INFO.

NumExtraData

Describes the number of entries of ExtraDataTbl. The maximum value of NumExtraData is 99. The value shall be equal to the NumExtraData field in PRG_INFO.

ã 2001 SD Association

CONFIDENTIAL

5.5 Program Information File

Program Information File contains the management information of the Program. This file has all information for playback of Program. This file shall not include any other data.

5.5.1 Program Information (PRG_INFO)

Program Information defines a Program.

Table 5.5-1 : PRG_INFO

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	DataType	Data type identifier	2
UINT32	DataSize	Size of Data	4
UINT16	Version	Version number of PRG_INFO	2
UINT16	Attribute	Attribute of Program	2
UINT16	Profile	Profile of Program	2
UINT16	VideoAttr	Attribute of video stream in Program	2
UINT16	AudioAttr	Attribute of audio stream in Program	2
UINT8	Reserved[11]	Reserved	11
UINT16	NumMoInfo	Number of Media Objects referred by Program	2
UINT8	NumMarker	Number of marker positions	1
UINT16	NumExtraData	Number of extra data of Program	2
MO_INFO	MoInfoTbl [NumMoInfo]	Table of the information of Media Objects referred by Program	8 * NumMoInfo
UINT32	MarkerTbl [NumMarker]	Table of marker positions in millisecond	4 * NumMarker
PRG_EXT	ExtraDataTbl [NumExtraData]	Table of extra data of Program	Variable
Total			Variable

DataType

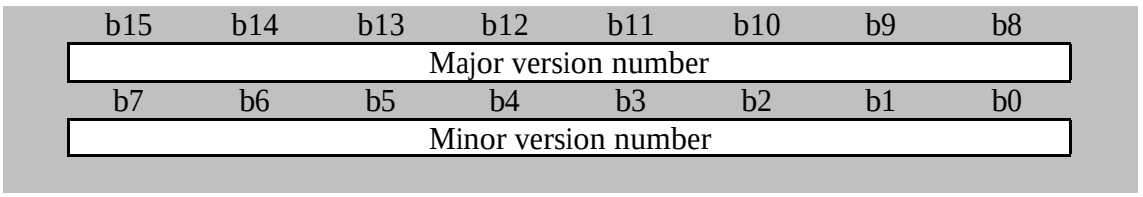
This field shall be “V3” with character set code of ISO/IEC 646:1983 (ASCII).

DataSize

Describes the size of PRG_INFO.

Version

Describes the version number of PRG_INFO.



In the first version, major version number shall be 1 and minor version shall be 0.

ã 2001 SD Association

CONFIDENTIAL

Attribute

Describes the attribute of the Program. The value shall be equal to the Attribute field in PRG_ATTR.

Profile

Describes the profile of the Program. All Media Objects referred by the Program shall be in conformity with this profile. The value shall be equal to the Profile field in PRG_ ATTR.

VideoAttr

Describes attributes of video streams that this Program contains. The value shall be equal to the VideoAttr field in PRG_ ATTR.

AudioAttr

Describes attributes of audio streams that this Program contains. The value shall be equal to the AudioAttr field in PRG_ ATTR.

NumMoInfo

Describes the number of entries of MoInfoTbl and it is equal to the number of Media Objects referred by the Program. The maximum value of NumMoInfo is 99. The value shall be equal to the NumMoInfo field in PRG_ATTR.

NumMarker

Describes the number of entries in MarkerTbl and it is equal to the number of Markers marked in the Program. The maximum value of NumMarker is 99. The value shall be equal to the NumMarker field in PRG_ ATTR.

NumExtraData

Describes the number of entries of ExtraDataTbl. The maximum value of NumExtraData is 99. The value shall be equal to the NumExtraData field in PRG_ ATTR.

MoInfoTbl

Describes the information of Media Object in tabular form that is referred by the Program. If Scene bit of Attribute field is 0, the Program is expected to playback in order of entries in MoInfoTbl. All Media Objects that are specified by MoInfoTbl shall exist in Program, and no Media Object that is not specified by MoInfoTbl shall exist. The specification of the information of Media Object (MO_INFO) are as follows.

Table 5.5-2 : MO_INFO

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	MoAttr	Attribute of Media Object referred by Program	2
UINT16	MoID	ID of Media Object referred by Program	2
UINT32	MoDuration	Playback duration time of Media Object in millisecond	4
Total			8

ã 2001 SD Association

CONFIDENTIAL

MoAttr

b15	b14	b13	b12	b11	b10	b9	b8
Reserved							
b7	b6	b5	b4	b3	b2	b1	b0
Reserved			MEnc	Sep	SEnc	Aud	AEnc

MEnc	0: Unencrypted Media Object 1: Encrypted Media Object
Sep	0: No Separately-Recorded Audio exists 1: Separately-Recorded Audio exists
SEnc	0: Unencrypted Separately-Recorded Audio 1: Encrypted Separately-Recorded Audio
Aud	0: No Replaced Audio exists 1: Replaced Audio exists
AEnc	0: Unencrypted Replaced Audio 1: Encrypted Replaced Audio

MoID

b15	b14	b13	b12	b11	b10	b9	b8
MediaObjectType				MediaObjectNumber[11..8]			
b7	b6	b5	b4	b3	b2	b1	b0
MediaObjectNumber [7..0]							

MediaObjectType	1: Media Object Data File 2: Advanced Streaming Format File Others: Reserved
MediaObjectNumber	File Number of Media Object

MoDuration

Describes the playback duration of Media Object in millisecond. In case that Media Object consists of Movie Data File and Separately-Recorded Audio Data File and/or Replaced Audio Data File, and their Playback Durations are different each other, it is up to the encoder design which value to adopt as the MoDuration.

MarkerTbl

Describe the table of the time offset of Marker in millisecond. All values of Marker shall be stored in ascending order, and shall be greater than or equal to zero, and less than the playback duration of the Program.

ExtraDataTbl

Describes the table of extra data. The specification of the extra data structure are as follows.

ã 2001 SD Association

CONFIDENTIAL

Table 5.5-3 : PRG_EXT

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	ExtraDataType	Data type identifier	2
UINT32	ExtraDataSize	Size of extra data	4
UINT8	CharCode	Character code of text	1
	Data	Contents of extra data	ExtraDataSize - 7
Total			ExtraDataSize

ExtraDataType

This field indicates the type of this extra data. The values of the field are as follows.

0001h	Author
0002h	Title
0003h	Copyright
0004h	Description
0005h	Editor
0006h	Producer
0007h	Recording Company
0008h	Date
Others	Reserved

ExtraDataSize

Describes the size of PRG_EXT. The value shall be 10kB or less.

CharCode

Describes the attribute in case of text. The values are as follows.

01h	ISO / IEC 646:1983(ASCII)
02h	JIS X 0201
03h	ISO 8859-1
81h	Shift-JIS Kanji
Others	Reserved

Data

Describes the extra data according to the value of ExtraDataType field.

ã 2001 SD Association

5.6 Playlist Manager File

Playlist Manager File contains Playlist Manager, and its file name shall be “PLST_MGR”. This file shall not include any other data. The size of this file shall be 32kB or less.

5.6.1 Playlist Manager (PLST_MGR)

Playlist Manager File (PLST_MGR) defines characteristics of all Playlists in SD-Video. All Playlists are user-defined, and specify sequential playback of more than one part of Program.

Table 5.6-1 : PLST_MGR

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	DataType	Data type identifier	2
UINT32	DataSize	Size of Data	4
UINT16	Version	Version number of PLST_MGR	2
UINT8	Reserved[30]	Reserved	30
UINT16	NumPlstInfo	Number of Playlist Information entries	2
PLST_INFO	PlstInfoTbl [NumPlstInfo]	Table of Playlist Information	Variable
Total			Variable

DataType

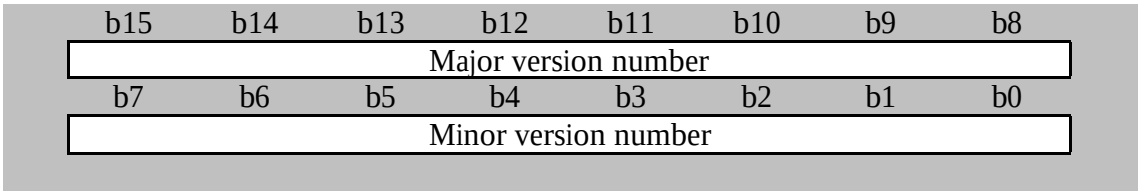
This field shall be “V4” with character set code of ISO/IEC 646:1983 (ASCII).

DataSize

Describes the size of PLST_MGR.

Version

Describes the version number of PLST_MGR.



In the first version, major version number shall be 1 and minor version shall be 0.

NumPlstInfo

Describes the number of Playlists, which is equal to the number of Playlist Information entries. The maximum value of NumPlstInfo is 99.

PlstInfoTbl

Describes the Playlist Information of each Program in tabular form. The size of Playlist Information has variable length, and is stored in DataSize field of PLST_INFO.

ã 2001 SD Association

CONFIDENTIAL

5.6.2 Playlist Information (PLST_INFO)

Playlist Information defines characteristics of a Playlist. The size of Playlist Information has variable length, and is stored in DataSize field.

Table 5.6-2 : PLST_INFO

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	DataType	Data type identifier	2
UINT32	DataSize	Size of Data	4
UINT16	Version	Version number of PLST_INFO	2
OBJECTID	PlstID	Object ID of Playlist	4
UINT32	PlaybackDuration	Playback duration of Playlist in millisecond	4
UINT16	Attribute	Attribute of Playlist	2
UINT32	RepOffset	Offset of Representation Picture in millisecond	4
UINT16	TextInfoAttr1	Attribute of text information 1	2
UINT16	TextInfoAttr2	Attribute of text information 2	2
UINT8	TextInfo[200]	Text information of Playlist	200
UINT8	Reserved[11]	Reserved	11
UINT16	NumPrgID	Number of Programs referred by Playlist	2
UINT8	NumMarker	Number of marker positions	1
PRGSECTION	PrgSectionTbl [NumPrgID]	Table of Program Section Structures of Program referred by Playlist	12 * NumPrgID
UINT32	MarkerTbl [NumMarker]	Table of marker positions in millisecond	4 * NumMarker
Total			Variable

DataType

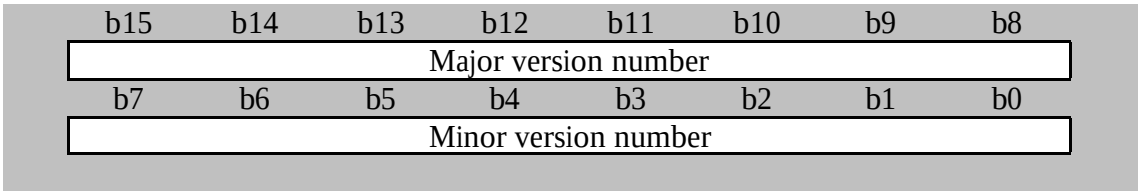
This field shall be “V5” with character set code of ISO/IEC 646:1983 (ASCII).

DataSize

Describes the size of PLST_INFO. This value is used to calculate the offset to the next Playlist Information.

Version

Describes the version number of PLST_INFO.



In the first version, major version number shall be 1 and minor version shall be 0.

PlstID

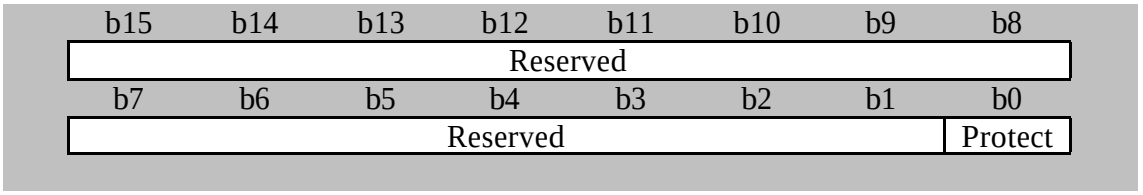
Describes Object ID of the Playlist.

PlaybackDuration

Describes the playback duration of the Playlist in millisecond. The value shall be equal to the sum of each playback duration of Program referred by the Playlist. The playback duration of Program shall be equal to the difference between StartPos field and EndPos field of PRGSECTION.

Attribute

Describes the attribute of the Playlist.



Protect	0: The Playlist is not write-protected. 1: The Playlist is write-protected.

RepOffset

Describes offset position of the Representation Picture of the Playlist. The Representation Picture is defined by time offset of the picture from the beginning of Playlist in millisecond. RepOffset shall be greater than or equal to zero, and shall be less than the playback duration of the Program. In case not supported, the value of this field shall be all zeros.

TextInfoAttr1

Describe the text attribute of the first sub field of TextInfo in the PLST_INFO. The value is as follows. In case not supported, the value of this field shall be 0000h.

0001h	ISO/IEC 646:1983(ASCII)
0002h	JIS X 0201
0003h	ISO 8859-1
Others	Reserved

TextInfoAttr2

Describe the text attribute of the second sub field of TextInfo in the PLST_INFO. The value is as follows. In case not supported, the value of this field shall be 0000h.

0001h	Shift JIS Kanji
Others	Reserved

TextInfo

Describe the text information of the Playlist. This field is divided into two sub fields by termination code '00h'. The first sub field is used for the text information of the Playlist in the character set specified by TextInfoAttr1. The second sub field is coded in another character set specified by TextInfoAttr2. The part after the second field shall be filled with '00h'. In other words, the maximum length of the sum of both sub fields is 198 bytes because at least two termination codes are needed.

If the both text information do not exist, this field shall be filled with '00h'. If only the first field exists, the part after the first sub field shall be filled with '00h'. If only the second field exists, the

ã 2001 SD Association

CONFIDENTIAL

first byte shall be '00h' and the second field follows after.

NumPrgID

Describes the number of entries in PrgSectionTbl and it shall be equal to the number of Program Sections referred by the Playlist. The maximum value of NumPrgID is 99.

NumMarker

Describes the number of entries in MarkerTbl and it shall be equal to the number of Markers marked in the Playlist. The maximum value of NumMarker is 99.

PrgSectionTbl

Describes the sections of Program in tabular form that is referred by the Playlist. The Playlist is expected to playback in order of entries in PrgSectionTbl. All Programs referred by the Playlist shall exist. The values of StartPos and EndPos shall be greater than or equal to zero, and less than the playback duration of specified Program. The value of StartPos shall be less than the value of EndPos in the same PrgSection. The specification of the PRGSECTION structure are as follows.

Table 5.6-3 : PRGSECTION

Field Type	Field Name	Contents	Field Size (Byte)
OBJECTID	ObjID	Object ID	4
UINT32	StartPos	Start time offset in millisecond	4
UINT32	EndPos	End time offset in millisecond	4
Total			12

ObjID

Describes Object ID of Program.

StartPos

Describes start time offset in the Object in millisecond.

EndPos

Describes end time offset in the Object in millisecond.

MarkerTbl

Describe the time offset of Marker in tabular form in millisecond. All values of Marker shall be stored in ascending order, and shall be greater than or equal to zero, and less than the playback duration of the Program.

ã 2001 SD Association

CONFIDENTIAL

5.7 Media Object Information File

Media Object Information File (*.MOI) contains Media Object Information. This file shall not include any other data.

5.7.1 Media Object Information (MOI)

Media Object Information (MOI) defines characteristics of a Media Object Data File with the same filename except for extension.

Table 5.7-1 : MOI

Field Type		Field Name	Contents	Field Size (Byte)
UINT16		DataType	Data type identifier	2
UINT32		DataSize	Size of Data	4
CREATION_TIME		CreationTime	Time of creation	8
UINT32		PlaybackDuration	Playback duration time of Media Object Data in millisecond	4
STRM_ATTR		StrmAttr	Attribute of streams in Media Object	16
UINT8		Reserved[93]	Reserved	93
UINT8		TstType	Type of Time Search Table	1
TstType==1	UINT16	TstInterval	Time interval for each Time Search Entry in millisecond	2
	UINT16	FrameTimeNmr	Numerator of playback time of 1 frame in millisecond	2
	UINT16	FrameTimeDnm	Denominator of playback time of 1 frame in millisecond	2
	UINT16	NumModui	Number of MODU Information entries	2
	MODU_INFO	ModuiTbl [NumModui]	Table of MODU Information	6 * NumModui
	UINT16	NumTseInfo1	Number of Time Search Entries	2
	TSE_INFO1	TseInfoTbl1 [NumTseInfo1]	Table of Time Search Entry Information 1	7 * NumTseInfo1
TstType==2	UINT16	TstInterval	Time Interval for each Time Search Entry in millisecond	2
	UINT16	NumTseInfo2	Number of Time Search Entries	2
	TSE_INFO2	TseInfoTbl2 [NumTseInfo2]	Table of Time Search Entry Information 2	5 * NumTseInfo2
TstType==3	UINT16	FrameTimeNmr	Numerator of playback time of 1 frame in millisecond	2
	UINT16	FrameTimeDnm	Denominator of playback time of 1 frame in millisecond	2
	UINT32	PackSize	Size of 1 pack in bytes	4
	UINT8	NumSkippedPack	Number of packs to the first AV stream Pack	1
	UINT8	NumFrame	Number of frames within 1 pack	1
Total				Variable

ã 2001 SD Association

CONFIDENTIAL

DataType

This field shall be “V6” with character set code of ISO/IEC 646:1983 (ASCII).

DataSize

Describes the size of MOI.

CreationTime

Describes the date and time of the initial creation of this Media Object Data File. The format is as follows. In case not supported, the value of this field sall be all zeros.

Table 5.7-2 : CREATION_TIME

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	Year	Year of the initial creation	2
UINT8	Month	Month of the initial creation	1
UINT8	Day	Day of the initial creation	1
UINT8	Hour	Hour of the initial creation	1
UINT8	Minute	Minute of the initial creation	1
UINT16	Millisecond	Millisecond of the initial creation	2
Total			8

Year

Describes the year as a number in the range ‘1’ to ‘9999’

Month

Describes the month as a number in the range ‘1’ to ‘12’

Day

Describes the day as a number in the range ‘1’ to ‘31’

Hour

Describes the hour as a number in the range ‘0’ to ‘23’

Minute

Describes the minute as a number in the range ‘0’ to ‘59’

Millisecond

Describes the millisecond as a number in the range ‘0’ to ‘59999’

PlaybackDuration

Describes the playback duration of Media Object Data File in millisecond.

StrmAttr

Describes the attribute of stream in Media Object Data File.

Table 5.7-3 : STRM_ATTR

Field Type	Field Name	Contents	Field Size (Byte)
UINT8	PrgStreamMap	Existence of program stream map	1
UINT8	CharCode	Character set of text	1
UINT8	Reserved[14]	Reserved	14
Total			16

ã 2001 SD Association

CONFIDENTIAL

PrgStreamMap

00h	No Program stream map exists
01h	Program stream map exists

CharCode

01h	ISO/IEC 646:1983 (ASCII)
02h	JIS X 0201
03h	ISO 8859-1
81h	Shift-JIS Kanji
Others	Reserved

TstType

Describes type of Time Search Table. The values are as follows.

00h	No Time Search Table
01h	Time Search Table for Media Object Data File that has MODU Information
02h	Time Search Table for Media Object Data File that has no MODU Information
03h	Time Search Table for Media Object Data File of fixed pack size audio or text
Others	Reserved

TstType: 00h

No Time Search Table fields exist after this.

TstType: 01h

TstInterval

Describes interval time between two successive Time Search Entries in millisecond.

FrameTimeNmr

Describes the numerator of playback duration time of 1 frame in millisecond. This field is used with the next field FrameTimeDnm. If Media Object Data File has variable frame-rate streams, this field shall have a value of zero.

FrameTimeDnm

Describes the denominator of playback duration time of 1 frame. This field is used with the previous field FrameTimeNmr. If Media Object Data File has variable frame-rate streams, this field shall have a value of zero.

NumModui

Describes the number of MODU Information entries, and it shall be equal to the number of MODUs.

ModuiTbl

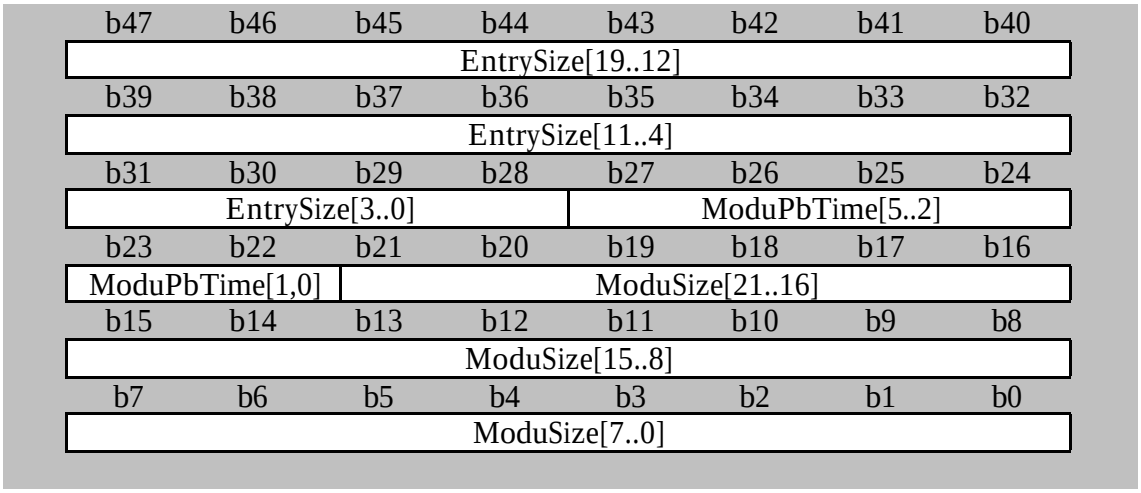
Describes the table of MODU Information. The structure of MODU Information is as follows.

ã 2001 SD Association

CONFIDENTIAL

Table 5.7-4 : MODU_INFO

Field Name	Contents	Field Size (Bit)
EntrySize	Total size of packs used by Entry Frame in bytes	20
ModuPbTime	Playback time of MODU by the number of frames	6
ModuSize	Size of MODU in bytes	22
Total		48 (6 Bytes)



EntrySize

Describes the total size of packs used by the Entry Frame.

ModuPbTime

Describes playback time of the MODU by the number of frames.

ModuSize

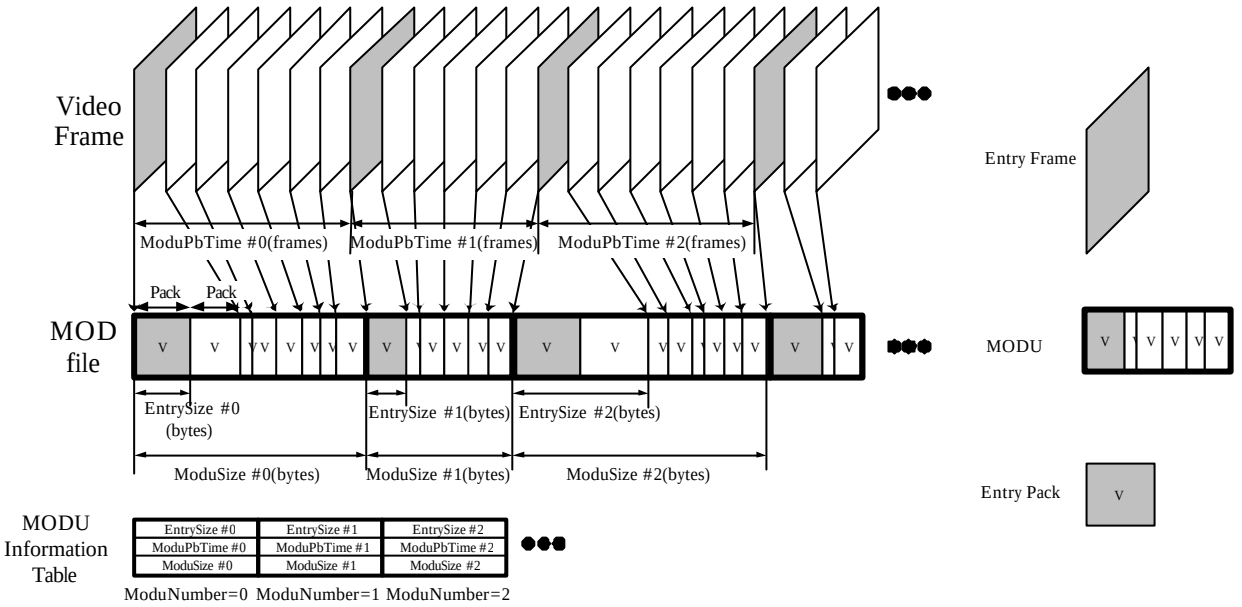
Describes the size of MODU in bytes.

ã 2001 SD Association

CONFIDENTIAL

Figure 5.7-1 shows the relations between the table of MODU Information and MOD file.

Figure 5.7-1 : Table of MODU Information



NumTseInfo1

Describes the number of entries in Time Search Table Information 1, and it shall be equal to the number of Time Search Entries.

TseInfoTbl1

Describes the table of Time Search Entry Information 1. The structure of Time Search Entry Information 1 is as follows.

Table 5.7-5 : TSE_INFO1

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	ModuNumber	MODU Number.	2
UINT8	EntryFrameDiff	Number of frames between Entry Frame and Time Search Entry	1
UINT32	ModuOffset	Offset of MODU in bytes	4
Total			7

ModuNumber

Describes the MODU Number.

EntryFrameDiff

Describes the number of frames between the Entry Frame just before the Time Search Entry and the Time Search Entry.

ModuOffset

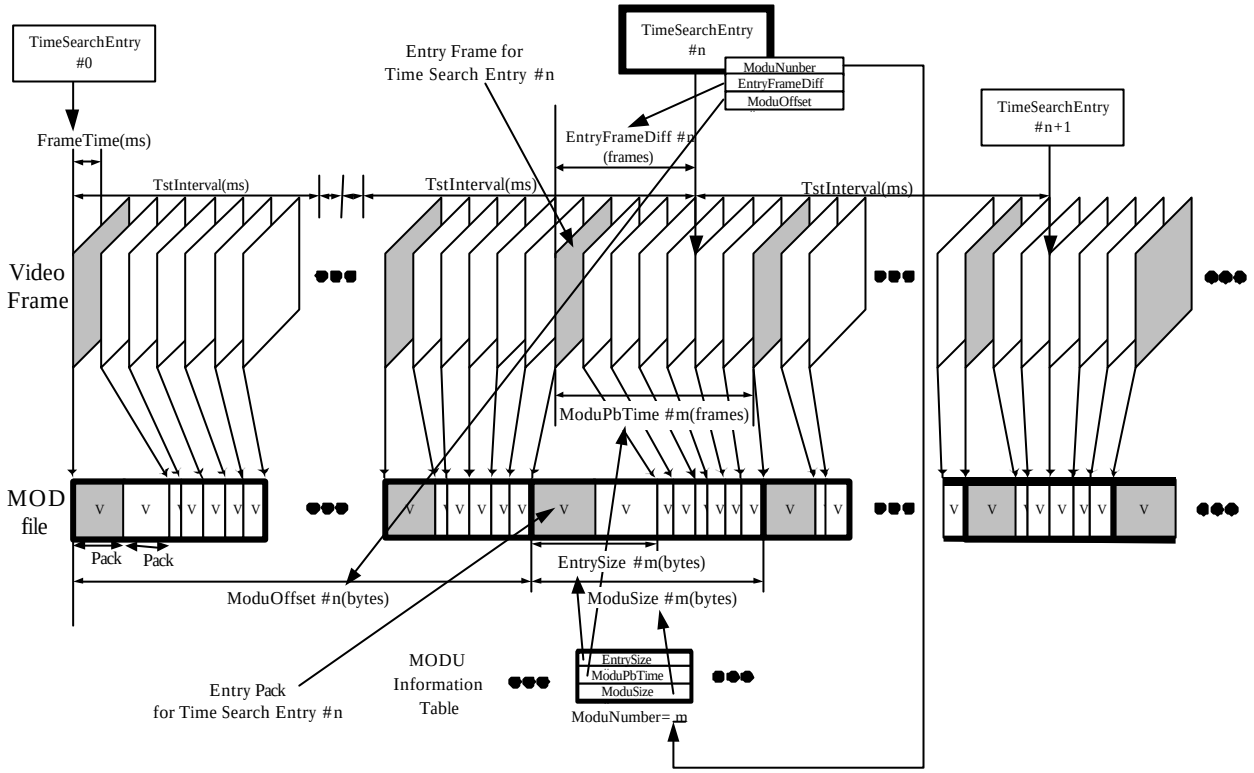
Describes the offset of MODU in bytes.

ã 2001 SD Association

CONFIDENTIAL

Figure 5.7-2 shows the relations between Time Search Table and MOD file in case of having MODU Information.

Figure 5.7-2 : Structure of Time Search Table in case of having MODU Information



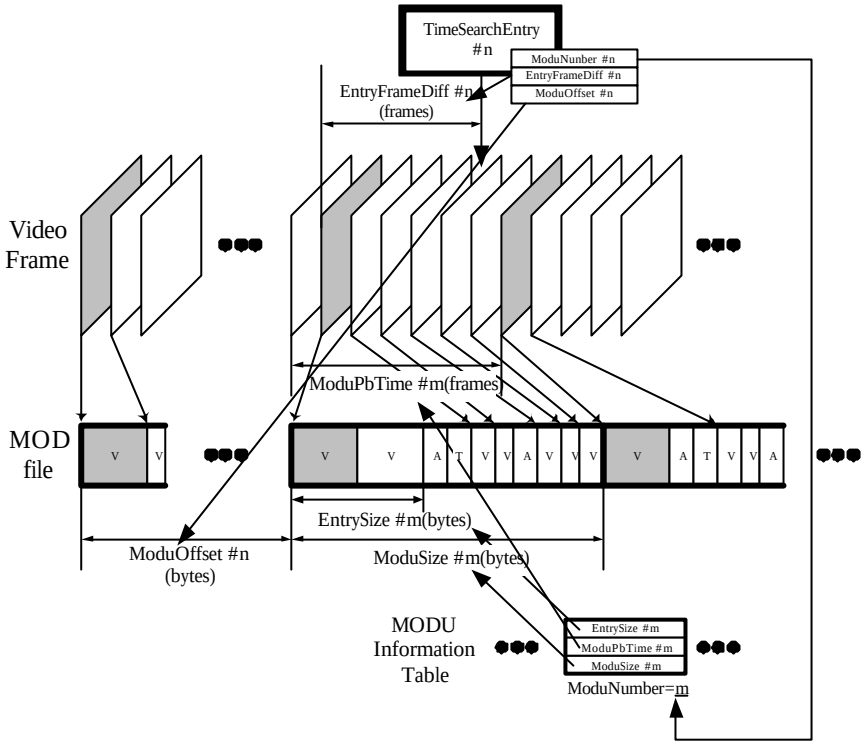
Also in case of video interleaved with audio or text, Time Search Entry indicates video frame.

ã 2001 SD Association

CONFIDENTIAL

Figure 5.7-3 shows in case of interleaved video and audio or text.

Figure 5.7-3 : Structure of Time Search Table in case of movie interleaved video and audio or text



TstType: 02h

TstInterval

Describes time interval of two successive Time Search Entries in millisecond.

NumTseInfo2

Describes the number of entries in Time Search Table Information 2, and it shall be equal to the number of Time Search Entries.

TseInfoTbl2

Describes the table of Time Search Entry Information 2. The structure of Time Search Entry Information 2 is as follows.

Table 5.7-6 : TSE_INFO2

Field Type	Field Name	Contents	Field Size (Byte)
UINT8	EntryFrameDiff	Number of frames between Entry Frame and Time Search Entry	1
UINT32	ModuOffset	Offset of MODU in bytes	4
Total			5

ã 2001 SD Association

CONFIDENTIAL

EntryFrameDiff

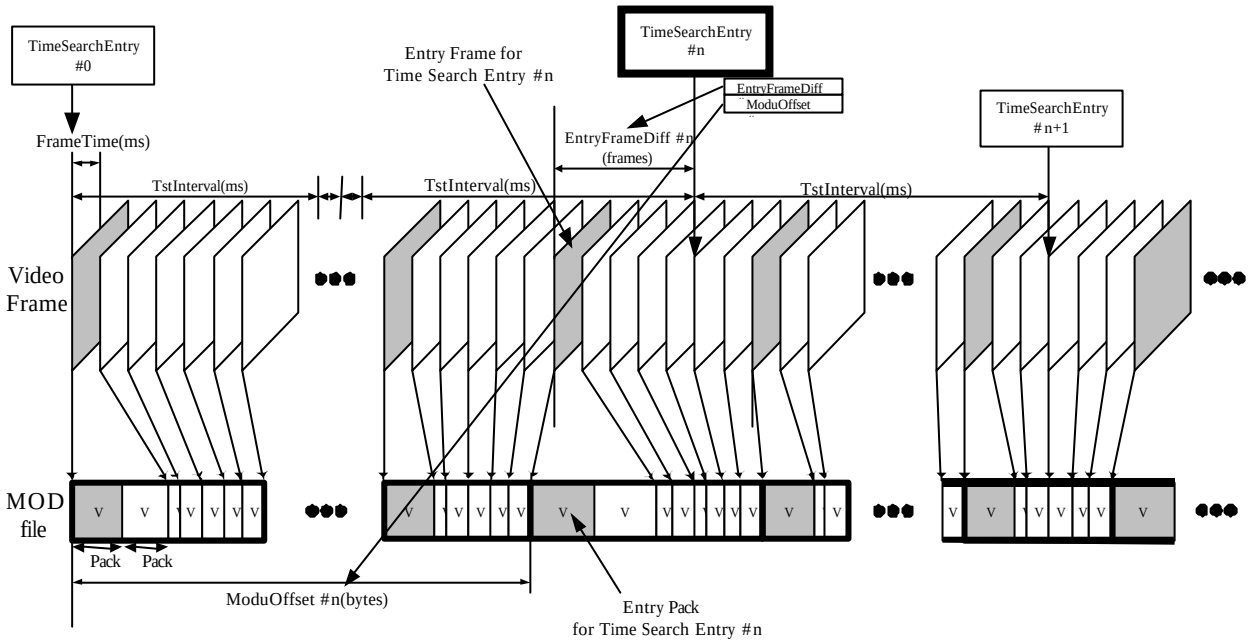
Describes the number of frames between the start frame of MODU where Time Search Entry exists and the Time Search Entry.

ModuOffset

Describes the offset of MODU in bytes.

Figure 5.7-4 shows the relations between Time Search Table and MOD file in case of no MODU Information.

Figure 5.7-4 : Structure of Time Search Table in case of no MODU Information

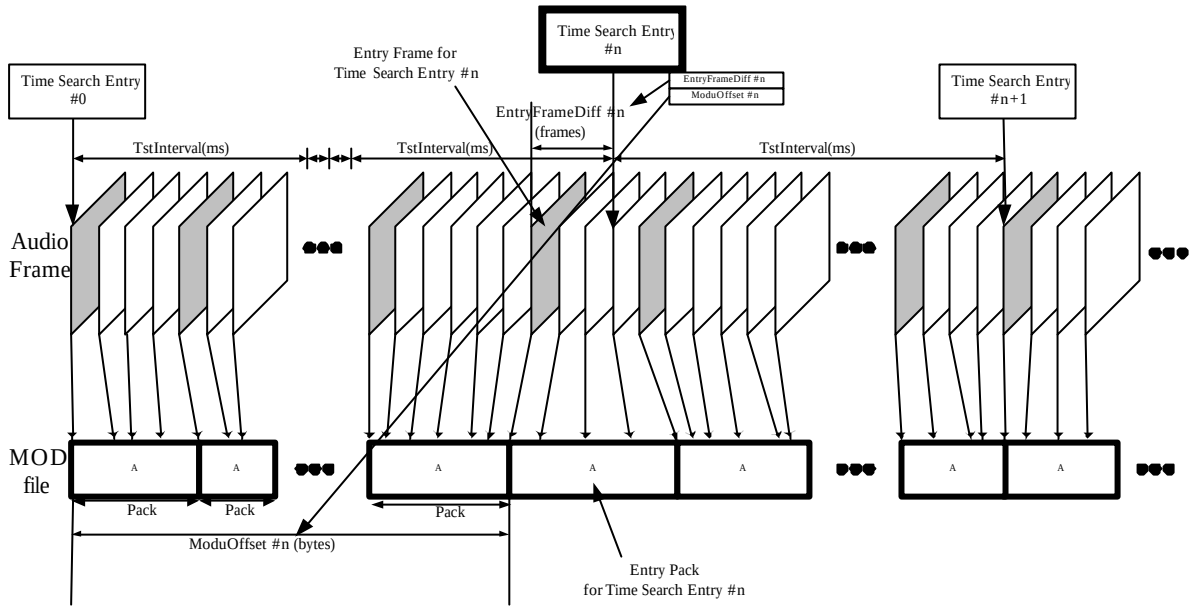


ã 2001 SD Association

CONFIDENTIAL

Figure 5.7-5 shows the relations between Time Search Table and MOD file in case that it has only variable pack size audio.

Figure 5.7-5 : Structure of Time Search Table in case of variable pack size audio



TstType: 03h

FrameTimeNmr

Describes the numerator of playback duration time of 1 frame in millisecond. This field is used with the next field FrameTimeDnm.

FrameTimeDnm

Describes the denominator of playback duration time of 1 frame. This field is used with the previous field FrameTimeNmr.

PackSize

Describes the size of a pack in bytes.

NumSkippedPack

Describes the number of packs to the first AV stream pack.

NumFrame

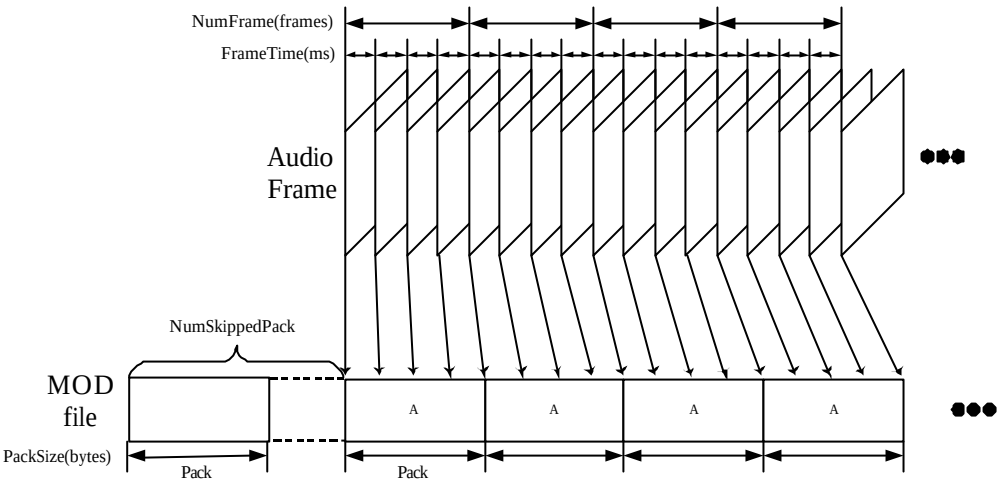
Describes the number of frames within 1 pack.

ã 2001 SD Association

CONFIDENTIAL

Figure 5.7-6 shows the relations between Time Search Table and MOD file in case of fixed pack size audio.

Figure 5.7-6 : Structure of Time Search Table in case of fixed pack size audio



This page is intentionally left blank

ã 2001 SD Association

6 Scene Description Data Structure

This specification uses SMIL (Synchronized Multimedia Integration Language) 2.0 Basic Profile as Scene Description data which is currently work in progress.

Refer to “http://www.w3.org/TR/smil20.

ã 2001 SD Association

CONFIDENTIAL

This page is intentionally left blank

ã 2001 SD Association

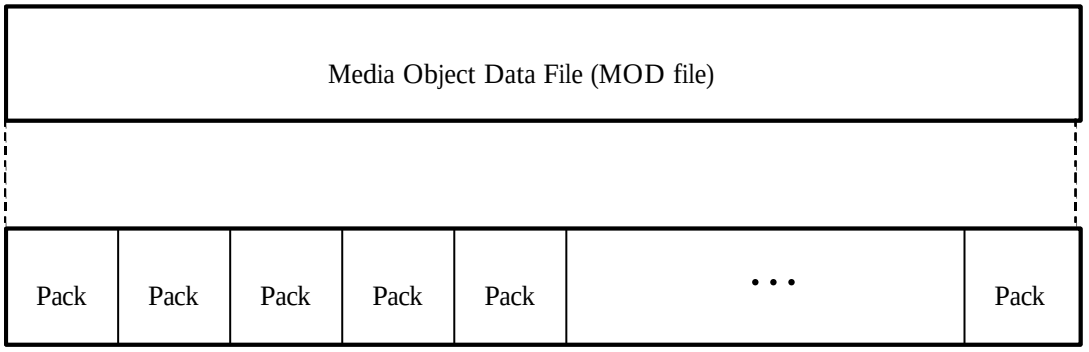
7. Media Object Data File

7.1 Definition of Media Object Data File

Media Object Data File (MOD file) is a file which contains Program Stream prescribed in "Information Technology - Generic coding of moving pictures and associated audio: Systems (ISO/IEC 13818-1)".

MOD file contains a sequence of packs as shown in Figure 7.1-1.

Figure 7.1-1 : Structure of pack (in case of no padding packet)



MOD file contains the Presentation Data which are packed into the following packs.

Table 7.1-1 : List of Pack types

Pack name	Data in Pack
Video Pack (V_PCK)	Video data
Audio Pack (A_PCK)	Audio data
Text Pack (T_PCK)	Text data
PSM Pack (P_PCK)	Program Stream Map

MOD file has the following restrictions :

1) The first Pack of MOD file shall contain a system header.
2) The first Pack of MOD file shall be P_PCK if MOD contains :
    a) A_PCK of MPEG-2 AAC, G.726, Windows Media Audio or AMR audio.
    b) V_PCK of MPEG-4 video.
    c) T_PCK.
3) Value of SCR in the first Pack of MOD file may be non-zero.
4) One MOD file may comprise more than one kind of Packs listed in Table 7.1-1.

Each Presentation Data is identified by the stream_id in the Program Stream.

ã 2001 SD Association

CONFIDENTIAL

Table 7.1-2 : stream_id

stream_id	stream coding type
1100 0000	Audio - ISO/IEC 13818-7 (MPEG-2 AAC) - Windows Media Audio - G.726 - AMR
1110 0000 - 1110 0011	Video - ISO/IEC 14496-2 (MPEG-4)
1011 1101	Text
1011 1100	Program Stream Map
1011 1110	Padding Stream
Others	no use

Stream coding type of Presentation Data is identified according to the following convention :

1) If the first Pack of MOD file is a P_PCK, stream coding type is defined by the stream_type field of the Program Stream Map.

Table 7.1-3 : stream_type

stream_type	stream coding type
0000 1111	ISO/IEC 13818-7 (MPEG-2 AAC)
0001 0000	ISO/IEC 14496-2 (MPEG-4 Video)
1000 0001	Windows Media Audio
1000 0010	G.726
1000 0011	AMR
1000 1000	Text
Others	no use

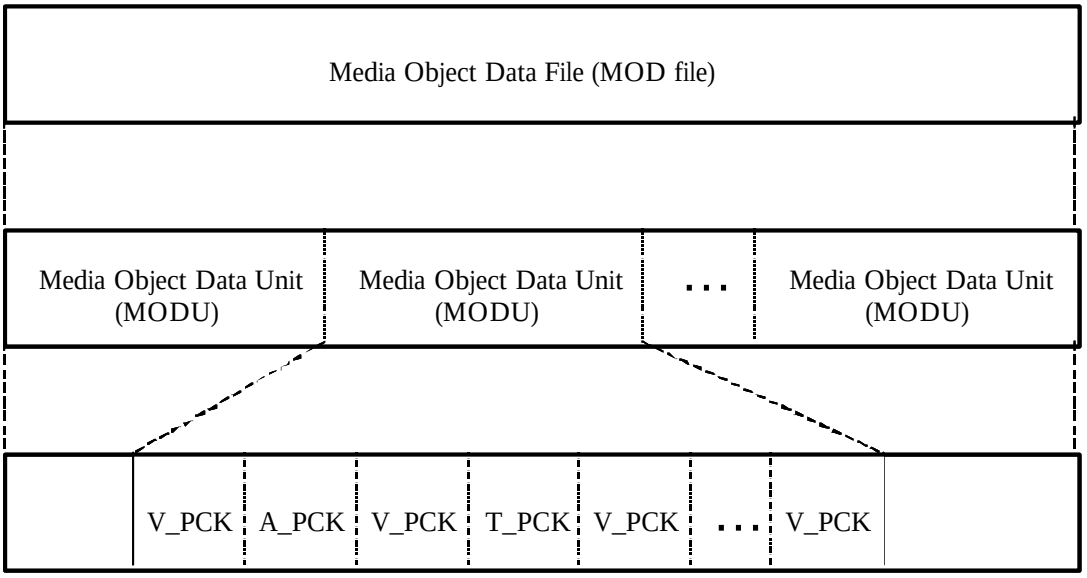
ã 2001 SD Association

CONFIDENTIAL

7.2 Media Object Data Unit

Media Object Data Unit (MODU) is a substructure of MOD file and facilitates random access to MOD file. MODU is a sequence of Packs in MOD file when MOD file contains MPEG-4 Video data if such Video data fulfills the following restrictions :

Figure 7.2-1 : Structure of MODU



1) MODU shall include video data which consists of one or more GOVs. The video data in each MODU shall begin with VisualObjectSequence followed by a GOV header and VideoObjectPlane.

2) MODU is a sequence of Packs in recording order. It shall start with a Pack, V_PCK which includes a system_header and shall end either immediately before the Pack which includes a system_header in the same MOD file or at the end of the MOD file. In other words, a system_header shall exist in the first Pack in a MODU, and shall not exist in any other Packs in MODU.

If MODU structure applies to MOD file, the TstType field shall be set to 01h in MOI file.

.

ã 2001 SD Association

CONFIDENTIAL

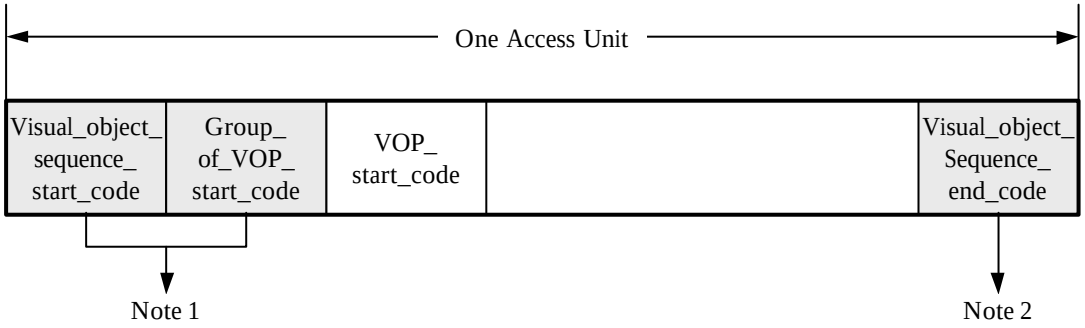
7.3 Access Unit

Access Unit is the coded representation of the presentation data unit. In case of Audio, a presentation data unit corresponds to a decoded single audio frame. In case of Video, a presentation data unit corresponds to a decoded picture.

Table 7.3-1 : Definitions of Access Unit

Data type	Definition
ISO/IEC 14496-2 (MPEG-4 Video)	See Figure 7.3-1.
ISO/IEC 13818-7 (MPEG-2 AAC) Windows Media Audio G.726 AMR	An audio frame
Text	A set of characters that is presented at the same time.

Figure 7.3-1 : Definitions of Video Access Unit



Note 1 : If a picture is not preceded by a Group_of_VOP_start_code or Visual_object_sequnece_start_code, the access unit shall begin with the VOP_start_code. If a picture is preceded by a Group_of_VOP_start_code and/or Visual_object_sequnece_start_code, the access unit shall begin with the first byte of these start codes.

Note 2 : If a Visual_object_Sequence_end_code presents, it shall belong to the preceding picture.

ã 2001 SD Association

CONFIDENTIAL

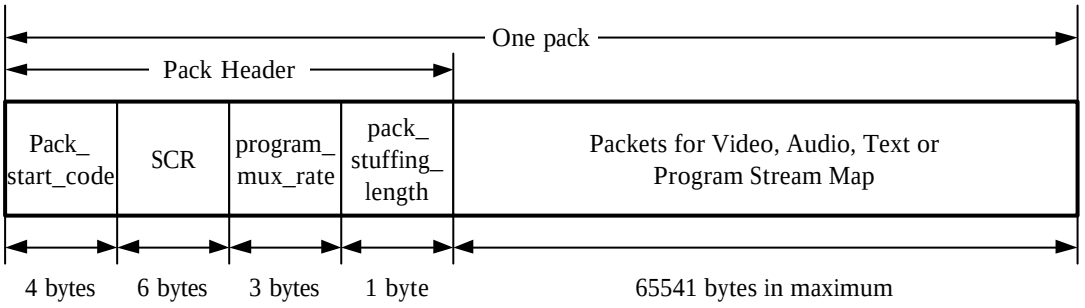
7.4 Pack

The structure of Pack and Packet shall comply with “The system part of the MPEG-2 standard (ISO/IEC 13818-1)”.

7.4.1 Structure of pack

Pack consists of a Pack header and one Packet as shown in Figure 7.4-1.

Figure 7.4-1 : Structure of Pack (in case of no padding Packet)



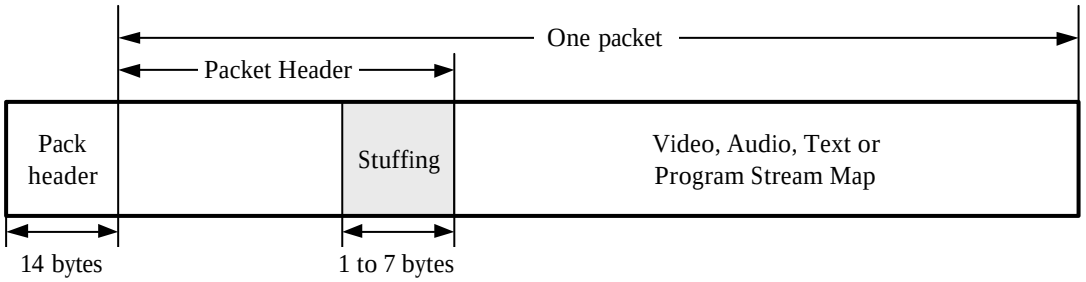
It is not necessary to adjust the Pack size to be constant value. However, if it is desired each Pack size to be constant, it shall be adjusted according to only one of the methods shown in Table 7.4.1-1 and Figure 7.4-1.

Table 7.4-1 : Pack length adjustment method

Number of adjusted data	Adjustment method
1 to 7 bytes	Insert stuffing bytes in a Packet header
8 bytes or more	Add a padding packet as the last Packet in a pack

Figure 7.4-2 : Structure of pack length adjustment

- When a Padding Packet is not inserted



ã 2001 SD Association

CONFIDENTIAL

- When a padding Packet is inserted

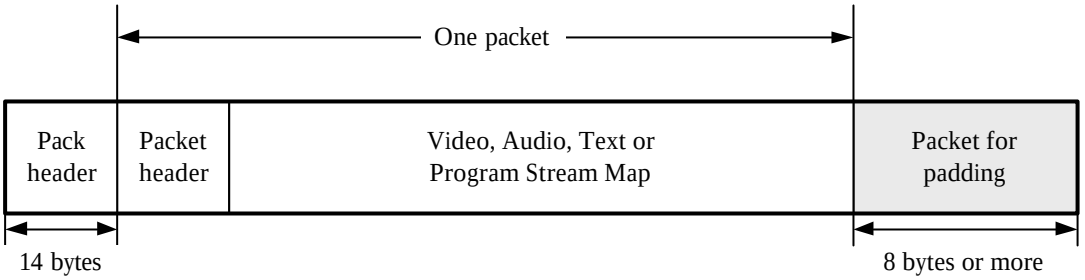


Table 7.4-2 : Pack header

Field	Number of bits	Number of bytes	Value	Comment
Pack_start_code	32	4	0000 01BAh	
'01'	2	6	Provider defined	01b
SCR_base[32...30]	3			(Note 1)
marker_bit	1			1
SCR_base[29...15]	15			
marker_bit	1			1
SCR_base[14...0]	15			
marker_bit	1			1
SCR_extension	9			
marker_bit	1			1
program_mux_rate	22	3	Provider defined	(Note 2)
marker_bit	1			1
marker_bit	1			1
reserved	5	1	F8h	11111b
pack_stuffing_length	3			no stuffing length = 000b

Note 1 : "SCR_base[32]" shall be set to ZERO.
Note 2 : If "program_mux_rate" is unknown, value '3F FFFFh' shall be set.

ã 2001 SD Association

CONFIDENTIAL

7.4.2 Video Pack (V_PCK)

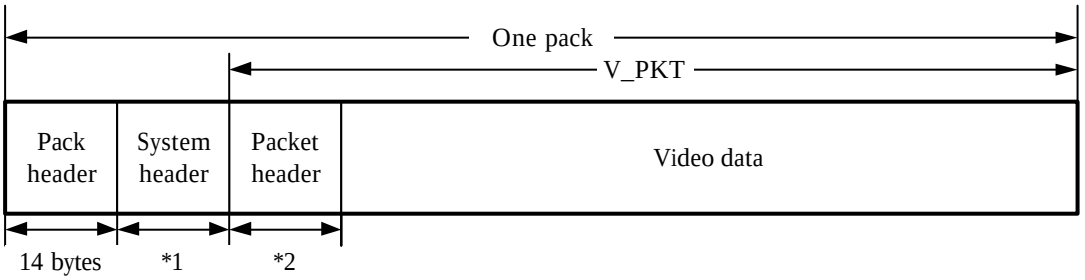
Video Pack (V_PCK) comprises a pack header and a Video Packet (V_PKT) as shown in Figure 7.4-3. A system header shall follow a Pack header when the following condition is true :

1. The V_PCK contains the first byte of the Visual_object_sequence_start_code of ISO/IEC 14496-2 (MPEG-4) Video.

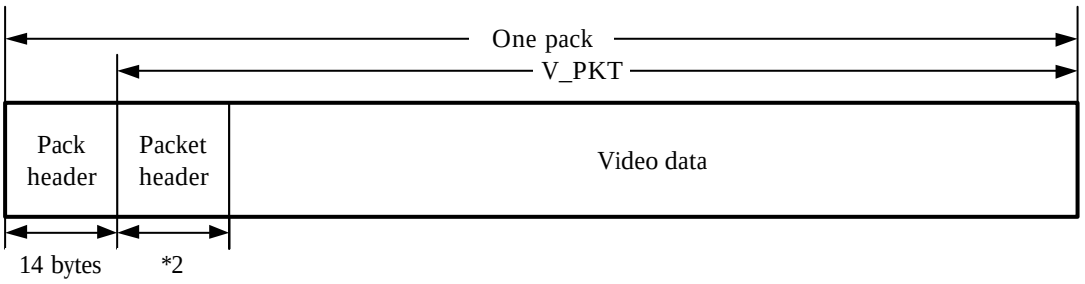
A V_PCK shall contain a whole or the part of an access unit. In other words, a V_PCK shall not accommodate more than one access unit. This restriction shall apply even if a user desires that the size of V_PCKs to be constant value. In this case, it shall be adjusted by either method shown in Table 7.4-1.

Figure 7.4-3 : Structure of Video Pack

- V_PCK when the above condition is true



- V_PCK when the above condition is false



*1 : The size of system header varies between 15 and 21 bytes.

*2 : The size of Packet header varies between 9 and 26 bytes according to the condition of flags in the Packet header.

ã 2001 SD Association

CONFIDENTIAL

Table 7.4-3 : System header in V_PCK

Field	Number of bits	Number of bytes	Value	Comment
System_header_start_code	32	4	0000 01BBh	
header_length	16	2		
marker_bit	1	3	1	
rate_bound	22		Provider defined	(Note 1)
marker_bit	1		1	
audio_bound	6	2	0 to 1	
fixed_flag	1		0 or 1	
CSPS_flag	1		0 or 1	
system_audio_lock_flag	1		1	
system_video_lock_flag	1		1	
marker_bit	1		1	
video_bound	5		0 to 4	
packet_rate_restriction_flag	1	1	0 or 1	
reserved_bit	7		7Fh	
stream_id	8	1	1011 1001b	all Video streams
'11'	2	2	11b	
P-STD_buf_bound_scale	1		1	buf_size * 1024 bytes
P-STD_buf_size_bound	13		232	buf_size = 81920 bytes
stream_id	8	1	1011 1000b	all Audio streams
'11'	2	2	11b	
P-STD_buf_bound_scale	1		0	buf_size * 128 bytes
P-STD_buf_size_bound	13		32	buf_size = 4096 bytes
stream_id	8	1	1011 1101b	Text stream
'11'	2	2	11b	
P-STD_buf_bound_scale	1		1	buf_size * 1024 bytes
P-STD_buf_size_bound	13		2	buf_size = 2048 bytes

Note 1 : In the case that the maximum bitrate of MOD file is unknown, the rate_bount field shall be set to 3FFFFFh.

ã 2001 SD Association

CONFIDENTIAL

Table 7.4-4 : Video Packet

Field	Number of bits	Number of bytes	Value	Comment
packet_start_code	24	3	00 0001h	
stream_id	8	1	1110 0000b to 1110 0011b	Video stream
PES_packet_length	16	2		
'10'	2	2	10b	
PES_scrambling_control	2		00b	
PES_priority	1		0	
data_alignment_indicator	1		0	
copyright	1		0	
original_or_copy	1		0 or 1	
PTS_DTS_flags	2			
ESCR_flag	1		0	
ES_rate_flag	1		0	
DSM_trick_mode_flag	1		0	
additional_copy_info_flag	1		0	
PES_CRC_flag	1		0	
PES_extension_flag	1		0	
PES_header_data_length	8	1	0 to 17	
'0011' or '0010'	4	5	Provider defined	(Note 1)
PTS[32...30]	3			
marker_bit	1			
PTS[29...15]	15			
marker_bit	1			
PTS[14...0]	15			
marker_bit	1			
'0001'	4	5	Provider defined	(Note 2)
DTS[32...30]	3			
marker_bit	1			
DTS[29...15]	15			
marker_bit	1			
DTS[14...0]	15			
marker_bit	1			
stuffing_byte	-	0 to 7		
'ideo data area				

Note 1 : PTS is mandatory in each V_PCK containing the first byte of access unit. "PTS[32]" shall be set to ZERO. PTS between two consecutive access units in display order shall be continuous.

Note 2 : DTS is present according to the restriction defined in ISO/IEC 13818-1. "DTS[32]" shall be set to ZERO.

ã 2001 SD Association

CONFIDENTIAL

7.4.3 Audio Pack (A_PCK)

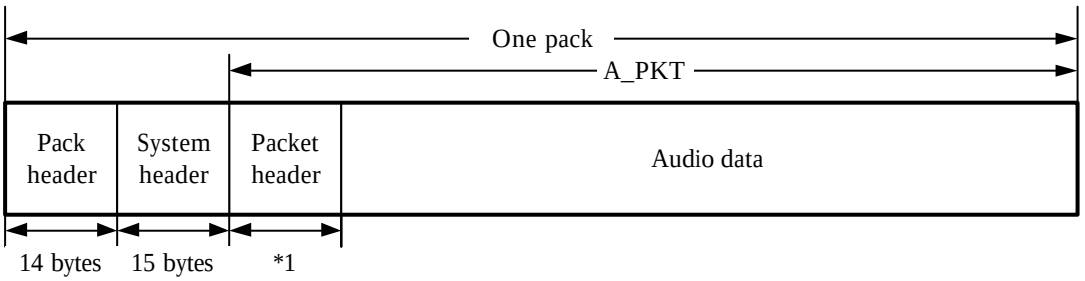
Audio pack (A_PCK) comprises a Pack header and an Audio Packet (A_PKT) as shown in Figure 7.4-4. In the case of audio only MOD file which consists of A_PCKs only, a system header shall follow the first Pack header.

An Audio access unit shall not be fragmented into more than one A_PCK. This restriction shall apply even if a user desires that the size of A_PCKs to be constant value. In this case, it shall be adjusted by either method shown in Table 7.4-1.

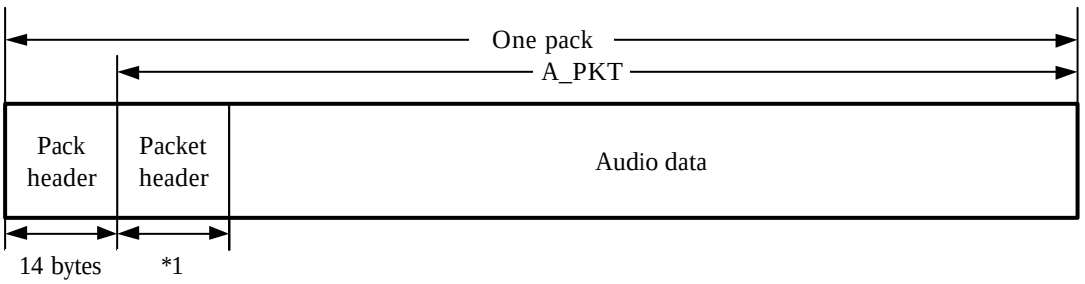
An A_PCK may contain more than one Audio access unit. The number of access units in an A_PCK shall not be changed in an MOD file except for the last A_PCK.

Figure 7.4-4 : Structure of Audio Pack

- A_PCK with System header



- A_PCK without System header



*1 : The size of Packet header is varied between 14 and 21 bytes.

Table 7.4-5 : System header in A_PCK

Field	Number of bits	Number of bytes	Value	Comment
System_header_start_code	32	4	0000 01BBh	
header_length	16	2		
marker_bit	1	3	1	
rate_bound	22		Provider defined	(Note 1)
marker_bit	1		1	
audio_bound	6	2	0 to 1	
fixed_flag	1		0 or 1	
CSPS_flag	1		0 or 1	
system_audio_lock_flag	1		1	
system_video_lock_flag	1		1	
marker_bit	1		1	
video_bound	5		0 to 1	
packet_rate_restriction_flag	1	1	0 or 1	
reserved_bit	7		7Fh	
stream_id	8	1	1011 1000b	all Audio streams
'11'	2	2	11b	
P-STD_buf_bound_scale	1		0	buf_size * 128 bytes
P-STD_buf_size_bound	13		32	buf_size = 4096 bytes

Note 1 : In case that the maximum bitrate of MOD file is unknown, the rate_bount field shall be set to 3FFFFFh.

ã 2001 SD Association

CONFIDENTIAL

Table 7.4-6 : Audio Packet

Field	Number of bits	Number of bytes	Value	Comment
packet_start_code	24	3	00 0001h	
stream_id	8	1	1100 0000b	Audio stream
PES_packet_length	16	2		
'10'	2	2	10b	
PES_scrambling_control	2		00b	
PES_priority	1		0	
data_alignment_indicator	1		0	
copyright	1		0	
original_or_copy	1		0 or 1	
PTS_DTS_flags	2		10b	
ESCR_flag	1		0	
ES_rate_flag	1		0	
DSM_trick_mode_flag	1		0	
additional_copy_info_flag	1		0	
PES_CRC_flag	1		0	
PES_extension_flag	1		0	
PES_header_data_length	8	1	5 to 12	
'0010'	4	5	Provider defined	(Note 1)
PTS[32...30]	3			
marker_bit	1			
PTS[29...15]	15			
marker_bit	1			
PTS[14...0]	15			
marker_bit	1			
stuffing_byte	-	0 to 7		
Audio data area				

Note 1 : PTS is mandatory in each A_PCK. "PTS[32]" shall be set to ZERO. PTS between two consecutive A_PCKs shall be continuous.

Note that Packet rate of this Pack may beyond the bound defined in "Constrained system parameter Program stream" of the ISO/IEC 13818-1.

ã 2001 SD Association

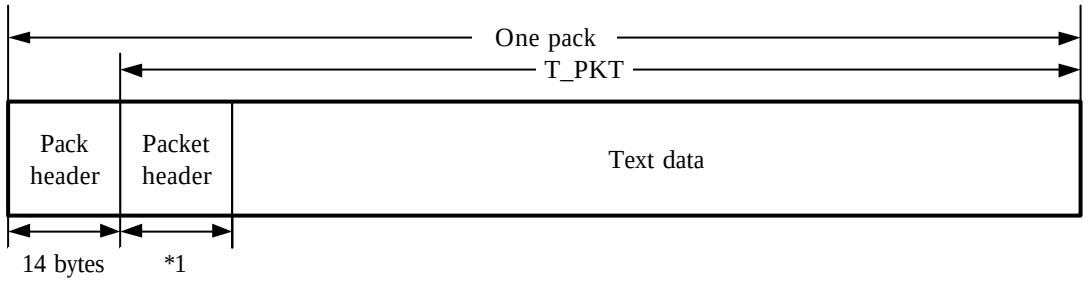
CONFIDENTIAL

7.4.4 Text Pack (T_PCK)

Text Pack (T_PCK) comprises a Pack header and an Text Packet (T_PKT) as shown in Figure 7.4-5.

A T_PCK shall consist of one complete Text access unit. If a user desires that the size of T_PCKs to be constant value, it shall be adjusted by either method shown in Table 7.4-1.

Figure 7.4-5 : Structure of Text pPack



*1 : The size of Packet header is varied between 14 and 21 bytes.

Table 7.4-7 : Text Packet

Field	Number of bits	Number of bytes	Value	Comment
packet_start_code	24	3	00 0001h	
stream_id	8	1	1011 1101b	private_stream_1
PES_packet_length	16	2		
'10'	2	2	10b	
PES_scrambling_control	2		00b	
PES_priority	1		0	
data_alignment_indicator	1		0	
copyright	1		0	
original_or_copy	1		0 or 1	
PTS_DTS_flags	2		10b	
ESCR_flag	1		0	
ES_rate_flag	1		0	
DSM_trick_mode_flag	1		0	
additional_copy_info_flag	1		0	
PES_CRC_flag	1		0	
PES_extension_flag	1		0	
PES_header_data_length	8	1	5 to 12	
'0010'	4	5	Provider defined	(Note 1)
PTS[32...30]	3			
marker_bit	1			
PTS[29...15]	15			
marker_bit	1			
PTS[14...0]	15			
marker_bit	1			
stuffing_byte	-	0 to 7		
Text data area				

Note 1 : PTS is mandatory in each T_PCK. "PTS[32]" shall be set to ZERO. PTS between two consecutive T_PCKs shall be continuous.

ã 2001 SD Association

CONFIDENTIAL

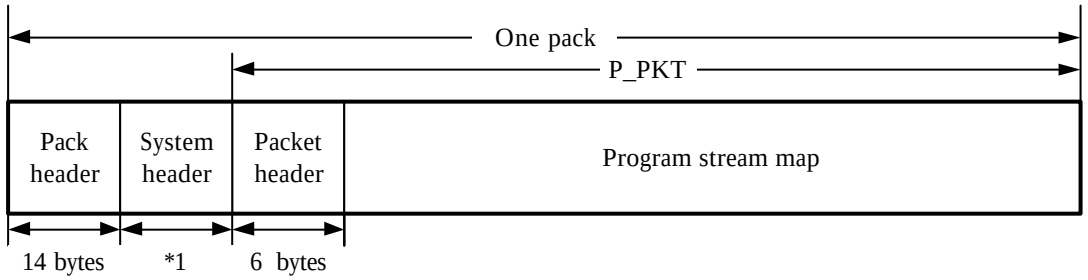
7.4.5 Program stream map Pack (P_PCK)

Program stream map Pack (P_PCK) comprises a Pack header, System header and Program stream map as shown in Figure 7.4-6.

If MOD file contains P_PCK, the P_PCK shall be the first Pack in the MOD file, and any PCK shall not be present thereafter (i.e., only one P_PCK at the beginning of MOD file is permitted) .

A complete program stream map shall be accommodated in a P_PCK.

Figure 7.4-6 : Structure of Program stream map Pack



*1 : The size of System header varies between 15 and 21 bytes. System header of P_PCK is shown in Table 7.4-3.

Table 7.4-8 : Program stream map Packet

Field	Number of bits	Number of bytes	Value	Comment
packet_start_code	24	3	00 0001h	
map_stream_id	8	1	1011 1100b	program_stream_map
program_stream_map_length	16	2		
current_next_indicator	1	1	1	
reserved	2		11b	
program_stream_map_version	5		0 0000b	
reserved	7	1	111 1111b	
marker_bit	1		1	
program_stream_info_length	16	2	00h	
elementary_stream_map_length	16	2		
here comes the following (a), (b) or (c) portions				
CRC_32	32	4		

ã 2001 SD Association

CONFIDENTIAL

Table 7.4-9: (a) Video portion of Program stream map Packet

Field	Number of bits	Number of bytes	Value	Comment
stream_type	8	1	0001 0000b	ISO/IEC 14496-2 Video
elementary_stream_id	8	1	1110 00**b	Video stream
elementary_stream_info_length	16	2	00h	no descriptor

** = 00, 01, 10 or 11

Table 7.4-10: (b) Audio portion of Program stream map Packet

Field	Number of bits	Number of bytes	Value	Comment
stream_type	8	1	0000 1111b or 1000 0001b or 1000 0010b or 1000 0011b	ISO/IEC 13818-7 Audio Windows Media Audio G.726 AMR
elementary_stream_id	8	1	1100 0000b	Audio stream
elementary_stream_info_length	16	2	03h	
descriptor_tag	8	1	40h	audio_frame_info_descriptor
descriptor_length	8	1	01h	
reserved	3	1	111b	
number_of_audio_frames	5			(Note 1)

Note 1 : “number_of_audio_frames” describes the number of audio frames in an A_PKT having the elementary_stream_id specified. The value 0 indicates the number of audio frames in an A_PKT is not defined and it may be variable.

Table 7.4-11: (c) Text portion of Program stream map Packet

Field	Number of bits	Number of bytes	Value	Comment
stream_type	8	1	1000 1000b	Text stream
elementary_stream_id	8	1	1011 1101b	private_stream_1
elementary_stream_info_length	16	2	00h	no descriptor

ã 2001 SD Association

CONFIDENTIAL

7.5 Player Reference Model

The player reference model is shown in Figure 7.5-1.

Input Buffer for P_PCK is not defined since P_PCK has no timestamp. When DEMUX receives a whole P_PCK, it is immediately sent to PSM decoder and decoded accordingly.

Input Buffer size in each target decoder and the timing of transfer from the Input Buffer to the decoder are shown in Table 7.5-1.

Table 7.5-1 : Input Buffer definitions

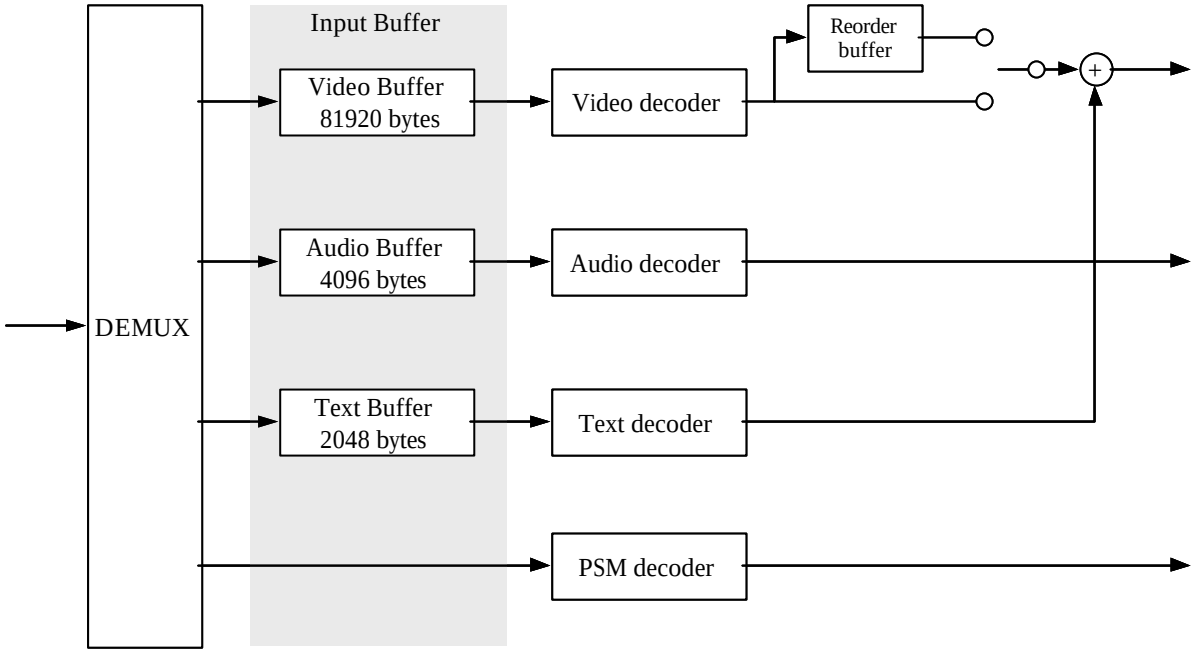
Target Decoder	Input Buffer size	Removale Timing from Input Buffer
Video	81920 bytes	When system time clock reaches PTS (in case of B VOP) or DTS (in case of I or P VOP).
Audio	4096 bytes	When system time clock reaches the presentation time of audio access unit. For the first access unit in each A_PCK, presentation time is given by PTS.
Text	2048 bytes	When system time clock reaches PTS.

Only the elementary stream data is passed to the Input Buffers. Bytes present in the Pack headers, system headers or Packet headers shall not delivered to any of the Input Buffers.

In case of multiple MPEG-4 Video streams, Input Buffer is shared among the Video streams.

ã 2001 SD Association

Figure 7.5-1 : Player Reference model



ã 2001 SD Association

Annex

Annex A. Mandatory Item of SD-Video Specification

In the following descriptions, the items described with "shall" are mandatory. The items described with "may" are optional.

Annex A.1. MGR_DATA

This file is mandatory and whether the contained fields are mandatory or not are as follows.

Table Annex A-1 : MGR_DATA

Field	in Card	by Player	Value in case not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency.
DataSize	Mandatory	shall be used	-	Player shall check for consistency.
Version	Mandatory	shall be used	-	Player shall check to know whether the data can be understood or not.
ResumeMarker	Option	may be used	all zero	In case not support, the player fills this field with value zero.
TextInfoAttr1	Option	may be used	zero	
TextInfoAttr2	Option	may be used	zero	
TextInfo	Option	may be used	all zero	

ã 2001 SD Association

CONFIDENTIAL

Annex A.2. PRG_MGR

This file is mandatory and whether the contained fields are mandatory or not are as follows.

Table Annex A-2 : PRG_MGR

Field	in Card	by Player	Value in case not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency.
DataSize	Mandatory	shall be used	-	Player shall check for consistency.
Version	Mandatory	Shall be used	-	Player shall check to know whether the data can be understood or not.
PlaybackDuration	Mandatory	may be used	-	
NumPrg	Mandatory	shall be used	-	
PrgAttrTbl	NumPrg entries shall exist	shall be used	-	
Version	Mandatory	shall be used	-	
PrgID	Mandatory	shall be used	-	
PlaybackDuration	Mandatory	may be used	-	
Attribute	Mandatory	shall be used	-	Players shall check Scene bit and Protect bit.
Profile	Mandatory	shall be used	-	Players shall check the profile to know whether the data can be playback or not.
VideoAttr	Mandatory	shall be used		Players shall check the profile to know whether the data can be playback or not.
AudioAttr	Mandatory	shall be used		Players shall check the profile to know whether the data can be playback or not.
RepOffset	Option	may be used	all zero	
TextInfoAttr1	Option	may be used	zero	
TextInfoAttr2	Option	may be used	zero	
TextInfo	Option	may be used	all zero	
NumMoInfo	Mandatory	shall be used	-	
NumMarker	Mandatory	may be used	-	
NumExtraData	Mandatory	may be used	-	

ã 2001 SD Association

CONFIDENTIAL

Annex A.3. PRG_INFO

This file is mandatory and whether the contained fields are mandatory or not are as follows.

Table Annex A- 3: PRG_INFO

Field	in Card	by Player	Value in case not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency.
DataSize	Mandatory	shall be used	-	Player shall check for consistency.
Version	Mandatory	Shall be used	-	Player shall check to know whether the data can be understood or not.
Attribute	Mandatory	shall be used	-	Players shall check Scene bit and Protect bit.
Profile	Mandatory	shall be used	-	Players shall check the profile to know whether the data can be playback or not.
VideoAttr	Mandatory	shall be used		Players shall check the profile to know whether the data can be playback or not.
AudioAttr	Mandatory	shall be used		Players shall check the profile to know whether the data can be playback or not.
NumMoInfo	Mandatory	shall be used	-	
NumMarker	Mandatory	may be used	-	
NumExtraData	Mandatory	may be used	-	
MoInfoTbl	NumMoInfo entries shall exist	may be used	-	
MarkerTbl	NumMarker entries shall exist	may be used	-	
ExtraDataTbl	NumExtraData entries shall exist	may be used		

ã 2001 SD Association

CONFIDENTIAL

Annex A.4. PLST_MGR

This file is optional. The Recorders that don't support playlist may not create or update this file, and the players that don't support playlist may not read this file, too. But the recorders that create or update this file and the players that read this file shall write or read according to the following conditions.

Table Annex A-4 : PLST_MGR

Field	in Card	by Player	Value in case not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency.
DataSize	Mandatory	shall be used	-	Player shall check for consistency.
Version	Mandatory	shall be used		
NumPlstInfo	Mandatory	shall be used	-	
PlstInfoTbl	NumPlstInfo entries shall exist		-	
DataType	Mandatory	shall be used	-	Player shall check for consistency.
DataSize	Mandatory	shall be used	-	Player shall check for consistency.
Version	Mandatory	shall be used	-	Player shall check to know whether the data can be understood or not.
PlstID	Mandatory	shall be used	-	
PlaybackDuration	Mandatory	may be used	-	
Attribute	Mandatory	shall be used	-	Players shall check Protect bit.
RepOffset	Option	may be used	all zero	
TextInfoAttr1	Option	may be used	zero	
TextInfoAttr2	Option	may be used	zero	
TextInfo	Option	may be used	all zero	
NumPrgID	Mandatory	shall be used	-	
NumMarker	Mandatory	shall be used	-	
PrgSectionTbl	NumPrgID entries shall exist	shall be used	-	
MarkerTbl	NumMarker entries shall exist	may be used	-	

ã 2001 SD Association

CONFIDENTIAL

Annex A.5. MOI

This file shall exist corresponding to the *.MOD file. Whether the contained fields are mandatory or not are as follows.

Table Annex A-5 : Header of MOI

Field	in Card	by Player	Value in case not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency.
DataSize	Mandatory	shall be used	-	Player shall check for consistency.
CreationTime	Option	may be used	zero	
Playback Duration	Mandatory	may be used	-	In case of using RefMoiTbl, this value shall be equal to the value of corresponding MoiDuration.
StrmAttr	Mandatory	shall be used	-	
TstType	Mandatory	shall be used	-	

In case of TstType==0, no fields shall follow.

In case of TstType==1, following fields shall follow.

Table Annex A-6 : Tail of MOI in case of TstType==1

Field	in Card	by Player	Value in case not supported	Note
TstInterval	Mandatory	shall be used	-	
FrameTimeNmr	Mandatory	shall be used	-	
FrameTimeDnm	Mandatory	shall be used	-	
NumModui	Mandatory	shall be used	-	
ModuiTbl	NumModui entries shall exist	shall be used	-	
NumTseInfo1	Mandatory	shall be used	-	
TseInfoTbl1	NumTseInfo 1entries shall exist	shall be used	-	

In case of TstType==2, following fields shall follow the Table Annex A-5.

Table Annex A-7 : Tail of MOI in case of TstType==2

Field	in Card	by Player	Value in case not supported	Note
TstInterval	Mandatory	shall be used	-	
NumTseInfo2	Mandatory	shall be used	-	
TseInfoTbl2	NumTseInfo 2entries shall exist	shall be used	-	

ã 2001 SD Association

CONFIDENTIAL

In case of TstType==3, following fields shall follow the Table Annex A-5.

Table Annex A-8 : Tail of MOI in case of TstType==3

Field	in Card	by Player	Value in case not supported	Note
FrameTimeNmr	Mandatory	shall be used	-	
FrameTimeDnm	Mandatory	shall be used	-	
PackSize	Mandatory	shall be used	-	
NumSkippedPack	Mandatory	shall be used	-	
NumFrame	Mandatory	shall be used	-	

ã 2001 SD Association

CONFIDENTIAL

Annex A.6. Media Object Data File

Media Object Data File (MOD file) is optional in Card. At least one file format among Media Object Data File and Advanced Streaming Format File shall be playable by Player.

Annex A.7. Advanced Streaming Format File

Advanced Streaming Format File can be used instead of MOD file and MOI file. Advanced Streaming Format File is optional in Card. At least one file format among Media Object Data File and Advanced Streaming Format File shall be playable by Player. In case Advanced Streaming Format Files are recorded in a Card, the bitstreams in the files shall comply with the Mobile Video Profile constraints. In case the interoperability with the mobile cellular phone applications is required, the recorded video bitstreams in the Card are recommended to comply with MPEG-4 Visual Simple Profile / Level 1 restricted.

Annex A.8. Program Title

There are two fields where a Program Title can be stored. They are TextInfo field in PRG_ATTR table in Program Manager File and Attribute field in ExtraDataTbl in Program Information File. In TextInfo field, there are two method to describe the Title, that is by TextInfoAttr1 or by TextInfoAttr2. It is recommended to use primarily TextInfoAttr2, secondarily TextInfoAttr1 and thirdly Attribute in ExtraDataTbl in order to display the Program Title.

ã 2001 SD Association

CONFIDENTIAL

Annex A.9. Player Functions

The following table lists the functions supported by players.

Table Annex A- 9 : Functions supported by players

Function	Player
Playlist Edit	optional
Program Playback	mandatory
Playlist Playback	optional
Random Access Playback (with Time Search Table)	optional
Fast Forward Playback	optional
Fast Reverse Playback	optional
Forward Skip	optional
Reverse Skip	optional
Separately-Recorded Audio Playback	mandatory
Audio Replacement Playback	mandatory

A.9.1 Playlist Edit()

Playlist Edit() edits PLST_INFO according to user intention.

A.9.2 Program Playback()

Program Playback() starts playback of program #n specified by user.

A.9.3 Playlist Playback()

Playlist Playback() starts playback of programs according to the PLST_INFO specified by user.

A.9.4 Random Access Playback()

Random Access Playback() starts playback of a program at the time point specified by user.

A.9.5 Fast Forward Playback()

Fast Forward Playback() plays back only entry frames of programs in forward way.

A.9.6 Fast Reverse Playback()

Fast Reverse Playback() plays back only entry frames of programs in backward way.

A.9.7 Forward Skip()

Forward Skip() stops the current playback and restart playback from the position specified by the next Marker point.

If there is no more Marker point, just ignore the Forward Skip() or end the playback.

ã 2001 SD Association

CONFIDENTIAL

A.9.8 Reverse Skip()

Reverse Skip() stops the current playback and restart playback from the position specified by the previous Marker point.
If there is no more Marker point, just ignore the Reverse Skip() or end the playback.

A.9.9 Separately-Recorded Audio Playback()

Separately-Recorded Audio Playback() plays back the Separately-Recorded Audio Data synchronously with the video streams in the corresponding Movie Data.

A.9.10 Audio Replacement Playback()

Audio Replacement Playback() plays back Replaced Audio Data instead of the audio stream multiplexed in the corresponding Movie Data or the corresponding Separately-Recorded Audio Data.

ã 2001 SD Association

CONFIDENTIAL

Annex A.10. Restrictions of Maximum Data Size

This section describes the restrictions of maximum data size in order to limit the required memory size.

A.10.1 Restrictions of Maximum Number of Objects

The restriction of maximum number of objects are as follows.

Table Annex A- 10 : Restrictions of Maximum Number of Objects

Object	Maximum number
MGR_DATA	1
PRG_MGR	1
PRG_ATTR	99
PRG_INFO	1 per Program
MoInfoTbl	99 per PRG_INFO
ExtraData	99 per PRG_INFO
PLST_MGR	1
PLST_INFO	99
PrgSectionTbl	99 per PLST_INFO
MarkerTbl	99 per PRG_INFO or per PLST_INFO
Packet	1 per Pack (Note 1)

Note 1: One Padding Packet may coexist with a Packet in a Pack.

A.10.2 Restrictions of Maximum Data Size of Information Tables

The restriction of maximum data size of information tables are as follows.

Table Annex A- 11: Restrictions of Maximum Data Size of Information Tables

Table name	Maximum data size
MGR_DATA	Fixed 256 Bytes
PRG_MGR	Variable
PRG_ATTR	Fixed 280 Bytes
PLST_MGR	Max. 32 k Bytes
PLST_INFO	Variable
PRG_INFO	Variable
MOI	Variable
ExtraData	Max. 10 k Bytes

ã 2001 SD Association

CONFIDENTIAL

Annex B Operation Example

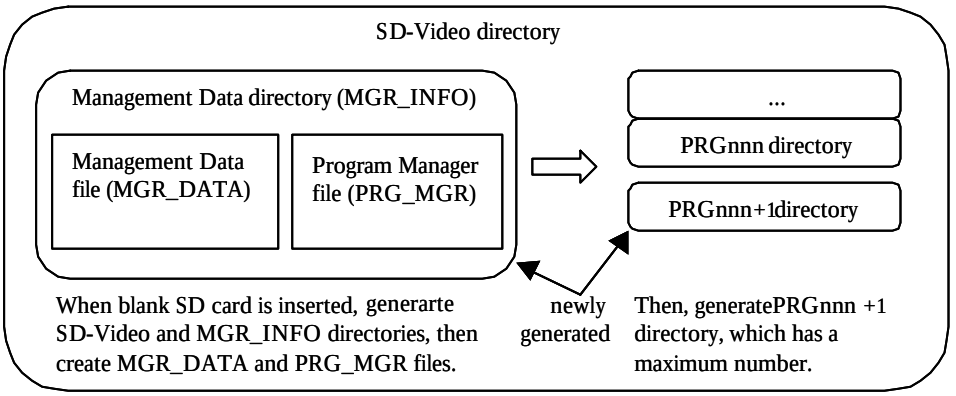
In this section, typical operation process examples for recording, editing, playback, trick play and Audio-Replacement on the SD-Video contents are described.

Annex B.1. Recording Procedure

When a blank SD Card is inserted in an SD-Video recorder, an SD-Video directory, a Management Data Directory (MGR_INFO), a Management Data File (MGR_DATA) and a Program Manager File (PRG_MGR) are generated and initialized.

When Rec. mode is selected by a user, Program directories are checked and the maximum program number (PRGnnn) is detected. Then, a new Program directory with the maximum number +1 (PRGnnn+1) and Program Information File (PRGnnn+1.PGI) are generated. Then, Recording Process is executed as follows.

Figure Annex B-1 : When a blank card is inserted, create directories and files

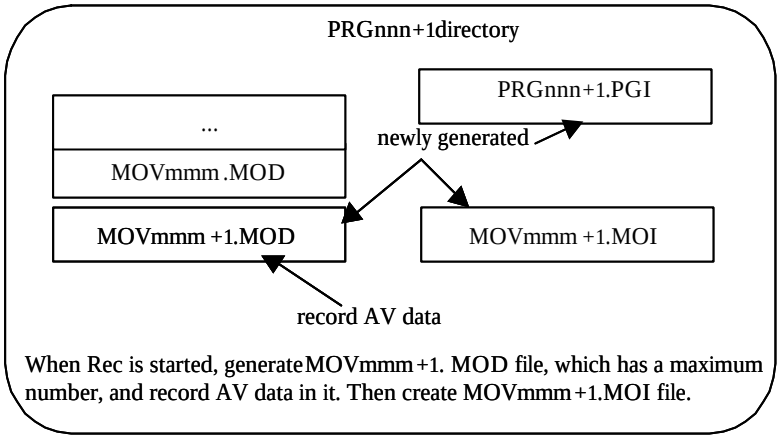


1. When Rec. button is pressed down, Media Object Data Files (MOVmmm.MOD) in the directory (PRGnnn+1) are checked and the maximum number “mmm” is detected. Then, a new Media Object Data Files (MOVmmm+1.MOD) with the maximum number +1 is generated. The AV data is recorded in the Media Object Data File (MOVmmm+1.MOD). A Media Object Information File (MOVmmm+1.MOI) with the same number is also created.

ã 2001 SD Association

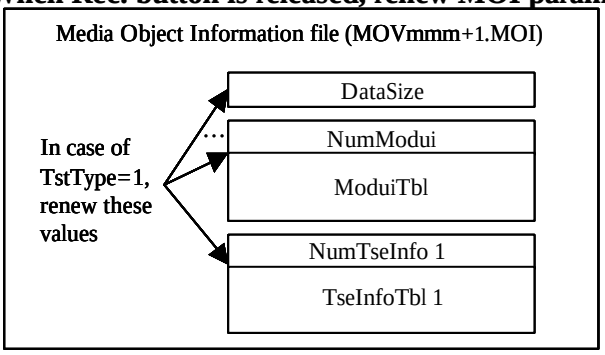
CONFIDENTIAL

Figure Annex B-2 : When Rec is Started, create new MOD file and record AV data



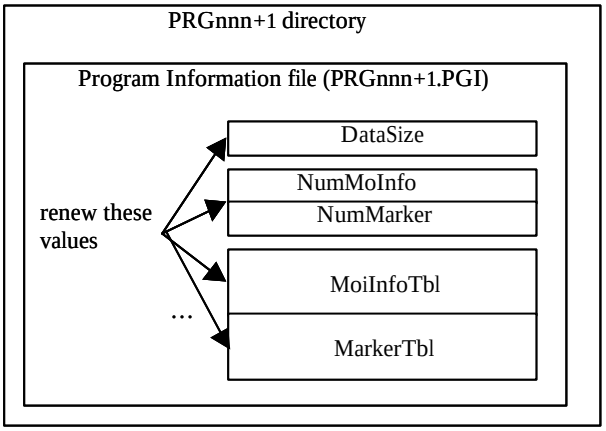
2. When Rec button is released, data recording is stopped and the contents of MOVmmm+1.MOI file are renewed. The renewed contents are Data Size (DataSize), Creation Time, Playback Duration, Number of MODU Information (NumModui), MODU Information Table (ModuiTbl), Number of Time Search Entry Infomation 1 (NumTseInfo1) and Time Search Entry Information Table 1 (TseInfoTbl1). In case that MODU is not used, Number of Time Search Entry Information 2 (NumTseInfo2) and Time Search Entry Information Table 2 (TseInfoTbl2) are renewed instead of the above mentioned contents.

Figure Annex B-3 : When Rec. button is released, renew MOI parameters



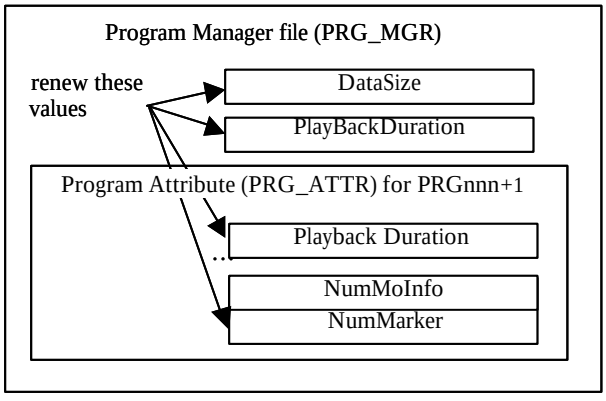
3. Then, the contents of the Program Information File (PGRnnn+1.PGI) is renewed. The renewed contents are DataSize, NumMoInfo, NumMarker, NumExtraData, MoInfoTbl, MarkerTbl, and if applicable, ExtraDataTbl.

Figure Annex B-4 : Then, renew Program Information File in Program nnn+1 directory



4. Then the contents of PRG_MGR file are renewed. The renewed contents are Data Size (DataSize), Playback Duration (PlaybackDuration). PlaybackDuration, NumMoInfo and NumMarker in the PRG_ATTR referred by the PRGnnn+1 are also renewed.

Figure Annex B-5 : Then, renew PRG_MGR and PRG_ATTR for Program nnn+1

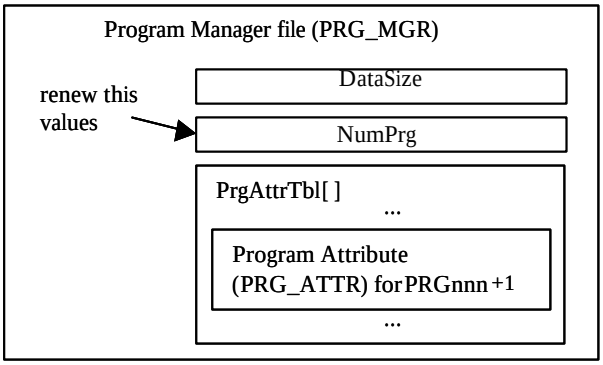


ã 2001 SD Association

CONFIDENTIAL

5. When Rec. button is pressed down again, the same procedure is performed increasing the Media Object number (mmm+2, mmm+3, …).
6. When Rec. mode is released, Number of Programs (NumPrg) in PRG_MGR is renewed.

Figure Annex B-6 : When Rec. mode is released, renew NumPro in PRG_MGR



Annex B.2. Editing Procedure

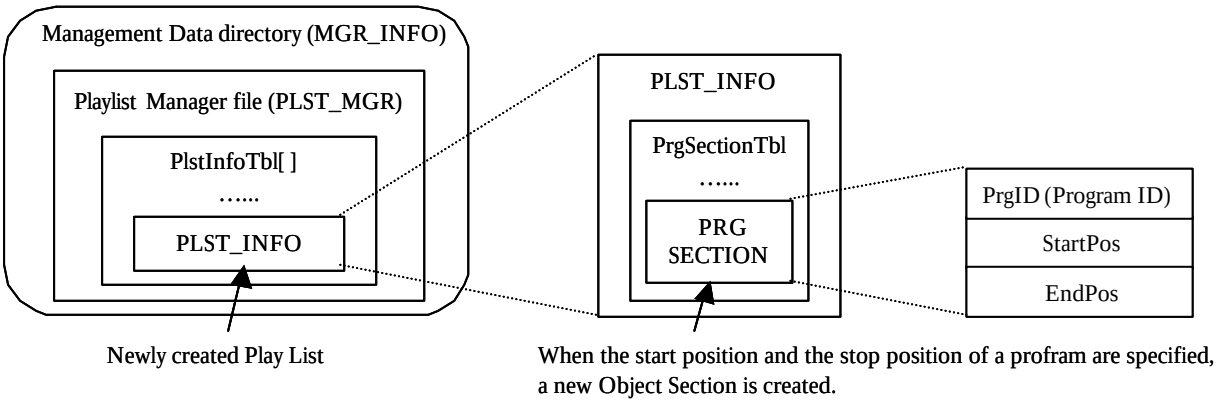
Editing is virtually done by editing Playlist Information (PLST_INFO) in Playlist Manager File (PLST_MGR). Only when a deleting area covers a whole Program completely, the Program directory is actually deleted. All files in the directory are also deleted. If some external files or directories are linked to the deleted Program by an application program, the application program must delete those files or directories.

Annex B.2.1. Creating a new Playlist

When a user creates a new Playlist, all recorded program names and their duration time are displayed on a display screen from the information in Program Manger File (PRG_MGR) and the Playlist creation process is executed as follows.

1. A new Playlist Information (PLST_INFO) is created in Playlist Manager File (PLST_MGR) and initialized.
2. When the user selects a program (PRGnnn) to register with the new Playlist Information (PLST_INFO) and specifies the starting time and stopping time in the Program, a new Program Section Structure (PRGSECTION) is added to PrgSectionTbl in the PLST_INFO. The Object ID (ObjID), Start Position (StartPos) and End Position (EndPos) in the PRGSECTION are registered with each values of PRGnnn, starting time and stopping time, respectively.

Figure Annex B-7 : New Playlist creation



3. Then, renew the Data Size (DataSize), Playback Duration (PlaybackDulation) and Number of referred Program (NumPrgID) in the PLST_INFO, then renew the Data Size (DataSize) of PLST_MGR.
4. Repeat the above registration procedure for the next program if specified by the user.

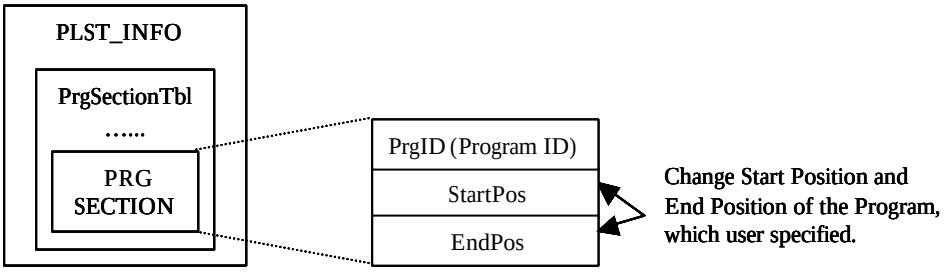
ã 2001 SD Association

CONFIDENTIAL

Annex B.2.2 Editing Existing Playlist

1. When the user changes an existing Playlist, the selected Playlist Information (PLST_INFO) is displayed on the display screen.
2. Then, user specifies a Program with new Starting time and/or Stopping time.
3. Then, the Start Position (StartPos) and End Position (EndPos) in the PRGSECTION of the specified Program are renewed through the same procedure described above.

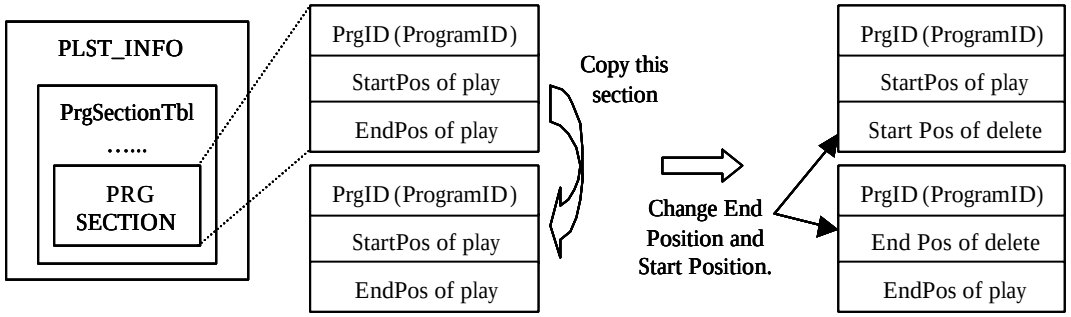
Figure Annex B-8 : Playlist Editing



Annex B.2.3. Deletion

1. When the user deletes some part of a program, specifying the starting time and stopping time of deletion area, the target PRGSECTION is duplicated and concatenated.
2. Then, the EndPos of the leading PRGSECTION is rewritten with the starting time of deletion area and the StartPos of the trailing PRGSECTION is rewritten with the stopping time of the deletion area.
3. Then, renew the Data Size (DataSize), Playback duration (PlaybackDulation) and Number of referred Program (NumPrgID) in the PLST_INFO, then renew the Data Size (DataSize) and total Playback duration (PlaybackDuration) in the PLST_MGR.

Figure Annex B-9 : Partial Deletion



ã 2001 SD Association

CONFIDENTIAL

Annex B.3. Playback Procedure

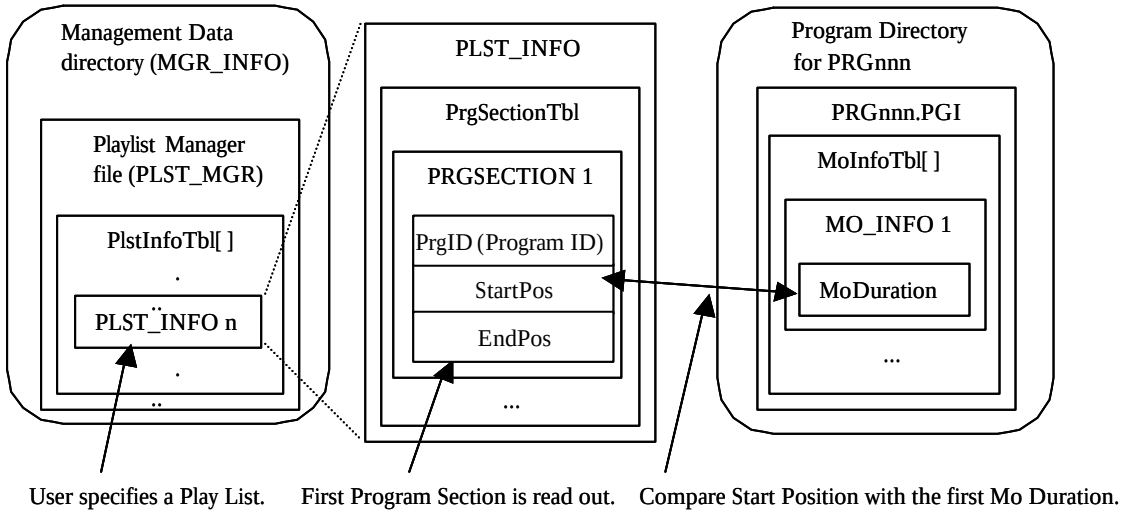
Four types of Playback mode are described below. They are Playlist Playback (Normal Playback), Random Access Playback, Fast Forward Playback and Fast Reverse Playback.

Annex B.3.1. Playlist Playback

In case a user specifies a Playlist and starts Playback, the specified Playlist Information (PLST_INFO) in Playlist Manager File (PLST_MGR) is used for the playback process.

1. The PRGSECTIONs in the PrgSectionTbl are read out from the beginning. The first Program (PRGnnn) is specified by the Program ID (PrgID) in the first PRGSECTION.
2. Then the value of Start Position (StartPos) is compared with the MoDuration of the first MO_INFO in the MoInfoTbl in the related Program Information File in the Program nnn directory.

Figure Annex B-10 : Playlist Playback: Compare StartPos of the first PLST_INFO with the MoDuration in the first MO_INFO in PRGnnn.PGI.



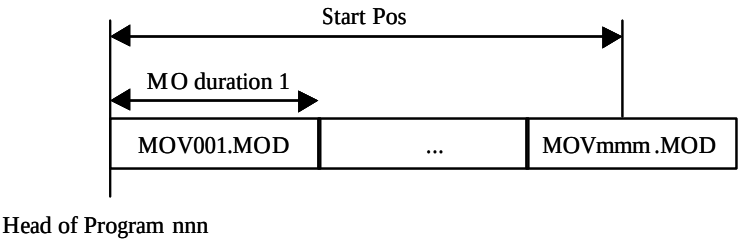
3. If the StartPos value is larger than the MoDuration value, then the value of the MoDuration is subtracted from the StartPos value. Also, subtract the MoDuration value from the End Position (EndPos). Then the residual of StartPos is compared with the MoDuration value of the next MO_INFO.

ã 2001 SD Association

CONFIDENTIAL

4. The same operation is repeated until the residual of StartPos becomes smaller than the MoDuration value of the next MO_INFO.

Figure Annex B-11 : Then, Compare the residue of Start Position with the next MO Durations until it becomes smaller than the next MODuration.



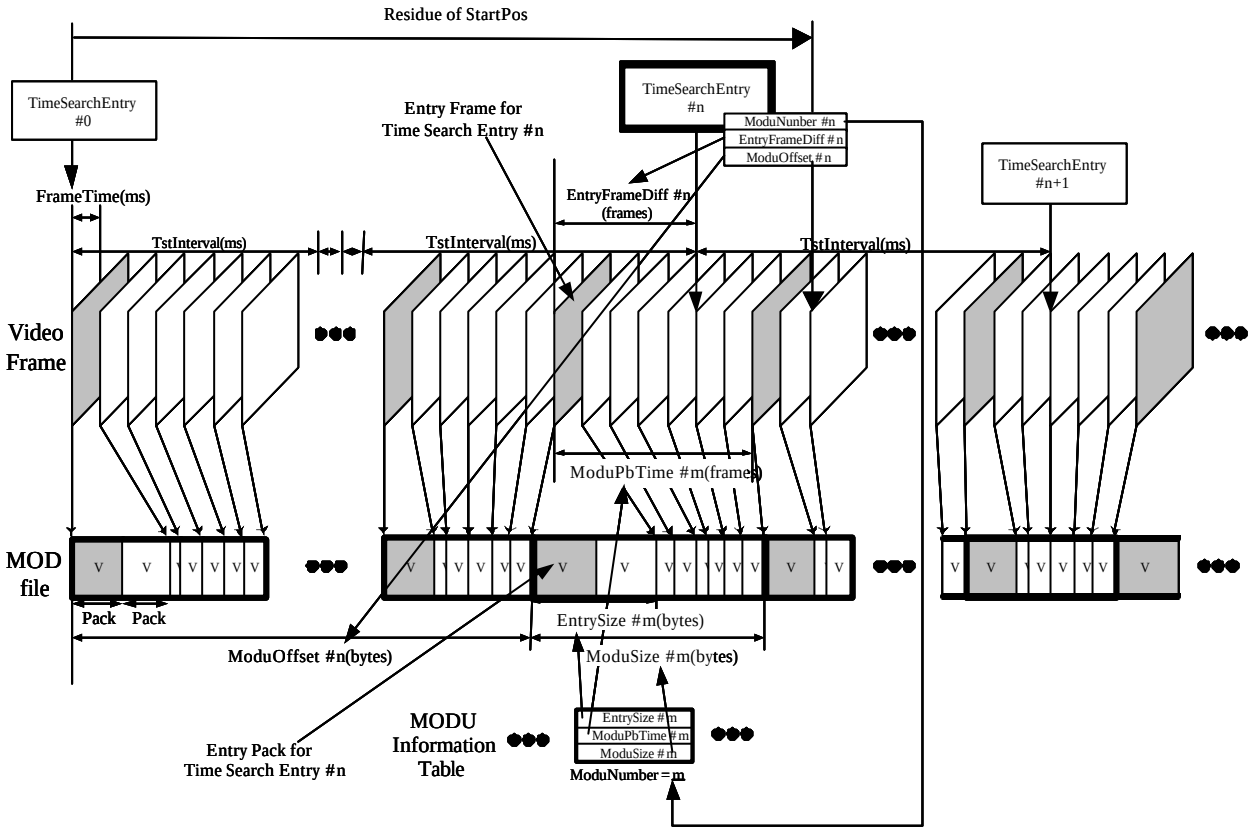
5. Clear an entry pointer register.
6. Repeat the following procedure until the residual of StartPos becomes smaller than the Time Interval of Time Search Entry (TstInterval) in the MOVmmm.MOI.
   Subtract the Time Interval value (TstInterval) from the residual value of StartPos and also from the residual value of EndPos and increase the entry pointer register value by one.

ã 2001 SD Association

CONFIDENTIAL

7. Then, get the Offset of MODU (ModuOffset) of the TSE_INFO1 in the Time Search Entry Information Table 1(TseInfoTbl1), which is indicated by the entry pointer register value. Subtract TstInterval value from the residual value of EndPos.

Figure Annex B-12 : Then, Compare the residue of Start Position with the next TstInterval until it becomes smaller than the next TstInterval.

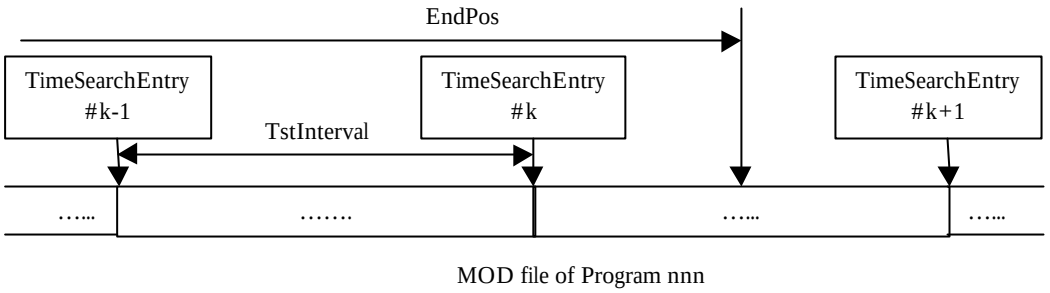


8. Then, start reading-out of the Media Object Data File (MOVmmm.MOD) from the point which is specified by the ModuOffset and feed the data to MPEG decoder.
9. Count the number of frames read out, until the number becomes equal to the value of EntryFrameDiff of the TSE_INFO1. Then, newly count the number of frames until the total time of the frames becomes larger than the residual value of Start Position. Then, release the output of the MPEG decoder to a display screen and also to speakers if video and audio streams are multiplexed in the MOVmmm.MOD. Thus, Playback starts.
10. Continue Playback operation with subtracting the TstInterval value from the residual value of EndPos until the residual becomes minus.

ã 2001 SD Association

CONFIDENTIAL

Figure Annex B-13 : Stop when the residue of End Position become minus.



11. Then stop the Playback of the Program (PRGnnn) and move to the next Program specified in the Playlist Information (PLST_INFO).
12. Continue the above mentioned Playback operation until all Programs specified in the PLST_INFO is played back.

In case the Time Search Table Type (TstType) indicates TstType==2 (no MODU Information), StartPos is converted to a Time Search Entry #n with the similar manner described above, looking up the Time Search Entry Information Table 2 (TseInfoTbl2).

In case the Time Search Table Type (TstType) indicates TstType==3 (fixed frame data size audio), StartPos is more directly converted to a Packet number #n, by dividing the StartPos value by FrameTimeNmr and by multiplying FrameTimeDnm, then by dividing the result by NumFrame.

If Scene Description File is specified in the Attribute in the Program Information File (PRGnnn.PGI), Playlist Playback is executed according to the Scene Description File.

ã 2001 SD Association

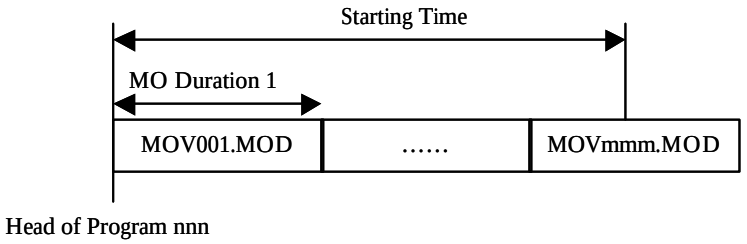
CONFIDENTIAL

Annex B.3.2. Random Access Playback

In case a user directly specifies a Program (PRGnnn) and its Starting Time then starts Playback, the Playback process is executed as follows.

1. The Starting Time is compared with the MoDuration of the first MO_INFO in the MoInfoTbl in the Program Information File in the specified Program File (PRGnnn).
2. If the Starting Time value is larger than the MoDuration value, then the value of the MoDuration is subtracted from the Starting Time value. Also, subtract the MoDuration value from the End Position (EndPos) in the PRGSECTION of the specified Program in a default Playlist Information (PLST_INFO) in Playlist Manager (PLST_MGR). Then the residual of Starting Time is compared with the MoDuration value of the next MO_INFO.

Figure Annex B-14 : Compare Starting Time with MoDurations in Program nnn.

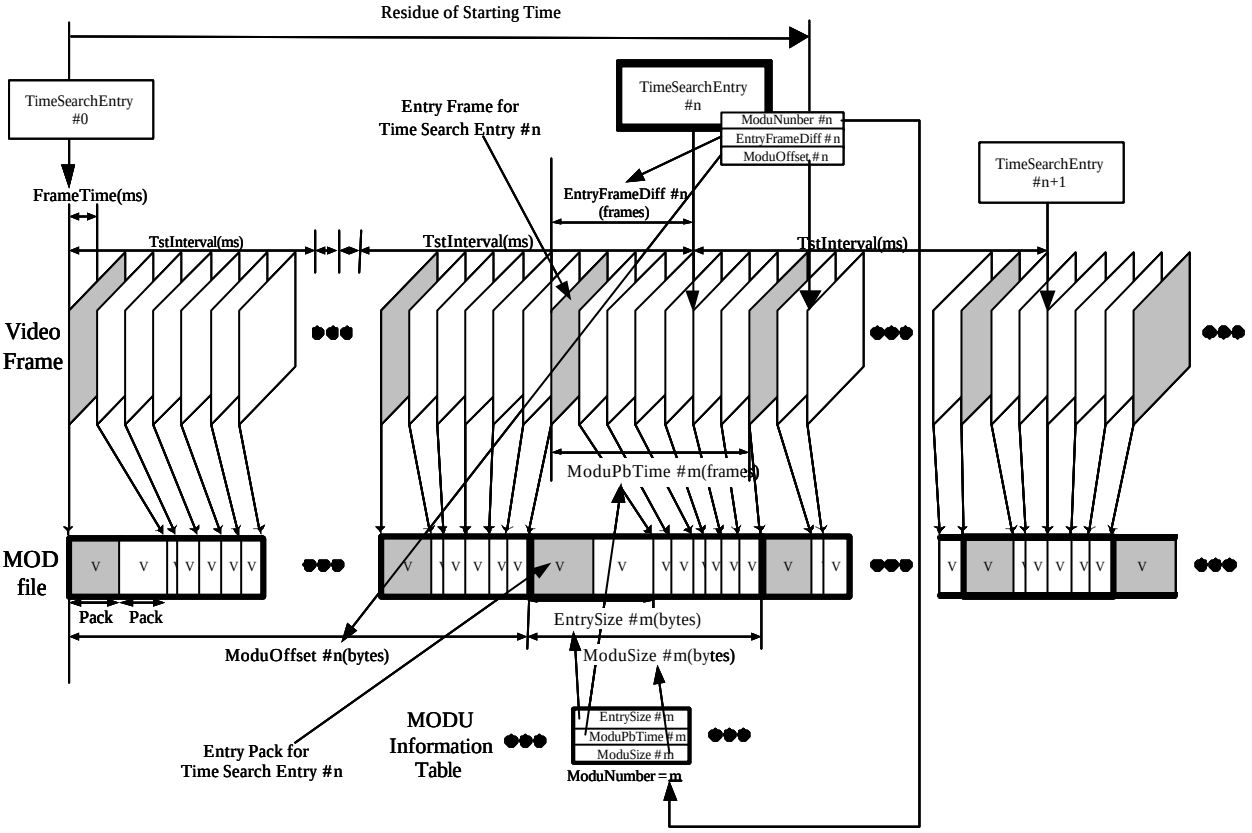


3. The same operation is repeated until the residual of starting time becomes smaller than the MoDuration value of the next MO_INFO.
4. Clear an Entry Pointer Register which is reserved by the player application software.
5. Repeat the following process until the residual of Starting Time becomes smaller than the Time Interval of Time Search Entry (TstInterval) of the MOVmmm.MOI.
6. Subtract the Time Interval value (TstInterval) from the residual value of Starting Time and also from the residual value of EndPos and increase the Entry Pointer Register value by one.
7. Then, get the Offset of MODU (ModuOffset) in the Time Search Entry Information Table 1 (TseInfoTbl1), which is specified by the Entry Pointer Register value. Subtract TstInterval value from the residual value of EndPos.

ã 2001 SD Association

CONFIDENTIAL

Figure Annex B-15: Then, Compare the residue of Starting Time with the next TstInterval until it becomes smaller than the next TstInterval.

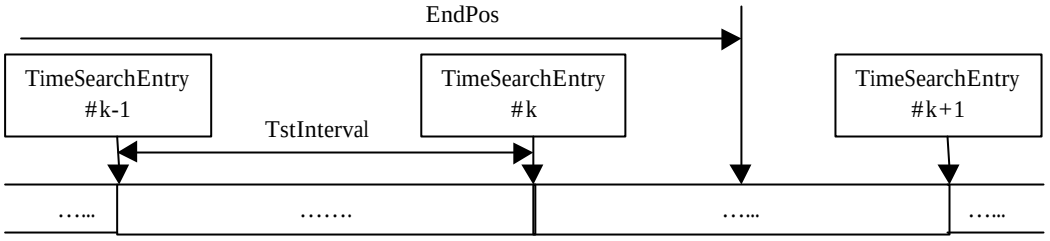


8. Then, start reading-out of the Media Object Data File (MOVmmm.MOD) from the point which is specified by the ModuOffset and feed the data to MPEG decoder.
9. Count the number of frames read out, until the number becomes equal to the value of EntryFrameDiff of the TSE_INFO1. Then, release the output of the MPEG decoder to a display screen and also to speakers if video and audio streams are multiplexed in the MOVmmm.MOD. Thus, Random Access Playback starts.
10. Continue Playback operation with subtracting the TstInterval value from the residual value of EndPos until the residue becomes minus.

ã 2001 SD Association

CONFIDENTIAL

Figure Annex B-16 : Continue Playback until the residue of End Position becomes minus.



MOD file of Program nnn

11. Then stop the Playback of the Program (PRGnnn) and move to the next Program specified in the Playlist Information (PLST_INFO).
12. Continue the above mentioned Playback operation until all Programs specified in the PLST_INFO is played back.

ã 2001 SD Association

CONFIDENTIAL

Annex B.3.3. Fast Forward Playback

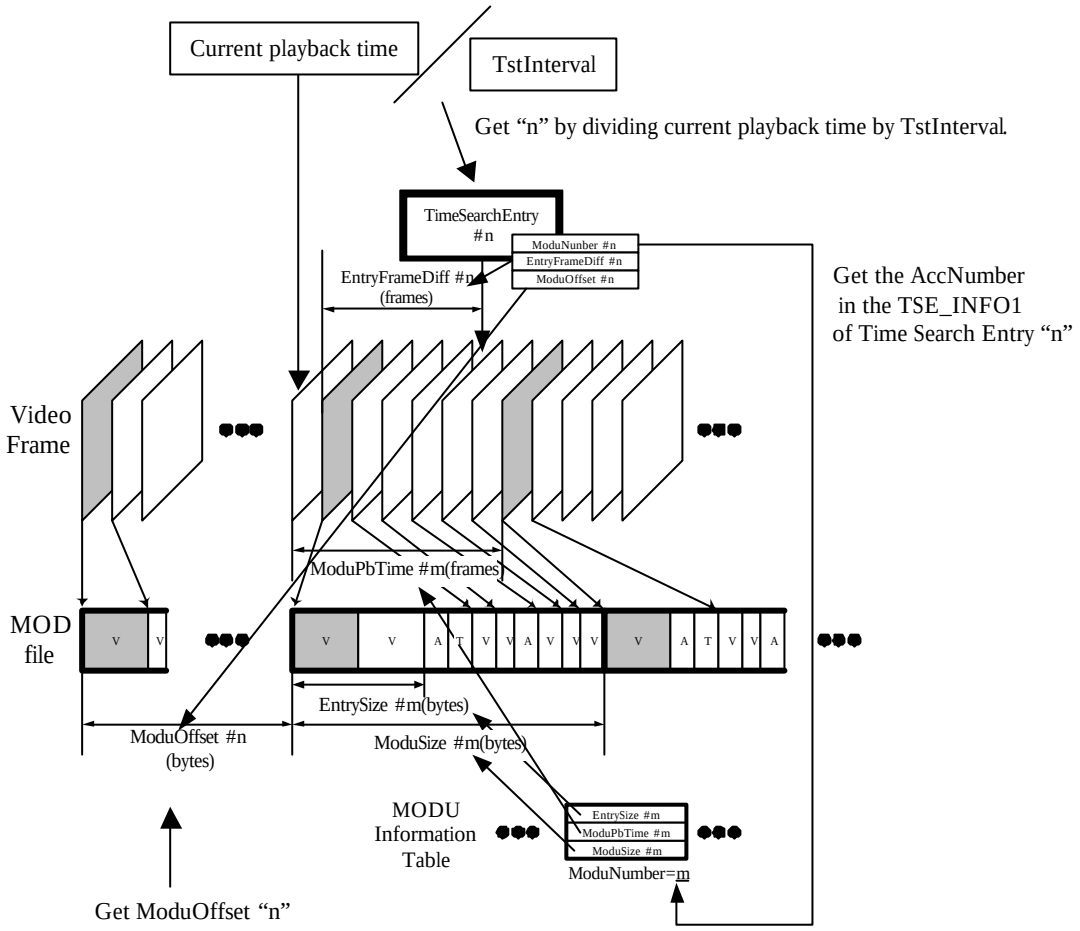
When a user presses FF (Fast Forward Playback) button during Playlist Playback process or Randon Access Playback process, the following process starts.

1. The current playback Program and the time should be kept in the Resume Marker in MGR_DATA. It is used for re-starting the playback process from where user stopped playback the last.
2. The re-starting Program is identified by the program ID stored in the Resume Marker. Then the re-starting time in the Resume Marker is compared with the MoDuration of the first MO_INFO in the MoInfoTbl in the PRGnnn.PGI in the related Program directory (PRGnnn).
3. If the re-starting time value is larger than the MoDuration value, then the value of the MoDuration is subtracted from the re-starting time value. Then the residual of re-starting time is compared with the MoDuration value of the next MO_INFO.
4. The same operation is repeated until the residual of re-starting time becomes smaller than the MoDuration value of the next MO_INFO.
5. Divide the residual of re-starting time by TstInterval in the Media Object Information (MOI), and get the Time Search Entry number “n” as the quotient of the division.
6. Get MODU Number (ModuNumber) of the Time Search Entry “n”, which is specified by the above quotient, in the Time Search Entry Information (TSE_INFO1) in the current MOI file.
7. Get the MODU Offset (ModuOffset) value of the Time Search Entry “n”.

ã 2001 SD Association

CONFIDENTIAL

Figure Annex B-17 : Fast Forward Playback

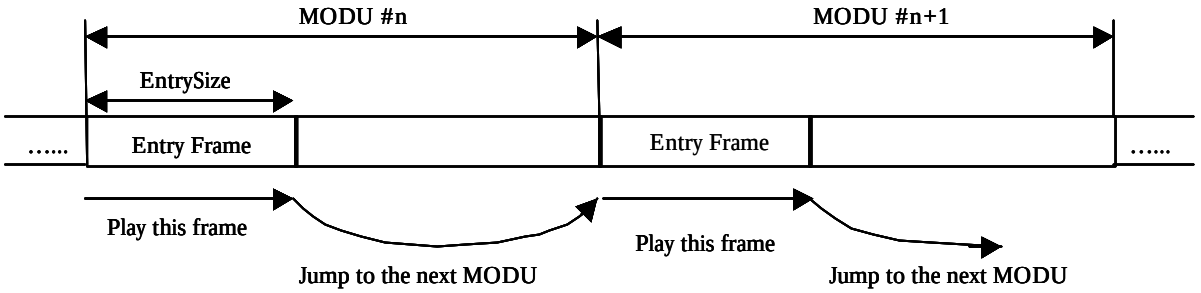


8. Read out the Entry Frame data in the current Media Object Data File (MOVnnn.MOD), which starting address is indicated by the ModuOffset value and which Frame length is indicated by the Total number of Packets used by Entry Frame (EntrySize) of the MODU Information (MODU_INFO) in the MODU Information Table (ModuiTbl) indicated by the MODU Number.
9. Feed the Entry Frame data to MPEG decoder, then add MODU Size (ModuSize) to the current ModuOffset value.
10. Then, repeat above reading out process for the next Entry Frame data until the FF button is released.

ã 2001 SD Association

CONFIDENTIAL

Figure Annex B-18 : Decode every Entry Frame



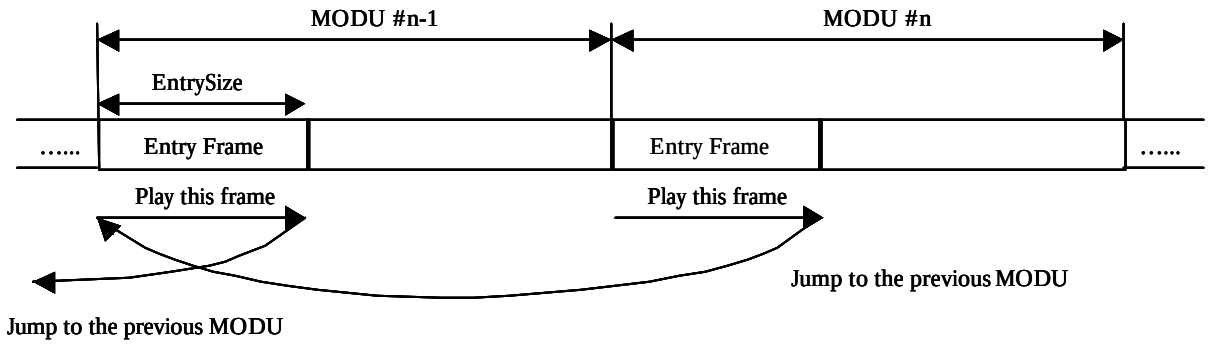
11. If necessary, the next MOI files or the next Program Information File are used in the same way to get the Entry Frames.

Annex B.3.4. Fast Reverse Playback

When a user presses FR (Fast Reverse Playback) button during Playlist Playback process or Randon Access Playback process, the following process starts.

1. The process to obtain the appropriate MODU address from the Resume Marker value is as same as in the Fast Forward Playback process.
2. Read out the Entry Frame data in the current Media Object Data File (MOVnnn.MOD), which re-starting address is indicated by the ModuOffset value and which Frame length is indicated by the Total number of packets used by Entry Frame (EntrySize) of the MODU Information (MODU_INFO) in the MODU Information Table (ModuiTbl) indicated by the (ModuNumber).
3. Feed the Entry Frame data to MPEG decoder, then subtract MODU Size (ModuSize) from the current ModuOffset value and move to the previous MODU specified by the subtracted ModuOffset value.
4. Then, repeat above reading out process for the previous Entry Frame data until the FR button is released.

Figure Annex B- 19 : Decode every Entry Frame backward



5. If necessary, the MOI files or the Program Information File, which number is smaller than the currently played-back one, is used in the same way to get the Entry Frames.

In case the Type of Time Search Table (TstType) of the current MOI indicates variable/fixed frame data size audio, appropriate packets are read out interspersingly and fed to MPEG decoder.

ã 2001 SD Association

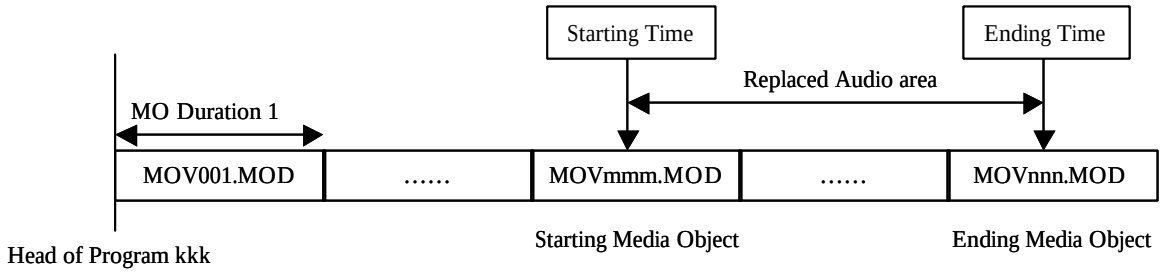
CONFIDENTIAL

Annex B.4. Audio-Replacement Procedure

In case of Audio-Replacement, a user specifies the Starting Time and Ending Time of the Audio-Replacement in a specified Program (PRGkkk).

The starting Media Object Data File (MOVmmm.MOD) and the ending Media Object Data File (MOVnnn.MOD) are determined by comparing the Starting Time and Ending Time with the (MoDuration) of Media Object Information (MO_INFO) in Program Information File (PRGkkk.PGI) in the specified Program directory (PRGkkk), adding them from the beginning.

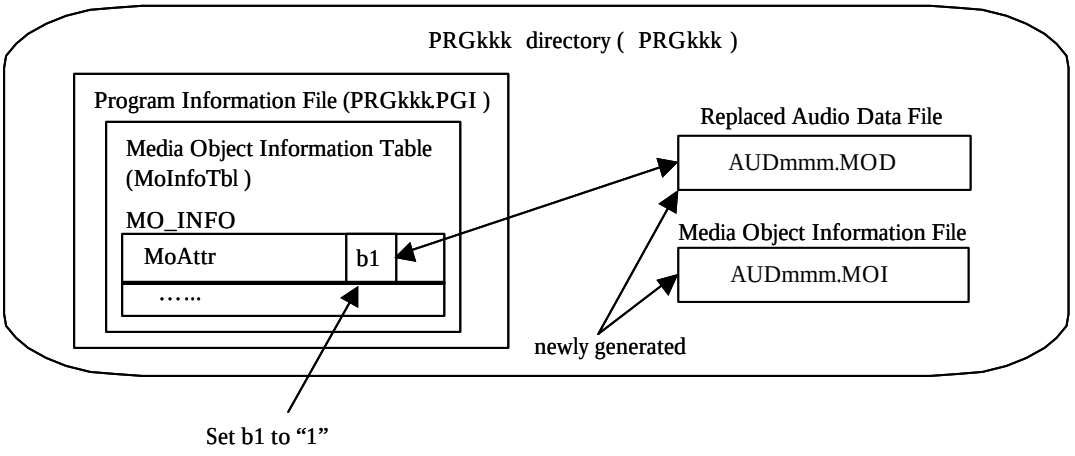
Figure Annex B-20 : Find stating Media Object and ending Media Object.



Then Playback process starts as described in the Playlist Playback.

1. First, set “1” to the Aud bit (b1) of (MoAttr) field in the curreent Media Object Information (MO_INFO) in the MoInfoTbl in the Program Information File in the specified Program directory (PRGkkk).
2. Create Replaced Audio Data File (AUDmmm.MOD) and the corresponding Media Object Information File (AUDmmm.MOI) in the PRGkkk directory.

Figure Annex B-21 : Set b1 of MoAttr in MO_INFO and create AUDmmm.MOD and AUDmmm.MOI.

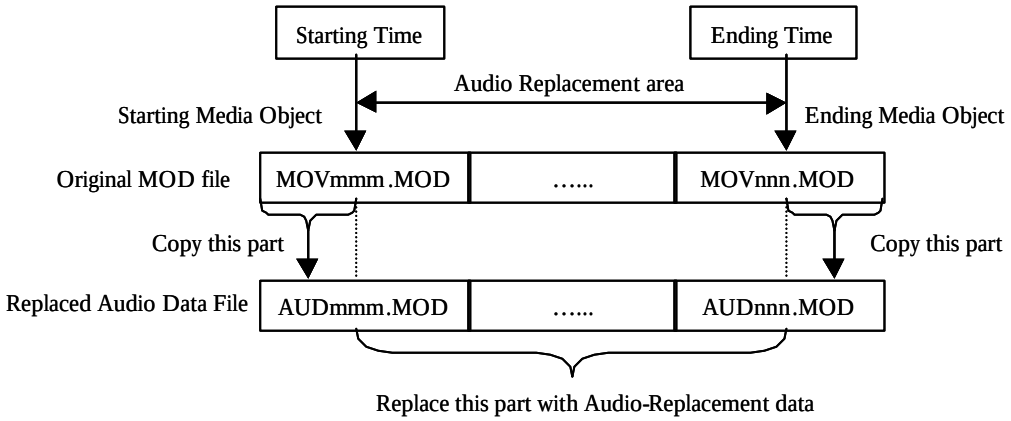


ã 2001 SD Association

CONFIDENTIAL

3. Record the played-back audio data in it until the current playback time becomes equal to the Starting Time of Audio-Replacement.
4. Then, switch the audio Encoder input to the Audio-Replacement data from the played-back data and continue both recording and playback until the current playback time becomes equal to the Ending Time of Audio-Replacement.
5. Then, switch the audio Encoder input to the played-back data from the Audio-Replacement data and continue both recording until the current MOVmmm.MOD is entirely played back. Then update the contents of AUDmmm.MOI appropriately.
6. If necessary, the Audio-Replacement process is continued to the following Media Object Data File (MOVmmm+1.MOD, …), until the current playback time becomes equal to the Ending Time of Audio-Replacement.
7. Then stop Audio-Replacement.

Figure Annex B-22 : Record Audio-Replacement data in AUDmmm.MOD up to AUDnnn.MOD.



SD Memory Card Specifications

Part 9
PIM SPECIFICATIONS

Version 1.0

CONFIDENTIAL

Conditions for publication

Publisher and Copyright Holder
SD Card Association

Confidentiality
This document shall be treated as confidential under the Non Disclosure Agreement which has been signed by the obtainer. Reproduction in whole or in part is prohibited without prior written permission of SD Card Association.

Exemption
None will be liable for any damages from use of this document.

Copyright Notice

vCard, and vCalendar are trademarks of Apple Computer, Inc., AT&T Corp., International Business Machines Corp., and Siemens.

©Copyright 2001 by SD Card Association All rights reserved

INDEX

Part 9 PIM SPECIFICATIONS

1. General 1
1.1 Scope 1
1.2 Normative References 2
2. Technical Elements 4
2.1 Definitions 4
2.2 Symbols 5
2.2.1 Relational operators 5
2.2.2 Arithmetic operators 5
2.3 Notations 5
2.3.1 Numerical notation 5
2.3.2 Bit fields 5
2.4 Terminology 6
2.4.1 SD-PIM Data 6
2.4.2 Used 6
2.4.3 Unused 6
2.4.4 Reader/Writer 6
2.5 Abbreviations 7
3. Overview 8
3.1 Content Model 8
3.2 Directory and File Structure 9
3.2.1 Design Concept of PIM Data Manager 13
3.2.2 Structure of PIM Data Manager (PIMMG) 15
3.3 Editing Procedure (Informative) 17
3.3.1 Initializing Procedure 17
3.3.2 Addition 17
3.3.3 Deletion 19
3.3.4 Order modification 20
4. PIM Data Manager Structure 21
4.1 PIM Data Manager (PIMMG) 21
4.2 PIM Data Manager Information (PIMMGI) 22
4.3 PIM Data Information (PIMDI) 24
4.4 Updating Procedure of a “current” PIM data (informative) 29
4.4.1 Creation 29
4.4.2 Update 30
5. PIM Data File Format 32
5.1 Common Structure of PIM Data File 32
5.2 Address Book Data File Format 33
5.3 Schedule/To Do List 38
5.4 Message 40
5.5 Notes 43
5.6 Bookmarks 44
Annex A. Relationship between PIMDI/PIMMG and PIM Data files 45
Annex B. Character Set Code 46
Annex C. Language Tag 47

©Copyright 2001 by SD Card Association All rights reserved

List of Figures

◊ Figure 1-1 Structure of SD-PIM Specifications 1
◊ Figure 3-1 The hierarchical structure 8
◊ Figure 3-2 Directory and File Configuration 9
◊ Figure 3-3 PIM Data Manager (PIMMG) 10
◊ Figure 3-4 Abstract structure of PIM Data Manager 13
◊ Figure 3-5 PIM Data Manager 15
◊ Figure 3-6 Adding a file 18
◊ Figure 3-7 Deleting a file 19
◊ Figure 4-1 PIM Data Manager (PIMMG) 21
◊ Figure 4-2 Creating a “current” PIM data file 29
◊ Figure 4-3 Updating a “current” PIM data file 31

©Copyright 2001 by SD Card Association All rights reserved

List of Tables

◊ Table 3-1 PIM Data Types 8
◊ Table 3-2 Specifications of naming convention of Address Book Data File 10
◊ Table 3-3 Specifications of naming convention of Schedule/To Do List Data File..... 11
◊ Table 3-4 Specifications of naming convention of Message Data File..... 11
◊ Table 3-5 Specifications of naming convention of Note Data File 12
◊ Table 3-6 Specifications of naming convention of Bookmark Data File..... 12
◊ Table 4-1 PIMMGI..... 22
◊ Table 4-2 PIMDI 24

©Copyright 2001 by SD Card Association All rights reserved

This page is intentionally left blank

©Copyright 2001 by SD Card Association All rights reserved

1. General

1.1 Scope

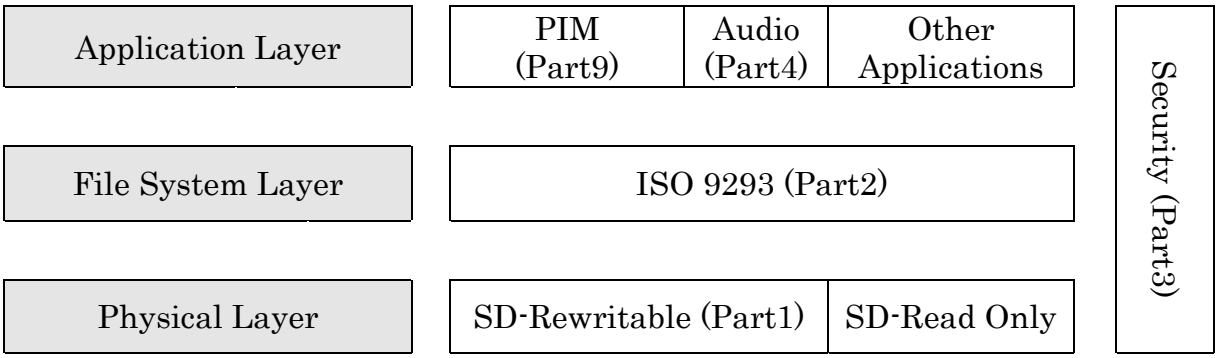
This part specifies the application format of the SD-PIM files that are recorded in the SD Memory Card. The SD-PIM files include 5 types of personal information : Address Book, Schedules/To Do List, Messages, Notes, and Bookmarks that will be exchanged with another device or saved for backup and archives.

The following features are not supported in this version of the SD-PIM specifications. These features will be supported in later version of the SD-PIM specifications if necessary.

1) Content protection
2) Synchronization

The card of which the data is defined in this specification shall comply with **"Part1. PHYSICAL SPECIFICATIONS", "Part2. FILE SYSTEM SPECIFICATIONS",** and **"Part3. SECURITY SPECIFICATIONS"** of SD memory card specifications.

◊ **Figure 1-1 Structure of SD-PIM Specifications**



©Copyright 2001 by SD Card Association All rights reserved.

1.2 Normative References

1) ISO 8601
   Representations of dates and times, 1988-06-15. :
   http://www.iso.ch/markete/8601.pdf
2) ISO 8601 Draft Revision
   Representations of dates and times, draft revision, 1998.
3) ISO 8859-1: 1987
   Information processing - 8-bit single-byte coded graphic character sets -
   Part 1:Latin alphabet No.1
4) ISO/IEC 10646-1:1993
   Information technology - Universal Multiple-Octet Coded Character Set (UCS) -
   Part 1: Architecture and Basic Multilingual Plane
5) ISO/IEC 10646-1:1993/Amd 2:1996
   UCS Transformation Format 8 (UTF-8)
6) JIS X 0201-1997
   7-bit and 8-bit coded character sets for information interchange (ISO/IEC 646)
7) JIS X 0208:1997
   7-bit and 8-bit coded 2 byte character sets for information interchange
8) vCard
   "vCard - The Electronic Business Card Exchange Format - Version 2.1
   The Internet Mail Consortium (IMC), September 18, 1996"
   http://www.imc.org/pdi/vcard-21.doc
9) Pronunciation Attribute Extension for vCard
   The Internet Mail Consortium (IMC), 1998-04-21
   http://www.imc.org/pdi/vcard-pronunciation.html
10) vCalendar
   "vCalendar - The Electronic Calendaring and Scheduling Format - Version 1.0
   The Internet Mail Consortium (IMC), September 18, 1996"
   http://www.imc.org/pdi/vcal-10.doc
11) vMessage
   Message Object format defined by Infrared Data Association
   “Specifications for Ir Mobile Communications (IrMC) Version 1.1, March 1, 1999”
12) vNote
   Note Object format defined by Infrared Data Association
   “Specifications for Ir Mobile Communications (IrMC) Version 1.1, March 1, 1999”
13) vBookmark
   Bookmark Object format defined by Infrared Data Association
   “Specifications for Ir Mobile Communications (IrMC) ERRATA 2000 07 18, July 28th 2000”

©Copyright 2001 by SD Card Association All rights reserved.

14) IANA
Internet Assigned Numbers Authority
http://www.iana.org
15) Shift-JIS
IANA MIB enum: 17
16) RFC2426
vCard MIME Directory Profile, September 1998.
17) SD-Audio
SD Memory Card Specifications Part 4 Audio specifications, version 1.0, February 2000
18) SD-Picture
SD Memory Card Specifications Part 5 Picture specifications, version 1.0, September 2000
19) SD-Video
SD Memory Card Specifications Part 8 Video specifications, version 1.0, January 2001

©Copyright 2001 by SD Card Association All rights reserved.

2. Technical Elements

2.1 Definitions

For the purpose of this specification, the following definitions apply:

RBP	:	Relative Byte Position within a descriptor, starting with zero.
Reserved	:	Reserved field. The term "reserved" indicates that the value may be used in the future for SD memory card specifications. Unless otherwise specified within this specification, all reserved bits shall be set to zero.

©Copyright 2001 by SD Card Association All rights reserved.

2.2 Symbols

2.2.1 Relational operators

== or =	Equality

2.2.2 Arithmetic operators

+	Addition
-	Subtraction
*	Multiplication
/	Division

2.3 Notations

2.3.1 Numerical notation

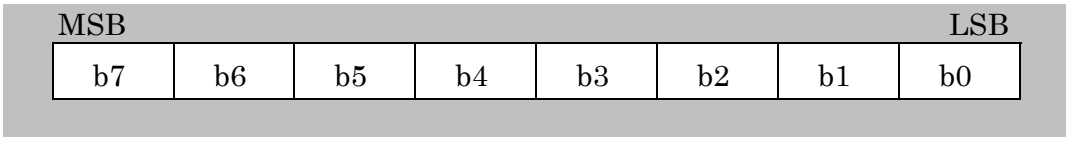
Numbers in decimal notation are represented by decimal digits, namely 0 to 9. Digit(s) represents any digit from digit 0 to 9.

Numbers in hexadecimal notation are represented as a sequence of one or more hexadecimal digits namely 0 to 9 and A to F, suffixed by "h" or prefixed by “0x”.

Multiple-byte numerical values in a description field MUST be recorded in the **BIG ENDIAN** representation. *e.g.*
the 16-bit hexadecimal number '1234h' shall be recorded as '12h', '34h'.

2.3.2 Bit fields

Certain fields or parts of fields are interpreted as an array of bits.
This array of bits shall be referred to as a bit field. Bit positions within an ***n*** bit field are numbered such that the least significant bit is numbered 0 and the most significant bit is numbered ***n–1***.



©Copyright 2001 by SD Card Association All rights reserved.

2.4 Terminology

2.4.1 SD-PIM Data

Text data based on vCard/vCalendar/vMessage/vNote/vBookmark format stored in an SD memory card.

2.4.2 Used

A data structure is “used” when it represents certain data. Overwriting this data structure is not permitted.

2.4.3 Unused

A data structure is “unused” when it does not represent any data. Overwriting this data structure is permitted.

2.4.4 Reader/Writer

Readers are devices which reproduce SD-PIM data on SD memory cards.
Writers are devices which store and delete SD-PIM data on SD memory cards.

©Copyright 2001 by SD Card Association All rights reserved.

2.5 Abbreviations

PIM	Personal Information Manager
PIMMG	PIM Data Manager
PIMMGI	PIM Data Manager Information
PIMDI	PIM Data Infomation
PIMMG_PIMDI_N	Number of existing PIM Data Information
PIMMG_PIMDI_NUSED	Number of used PIM Data Information
PIMMG_PIMDI_HEAD	The Head of used PIM Data Information List
PIMMG_PIMDI_TAIL	The Tail of used PIM Data Information List
P_PIMDI_SRN	Search Number of Previous PIM Data Information
N_PIMDI_SRN	Search Number of Next PIM Data Information
CHAR_CODE	Character Set Code
FL_SRN	File Search Number
UI	Used Information
CST	Status for Current Data
IANA	Internet Assigned Numbers Authority

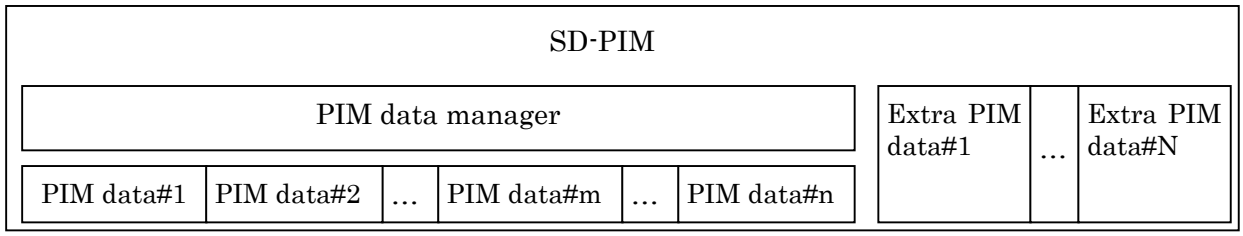
©Copyright 2001 by SD Card Association All rights reserved.

3. Overview

3.1 Content Model

The content which complies with SD-PIM specifications is accessed via a content model which consists of PIM data manager and PIM data as shown in [Figure 3-1]:

◊ **Figure 3-1 The hierarchical structure**



- **PIM data manager**

PIM data manager contains attributes of each PIM data files. PIM data manager should be unique in an SD memory card.

- **PIM data**

PIM data consists of 5 types of information as shown in [Table 3-1].

- **Extra PIM data**

Extra PIM data contains SD-PIM data. Writers may place PIM data files, when such PIM files agrees with PIM data file format described in chapter 5. The name of extra PIM data files MUST not start with "PIM". Writers must not create subdirectories. Readers are not required to read and/or to interpret extra PIM data files.

◊ **Table 3-1 PIM Data Types**

Data Type	File Name	Data Format
Address Book	PIMxxxxx.VCF	Based on vCard 2.1 Object format
Schedule/To Do List	PIMxxxxx.VCS	Based on vCalendar 1.0 Object format
Messages	PIMxxxxx.VMG	Based on vMessage 1.1 Object format
Notes	PIMxxxxx.VNT	Based on vNote 1.1 Object format
Bookmarks	PIMxxxxx.VBM	Based on vBookmark 1.0 Object format

$00001 \leq xxxxx \leq 65535$*

*In FAT File System, maximum total number of logical files is limited by total number of clusters (ex. On a FAT12 64MB SD memory card, total number of clusters is 4053.)

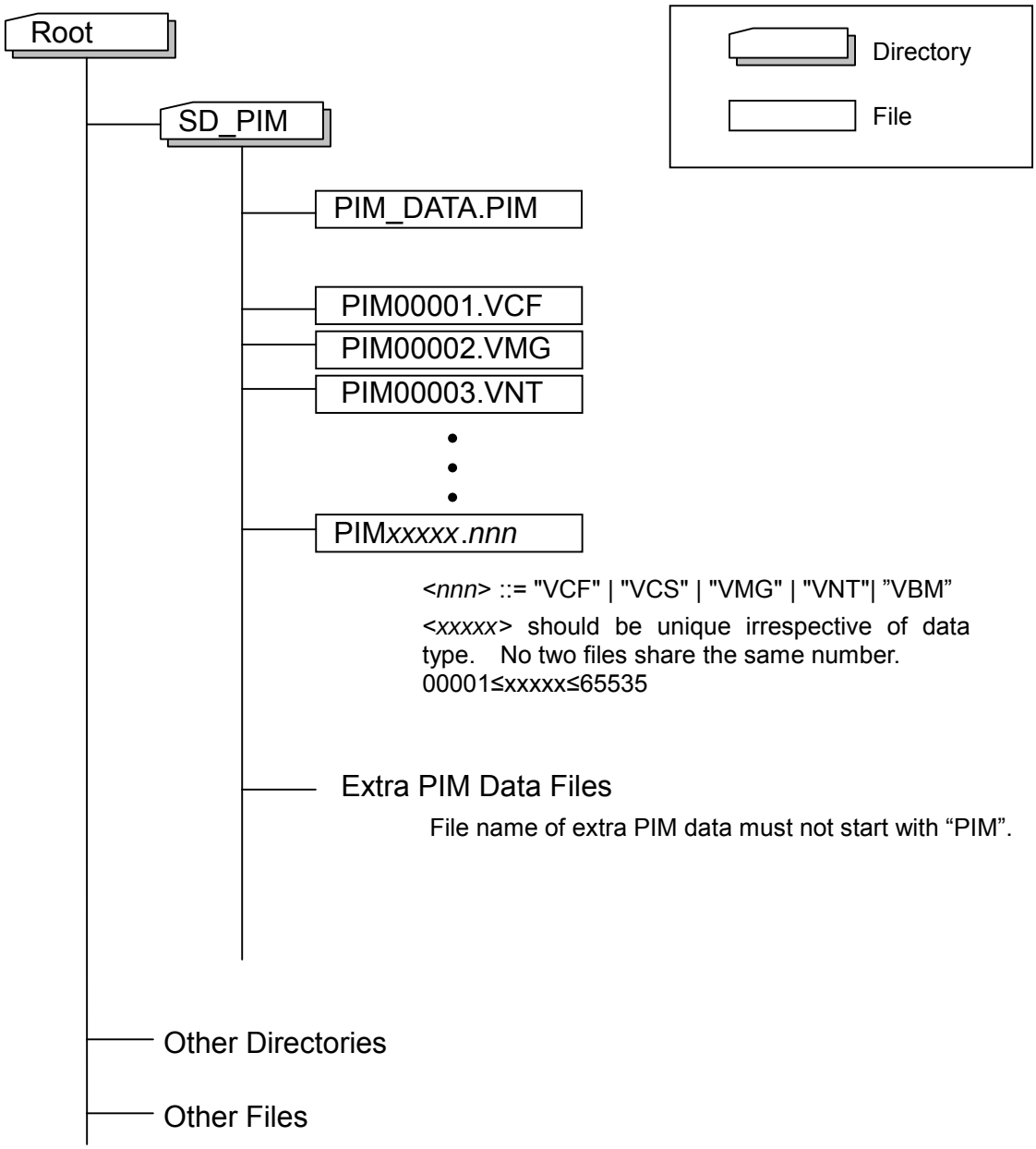
Note) "xxxxx" should be unique irrespective of Data Type. No two PIM data files shall share the same number.

©Copyright 2001 by SD Card Association All rights reserved.

3.2 Directory and File Structure

[Figure 3-2] illustrates the directory and files where the data which comply with this specifications are recorded. All files are placed under SD_PIM directory. It is not allowed to make subdirectories.

◊ **Figure 3-2 Directory and File Configuration**



©Copyright 2001 by SD Card Association All rights reserved.

- **SD_PIM (Directory)**
  All files that are based on SD-PIM specifications are placed under this directory.

- **PIM_DATA.PIM (File)**
  The management information for all of the PIM data files is described under the PIM data manager as shown if [Figure 3-3].

◊ **Figure 3-3 PIM Data Manager (PIMMG)**

PIMMG

PIM Data Manager
PIM Data Manager Information
PIM Data Information #1
PIM Data Information #2
. .
PIM Data Information #n

(n ≤ 65,535)

- **PIMxxxxx.VCF (File)**

Address Book Data File
The file name shall have 8 characters as base name and 4 characters as extension (including period).
The first three characters shall be "PIM". The next five characters shall be a decimal number between "00001" and "65535". The number "00000" shall not be used. This number shall agree with file search number in manager file (see section 4.3). The extension of the file name shall be ".VCF".
The specifications of the naming convention of Address Book Data File are shown in [Table 3-2]

◊ **Table 3-2 Specifications of naming convention of Address Book Data File**

	Fixed string part			Decimal number part (FL_SRN)					Alternative string part			
Specifications	P	I	M	0	0	0	0	1	.	V	C	F
				≀								
				6	5	5	3	5				
Example	P	I	M	0	1	2	3	4	.	V	C	F

©Copyright 2001 by SD Card Association All rights reserved.

- **PIMxxxxx.VCS (File)**

Schedule/To Do List Data File
The file name shall have 8 characters as base name and 4 characters as extension (including period).
The first three characters shall be "PIM". The next five characters shall be a decimal number between "00001" and "65535". The number "00000" shall not be used. This number shall agree with file search number in manager file (see section 4.3). The extension of the file name shall be ".VCS".
The specifications of the naming convention of Schedule/To Do List Data File are shown in [Table 3-3]

◊ **Table 3-3 Specifications of naming convention of Schedule/To Do List Data File**

	Fixed string part			Decimal number part (FL_SRN)					Alternative string part			
Specifications	P	I	M	0	0	0	0	1	.	V	C	S
				≀								
				6	5	5	3	5				
Example	P	I	M	0	1	2	3	4	.	V	C	S

- **PIMxxxxx.VMG (File)**

Message Data File
The file name shall have 8 characters as base name and 4 characters as extension (including period).
The first three characters shall be "PIM". The next five characters shall be a decimal number between "00001" and "65535". The number "00000" shall not be used. This number shall agree with file search number in manager file (see section 4.3). The extension of the file name shall be ".VMG".
The specifications of the naming convention of Message Data File are shown in [Table 3-4]

◊ **Table 3-4 Specifications of naming convention of Message Data File**

	Fixed string part			Decimal number part (FL_SRN)					Alternative string part			
Specifications	P	I	M	0	0	0	0	1	.	V	M	G
				≀								
				6	5	5	3	5				
Example	P	I	M	0	1	2	3	4	.	V	M	G

©Copyright 2001 by SD Card Association All rights reserved.

- **PIMxxxxx.VNT (File)**

Note Data File
The file name shall have 8 characters as base name and 4 characters as extension (including period).
The first three characters shall be "PIM". The next five characters shall be a decimal number between "00001" and "65535". The number "00000" shall not be used. This number shall agree with file search number in manager file (see section 4.3). The extension of the file name shall be ".VNT".
The specifications of the naming convention of Note Data File are shown in [Table 3-5]

◊ **Table 3-5 Specifications of naming convention of Note Data File**

	Fixed string part			Decimal number part (FL_SRN)					Alternative string part			
Specifications	P	I	M	0	0	0	0	1	.	V	N	T
				≀								
				6	5	5	3	5				
Example	P	I	M	0	1	2	3	4	.	V	N	T

- **PIMxxxxx.VBM (File)**

Bookmark Data File
The file name shall have 8 characters as base name and 4 characters as extension (including period).
The first three characters shall be "PIM". The next five characters shall be a decimal number between "00001" and "65535". The number "00000" shall not be used. This number shall agree with file search number in manager file (see section 4.3). The extension of the file name shall be ".VBM".
The specifications of the naming convention of Bookmark Data File are shown in [Table 3-6]

◊ **Table 3-6 Specifications of naming convention of Bookmark Data File**

	Fixed string part			Decimal number part (FL_SRN)					Alternative string part			
Specifications	P	I	M	0	0	0	0	1	.	V	B	M
				≀								
				6	5	5	3	5				
Example	P	I	M	0	1	2	3	4	.	V	B	M

- **Extra PIM data files**

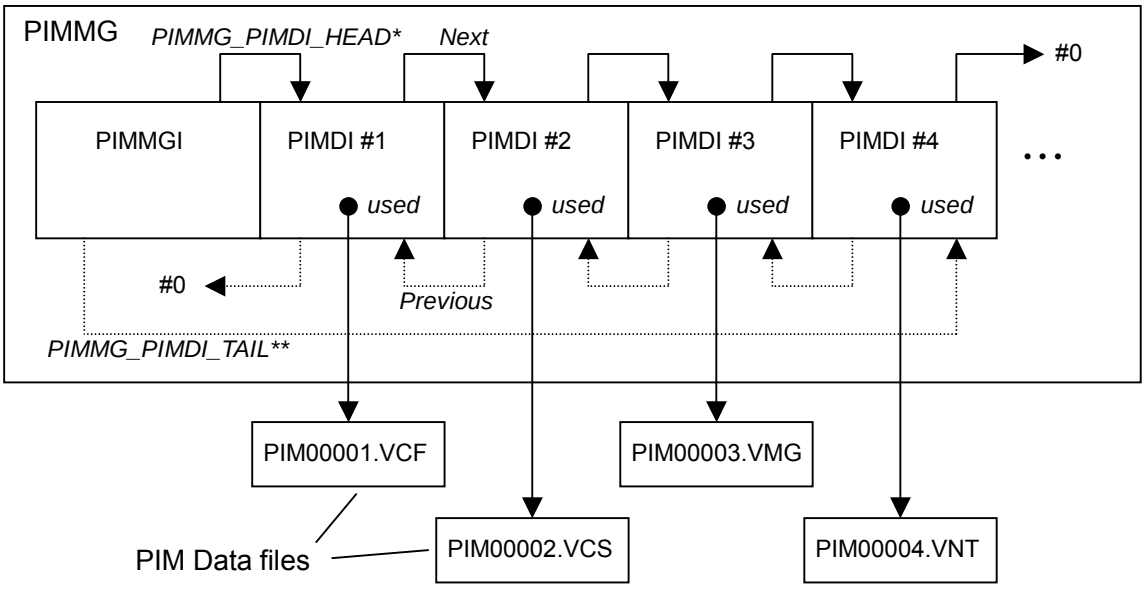
Any PIM data file that agrees with the PIM data file format described in chapter 5 may be placed under this directory. The file name must not start with "PIM". It is recommended to use ".VCF" for the extension of the address book data file name, ".VCS" for the extensions of the schedule/to do list data file name, ".VMG" for the extension of the message data file name, ".VNT" for the extension of the note data file name, and ".VBM" for the extension of the bookmarks data file name.

©Copyright 2001 by SD Card Association All rights reserved.

3.2.1 Design Concept of PIM Data Manager

[Figure 3-4] shows the abstract structure of PIM data manager:

◊ **Figure 3-4 Abstract structure of PIM Data Manager**



*PIMMG_PIMDI_HEAD : The Head of used PIMDI list (see section 4.2)
** PIMMG_PIMDI_TAIL : The Tail of used PIMDI list (see section 4.2)

PIM data manager(PIMMG) consists of one PIM data manager information(PIMMGI) followed by zero or more PIM data information(PIMDI). PIMMGI contains the information of the numbers of PIMDI's and the head and the tail of list structure of PIMDI's. A PIMDI has the metadata of PIM file along with the bidirectional list structure links.

The bidirectional list structure is prepared so that readers with small computing resource may list the stored PIM data to users in proper order.

Each PIMDI has Used Information(UI), a used/unused flip-flop, so that writers are not required to erase data from PIMMG when deleting PIM data files to avoid unnecessary complicated calculation.

Devices are expected to operate as follows:

When initializing, a writer shall create a PIMMG which consists of PIMMGI without PIMDI's.

When storing a PIM data file, a writer shall reuse an existing unused PIMDI area. When and only when no unused PIMDI exists, a writer shall create a new PIMDI.

When deleting a PIM data file, a writer shall set an unused flag in Used Information field of the corresponding PIMDI.

©Copyright 2001 by SD Card Association All rights reserved.

Deleting a PIMDI is not recommended. However, it is not prohibited for writers to delete PIMDI for a garbage collection purpose. In such cases, writers must maintain the structure by renaming PIM data files.

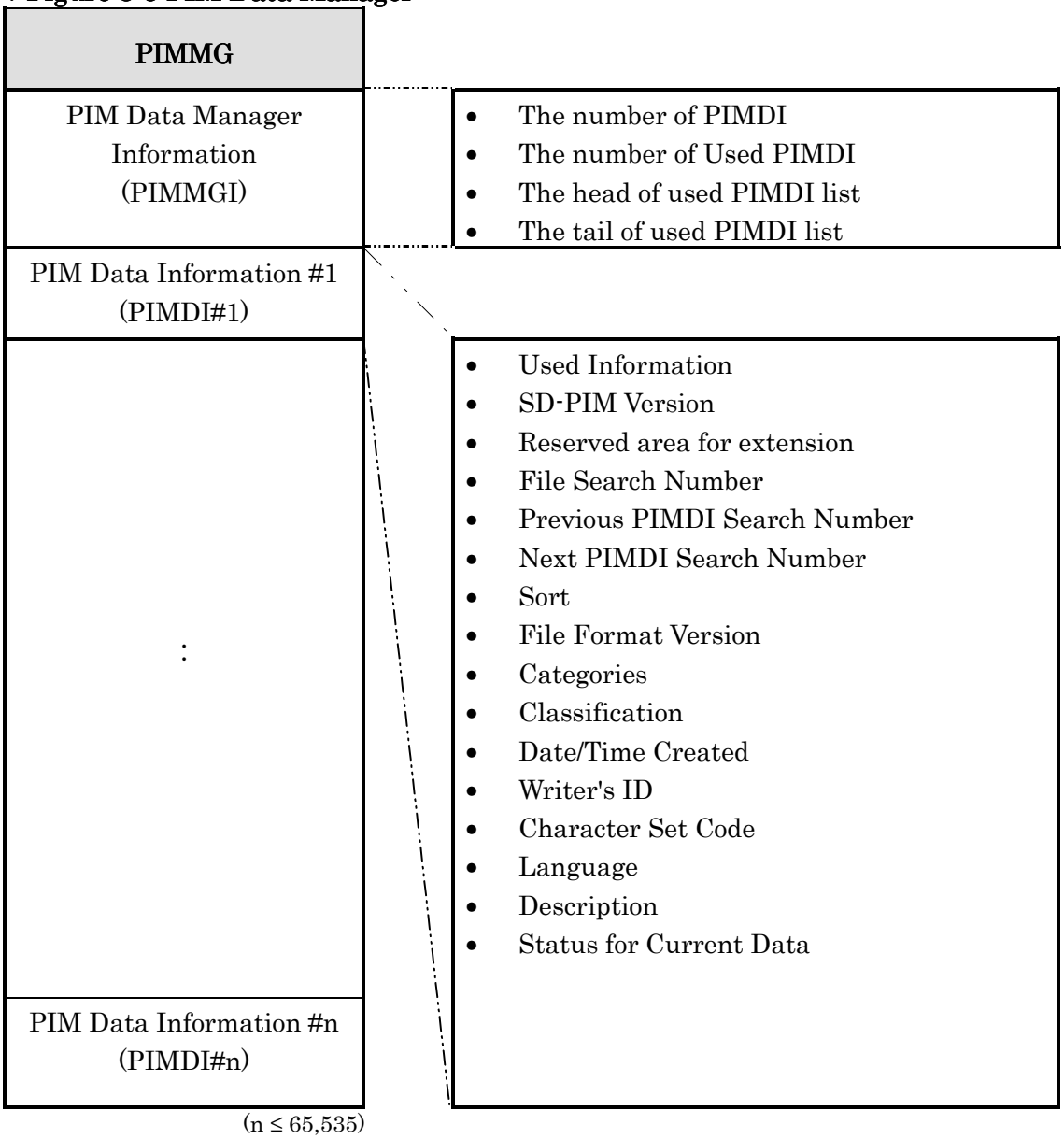
Writers may erase whole PIMMG when no used PIMDI exists.

©Copyright 2001 by SD Card Association All rights reserved.

3.2.2 Structure of PIM Data Manager (PIMMG)

PIM Data Manager (PIMMG) consists of 2 elements: the PIM Data Manager Information (PIMMGI) and the PIM Data Information (PIMDI).

◊ **Figure 3-5 PIM Data Manager**



- **PIM Data Manager Information (PIMMGI)**
  PIMMGI describes PIMMG information.
- **PIM Data Information (PIMDI)**
  PIMDI consists of:

• **Used Information**
  This information describes if PIMDI is used or unused.
  If a PIMDI points a certain PIM data file, the PIMDI is 'used'.
  If a PIMDI does not point a valid PIM data file, the PIMDI is 'unused'.

©Copyright 2001 by SD Card Association All rights reserved.

- **SD-PIM Version**
  the version of SD-PIM supported
- **Reserved area for Extension**
- **File Search Number**
  File number of the PIM Data file.
- **Previous PIMDI Search Number**
- **Next PIMDI Search Number**
- **Sort**
- **File Format Version**
- **Categories**
- **Classification**
- **Date/Time Created**
  Date and time at the PIM data file created.
- **Writer's ID**
- **Character Set Code**
  Character code information of the PIM data information.
- **Language**
  A language used in the PIM data information
- **Description**
- **Status for Current Data**

©Copyright 2001 by SD Card Association All rights reserved.

3.3 Editing Procedure (Informative)

3.3.1 Initializing Procedure

If SD_PIM directory is empty, initializing process to make the PIM data manager file is necessary.

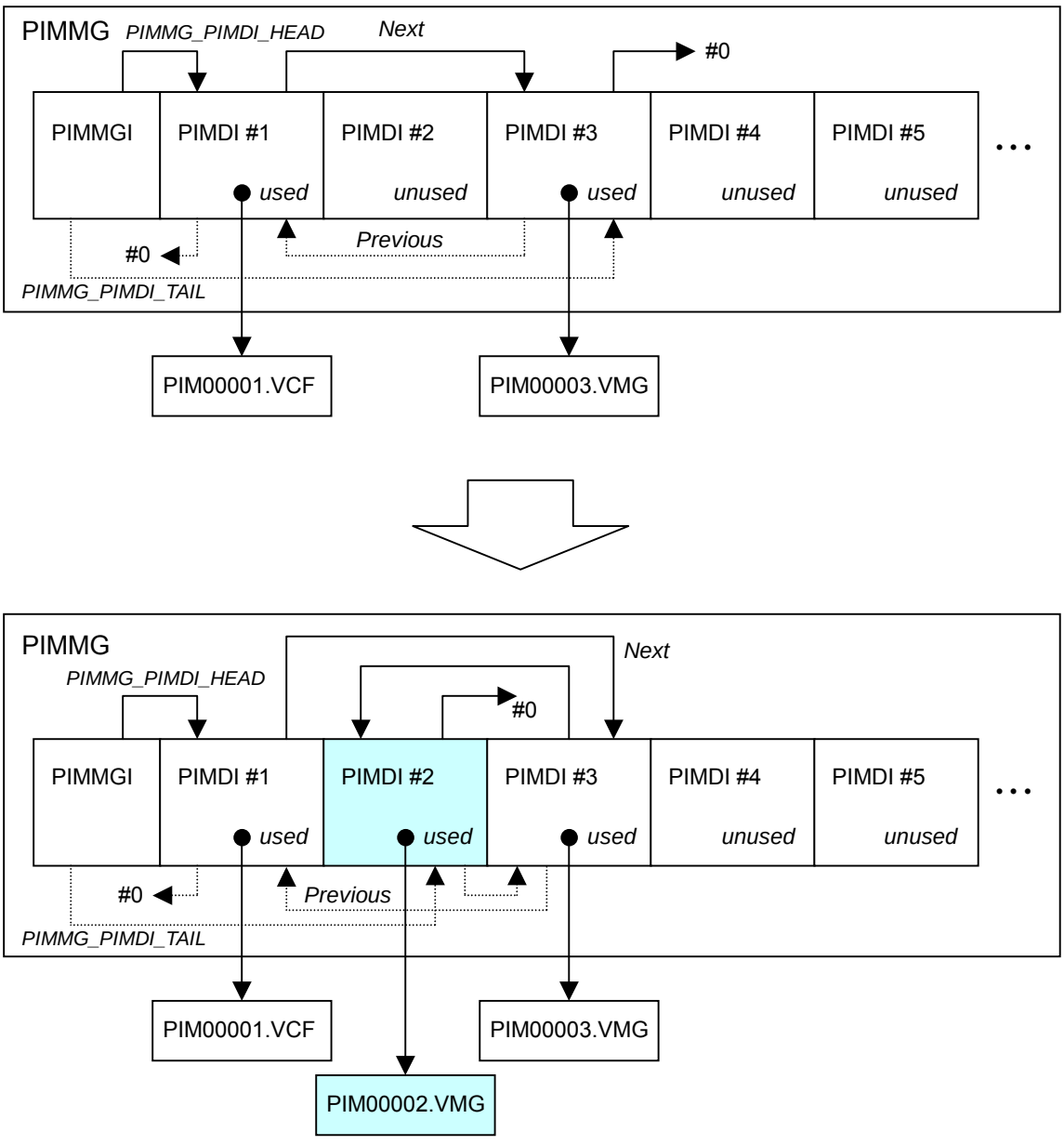
3.3.2 Addition

1. Search the first “unused” PIMDI (PIMDI#2).
2. Change PIMDI#2 to “used”.
3. Renew P_PIMDI_SRN and N_PIMDI_SRN of PIMDI#2.
4. Renew N_PIMDI_SRN of PIMDI#3.
5. Renew PIMMGI.
6. Store a PIM data file ("PIM00002.VMG").
7. Renew the other information fields of PIMDI#2.

Exclusive access control to PIMMGI is necessary from procedure 1 to 5.
Exclusive access control to PIMDI#2 is necessary in procedure 7.

©Copyright 2001 by SD Card Association All rights reserved.

◊ **Figure 3-6 Adding a file**



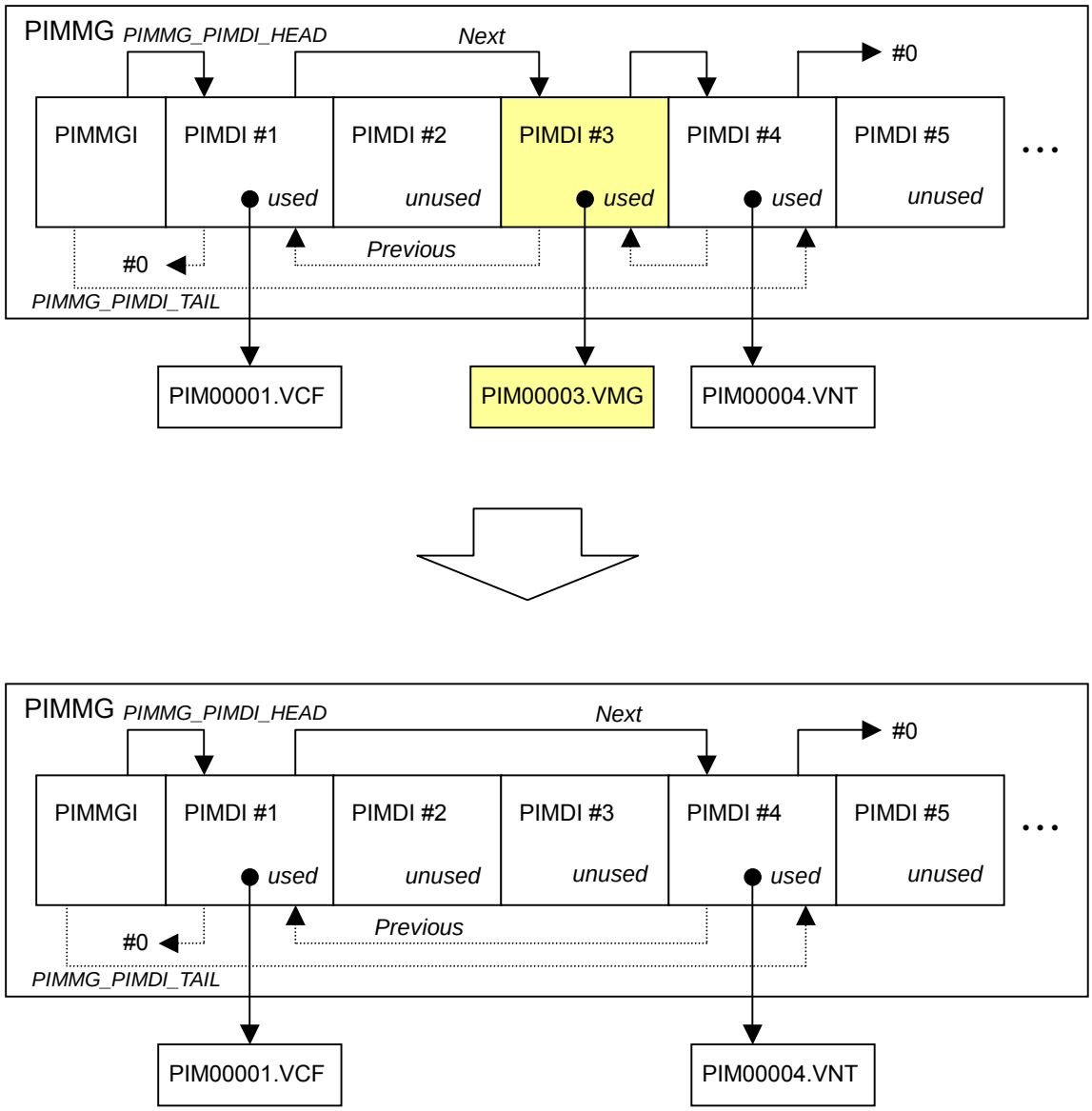
©Copyright 2001 by SD Card Association All rights reserved.

3.3.3 Deletion

To delete "PIM00003.VMG"
1. Change PIMDI chain list structure for deleting PIMDI#3 (modify PIMDI#1 and #4)
2. Renew PIMDI#3 “unused”
3. Renew PIMMGI

Exclusive access control to PIMMGI is necessary from procedure 1 to 3.

◊ **Figure 3-7 Deleting a file**



©Copyright 2001 by SD Card Association All rights reserved.

3.3.4 Order modification

Order modification is achieved by changing the list structure of PIMDIs. If order of the first PIMDI is modified, PIMMG_PIMDI_HEAD[see section 4.2] in PIMMGI must be renewed.

The order modification is not encouraged. This means that readers can assume the order of PIMDI's is the history of PIMDI additions.

©Copyright 2001 by SD Card Association All rights reserved.

4. PIM Data Manager Structure

4.1 PIM Data Manager (PIMMG)

PIMMG describes information about PIM data files.
PIMMG starts with PIM Data Manager Information (PIMMGI), followed by a set of PIM Data Information (PIMDI), as shown in [Figure 4-1].

◊ **Figure 4-1 PIM Data Manager (PIMMG)**

PIM Data Manager (PIMMG)

PIM Data Manager Information (PIMMGI) (Mandatory)	Fixed Size(16B)
PIM Data Information #1 (PIMDI #1) (Optional)	Fixed Size(128B)
PIM Data Information #2 (PIMDI #2) (Optional)	Fixed Size(128B)
:	:
PIM Data Information #n (PIMDI #n) (Optional)	Fixed Size(128B)

(n ≤ 65,535)

©Copyright 2001 by SD Card Association All rights reserved.

4.2 PIM Data Manager Information (PIMMGI)

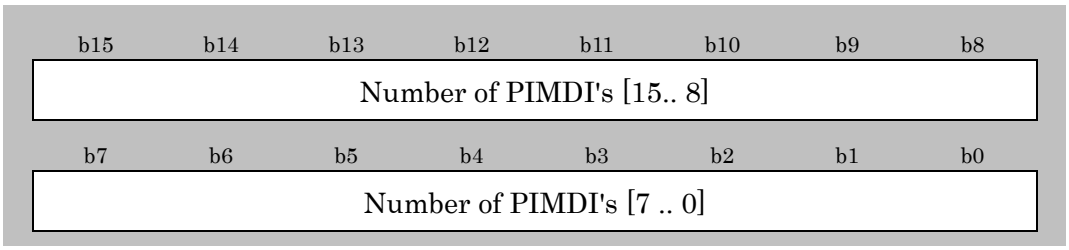
As shown in [Table 4-1], number of PIM data information, number of used PIM data information, and the head and the tail of "used" PIMDI list are described in PIM data manager information (PIMMGI).

◊ Table 4-1 PIMMGI (Description order)

RBP		Contents	Number of bytes
0 to 1	PIMMG_PIMDI_N	Number of existing PIMDI's	2 bytes
2 to 3	PIMMG_PIMDI_NUSED	Number of used PIMDI's	2 bytes
4 to 5	PIMMG_PIMDI_HEAD	The head of used PIMDI list	2 bytes
6 to 7	PIMMG_PIMDI_TAIL	The tail of used PIMDI list	2 bytes
8 to 15	Reserved	Enter 00h	8 bytes
Total			16 bytes

(RBP 0 to 1) PIMMG_PIMDI_N

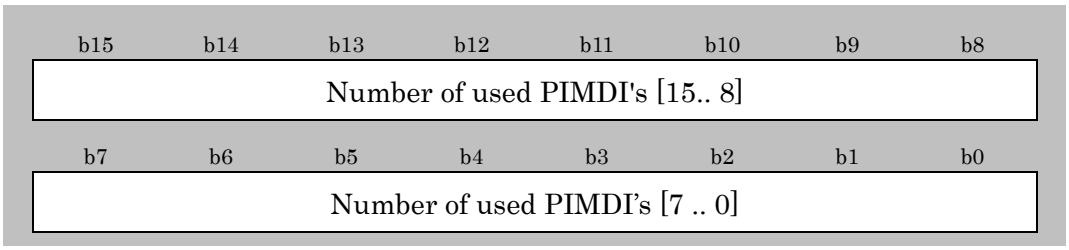
This field describes the number of existing PIMDI's in PIMMG.



Number of PIMDI's ... number between '0' and '65,535'.

(RBP 2 to 3) PIMMG_PIMDI_NUSED

This field describes the number of used PIMDI's in PIMMG.

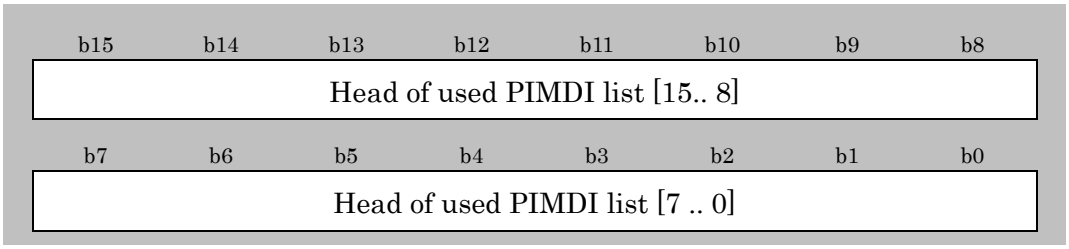


Number of used PIMDI's ... number between '0' and '65,535'.

©Copyright 2001 by SD Card Association All rights reserved.

(RBP 4 to 5) PIMMG_PIMDI_HEAD

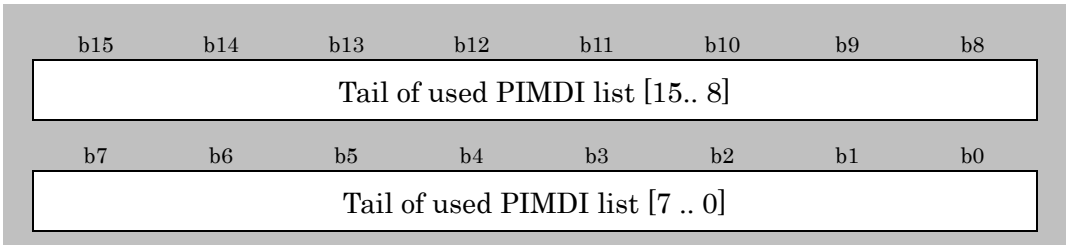
This field describes the search number of the head of used PIMDI list.
The value "0" means that there are NO “used” PIMDI's in PIMMG.



Head of used PIMDI list ... number between '0' and '65,535'.

(RBP 6 to 7) PIMMG_PIMDI_TAIL

This field describes the search number of the tail of used PIMDI list.
The value "0" means that there are NO “used” PIMDI's in PIMMG.



Tail of used PIMDI list ... number between '0' and '65,535'.

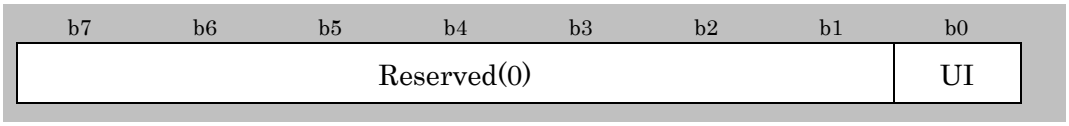
©Copyright 2001 by SD Card Association All rights reserved.

4.3 PIM Data Information (PIMDI)

◊ **Table 4-2 PIMDI** (Description order)

RBP		Contents	Number of bytes
0	UI	Used Information	1 byte
1	VERN	SD-PIM Version	1 byte
2 to 3	FL_SRN	File Search Number	2 bytes
4 to 5	P_PIMDI_SRN	Previous PIMDI Search Number	2 bytes
6 to 7	N_PIMDI_SRN	Next PIMDI Search Number	2 bytes
8	Reserved	Enter 00h	1 byte
9	SORT	Sort of PIM Data File	1 byte
10 to 17	FORMAT_VER	File Format Version	8 bytes
18	CATEGORIES	Categories	1 byte
19	CLASS	Classification	1 byte
20 to 51	DCREATED	Date/Time Created	32 bytes
52 to 67	WRITER_ID	Writer's ID	16 bytes
68 to 69	CHAR_CODE	Character Set Code	2 bytes
70 to 77	LANG	Language	8 bytes
78 to 109	DESCRIPTION	Description	32 bytes
110	CST	Status for Current Data	1 byte
111 to 127	Reserved	Enter 00h	17 bytes
Total			128 bytes

(RBP 0) UI
This field describes used information.

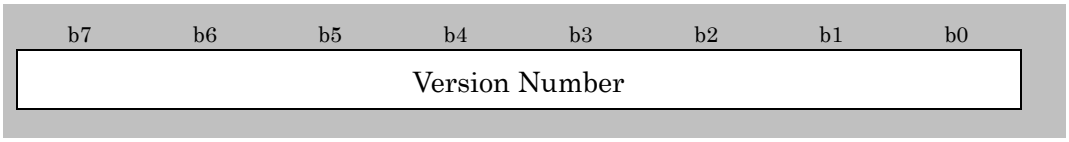


Used Information ... 0, when the PIMDI is not in use.
1, when the PIMDI is in use.

©Copyright 2001 by SD Card Association All rights reserved.

(RBP1) VERN

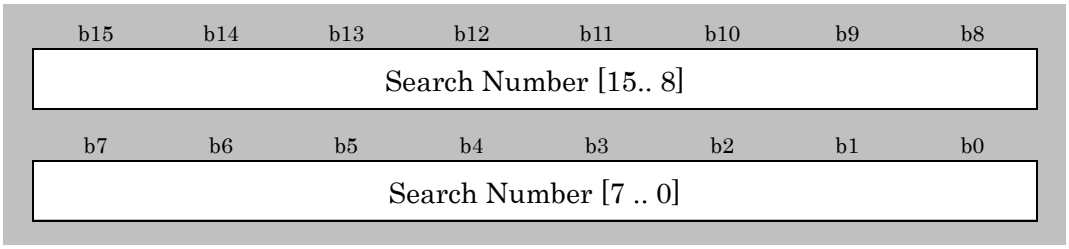
This field describes version number of SD-PIM specifications employed by the writer when writing.



Version number ... 10h : version 1.0
Others : reserved

(RBP 2 to 3) FL_SRN

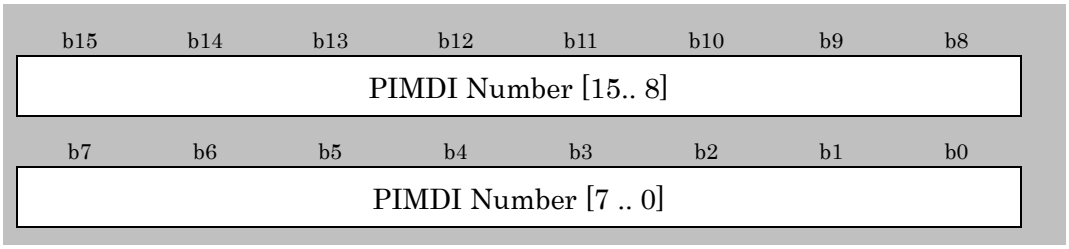
This field describes the search number of referred PIM data file.
FL_SRN must be equal to :
((absolute byte position of this PIMDI) – (byte size of PIMMGI)) / (byte size of PIMDI) + 1.



Search Number ... number between '1' and '65,535'.

(RBP 4 to 5) P_PIMDI_SRN

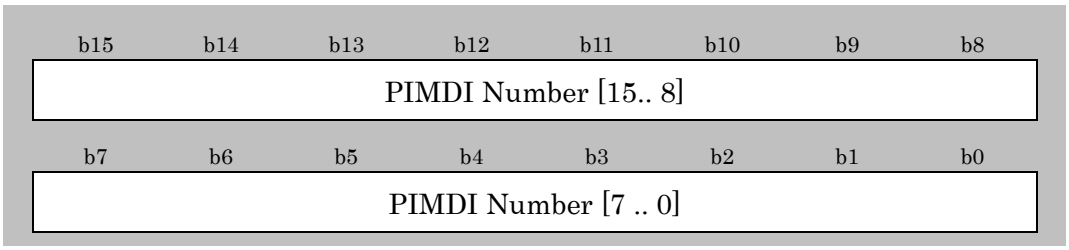
This field describes the search number of the previous PIMDI.



PIMDI Number ... number between '1' and '65,535': search number of previous PIMDI
0, when previous PIMDI is not present.

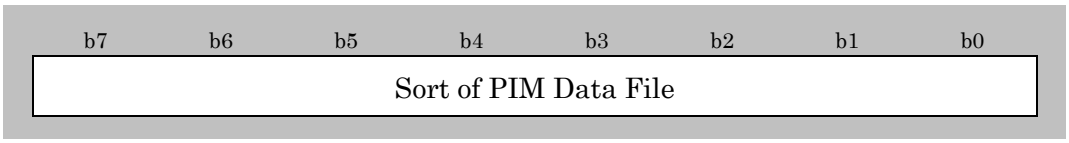
©Copyright 2001 by SD Card Association All rights reserved.

(RBP 6 to 7) N_PIMDI_SRN
This field describes the search number of the next PIMDI.



PIMDI Number ... number between '1' and '65,535': search number of next PIMDI
0, when next PIMDI is not present.

(RBP 9) SORT
This field describes the sort of PIM data file.



Sort of PIM Data File ... 01h : Address Book
02h : Schedules/To Do List
04h : Messages
08h : Notes
10h : Bookmarks

Others : reserved

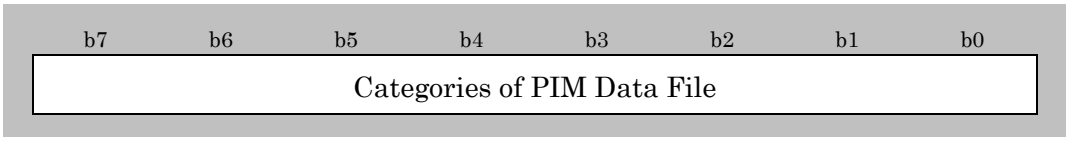
(RBP 10 to 17) FORMAT_VER
This field describes file format version format version of the PIM data represented in US-ASCII. This field must be terminated by NULL character.
FORMAT_VER … US-ASCII text of File Format Version (ex. “2.1”)

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

(RBP 18) CATEGORIES

This field describes the categories of the PIM data file. Multiple categories may be assigned to one PIM data file.

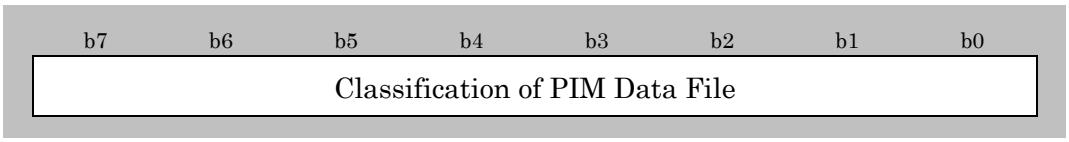


Categories of PIM Data File ... 00h : unspecified
01h : Backup
02h : Export
04h : Archive
08h : Temporary
10h : Current
80h : Miscellaneous

Others : reserved

(RBP 19) CLASS

This field describes the classification of the PIM data file.



Classification of PIM data file ... 00h : unspecified
01h : Public
02h : Private
04h : Confidential

Others : reserved

(RBP 20 to 51) DCREATED

This field describes the date and time of creation of the PIM data file.
Format of the date and time is based on ISO8601 and must be terminated by NULL characters.
yyyy-mm-ddThh:mm:ss+hh:mm (ex. “2000-09-15T21:15:00+09:00”)
When DCREATED is unknown, this field must be filled with NULL characters.

©Copyright 2001 by SD Card Association All rights reserved.

(RBP 52 to 67) WRITER_ID

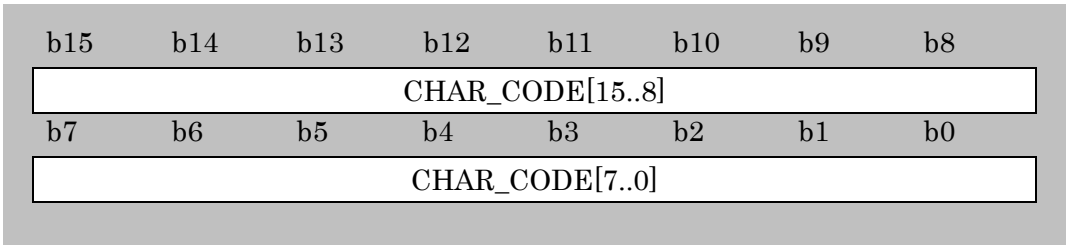
This field describes writer's ID represented in US-ASCII.
This field must be terminated by NULL characters.

WRITER_ID … US-ASCII text of writer's ID (ex. “Pocket PIM SX-2”)

When WRITER_ID is unknown, this field must be filled with NULL characters.

(RBP 68 to 69) CHAR_CODE

This field describes character set code of the character field of this PIMDI.



Character Set code … IANA Character Sets
See ANNEX B

(RBP 70 to 77) LANG

This field describes TAG of a language used in the PIM data information.
This field must be terminated by NULL characters.

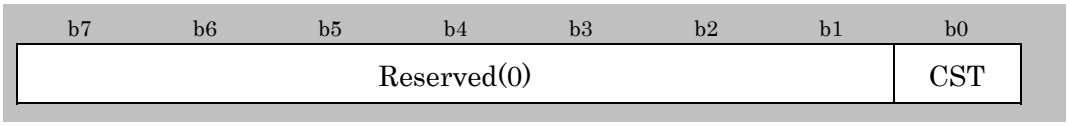
LANG … Language TAG (ex. “en-us”)
See ANNEX C

(RBP 78 to 109) DESCRIPTION

This field describes the description of the PIM data file
This field must be terminated by NULL characters. This field is represented by the character set specified in the CHAR_CODE field of the PIMDI.
When DESCRIPTION is unspecified, this field must be filled with NULL character.

(RBP 110) CST

This field describes the result of updating operation to PIM data.



Status for Current Data ... 0, when the PIM Data File has been updated.
1, when the PIM Data File has not been updated.

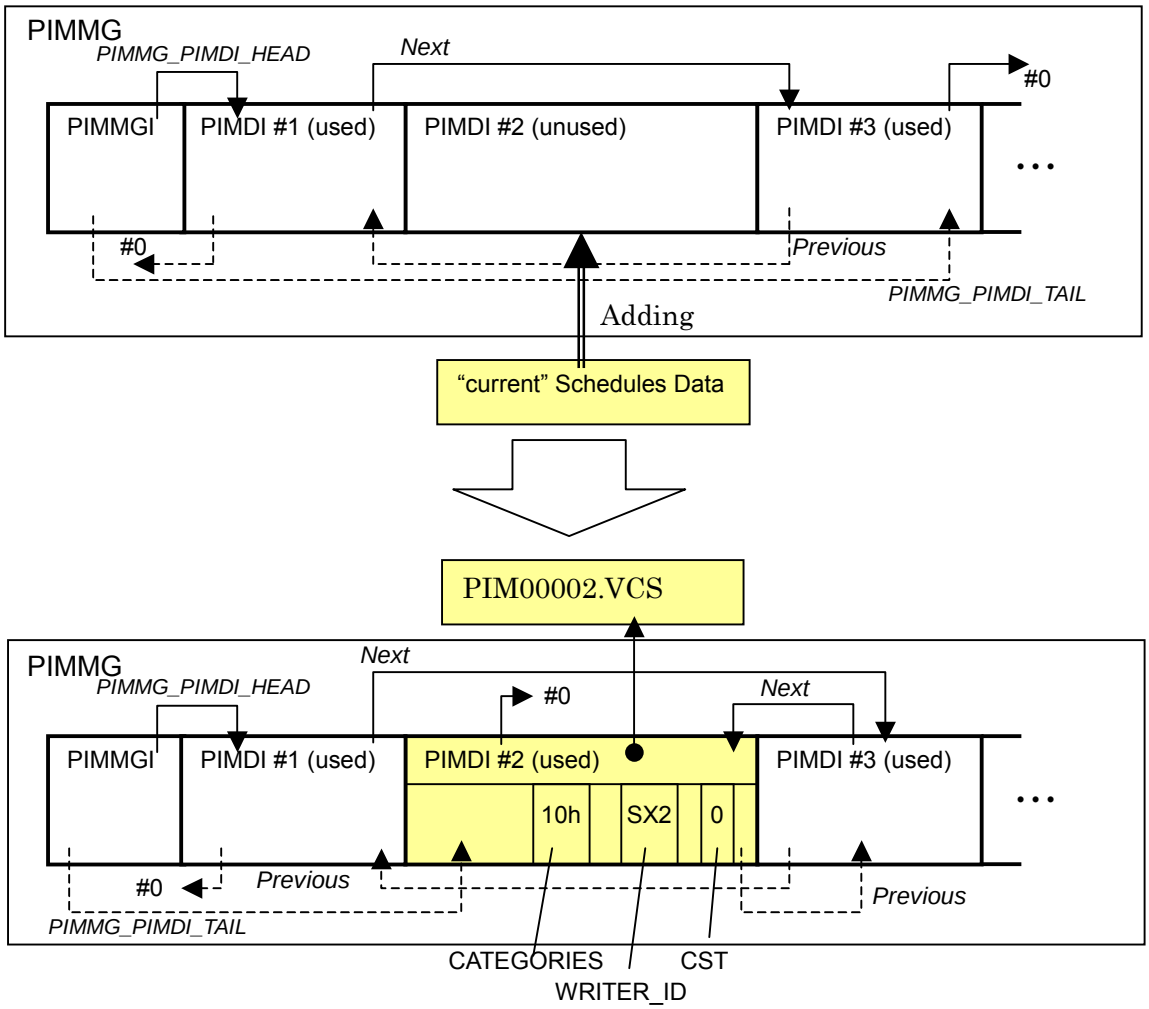
©Copyright 2001 by SD Card Association All rights reserved.

4.4 Updating Procedure of a "current" PIM data (informative)

4.4.1 Creation

1. Search the first "unused" PIMDI (PIMDI#2).
2. Change PIMDI#2 to "used"
3. Renew P_PIMDI_SRN and N_PIMDI_SRN of PIMDI#2.
4. Renew N_PIMDI_SRM of PIMDI#3.
5. Renew PIMMGI.
6. Store a PIM data file ("PIM00002.VCS").
7. Store PIMDI#2.
   - Store 10h in CATEGORIES of PIMDI#2. (10h means that this PIMDI is a "current" PIM data.)
   - Store WRITER_ID in PIMDI#2. ("SX2")
   - Set 0 in CST of PIMDI#2. (0 means that this PIMDI has been updated.)

◊ **Figure 4-2 Creating a "current" PIM data file**



©Copyright 2001 by SD Card Association All rights reserved.

4.4.2 Update

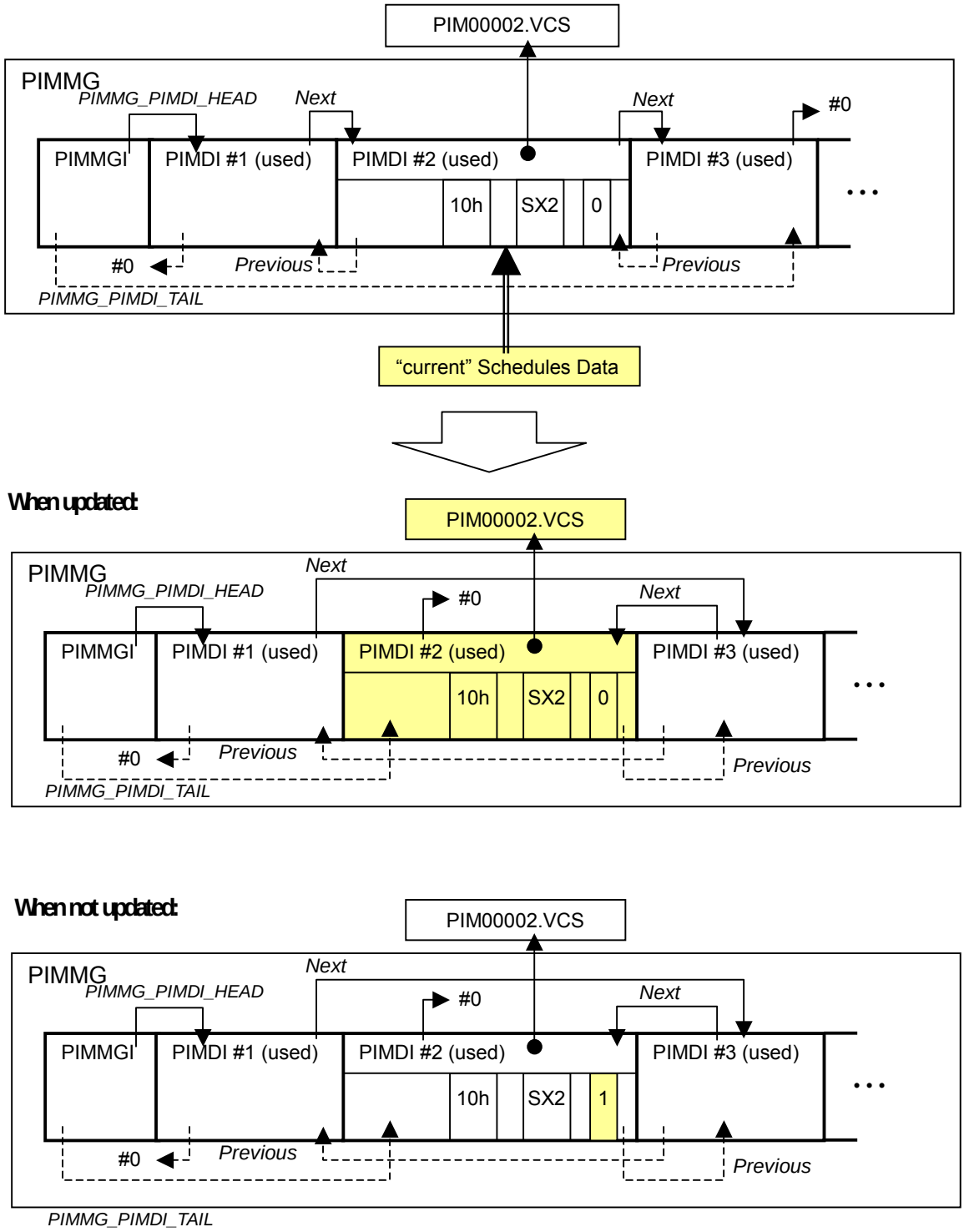
Updating PIM00002.VCS

0. On error: restore structure and goto step 15.
1. Set 0 to CST of PIMDI#2.
2. Check if updating is necessary.
3. If yes, continue. If no, goto step 12.
4. Renew P_PIMDI_SRN and N_PIMDI_SRN of PIMDI#2.
5. Renew N_PIMDI_SRN of PIMDI#3.
6. Renew PIMMGI.
7. Rename the present PIM data file by changing the extension to ".TMP"(PIM00002.TMP).
8. Store the new PIM data file (PIM00002.VCF).
9. Renew PIMDI#2.
10. Delete renamed the PIM data file(PIM00002.TMP).
11. goto step 16.
12. Renew P_PIMDI_SRN and N_PIMDI_SRN of PIMDI#2.
13. Renew N_PIMDI_SRN of PIMDI#3.
14. Renew PIMMGI.
15. Set 1 to CST of PIMDI#2. (1 means that this PIMDI has not been updated.)
16. Exit.

Note) It is not recommended to rewrite PIMDI which has the different value in WRITER_ID without user's explicit prior approval.

©Copyright 2001 by SD Card Association All rights reserved.

◊ **Figure 4-3 Updating a "current" PIM data file**



©Copyright 2001 by SD Card Association All rights reserved.

5. PIM Data File Format

This section describes PIM data file format and data fields required or recommended by this specification.

5.1 Common Structure of PIM Data File

PIM data file format is basically based on the structure of the vCard, vCalendar, vMessage, vNote, and vBookmark for the purpose of interoperability.
Following properties are extended.

Description	Property	Implementation Level	
SD-PIM Version	X-SD-VERN	Option	
File Format Version	X-SD-FORMAT_VER	Option	
Categories	X-SD-CATEGORIES	Option	
Classification	X-SD-CLASS	Option	
Date/Time Created	X-SD-DCREATED	Option	
Character Set Code	X-SD-CHAR_CODE	Option	
Description	X-SD-DESCRIPTION	Option	

These extended properties correspond to the associated data in the PIM Data Information (PIMDI).

Note) In the case of PIMDI, the length of the Description data is limited, but **X-SD-DESCRIPTION** property is not limited logically. Therefore it is recommended to copy the leading data of this extended property to **DESCRIPTION** field of PIMDI.

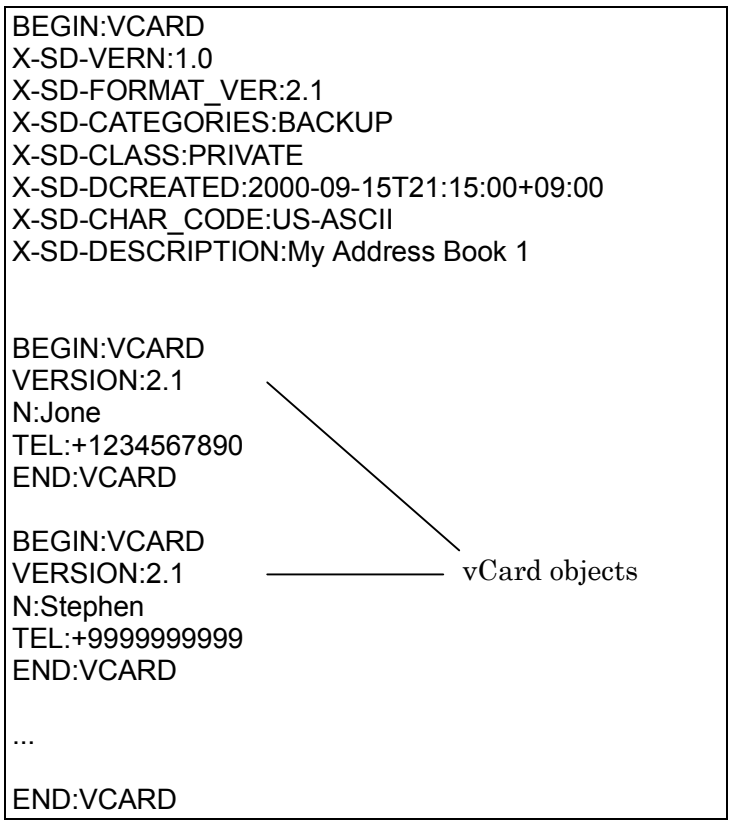
These extended properties must appear after the **BEGIN:Vxxxx** delimiter but before the occurrence of all the **Vxxxx** objects contained in a PIM Data file. These properties apply to the **Vxxxx** objects as a whole.

©Copyright 2001 by SD Card Association All rights reserved.

5.2 Address Book Data File Format

Address Book data file format is based on the structure of the specification vCard (The Electronic Business Card) Version 2.1 defined.

Structure of Address Book data file is shown as below.



Description	Property	Implementation Level	Compliance with vCard Version 2.1
Formatted Name	FN	Option	✓
Name	N	Mandatory	✓
Nickname	NICKNAME	Option	1)
Photograph	PHOTO	Option	6) ✓
Birthday	BDAY	Option	✓
Delivery Address	ADR	Option	✓
Delivery Label	LABEL	Option	✓
Telephone Number	TEL	Option	2)
Electronic Mail	EMAIL	Option	3)
Mailer	MAILER	Option	✓
Time Zone	TZ	Option	✓
Geographic Position	GEO	Option	4)
Title	TITLE	Option	✓
Business Category	ROLE	Option	✓

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

Logo	LOGO	Option	6)	✓
Agent	AGENT	Option		✓
Organization Name and Organizational Unit	ORG	Option		✓
Categories	CATEGORIES	Option	1)	
Comment	NOTE	Option		✓
Product Identifier	PRODID	Option	1)	
Last Revision	REV	Option		✓
Sort String	SORT-STRING	Option	1)	
Sound	SOUND	Option	5)	✓
Uniform Resource Locator	URL	Option		✓
Unique Identifier	UID	Option		✓
Version	VERSION	Mandatory		✓
Classification	CLASS	Option	1)	
Public Key	KEY	Option		✓

1) These properties are compliant with vCard Version 3.0 (IETF RFC 2426).

2) Property **TEL** is basically compliant with vCard Version 2.1. This specification extends the following property parameter values.

Description	Property Parameter Value
Indicates a cellular number #2 *	CELL2
Indicates a cellular number #3 *	CELL3
Indicates a telephone number of mobile data terminal	PDA

* These values are used for DECT, PHS, IMT2000, etc.

3) Property **EMAIL** is basically compliant with vCard Version 2.1. This specification extends following property parameter values.

Description	Property Parameter Value
Indicates a home E-mail address	HOME
Indicates an office E-mail address	WORK
Indicates an E-mail address of facsimile	FAX
Indicates an E-mail address of message service	MSG
Indicates an E-mail address of cellular	CELL
Indicates an E-mail address of cellular #2	CELL2 *
Indicates an E-mail address of cellular #3	CELL3 *
Indicates an E-mail address of pager	PAGER
Indicates an E-mail address of car-phone	CAR
Indicates an E-mail address of mobile data terminal	PDA

* These values are used for DECT, PHS, IMT2000, etc.

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

4) Property **GEO** is basically compliant with vCard Version 2.1. This specification differs in two point listed below.
The first point is that two values of **GEO,** latitude and the longitude, should be written according to ITRF(International Terrestrial Reference Frame). It is recommended that these latitude and longitude are represented by numbers with at least six fractions.
The second point is that an altitude is permitted to be shown as an option after the latitude and the longitude.
A positive value of altitude represents that the point higher than the altitude standard level of ITRF. A negative value represents the lower point. It can be omitted the sign of +. The unit of the altitude is 'meter'.
The following is an example of this property:

```
GEO:37.778889,-122.766944,32.7
```

5) The pronunciation of the fields within the vCards should be specified using the **SOUND** field. It is recommended that readers are also capable to handle the format specified in Pronunciation Attribute Extension for vCard. Writers may write both formats, although they MUST contain the same information.
The following is an example of these property:

```
N:Public;John;;Mr.;Esq.
SOUND;X-IRMC-N:PUBLIK;JON;;;
```

6) Properties PHOTO, LOGO are compliant with vCard Version 2.1. However, it is strongly recommended to use image data formats allowed in other SD Memory Card Specifications, such as SD-AUDIO 1.1 specifications, SD-PICTURE 1.0 specifications, and SD-VIDEO 1.0 specifications.

The following extended properties are recommended.

Description	Property	Implementation Level	Compliance with vCard Version 2.1
Joint Name	X-JOINTNAME	Option	
Former Name	X-FORMERNAME	Option	
Blood Type	X-BLOODTYPE	Option	
Hobby	X-HOBBY	Option	
Group Number	X-GNO	Option	
Group Name	X-GN	Option	
Speed Dial Number	X-SDNUMBER	Option	
PIN Code	X-PIN	Option	
Memorized Dial Number	X-MDNO	Option	

Joint Name

This extended property specifies joint names which may appear in the originator or the destination of the letter. This property is identified by the property name **X-JOINTNAME**. The following is an example of this property:

```
X-JOINTNAME:Jone;Kate
```

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

Former Name

This extended property specifies a former name. This property is identified by the property name **X-FORMERNAME**. The following is an example of this property:

```
X-FORMERNAME:Morley
```

Blood Type

This extended property specifies a blood type. This property is identified by the property name **X-BLOODTYPE**. The following is an example of this property:

```
X-BLOODTYPE:AB
```

Hobby

This extended property specifies a hobby. This property is identified by the property name **X-HOBBY**. The following is an example of this property:

```
X-HOBBY:skiing
```

Group Number

This extended property specifies a group number of the address book data. This property is identified by the property name **X-GNO**. The following is an example of this property:

```
X-GNO:12
```

Group Name

This extended property specifies a group name associated with the group number as shown above. This property is identified by the property name **X-GN**. The following is an example of this property:

```
X-GN:MySportsClub
```

Speed Dial Number

This extended property specifies a speed dial number which enables to call quickly with only two or three buttons. This property is identified by the property name **X-SDNUMBER**. The following is an example of this property:

```
X-SDNUMBER:01
```

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

PIN Code

This extended property specifies a PIN Code. This property is identified by the property name **X-PIN**. The following is an example of this property:

```
X-PIN:1234
```

Memorized Dial Number

This extended property specifies a memorized dial number which enables to associate a certain dial number or a person easily, or to sort the address book data, especially in case of the mobile phone. This property is identified by the property name **X-MDNO**. The following is an example of this property:

```
X-MDNO:183
```

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

5.3 Schedule / To Do List

Schedule/To Do List data file format is based on the structure of the specification vCalendar (The Electronic Calendaring and Scheduling Exchange Format) Version 1.0.

Structure of Schedule/To Do List data file is shown as below.

```
BEGIN:VCALENDAR
X-SD-VERN:1.0
X-SD-FORMAT_VER:1.0
X-SD-CATEGORIES:EXPORT
X-SD-CLASS:PRIVATE
X-SD-DCREATED:2000-09-15T21:15:00+09:00
X-SD-CHAR_CODE:US-ASCII
X-SD-DESCRIPTION:My Schedule 1


BEGIN:VCALENDAR
VERSION:1.0                          ——————  vCalendar
BEGIN:VEVENT                                 objects
DESCRIPTION:Lunch
DTSTART:20001008T120000
END:VEVENT
END:VCALENADAR

BEGIN:VCALENDAR
VERSION:1.0
BEGIN:VTODO
DALARM:20000228T150000;PT5M;2;CallHome!
END:VTODO
END:VCALENDAR

...

END:VCALENDAR
```

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

Description	Property	Implementation Level	Compliance with vCalendar Version1.0
Version	VERSION	Mandatory	✓
Daylight Saving Rule	DAYLIGHT	Option	✓
Geographic Position	GEO	Option	✓
Product Identifier	PRODID	Option	✓
Time Zone	TZ	Option	✓
Attachment	ATTACH	Option	✓
Attendee	ATTENDEE	Option	✓
Auto Reminder	AALARM	Option	✓
Categories	CATEGORIES	Option	✓
Classification	CLASS	Option	✓
Date/Time Created	DCREATED	Option	✓
Date/Time Completed	COMPLETED	Option	✓
Description	DESCRIPTION	Mandatory	✓
Display Reminder	DALARM	Option	✓
Due Date/Time	DUE	Option	✓
End Date/Time	DTEND	Option	✓
Exception Rule	EXRULE	Option	✓
Last Modified	LAST-MODIFIED	Option	✓
Location	LOCATION	Option	✓
Mail Reminder	MALARM	Option	✓
Number Recurrences	RNUM	Option	✓
Priority	PRIORITY	Option	✓
Procedure Reminder	PALARM	Option	✓
Related To	RELATED-TO	Option	✓
Recurrence Date/Times	RDATE	Option	✓
Recurrence Rule	RRULE	Option	✓
Resources	RESOURCES	Option	✓
Sequence Number	SEQUENCE	Option	✓
Start Date/Time	DTSTART	Option	✓
Status	STATUS	Option	✓
Summary	SUMMARY	Option	✓
Time Transparency	TRANSP	Option	✓
Uniform Resource Locator	URL	Option	✓
Unique Identifier	UID	Option	✓

Though the table above shows both Start Date/Time(DTSTART) and End Date/Time(DTEND) are optional, vEvent should have either property of them.

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

5.4 Message

Message data file format is based on the structure of the specification vMessage Version 1.1.

Structure of Message Data File is shown as below.

```
BEGIN:VMSG
X-SD-VERN:1.0
X-SD-FORMAT_VER:1.1
X-SD-CATEGORIES:ARCHIVE
X-SD-CLASS:PRIVATE
X-SD-DCREATED:2000-09-15T21:15:00+09:00
X-SD-CHAR_CODE:US-ASCII
X-SD-DESCRIPTION:My Folder 1


BEGIN:VMSG
VERSION:1.1
BEGIN:VCARD
VERSION:2.1
N:John
TEL:012-3456
END:VCARD
BEGIN:VENV
BEGIN:VBODY
Subject:Remainder
Don't forget to buy a newspaper.
END:VBODY
END:VENV
END:VMSG

BEGIN:VMSG
VERSION:1.1
BEGIN:VCARD
VERSION:2.1
N:Tom
TEL:999-9999
END:VCARD
BEGIN:VENV
BEGIN:VBODY
From:Mary@office
Subject:Meeting
2pm room 21.
END:VBODY
END:VENV
END:VMSG

...

END:VMSG
```

vMessage objects

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

Description	Property	Implementation Level	Compliance with vMessage Version 1.1
Version	VERSION	Mandatory	✓
Read Indication for vMessage	X-IRMC-STATUS	Option	✓
Message Type Indication for vMessage	X-IRMC-TYPE	Option	✓
A Box to direct Messages for vMessage	X-IRMC-BOX	Option	1)

1)Property **X-IRMC-BOX** is basically compliant with vMessage Version 1.1, but in this specification, it is recommended that property parameter values will contain one of the following values.

Description	Property Parameter Value
A collection of messages that have been received	INBOX
A collection of messages that may be transmitted	OUTBOX
A collection of messages that have been transmitted	SENTBOX

It is also possible to use a vendor specific destination box, such as Draft. If the receiving device does not understand the destination, it should default to the INBOX.

The following extension properties are recommended in this specification.

Description	Property	Implementation Level	Compliance with vMessage Version 1.1
Message Status	X-SD-STATUS	Option	
Message Group	X-SD-GROUP	Option	

Message Status

This extension property specifies a message status. This property is identified by the property name **X-SD-STATUS**. This property can have the following values:

Description	Property Value
Indicates the message is locked	LOCK
Indicates the message is unlocked	UNLOCK

The following is an example of this property:

```
X-SD-STATUS:LOCK
```

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

Message Group

This extension property specifies a message groups. This property is identified by the property name **X-SD-GROUP**. This property can have a string value as group name.

The following is an example of this property:

```
X-SD-GROUP:FRIENDS
```

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

5.5 Notes

Notes data file format is based on the structure of the specification vNote Version 1.1.

Structure of Notes data file is shown as below.

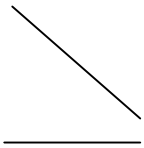
```
BEGIN:VNOTE
X-SD-VERN:1.0
X-SD-FORMAT_VER:1.1
X-SD-CATEGORIES:BACKUP
X-SD-CLASS:PRIVATE
X-SD-DCREATED:2000-09-15T21:15:00+09:00
X-SD-CHAR_CODE:US-ASCII
X-SD-DESCRIPTION:My Memo


BEGIN:VNOTE
VERSION:1.1
BODY:Things I want
to do.
END:VNOTE

BEGIN:VNOTE
VERSION:1.1
BODY:This is my important
thing.
END:VNOTE

...

END:VNOTE
```



Description	Property	Implementation Level	Compliance with vNote Version 1.1
Version	VERSION	Mandatory	✓
Unique ID	X-IRMC-LUID	Option	✓
Date/Time Created	DCREATED	Option	✓
Last Modified	LAST-MODIFIED	Option	✓
Summary	SUMMARY	Option	✓
Body	BODY	Mandatory	✓
Categories	CATEGORIES	Option	✓
Classification	CLASS	Option	✓

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

5.6 Bookmarks

Bookmarks data file format is based on the structure of the specification vBookmark Version 1.0.

Structure of Bookmarks data file is shown as below.

```
BEGIN:VBKM
X-SD-VERN:1.0
X-SD-FORMAT_VER:1.0
X-SD-CATEGORIES:BACKUP
X-SD-CLASS:PRIVATE
X-SD-DCREATED:2000-09-15T21:15:00+09:00
X-SD-CHAR_CODE:US-ASCII
X-SD-DESCRIPTION:My Bookmarks


BEGIN:VBKM
VERSION:1.0
URL:http://www.sdcard.org/
TITLE:SD Card Association Home Page
END:VBKM

BEGIN:VBKM
VERSION:1.0
URL:http://www.sdcard.org/news.htm
TITLE:SD Card Association News Page
END:VBKM

...

END:VBKM
```

vBookmark objects

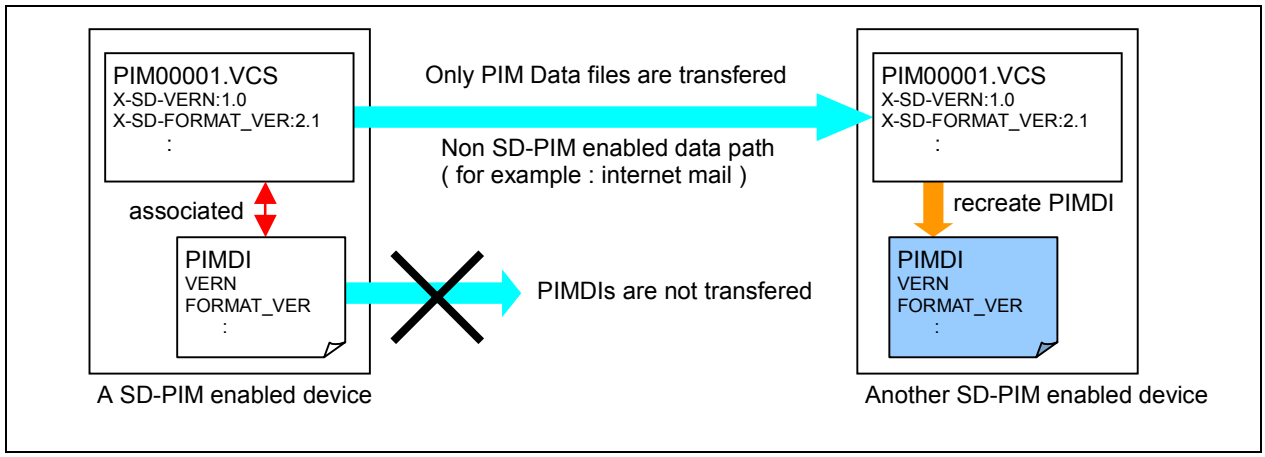
Description	Property	Implementation Level	Compliance with vBookmark Version 1.0
Version	VERSION	Mandatory	✓
URL	URL	Mandatory	✓
TITLE	TITLE	Option	✓
X-IRMC-URL	X-IRMC-URL	Option	✓

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

Annex A. Relationship between PIMDI/PIMMG and PIM Data files

In order to realize an interoperability with non SD-PIM specification enabled devices, this specification recommends to put the same attributes of PIMDI into PIM data files as a header. This enables the reader devices to import PIM data received via internet mail and to reconstruct PIMDI easily as shown below.



©Copyright 2001 by SD Card Association All rights reserved.

Annex B. Character Set Code

Numbers of character sets or encoding names are defined by IANA.
The following numbers can be used as character set codes in this specification.

Character Set code … Assigned MIB enum by IANA : Character Sets name
3 : US-ASCII
4 : ISO8859-1
5 : ISO8859-2
6 : ISO8859-3
7 : ISO8859-4
8 : ISO8859-5
9 : ISO8859-6
10 : ISO8859-7
11 : ISO8859-8
12 : ISO8859-9
15 : JIS X 0201
17 : Shift-JIS
106 : UTF-8 [RFC2279]
2001 : ISO-8859-1-Windows-3.1-Latin-1
2002 : ISO-8859-2-Windows-Latin-2
2003 : ISO-8859-9-Windows-Latin-5

Notes:
・Euro (currency symbol) is often assigned 0x80.

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

Annex C. Language Tag

Language Tags defined in RFC1766 “Tags for the Identification of Languages” are used in this specification.

For example:

“en-us”	American English
“en”	English
“fr”	French
“du”	German
“ja”	Japanese
“ko”	Korean
“zh”	Chinese

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

This page is intentionally left blank

©Copyright 2001 by SD Card Association All rights reserved.

CONFIDENTIAL

SD Memory Card Specifications

Part 10
IMAGE SPECIFICATION

Version 1.0 Draft

January 2001

Technical Committee

SD Association

Conditions for publication

Publisher and Copyright Holder

SD Association

Exemption

None will be liable for any damages from use of this document.

ã 2001 SD Association

This page is intentionally left blank.

ã 2001 SD Association

Part X IMAGE SPECIFICATION

1. General 1

1.1. Scope 1
1.2. General Specification of Presentation Data 2
1.2.1. Presentation Data of SD-Image Specification 2
1.2.2. Functions of the SD-Image Specification 4
1.3. Normative References 5

2. Technical Elements 7

2.1. Definitions 7
2.2. Arithmetic Operators 7
2.3. Notations 8
2.3.1. Numerical Notation 8
2.3.2. Range 8
2.3.3. Bit Fields 8
2.4. Terminology 9
2.5. List of Abbreviations 11

3. Introduction 13

3.1. Functions of the SD-Image Specification 13
3.1.1. Image Binder 13
3.1.2. Playlist 15
3.1.3. Content Meta Data 17
3.2. Structure of Directories and Files for Functions 18
3.2.1. Structure of Image Binder and Playlist 18
3.2.2. Structure of All Directories and Files for SD-Image 19
3.3. Navigation Structure 25
3.3.1. Identification Information of SD-Image 25
3.3.2. Access Method to Image Objects and Identification Information for Image Object 29
3.3.3. Titles and Other Texts 30
3.3.4. Structure for Playlist 31
3.3.5. Resuming Last Playback 31

4. Directory and File Naming Convention 32

4.1. SD-Image Directory 32
4.2. Management Data Directory 32
4.3. Image Binder Directory 33
4.4. Image Binder Component File 34
4.4.1. Image Binder Infomation File 35
4.4.2. Still Image Data File 36

ã 2001 SD Association

4.4.3. Animation Image Data File 37
4.4.4. Thumbnail Image Data File 38
4.4.5. Content Meta-Data File of Image Binder or Object 39
4.5. Scene Description 40
4.5.1. Scene Description Data File 40
4.5.2. Content Meta-Data File of Scene Description 41
4.6. Method of File Name Generation with Decimal Number 42

5. Navigation Data Structure 43

5.1. Miscellaneous Definitions 43
5.1.1. Content ID 43
5.1.2. Data Types 44
5.2. Considerations 44
5.2.1. Reserved Fields 44
5.2.2. Version 44
5.3. Management Data File 46
5.4. Image Binder Manager File 48
5.5. Image Binder Information File 52
5.6. Object Identification Information File 56
5.7. Playlist Manager File 59
5.8. Playlist Information File 61
5.9. Content Meta-Data File 63

6. Scene Description 67

7. Annexes 68

Annex A. Mandatory Items in File of SD-Image Specification 68
Annex A.1. MGR_DATA 68
Annex A.2. IMB_MGR 68
Annex A.3. IBI (Image Binder Information) 69
Annex A.4. PLST_MGR 69
Annex A.5. PLST_PLI (Playlist Information) 70
Annex A.6. OBJ_IDI 70
Annex A.7. CMD (Content Meta-Data) 71
Annex B. Mandatory Items in Data Structure of SD-Image Specification 72
Annex B.1. CNTID 72
Annex B.2. REFOBJ 72
Annex B.3. MGR_IDI (Identification Information for Manager) 73

ã 2001 SD Association

List of Figures

Figure 1.1-1: Position of SD-Image Specification 1
Figure 3.1-1: Display Examples of Identification Information of Image Binder 13
Figure 3.1-2: Relation between Image Binder Information and Image Binder 14
Figure 3.1-3: Display Examples of Identification Information of Playlist 15
Figure 3.1-4: Relation among Playlist, Image Binder, and Slide Show 16
Figure 3.1-5: Object and Content Meta Data File 17
Figure 3.2-1: File Structure of Image Binder and Playlist 18
Figure 3.2-2: Structure of All Directories and Files for SD-Image 19
Figure 3.2-3: Directory and Scene Description Data File 23
Figure 3.3-1: All Identification Information Files in SD-Image 25
Figure 3.3-2: Identification Information for Image Binders 26
Figure 3.3-3: Identification Information for Playlist 28
Figure 3.3-4: Access Method to Image Objects and Identification Information 29
Figure 3.3-5: Title for Image Object and Playlist 30
Figure 3.3-6: Data Structure for Playlist 31
Figure 5.4-1:Structure of ImbIDITbl in IMB_MGR File 48
Figure 5.6-1: Structure of Object Identification Information File 56
Figure 5.8-1: Structure of Playlist Information File 61
Figure 6-1: Scene Description 67

ã 2001 SD Association

List of Tables

Table 1.2-1: Presentation Data....2
Table 1.2-2: SD-Image Specification and Presentation Data....3
Table 1.2-3: File Structure of Image Binder....4
Table 1.2-4: File Structure of Playlist....4
Table 1.2-5: File Structure of Content Meta-Data....4
Table 2.4-1: Content Meta-Data File in the SD-Image Directory....10
Table 3.2-1: Attribute of Files in Management Data Directory....20
Table 3.2-2: Attribute of Object Files in Image Binder Directory....21
Table 3.2-3: Attribute of Management Files and Meta-Data Files in Image Binder....22
Table 3.2-4: Attribute of Scene Description Files....23
Table 3.3-1: Identification Information in SD-Image....25
Table 4.2-1: Management Files in Management Data Directory....32
Table 4.3-1: Specification of Naming Convention of Image Binder Directory....33
Table 4.4-1: Image Binder Component Files under Image Binder Directory....34
Table 4.4-2: Specification of Naming Convention of Image Binder Information File....35
Table 4.4-3: Specification of Naming Convention of Still Image Data File....36
Table 4.4-4: Specification of Naming Convention of Animation Image Data File....37
Table 4.4-5: Specification of Naming Convention of Thumbnail Data File....38
Table 4.4-6: Specification of Naming Convention of Content Meta-Data File for IMB....39
Table 4.4-7: Specification of Naming Convention of Content Meta-Data File for Object....39
Table 4.5-1: Specifications of Naming Convention of Scene Description Data File....40
Table 4.5-2: Specification of Naming Convention of Meta-Data File of Scene Description....41
Table 5.3-1: MGR_DATA....46
Table 5.4-1: IMB_MGR....48
Table 5.4-2: MGR_IDI....50
Table 5.5-1: IMB_INFO....52
Table 5.5-2: REFOBJ....54
Table 5.6-1: OBJ_IDI....56
Table 5.7-1: PLST_MGR....59
Table 5.8-1: PLST_INFO....62
Table 5.9-1: CMD....63
Table 5.9-2: META_DATA....64

ã 2001 SD Association

List of Table Annexes

Table Annex A- 1: MGR_DATA 68
Table Annex A- 2: IMB_MGR 68
Table Annex A- 3: IBI 69
Table Annex A- 4: PLST_MGR 69
Table Annex A- 5: PLST_PLI 70
Table Annex A- 6: OBJ_IDI for Object 70
Table Annex A- 7: CMD 71

Table Annex B- 1: CNTID 72
Table Annex B- 2: REFOBJ 72
Table Annex B- 3: IDI for Each Manager File 73

ã 2001 SD Association

1. General

1.1. Scope

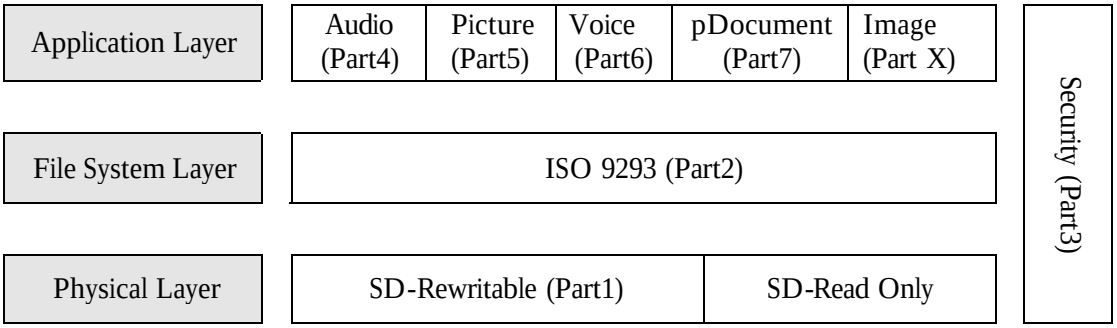
This SD-Image Specification is the Part X of the SD Memory Card Specification Family. In the SD-Image Specification, image presentation data complies with Bitmap, GIF (87a, 89a), JPEG (Exif, JFIF), or PNG Standards depending on an application.

The following features are not supported in this version. These features will be supported in a later version of the SD-Image Specification if necessary.

- CPRM
- Sound

The SD Memory Card applying this specification shall comply with **"Part1. PHYSICAL SPECIFICATION", "Part2. FILE SYSTEM SPECIFICATION",** and **"Part3. SECURITY SPECIFICATION"** of the SD Memory Card Specifications.

Figure 1.1-1: Position of SD-Image Specification



1.2. General Specification of Presentation Data

The SD-Image Specification consists of Objects to be presented and Functions to handle Objects. The features of the Objects and Functions in the SD-Image Specification are as follows.

1.2.1. Presentation Data of SD-Image Specification

SD-Image handles two types of Objects: Image Objects and Sound Objects. Image Objects are mandatory, and Sound Objects are optional. Sound Objects will be supported in a later version if necessary.
The features of these Objects and other presentation data in this specification are shown in Table 1.2-1.

Table 1.2-1: Presentation Data

Data	General Specification	
Image	Coding Mode of Still Images	Bitmap GIF (87a, 89a) JPEG (Exif2.1) JPEG (JFIF) PNG
	Coding Mode of Animations Number of Pixels (horizontal x vertical)	GIF(89a) 96x96 (Recommended) 160x120 (Recommended) 160x120 – 2560x1920 (Optional)
	Chrominance sampling (JPEG)	4:2:2 4:2:0
Text	ISO646, JISX0201, ISO8859-1, JISX0208	
Scene Description	SMIL 2.0 Basic Profile	
Sound	This feature will be supported in a later version if necessary.	

Table 1.2-2: SD-Image Specification and Presentation Data

Data	Item	In Card	For Players
Image	Bitmap	At least one coding mode is mandatory	
	GIF (87a, 89a) JPEG (Exif2.1)		
	JPEG (JFIF)		
	PNG		
	GIF (89a) Animation		
Sound	This feature will be supported in a later version if necessary.	Optional	Optional
Text	ISO646, JISX0201, ISO8859-1, JISX0208	Optional	Optional
Scene Description	SMIL 2.0 Basic Profile	Optional	Optional

The coding mode of Still Image Objects in this specification shall be Bitmap, GIF (87a, 89a), JPEG (Exif2.1), JPEG (JFIF) or PNG. The coding mode of Animations shall be GIF (89a). It is mandatory that each WRITER is able to store Image Objects with at least one image coding mode stated above. Also it is mandatory that each PLAYER is able to render them with at least one image coding mode stated above.

The horizontal size and the vertical size of Image Objects shall not be larger than 2560 x 1920 pixels, respectively, to ensure compatibility with the SD-Picture Specification. All WRITERs are strongly expected to be able to store recommended sizes of Image Objects. Also all PLAYERs are strongly expected to be able to render them.

The coding mode of Sound Objects in this specification is T.B.D. It is optional that a WRITER is able to store Sound Objects. Also it is optional that a PLAYER is able to play them.

No WRITER shall store any Sound Object irrelevant to any Image Objects. Also no PLAYER shall play it without showing any Image Objects at the same time.

1.2.2. Functions of the SD-Image Specification

This SD-Image Specification provides functions to handle Objects. Those functions and their structure are described below.

1.2.2.1. Image Binder

Image Binder is a mandatory function. An Image Binder is defined as a group of Image Objects with optional Sound Objects in one Image Binder Directory.

Table 1.2-3: File Structure of Image Binder

Hierarchy name	Maximum number
Image Binder Manager	1 per card
Image Binder Directory	900 per card
Image Binder Information File	1 per Image Binder Directory
Object File	999 per Image Binder Directory

1.2.2.2. Playlist

Playlist is an optional function. A Playlist is defined as a group of favorite Image Objects with optional Sound Objects in one or more Image Binder Directories.

Table 1.2-4: File Structure of Playlist

Hierarchy name	Maximum number
Playlist Manager	1 per card
Playlist Information File	1 per card (up to 999 pieces of Playlist Information per file)

1.2.2.3. Content Meta-Data

Content Meta Data is an optional function. Content Meta Data is defined as optional information data about an Image Binder, Image Object.

Table 1.2-5: File Structure of Content Meta-Data

Hierarchy name	Maximum number
Content Meta Data of Image Binder	900 per card
Content Meta Data of Image Object	999 per Image Binder

1.3. Normative References

1) ISO / IEC 646: 1983 (ASCII)
Information processing – ISO 7-bit coded character set for information interchange

2) ISO 8859-1: 1987
Information processing – 8-bit single-byte coded graphic character sets – Part 1
Latin alphabet No.1

3) JIS X 0201: 1997 (Kana)
7-bit and 8-bit coded character sets for information interchange

4) JIS X 0208: 1997 (Shift JIS Kanji)
7-bit and 8-bit 2-byte coded Kanji character sets for information interchange

5) W3C Working Draft 21 September 2000 (SMIL)
Synchronized Multimedia Integration Language: (SMIL 2.0) Specification

6) SD Memory Card Specifications
Part 1: PHYSICAL SPECIFICATION

7) SD Memory Card Specifications
Part 2: FILE SYSTEM SPECIFICATION

8) SD Memory Card Specifications
Part 3: SECURITY SPECIFICATION

9) SD Memory Card Specifications
Part 4: AUDIO SPECIFICATIONS

10) SD Memory Card Specifications
Part 5: PICTURE SPECIFICATIONS

11) SD Memory Card Specifications
Part 6: VOICE SPECIFICATIONS

12) SD Memory Card Specifications
Part 7: pDOCUMENT SPECIFICATIONS

13) ISO / IEC 9293: 1994

Information technology - Volume and file structure of disk cartridges for information interchange

2. Technical Elements

2.1. Definitions

In this specification, the following definitions shall be applied :

Mandatory	All mandatory functions shall be implemented. Any WRITER or PLAYER is NOT compliant with this specification unless it implements every one of mandatory functions.
Optional	WRITERs and PLAYERs that are compliant with this specification may implement optional functions defined in it. Manufactures shall make sure of inter-operability among WRITERs and PLAYERs that implement such functions.
Recommended	Recommended functions are strongly expected to be implemented, but may NOT be implemented in full specs. WRITERs and PLAYERs that are compliant with this specification shall perform as many recommended functions as possible.
Reserved	Reserved fields. These are reserved for the future use of this specification. Unless otherwise specified within this specification, all bits of these shall be set to zero.
Zero	Represents a single bit with the value 0.
Data_element[m..n]	The inclusive range of bits between bit m and bit n in the data_element

2.2. Arithmetic Operators

+	Addition
-	Subtraction
*	Multiplication

2.3. Notations

2.3.1. Numerical Notation

Numbers in decimal notation are represented by decimal digits, namely 0 to 9. **Digit(s)** represents any digit from digit 0 to 9.

Numbers in hexadecimal notation are represented as a sequence of one or more hexadecimal digits namely 0 to 9 and A to F, suffixed by "h".

Multiple-byte numerical values in a description field shall be recorded in the **big endian** representation. e.g.) The 32-bit hexadecimal number '12345678h' shall be recorded as '12h', '34h', '56h', '78h'.

Numbers in decimal notation are represented as "xxx" in the case of Image Binder Number from 100 to 999 and are represented as "nnnn" in the case of File Number from 0001 to0999. The File Number is a 4-digit number for future extension.

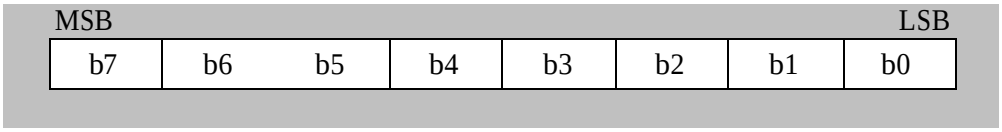
1 KB represents 2^{10} bytes.
1 MB represents 2^{20} bytes.
1 GB represents 2^{30} bytes.

2.3.2. Range

Constant_1..Constant_2 or [Constant1..Constant2] denotes the range from Constant_1 up to Constant_2, both inclusive, in increments of 1.

2.3.3. Bit Fields

Certain fields or parts of fields are interpreted as an array of bits.
This array of bits shall be referred to as a bit field. Bit positions within an ***n*** bit field are numbered such that the least significant bit is numbered 0 and the most significant bit is numbered ***n–1***.



2.4. Terminology

SD-Image Directory

This is the top directory named "SD_IMAGE" in the root directory. All of SD-Image compliant files and directories shall be created under this directory.

Object

There are two kinds of Objects: Image Objects and Sound Objects. Image Objects include Still Images and Animation Images.

Still Image Data File

This file contains a still image. It is one of Image Object Files. Its format shall be Bitmap, GIF (87a, 89a), JPEG (Exif), JPEG (JFIF), or PNG. It may be referred to by Playlists or Scene Description Data Files.

Animation Image Data File

This file contains an animation image. It is one of Image Object Files. Its format shall be GIF (89a) called GIF Animation. It may be referred to by Playlists or Scene Description Data Files.

Thumbnail Image Data File

This file contains a piece of thumbnail data of another still image. Its format shall be Bitmap, GIF (87a, 89a), JPEG (Exif), JPEG(JFIF), or PNG.

Sound Data File

This file contains a piece of sound data. It is a Sound Object File. It may be played only while Image Objects are shown. The feature will be defined in a later version if necessary.

Scene Description Data File

This file contains a scene described in the Scene Description Language, SMIL. It may be created in Image Binder Directories or in the Scene Description Data File Directory.

Identification Information File

This file contains identification information such as the Title and Create Time of an Image Object, Image Binder, or Playlist.

Content Meta-Data File

This file contains meta data such as the description and author of an Image Object or Image Binder. All Content Meta-Data Files are shown in Table 2.4-1.

Table 2.4-1: Content Meta-Data File in the SD-Image Directory

	File Name	Abbreviation	in directory
Image Binder	Image Binder CMD File	IMBCMD	Image Binder Directory (xxxSDIMG)
Image Object	Object CMD File	OBJCMD	

Management Data File

This file contains global information of SD-Image management data, including its version number, volume label, and other information.

Manager File

There are two kinds of Manager Files: Image Binder Manager File and Playlist Manager File. Each of them contains the number of Image Binders or Playlists as well as other information in the SD-Image Directory.

Information File

There are two kinds of Information Files: Image Binder Information File and Playlist Information File. Each of them contains the number of Objects and information on each Object.

Image Binder Number

This is a unique decimal number (from 100 to 999) assigned to each Image Binder.

File Number

This is a unique decimal number (from 0001 to 0999) assigned to each file in the SD-Image Directory.

Content ID

This is a 32-bit ID to identify an Image Binder, Playlist, Scene Description Data File, or Object such as a Still Image Data File, Animation Image Data File.

Resume Marker

This is a Content ID which indicates an Object in an Image Binder or Playlist where its display has been stopped.

2.5. List of Abbreviations

ANIM	Prefix of Animation File
BMP	[Extension of] Bit Map File
GIF	[Extension of] Graphics Interchange Format
IBI	Image Binder Information File
IDI	Image Identification Information
IEC	International Electrotechnical Commission
IMB	Image Binder
IMB_INFO	Image Binder Information of an Image Binder
IMB_MGR	Image Binder Manager
IMBCMD	Content Meta-Data File of an Image Binder
INFO	Information
INT	Integer
ISO	International Organization for Standardization
JIS	Japanese Industrial Standards
JPG	[Extension of] JPEG (Joint Photographic coding Experts Group) File
MGR_DATA	[Directory of] Management Data
MGR_INFO	Management Information
Num	Number
OBJ	Object
OBJ_IDI	Image Identification Information of Image Object
OBJCMD	Content Meta-Data File of an Object
CMD	[Extension of] Content Meta-Data [File]
PICT	Prefix of Still Image File
PLST_INFO	Playlist Information
PLST_MGR	Playlist Manager
PLI	Playlist Information File
PNG	[Extension of] Portable Network Graphics
POS	Position
PLAYER	Device that can render images in the SD-Image Directory
SC1	Extension of Secure Content Meta-Data File
SCEN	Extension of Scene Description Data File
SD	Secure Digital
SDIMG	Suffix of Image Binder Directory Name
SG1	Extension of Secure Animation Image Data File
SI1	Extension of Secure Still Image Data File
SL1	Extension of Secure SML File
SML	SMIL (Synchronized Multimedia Integration Language)
SN1	Extension of Secure Animation Image Data File
T.B.D	To Be Defined

Tbl	Table
THMN	Prefix of a Thumbnail File
UINT	Unsigned Integer
WRITER	Device that can write images in the SD-Image

3. Introduction

3.1. Functions of the SD-Image Specification

In this section, the data structure of SD-Image contents is outlined. First of all, the important functions on which all SD-Image contents are based are expounded.

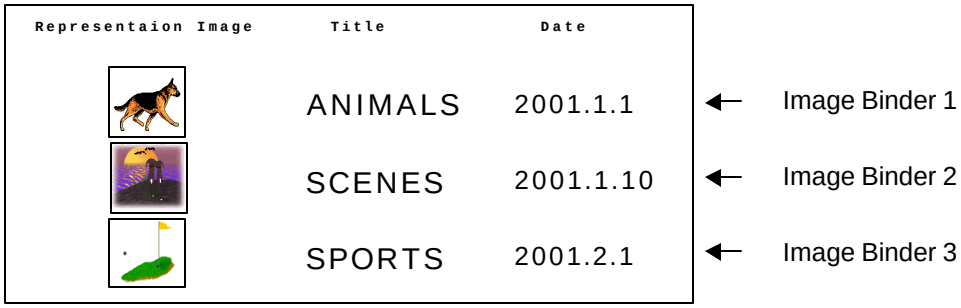
3.1.1. Image Binder

3.1.1.1. Image Binder Manager and its Handling

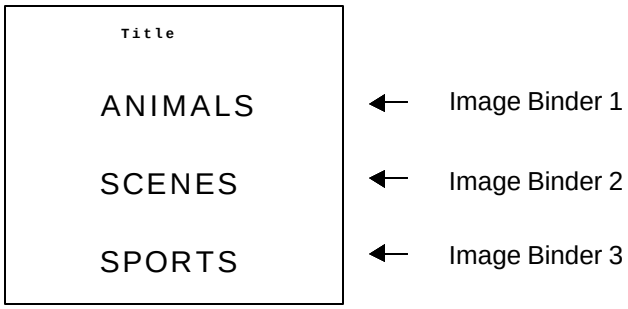
An Image Binder is defined as a group of Image Objects and some related files. All files of an Image Binder are created under an Image Binder Directory.
A WRITER may create or delete Image Binders for each specified use. When a WRITER creates or deletes an Image Binder, it shall set or delete some information in the Image Binder Manager File and its related Image Binder Information File.
A WRITER may set Identification Information (IDI) in the Image Binder Manager File such as Representation Object File, Title, and Date of each Image Binder.
The minimum number of Image Binders shall be "0". Therefore, until any Object is registered in its Image Binder, the number of Image Binders shall be "0".

Figure 3.1-1: Display Examples of Identification Information of Image Binder

Example 1



Example 2



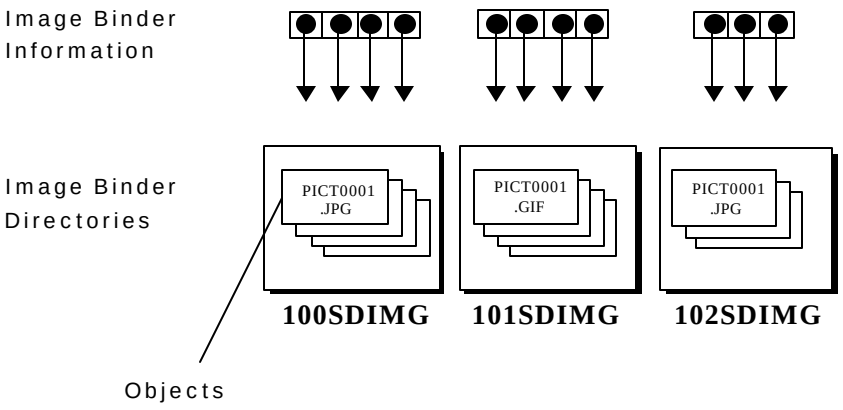
3.1.1.2. Image Object and its Handling

When a WRITER registers an Object to an Image Binder, some information shall be set to the Image Binder Information File in the same directory. On the other hand, when a WRITER deletes an Object File from an Image Binder, some information on the Object shall be deleted from the Image Binder Information File.

A WRITER may set Identification Information (IDI) such as Title of each Object File. Identification Information of an Object File is stored in an Object Identification Information File(OBJ_IDI) in the Image Binder Directory.

Even when the minimum Object file number of a certain Image Binder is "0", its Image Binder Information File exists.

Figure 3.1-2: Relation between Image Binder Information and Image Binder



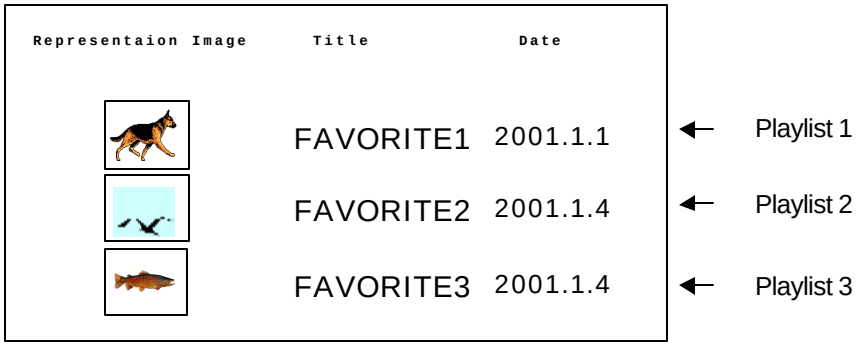
3.1.2. Playlist

3.1.2.1. Playlist and its Handling

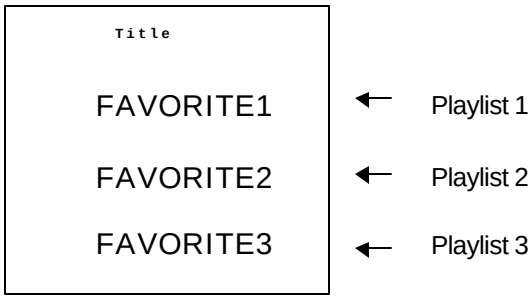
A Playlist is defined as a group of favorite Objects. A Playlist may refer to any Object in any Image Binder of SD-Image. User-defined Objects lists in SD -Image can be made by the playlist function.
A WRITER may create or edit a Playlist for each specified use and handle Object Files in each Playlist. When a WRITER creates or deletes a Playlist, it shall set or delete some information on Objects in the Playlist Manager File and Playlist Information File.
A WRITER may set Identification Information (IDI) in the Playlist Manager File such as Representation Object File and Title of each Playlist.
The minimum number of Playlists shall be “0”. Therefore, until any Object is registered in its Playlist, the number of Playlists shall be “0”. Even when the number of Playlists is “0”, PLST_PLI (Size:0) shall exist.

Figure 3.1-3: Display Examples of Identification Information of Playlist

Example1



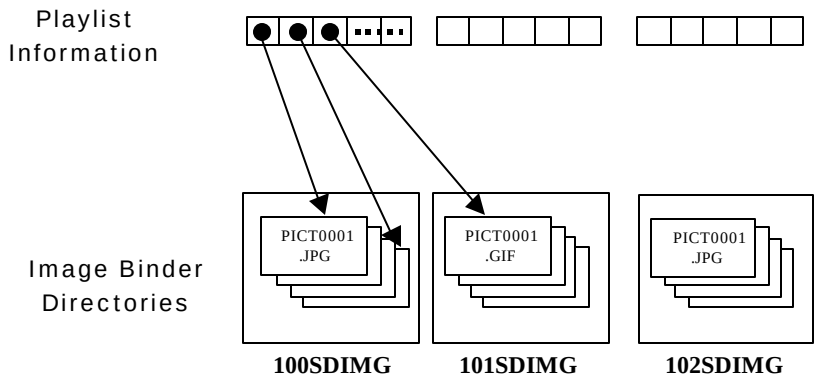
Example2



3.1.2.2. Playlist and Object

When a WRITER registers an Object to a Playlist, some information on the Object shall be set to the Playlist Information File in the Management Data Directory. On the other hand, when a WRITER deletes an Object from a Playlist, some information on the Object shall be deleted from the Playlist Information File. A PLAYER can identify the Object list according to the information described in the Playlist Information File.

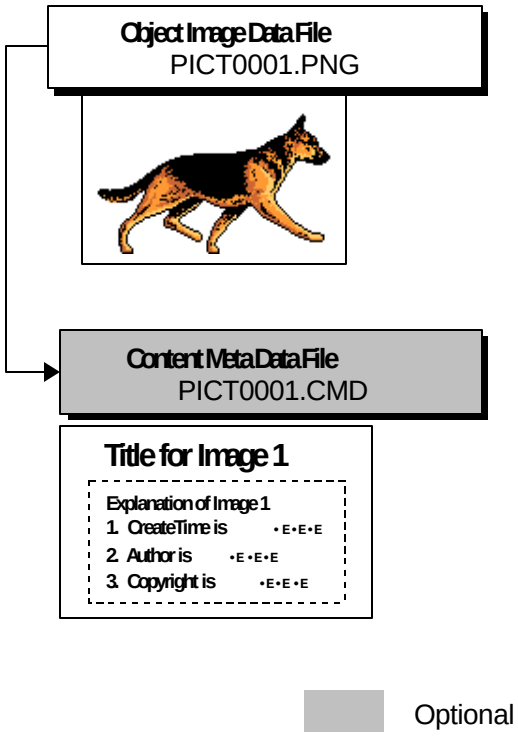
Figure 3.1-4: Relation among Playlist, Image Binder, and Slide Show



3.1.3. Content Meta Data

A Content Meta Data File contains information about an Object Data File or Image Binder. This information is called Meta-Data. Content Meta Data has many kinds of meta data, such as Title, CreationTime (in milliseconds), Author, Copyright, Description about Object, Recording Company, Editor, Thumbnail information of Image Object, and Rotation information of Image Object.
A PLAYER can notice the existence of a Content Meta Data File about an Object File or Image Binder by checking the Content Meta Data existence flag in an Image Binder Information File.

Figure 3.1-5: Object and Content Meta Data File

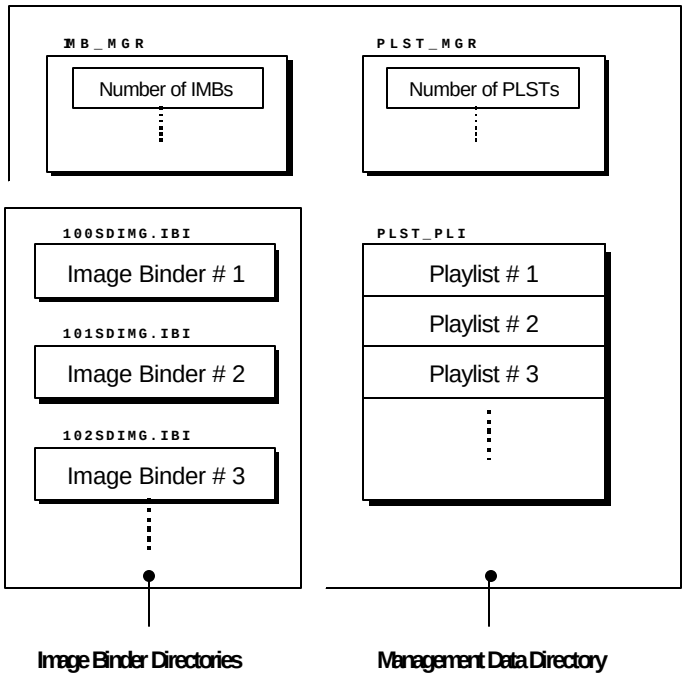


3.2. Structure of Directories and Files for Functions

3.2.1. Structure of Image Binder and Playlist

The file structure of Image Binders and Playlists is illustrated in Figure 3.2-1.

Figure 3.2-1: File Structure of Image Binder and Playlist



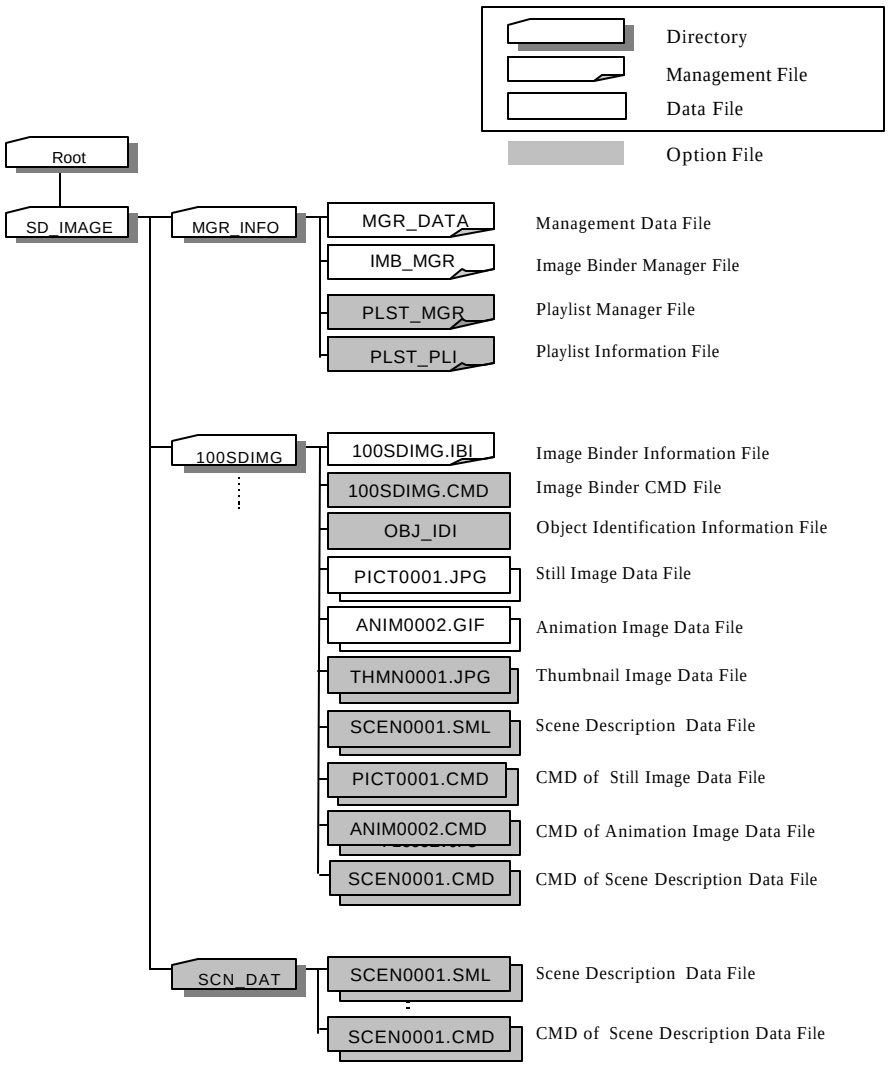
The **Management Data Directory** shall contain one **Image Binder Manager File** (IMB_MGR) that includes the number of Image Binders and other information.
Also it may contain one **Playlist Manager File** (PLST_MGR) that includes information on all Playlists (e.g. the number of Playlists) and one **Playlist Information File** (PLST_PLI) which includes the Image Objects list of each Playlist.

Each Image Binder Directory shall contain one **Image Binder Information File** (100SDIMG.IBI~) that includes information on all Image Objects in itself.

3.2.2. Structure of All Directories and Files for SD-Image

The structure of all directories and files is illustrated in Figure 3.2-2.

Figure 3.2-2: Structure of All Directories and Files for SD-Image



The SD-Image Directory (SD_IMAGE) is the top directory in this specification. Management Data Directory (MGR_INFO), Image Binder Directories (xxxSDIMG), and Scene Description Data Directory (SCN_DAT) shall be placed under the SD-Image Directory.

3.2.2.1. Management Data Directory and Files

A **Management Data Directory** named "MGR_INFO" is mandatory in the SD-Image Directory. This Management Data Directory includes the following files. The attribute of each file is outlined below.

Table 3.2-1: Attribute of Files in Management Data Directory

File Type	File name	M/O
Management Data File	MGR_DATA	Mandatory
Image Binder Manager File	IMB_MGR	Mandatory
Playlist Manager File	PLST_MGR	Optional
Playlist Information File	PLST_PLI	Optional

A **Management Data File** named 'MGR_DATA' is mandatory. It contains global information that is created, used, and updated by WRITERs at the initial or final access to a SD Memory Card. The file contains the version number of this specification, a reference to the position where suspended display will be resumed, and other related information. WRITERs and PLAYERs shall read this file and use the global information.

An **Image Binder Manager File** named 'IMB_MGR' is mandatory. It contains the number of Image Binders as well as management and Identification Information on all Image Binders in the SD-Image Directory. The Identification Information, which is optional, includes Title and Representation objects of all Image Binders.

A **Playlist Manager File** named 'PLST_MGR' is optional. It contains the number of Playlists as well as management and Identification Information on all Playlists in the SD-Image Directory. The Identification Information, which is optional, includes Title and Representation objects of all Playlists.

A **Playlist Information File** named 'PLST_PLI' is optional. It contains management information on all Playlists in the SD-Image Directory. It contains the list of Objects for each Playlist.

3.2.2.2 Image Binder Directory and Files

An **Image Binder Directory** named "xxxSDIMG" is mandatory in the SD-Image Directory. Each Image Binder Directory includes many kinds of files. The attribute of each file is outlined below.

Table 3.2-2: Attribute of Object Files in Image Binder Directory

File Type	File Name	M/O
Still Image Data File	PICTnnnn.JPG/GIF/PNG/BMP	At least one coding mode is mandatory
Animation Image Data File	ANIMnnnnGIF	
Thumbnail Data File	THMNnnn.JPG/GIF/PNG/BMP	Optional

A **Still Image Data File** contains a still image. Its name shall have the 4-character prefix 'PICT' followed by 4-digit File Number in decimal notation and the 3-character extension 'BMP', 'GIF', 'JPG', or 'PNG' preceded by a period. If this file is encrypted, its extension shall be changed to 'SI1'.

An **Animation Image Data File** contains an animation image. Its name shall have the 4-character prefix 'ANIM' followed by 4-digit File Number in decimal notation and the 3-character extension 'GIF' preceded by a period. If this file is encrypted, its extension shall be changed to 'SG1'.

A **Thumbnail Image Data File** contains thumbnail image data on a still image. Its name shall have the 4-character prefix 'THMN' followed by 4-digit File Number in decimal notation and the 3-character extension 'BMP', 'GIF', 'JPG', or 'PNG' preceded by a period. If this file is encrypted, its extension shall be changed to 'SI1'.

ã 2001 SD Association

Table 3.2-3: Attribute of Management Files and Meta-Data Files in Image Binder

File Type	File Name	M/O
Image Binder Information File	xxxSDIMG.IBI	Mandatory
Object Identification Information File	OBJ_IDI	Optional
CMD of Image Object	PICTnnnn.CMD ANIMnnnn.CMD	Optional
CMD of Image Binder	xxxSDIMG.CMD	Optional

An **Image Binder Information File** is mandatory. It contains the number of Image Objects and management information on each Image Binder in the SD-Image Directory. It also contains the list of Image Objects for every Image Binder. The file name shall be 'xxxSDIMG.IBI'.

An **Object Identification Information File** contains the identification information on all Image Objects in an Image Binder. It contains identification information such as Title and Creation Time of all Objects in every Image Binder. The file name shall be 'OBJ_IDI'.

A **Content Meta-Data File of Image Object** contains the Meta-Data of its related Still Image Data File or Animation Image Data File. Its name shall have the 4-character prefix 'PICT' or 'ANIM' followed by 4-digit File Number in decimal notation and the 3-character extension 'CMD' preceded by a period. If this file is encrypted, its extension shall be changed to 'SC1'.

A **Content Meta-Data File of Image Binder** contains the Meta-Data of its related Image Binder. Its name shall have the 3-digit File Number in decimal notation followed by 5-character suffix 'SDIMG' and the 3-character extension 'CMD' preceded by a period. If this file is encrypted, its extension shall be changed to 'SC1'.

3.2.2.3. Scene Description Data Directory and Files

Scene Description Data Directory

This is an optional directory named "SCN_DAT" in the SD-Image Directory and contains Scene Description Data Files.

Scene Description Data Files shall be in the Image Binder Directories or the Scene Description Data Directory. Scene Description Files that control Object Files within an Image Binder Directory may be created only under the same directory. Scene Description Files that control Object Files in two or more Image Binders may be created only under the Scene Description Data Directory.

The name of the Scene Description Data Directory shall be 'SCN_DAT'. The Attributes of Scene Description Data Files are described below.

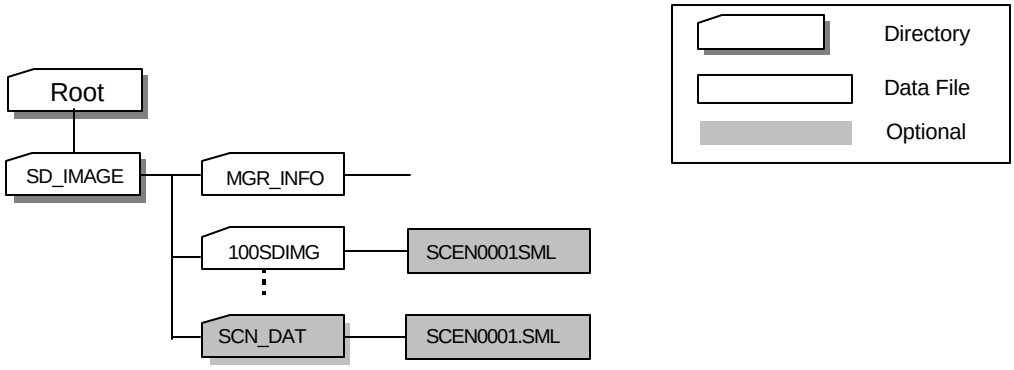
Table 3.2-4: Attribute of Scene Description Files

Data Type	File Name	M/O
Scene Description Data File	SCENnnnn.SML	Optional
CMD of Scene Description Data File	SCENnnnn.CMD	Optional

A **Scene Description Data File** is a SMIL document that specifies the scene description of either an Image Binder or Scene Description Data File Directory. Its name shall have the 4-character prefix 'SCEN' followed by 4-digit File Number in decimal notation and the 3-character extension 'SML' preceded by a period. If this file is encrypted, its extension shall be changed to 'SL1'.

Each File Number may be provided in each Image Binder Directory or Scene Description Data Directory. For example, Figure 3.2-3 shows a case where "SCEN0001.SML" exists in "100SDIMG" and "SCEN0001.SML" exists in "SCN_DAT".

Figure 3.2-3: Directory and Scene Description Data File



Though a Scene Description Data File may be described in two or more Scene Description Data Files in each Directory, only the file that has the name 'SCEN001.SML' in each Directory shall be used to provide the initial Scene Description. The others are referred to directly or indirectly by the initial description.

3.3. Navigation Structure

In this section, the structure and mechanism of navigation data in the SD-Image specification are outlined.

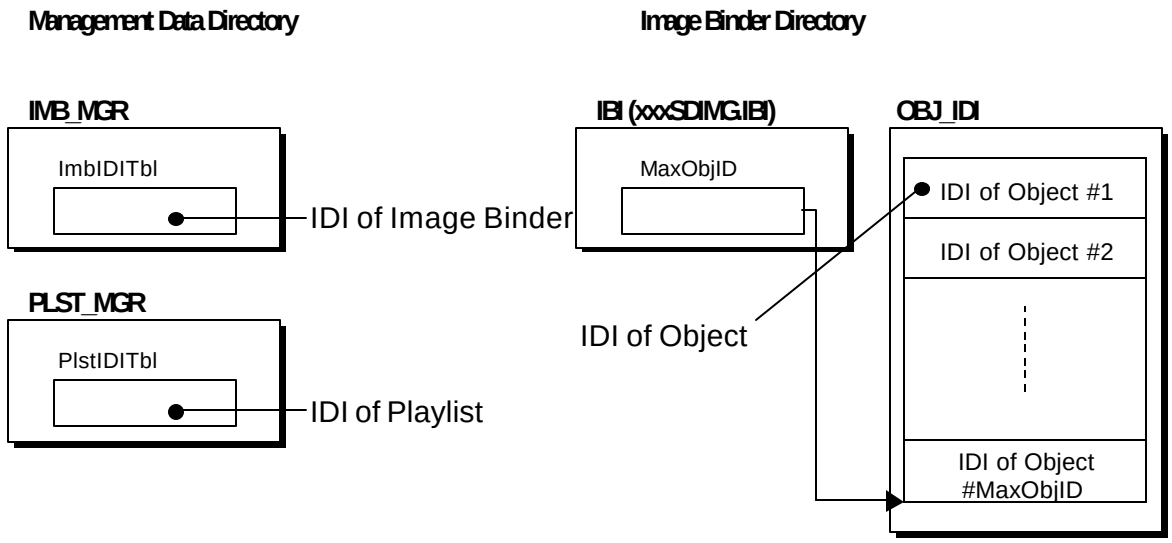
3.3.1. Identification Information of SD-Image

The Identification Information (IDI) such as Titles and Create Times of Image Binders, Playlists, and Objects in the SD-Image Directory are described.

Table 3.3-1: Identification Information in SD-Image

	Described in	Abbreviation	M/O(File existence)
Image Binder	Image Binder Manager	IMB_MGR	Mandatory
Playlist	Playlist Manager	PLST_MGR	Mandatory for Playlist
Image Object	Object Identification Information File	OBJ_IDI	Option

Figure 3.3-1: All Identification Information Files in SD-Image



PLAYERs can obtain IDIs of valid Image Binders or Playlists by checking all records of ImbIDITbl' in IMB_MGR or 'PlstIDITbl' in PLST_MGR. In addition, PLAYERs can obtain IDIs of Objects in an Image Binder Directory by checking all records of the OBJ_IDI file in the directory.

3.3.1.1. Identification Information for Image Binders

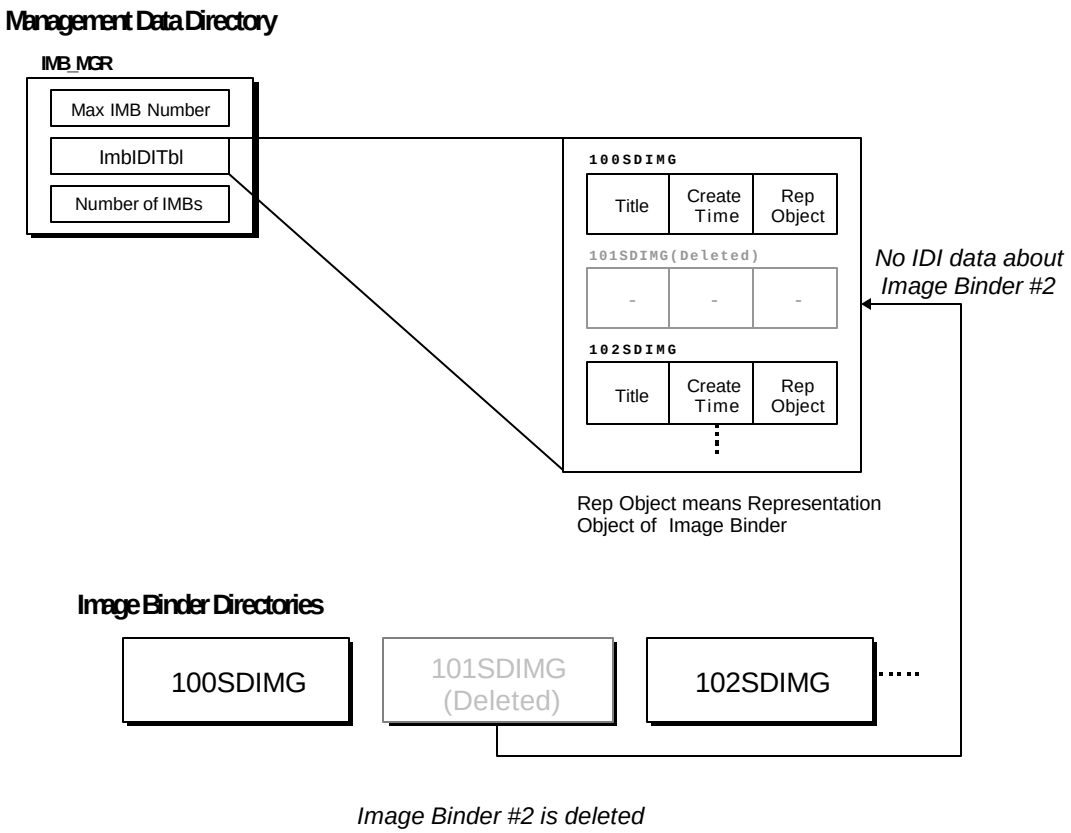
The structure of Identification Information of an **Image Binder Manager** (IMB_MGR) is described below.

An **Image Binder Manager** (IMB_MGR) contains such fields as 'Number of IMBs', 'MaxIMB Number', and the table 'ImbIDITbl. 'Number of IMBs' is the number of Image Binders in the SD-Image. 'MaxIMB Number' is the maximum Image Binder Number between "100" and "999", both inclusive.

The 'ImbIDITbl' has some valid records and/or some invalid records. A valid record in the table contains valid information related to an existing Image Binder. On the other hand, an invalid record in the table contains an invalid value indicating a deleted Image Binder.

The record of a deleted Image Binder in ImbIDITbl shall have its CreationTime field filled with '0xFF', and other fields except the DataType field shall be filled with any value. (The DataType field shall be a correct value.) If there is no information to be set in the record in ImbIDITbl of an Image Binder that exists, each field except the DataType field shall be filled with '0x00.' (The DataType field shall be a correct value.)

Figure 3.3-2: Identification Information for Image Binders



When WRITERs create a new Image Binder, a new record shall be appended to 'ImbIDITbl' and the values of both 'Number of IMBs' and 'Max IMB Number' shall be incremented. When WRITERs delete an Image Binder, its related record shall be set to an invalid value in the 'ImbIDITbl' (In the CreationTime field, '0xFF' shall be filled.). If the value of the Image Binder Number of the deleted Image Binder is not equal to the 'Max IMB Number', the value of 'Number of IMBs' shall be decremented, but the value of 'Max IMB Number' shall remain the same.
If an Image Binder with its Image Binder Number equal to 'Max IMB Number' is deleted, 'Max IMB Number' shall be updated to a new 'Max IMB Number'.
(Refer to the number assignment rules for Image Binder which is described in section 4.6.)

PLAYERs can obtain all IDIs of valid Image Binders by checking all records in the 'ImbIDITbl' without searching all Image Binder Directories.

3.3.1.2. Identification Information for Playlist

The structure of 'PlstIDITbl' in PLST_MGR for a Playlist is the same as that of 'ImbIDITbl' in IMB_MGR for an Image Binder. Playlist Manager, 'Max Playlist Number', and 'Number of Playlists' for Playlist correspond to Image Binder Manager, 'MaxIMB Number', and 'Number of IMBs', respectively.

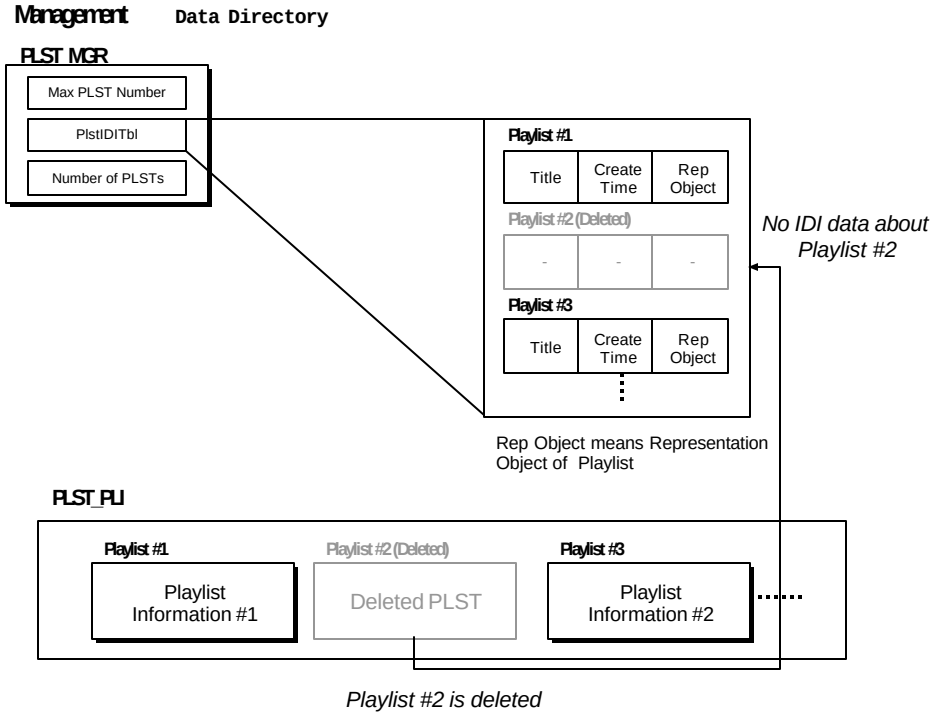
When WRITERs create a new Playlist, a new record shall be appended to 'PlstIDITbl' and the values of both 'Number of Playlists (PLSTs)' and 'Max PLST Number' shall be incremented.

When WRITERs delete a Playlist, its related record shall be set to an invalid value in the 'PlstID ITbl' (In the CreationTime field, '0xFF' shall be filled.). If a Playlist with its Playlist Number not equal to the 'Max PLST Number' is deleted, the value of 'Number of PLSTs' shall be decremented but the value of the 'Max PLST Number' shall remain the same.

If a Playlist with its Playlist Number equal to 'Max PLST Number' is deleted, the 'Max PLST Number' shall be updated to a new 'Max PLST Number'.

PLAYERs can obtain all IDIs of valid Playlists only by checking all records in the 'PlstIDITbl'.

Figure 3.3-3: Identification Information for Playlist



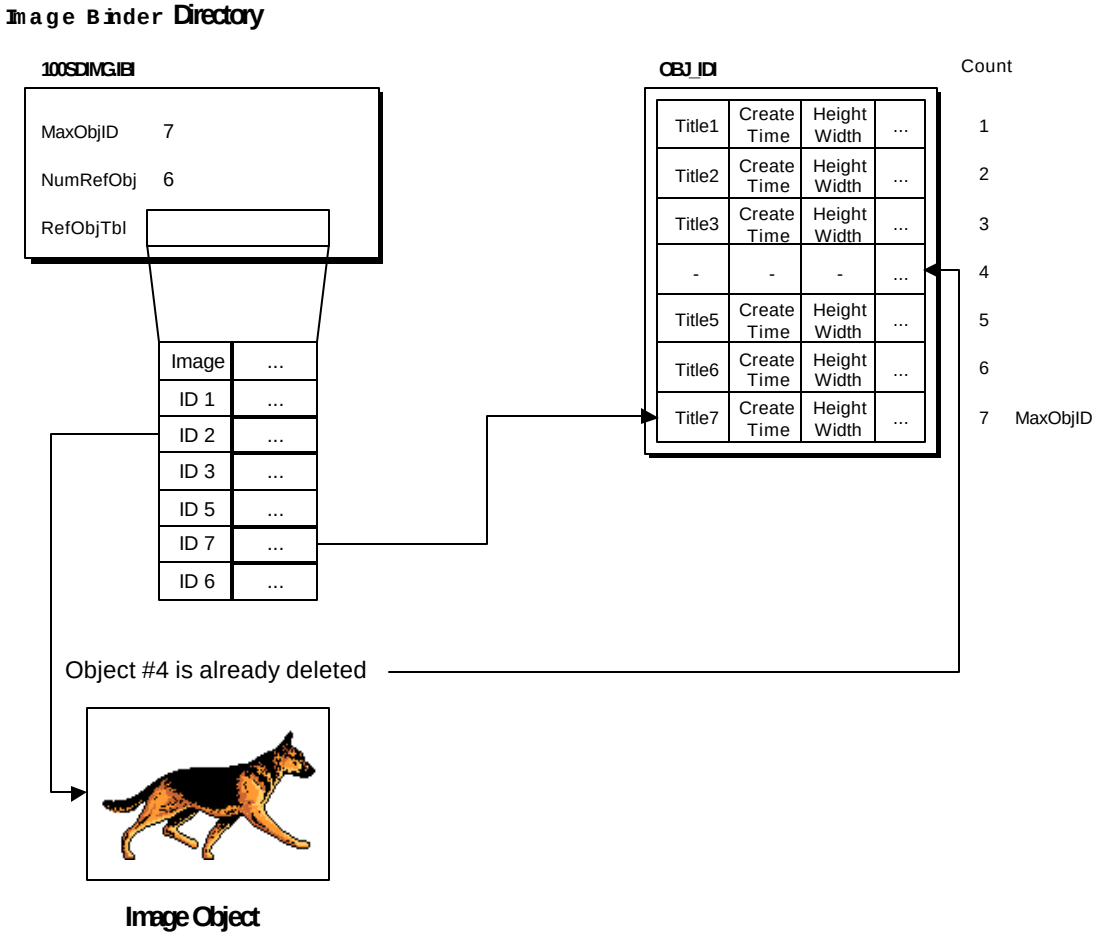
3.3.2. Access Method to Image Objects and Identification Information for Image Object

An **Image Binder Information File** (IBI) in an Image Binder contains such fields as 'NumRefObj', 'MaxObjID', and the table 'RefObjTbl'. The 'NumRefObj' is the number of Objects in the Image Binder. The 'MaxObjID' is the maximum file number in it. The 'RefObjTbl' is a table which contains the ID of an Image Object, and other information. Each Image Object is linked with the related record in the file OBJ_IDI. They are capable of random access by selecting an arbitrary Image Object from the 'RefObjTbl'.

The record in OBJ_IDI of the deleted Image Object shall have its CreationTime field filled with '0xFF' and other fields shall be filled with any value except DataType field (DataType field shall be correct value). If there is no information to set to the record in OBJ_IDI of an Image Object that exist, each field shall be filled with '0x00' except DataType field (DataType field shall be correct value).

Refer to the number assignment rules for Image Object which is described in section 4.6.

Figure 3.3-4: Access Method to Image Objects and Identification Information



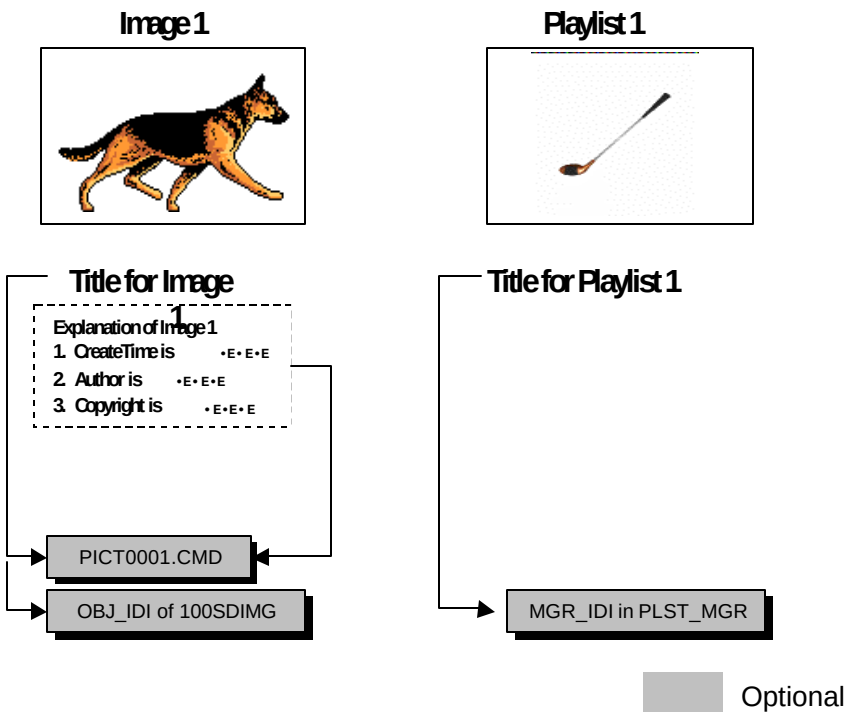
3.3.3. Titles and Other Texts

Several kinds of text information can be handled in this SD-Image Specification. Text information such as Title and Creation Time of an Image Object is stored in each **Object Identification Information File** (OBJ_IDI). In the case of Image Binders, such text information is stored in each **Image Binder Manager File** (IMB_MGR), and in the case of Playlists, in each **Playlist Manager File** (PLST_MGR).

In addition, optional **Content Meta Data Files** of Image Objects also include text information such as Title. Title contained in OBJ_IDI shall be simple and short, and Title in Content Meta-Data Files shall be longer than the one in OBJ_IDI.

If a Content Meta-Data file and its related file has the same data, a PLAYER may select the data to display either in the IDI File or in the CMD File.

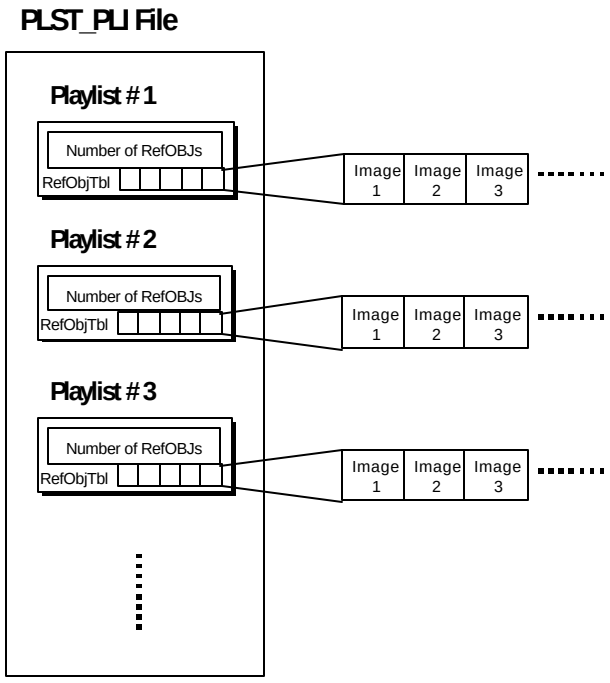
Figure 3.3-5: Title for Image Object and Playlist



3.3.4. Structure for Playlist

By creating or editing a **Playlist Manager** (PLST_MGR) and a **Playlist Information File** (PLST_PLI), users can define a favorite group of Image Objects. Image Objects in a Playlist are specified in the RefObjTbl of Playlist information in the PLST_PLI file as shown below.

Figure 3.3-6: Data Structure for Playlist



3.3.5. Resuming Last Playback

Even if a user turns off a PLAYER while displaying an Image Object, the PLAYER can describe the ContentID of the Image Object, Image Binder, or Playlist indicating the last display or playback in the **Resume Marker** field in the Management Data File. Thus, whenever a PLAYER is turned off, Resume Marker shall be updated to the correct Resume Marker. If a PLAYER does not support the resuming function, it shall set the Resume Marker field to zero when it is turned off.

4. Directory and File Naming Convention

This chapter describes the naming convention of directories and files in SD_Image.

4.1. SD-Image Directory

A directory named "SD_IMAGE" directly under the root directory is called SD -Image Directory.

4.2. Management Data Directory

A directory named "MGR_INFO" directly under the SD-Image Directory is called Management Data Directory.
This directory contains files shown in Table4.2-1.

Table 4.2-1: Management Files in Management Data Directory

File	File Name
Management Data File	MGR_DATA
Image Binder Manager File	IMB_MGR
Playlist Manager File	PLST_MGR
Playlist Information File	PLST_PLI

4.3. Image Binder Directory

A Directory that contains Image Objects called Image Binder Directory. It is created directly under the SD-Image Directory. The naming convention for the Image Binder Directory is as follows.

- The directory name shall be 8 characters in length.
- The first 3 characters shall be a decimal number between "100" and "999", both inclusive. The number less than "100" shall not be used.
- The next 5 characters shall be "SDIMG".
- Extensions shall not be used.

The specification of this naming convention is shown in Table 4.3-1.

Table 4.3-1: Specification of Naming Convention of Image Binder Directory

	Decimal number part (Image Binder Number)			Fixed string part				
Specification	1	0	0	S	D	I	M	G
	~							
	9	9	9					
Example	1	2	3	S	D	I	M	G

4.4. Image Binder Component File

The following Image Binder Component Files are created under an Image Binder Directory and constitute an Image Binder.

Table 4.4-1: Image Binder Component Files under Image Binder Directory

File	File Name
Image Binder Information File	xxxSDIMG.IBI
Object Identification Information File	OBJ_IDI
Still Image Data File	PICTnnnn.BMP PICTnnnn.GIF PICTnnnn.JPG PICTnnnn.PNG
Animation Image Data File	ANIMnnnn.GIF
Thumbnail Data File	THMNnnnn.BMP/GIF/JPG/PNG
Scene Description Data File	SCENnnnn.SML
Content Meta-Data File	xxxSDIMG.CMD PICTnnnn.CMD ANIMnnnn.CMD SCENnnnn.CMD

xxx : from 100 to 999
nnnn : from 0001 to 0999

4.4.1. Image Binder Information File

An Image Binder Information File contains the number and information of Image Objects. The naming convention for this file is as follows.

- The file name shall be 8 characters as base name and 4 characters as extension (including period).
- The first 3 characters shall be a decimal number between "100" and "999", both inclusive. The number less than "100" shall not be used.
- The next 5 characters shall be "SDIMG".
- The extension shall be ".IBI".

The specification of this naming convention is shown in Table 4.4-2.

Table 4.4-2: Specification of Naming Convention of Image Binder Information File

	Decimal number part (File Number)			Fixed string part					Alternative string part			
Specification	1	0	0	S	D	I	M	G	.	I	B	I
	≀											
	9	9	9									
Example	1	2	3	S	D	I	M	G	.	I	B	I

4.4.2. Still Image Data File

A Still Image Data File contains a single still image. The naming convention for this file is as follows.

- The file name shall be 8 characters as base name and 4 characters as extension (including period).
- The first 4 characters shall be "PICT".
- The next 4 characters shall be a decimal number between "0001" and "0999", both inclusive. The number "0000" shall not be used. The number used by Animation Image Data Files shall not be used. For example, the coexistent of 'ANIM0001.GIF' and 'PICT0001.JPG' is prohibited.
- The extension shall be one of ".BMP", ".GIF", ".JPG", or ".PNG". If this file is encrypted, its extension shall be ".SI1".

The specification of this naming convention is shown in Table 4.4-3.

Table 4.4-3: Specification of Naming Convention of Still Image Data File

	Fixed string part				Decimal number part (File Number)				Alternative string part			
Specification	P	I	C	T	0	0	0	1	.	B	M	P
									.	G	I	F
									.	J	P	G
									.	P	N	G
									or			
					0	9	9	9	.	S	I	1
Example	P	I	C	T	0	1	2	3	.	P	N	G

4.4.3. Animation Image Data File

An Animation Image Data File contains animation images. The naming convention for this file is as follows.

- The file name shall be 8 characters as base name and 4 characters as extension (including period).
- The first 4 characters shall be "ANIM".
- The next 4 characters shall be a decimal number between "0001" and "0999", both inclusive. The number "0000" shall not be used. The number used by Still Image Data Files shall not be used. For example, the coexistence of 'PICT0001.JPG' and 'ANIM0001.GIF' is prohibited.
- The extension shall be ".GIF". If this file is encrypted, its extension shall be ".SG1".

The specification of this naming convention is shown in Table 4.4-4.

Table 4.4-4: Specification of Naming Convention of Animation Image Data File

	Fixed string part				Decimal number part (File Number)				Alternative string part			
Specification	A	N	I	M	0	0	0	1	.	G	I	F
					~				or			
					0	9	9	9	.	S	G	1
Example	A	N	I	M	0	1	2	3	.	G	I	F

4.4.4. Thumbnail Image Data File

A Thumbnail Image Data File contains a still image. The naming convention for this file is as follows.

- The file name shall be 8 characters as base name and 4 characters as extension (including period).
- The first 4 characters shall be "THMN".
- The next 4 characters shall be a decimal number between "0001" and "0999", both inclusive. The number "0000" shall not be used. The same number as in its Original file shall be used. The Thumbnail Image Data file of a Still Image Data File can exist in the SD-Image.
- The extension shall be one of ".BMP", ".GIF", ".JPG", or ".PNG". If this file is encrypted, its extension shall be ".SI1".

The specification of this naming convention is shown in Table 4.4-5.

Table 4.4-5: Specification of Naming Convention of Thumbnail Data File

	Fixed string part				Decimal number part (File Number)				Alternative string part			
Specification	T	H	M	N	0	0	0	1	.	B	M	P
					2				.	G	I	F
									.	J	P	G
									.	P	N	G
									or			
					0	9	9	9	.	S	I	1
Example	T	H	M	N	0	1	2	3	.	P	N	G

4.4.5. Content Meta-Data File of Image Binder or Object

Each Content Meta-Data File for Image Binders and Objects contains Meta-Data of an Image Binder and an Object, respectively. Each file is recorded in an Image Binder Directory. The naming convention for this file is as follows.

- For an Image Binder, the file name shall be 8 characters as base name and 4 characters as extension (including period).
- The first 3 characters shall be a decimal number between "100" and "999", both inclusive. The number less than "100" shall not be used.
- The next 5 characters shall be "SDIMG".
- For an Object, the file name shall be 8 characters as base name and 4 characters as extension (including period).
- The first 4 characters shall be "PICT" for Still Images, or "ANIM" for Animation Images.
- For an Object, the next 4 characters shall be a decimal number between "0001" and "0999", both inclusive. The number "0000" shall not be used.
- The extension shall be ".CMD". If this file is encrypted, its extension shall be ".SC1".

The specification of this naming convention is shown in Table 4.4-6. and Table 4.4-7.

Table 4.4-6: Specification of Naming Convention of Content Meta-Data File for IMB

	Decimal number part (File Number)			Fixed string part					Alternative string part			
Specification	1	0	0	S	D	I	M	G	.	C	M	D
	≀											
	9	9	9									
Example	1	2	3	S	D	I	M	G	.	C	M	D

Table 4.4-7: Specification of Naming Convention of Content Meta-Data File for Object

	Fixed string part				Decimal number part (File Number)				Alternative string part			
Specification	P	I	C	T	0	0	0	1	.	C	M	D
	A	N	I	M	≀				or			
	S	U	N	D	0	9	9	9	.	S	C	1
Example	P	I	C	T	0	1	2	3	.	C	M	D

4.5. Scene Description

4.5.1. Scene Description Data File

A Scene Description Data File is in an Image Binder Directory or the "SCN_DAT" Directory, and it contains a Scene Description for the playback of Image Objects. The naming convention for this file is as follows.

- The file name shall be 8 characters as base name and 4 characters as extension (including period).
- The first 4 characters shall be "SCEN".
- The next 4 characters shall be a decimal number between "0001" and "0999", both inclusive. The number "0000" shall not be used.
- The extension shall be ".SML". If this file is encrypted, its extension shall be ".SL1".

The specification of this naming convention is shown in Table 4.5-1.

Table 4.5-1: Specifications of Naming Convention of Scene Description Data File

	Fixed string part				Decimal number part (File Number)				Alternative string part			
Specification	S	C	E	N	0	0	0	1	.	S	M	L
					≀				or			
					0	9	9	9	.	S	L	1
Example	S	C	E	N	0	1	2	3	.	S	M	L

4.5.2. Content Meta-Data File of Scene Description

This file contains the Meta-Data of Scene Description. Its naming convention is as follows.

- The file name shall be 8 characters as base name and 4 characters as extension (including period).
- The first 4 characters shall be "SCEN".
- The next 4 characters shall be a decimal number between "0001" and "0999", both inclusive. The number "0000" shall not be used.
- The extension shall be ".CMD".

The specification of this naming convention is shown in Table 4.5-2.

Table 4.5-2: Specification of Naming Convention of Meta-Data File of Scene Description

	Fixed string part				Decimal number part (File Number)				Alternative string part			
Specification	S	C	E	N	0	0	0	1	.	C	M	D
					~				or			
					0	9	9	9	.	S	C	1
Example	S	C	E	N	0	1	2	3	.	C	M	D

4.6. Method of File Name Generation with Decimal Number

Each name of Image Binder Directories and Image Binder Component Files has a decimal number part. This part is assigned sequentially for each file type when a new file/directory is created. For example, PICT0001.BMP and PICT0002.PNG can coexist in the same Image Binder Directory, but PICT0001.BMP and PICT0001.SI1 can NOT coexist in the same Image Binder Directory. As described in the Still and Animation Image Data File sections, PICT0001.BMP and ANIM0001.GIF can NOT coexist in the same Image Binder Directory.

The number assignment rules are as follows.

(1) Find the maximum files/directory number (refer to as 'Max IMB Number' or 'MaxObjID') within the directory in which a new file/directory is being created.
(2) Create a file/directory with the number greater than the maximum file/directory number by one.

If the assigned number reaches the maximum number limit, new file/directory creation will fail. Or a WRITER may search a deleted number for reuse.

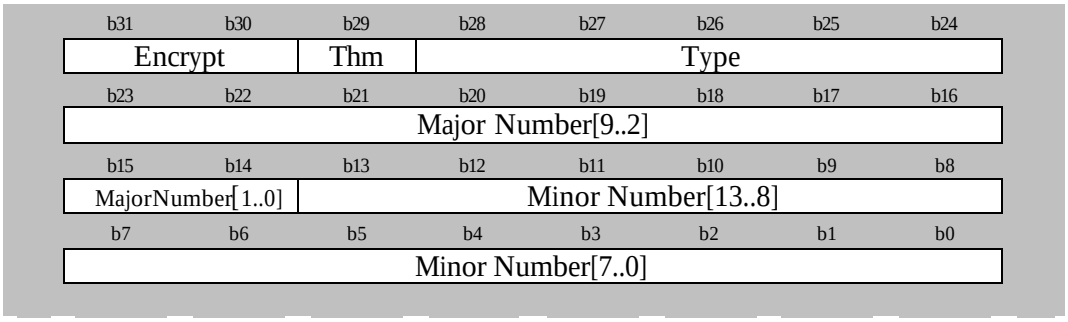
5. Navigation Data Structure

Navigation Data is used to render Image Objects. It specifies ways to present the objects, their identification information and Meta Data. The Navigation Data is stored in seven different types of files; Management Data File, Image Binder Manager File, Image Binder Information Files, Object Identification Information Files, Playlist Manager File, Playlist Information Files and Content Meta-Data Files.

5.1. Miscellaneous Definitions

5.1.1. Content ID

The Navigation Data Structure contains a field for Content ID. The Content ID is used to identify contents such as an Image Binder, Playlist, Still Image Object, Animation Image Object, thumbnail or Scene Description Data File. A Content ID has a 32-bit value associated with the file/directory name of an Object. The format of a Content ID is as follows.



Encrypt	0h: Unencrypted object 2h: Encrypted object (if Type is 01h, 02h, 03h, 04h, 05h, 09h or 0fh) Others: Reserved
Thm	0h: Not Thumbnail Data File 1h: Thumbnail Data File of Other Still Image DataFiles
Type	01h: Still Image Data File for GIF 02h: Still Image Data File for JPEG (Exif) 03h: Still Image Data File for JPEG (JFIF) 04h: Still Image Data File for PNG 05h: Still Image Data File for Bitmap 09h: Animation Image Data File for GIF 0fh: Scene Description Data File 10h: Image Binder 11h: Playlist Others: Reserved
Major Number	If the Type is 01h, 02h, 03h, 04h, 05h, 09h or 10h, this field shall be an Image Binder Number (100-999). If the Type is 11h, this field shall be a Playlist Number.

	If the Type is 0fh and Scene Description Data File exists in an Image Binder Directory, this field shall be an Image Binder Number. If the Scene Description Data File is in 'SCN_DAT' Directory, 0 shall be specified in this field.
Minor Number	If the Type is 10h or 11h, 0 shall be specified in this field. However when the ContentID is used for ResumeMarker in MGR_DATA, this field shall be specified with a sequence number from the top of 'RefObjTbl' in Image Binder Information or Playlist Information whose Object was resumed. This number shall start with 1. In neither of the above cases, this field shall be specified with the File Number of an Object or Scene Description File.

5.1.2. Data Types

UINT8

The type, UINT8 indicates an unsigned 8-bit data item.

UINT16

The type, UINT16 indicates an unsigned 16-bit data item.

UINT32

The type, UINT32 indicates an unsigned 32-bit data item.

UINT64

The type, UINT64 indicates an unsigned 64-bit data item.

CNTID

CNTID indicates an unsigned 32-bit value which maintains Content ID.

5.2. Considerations

5.2.1. Reserved Fields

There are several reserved fields in the data structure that are explained later. These fields are for future extension as well as compatibility with other specifications, and shall have the value of zero.

5.2.2. Version

There are also several version fields in data structures, which are MGR_DATA, IMB_MGR, PLST_MGR and CMD File, and explained later. The same number shall be filled in every one of these files. The Version fields have major version part and minor version part. The major version part indicates the major version of the data structure, which shall be increased when the structure cannot keep compatibility with an older version. The minor version part indicates the minor version

of the data structure, which shall be increased when the structure can keep compatibility with an older version. In the first version of the SD-Image, the major version shall be 1 and minor version zero.

A PLAYER or WRITER shall check the version and find out whether it is capable of accessing its data structure. If the major version of a PLAYER or WRITER is older than that of the data, a PLAYER or WRITER shall stop accessing the data. If the major version of a PLAYER or WRITER is a later or the same version of that of the data, a PLAYER or WRITER shall be capable of accessing the data.

5.3. Management Data File

Management Data File, which is mandatory, defines global characteristics of SD-Image management data. Its file name shall be "MGR_DATA". If SD-Image specification is supported, Management Data File shall exist under the "SD_IMAGE" directory..

Table 5.3-1: MGR_DATA

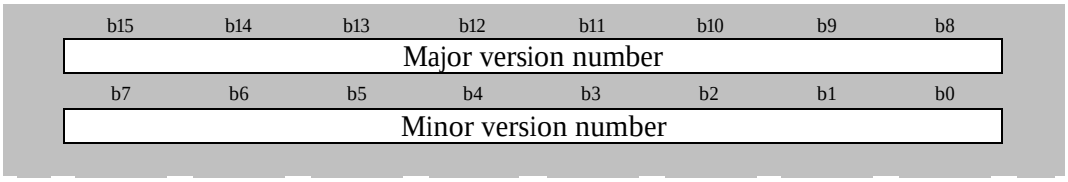
Field Type	Field Name	Contents	Field Size (Byte)
UNIT16	DataType	Data type identifier	2
UINT16	Version	Version number of MGR_DATA	2
CNTID	ResumeMarker	Position of the Object most recently played	4
UINT8	Reserved	Reserved	1
UINT8	TextAttribute	Attribute of VolumeLabel	1
UINT8	VolumeLabel[32]	Volume Label text for SD-Image	32
Total			42

DataType

This field shall be "I1" with character set code of ISO/IEC 646:1983(ASCII).

Version

Describes the version number of MGR_DATA.



In the first version, the major version number is 1 and the minor version 0.

ResumeMarker

Describes the position of the Image Binder or Playlist with the object that was the last to be displayed or played. The structure of the Resume Marker is CNTID.

TextAttribute

Describes a character code set which is used in VolumeLabel field of the MGR_DATA. The set value is as follows.

1h:	ISO/IEC 646:1983(ASCII)
2h:	JIS X 0201
3h:	ISO 8859-1
9h:	Shift-JIS kanji
Others:	Reserved

VolumeLabel

Describes the text information for SD-Image volume on an SD card. This field shall be terminated with the code '00h' unless the field is completely filled with the text information. The text information shall be formed by using character code sets that are specified in the TextAttribute field.

5.4. Image Binder Manager File

Image Binder Manager File, which is mandatory, contains Image Binder Manager. The file shall be stored in MGR_INFO directory, and its file name shall be “IMB_MGR”. If SD-Image specification is supported, the Image Binder Manager File shall exist.
The Image Binder Manager defines characteristics of all Image Binders in the SD-Image volume.

Figure 5.4-1:Structure of ImbIDITbl in IMB_MGR File

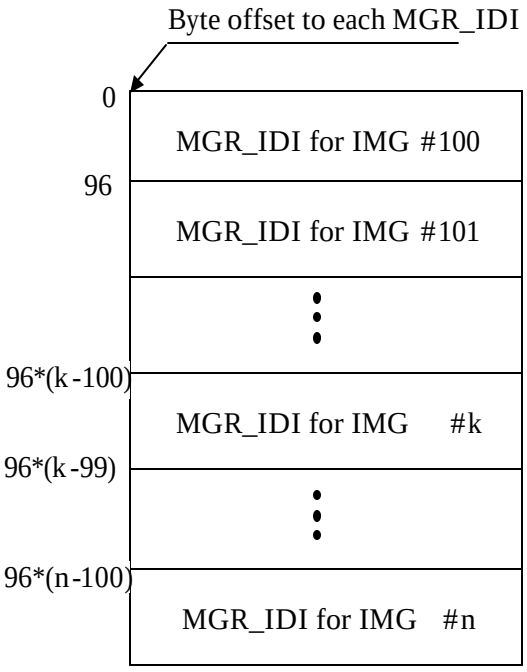


Table 5.4-1: IMB_MGR

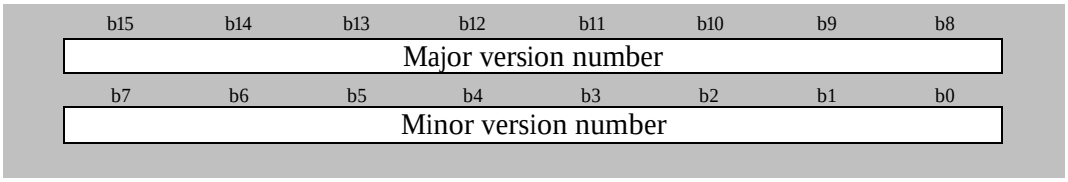
Field Type	Field Name	Contents	FieldSize (Byte)
UNIT16	DataType	Data type identifier	2
UINT16	Version	Version number of IMB_MGR	2
UINT16	MaxImbID	The largest number of the current Image Binder Numbers	2
UINT16	NumImb	Number of Image Binders	2
UINT8	Reserved[4]	Reserved	4
MGR_IDI	ImbIDITbl[MaxImbID]	Table of Identification Information for each Image Binder	96*MaxImbID
Total			12+96*MaxImbID

DataType

This field shall be "I2" with character set code of ISO/IEC 646:1983(ASCII).

Version

Describes a version number of IMB_MGR.



In the first version, the major version number is 1 and the minor version 0.

MaxImbID

Describes the largest number of the Image Binder Numbers currently assigned to Image Binders. If the value of this field ("MaxImbID"-99) is the same as the number of Image Binders and a WRITER creates a new Image Binder, this field shall be updated with ("MaxImbID"+1). If an Image Binder with its Image Binder Number equal to MaxImbID is deleted, this field shall be updated with a new maximum Image Binder Number. If an Image Binder with its Image Binder Number not equal to MaxImbID is deleted, this field shall not be updated. The maximum number of MaxImbID is 999.

NumImb

Describes the number of Image Binders.

ImbIDITbl

Describes the Identification Information of each Image Binder in tabular form. Identification Information (IDI) for an Image Binder shall reside in MGR_IDI (see Table 5.4-2) including a title, creation time, and representation image of the Image Binder. The MGR_IDI for Image Binder *n* (*n* is Image Binder Number; 100-999) shall always be the (*n*-99)-th entry of ImbIDITbl. If there is no information to set to the record in MGR_IDI, CreationTime field of (*n*-99)-th entry shall be filled with 00h. Even if the Image Binder *n* has been deleted, the MGR_IDI entry for the Image Binder *n* shall be preserved in the CreationTime field filled with the value FFh. The preserved MGR_IDI entry shall be reused for a new Image Binder *n*. on regeneration of the Image Binder *n*. MGR_IDI is optional. If MGR_IDI is not used despite the existence of an Image Binder, all of this structure or every unused field shall be filled with 0.

Table 5.4-2: MGR_IDI

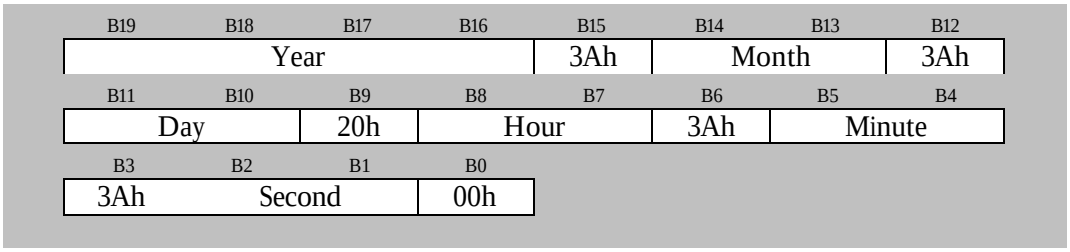
Field Type	Field Name	Contents	Field Size (Byte)
UNIT16	DataType	Data type identifier	2
UINT8	CreationTime[20]	Creation time of IMB or PLST	20
UNIT16	TitleAttribute	Attribute of Title field	2
UINT8	Title1[32]	Title 1 of IMB or PLST	32
UINT8	Title2[32]	Title 2 of IMB or PLST	32
CNTID	RepObj	Representation Object of IMB or PLST	4
UINT8	Reserved[4]	Reserved	4
Total			96

DataType

This field shall be "I3" with character set code of ISO/IEC 646:1983(ASCII).

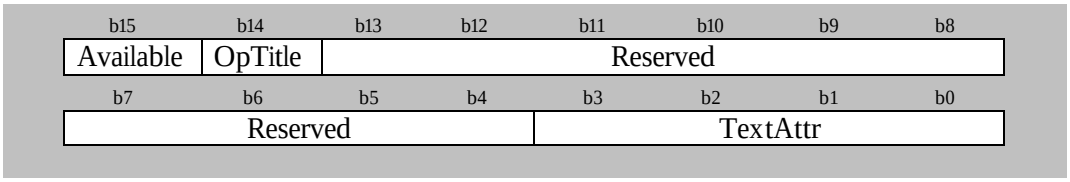
CreationTime

Describes the creation date and time of an Image Binder or Playlist which shall be formatted in ISO/IEC 646 (ASCII) text in the following manner. The text shall be terminated with the code of 00h. The Creation Time is a user definable item, which means the user can set any date and time in this field to identify an Image Binder or Playlist. If the whole field is filled with the value FFh this field shall be invalid, because the corresponding Image Binder or Playlist has been deleted.



Year	The year of CreationTime in 4-digit number from 0001 to 9999.
Month	The month of CreationTime in 2-digit number from 01 to 12.
Day	The day of month of CreationTime in 2-digit number from 01 to 31.
Hour	The hour of CreationTime in 2-digit number from 00 to 23.
Minute	The minute of CreationTime in 2-digit number from 00 to 59.
Second	The second of CreationTime in 2-digit number from 00 to 59.

TitleAttribute



Available	1:Display a Title 0:No Title to be displayed (in neither MGR_IDI nor CMD files) The PLAYER that displays a Title shall refer to this field.
OpTitle	0:Title does not exist in CMD File. 1:Title exists in CMD File. If MGR_IDI is part of IMB_MGR, CMD File shall be IMBCMD. If MGR_IDI is part of PLST_MGR, the value shall always be zero.
Reserved	Reserve bits for the future.
TextAttr	Describes a character code set of the data stored in the Title 2 field. 1h: ISO/IEC 646:1983 (ASCII) 2h: JIS X 0201 3h: ISO 8859-1 9h: Shift-JIS kanji Others: Reserved

Title 1

Describes a title of an Image Binder or Playlist. The title shall be formatted in a text represented by ISO/IEC 646 character code.

Title 2

Describes a title of an Image Binder or Playlist. The title shall be formatted in a text represented by the above character code.

Each field of a Title shall be terminated by the code 00h, unless the length of the text is equal to that of the field. Title is a user definable item, which means the user can set any text to this field to identify a Image Binder and/or Playlist. These Titles shall be shorter than the Title described in IMBCMD.

RepObj

Describes the Content ID of the representation Object of an Image Binder or Playlist.

5.5. Image Binder Information File

Image Binder Information File (IBI file), which is mandatory, contains the Image Binder Information (IMB_INFO) of a certain Image Binder. The file shall be stored in the directory assigned to its Image Binder and the file name shall have a base name in 8 characters, which is the name identical to its Image Binder Directory name plus 3-character extension of "IBI" preceded by a period. For example, the IBI file of Image Binder '112SDIMG' shall be named as "112SDIMG.IBI". The Image Binder Information defines characteristics of an Image Binder. Its specification is shown in Table 5.5-1.

Table 5.5-1: IMB_INFO

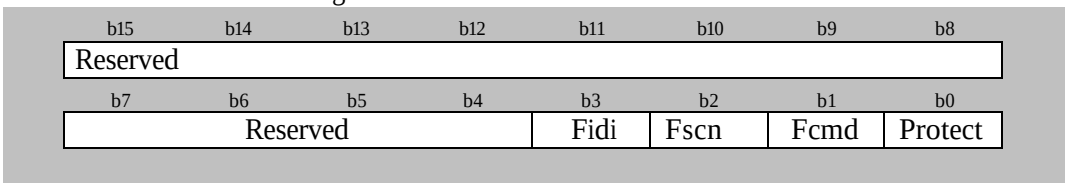
Field Type	Field Name	Contents	Field Size (Byte)
UNIT16	DataType	Data type identifier	2
UINT16 UINT16	Attribute NumRefObj	Attribute of an Image Binder Number of Objects managed by the Image Binder	2 2
UINT16	MaxObjID	The largest number of Objects currently managed by the Image Binder	2
UINT8	Reserved[10]	Reserved	10
REFOBJ	RefObjTbl [NumRefObj]	Table of the Objects referred to by Image Binder	28*NumRefObj
Total			18 +28*NumRefObj

DataType

This field shall be "I4" with character set code of ISO/IEC 646:1983(ASCII).

Attribute

Describes the attribute of an Image Binder.



Fidi	0:OBJ_IDI does not exist in the current Image Binder. 1:OBJ_IDI exists in the current Image Binder.
Fscn	0:Scene Description Data File does not exist in the current Image Binder 1:Scene Description Data File exists in the current Image Binder
Fcmd	0:CMD File of the current Image Binder does not exist 1:CMD File of the current Image Binder exists.
Protect	0:The Image Binder is not write-protected. 1:The Image Binder is write-protected.

NumRefObj

Describes the number of entries of RefObjTbl, which is the number of Objects referred to by the Image Binder.

MaxObjID

Describes the maximum Object Number currently assigned to Image Objects and managed by an Image Binder. If the value of this field "MaxObjID" is the same as the number of Image Objects and a WRITER creates a new Image Object, this field shall be updated with ("MaxObjID"+1). If an Image Object with its Object Number equal to MaxObjID is deleted, this field shall be updated with a new maximum Object Number. If an Image Object with its Object Number not equal to MaxObjID is deleted, this field shall not be updated. The maximum number of MaxObjID is 999.

RefObjTbl

Describes a table of Image Objects referred to by an Image Binder. Each entry of the table shall comply with the data structure of REFOBJ in Table 5.5-2. The table describes the Content ID of the referred Object and the information of the Object.

Table 5.5-2: REFOBJ

Field Type	Field Name	Contents	Field Size (Byte)
UNIT16	DataType	Data type identifier	2
CNTID	ImageObjID	Content ID of the referred Image Object	4
UINT16 UINT16	Attribute ImageWidth	Attribute of the Object Width of the Image Object in pixels	2 2
UINT16	ImageHeight	Height of the Image Object in pixels	2
UINT16	LockCnt	Counter of reference from Playlists	2
UINT16	UserLockCnt	Counter of reference (Other than Playlist)	2
UINT8	Reserved[12]	Reserved	12
Total			28

DataType

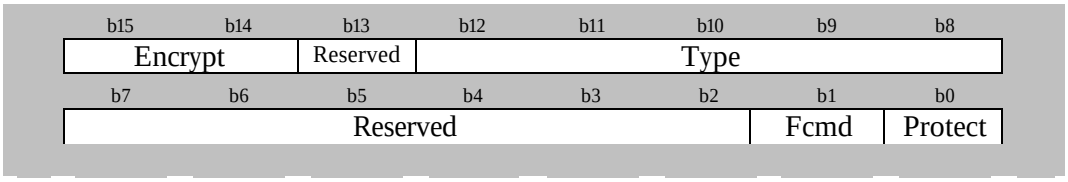
This field shall be "I5" with character set code of ISO/IEC 646:1983(ASCII).

ImageObjID

Describes the Content ID of a referred Image Object. ImageObjID shall not refer to a Thumbnail Image Data File.

Attribute

Describes the attribute of an Object. Encrypt and Type fields shall be the same fields corresponding to the Content ID of the Object. Therefore, b15 to b8 (except b13) of this field shall be equal to b31 to b24 (except b29) of the Content ID of the Object.



Encrypt	0: Not encrypted. 2: Encrypted.
Type	1h: Still Image Data File of GIF 2h: Still Image Data File of JPEG (Exif) 3h: Still Image Data File of JPEG (JFIF) 4h: Still Image Data File of PNG 5h: Still Image Data File of Bitmap 9h: Animation Image Data File of GIF
Fcmd	0: CMD File of the current Image Object does not exist. 1: CMD File of the current Image Object exists.
Protect	0: The Object is not write-protected.

1: The Object is write-protected

ImageWidth
Describes the width of an Image Object in pixels. It shall not be lager than 2560 pixels.

ImageHeight
Describes the height of an Image Object in pixels. It shall not be lager than 1920 pixels.

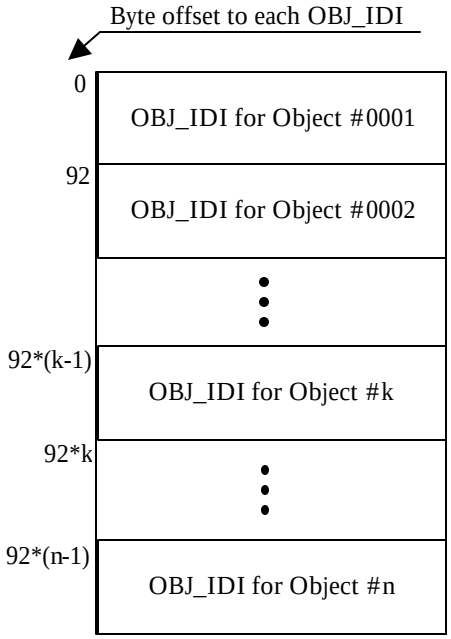
LockCnt
Describes the number of references by Playlists. When a reference to an Object is added to a Playlist, this field shall be incremented by 1. When an object reference is deleted from a Playlist, this field shall be decremented by 1. Therefore, if a certain Playlist is deleted, all the LockCnt fields of Objects related to the Playlist shall be decremented. If a certain Playlist is edited or a new Playlist is generated and is available, all the LockCnt fields of Objects related to the Playlist shall be updated. To delete an object, this field shall be checked. If this field has a value greater than 0, some of the Playlists shall have their reference to an Object. Therefore, the user's confirmation shall be required before the deletion of an object. If a Playlist is not supported or if there is no Playlists created in the SD-Image, the value of this field shall be 0.

UserLockCnt
Describes the number of references by some structure other than Playlists. This can be used to protect Objects in Image Binders referred to by some structure other than Playlists. This field shall be incremented by 1 when a reference by others is added. It shall be decremented by 1 when a reference to an Object is deleted. This field is available by user's intention. If there is no reference by others than Playlists, the value of this field shall be 0. For example, UserLockCnt can be used as a reference counter of an Object from the Scene Description File. If this field has a value greater than 0, the Object is referred to by some of the Scene Descriptions. Therefore, the confirmation or prohibition of deletion of Objects by the user shall be implemented.

5.6. Object Identification Information File

Object Identification Information File is optional and may be stored in the Image Binder Directory. The file name shall be "OBJ_IDI". The file contains only a table of the Identification Information (IDI) of Image Objects that are handled by a Image Binder. Each entry of the table shall be the data structure of OBJ_IDI, which is shown in Table 5.6-1. The structure of this file is illustrated in Figure 5.6-1.

Figure 5.6-1: Structure of Object Identification Information File



As illustrated in Figure 5.6-1, the section where OBJ_IDI for Image Object *k* (*k* is Object Number; 0001-0999) is stored shall start at the position of 92*(*k*-1) bytes from the beginning of the Object Identification Information File, and end before the position 92**k*.

Table 5.6-1: OBJ_IDI

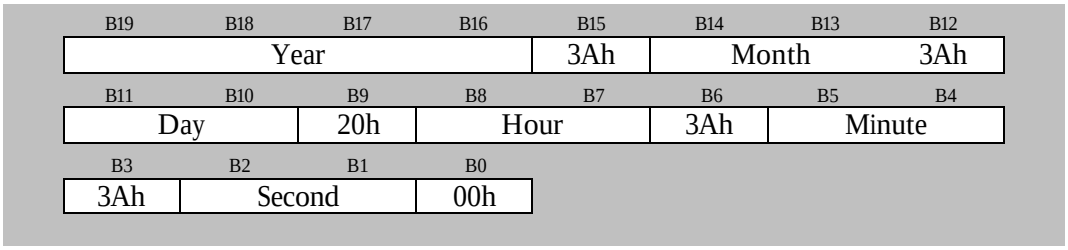
Field Type	Field Name	Contents	Field Size (Byte)
UNIT16	DataType	Data type identifier	2
UINT8	CreationTime[20]	Creation time of the Object	20
UNIT16	TitleAttribute	Attribute of Title field	2
UINT8	Title1[32]	Title1 of the Object	32
UNIT8	Title2[32]	Title2 of the Object	32
UNIT8	Reserved[4]	Reserved[4]	4
Total			92

DataType

This field shall be "I6" with character set code of ISO/IEC 646:1983(ASCII).

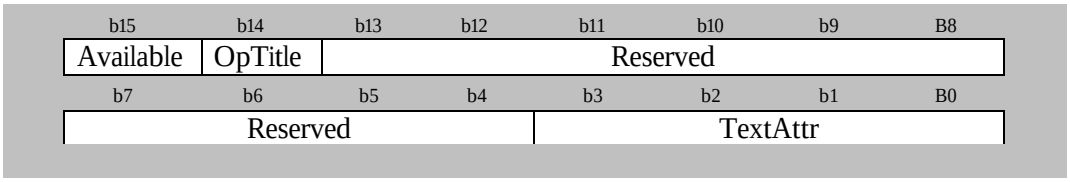
CreationTime

Describes the creation date and time of an Object. The information shall be formatted in ISO/IEC 646 (ASCII) text in the following manner. The text shall be terminated with the code 00h. The Creation Time is a user definable item, which means the user can set any date and time to this field for identification of the Object. When there are some fields filled with the value of FFh, the Object related to these fields does not exist in the Image Binder.



Year	The year of CreationTime in 4-digit number from 0001 to 9999.
Month	The month of CreationTime in 2-digit number from 01 to 12.
Day	The day of the month of CreationTime in 2-digit number from 01 to 31.
Hour	The hour of CreationTime in 2-digit number from 00 to 23.
Minute	The minute of CreationTime in 2-digit number from 00 to 59.
Second	The second of CreationTime in 2-digit number from 00 to 59.

TitleAttribute



Available	1:Display Titles 0:No Title to be displayed (in neither OBJ_IDI nor OBJCMD files) The PLAYER that displays a Title shall refer to this field.
OpTitle	0:Title does not exist in OBJCMD, 1:The Title exists in OBJCMD
Reserved	Reserve bits for future.
TextAttr	Describes the character code set of the data stored in Title2 field. 1h: ISO/IEC 646:1983 (ASCII) 2h: JIS X 0201 3h: ISO 8859-1 9h: Shift-JIS kanji Others: Reserved

Title1

Describes the title of an Object, which shall be formatted in text represented by ISO/IEC 646 character code.

Title2

Describes the title of an Object. The title shall be formatted in a text represented by the character codes described in the above.

Each field of a Title is terminated by the code 00h, unless the length of its text is equal to that of this field. A Title is a user definable item, which means the user can set any text to this field as identification of an Object. These Titles shall be shorter than the Titles described in OBJCMD.

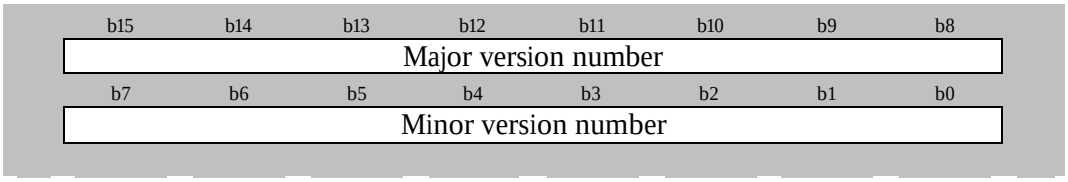
5.7. Playlist Manager File

Playlist Manager File is optional and contains Playlist Manager (PLST_MGR). The file shall be stored in MGR_INFO directory and its name be "PLST_MGR".
If Playlist is supported, PLST_MGR shall exist.
Playlist Manager (PLST_MGR) defines general information of all Playlists in SD-Image. All Playlists are defined as a group of favorite Objects by usr.

Table 5.7-1: PLST_MGR

Field Type	Field Name	Contents	Field Size (Byte)
UNIT16	DataType	Data type identifier	2
UINT16	Version	Version number of PLST_MGR	2
UINT16	MaxPlstID	The maximum number of the current Playlists	2
UINT16	NumPlst	The Number of Playlists	2
UINT8	Reserved[4]	Reserved	4
MGR_IDI	PlstIDITbl[MaxPlstID]	A Table for Identification Information of each Playlist	96*MaxPlstID
Total			12+96*MaxPlstID

DataType
This field shall be "I7" with character set code of ISO/IEC 646:1983(ASCII).
Version
Describes the version number of PLST_MGR.



In the first version, the major version number is 1 and the minor version 0.

MaxPlstID
Describes the maximum number of the Playlists currently assigned in the SD-Image volume. If the value of this field "MaxPlstID" is the same as the number of Playlists and a WRITER creates a new Playlist, this field shall be updated with ("MaxPlstID"+1). If a Playlist with its Playlist Number equal to MaxPlstID is deleted, this field shall be updated with a new maximum Playlist Number. If a Playlist with its Playlist Number not equal to MaxPlstID is deleted, this field shall not be updated. The maximum number of MaxPlstID is 999.

NumPlst
Describes the number of Playlists.

PlstIDITbl

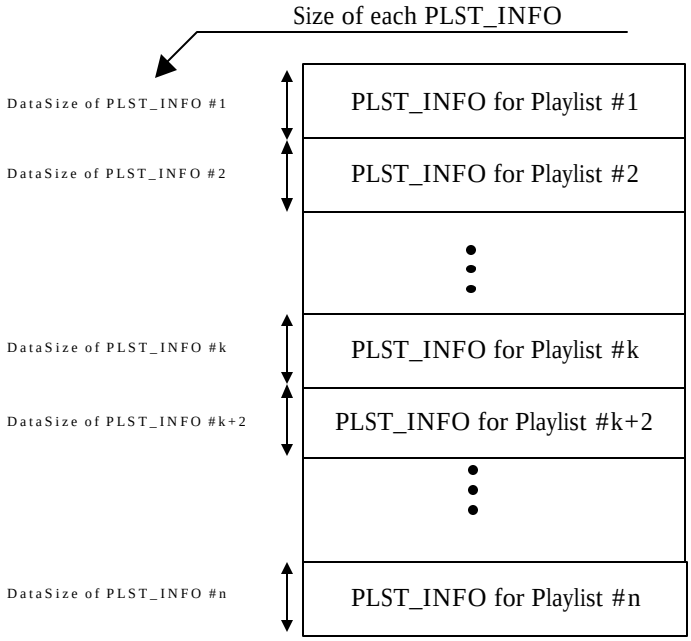
Describes the Identification Information of each Playlist in tabular form. Identification Information (IDI) for Playlists including their title, creation time and representation image shall be managed in MGR_IDI (see Table 5.4-2). The MGR_IDI for Playlist *n* (*n* is Playlist Number; 0001-0999) shall always be in the *n*-th entry of PlstIDITbl. If there is no information to set to the record in MGR_IDI, CreationTime field of *n*-th entry shall be filled with 00h. Even if Playlist *n* has been deleted, the MGR_IDI entry for the Playlist *n* shall be preserved with the CreationTime field filled with the value of FFh. On recreation of the Playlist *n*, the preserved MGR_IDI entry shall be reused for the new Playlist *n*.

ã 2001 SD Association

5.8. Playlist Information File

Playlist Information File is optional and may be created in MGR_INFO Directory. The file name shall be "PLST_PLI". If a WRITER supports Playlist functions, this file shall be created. The file contains a table of Playlist Information for each Playlist. Each entry of the table shall be the data structure of PLST_INFO as shown in Table 5.8-1. If Playlist is supported, PLST_PLI shall exist.

Figure 5.8-1: Structure of Playlist Information File



As illustrated in Figure 5.8-1, the position of where to store the PLST_INFO for Playlist *k* (*k* is Playlist Number; 001-999) shall be predetermined by adding up the DataSize of PLST_INFO. When a Playlist is created, the section of its PLST_INFO is added at the end of the file. When a Playlist is deleted, the section of its PLST_INFO is deleted and the PLST_INFO after the playlist moves up. If the deleted Playlist is the last one, moving of the PLST_INFO does not occur.

Table 5.8-1: PLST_INFO

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	DataType	Data type identifier	2
UINT16	PlstNumber	Number of Playlists	2
UINT16	Attribute	Attribute of the Playlists	2
UINT16	NumRefObj	Number of Objects referred to by the Playlist	2
UINT8	Reserved[8]	Reserved	8
REFOBJ	RefObjTbl [NumRefObj]	Reference table of Objects referred to by the Playlist	28* NumRefObj
Total			16+28*NumRefObj

DataType

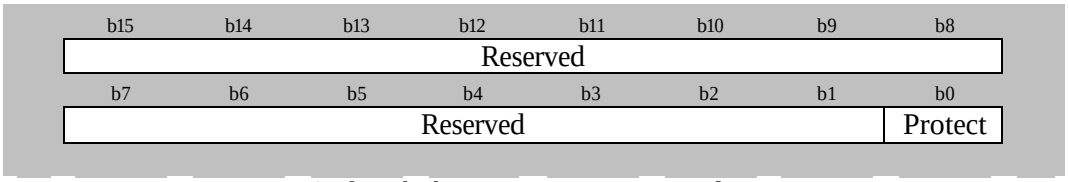
This field shall be "I8" with character set code of ISO/IEC 646:1983(ASCII).

PlstNumber

Describes the number of Playlists. The value shall not be 0 and the maximum value shall be 999.

Attribute

Describes the attribute of a Playlist.



Protect 0:The Playlist is not write-protected.
1:The Playlist is write-protected.

NumRefObj

Describes the number of entries of RefObjTbl. The number shall be equal to that of Objects referred to by a Playlist.

RefObjTbl

Describes the reference information of Objects that are referred by a Playlist in tabular form. Each entry of RefObjTbl shall be REFOBJ as described in Table 5.5-2.

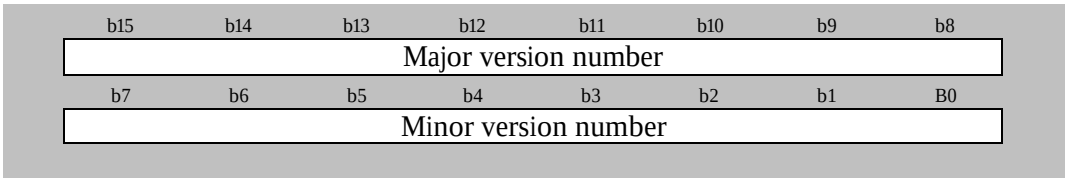
5.9. Content Meta-Data File

Content Meta-Data File (CMD File) is an optional file which contains Content Meta-Data (CMD). The file shall be associated with both Image Binder Information File and Object Data Files. The file name shall be the same as that of the associated file except that its extension is "CMD". For example, a CMD File which is accompanied by the Image Binder Information File "105SDIMG.IBI" shall be named as "105SDIMG.CMD," and a CMD File accompanied by the Image Object Data File "PICT0123.JPG" shall be named as "PICT0123.CMD".
Content Meta-Data defines the following meta data of an Object: title, author, and other content descriptions.

Table 5.9-1: CMD

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	DataType	Data type identifier	2
UINT16	Version	Version number of CMD	2
UNIT16	DataSize	Size of CMD File	2
UINT16 META_DATA	NumMetaData MetaDataTbl [NumMetaData]	Number of meta data for an Object Table of meta data for the Object	2 Variable
Total			Variable

DataType
This field shall be "I9" with character set code of ISO/IEC 646:1983(ASCII).
Version
Describes the version number of a CMD.



In the first version, major version number is 1 and the minor version is 0.

DataSize
Describes the size of Content Meta Data File.
NumMetaData
Describes the number of entries of a MetaDataTbl.
MetaDataTbl
Describes the table of the meta data for each Image Binder or Object. The specification of the meta data structure of META_DATA is described in Table 5.9-2.

Table 5.9-2: META_DATA

Field Type	Field Name	Contents	Field Size (Byte)
UINT16	MetaDataType	Meta data type identifier	2
UINT32	DataSize	Size of meta data	4
UINT8	DataAttribute	Attribute of data	1
UINT8	Data[DataSize-7]	The contents of meta data	DataSize-7
Total			DataSize

MetaDataType

This field indicates a type of meta data. The values of the field are as follows.

If a CMD File is supported, a WRITER may write at least one meta data in an SD memory card. If any of DataTypes from 0001h to 0003h are specified, a WRITER shall write meta data in ascending order of the DataType field value. If any of DataTypes of 0004h or later are specified, a WRITER may write meta data in any order. When a Create Time is provided, a Sub Create Time may be written in the SD memory card.

0001h	Title (It is longer than both of Title1 and Title2 in MGR_IDI.)
0002h	Create Time
0003h	Sub Create Time
0004h	Author
0005h	Copyright
0006h	Description
0007h	Editor
0008h	Producer
0009h	Recording Company
000Ah	Thumbnail Information
000Bh	Rotate
000Ch	Image Size
000Dh	Writer Vender
000Eh	Writer Equipment
Others	Reserved

DataSize

Describes the size of a META_DATA. The DataSize shall be less than 10k bytes.

DataAttribute

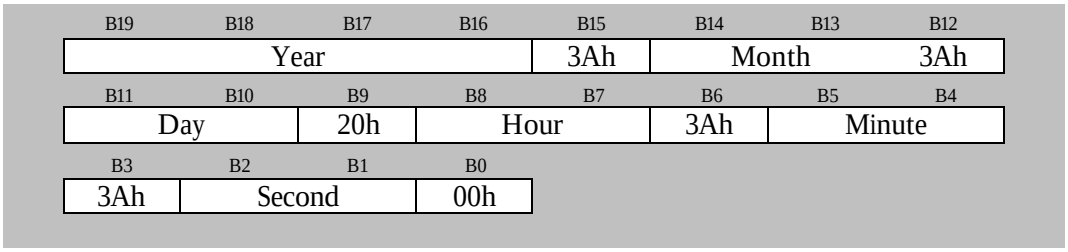
Describes a character code set of the data stored in a Data field. The value shall be one of the followings.

1h	ISO/IEC 646:1983 (ASCII)
2h	JIS X 0201
3h	ISO 8859-1
9h	Shift-JIS kanji
Others	Reserved

Data

Describes meta data itself. If a DataType field has the value of 0001h, 0004h, 0005h, 0006h, 0007h, 0008h, 0009h, 000Dh or 000Eh, this field shall be in text-formatted in the character code set as specified in the DataAttribute field, and the file shall be terminated with the code of 00h. If another type is specified, this field shall be formatted as follows.

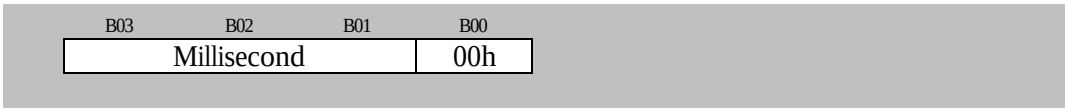
- DataType=0002h: 20-byte long data formatted as follows.



Year	The year of Create Time in 4-digit number from 0001 to 9999.
Month	The month of Create Time in 2-digit number from 01 to 12.
Day	The day of the month of Create Time in 2-digit number from 01 to 31.
Hour	The hour of Create Time in 2-digit number from 00 to 23.
Minute	The minute of Create Time in 2-digit number from 00 to 59.
Second	The second of Create Time in 2-digit number from 00 to 59.

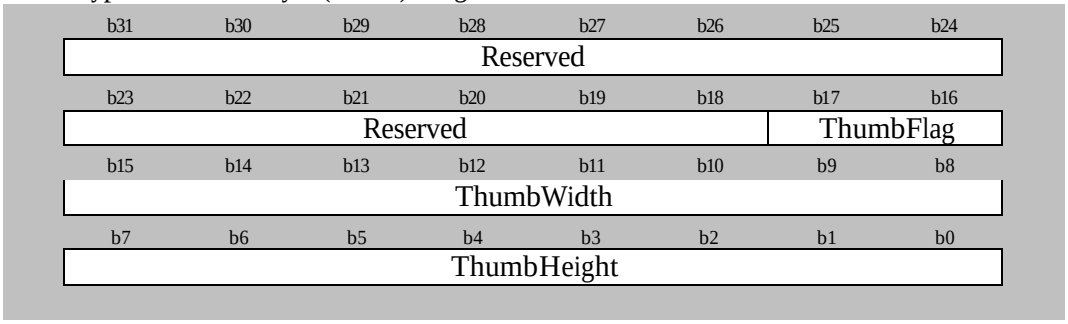
All the fields in the above shall be formatted with ISO/IEC 646:1983.

- DataType=0003h: 4-byte long data formatted as follows



Millisecond: Millisecond of Create Time in 3-digit number from 000 to 999.
This field shall be formatted with ISO/IEC 646:1983.

- DataType=000Ah: 4-byte (32-bit) long data formatted as follows

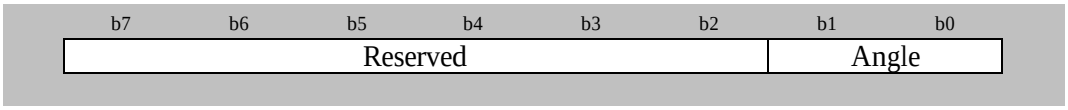


ThumbFlag: indicates the existence of a thumbnail image for an Object. The value shall be one of the followings.

00 No thumbnail image exists
01 Thumbnail image exists and is stored within the Object Data File of the Object.
02 Thumbnail image exists and is stored in a file with the prefix of "THMN".
03 Reserved

ThumbWidth: Width of the thumbnail image.
ThumbHeight: Height of the thumbnail image.

- DataType=000Bh: 1-byte long data formatted as follows



Angle: indicates a rotation angle of an Object. A PLAYER should rotate the Object by the angle in the clockwise direction before the Object is presented. The value shall be one of the followings.

00 0 degree, that is, no rotation is needed on presentation.
01 90 degrees.
02 180 degrees.
03 270 degrees.

- DataType=000Ch: 4-byte long data formatted as follows.

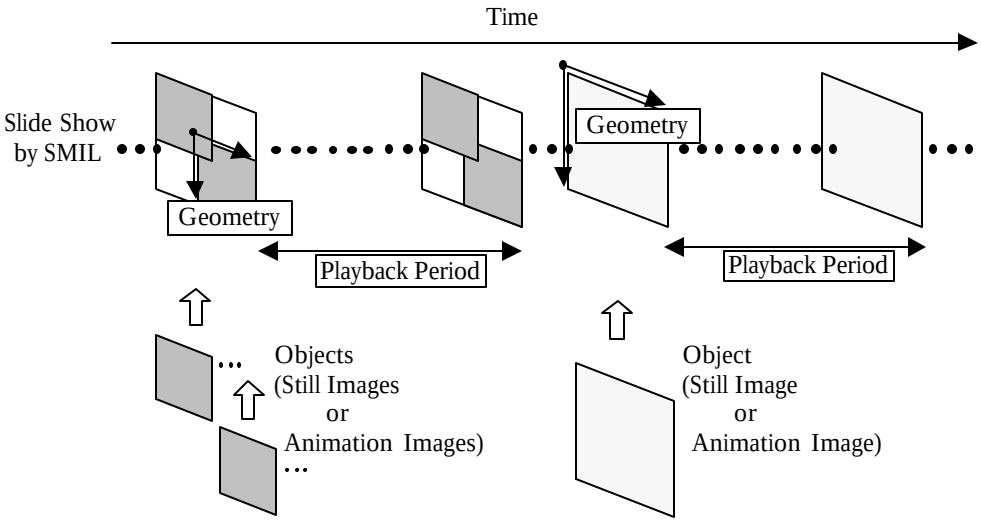
ImageWidth: Width of the Object.
ImageHeight: Height of the Object.

6. Scene Description

Scene Description tells a PLAYER how to display plural Image Objects in the same screen. It can describe the geometry (position and size) and rendering period of each Image Object. Scene Description is composed of multiple Scene Description Data Files.

For Scene Description data, this specification uses SMIL (Synchronized Multimedia Integration Language) 2.0 Basic Profile which is currently work in progress.
Refer to http://www.w3.org/TR/smil20.

Figure 6-1: Scene Description



7. Annexes

Annex A. Mandatory Items in File of SD-Image Specification

In the following descriptions, items described with "shall" are mandatory. Items described with "may" are optional.

Annex A.1. MGR_DATA

This file is mandatory. Whether the fields contained in this file are mandatory or not is shown in the table below.

Table Annex A-1: MGR_DATA

Field	in Card	by Player	Value if not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency
Version	Mandatory	shall be used	-	Player shall check whether data in Card can be understood or not.
ResumeMarker	Optional	may be used	Zero	This field is structure of **CNTID**.
TextAttribute	Optional	may be used	Zero	
Volume Label	Optional	may be used	All Zero	

Annex A.2. IMB_MGR

This file is mandatory. Whether the fields contained in this file are mandatory or not is shown in the table below.

Table Annex A-2: IMB_MGR

Field	in Card	by Player	Value if not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency
Version	Mandatory	shall be used	-	Player shall check whether Image Binder data in Card can be understood or not.
MaxImbID	Mandatory	may be used	-	
NumImbID	Mandatory	may be used	-	
ImbIDITbl	Mandatory	shall be used	-	Player shall check for consistency. This field is structure of **MGR_IDI**.

Annex A.3. IBI (Image Binder Information)

This file is mandatory. Whether the fields contained in this file are mandatory or not is shown in the table below.

Table Annex A- 3: IBI

Field	in Card	by Player	Value if not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency
Attribute	Mandatory	shall be used	-	
NumRefObj	Mandatory	may be used	-	
MaxObjID	Mandatory	may be used	-	
RefObjTbl	Mandatory	shall be used	-	This filed is structure of **REFOBJ**.

Annex A.4. PLST_MGR

This file is optional. WRITERs that do not support Playlists are not required to create or update this file, and PLAYERs that do not support Playlists are not required to read this file. However, recorders that create or update this file and PLAYERs that read this file shall write or read it according to the following rule.

Table Annex A- 4: PLST_MGR

Field	in Card	by Player	Value if not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency
Version	Mandatory	shall be used	-	Player shall check whether Playlist data in Card can be understood or not.
MaxPlstID	Mandatory	may be used	-	
NumPlst	Mandatory	may be used	-	
PlstIDITbl	Mandatory	shall be used	-	This filed is structure of **MGR_IDI**.

If PLST_MGR is supported, all fields are mandatory “in Card”.

Annex A.5. PLST_PLI (Playlist Information)

This file is optional. WRITERs that do not support Playlists are not required to create or update this file, and PLAYERs that do not support Playlists are not required to read this file. However, recorders that create or update this file and PLAYERs that read this file shall write or read it according to the following rule.

Table Annex A- 5: PLST_PLI

Field	in Card	by Player	Value if not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency
PlstNumber	Mandatory	shall be used	-	
Attribute	Mandatory	shall be used	-	
NumRefObj	Mandatory	may be used	-	
RefObjTbl	Mandatory	shall be used	-	This field is structure of **REFOBJ**.

Annex A.6. OBJ_IDI

This file is optional, and it provides the Object Information of a Referenced Image Object.

Table Annex A- 6: OBJ_IDI for Object

Field		in Card	by Player	Value if not supported	Note
DataType		Optional	may be used	Zero	Player shall check for consistency
CreationTime		Optional	may be used	Zero	
TitleAttr		Optional	may be used	Zero	
	Available	Optional	may be used	Zero	
	OptionTitle	Optional	may be used	Zero	
	TextAttr	Optional	may be used	Zero	
Title1		Optional	may be used	All Zero	
Title2		Optional	may be used	All Zero	

Annex A.7. CMD (Content Meta-Data)

This file is optional. WRITERs that do not support Meta-Data are not required to create or update this file, and PLAYERs that do not support Meta-Data are not required to read this file. However, recorders that create or update this file and PLAYERs that read this file shall write or read it according to the following rule.

Table Annex A- 7: CMD

Field	in Card	by Player	Value if not supported	Note
DataType	Mandatory	shall be used	-	Player shall check for consistency
Version	Mandatory	shall be used	-	Player shall check whether Content Meta-Data can be understood or not.
DataSize	Mandatory	shall be used	-	
NumMetaData	Mandatory	shall be used	-	
MetaDataTbl	Mandatory	shall be used	-	
MetaDataType	Mandatory	shall be used	-	
DataSize	Mandatory	shall be used	-	
Attribute	Mandatory	shall be used	-	
Data	Mandatory	shall be used	-	

If CMD is supported, all fields are mandatory.

Annex B. Mandatory Items in Data Structure of SD-Image Specification

Annex B.1. CNTID

This is the structure providing the Content ID of an Object.

Table Annex B- 1: CNTID

Field	in Card	By Player	Value if not supported	Note
Encrypt	Mandatory	Shall be used	-	Player shall check for consistency.
Attribute	Mandatory	Shall be used	-	Player shall check for consistency.
MajorNumber	Mandatory	Shall be used	-	Player shall check for consistency.
MinorNumber	Mandatory	Shall be used	-	Player shall check for consistency.

Annex B.2. REFOBJ

This is the structure providing the Content ID of a Referenced Object.

Table Annex B- 2: REFOBJ

Field	in Card	By Player	Value if not supported	Note
DataType	Mandatory	Shall be used	-	Player shall check for consistency
ImageObjID	Mandatory	shall be used	-	Player shall check for consistency. This field is structure of **CNTID**.
Attribute	Mandatory	may be used	-	
ImageWidth	Mandatory	may be used	-	
ImageHeight	Mandatory	may be used	-	
Lockcnt	Mandatory	may be used	-	Player which can edit Playlist shall write these fields.
UserLockcnt	Mandatory	may be used	-	

Annex B.3. MGR_IDI (Identification Information for Manager)

The structure of MGR_IDI for IMB_MGR or PLST_MGR is mandatory. Whether the fields contained in this file are mandatory or not is shown in the table below.

Table Annex B- 3: IDI for Each Manager File

Field	in Card	by Player	Value if not supported	Note
DataType	Mandatory	shall be used	-	
CreationTime	Optional	may be used	Zero	
TitleAttr	Optional	may be used	Zero	
Available	Optional	may be used	Zero	
OpTitle	Optional	may be used	Zero	
TextAttr	Optional	may be used	Zero	
Title 1	Optional	may be used	All Zero	
Title2	Optional	may be used	All Zero	
RepObj	Optional	may be used	Zero	This field is structure of **CNTID** if it is supported.

SDIO Card Specification

Version 1.0 (Draft 4)

May , 2001

This document is controlled under NDA by the SD Association. It is to be distributed only to members of the SDIO Working Group.

Revision History

Date	Version	Changes compared to previous issue
May 8, 2001	1.0	Base version (draft 4)

Copyright © 2000,2001 SD Association

Conditions for publication

Publisher and Copyright Holder:
SD Association

Exemption:
None will be liable for any damages from use of this document.

Table of Contents

1 General Description....1
1.1 SDIO features....1
1.2 Primary Reference Document....1
2 SDIO Signaling Definition....1
2.1 SDIO Card Types....1
2.2 SDIO Card modes....2
2.2.1 SPI (Card mandatory support)....2
2.2.2 1-bit SD data transfer mode (Card mandatory support)....2
2.2.3 4-bit SD data transfer mode (mandatory for High-Speed cards, optional for Low-Speed)....2
2.3 SDIO Host Modes....2
2.4 Signal Pins....2
2.5 Host Requirements for SDIO....3
3 SDIO Card Initialization....4
3.1 Differences in I/O card initialization....4
3.2 The IO_SEND_OP_COND Command (CMD5)....8
3.3 The IO_SEND_OP_COND Response (R4)....9
4 Differences with SD Memory Specification....9
4.1 SDIO Command List....9
4.2 Unsupported SD Memory commands....10
4.3 Modified R6 Response....10
4.4 Bus Width....11
4.5 Card Detect Resistor....11
4.6 Data Transfer Abort....11
4.6.1 Read Abort....11
4.6.2 Write Abort....12
4.7 Changes to SD Memory Fixed Registers....13
4.7.1 OCR Register....13
4.7.2 CID Register....13
4.7.3 CSD Register....13
4.7.4 RCA Register....13
4.7.5 DSR Register....13
4.7.6 SCR Register....13
4.7.7 SD Status....14
4.7.8 Card Status Register....14
5 New I/O Read/Write Commands....15
5.1 IO_RW_DIRECT command (CMD52)....15
5.2 IO_RW_DIRECT Response (R5)....16
5.2.1 CMD52 Response (SD modes)....16
5.2.2 R5, IO_RW_DIRECT Response (SPI mode)....17
5.3 IO_RW_EXTENDED command (CMD53)....17
5.3.1 CMD53 Data Transfer Format....19
5.3.2 Special Timing for CMD53 multi-block read....19
6 SDIO Card Internal Operation....20
6.1 Overview....20
6.2 Register Access Time....20
6.3 Interrupts....20
6.4 Suspend/Resume....20
6.5 Read Wait....21
6.6 SDIO Fixed Internal Map....21
6.7 Common I/O Area (CIA)....22
6.8 Card Common Control Registers (CCCR)....22
6.9 Function Basic Registers (FBR)....26

6.10 Card Information Structure (CIS) 28
6.11 Multiple Function SD Cards 28
7 Embedded I/O Code Storage Area (CSA) 29
7.1 CSA Access 29
7.2 CSA Data Format 29
8 SDIO Interrupts 30
8.1 Interrupt Timing 30
8.1.1 SPI and SD 1-bit mode interrupts 30
8.1.2 SD 4-bit mode 30
8.1.3 Interrupts with Single Data Transaction 30
8.1.4 Interrupts with 4-bit Multiple Block Data Transactions (Optional) 32
8.1.5 Inhibited Interrupts 33
8.1.6 End of Interrupt Cycles 34
8.1.7 Terminated Data Transfer Interrupt Cycle 35
8.1.8 Interrupt Clear Timing 36
9 SDIO Suspend/Resume Operation 37
10 SDIO Read Wait Operation 39
11 SDIO Physical Properties 41
11.1 SDIO Size 41
11.2 SDIO Card Package 41
12 SDIO Mechanical Extensions 41
12.1 Additional ESD/EMI Ground Point 42
12.2 Extended Case 42
12.3 SDIO Power 42
12.3.1 Voltage 42
12.3.2 Current 42
13 CIS Formats 44
13.1 CIS Reference Document 44
13.2 Basic Tuple Format and Tuple Chain Structure 44
13.3 Byte Order Within Tuples 44
13.4 SDIO Card Metaformat 45
13.5 CISTPL_MANFID: Manufacturer Identification String Tuple 45
13.6 SDIO Specific extensions 45
13.6.1 CISTPL_FUNCID: Function Identification Tuple 46
13.6.2 CISTPL_FUNCE: Function Extension Tuple 46
13.6.3 TPLFID_FUNCTION tuple for function 0 (common) 46
13.6.4 TPLFID_FUNCTION tuple for function 1-7 47
14 Abbreviations and terms 50
A1 SD and SPI Command List ....A
B1 Normative References C
C1 Example SDIO Controller Design ....D

Table of Tables

Table 1 SDIO pin definitions 3
Table 2 OCR values for CMD5 8
Table 3 Unsupported SD Memory Commands 10
Table 4 R6 response to CMD3 11
Table 5 SDIO R6 Status Bits 11
Table 6 SDIO Status Register Structure 15
Table 7 Flag data for IO_RW_DIRECT SD Response 17
Table 8 IO_RW_ EXTENDED command Op Code definition 18
Table 9 Card Common Control Registers (CCCR) 23
Table 10 CCCR bit definitions 26
Table 11 Function Basic Information Registers (FBR) 27
Table 12 FBR bit definitions 28
Table 13 Card Information Structure (CIS) and reserved area of CIA 28
Table 14 Timing Diagram Symbols 36
Table 15 SDIO exceptions to SD physical section 8.1 requirements 41
Table 16 Basic Tuple Format 44
Table 17 Tuples Supported by SDIO Cards 45
Table 18 CISTPL_MANFID: Manufacturer Identification Tuple 45
Table 19 CISTPL_FUNCID tuple 46
Table 20 CISTPL_FUNCE tuple general structure 46
Table 21 TPLFID_FUNCTION tuple for function 0 (common) 46
Table 22 TPLFID_FUNCTION field descriptions for function 0 (common) 47
Table 23 TPLFID_FUNCTION tuple for function 1-7 47
Table 24 TPLFID_FUNCTION field descriptions for functions 1-7 48
Table 25 TPLFE_FUNCTION_INFO definition 49
Table 26 TPLFE_CSA_PROPERTY definition 49
Table 27 SD Mode Command List ..... A
Table 28 SPI Mode Command List ..... B

Table of Figures

Figure 1 Signal connection to two 4-bit SDIO cards....2
Figure 2 SDIO response to non-I/O aware initialization....4
Figure 3 Card initialization flow in SD mode (SDIO aware host)....6
Figure 4 Card initialization flow in SPI mode (SDIO aware host)....7
Figure 5 IO_SEND_OP_COND Command (CMD5)....8
Figure 6 Response R4 in SD mode....9
Figure 7 Response R4 in SPI mode....9
Figure 8 I/O Abort during Read Data Transfer....12
Figure 9 I/O Abort during Write CRC Response (Good)....12
Figure 10 I/O Abort during Write CRC Response (Bad)....12
Figure 11 I/O Abort after Write CRC Response....12
Figure 12 IO_RW_DIRECT Command....15
Figure 13 R5 IO_RW_DIRECT Response (SD modes)....16
Figure 14 IO_RW_DIRECT Response in SPI Mode....17
Figure 15 IO_RW_EXTENDED Command....18
Figure 16 Multi-Block Read Timing....19
Figure 17 SDIO Internal Map....22
Figure 18 Read Interrupt Cycle Timing....31
Figure 19 Alternate Read Interrupt Cycle Timing....31
Figure 20 Write Interrupt Cycle Timing....31
Figure 21 Continuous Interrupt Cycle....32
Figure 22 Multiple Block 4-Bit Read Interrupt Cycle Timing....33
Figure 23 Multiple Block 4-Bit Write Interrupt Cycle Timing....33
Figure 24 Inhibited Interrupts Timing....34
Figure 25 Interrupt Cycle Timing....34
Figure 26 Alternate Interrupt Cycle Timing....35
Figure 27 Terminated Read Multiple Interrupt timing (Case 1)....35
Figure 28 Terminated Read Multiple Interrupt timing (Case 2)....36
Figure 29 Device with long read latency will accept bus suspend....37
Figure 30 Function2 read cycle is inserted during Function1 multiple read cycle....37
Figure 31 Write suspended during busy (case 1)....38
Figure 32 Write suspended during busy (case 2)....38
Figure 33 Relationship between interrupt period and suspend/resume (case 1)....38
Figure 34 Relationship between Interrupt Period and Suspend/Resume (case 2)....39
Figure 35 Suspend/Resume timing....39
Figure 36 Read wait control by stopping SDCLK....40
Figure 37 Read wait delay using Dat[2]....40
Figure 38 SDIO Mechanical Extensions....43
Figure 39 SDIO Internal State Machine example....D
Figure 40 State Diagram for Bus State Machine....D
Figure 41 State Diagram for Function State Machine....E

1 General Description

The SDIO (Secure Digital I/O) card is based on and compatible with the SD memory card. This compatibility includes mechanical, electrical, power, signaling and software. The intent of the SDIO card is to provide high-speed data I/O with low power consumption for mobile electronic devices. A primary goal is that an SDIO card inserted into a non-SDIO aware host will cause no physical damage or disruption of that device or it's software. In this case, the SDIO card should simply be ignored. Once inserted into an SDIO aware host, the detection of the card will be via the normal means described in the SD specification with some extensions. In this state, the SDIO card will be idle and draw a small amount of power (15 mA averaged over 1 second). During the normal initialization and interrogation of the card by the host, the card will identify itself as an SDIO device. The host software will then obtain the card information in a tuple (linked list) format and determine if that card's I/O function(s) are acceptable to activate. This decision will be based on such parameters as power requirements or the availability of appropriate software drivers. If the card is acceptable, it will be allowed to power up fully and start the I/O function(s) built into it.

1.1 SDIO features

- Targeted for portable and stationary applications
- Minimal or no modification to SD Physical bus is required
- Minimal change to memory driver software
- Extended physical form factor available for specialized applications
- Plug and play (PnP) support
- All SD bus modes supported including SPI, 1 and 4 bit SD
- Multi-function support including multiple I/O and combined I/O and memory
- Up to 7 I/O functions plus one memory supported on one card.
- Allows card to interrupt host
- Initialization Voltage: 2.0 to 3.6V
- Operational Voltage range: 3.1 to 3.5V

1.2 Primary Reference Document

This spec is based on and refers extensively to the SDA document:

SD Memory Card Specifications
Part 1
PHYSICAL LAYER SPECIFICATION
September 2000
Version 1.01

The reader is directed to this document for more information on the basic operation of SD devices. In addition, other documents are referenced in this specification. A complete list can be found in section B1.

2 SDIO Signaling Definition

2.1 SDIO Card Types

This specification defines two types of SDIO cards. The Full-Speed card supports SPI, 1-bit SD and the 4-bit SD transfer modes at the full clock range of 0-25MHz. The Full-Speed SDIO devices have a data transfer rate of over 100 Mb/second (10 MB/Sec). A second version of the SDIO card is the Low-Speed SDIO card. This card requires only the SPI and 1-bit SD transfer modes. 4-bit support is optional. In addition, Low-Speed SDIO

cards shall support a full clock range of 0-400 KHz. The intended use of Low-Speed cards is to support low-speed IO devices with a minimum of hardware. The Low-Speed cards support such functions as modems, bar-code scanners, GPS receivers etc. If a card is a 'Combo card' (memory plus SDIO) then Full-Speed and 4-bit operation is mandatory for both the memory and SDIO portions of the card.

2.2 SDIO Card modes

There are 3 signaling modes defined for SD physical specification version 1.01 memory cards that also apply to SDIO Card:

2.2.1 SPI (Card mandatory support)

The SPI bus topology is defined in section 3.1.2 and the protocol is defined in sections 3.2.2 and 7 of the SD Memory Card Specifications, PHYSICAL LAYER SPECIFICATION, Part 1, September 2000 Version 1.01. In this mode pin 8, which is undefined for memory, is used as the interrupt pin. All other pins and signaling protocols are identical to the SD Memory specification.

2.2.2 1-bit SD data transfer mode (Card mandatory support)

This mode is identical to the 1 data bit (narrow) mode defined for SD Memory in section 3.2.1 of the SD Memory Card specification. In this mode, data is transferred on the DAT0 pin only. In this mode pin 8, which is undefined for memory, is used as the interrupt pin. All other pins and signaling protocols are identical to the SD Memory specification.

2.2.3 4-bit SD data transfer mode (mandatory for High-Speed cards, optional for Low-Speed)

This mode is identical to the 4 data bit mode (wide) defined for SD Memory in section 3.2.1 of the SD Memory Card specification. In this mode, data is transferred on all 4 data pins (DAT0-3). In this mode the interrupt pin is not available for exclusive use as it is utilized as a data transfer line. Thus, if the interrupt function is required, a special timing is required to provide interrupts. See section 8.1.2 for details of this operation. The 4-bit SD mode provides the highest data transfer possible, up to 100 Mb/sec.

2.3 SDIO Host Modes

If a SDIO aware host supports the SD transfer mode, it is recommended that both the 1-bit and 4-bit modes be supported. While a SDIO host that supports *only* the 4-bit transfer mode is possible, it's performance with a Low-Speed SDIO card would be reduced. This is because the only means to transfer data to and from a Low-Speed card would be the single byte per command transfer (using the IO_RW_DIRECT command (CMD52) see 5.1).

2.4 Signal Pins

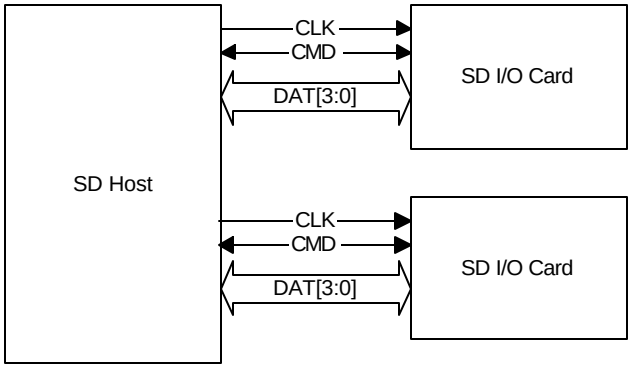


Figure 1 Signal connection to two 4-bit SDIO cards

Pin	SD 4-bit mode		SD 1-bit mode		SPI mode	
1	CD/DAT[3]	Data line 3	N/C	Not Used	CS	Chip select
2	CMD	Command line	CMD	Command line	DI	Data input
3	VSS1	Ground	VSS1	Ground	VSS1	Ground
4	VDD	Supply voltage	VDD	Supply voltage	VDD	Supply voltage
5	CLK	Clock	CLK	Clock	SCLK	Clock
6	VSS2	Ground	VSS2	Ground	VSS2	Ground
7	DAT[0]	Data line 0	DATA	Data line	DO	Data output
8	DAT[1]	Data line 1 or Interrupt (optional)	IRQ	Interrupt	IRQ	Interrupt
9	DAT[2]	Data line 2 or Read Wait (optional)	RW	Read Wait (optional)	NC	Not Used

Table 1 SDIO pin definitions

It is recommended that multi-slot hosts intending to support SDIO (SDIO aware) provide a separate CLK to each slot, to allow the I/O devices to be placed in a low power state on a slot-by-slot basis. After reset, all DAT lines should be in the hi-Z state on both the host and card(s) to avoid bus conflict. Access to the Bus Interface Control register within the CCCR (Table 9) determines DAT line mode.

2.5 Host Requirements for SDIO

In order for a host to completely support all of the capabilities of the SDIO cards, some signal connections must be supported. If interrupts are to be supported, Pin 8 needs to be connected from the card to the host to provide interrupt signaling. This is true even if the host will only support the SPI or 1 bit SD mode. In addition, if the host supports more than 1 card in either SD mode, the CMD and all 4 DAT lines should not be bussed together, but rather routed separately to the host. This will allow the mixing of card types in the different sockets without interference. Both the SD Memory specification and the SDIO specification support the concept of "unifying" (connecting together) the CMD lines in a multi-slot system after initialization.
In addition, there some additional design details that the designer of a host intending to support SDIO devices must be aware of.
If a host supports both the 4-bit SD bus and interrupts during 4-bit transfer the host must control the value placed on the DAT[1-3] lines. Those conditions are:

1) During a multiple block write:

   According to the SD Memory Card Specifications "PHYSICAL LAYER SPECIFICATION Version 1.01" figures 9 and 28, the DAT[1-3] lines are described as "don't care" (X) during the CRC status period. If a host actively drives these lines during this period, it will interfere with interrupt signaling from an SDIO card. In order to prevent this conflict, if a host supports interrupts during 4-bit data transfers, it shall not drive DAT[1-3] during this period (hi-Z rather than don't care.)

2) During a multiple block read:

   According to the SD Memory Card Specifications "PHYSICAL LAYER SPECIFICATION Version 1.01" figure 26, the DAT[1] line is described as "P" (one-cycle pull-up) between read data packets. In order to support interrupts during the 4-bit mode, the host shall not drive the DAT[1] line during the 2 clock Interrupt Period defined in section 8.1.2

3 SDIO Card Initialization

3.1 Differences in I/O card initialization

A requirement for the SDIO specification is that an SDIO card must not cause non-I/O aware hosts to fail when inserted. In order to prevent operation of I/O Functions in non-I/O aware hosts, a change to the SD card identification mode flowchart is needed. A new command (IO_SEND_OP_COND, CMD5) is added to replace the ACMD41 for SDIO initialization by I/O aware hosts (see 3.2).

After reset or power-up, all I/O functions on the card are disabled and the I/O portion of the card will not respond to any operation except CMD5. If there is SD memory installed on the card (also called a combo card), that memory will respond normally to all normal mandatory memory commands.

An I/O only card will not respond to the ACMD41 and thus appear initially as an MMC card (See B1 for information on the MMC specification). The I/O only card will also not respond to the CMD1 used to initialize the MMC cards and appear as a non-responsive card. The host will then give up and disable this device Thus, the non-aware host will receive no response from an I/O only card and force it to the inactive state. The operation of an I/O card with a non-I/O aware host is shown in Figure 2. Note that the solid lines are the actual paths taken while the dashed lines are not executed.

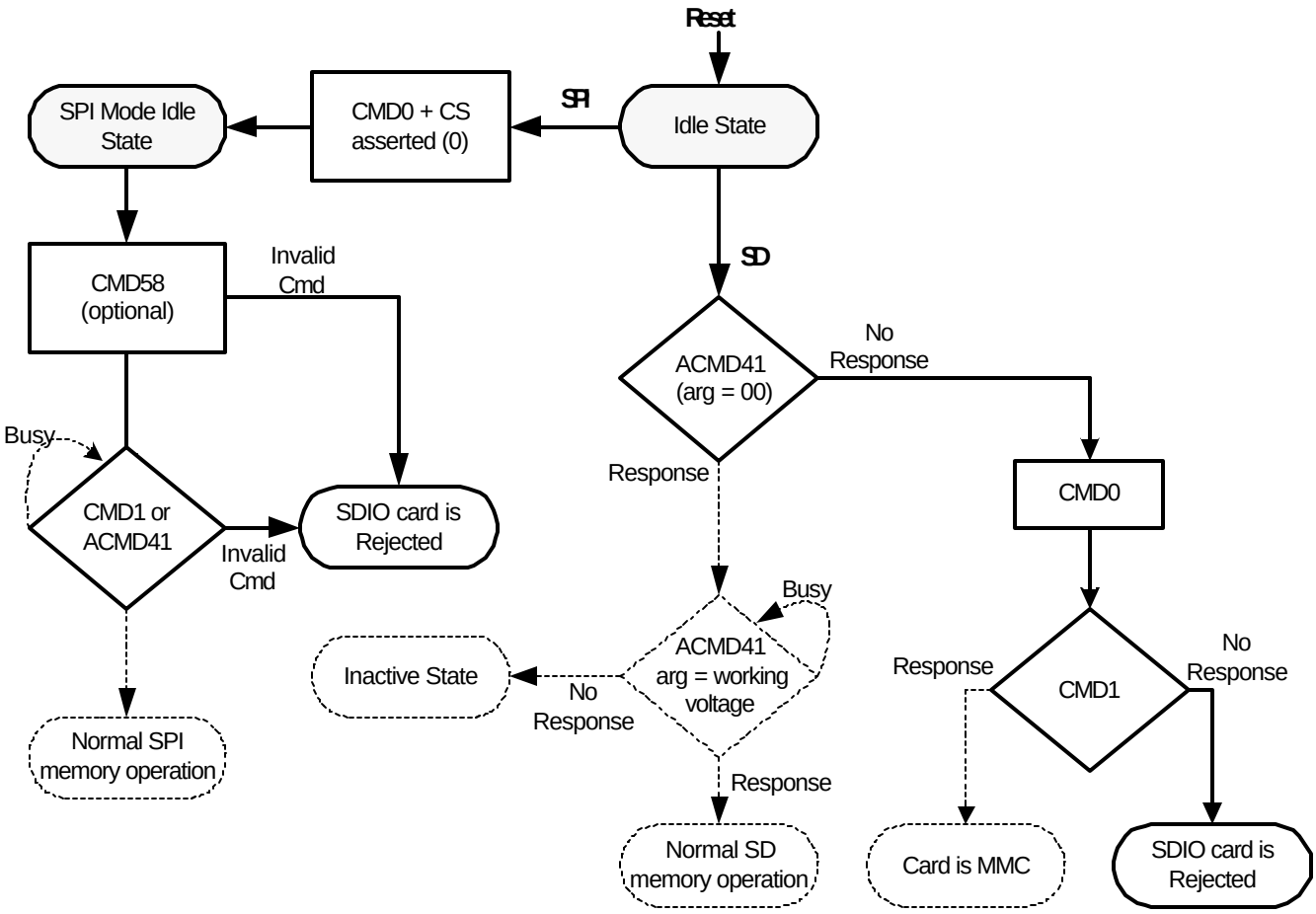


Figure 2 SDIO response to non-I/O aware initialization

An SDIO aware host will send CMD5 prior to the CMD55/ACMD41 pair, and thus would receive a valid OCR in the R4 response to CMD5 and continue to initialize the card. Figure 3 shows the operation of an SDIO aware host operating in the SD modes and Figure 4 shows the same operation for a host that operates in the SPI mode.

If the I/O portion of a card has received no CMD5, the I/O section remains inactive and will not respond to any command except CMD5. A combo card stays in the memory-only mode. If no memory is installed on the card (i.e. an I/O only card in a non-SDIO aware host) the card would not respond to any memory command. This satisfies the condition where a user uses some I/O function on the card such as Ethernet to load a music file to the memory function of that card. The card is then removed and inserted into a non-SDIO aware host. That device would not enable the I/O function (no CMD5) so would appear to the player as a memory-only card. If the host were I/O aware, it would send the CMD5 to the card and the card would respond with R4. The host reads that R4 value and knows the number of available I/O functions and about the existence of any SD memory.

After the host has initialized the I/O portion of the card, it then reads the Common Information Area (CIA) of the card (see 6.7). This is done by issuing a read command, starting at byte 00 of I/O function 0. The CIA contains the Card Common Control Registers (CCCR) and the Function Basic Registers (FBR). Also included in the CIA are pointers to the card' s global Card Information Structure (CIS) and each individual function' s CIS. The CIS structure is defined in section 0. The CIS includes information on power, function, manufacturer and other things the host needs to determine if the I/O function(s) is appropriate to power-up. If the host determines that the card should be activated, a register in the CCCR area enables the card and each individual function. At this time, all functions of the I/O card are fully available. In addition, the host can control the power consumption and enable/disable interrupts on a function-by-function basis. This access to I/O will not interfere with memory access to the card if present.

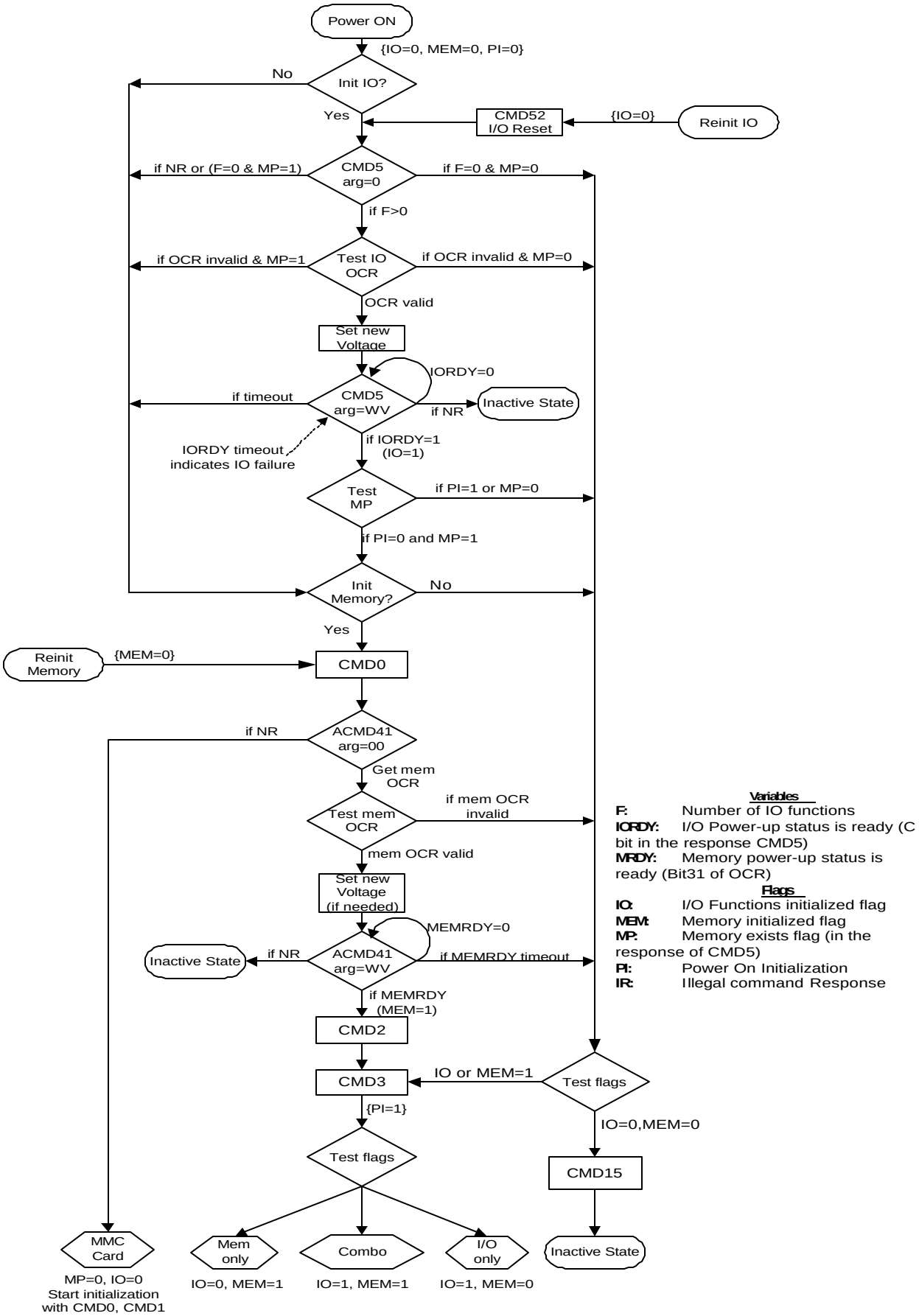


Figure 3 Card initialization flow in SD mode (SDIO aware host)

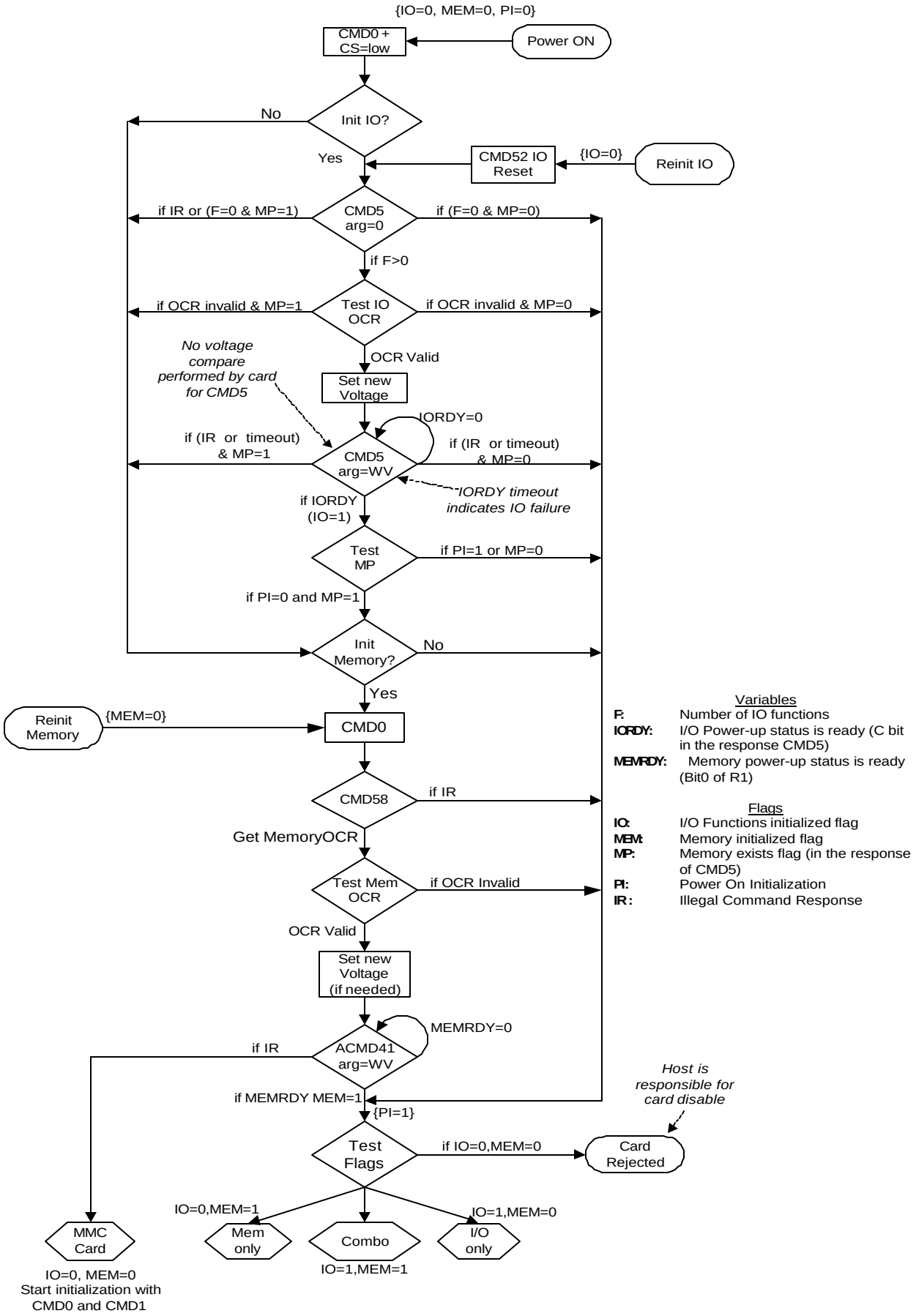


Figure 4 Card initialization flow in SPI mode (SDIO aware host)

3.2 The IO_SEND_OP_COND Command (CMD5)

Figure 5 shows the format of the IO_SEND_OP_COND command (CMD5). The function of CMD5 for SDIO cards is similar to the operation of ACMD41 for SD memory cards. It is used to inquire about the voltage range needed by the I/O card. The normal response to CMD5 is R4 in either SD or SPI format. The R4 response in SD mode is shown in Figure 6 and the SPI version is shown in Figure 7.

Figure 5 IO_SEND_OP_COND Command (CMD5)

The IO_SEND_OP_COND Command contains the following fields:

S(tart bit): Start bit. Always 0
D(irection): Direction. Always1 indicates transfer from host to card.
Command Index: Identifies the CMD5 command with a value of 000101b
Stuff Bits: Not used, shall be set to 0.
I/O OCR: Operation Conditions Register. The supported minimum and maximum values for VDD. The layout of the OCR is shown in Table 2. See section 4.7.1 for additional information.
CRC7: 7 bits of CRC data
E(nd bit): End bit, always 1

I/O OCR bit position	VDD voltage window (in Volts)
0-3	reserved
4	1.6-1.7
5	1.7-1.8
6	1.8-1.9
7	1.9-2.0
8	2.0-2.1
9	2.1-2.2
10	2.2-2.3
11	2.3-2.4
12	2.4-2.5
13	2.5-2.6
14	2.6-2.7
15	2.7-2.8
16	2.8-2.9
17	2.9-3.0
18	3.0-3.1
19	3.1-3.2
20	3.2-3.3
21	3.3-3.4
22	3.4-3.5
23	3.5-3.6

Table 2 OCR values for CMD5

3.3 The IO_SEND_OP_COND Response (R4)

An SDIO card receiving the CMD5 will respond with a SDIO unique response, R4. The format of R4 for both the SD and SPI modes is:

S	D	Reserved	C	Number of I/O functions	Memory Present	Stuff Bits	I/O OCR	Reserved	E
1	1	6	1	3	1	3	24	7	1

Figure 6 Response R4 in SD mode

R1	C	Number of I/O functions	Memory Present	Stuff Bits	I/O OCR
8	1	3	1	3	24

Figure 7 Response R4 in SPI mode

The Response, R4 contains the following data:

S(tart bit):	Start bit. Always 0
D(irection):	Direction. Always 0. Indicates transfer from card to host.
Reserved:	Bits reserved for future use. These bits will be set to 1.
C:	Set to 1 if Card is ready to operate after initialization
I/O OCR:	Operation Conditions Register. The supported minimum and maximum values for VDD. The layout of the OCR is shown in Table 2. See section 4.7.1 for additional information.
Memory Present:	Set to 1 if the card also contains SD memory. Returns 0 if the card is I/O only.
Number of I/O Functions:	Indicates the total number of I/O functions supported by this card. The range is 0-7. Note that the common area present on all I/O cards at Function 0 is not included in this count.
R1:	The normal SPI R1 response byte as described in Fig 47 of the SD physical specification.
Stuff Bits:	Not used, shall be set to 0.

Once an SDIO card has received a CMD5, the I/O portion of that card is enabled to respond normally to all further commands. This I/O enable of the functions within the I/O card will remain set until a reset, power cycle or CMD52 with write to I/O reset is received by the card. Note that a SD memory only card *may* respond to a CMD5. The proper response for a memory only card would be *Memory Present* = 1 and *Number of I/O Functions* = 0. A memory only card built to SD Memory Card specification version 1.01 would detect the CMD5 as an illegal command and not respond. Note that unlike the similar memory command ACMD41, The SPI response to CMD5 does contain the OCR value from the card

The I/O aware host will send CMD5. If the card responds with response R4, the host determines the card' s configuration based on the data contained within the R4.

4 Differences with SD Memory Specification

4.1 SDIO Command List

Table 27 shows the list of commands accepted by SD memory and SDIO cards when using the SD bus

interface. Table 28 shows the list of commands accepted by SD memory and SDIO cards when using the SPI bus interface.

4.2 Unsupported SD Memory commands

Several commands required for SD Memory devices are not supported by either SDIO-only cards or the I/O portion of Combo cards. Some of these commands have no use in SDIO devices such as Erase commands and thus are not supported in SDIO. In addition, there are several commands for SD memory cards that have different commands when used with the SDIO section of a card. Table 3 lists these SD Memory commands and the equivalent SDIO commands. For a complete list of supported and unsupported commands, see Table 27 and Table 28.

SD Memory Command	SDIO Command	Comment
CMD0	CMD52 (write to I/O reset in CCCR)	In the SD mode, the reset command (CMD0) is only used for memory or the memory portion of Combo cards. In order to reset an I/O only card or the I/O portion of a combo card, use CMD52 to write a 1 to the RES bit in the CCCR (bit 3 of register 6). Note that in SPI mode, CMD0 is only used to indicate entry into SPI mode and must be supported. An I/O only card or the I/O portion of a combo card is **not** reset with CMD0
CMD12	CMD52 (write to I/O abort)	In order to abort the block transfer of data, SD memory use CMD12. In order to abort an I/O transaction, use CMD52 to write to the abort register in the CCCR (bits 2:0 of register 6) See 4.6 for details.
CMD16	CMD52 (write to I/O Block Length)	CMD16 sets the block length for SD memory. In order to set the block length for each I/O function, use CMD52 to write the block length in the FBR
CMD2	NONE	The CID register does not exist in an SDIO only card
CMD4	NONE	The DSR register does not exist in an SDIO only card
CMD9	NONE	The CSD register does not exist in an SDIO only card
CMD10	NONE	The CID register does not exist in an SDIO only card
CMD13	NONE	An SDIO only card or the I/O portion of a combo card does not support the same SEND_STATUS (CMD13) protocol the SD memory uses. See 4.7.8.
ACMD6	CMD52 (write to Bus_Width [1:0] in CCCR)	SET_BUS_WIDTH is handled by a write to the CCCR. See 4.4 for details.
ACMD13	NONE	The SD Status register does not exist in an SDIO only card
ACMD41	CMD5	SDIO devices use the IO_SEND_OP_COND Command (CMD5). See 3.2
ACMD51	NONE	The SCR register does not exist in an SDIO only card
CMD17, CMD18, CMD24, CMD25	CMD53	I/O block operations use CMD53, rather than memory block read/write commands.

Table 3 Unsupported SD Memory Commands

4.3 Modified R6 Response

The normal response to CMD3 by a memory device is R6 as shown in Table 4. The *card status* bits (23-8) are changed when CMD3 is sent to an I/O only card. In this case, the 16 bits of response will be the SDIO-only values shown in Table 5.

Bit position	47	46	[45:40]	[39:8] Argument field		[7:1]	0
Width (bits)	1	1	6	16	16	7	1
Value	‘ 0’	‘ 0’	x	x	x	x	‘ 1’
Description	start bit	Direction bit	Command index (‘ 000011’)	New published RCA [31:16] of the card	[15:0] *card status* (see Table 5)	CRC7	end bit

Table 4 R6 response to CMD3

Bits	Identifier	Type	Value	Description	Clear Condition
15	COM_CRC_ERROR	E R	’ 0’ = no error ’ 1’ = error	The CRC check of the previous command failed	B
14	ILLEGAL_COMMAND	E R	’ 0’ = no error ’ 1’ = error	Command not legal for the card state	B
13	ERROR	E R X	’ 0’ = no error ’ 1’ = error	A general or an unknown error occurred during the operation	C
12: 0	***Reserved for future use (RFU). Will be read as 0***				

Table 5 SDIO R6 Status Bits

4.4 Bus Width

For a SD memory card, the bus width for SD mode is set using ACMD6. The SDIO card uses a write to the CCCR using CMD52 to select bus width. In the case of a combo card, both selection methods exist. In this case, the host shall set the bus width in both locations by issuing both the ACMD6 and the CCCR write using CMD52 with the same width before starting any data transfers. For details see Table 10. Note that Low-Speed SDIO cards support 4-bit transfer as an option. When communication with a Low-Speed SDIO card, the host must first determine if the card supports 4-bit transfer prior to attempting to select that mode.

4.5 Card Detect Resistor

SD memory and I/O cards use a pull-up resistor on DAT3 to detect card insertion. The procedure to enable/disable this resistor is different between SD memory and SDIO. SD memory uses ACMD42 to control this resistor while SDIO uses writes to the CCCR using CMD52. In the case of a combo card, both control locations exist and must be managed by the host. For a combo card, the resistor is enabled unless both the memory and the I/O control registers have the resistor disabled. After power-up, both locations default to resistor enabled. Note that after an I/O reset, the I/O resistor enable is not changed.

4.6 Data Transfer Abort

A host communicating with a SD memory device uses CMD12 to abort the transfer of read or write data to/from the card. For an SDIO device, CMD12 abort is replaced by a write to the ASx bits in the CCCR. Normally, the abort is used to stop an infinite block transfer (block count=0). If an exact number of blocks to be transferred, it is recommended that the host issue a block command with the correct block count, rather than using an infinite count and aborting the data at the correct time.

4.6.1 Read Abort

The host may issue an I/O abort by writing to the CCCR at any time during I/O extended read operation. The data transmission stops 2 clocks cycles after the end bit of the I/O abort command, even If the card has already begun transferring an unwanted data block while the host is issuing the abort.

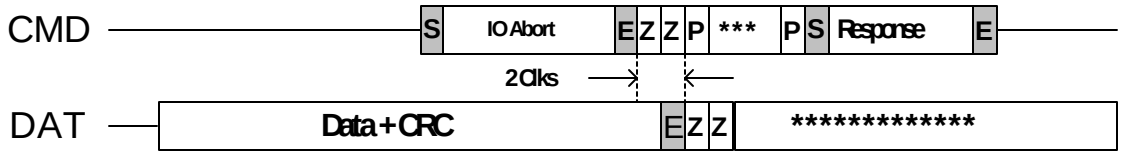
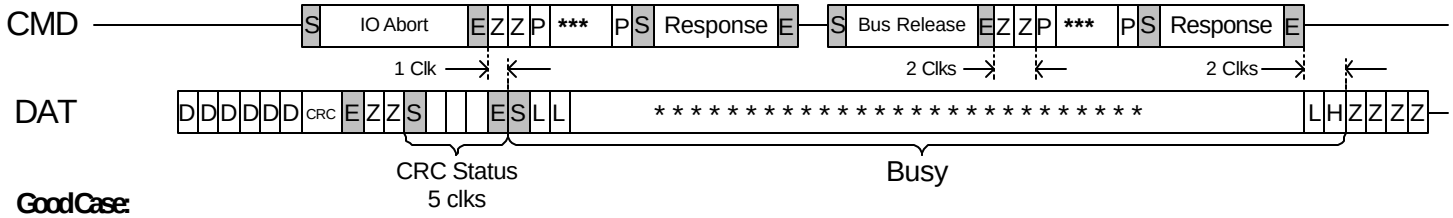


Figure 8 I/O Abort during Read Data Transfer

4.6.2 Write Abort

The host may issue an I/O abort by writing to the CCCR at any time between data blocks during I/O extended write operation. In this case, the final block transfer (including the CRC response from the card) must have been completed. This requires that the end bit of the I/O abort command should appear a maximum of two clocks before the end bit of the CRC response to the last data block. Note that the I/O abort command may be sent any time after the CRC response to the last data block. The host shall not abort in the middle of a write block. After the I/O abort is sent to the card, the card signals ' Busy' (by pulling DAT0 line to ' 0') until it has finished processing the last transferred data block. During that Busy period, the host may release the bus by writing to the CCCR BR bit. There exists some special cases when the abort is issued near the end of the CRC response to a write multiple command Figure 9 shows the case where the abort occurs after the CRC end bit (1 clock delayed). In this case, the CRC response is valid and the card accepts the prior block of write data. Figure 10 shows the illegal case where the abort is sent prior to the CRC response completion. In this case, the CRC response is invalid and the last block of data is not written to the card. Figure 11 shows the case where the abort is issued after the end of the CRC response.

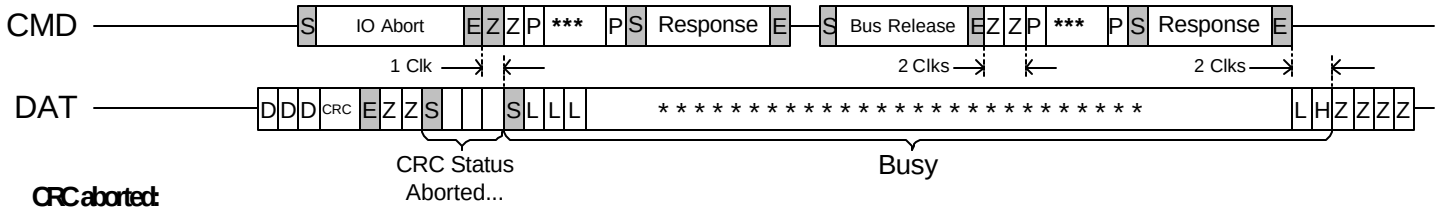


Good Case:

-Good CRC response accepted by host

-Data accepted by card

Figure 9 I/O Abort during Write CRC Response (Good)



CRC aborted:

-Partial CRC response accepted by host

-Data was not accepted by card

Figure 10 I/O Abort during Write CRC Response (Bad)

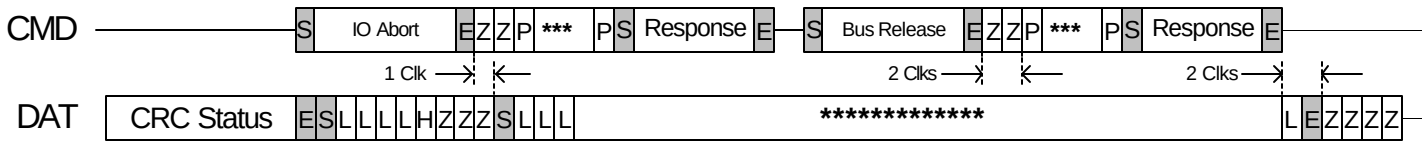


Figure 11 I/O Abort after Write CRC R esponse

4.7 Changes to SD Memory Fixed Registers

The SD memory specification defines 6 fixed card registers. They are:

1. OCR Register (32 bits)
2. CID Register (120 bits)
3. CSD Register (120 bits)
4. RCA Register (16 bits)
5. DSR Register (16 bits, optional)
6. SCR Register (64 bits)
7. SD_CARD_STATUS (512 bits)

In addition, there is a status register that is returned to the host in the form of a response (i.e. the R1b response). An SDIO only card eliminates some registers and changes some of the bits in the remaining registers. The description of these register changes follows:

4.7.1 OCR Register

All SD cards (memory, I/O and combo) shall have at least one OCR register. If the card is a combo card, it *may* have two OCR' s (one for memory and one for I/O). The memory portion of a combo card has an OCR accessed using ACMD41 and CMD58. The I/O portion of a card has an OCR with the same structure that is accessed via CMD5. If there are multiple OCR' s the voltage range may not be identical. Some I/O functions may have a wider VDD range than that reflected in the I/O OCR register. The I/O OCR will be the logical AND of the voltage ranges(s) of all I/O functions. Note that the I/O OCR format is different from the memory version in that it is only 24 bits long. For details, see Table 2. The per-function voltage for each I/O function can be read in the CIS for the card.

4.7.2 CID Register

There shall be a *maximum* of one CID register per SD card. If the card contains both memory and I/0, the CID register information is unchanged from the SD 1.01 version and reflects the information from the memory portion of the card. If the card is I/O only, the CID register and the associated access command (CMD10) are **not** supported. If the host attempts to access this register in an I/O only device, a card in SPI mode will respond with an "Invalid Command" error response and a card in SD mode will not respond.

4.7.3 CSD Register

There shall be a *maximum* of one CSD register per SD card. If the card contains both memory and I/0, the CSD register information is unchanged from the SD 1.01 version and reflects the information from the memory portion of the card. If the card is I/O only, the CSD register and the associated access command (CMD9) are **not** supported. If the host attempts to access this register in an I/O only device, a card in SPI mode will respond with an "Invalid Command" error response and a card in SD mode will not respond.

4.7.4 RCA Register

There shall only be one RCA register per SD card. The RCA value shall apply to the card as a whole. All functions and any memory share the same card address.

4.7.5 DSR Register

SDIO only cards do not support the DSR register. In the case of combo cards, support is optional as defined in the SD physical specification.

4.7.6 SCR Register

There shall be a *maximum* of one SCR register per SD card. If the card contains both memory and I/0, the SCR register information is unchanged from the SD 1.01 version and reflects the information from the memory portion of the card. If the card is I/O only, the SCR register and the associated access command (ACMD51)

are **not** supported. If the host attempts to access this register in an I/O only device, a card in SPI mode will respond with an "Invalid Command" error response and a card in SD mode will not respond.

4.7.7 SD Status

There shall be a *maximum* of one SD Status register per SD card. If the card contains both memory and I/0, the SD Status register information is unchanged from the SD 1.01 version and reflects the information from the memory portion of the card. If the card is I/O only, the SD Status register and the associated access command (ACMD13) are **not** supported. If the host attempts to access this register in an I/O only device, a card in SPI mode will respond with an "Invalid Command" error response and a card in SD mode will not respond.

4.7.8 Card Status Register

The structure of the SDIO status register is shown in Table 6. For SDIO specific operations in the SD mode that return the card status register contents (i.e. the response to CMD7), some bits are not applicable to I/O operations and will be returned as 0. These unused bits are identified as type N/A. In addition, bit 4, which is reserved in SD memory devices, is re-defined as indicating an error in the function number field of the command. For combo cards, the values returned will reflect the memory status. The CURRENT_STATE bits (12:9) will reflect the memory Controller State. For an I/O only card, the unused bits are 0 and the Current_State bits (12:9) will be 0xF (15) to identify it as an I/O only response.

Bit	Identifier	Type	Value	Description	Clear Condition
31	OUT_OF_RANGE	E R	' 0' = no error ' 1' = error	The command' s argument was out of the allowed range for this card.	C
30	ADDRESS_ERROR	N/A	0	Not used with SDIO operation	C
29	BLOCK_LEN_ERROR	N/A	0	Not used with SDIO operation	C
28	ERASE_SEQ_ERROR	N/A	0	Not used with SDIO operation	C
27	ERASE_PARAM	N/A	0	Not used with SDIO operation	C
26	WP_VIOLATION	N/A	0	Not used with SDIO operation	C
25	CARD_IS_LOCKED	N/A	0	Not used with SDIO operation	C
24	LOCK_UNLOCK_FAILED	N/A	0	Not used with SDIO operation	C
23	COM_CRC_ERROR	E R	' 0' = no error ' 1' = error	The CRC check of the previous command failed.	B
22	ILLEGAL_COMMAND	E R	' 0' = no error ' 1' = error	Command not legal for the card state	B
21	CARD_ECC_FAILED	N/A	0	Not used with SDIO operation	C
20	CC_ERROR	N/A	0	Not used with SDIO operation	C
19	ERROR	E R	' 0' = no error ' 1' = error	A general or an unknown error occurred during the operation.	C
18	UNDERRUN	N/A	0	Not used with SDIO operation	C
17	OVERRUN	N/A	0	Not used with SDIO operation	C
16	CID/ CSD_OVERWRITE	N/A	0	Not used with SDIO operation	C
15	WP_ERASE_SKIP	N/A	0	Not used with SDIO operation	C
14	CARD_ECC_DISABLED	N/A	0	Not used with SDIO operation	C
13	ERASE_RESET	N/A	0	Not used with SDIO operation	C
12:9	CURRENT_STATE	S X	15=I/O only	For an I/O only card, the current state will be fixed at a value of 0x0F. This indicates that it is an I/O only card and the normal memory states do not apply	B
8	READY_FOR_DATA	N/A	0	Not used with SDIO operation	C
7:6	***Reserved***				

5	APP_CMD	N/A	0	CMD55 not used in SDIO operation	C
4	FUNCTION_NUMBER ERROR	E X	' 0' = no error ' 1' = error	An invalid function number was requested. Not used in SD memory	C
3	AKE_SEQ_ERROR (SD Memory Card app.spec.)	N/A	0	Not used with SDIO operation	C
2	***Reserved for application specific commands***				
1, 0	***Reserved for manufacturer test mode***				

Table 6 SDIO Status Register Structure

5 New I/O Read/Write Commands

Two additional data transfer instructions have been added to support I/O. IO_RW_DIRECT, a direct I/O command similar to the MMC 'Fast I/O' command, and IO_RW_EXTENDED, which allows large address spaces and fast access with byte or block addresses. Both commands are in class 9 (I/O Commands).

5.1 IO_RW_DIRECT command (CMD52)

The IO_RW_DIRECT is the simplest means to access a single register within the total 128K of register space in any I/O function, including the common I/O area (CIA). This command reads or writes 1 byte using only 1 command/response pair. A common use is to initialize registers or monitor status values for I/O functions. This command is the fastest means to read or write single I/O registers, as it requires only a single command/response pair.

S	D	Command Index 110100b	R/W flag	Function Number	RAW flag	Stuff	Register Address	Stuff	Write Data or Stuff Bits	CRC 7	E
1	1	6	1	3	1	1	17	1	8	7	1

Figure 12 IO_RW_DIRECT Command

The IO_RW_DIRECT Command contains the following fields:

S(tart bit): Start bit. Always 0

D(irection): Direction. Always1 indicates transfer host to card.

Command Index: Identifies the "IO_RW_DIRECT" command with a value of 110100b

R/W Flag: This bit determines the direction of the I/O operation. If this bit is 0, this command will read data from the SDIO device at the address specified by the Function Number and the Register Address to the host. This data byte will be returned in the response, R5. If this bit is set to 1, the command will write the bytes in the Write Data field to the I/O location addressed by the Function Number and the Register Address. If the RAW flag is 0, then the data in the register that was written will be read and that value returned in the response.

RAW Flag: The Read after Write flag. If this bit is set to 1 and the R/W flag is set to 1, then the command will read the value of the register after the write. This is useful to allow writing to a control register and reading the status at the same address. If this bit is cleared, the value returned in the R5 response shall be the same as the write data in the command. If this bit is set, the data field of the R5 response will contain the value read from the addressed register after the write operation.

Function Number: The number of the function within the I/O card you wish to read or write. Note that function 00 selects the common I/O area (CIA).

Register Address: This is the address of the byte of data inside of the selected function that will be read or written. There are 17 bits of address available so the register must be

located within the first 128K (131,072) addresses of that function.

Write Data/Stuff Bits: For a direct write command (R/W=1), this is the byte that will be written to the selected address. For a direct read (R/W=0), this field is not used and shall be set to 0.

CRC7: 7 bits of CRC data

E(nd bit): End bit, always 1

5.2 IO_RW_DIRECT Response (R5)

The SDIO card's response to CMD52 will be in one of two formats. If the communication between the card and host is in the 1-bit or 4-bit SD mode, the response will be in a 48-bit response (R5) as described in 5.2.1. If the communication is using the SPI mode, the response will be a 16-bit R1 response as described in 5.2.2.

5.2.1 CMD52 Response (SD modes)

The SDIO card's response to CMD52 in the SD mode is shown in Figure 13. If the operation was a read command, the data being read is returned as an 8-bit value. In addition, 15 bits of status information is returned. The format of the SD response is as follows:

S	D	Command Index 110100b	Stuff	Response Flags Bit 7----------0	Read or Write Data	CRC7	E
1	1	6	16	8	8	7	1

Figure 13 R5 IO_RW_DIRECT Response (SD modes)

The IO_RW_DIRECT response (R5) contains the following fields:

S(tart bit): Start bit. Always 0

D(irection): Direction. 0 indicates transfer card to host (Response)

Command Index: Identifies the "IO_RW_DIRECT" command with a value of 110100b

Stuff Bits Not used, shall be set to 0

Response Flags 8 Bits of flag data indicating the status of the SDIO card. Table 7 shows the format of these flag bits.

Read or Write Data: For an I/O write (R/W=1) with the RAW Flag set (RAW=1) this field will contain the value read from the addressed register after the write of the data contained in the command. Note that in this case, the read-back data may not be the same as the data written to the register, depending on the design of the hardware. For an I/O write with the RAW bit=0, the SDIO function will not do a read after write operation, and the data in this field will be identical to the data byte in the write command. For an I/O read (R/W=0), the actual value read from that I/O location is returned in this field.

CRC7: 7 bits of CRC data

E(nd bit): End bit, always 1

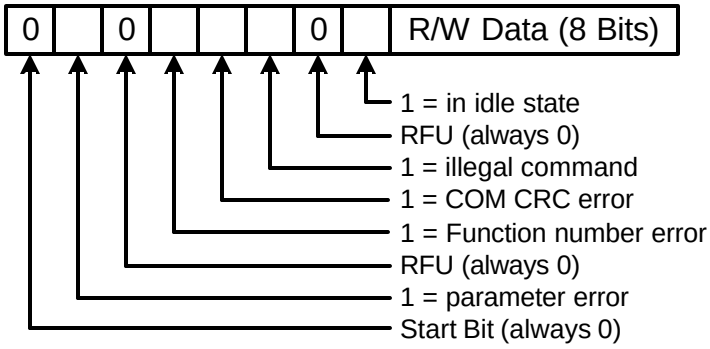
Bits	Identifier	Type	Value	Description	Clear Condition
7	COM_CRC_ERROR	E R	'0' = no error '1' = error	The CRC check of the previous command failed.	B
6	ILLEGAL_COMMAND	E R	'0' = no error '1' = error	Command not legal for the card State.	B
5-4	IO_CURRENT_STATE	S	00=DIS 01=CMD	**DIS=Disabled:** Initialize, Standby and	B

			02=TRN 03=RFU	Inactive States (card not selected) **CMD=DAT lines free:** 1. Command waiting (No transaction suspended) 2. Command waiting (All CMD53 transactions suspended) 3. Executing CMD52 **TRN=Transfer:** Command executing with data transfer using DAT line	
3	ERROR	E R	' 0' = no error ' 1' = error	A general or an unknown error occurred during the operation.	C
2	RFU	--	Fixed at 0	Reserved for Future Use	C
1	FUNCTION_NUMBER	E X	' 0' = no err ' 1' = error	An invalid function number was requested	C
0	OUT_OF_RANGE	E R	' 0' = no error ' 1' = error	The command' s argument was out of the allowed range for this card.	C

Table 7 Flag data for IO_RW_DIRECT SD Response

5.2.2 R5, IO_RW_DIRECT Response (SPI mode)

The SDIO card' s response to CMD52 in the SPI mode is shown in Figure 14. If the operation was a read command, the data being read is returned as an 8-bit value. In addition, 7 bits of status information is returned. The format of the SD response is shown in Figure 14.



Note the read/write data is identical to the read/write data described for the SD R5 response (see 5.2.1)

Figure 14 IO_RW_DIRECT Response in SPI Mode

5.3 IO_RW_EXTENDED command (CMD53)

In order to read and write multiple I/O registers with a single command, a new command, IO_RW_EXTENDED is defined. This new command will be included in the new command class 9 (I/O Commands). This command allows the reading or writing of a large number of I/O registers with a single command. Since this is a data transfer command, it provides the highest possible transfer rate.

S	D	Command Index 110101b	R/W flag	Function Number	Block Mode	OP Code	Register Address	Byte/Block Count	CRC 7	E
1	1	6	1	3	1	1	17	9	7	1

Figure 15 IO_RW_EXTENDED Command

The IO_RW_EXTENDED Command contains the following fields:

S(tart bit): Start bit. Always 0

D(irection): Direction. Always1 indicates transfer host to card.

Command Index: Identifies the "IO_RW_EXTENDED" command with a value of 110101b

R/W Flag: This bit determines the direction of the I/O operation. If this bit is 0, this command will read data from the SDIO device at the address specified by the Function Number and the Register Address to the host. The read data will be returned on the DATx lines. If this bit is set to 1, the command will write the bytes from the DATx lines to the I/O location addressed by the Function Number and the Register Address.

Function Number: The number of the function within the I/O card you wish to read or write. Note that function 0x00 selects the common I/O area (CIA).

Block Mode (Optional) this bit, if set to 1, indicates that the read or write operation will be performed on a block basis, rather than the normal byte basis. If this bit is set, the Byte/Block count value will contain the number of blocks to be read/written. The block size for functions 1-7 is set by writing the block size to the I/O block size register in the FBR (See Table 11 and Table 12). The block size for function 0 is set by writing to the FN0 Block Size register in the CCCR. Card and host support of the block I/O mode is optional. The host can determine if a card supports block I/O by reading the *Card supports MBIO* bit (SMB) in the CCCR (see Table 10).

OP code Defines the read/write operation as described in Table 8

Register Address: Start Address of I/O register to read or write. Range is [0x1FFFF:0]

Byte/Block Count If the command is operating on bytes (Block Mode = 0), then this field contains the number of bytes to read or write. A value of 0x000 will cause 512 bytes to be read or written. If the command is in block mode (Block Mode=1), the Block Count field specifies the number of Data Blocks to be transferred following this command. A value of 0x00 indicates that the count set to infinite. In this case, the I/O blocks will be transferred until the operation is aborted by writing to the I/O abort function select bits (ASx) in the CCCR (see Table 9 and Table 10).

CRC7: 7 bits of CRC data

E(nd bit): End bit, always 1

OP code	Command operation
0	Multi byte R/W to fixed address
1	Multi byte R/W to incrementing address

Table 8 IO_RW_EXTENDED command Op Code definition

- OP Code 0 is used to read or write multiple bytes of data to/from a single I/O register address. This command is useful when I/O data is transferred using a FIFO inside of the I/O device. In this case, multiple bytes of data are transferred to/from a single register address. For this operation, the address of the register is set into the Register Address field. Data is transferred on the DAT lines as defined for SD memory cards. The number of bytes to be written or read is set in the Byte Count field. If the block length is set to 0, then 512 bytes of data will be written or read.

- OP Code 1 is used to read or write multiple bytes of data to/from an I/O register address that increment by 1after each operation. This command is used when large amounts of I/O data exist within the I/O card in a RAM like data buffer. In this operation, the start address is loaded into the Register Address field. The first operation occurs at that address within the I/O card. The next operation will occur at address+1 with the address incrementing by 1 until the operation has completed. As with OP Code 0, the number of bytes is set in the Byte Count field of the command.

The response from the SDIO card to CMD53 will be R5 (the same as CMD52) as defined in 5.2. For CMD53, the 8-bit data field will be stuff bits and shall be read as 0x00.

5.3.1 CMD53 Data Transfer Format

When executing the IO_RW_EXTENDED (CMD53), the multi-byte or multi-block data transfer is similar to the data transfer for memory. For the multi-byte transfer modes (block mode=0) the following applies:

IO_RW_EXTENDED byte read is similar to CMD17 (READ_SINGLE_BLOCK)

IO_RW_EXTENDED byte write is similar to CMD24 (WRITE_BLOCK)

Note that the byte count for this transfer is set in the command, rather than the fixed block size. Thus, the size of the data payload will be in the range of 1-512 bytes. The block mode is similar to the following memory commands:

IO_RW_EXTENDED block read is similar to CMD18 (READ_MULTIPLE_BLOCK)

IO_RW_EXTENDED block write is similar to CMD25 (WRITE_MULTIPLE_BLOCK)

For the block mode the only difference is that for a fixed block count, the host does not need to stop the transfer, as it will continue until the block count is satisfied. If the block count is set to zero, the operation is identical to the memory mode in that the host must stop the transfer.

5.3.2 Special Timing for CMD53 multi -block read

The optional read multi-block operation using CMD53 has a special timing requirement for the SDIO card. In order to allow the host to stall the read data of the first block of read data; the transfer of data from the card to the host shall not start until after the end bit of the response. Figure 16 shows the timing for the first block of read data.

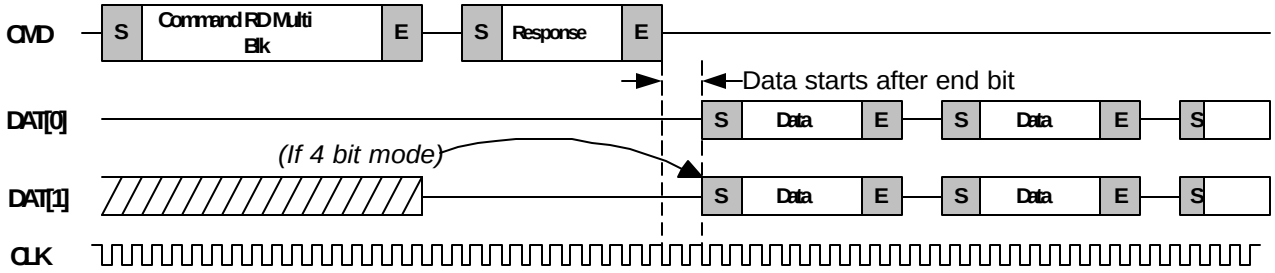


Figure 16 Multi -Block Read Timing

6 SDIO Card Internal Operation

I/O access differs from memory in that the registers can be written and read individually and directly without a FAT file structure or the concept of blocks (although block access is supported). These registers allow access to the I/O data, control of the I/O function and report on status or transfer I/O data to/from the host. The SD memory relies on the concept of a fixed block length with commands reading/writing multiples of these fixed size blocks. I/O may or may not have fixed block lengths and the read size may be different from the write size. Because of this, I/O operations may be based on either a length (byte count) or a block size.

6.1 Overview

Each SDIO card may have from 1 to 7 functions plus one memory function built into it. A function is a self contained I/O device. I/O functions may be identical or completely different from each other. All I/O functions are organized as a collection of registers. There is a maximum of 131,072 (2^{17}) registers possible for each I/O function. These registers and their individual bits may be read Only (RO), Write Only (WO) or Read/Write (R/W). These registers can be 8, 16 or 32 bits wide within the card. All addressing is based on byte access. These registers can be written and/or read one at a time, multiply to the same address or multiply to an incrementing address. The single R/W access is often used to initialize the I/O function or to read a single status or data value. The multiple reads to a fixed address are used to read or write data from a data FIFO register in the card. The read to incrementing addresses is used to read or write a collection of data to/from a RAM area inside of the card. Figure 17 shows the mapping of the CIA and optional CSA space for an SDIO card.

6.2 Register Access Time

All registers in SDIO only cards and the SDIO portion of Combo cards must complete read and write data transfer in less than 1 second. The host can use 1 second as the timeout value for a non-responding location. If a functions needs to support an access time greater than 1 second. The card maker will use some function specific method that is not defined in this specification. The 1-second response time is dependant on the SDCLK frequency. If the Average frequency of the SDCLK is less than 100KHz then the card may not be able to respond within the 1-second limit. The host should adjust the timeout in the case the average frequency of the SDCLK is less than 100KHz.

6.3 Interrupts

All SDIO hosts shall support interrupts in the SPI and 1-bit SD modes. Each function within an SDIO or Combo card may implement interrupts as needed. The interrupt used on SDIO functions is a type commonly called " level sensitive" . Level sensitive means that any function may signal for an interrupt. Since there is only 1 interrupt line, it may be shared by multiple interrupt sources. The function must continue to signal the interrupt until the host responds and clears the interrupt. Since multiple interrupts may be active at once, it is the responsibility of the host to determine the interrupt source(s) and deal with it as needed. This is done on the SDIO function by the use of two bits, the interrupt enable and interrupt pending. Each function that may generate an interrupt has an interrupt enable bit. In addition, the SDIO card has a master interrupt enable that controls all functions. An interrupt will only be signaled to the SD bus if both the function' s enable and the card' s master enable are set. The second interrupt bit is called interrupt pending. This read-only bit tells the host which function(s) may be signaling for an interrupt. There is an interrupt pending bit for each function that can generate interrupts. These bits are located in the CCCR area. For more details, see Table 9 and Table 10. Interrupt operation is described more fully in section 8.

6.4 Suspend/Resume

Within a multi-function SDIO or a Combo card, there are multiple devices (I/O and memory) that must share access to the SD bus. In order to allow the sharing of access to the host among multiple devices, SDIO and

combo cards can implement the *optional* concept of suspend/resume. If a card supports suspend/resume, the host may temporarily halt a data transfer operation to one function or memory (suspend) in order to free the bus for a higher priority transfer to a different function of memory. Once this higher-priority transfer is complete, the original transfer is re-started where it left off (resume). Support of suspend/resume is optional on a per-card basis. If suspend/resume is implemented, it shall be supported by the memory (if any) of a Combo card and all I/O functions *except* 0 (the CIA). Note that the host can suspend multiple transactions and resume them in any order desired. I/O function 0 does ***not*** support suspend/resume. Suspend/Resume is described in more detail in section 9. Note that Suspend/Resume is defined only for the SD 1 and 4-bit modes. It does not apply to SPI transfers.

6.5 Read Wait

Host devices built to the SD Physical specification version 1.01 must control the SDCLK to stop the read data block output from a card executing a multiple read command whenever the host cannot accept more data. During the time that the host has stopped the SDCLK, a CMD52 cannot be issued. This limitation causes a problem in that a host device built to the SD Physical specification version 1.01 cannot perform the I/O command during multiple read cycle.
In order to eliminate this limitation, the SDIO specification adds the read wait control to enable the host to issue CMD52 during a multiple read cycle. Read Wait uses the DAT[2] line to allow the host to signal the card to temporarily halt the sending of read data by a card. This feature is optional for an SDIO or combo card. However, if an SDIO or combo supports Read Wait, all functions and any memory must support read wait. Read Wait is described in more detail in section 10. Note that Read Wait is defined only for the SD 1 and 4-bit modes. It does not apply to SPI transfers.

6.6 SDIO Fixed Internal Map

The SDIO card has a fixed internal register space and a Function unique area. The fixed area contains information about the card and certain mandatory and optional registers in fixed locations. The fixed locations allow any host to obtain information about the card and perform simple operations such as enable in a common manner. The function unique area is a per-function area, which is defined by the vendor and different for each SDIO function. Figure 17 shows the internal map of an SDIO card with multiple functions.

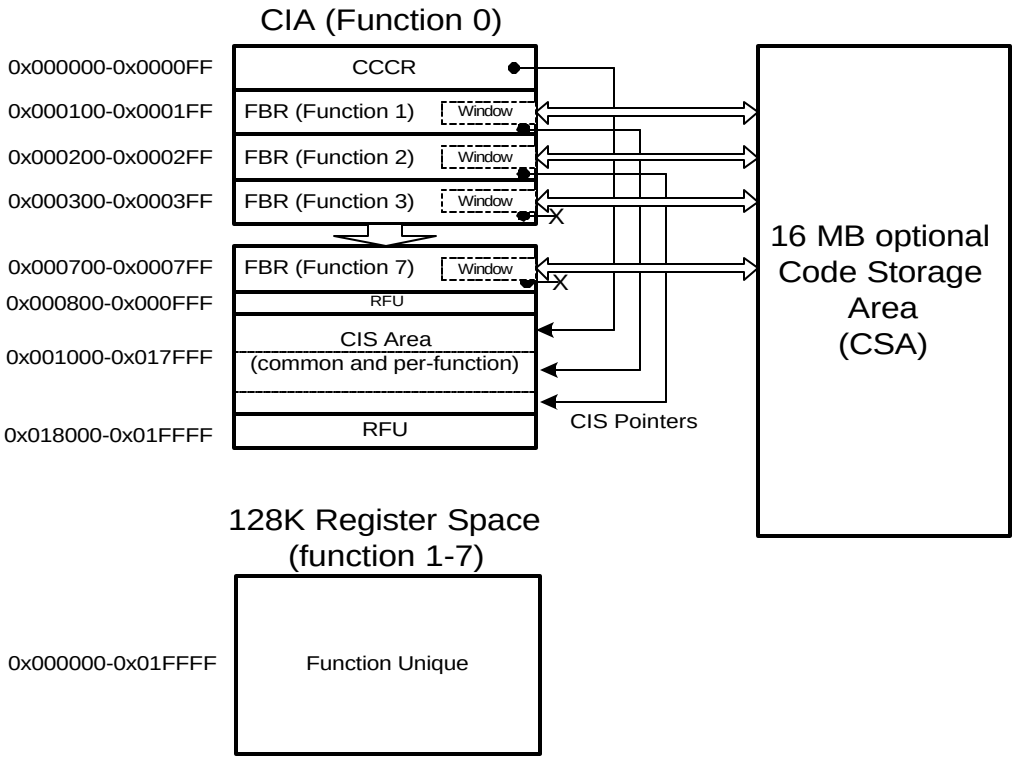


Figure 17 SDIO Internal Map

6.7 Common I/O Area (CIA)

The Common I/O Area (CIA) shall be implemented on all SDIO cards. The CIA is accessed by the host via I/O reads and writes to function 0. The registers within the CIA are provided to enable/disable the operation of the I/O function(s), control the generation of interrupts and optionally load software to support the I/O functions. The registers in the CIA also provide information about the function(s) abilities and requirements. There are three distinct register structures supported within the CIA. They are:

1. Card Common Control Registers (CCCR)
2. Function Basic Registers (FBR)
3. Card Information Structure (CIS)

6.8 Card Common Control Registers (CCCR)

The Card Common Control Registers allow for quick host checking and control of an I/O card' s enable and interrupts on a per card (master) and per function basis. The bits in the CCCR are mixed Read/Write and read only. If all 7 functions are not provided on an SDIO card, the bits corresponding to unused functions shall all be read-only and read as 0. All reserved for future use bits (RFU) shall be read-only and return a value of 0. Access to the CCCR is possible even after initialization when the I/O functions are disabled. Access is performed using the I/O read and write commands defined in section 5. This allows the host to enable functions after initialization. The CCCR is organized as follows:

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00	CCCR/SDIO Revision	SDIO bit 3	SDIO bit 2	SDIO bit 1	SDIO bit 0	CCCR bit 3	CCCR bit 2	CCCR bit 1	CCCR bit 0
0x01	SD Specification Revision	RFU	RFU	RFU	RFU	SD bit 3	SD bit 2	SD bit 1	SD bit 0
0x02	I/O Enable	IOE7	IOE6	IOE5	IOE4	IOE3	IOE2	IOE1	RFU
0x03	I/O Ready	IOR7	IOR6	IOR5	IOR4	IOR3	IOR2	IOR1	RFU
0x04	Int Enable	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IENM
0x05	Int Pending	INT7	INT6	INT5	INT4	INT3	INT2	INT1	RFU
0x06	I/O Abort	RFU	RFU	RFU	RFU	RES	AS2	AS1	AS0
0x07	Bus Interface Control	CD Disable	RFU	RFU	RFU	RFU	RFU	Bus Width 1	Bus Width 0
0x08	Card Capability	4BLS	LSC	E4MI	S4MI	SBS	SRW	SMB	SDC
0x09-0x0B	Common CIS Pointer	Pointer to card' s*common* Card Information Structure (CIS)							
0x0C	Bus Suspend	RFU	RFU	RFU	RFU	RFU	RFU	BR	BS
0x 0D	Function Select	DF	RFU	RFU	RFU	FS3	FS2	FS1	FS0
0x 0E	Exec Flags	EX7	EX6	EX5	EX4	EX3	EX2	EX1	EXM
0x 0F	Ready Flags	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RFM
0x10-0x11	FN0 Block Size	I/O block size for Function 0							
0x12-0xFF	RFU	Reserved for Future Use (RFU)							

Table 9 Card Common Control Registers (CCCR)

Field	Type	Description
CCCRx	R/O	CCCR Format Version number. These 4 bits contain the version of the CCCR format that this card supports. The codes for the CCCR formats are as follows: Value CCCR Format Version 0x00 CCCR Version 1.01 0x01-0x0F Reserved for Future Use
SDIOx	R/O	SDIO Specification Revision number. These 4 bits contain the version of the SDIO specification that this card supports. The codes for the SDIO specifications are as follows: Value SDIO Specification 0x00 SDIO Specification 1.01 0x01-0x0F Reserved for Future Use
SDx	R/O	SD Format Version number. These 4 bits contain the version of the SD Physical specification that this card supports. The codes for the SD physical specification are as follows: Value SD Physical Specification 0x00 SD Physical Specification 1.01 (March 2000) 0x01-0x0F Reserved for Future Use
IOEx	R/W	Enable Function - If this bit is reset to 0, the function is disabled. If this bit is set to 1, the function is enabled. On power up or after a reset, this bit shall be set to 0
IORx	R/O	I/O Function Ready- If this bit is reset to 0, the function is not ready to operate. This may be caused by the function being disabled or not ready due to internal causes such as a built-in self-test in progress. If this bit is set to 1, the function is ready to

		operate. On power up or after a reset, this bit shall be set to 0. For any function that is not implemented on an SDIO card, this bit will always be 0.
IENx	R/W	Interrupt Enable for function x. If this bit is cleared to 0, any interrupt from this function will not be sent to the host. If this bit is set to 1, then this function's interrupt will be sent to the host if the master Interrupt Enable (bit 0) is also set to 1.
IENM	R/W	Interrupt Enable Master. If this bit is cleared to 0, no interrupts from this card will be sent to the host. If this bit is set to 1, then any function's interrupt will be sent to the host.
INTx	R/O	Interrupt Pending for function x. If this bit is cleared to 0, this indicates that no interrupts are pending from this function. If this bit is set to 1, then this function has interrupt pending. Note that if the IENx *or* IENM bits are not set, the host will not receive this pending interrupt.
ASx	W/O	Abort Select In order to abort an I/O read or write and free the SD bus, the function that is currently transferring data must be addressed. These 3 bits define which function's transfer to stop. For example, the abort the transfer to function number 3, the value of 0x03 would be written to these bits. If the abort is addressed to a suspended function, it does not affect current data transaction. Note that this is an abort, not a reset. The addressed function will return to the CMD state and data transfer pending to that function will be halted. This abort procedure will not work for SPI write operations. To abort an SPI write data transfer use the STOP_TRAN token as defined in section 7.3.3 of the SD 1.01 Physical specification. This form of abort applies only to functions of an SDIO card. For the memory of a combo card, the abort methods defined in the SD physical specification shall be used to abort transfers to/from memory
RES	W/O	I/O CARD RESET Setting the RES to 1 will cause all functions in an SDIO or Combo card to perform a hard reset identical to a power-on. When RES=1, the values of AS2-0 are don't cares. The RES bit is auto cleared, so there is no need to rewrite a value of 0. This bit is write-only, any read will return an undetermined value. Memory in a combo card is not affected.
Bus Width 1:0	R/W	Defines the data bus width (' 00' =1bit or' 10' =4 bits bus) to be used for data transfer. All SDIO cards support both 1 and 4-bit bus. On reset or power-on these bits are cleared to 00.
CD Disable	R/W	Connect[0]/Disconnect[1] the 50Kohm pull-up resistor on CD/DAT3 (pin 1) of the card. The pull-up may be used for card detection. This bit is cleared to 0 on power-on (connected). Its state is not affected by a reset command.
SDC	R/O	This flag bit reports the SDIO card's ability to execute CMD52 while data transfer is in progress If this bit is set, all I/O functions will accept and execute the CMD52 while data transfer is underway on the DAT[] lines. Also, any memory in a combo card must allow the CMD52 to execute while it is transferring data. Since the CMD52 does not use the DAT[] lines, it is possible to execute while data transfer to a different function on the card is underway. CMD52 is described in 5.1.
SMB	R/O	This flag bit reports the SDIO card's ability to execute the IO_RW_EXTENDED command (CMD53) in the block mode. If this bit is set, all I/O functions (0-7) will accept and execute CMD53 with the optional block mode bit set. The IO_RW_EXTENDED command is described in 5.3
SRW	R/O	Card Supports Read Wait. This flag bit reports the SDIO card's ability to support the Read Wait Control (RWC) operation. If set, this bit identifies the card as able to accept the wait signal on DAT[2]. RWC operation is described in section 6.5.
SBS	R/O	Card supports Suspend/Resume. This flag bit reports the SDIO card's ability to Suspend and Resume operations at the request of the host. If this bit is set, all functions except 0 and memory (if any) will accept a request to suspend operations and resume under host control. Suspend/Resume operation is described in 6.4.
S4MI	R/O	Supports interrupt between blocks of data in 4-bit mode. This flag bit reports the SDIO card's ability to generate interrupts during a 4-bit multi-block data transfer. If this bit is 0, then the SDIO card is not able to signal an interrupt during a multi-block

		data transfer in 4-bit mode. In this case, the interrupt will not be signaled until after the data transfer is complete. If this bit is 1, then the SDIO card is able to signal an interrupt between blocks while data transfer is in progress. This operation is described in 8.1.4
E4MI	R/W	Enable interrupt between blocks of data in 4-bit mode. Enable the multi-block IRQ during 4-bit transfer for the SDIO card. When this bit is 0, the card will not signal interrupts during a 4-bit multi-block data transfer. If this bit is 1, the card will generate interrupts during 4 bit multi-block data transfers as described in 8.1.4 If this SDIO card does not support 4 bit multi-block IRQ' s (bit 4=0), then this bit will be R/O and always read as 0. This bit will be cleared to 0 by any reset
LSC	R/O	Card is a Low-Speed device. If this bit is set, it indicates that the SDIO card is a Low-Speed device (see 2.1). If this bit is clear, the SDIO card is a Full-Speed device.
4BLS	R/O	4-bit support for Low-Speed cards. If the SDIO card is a Low-Speed card (LSC=1) and it supports 4-bit data transfer, then this bit will be set. If the card is not Low-Speed or if the card does not support 4-bit transfer, then this bit will be zero.
Pointer to card' s common CIS	R/O	This 3-byte pointer points to the start of the card' s common CIS. The common CIS contains information relation to the entire card. The card common CIS will be located within the CIS space of function 0 (0x001000- 0x017FFF) as described in section 6.10. A card common CIS is mandatory for all SDIO cards. This pointer is stored in little-endian format (LSB first).
BS	R/O	Bus status: If this bit is set to 1, then the currently addressed function (selected by FSx or by the function number in an I/O command) is currently executing a command which will transfer data on the DAT[] line(s). If this bit is 0, then the addressed function is not using the data bus. This bit is used by the host to determine which function of a multi-function or combo card is currently performing data transfer. Note that this bit is a part of the optional Suspend/Resume protocol. If the card does not support Suspend/Resume, this bit will be read as 0. Any access to the CIA may not be suspended, so in this case, BS will always be set to 1, irrespective of the host setting BR to 1.
BR	R/W	Bus Release Request/Status: This bit is used to request that the addressed function (selected by FSx or by the function number in an I/O command) release the Data lines and suspend operation. If the host sets this bit to 1, the addressed function will temporarily halt data transfer on the DAT[] lines and suspend the command that is in process. The BR bit will remain set to 1 until the release is complete. Once the function is in suspend, it will signal the host by clearing the BS and BR bits. The host can monitor the status of the suspend request by reading the BR bit. If it is set, the suspend request is still in progress. A pending suspend request can be cancelled by the host by writing 0 to the BR bit. However, the host should monitor the BS and Exec bits to confirm that the suspend request was cancelled rather that granted. If this card does not support Suspend/Resume, this bit shall be R/O and read as 0.
FSx	R/W	Select Function bits 3:0 These four bits are used to select a function number (0-7) or the memory of a combo card (8) for Suspend/Resume. There are 2 means to write the value of FSx. First, an I/O writes to the register in the CCCR and second, a new I/O command will cause the FSx to be set to the function number in that command. The value of FSx will remain until overwritten. If a function or memory is currently suspended, the writing of it' s number to FSx will restart (resume) the data transfer operation When reading FSx, the value returned will be the number of the currently addressed function. Note that when reading FSx, if the Bus Status is 0 (BS=0), the FSx value is undefined. The FSx bits are coded as follows: **FSx** — **Current Transaction** 0000 — Transaction of function 0 (CIA) 0001-0111 — Transaction to functions 1-7

		1000 Transaction of memory in combo card 1001-1111 Not defined, reserved for future use
DF	R/O	Resume Data Flag: A data transaction is resumed by writing its number to FSx. Once the transaction is resumed, the DF indicates if more data will be transferred. If DF is cleared to 0, then no additional data will be transferred after the function or memory is resumed. If DF is set to 1, then there is more data to transfer that will begin after the function or memory in resumed. The DF flag can be used to control the interrupt cycle in 4-bit mode. If DF=1, there is more data to transfer after restoring the function. In this case, the interrupt cycle should be disabled. If DF=0, the function or memory was suspended at end of data transfer (during busy). In this case, no data transfer will begin after resume so the host can detect a start interrupt cycle after restore.
EXx	R/O	Execution Flag bits 7:0 These bits are used by the host to determine the current execution status of all functions (7-1) and memory (0). The bit is set to 1 for each function or memory that is currently executing a command. The EXx bits tell the host that a function or memory is currently executing a command so no additional command should be issued to that function/memory. These bits are only active if the card supports suspend/resume and suspend/resume is currently active. If suspend/resume is not active these bits shall be read as zero.
RFx	R/O	Ready Flag bits 7:0 These bits tell the host the read or write busy status for functions (1-7) and memory (0). If a function or memory is executing a write transaction, an RFx bit cleared to 0 indicates the function/memory is busy and not ready to accept more data. If the RFx bit I set to 1, then the function/memory can accept write data. If a function/memory is executing a read command, if the RFx bit is cleared to 0, it indicates that read data is NOT available. If the bit is set to 1, it indicates that read data is ready to be transferred. These bits are only active if the card supports suspend/resume and suspend/resume is currently active. If suspend/resume is not active these bits shall be read as zero
FN0 Block Size	R/W	This 16-bit register sets the block size for I/O block operations for Function 0 only. If this card does not support I/O block operations (SMB=0), then this register becomes read-only and will always read 0x0000. The maximum block size is 2048 (0x0800) and the minimum is 1. At power-up or reset, this register will be initially loaded with a value of 0x0000. The host is responsible for setting the appropriate value for the block size supported by function 0. This pointer is stored in little-endian format (LSB first).
RFU	R/O	Any bit defined as Reserved for Future Use (RFU) shall be read-only and shall be read as 0.

Table 10 CCCR bit definitions

6.9 Function Basic Registers (FBR)

In addition to the CCCR, each supported I/O function has a 256-byte area used to allow the host to quickly determine the abilities and requirements of each function and to enable software loading. The address of this area is from 0x00n00 to 0x00nFF where n is the function number (0x1 to 0x7). This per-function area is structured as follows:

Address	7	6	5	4	3	2	1	0
0x100	Function 1 CSA enable	Function 1 supports CSA	RFU	RFU	Function 1 Standard I/O device interface code			
0x101-0x108	Reserved for Future Use (RFU)							
0x109-0x10B	Pointer to Function 1 Card Information Structure (CIS)							
0x10C-0x10E	Pointer to Function 1 Code Storage Area (CSA)							
0x10F	Data access window to Function 1 Code Storage Area (CSA)							
0x110-0x111	I/O block size for Function 1							
0x112-0x1FF	Reserved for Future Use							
0x200-0x7FF	Function 2 to 7 Function Basic Information Registers (FBR)							
0x800-0xFFF	Reserved for Future Use							

Table 11 Function Basic Information Registers (FBR)

Field	Type	Description
I/O device interface code	R/O	The Standard I/O device code identifies those I/O functions, which implement the recommended standard interface as defined in a separate application specification. The codes assigned to those standard interfaces are: 0x0 No SDIO standard interface supported by this function 0x1 Indicates that this function supports the SDIO UART standard interface 0x2 Indicates that this function supports the SDIO "thin" Bluetooth standard interface 0x3 Indicates that this function supports the SDIO "fat" Bluetooth standard interface 0x4-0xF Not assigned, reserved for future use
Function supports CSA	R/O	If this function supports and contains a Code Storage Area (CSA), this bit will be set to 1. If this function does not support a CSA, this bit will be cleared to 0. CSA enable is controlled by bit 7 of register 0xn00.
Function CSA enable	R/W	This bit controls access to the Code Storage Area for this function. If this bit is cleared to 0, then any read or write access to the CSA will be blocked. If this bit is set to 1, then access to the CSA is allowed. This bit is cleared to 0 upon reset. If this function does not support CSA (0xn00 bit 6=0), then this bit shall be R/O and always read as 0.
Address pointer to Function CIS	R/O	These three bytes make up a 24-bit pointer (only the lower 17 bits are used) to the start of the Card Information Structure (CIS) that is associated with each function. The CIS is defined in section 6.10. A CIS is mandatory for each function on an SDIO card. This pointer is stored in little-endian format (LSB first).
Address pointer to Function CSA	R/W	These three bytes make up a 24-bit pointer to the desired byte in the CSA to read or write. After any read or write to the CSA access window register, this pointer will be automatically incremented by 1. If this function does not support CSA (0xn00 bit 6=0), then these 24 bits shall be R/O and always read as 0x000000. This pointer is stored in little-endian format (LSB first).
Data access window to CSA	R/W	Any read or write to this address when the CSA is enabled (0xn00 bit 7=1), will pass data to/from the byte addressed by the CSA address pointer. If this function does not support CSA (0xn00 bit 6=0), then these 24 bits shall be R/O and always read as 0x000000. This pointer is stored in little-endian format (LSB first).

Function 1-7 I/O Block Size	R/W	This 16-bit register sets the block size for I/O block operations for each function (1-7). If this card does not support I/O block operations (SMB=0), then this register becomes read-only and will always read 0x0000. The maximum block size is 2048 (0x0800) and the minimum is 1. At power-up or reset, this register will be initially loaded with a value of 0x0000. The host is responsible for setting the appropriate value for the block size supported by each function. This pointer is stored in little-endian format (LSB first).

Table 12 FBR bit definitions

6.10 Card Information Structure (CIS)

The Card Information Structure provides more complete information about the card and the individual functions. The CIS is the common area to read information about all I/O functions that exist in a card. The design is based on the PC Card16 design standardized by PCMCIA. All cards that support I/O must have a common CIS and a CIS for each function. The CIS is accessed by reads to a fixed area as shown in Table 13 This one area serves the card as a Global CIS and also as the storage area for each function. The common area and each function have a pointer to the start of its CIS within this memory space.

Address	7	6	5	4	3	2	1	0
0x0001000 - 0x017FFF	Card Global Card Information Structure (CIS) area for card common and all functions							
0x018000- 0x01FFFF	Reserved for Future Use							

Table 13 Card Information Structure (CIS) and reserved area of CIA

The valid Tuples (storage structures) from the PCMCIA specification and new tuples created for SDIO are defined in section 13.6.

6.11 Multiple Function SD Cards

Multiple Function SDIO Cards shall have a separate set of Configuration registers for each function on the card. Multiple Function SDIO Cards shall use a combination of a global CIS common to all functions on the card and a separate function-specific CIS specific to each function on the card. The global CIS describes features that are common to all functions on the card. Each function-specific CIS describes features specific to a particular function on the SDIO Card.

7 Embedded I/O Code Storage Area (CSA)

In order to support the concept of "Plug-and-Play" for SDIO cards, each function contained in a card may need to contain a block of memory for the storage of drivers and/or applications. In addition, since the same SDIO card may be used on multiple different host platforms, several different versions of the code may be needed for each function. One option is to store these programs in a standard SD Memory section of a combo card. Alternately, a standard access means to load the code is contained in the optional Code Storage Area (CSA). The CSA is a separate 16MB memory area that is accessed using the CSA address pointer and the CSA window register contained in the FBR registers. Note that each function may have it's own CSA to support it. The CSA data can be read only or R/W. The actual storage method for the CSA is not a part of this specification and left to the implementers.

7.1 CSA Access

In order for the host to access a function's CSA, it first must determine if that function supports a CSA. The host reads the FBR register at address 0x00n00 where n is the function number (0x1 to 0x7). If bit 6=1, then the function supports a CSA and the host enables access by writing bit 7=1. The next step is for the host to load the 24-bit address to start reading or writing. This is accomplished by writing registers 0x00n0C to 0x00n0E where n is the function number (0x1 to 0x7). Once the start address is written, data can be read or written by accessing register 0x00n0F, the CSA data window register. If more than 1 byte needs to be read or written, an extended I/O command (byte or block) can be performed with an OP code of 0 (fixed address). The address pointer will be automatically incremented with each access to the window register, so the access will be to sequential addresses within the CSA. Once the operation is complete, the address of the NEXT operation will be held in the 24-bit address register for the host to read.

7.2 CSA Data Format

The data stored in the CSA shall be structured using the FAT12/FAT16 format. This format is defined in the ISO specification*: ISO/IEC9293:1994 Information technology - Volume and file structure of disk cartridges for information interchange*. This specification is also the basis for the SD memory cards. That information can be found in the SDA publication*: Part 2 FILE SYSTEM SPECIFICATION Version 1.0 February 2000*. The actual layout of files within the CSA is undefined by this specification.

The use of the CSA for program or data storage for different host types requires that the SDIO card manufacturer load the programs and data in a file format that will be recognized by the host. An example of this would be the use of a specific file name saved within a specific subdirectory that is recognized and executed by a particular host operating system. Such formats are specific and sometimes proprietary to different host implementations and operating systems.

8 SDIO Interrupts

In order to allow the SDIO card to interrupt the host, an interrupt function is added to a pin on the SD interface. Pin number 8, which is used as DAT[1] when operating in the 4-bit SD mode, is used to signal the card' s interrupt to the host. The use of interrupt is optional for each card or function within a card. The SDIO interrupt is " level sensitive" , that is, the interrupt line must be held active (low) until it is either recognized and acted upon by the host or de-asserted due to the end of the Interrupt Period (see 8.1.2). Once the host has serviced the interrupt, it is cleared via some function unique I/O operation. The interrupt output of all SDIO cards is active low. All hosts must provide pull-up resistors on all data lines DAT[3::0] as described in section 4.3.2 of the SD Physical Specification.

8.1 Interrupt Timing

The operation of the interrupt pin is different between the SPI mode and the SD mode. The operation of the interrupt pin is defined as follows:

8.1.1 SPI and SD 1-bit mode interrupts

In the SPI and 1-bit SD mode, Pin 8 is dedicated to the interrupt function. Thus, in the 1-bit modes there are no timing constraints on interrupts. A card in the SPI or 1-bit SD mode signals an interrupt to the host by asserting pin 8 low. The host detects this pending interrupt using a level sensitive input. The host is responsible for clearing the interrupt.

8.1.2 SD 4-bit mode

Since Pin 8 is shared between the IRQ and DAT[1] use in the 4-bit SD mode, an interrupt shall only be sent by the card and recognized by the host during a specific time. The time that a low on Pin 8 will be recognized as an interrupt is defined as the ***Interrupt Period***.

An SDIO host shall only sample the level on Pin 8 (DAT[1]/IRQ) into the interrupt detector during the Interrupt Period. At all other times, the host interrupt controller shall ignore the level on Pin 8. Note that the Interrupt Period is applicable for both memory and I/O operations

The definition of the Interrupt Period is different for operations with single block and multiple block data transfer.

8.1.3 Interrupts with Single Data Transaction

In case of single data block transmission, the Interrupt Period is defined as the time between:

- Two clocks after the completion of a data packet and:
- The completion of sending the end bit of the next command that has a data block on the DAT line.

Figure 18 and Figure 19 show the timing of the Interrupt Period for two different types of single data transaction read cycles. Figure 20 shows the timing of a single data block transaction write cycle.

This Interrupt Period is intended to prevent the interaction of the DAT[1] data and the interrupt signals on a common pin. Note that the Interrupt period includes the times when there is no command or data activity between the host and the card. In the case where the interrupt mechanism is used to wake the host while the card is in a low power state (i.e. no clocks), Both the card and the host shall be placed into the 1-bit SD mode prior to stopping the clock.

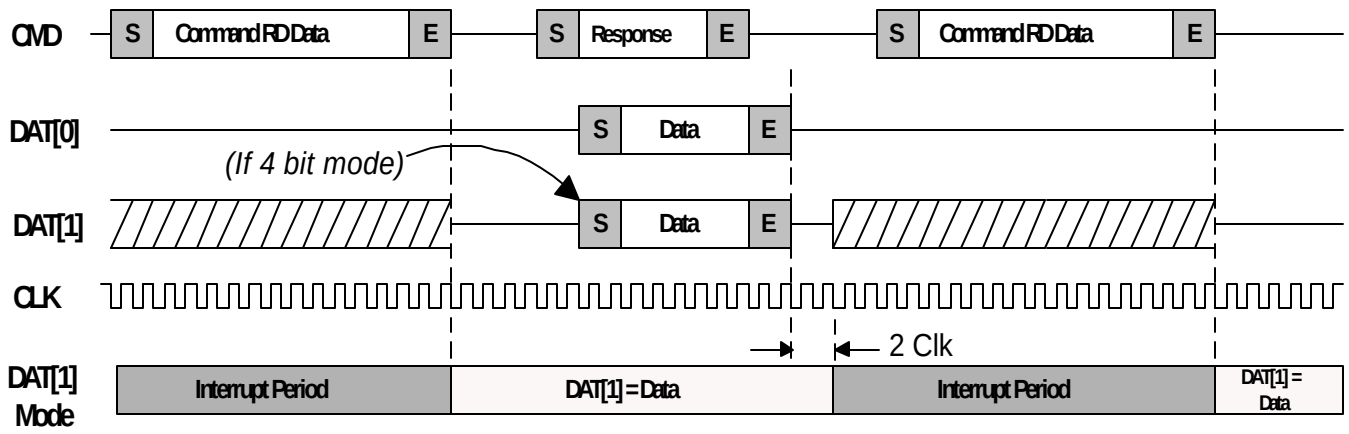


Figure 18 Read Interrupt Cycle Timing

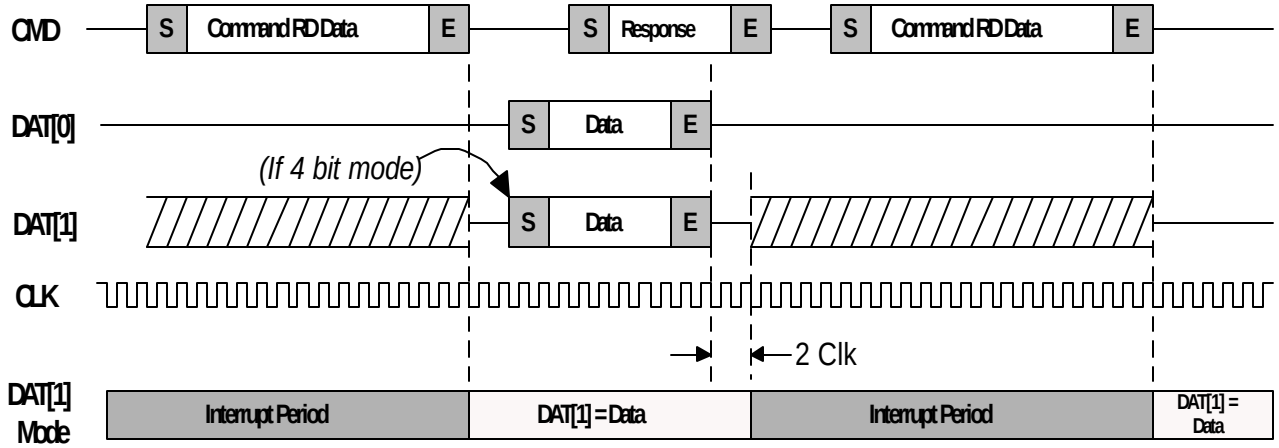


Figure 19 Alternate Read Interrupt Cycle Timing

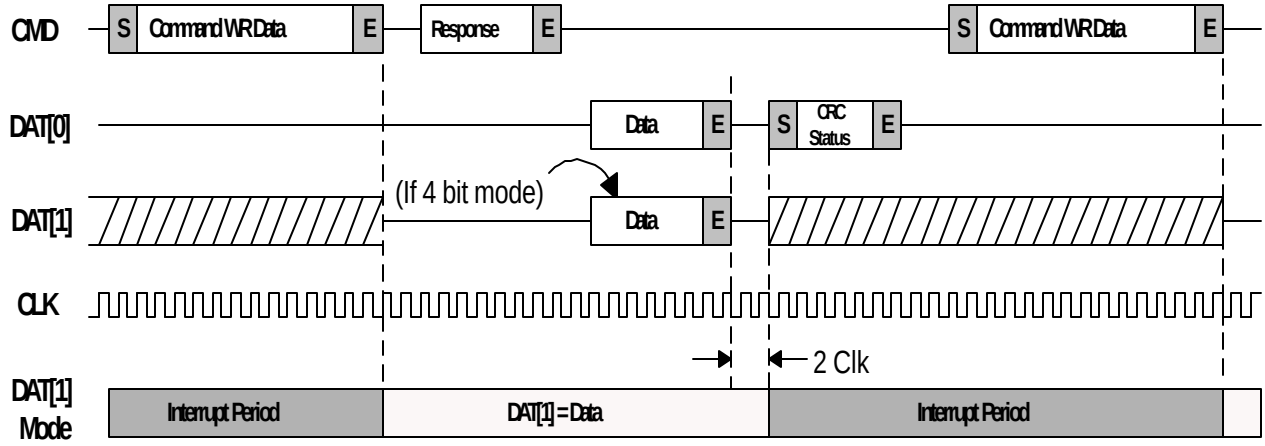


Figure 20 Write Interrupt Cycle Timing

Figure 21 describes another case of the Interrupt Period, The case where a command is followed by neither data blocks nor response after the current command. In this case, the end bit timing of this command should not terminate the Interrupt Period

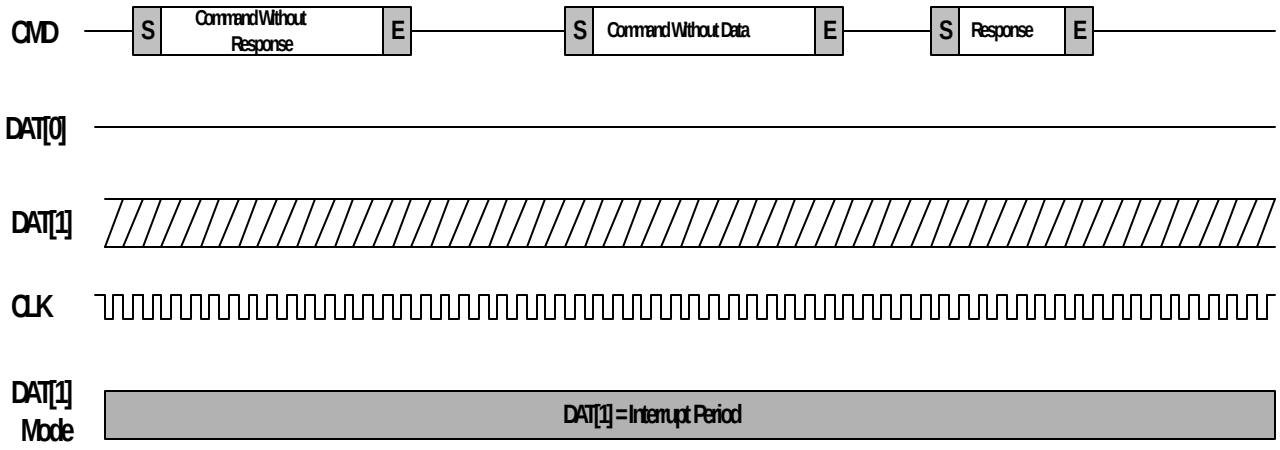


Figure 21 Continuous Interrupt Cycle

8.1.4 Interrupts with 4-bit Multiple Block Data Transactions (Optional)

Due to the tight timing and limited data line availability when transferring multiple blocks of data in the 4-bit SD mode, a special definition of the interrupt period is required. In order to allow the highest possible data throughput, the interrupt period is limited to a 2-clock interrupt period that begins 2 clocks after the End bit of the data blocks. This interrupt period is the same for Read and Write operations. Because only 2 clock cycles are available in the interrupt period, any card that wishes to signal an interrupt to the host during 4-bit multi-block operations shall assert DAT[1] low for the first clock and high for the second clock. The card shall then release DAT[1] into the hi-Z State. Figure 22 shows this operation for an interrupt during a 4-bit multi-block read and Figure 23 illustrates the interrupt during a write sequence.

It is important to note that if a combo card supports interrupts, then the interrupt can occur during data transfers by both the I/O and the memory portions of the card. It is also important to note that since the interrupt can occur only between block of read or write data in the 4-bit mode, certain conditions may cause very large delays in signaling the interrupt to the host. For example, if the host stops read data by stopping the clock, or delays the transmission or write data blocks, the interrupt will be delayed until after the block of data is transferred.

An important point to note is that the interrupt period occurs during the same time that the Start bit could be placed on the bus by the card for a read operation. In order to allow the host to differentiate an interrupt from a Start bit, the card shall maintain the DAT[0], DAT[2] and DAT[3] lines in the hi-Z State during the interrupt period. The host can then differentiate the interrupt from a start bit by looking for a low on all 4 DAT lines (Start bit) or a low only on DAT[1] (IRQ).
After sending a last data block, the Interrupt Period starts 2cycles after the End bit of the last data block, and ends after the End bit of the next command, which is identical to the definition described in 8.1.3

Card support of the 4-bit SDIO Interrupt on multiple block data transfers as described in section 8.1.4 is optional. In the case of combo cards (SD memory plus IO) that support 4-bit interrupts, to avoid the conflict between interrupt cycle and data transfer, it is necessary to design memory control to be able to insert interrupt cycle between data blocks

Hosts can determine if a card or function supports this mode by reading corresponding bits of CCCR as described in 6.8

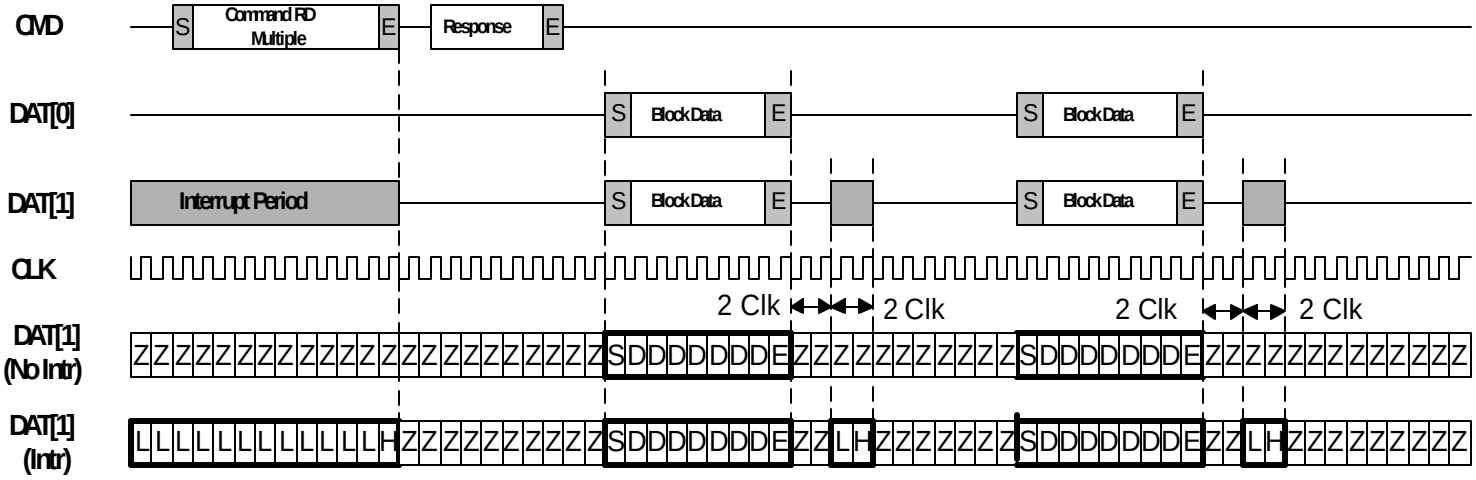


Figure 22 Multiple Block 4-Bit Read Interrupt Cycle Timing

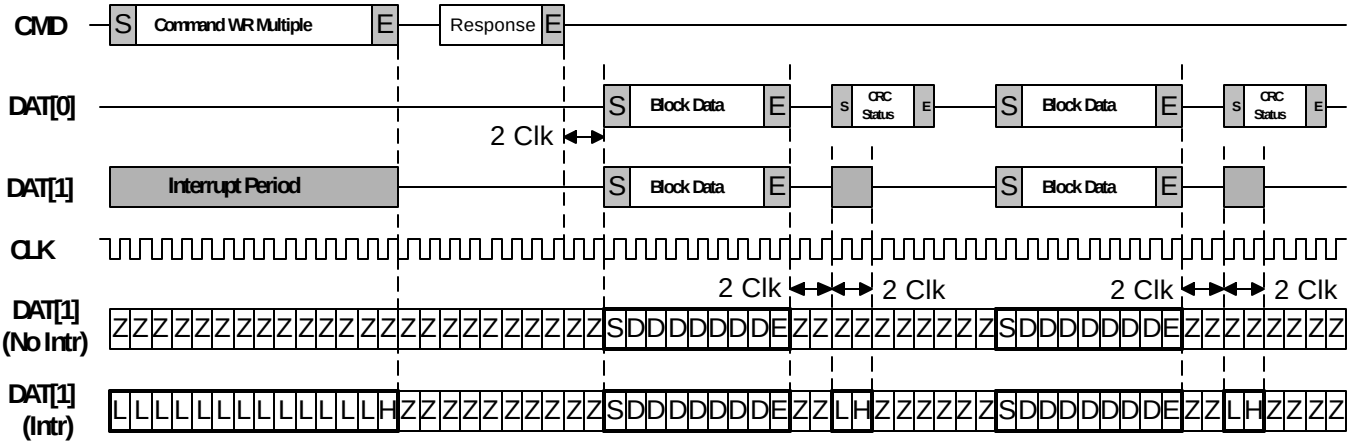


Figure 23 Multiple Block 4-Bit Write Interrupt Cycle Timing

8.1.5 Inhibited Interrupts

It is important to note that SDIO hosts that use the 4-bit interrupt should not issue new commands with data transfers while the previous command's data is on the bus. The overlapping of previous data and a new command followed by data block may prevent card from signaling interrupt to host. Figure 24 shows the condition where interrupts are inhibited by a command with data sent during the previous command's data transfer.

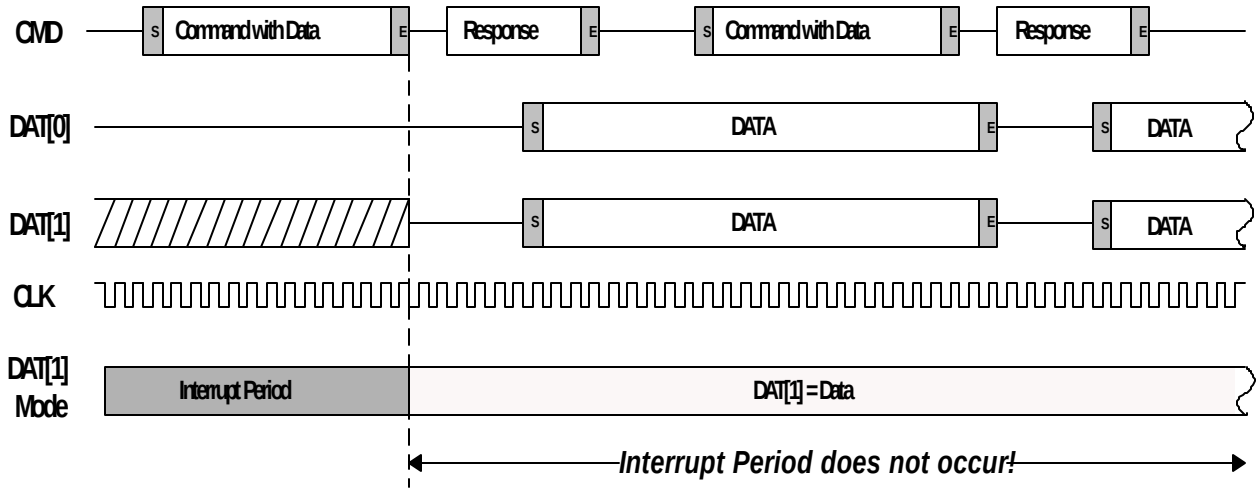


Figure 24 Inhibited Interrupts Timing

8.1.6 End of Interrupt Cycles

In SPI and both SD modes, The card must drive DAT[1] high for a minimum of 1 clock cycle before terminating the Interrupt Period. This signaling is shown in Figure 25 shows the situation where the period of interrupt signaling is shorter than that of the Interrupt Period. Figure 26 shows the case where interrupt is de-asserted at the end of the Interrupt Period. In both cases, the DAT[1] is driven to a high level prior to the card releasing the line to the hi-Z state.

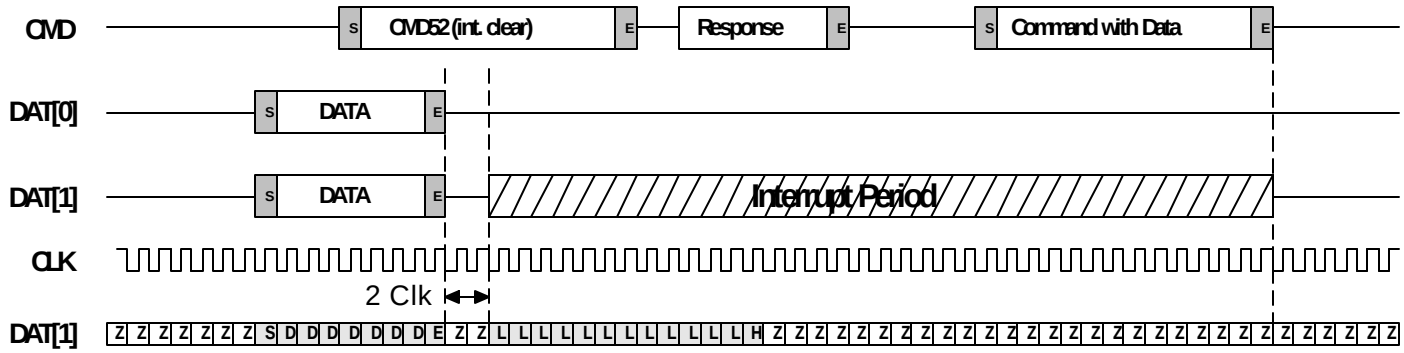


Figure 25 Interrupt Cycle Timing

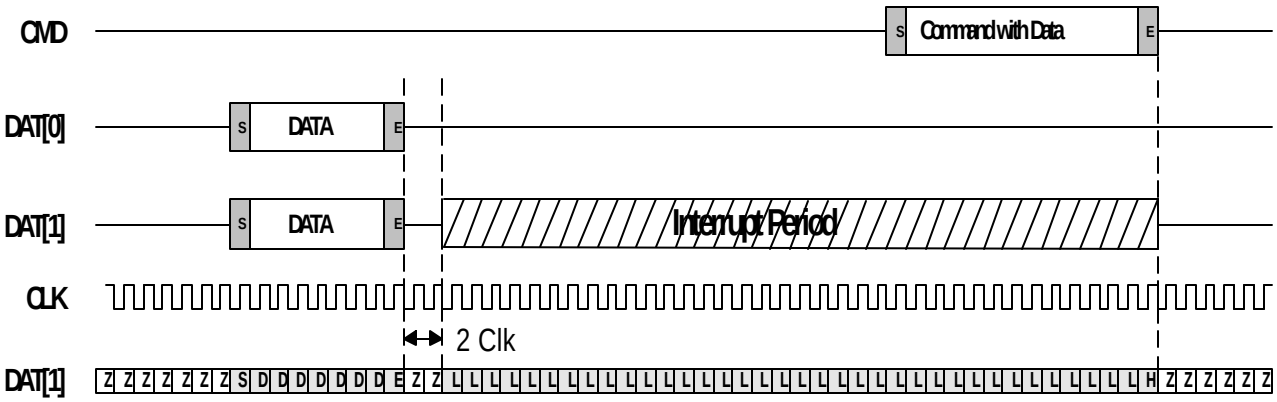


Figure 26 Alternate Interrupt Cycle Timing

8.1.7 Terminated Data Transfer Interrupt Cycle

In the case of the Read Multiple Block or Write Multiple Block commands to the memory of a combo card, a special case for the timing of the interrupt period is created when the transfer is terminated. The Read Multiple and Write Multiple commands causes the card or host to transmit blocks of data until the operation is terminated by the host issuing a CMD12 or I/O Abort. In the case of a host transfer abort (memory or IO), the timing of the interrupt period is the same. The interrupt period (if active) will end 2 clocks after the end bit of the stop transmission command (CMD12 or I/O Abort). The interrupt period will re-start 2 clocks after the end bit of the response. There are 2 cases for the interrupt period, depending on when the host issues the CMD12 or I/O Abort. Figure 27 shows the timing of the interrupt period for a Read Multiple command if the stop transmission is issued after a block of data has begun. In this case, the normal interrupt period begins 2 clocks after the end of the last data block. The interrupt period again begins 2 clocks after the end of the response to the transfer abort.

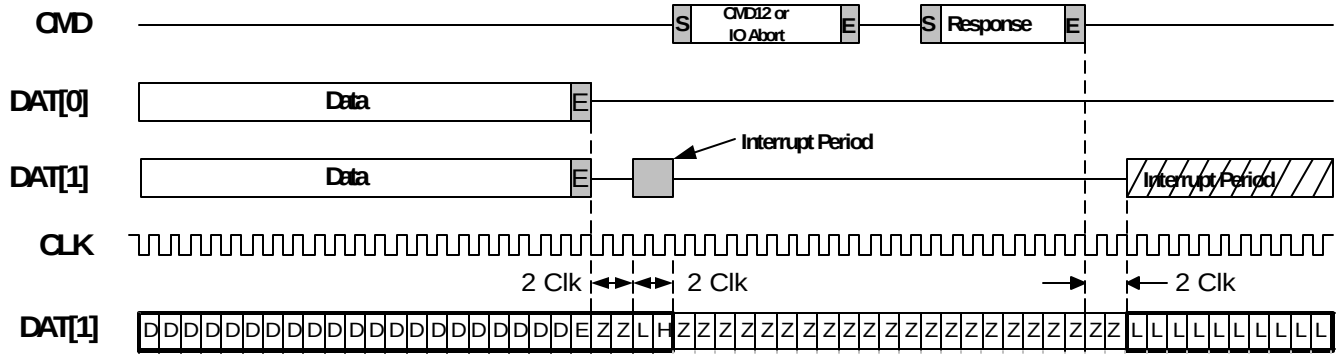


Figure 27 Terminated Read Multiple Interrupt timing (Case 1)

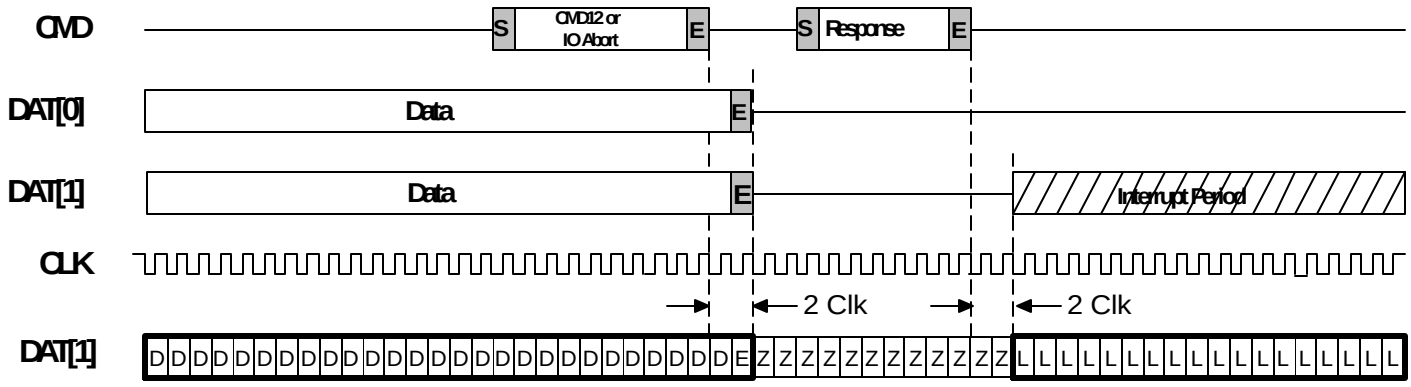


Figure 28 Terminated Read Multiple Interrupt timing (Case 2)

Figure 28 shows a second case where the read multiple transfer was terminated during data transfer and the effect on the interrupt period. The read data transfer is terminated by a CMD12 (memory) or an abort I/O write (CMD52). One clock after the abort, the data will terminate and the card will send a single end bit before releasing the data bus. Two clocks after the response end, the interrupt period begins when the card can signal interrupt by asserting a 0 (low) on the DAT[1] line.
The termination of a Write Multiple command has the same timing as shown in Figure 27 and Figure 28.

Symbol	Data Line State
S	Start bit (= ‘ 0’)
E	End bit (=’ 1’)
Z	High impedance state (-> = ‘ 1’)
D	Data bit
L	Drive low
H	Drive high

Table 14 Timing Diagram Symbols

8.1.8 Interrupt Clear Timing

Since the SDIO card uses level sensitive interrupts, the host must clear pending interrupts with an I/O read or write to some function unique area. In some host implementations, the sending of a CMD52 to the card is handled by host adapter hardware while the host CPU can execute other operations. This condition may allow an interrupt that has already been handled to re-interrupt the host if the timing of the interrupt clear is not controlled. To prevent this condition, Any SDIO card that implements interrupts must follow some required timing with respect to removing the interrupt from the DAT[1] line after the write to the function unique area that clears the interrupt. The clearing of the interrupt can be caused by an IO write in a function unique method, or by a function unique IO read. An example of clearing an interrupt using an IO read would be a function where the reading of a data register may automatically clear the data ready interrupt. Since the clearing of the pending interrupt may be caused by a read or write operation to function unique areas of the SDIO card, the following requirements are placed on the card:

- In the case of a CMD52 read or write, the interrupt shall be removed from the DAT[1] line prior to the end of the R5 response.
- In the case of a CMD53 read or write the interrupt shall be removed prior to the end of the data response (in write operations) or the end of CRC in the data token (in read operations) for the data block that cleared the interrupt.

9 SDIO Suspend/Resume Operation

The procedure used to perform the Suspend/Resume operation on the SD bus is:
1. The host determines which function is currently using the DAT line(s).
2. The host requests the lower priority or slower transaction to suspend.
3. The host checks for the transaction suspension to complete.
4. The host begins the higher priority transaction.
5. The host waits for the completion of the higher priority transaction.
6. The host restores the suspended transaction.

The operation of suspend resume is diagrammed in Figure 29 to Figure 34. For these figures, showing suspend request using DIRECT command in the Write and Read (RAW) mode, suspend is requested by setting BR=1 at write, and in the response, a return of BS=0 indicate bus release. BR is automatically cleared by the condition BS=0.

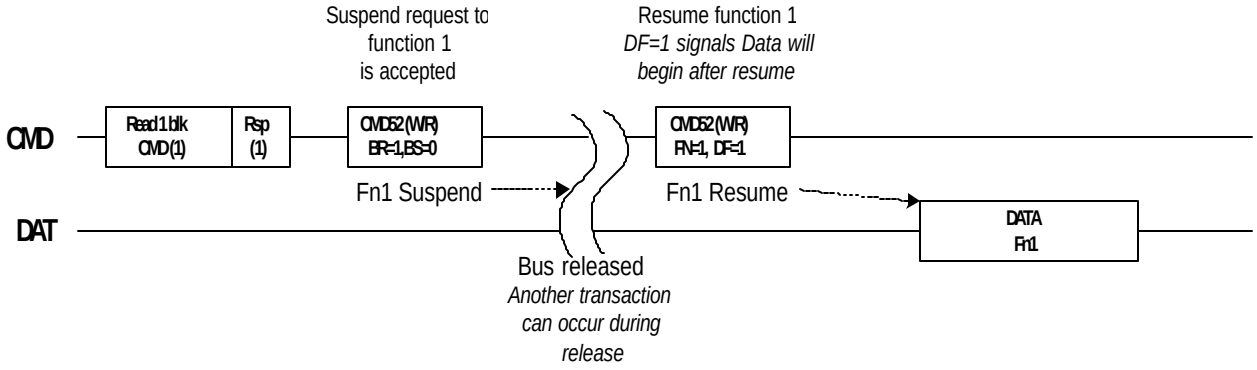


Figure 29 Device with long read latency will accept bus suspend

Figure 29 shows a device with a long read latency receiving and accepting a suspend request prior to the start of data transfer. The request is accepted and the bus is free for another transaction. At the end of this transaction, the resume causes the delayed read transfer to begin.

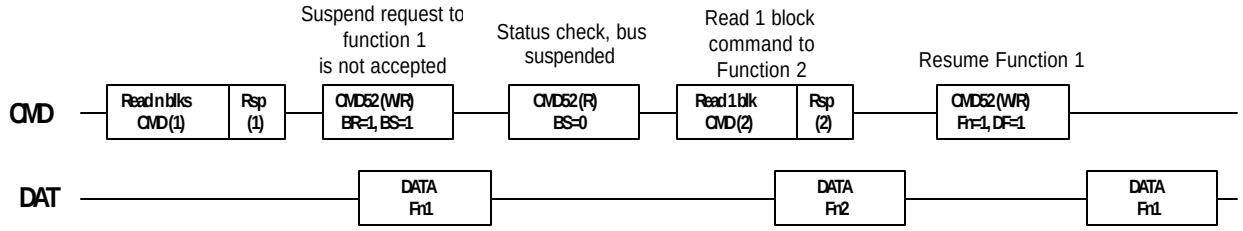


Figure 30 Function2 read cycle is inserted during Function1 multiple read cycle

Figure 30 shows a condition where the first suspend request is not immediately accepted. The host then checks the status of the request with a read and determines that the bus has now been released (BS=0). At this time, a read to function 2 is started. Once that single block read is complete, the resume is issued to function, causing the data transfer to resume (DF=1).

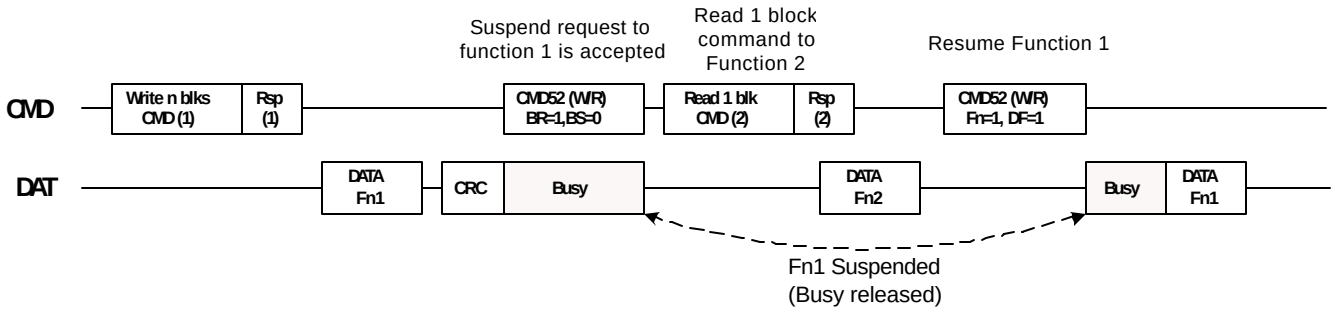


Figure 31 Write suspended during busy (case 1)

Figure 31 shows a write operation that causes a busy condition. The host then suspends that transaction and begins another transaction to function 2. Once that transaction is complete, the host resumes the suspended function 1, which is still busy. Once the busy is clear, the write data transfer begins again.

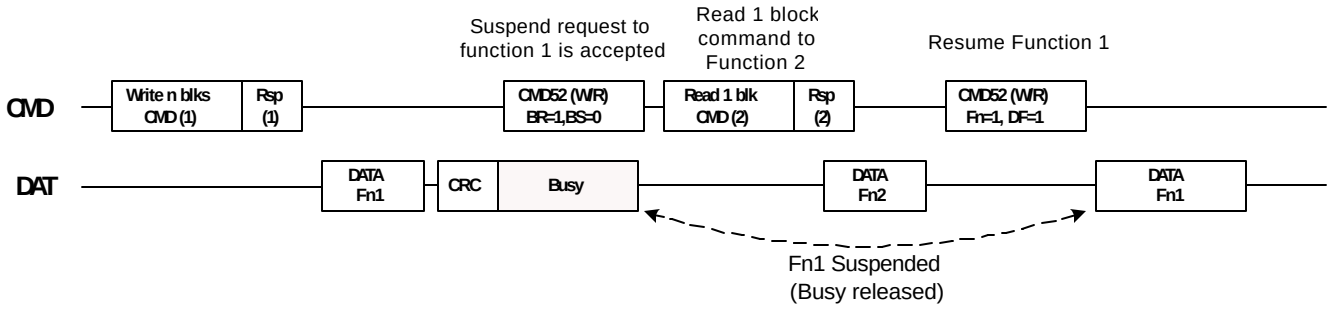


Figure 32 Write suspended during busy (case 2)

Figure 32 is identical to Figure 31 except that the write busy was terminated by function 1 during the time it was suspended. In this case, upon resume function 1 begins to transfer write data immediately.

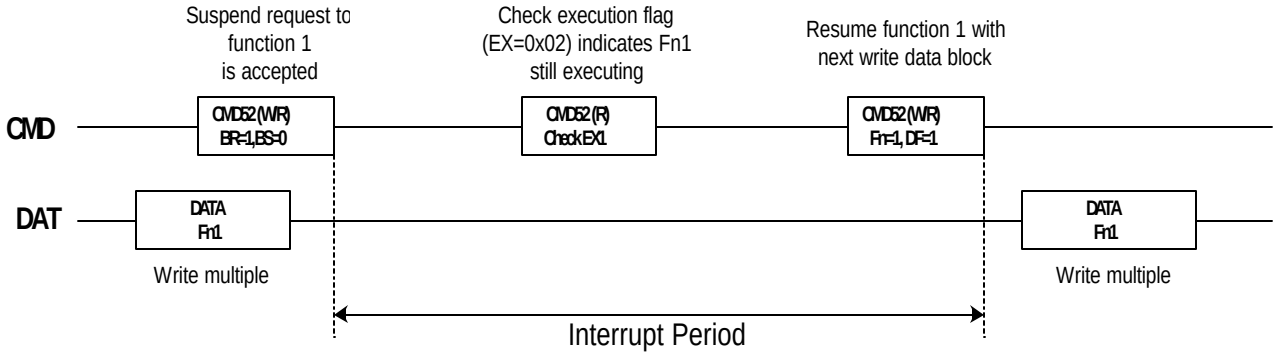


Figure 33 Relationship between interrupt period and suspend/resume (case 1)

Figure 33 shows the relationship between the interrupt period and the suspend/resume operation. In this case, the interrupt period begins after the suspend request is accepted. The interrupt period continues until the resume is executed with data pending (DF=1).

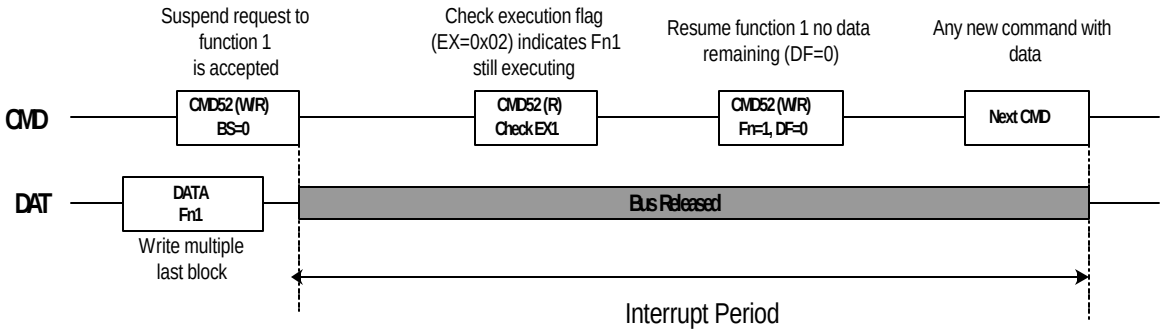


Figure 34 Relationship between Interrupt Period and Suspend/Resume (case 2)

Figure 34 is similar to Figure 33 but in this case, there is no additional data to transfer after function 1 is resumed. In this case, the interrupt period will extend until there is a command that requires data transfer (the next command in this example).

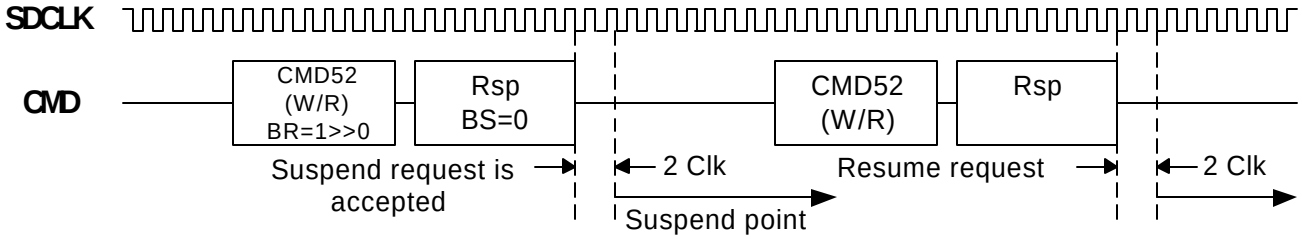


Figure 35 Suspend/Resume timing

Figure 35 Shows the timing of the actual suspend and resume operations. The Suspend point is 2clock after the accepted suspend request response (BS=0). At the suspend point:

- If a function is executing, the function suspends.
- The Interrupt period re-starts
- If the Busy signal (Dat[0]) is asserted (low), it is de-asserted

The resume point is 2 clocks after the response to the resume command. At the resume point:
If the selected function is suspended, the function resumes

- If DF=1, the Interrupt period ends
- If DF=0, the Interrupt period continues
- If the selected function is still busy, the busy signal (Dat[0]) is re-asserted (low).

10 SDIO Read Wait Operation

The optional Read Wait (RW) operation is defined only for the SD 1-bit and 4-bit modes. The read Wait operation allows a host to signal a card that it is doing a read multiple (CMD53) operation to temporarily stall the data transfer while allowing the host to send commands to any function within the SDIO device. To determine if a card supports the Read Wait protocol, the host must test capability bits in CCCR. The timing for Read Wait is based on the Interrupt Period that is defined in section 8.1

If a card does not support the Read Wait protocol, the only means a host has to stall (not abort) data in a read multiple command is to control the SDCLK.

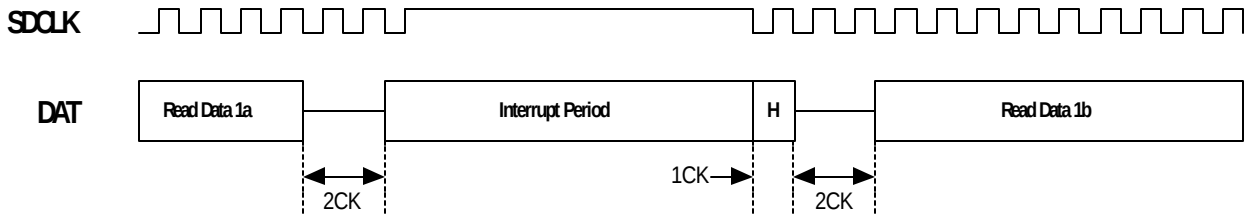


Figure 36 Read wait control by stopping SDCLK

Figure 36 shows the effect of the host stopping the SDCLK in order to stop read data. As usual, the interrupt period begins 2 clocks after the end of data block 1a. In this case, the host wishes to temporarily stop the read data. It does this by stopping the SDCLK after the 2-clock wait period. Because the card has no clock, it will not begin to send the next block of data and the interrupt period is stretched until the clock is resumed. When ready, the host will resume by restarting the SDCLK signal. Since the end of the interrupt period is determined by the SDCLK, it will end as the clock resumes. 2 clocks after the end of the interrupt period, the card will begin to transmit the next block of read data. The limitation of this method is that with the clock stopped, the host cannot issue any commands, and so cannot perform other operations during the delay time.

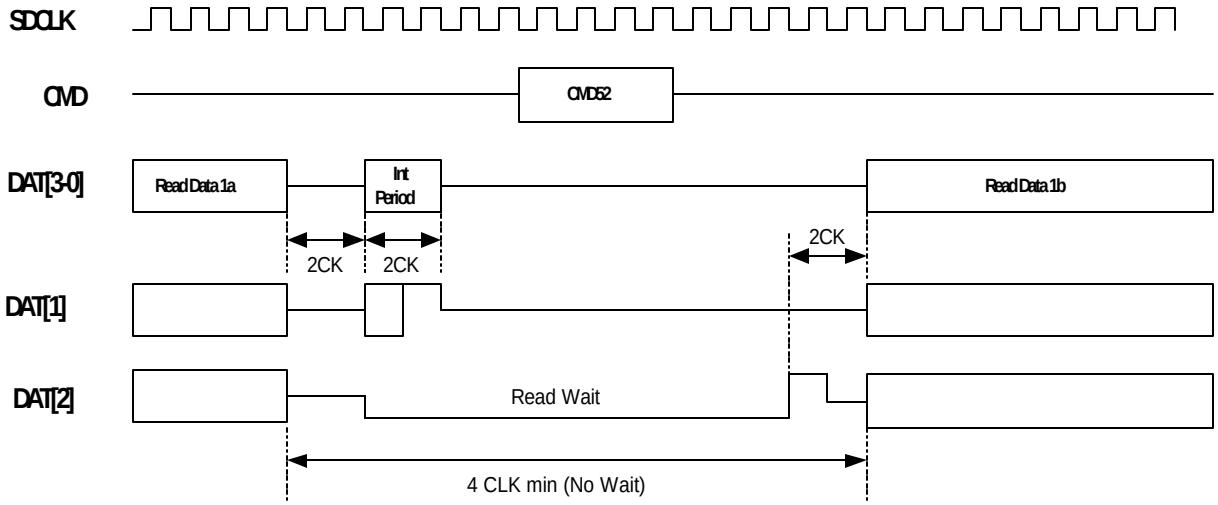


Figure 37 Read wait delay using Dat[2]

Figure 37 shows the Read Wait operation using Dat[2] to request a wait state. During the interrupt period, which starts 2 clocks after the end of the data block, the host can signal the card to enter the read wait state. This is done by asserting Dat[2] low during the interrupt period. The card will detect the low on Dat[2] and will stall the data transfer at that point. The host can maintain the low on Dat[2] for as long as it needs to communicate with other functions. During the Read Wait time, the host can communicate with any function using CMD52. This allows the host to temporarily stall the data transfer from one function or memory and access other functions. In order to restart the stalled transfer, the host must raise Dat[2] to a high for 1 clock and then release it. At that time, the card will restart the stalled transfer. If a card supports the Read Wait protocol, it must allow a minimum of 4 clocks between data blocks rather than the normal 2 in order to allow times for the Read Wait signaling.

11 SDIO Physical Properties

11.1 SDIO Size

The SDIO card is compatible with host sockets designed for SD memory cards. In addition, the SDIO cards can be extended to allow for external connectors, antennas etc. With the exception of the write protect tab, all SDIO cards must meet the mechanical specifications described in the SD Physical spec version 1.01 for that portion of the card that is not extended. The WP tab is optional on SDIO only cards. If it is implemented, it must meet all specifications from the SD Physical Specification. If the SDIO card is a combo card, the WP tab is mandatory.

11.2 SDIO Card Package

An SDIO card shall meet all requirements for Card Packaging identified in section 8.1 of the SD Physical Specification version 1.01 except as noted in Table 15.

SD Physical section	Title	Exceptions for SDIO
8.1.1	External signal contacts (ESC)	NONE
8.1.2	Design and format	32mm size limit does not apply to SDIO. In addition, any dimension limits do not apply in the SDIO extended area. See 11.1 and 12.2
8.1.3	Reliability and durability	Bending and Torque shall be measured at a point 32mm away from contact end irrespective of actual length. WP requirements apply only if WP switch installed (optional on SDIO only cards). Drop test does not apply to SDIO cards.
8.1.4	Electrical Static Discharge (ESD) Requirements	Contact Pads test shall apply only to the 9 SD pins, not any additional vendor specific contacts. Non Contact Pads area for ESD discharge testing shall be in the 24mm by 32mm area of a standard SD card
8.1.5	Quality assurance	Not applicable, SDIO card support of unique ID is optional

Table 15 SDIO exceptions to SD physical section 8.1 requirements

12 SDIO Mechanical Extensions

In order to implement some function in the SD card form factor, some extensions to the standard card size and constructions may be needed. There are two areas of extension defined for SDIO devices. Both of these extensions are optional, and may be used by card vendors based upon their needs. The two extensions are:

1. Additional ESD/EMI ground contact
2. Extended case dimensions

Figure 38 shows the details of both optional extensions.

12.1 Additional ESD/EMI Ground Point

For some SDIO devices, there may be a need for a lower impedance ground connection to the host. This may be needed to reduce the card' s EMI emission or susceptibility. Also, an additional ground may be needed to discharge ESD on insertion or operation. If additional grounding is required, the card may implement an additional ground contact as shown in Figure 38. In this drawing, three suggested designs of the ground strip are shown. The choice of the actual pattern is left to the manufacturer.

12.2 Extended Case

In order to provide useful I/O devices in the SD form factor, it may be necessary to extend the size of the case to provide room for connectors, antennas etc. In order to provide the extension room and maintain compatibility with existing SD hosts, the SDIO extension area has been defined. Figure 38 shows this extension area. The SDIO card vendor may extend the case of its SDIO products within the area identified. Please note that there is no limit in this specification as to the amount or direction of growth in this area. The vendor is cautioned that extensions may cause interference problems with SD hosts, depending on the amount and direction of growth. The drawing shows 2 additional lock notches beyond the normal SD area. These notches are intended to provide additional retention for SDIO devices to prevent accidental withdrawal due to cables or additional weight of extended cases. The support of these notches is optional for both the host and the card vendor, but their use is highly recommended for both to prevent unintended disconnects.

12.3 SDIO Power

12.3.1 Voltage

SDIO cards follow the same voltage and current requirements as SD memory cards. This means that an SDIO or combo card must allow read/write access to the CCCR with an initial voltage range of 2.0 to 3.6V.

12.3.2 Current

Upon initial insertion, SDIO and combo cards shall draw a maximum of 15mA, averaged over a 1 second interval. Note that a memory or combo card built to meet the SD Physical specification version 1.0 may not meet this initial current requirement. Any card built to meet a specification later than 1.01 shall meet this requirement.

Once the card receives the CMD5 or ACMD41 initialization commands, the average current shall be:

- 50 mA or less, averaged over a 400 µS period for SDIO only cards
- 100 mA or less, averaged over a 1 second period for combo cards

This current limit will continue until the card enters the standby state (stby_state).

SDIO cards shall not draw a peak current greater than 200 mA at any time. The current requirements of an SDIO device are read from the CIS area. If the card is a combo card, the current consumption of the card will be the sum of the I/O requirements from the CIS and the memory requirements read from the CSD register. In any case, all SDIO and combo cards shall support a minimum operational range of 3.1 to 3.5V.

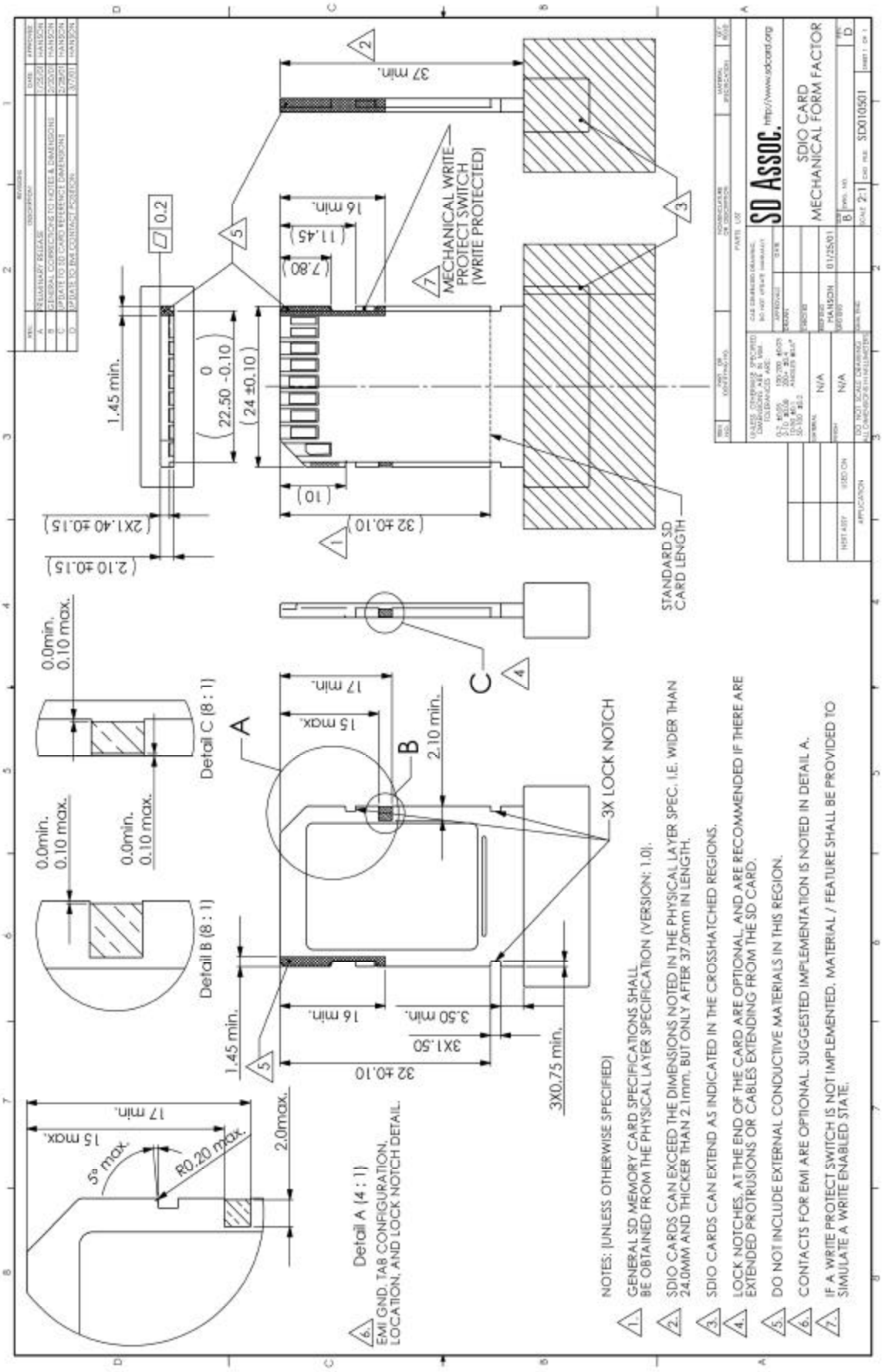


Figure 38 SDIO Mechanical Extensions

13 CIS Formats

13.1 CIS Reference Document

The CIS used by SDIO is based directly upon the metaformat specification used by PCMCIA and Compact Flash. The user of this specification is directed to:

PC Card Standard
Volume 4
Metaformat Specification

Published by:
PCMCIA (Personal Computer Memory Card International Association)
2635 North First Street, Suite 209
San Jose, CA 95134 USA
+1-408-433-2273
+1-408-433-9558 (Fax)

13.2 Basic Tuple Format and Tuple Chain Structure

The Card Information Structure is one or more chains (or linked lists) of data blocks or tuples. The basic format of tuples is shown in Table 16.

Byte	7	6	5	4	3	2	1	0
0x00	TPL_CODE Tuple code: CISTPL_xxx							
0x01	TPL_LINK Offset to next tuple in chain. This is the number of bytes in the tuple body. (n)							
0x02.. (n+2)	The tuple body. (n bytes)							

Table 16 Basic Tuple Format

Byte 0 of each tuple contains a tuple code. A tuple code of 0xFF is a special mark indicating that there are no more tuples in the chain. There are 2 tuples with only a tuple code, the CISTPL_NULL and the CISTPL_END (see Table 17). These tuples do not have any additional bytes. For all other tuples, byte 1 of each tuple contains a link to the next tuple in the chain. If the link field is 0, then the tuple body is empty. If the link field contains 0xFF, then this tuple is the last tuple in its chain. There are two ways of marking the end of a tuple chain for SDIO cards: a tuple code of 0xFF, or a tuple link of 0xFF. The use of an FFH link value is allowed in SDIO cards, but it is recommended to use the End of Chain tuple. System software must use the link field to validate tuples. No SDIO card tuple can be longer than 257 bytes: 1 byte *TPL_CODE* + 1 byte *TPL_LINK* + FFH byte tuple body (and this 257 byte tuple ends the chain). Some tuples provide a termination or stop byte that marks the end of the tuple. In this case, the tuple can effectively be shorter than the value implied by its link field. However, software must not scan beyond the implied length of the tuple, even if a termination byte has not been seen.

13.3 Byte Order Within Tuples

Within tuples, all multi-byte numeric data shall be recorded in little-endian order. That is, the least-significant byte of a data item shall be stored in the first byte of a given field. Within tuples, all character data shall be stored in the natural order. That is, the first character of the field shall be stored in the first byte of the field. Fixed-length character fields shall be padded with null characters, if necessary.

13.4 SDIO Card Metaformat

Unlike the PCMCIA card, the SDIO card has multiple CIS areas. There is a common CIS for the entire card and a CIS assigned to each function. Because of the multiple CIS areas, the SDIO card does not need to support the *CISTPL_LONGLINK_MFC* tuple or the *CISTPL_LINKTARGET* as described in section 2.3.6 of the PCMCIA spec. Table 17 lists the tuple codes supported by SDIO devices. The type field indicates is a tuple is Optional, Mandatory or Recommended. For more details on each tuple, see the PCMCIA metaformat specification section referenced.

Code	Name	Description	PCMCIA Reference	Type
0x00	CISTPL_NULL	Null tuple	3.1.9	O
0x10	CISTPL_CHECKSUM	Checksum control	3.1.1	R
0x15	CISTPL_VERS_1	Level 1 version/product-information	3.2.10	O
0x16	CISTPL_ALTSTR	The Alternate Language String Tuple	3.2.1	O
0x20	CISTPL_MANFID	Manufacturer Identification string Tuple	3.2.9	M
0x21	CISTPL_FUNCID	Function Identification Tuple	3.2.7	M
0x22	CISTPL_FUNCE	Function Extensions	3.2.6	M
0x80-0x8F		Vendor unique tuples	None	O
0xFF	CISTPL_END	The end-of-chain tuple	3.1.2	M

Table 17 Tuples Supported by SDIO Cards

13.5 CISTPL_MANFID: Manufacturer Identification String Tuple

The manufacturer identification tuple contains information about the manufacturer of a SDIO Card. Two types of information are provided: the SDIO Card's manufacturer and a manufacturer card number. Only one manufacturer identification tuple may be present in the card information structure.

Byte	7	6	5	4	3	2	1	0
0x00	TPL_CODE CISTPL_MANFID (20H)							
0x01	TPL_LINK Link to next tuple (at least 4)							
0x02-0x03	TPLMID_MANF SDIO Card manufacturer code							
0x04-0x05	TPLMID_CARD manufacturer information (Part Number and/or Revision)							

Table 18 CISTPL_MANFID: Manufacturer Identification Tuple

The *TPLMID_MANF* field identifies the SDIO Card's manufacturer. New codes are assigned by both PCMCIA and JEIDA. The first 256 identifiers (0x0000 through 0x00FF) are reserved for manufacturers who have JEDEC IDs assigned by JEDEC Publication 106. Manufacturers with JEDEC IDs may use their eight-bit JEDEC manufacturer code as the least significant eight bits of their SDIO Card manufacturer code. In this case, the most significant eight bits shall be zero (0). For example, if a JEDEC manufacturer code is 89H, their SDIO Card manufacturer code is 0x0089. If a SDIO card manufacturer does not currently have a *TPLMID_MANF* assigned, one can be obtained at little or no cost from the PCMCIA. The TPLMID_CARD field is reserved for use by the SDIO Card's manufacturer. It is anticipated that the field will be used to store card identifier and revision information.

13.6 SDIO Specific extensions

SDIO devices use two tuples to provide additional information about the card (common) and each function. The first is the Function ID tuple. The changes for SDIO are detailed in the next sections.

13.6.1 CISTPL_FUNCID: Function Identification Tuple

To identify an SDIO device, the CISTPL_FUNCID tuple must exist in all CIS areas. This means there will be a CISTPL_FUNCID in the common CIS space chain and one in each function' s CIS space chain. The format of this tuple is shown in

Byte	7	6	5	4	3	2	1	0
0x00	TPL_CODE CISTPL_FUNCID (0x21)							
0x01	TPL_LINK Link to next tuple (0x02)							
0x02	TPLFID_FUNCTION Card function code (0x0C)							
0x03	TPLFID_SYSINIT System initialization bit mask. (Not used, set to 0x00)							

Table 19 CISTPL_FUNCID tuple

The function identification tuple contains information about the functionality provided by an SDIO Card. Information is also provided to enable system utilities to decide if the SDIO Card should be configured during system initialization. Since additional function specific information is available, one or more function extension tuples follow this tuple. The TPLFID_FUNCTION field contains an identifier assigned by PCMCIA to identify the SDIO device class

13.6.2 CISTPL_FUNCE: Function Extension Tuple

The CISTPL_FUNCE tuple provides standard information about the card (common) and each individual function. There shall be one CISTPL_FUNCE in each function' s CIS following the TPLFID_FUNCTION tuple. The general format of the CISTPL_FUNCE is shown in Table 20.

Byte	7	6	5	4	3	2	1	0
0x00	TPL_CODE CISTPL_FUNCE (0x22)							
0x01	TPL_LINK Link to next tuple (see following sections)							
0x02	TPLFE_TYPE Type of extended data (see following sections)							
0x03-n	TPLFE_DATA Function information (see following sections)							

Table 20 CISTPL_FUNCE tuple general structure

There are two versions of the TPLFID_FUNCTION tuple, one for the common CIS (function 0) and a version used by the individual function' s CIS (1-7). Both types are described below.

13.6.3 TPLFID_FUNCTION tuple for function 0 (common)

This version of the TPLFID_FUNCTION tuple gives the host common information about the card. There shall be only one of these tuples, located in the CIS for function 0 following the CISTPL_FUNCID. The format of this tuple is shown in Table 21.

Byte	7	6	5	4	3	2	1	0
0x00	TPL_CODE CISTPL_FUNCE (0x22)							
0x01	TPL_LINK Link to next tuple (0x04)							
0x02	TPLFE_TYPE Type of extended data (0x00)							
0x03-0x04	TPLFE_FN0_BLK_SIZE							
0x05	TPLFE_MAX_TRAN_SPEED							

Table 21 TPLFID_FUNCTION tuple for function 0 (common)

The fields in this tuple have the following definition:

Field	Description
TPLFE_FN0_BLK_SIZE	This is the only block size (in bytes) that function 0 can support. A value of 0 indicates block mode is not supported.
TPLFE_MAX_TRAN_SPEED	This byte indicates the maximum transfer rate per one data line during data transfer. This value applies to all functions in the SDIO card. This value shall be 25 Mb/Sec (0x32) for all Full-Speed SDIO devices. The minimum value for Low-Speed SDIO cards shall be 400 Kb/Sec (0x48). The format is identical to the TRAN_SPEED value stored in the CSD of SD memory cards. The maximum data transfer rate is coded according to the following method: **Bits 2:0 contain the transfer rate unit coded as follows:** 0=100kbit/s, 1=1Mbit/s, 2=10Mbit/s, 3=100Mbit/s, 4... 7=reserved **Bits 6:3 contain the time value codes as follows:** 0=reserved, 1=1.0, 2=1.2, 3=1.3, 4=1.5, 5=2.0, 6=2.5, 7=3.0, 8=3.5, 9=4.0, A=4.5, B=5.0, C=5.5, D=6.0, E=7.0, F=8.0 **Bit 7 is reserved and shall be zero**

Table 22 TPLFID_FUNCTION field descriptions for function 0 (common)

13.6.4 TPLFID_FUNCTION tuple for function 1 -7

This version of the TPLFID_FUNCTION tuple gives the host common information about each individual function on a per-function basis. There shall be one of these tuples, located in the CIS for each function following the CISTPL_FUNCID. The format of this tuple is shown in Table 23.

Byte	7	6	5	4	3	2	1	0
0x00	TPL_CODE CISTPL_FUNCE (0x22)							
0x01	TPL_LINK Link to next tuple (0x1C)							
0x02	TPLFE_TYPE Type of extended data (0x01)							
0x03	TPLFE_FUNCTION_INFO							
0x04	TPLFE_STD_IO_REV							
0x05-0x08	TPLFE_CARD_PSN							
0x09-0x0C	TPLFE_CSA_SIZE							
0x0D	TPLFE_CSA_PROPERTY							
0x0E-0x0F	TPLFE_MAX_BLK_SIZE							
0x10-0x13	TPLFE_OCR							
0x14	TPLFE_OP_MIN_PWR							
0x15	TPLFE_OP_AVG_PWR							
0x16	TPLFE_OP_MAX_PWR							
0x17	TPLFE_SB_MIN_PWR							
0x18	TPLFE_SB_AVG_PWR							
0x19	TPLFE_SB_MAX_PWR							
0x1A-0x1B	TPLFE_MIN_BW							
0x1C-0x1D	TPLFE_OPT_BW							

Table 23 TPLFID_FUNCTION tuple for function 1 -7

The fields in this tuple have the following definition:

Field	Description
TPLFE_FUNCTION_INFO	Bit significant information about the Function The bits are defined in Table 25
TPLFE_STD_IO_REV	This 8-bit value contains the revision level of the Standard IO device specification that this function supports. The format is x.y where x is the major revision (4-bits) and y is the minor revision level. For example if the revision is 2.4 the value would be 0x24. If this function does not support an IO standard interface, the value shall be 0x00
TPLFE_CARD_PSN	The Product Serial Number is a 32 bit unsigned binary integer. Support of a serial number is optional, if there is no serial number, this field shall be 0x00000000
TPLFE_CSA_SIZE	Size of the CSA space available for this function in bytes
TPLFE_CSA_PROPERTY	This byte contains flags identifying properties of this function's CSA. The bits are defined in Table 26
TPLFE_MAX_BLK_SIZE	This is the maximum block size that this function can support. A value of 0 indicates block mode is not supported.
TPLFE_OCR	This is the OCR value for this function. The format is identical to the 32-bit OCR format used by SD memory devices. For more details, see section 5.1 of the SD Physical Specification.
TPLFE_OP_MIN_PWR	This is the minimum current, in mA, required by this function when operating. This value is valid for all voltages supported by this function.
TPLFE_OP_AVG_PWR	This is the average current, in mA, required by this function when operating. This value is valid for all voltages supported by this function.
TPLFE_OP_MAX_PWR	This is the maximum (peak) current, in mA, required by this function when operating. This value is valid for all voltages supported by this function.
TPLFE_SB_MIN_PWR	This is the minimum current, in mA, required by this function when in the standby condition. If this function does not support standby, this value will be 0x00. The method to place this function in the standby state and the capabilities it has while in standby are vendor defined. This value is valid for all voltages supported by this function.
TPLFE_SB_AVG_PWR	This is the average current, in mA, required by this function when in the standby condition. If this function does not support standby, this value will be 0x00. The method to place this function in the standby state and the capabilities it has while in standby are vendor defined. This value is valid for all voltages supported by this function.
TPLFE_SB_MAX_PWR	This is the maximum current, in mA, required by this function when in the standby condition. If this function does not support standby, this value will be 0x00. The method to place this function in the standby state and the capabilities it has while in standby are vendor defined. This value is valid for all voltages supported by this function.
TPLFE_MIN_BW	This is the minimum data transfer bandwidth, in KB/sec, needed by this function to successfully operate. If this function has no minimum necessary bandwidth, these bytes shall be 0x0000.
TPLFE_OPT_BW	This is the data transfer bandwidth, in KB/sec, needed by this function to function at an optimum level. If this function has no optimum bandwidth, these bytes shall be 0x0000.

Table 24 TPLFID_FUNCTION field descriptions for functions 1 -7

Bit	7	6	5	4	3	2	1	0
Name	RFU	RFU	RFU	RFU	RFU	RFU	RFU	FN_WUS

Table 25 TPLFE_FUNCTION_INFO definition

The FN_WUS (Wake Up Support) bit signals the function's support of wakeup when the function is placed into a low power state and the SDCLK is stopped. If this bit is set to 1, then this function may be placed into a low power state using some vendor defined method and the SDCLK can be stopped with the power remaining applied to the card.

Bit	7	6	5	4	3	2	1	0
Name	RFU	RFU	RFU	RFU	RFU	RFU	CSA_NF	CSA_WP

Table 26 TPLFE_CSA_PROPERTY definition

The CSA_WP bit is used to write protect the CSA for this function, or identifies it as read-only. If this bit is set to 1, then the CSA of this function is either write protected or read-only. If this bit is clear, then the host may write the CSA. Setting of this bit for R/W devices is handled in a vendor-defined method.

The CSA_NF bit is used to indicate to the host that the CSA area should not be reformatted. If the bit is set to 1, the host may not format the CSA file structure. If this bit is cleared to 0, then the host may reformat the file system of the CSA.

14 Abbreviations and terms

Block	A number of bytes, basic data transfer unit
CCCR	Common Card Control Registers
CD	Connect/Disconnect
CIA	Common Information Area
CID	Card IDentification number register
CIS	Card Information Structure
CLK	Clock signal
CMD	Command line or SD bus command (if extended CMDXX)
combo card	A card that includes both SDIO and SD memory
CRC	Cyclic Redundancy Check
CS	Chip Select
CSA	Code Storage Area
CSD	Card Specific Data register
DAT	Data line
DSR	Driver Stage Register
ECC	Error Correction Code
EMI	Electro-Magnetic Interference
ESC	External Signal Contacts
ESD	Electro-Static Discharge
FAT	File Allocation Table
FBR	Function Basic Registers
FIFO	First In-First Out buffer
Flash	a type of multiple time programmable non volatile memory
Function	An IO device contained within an SDIO card
hi-Z	A three-state driver in the high-impedance state
Interrupt	A signal from the SDIO device to the host signaling the need for attention
Interrupt Period	The times that a card may generate an interrupt signal on the SD bus
Legacy Slot	SD Slot that supports only the SD 1.01 specification
LOW, HIGH	binary interface states with defined assignment to a voltage level
MBIO	Multi-Block IO
MMC	MultiMedia Card
MSB, LSB	the Most Significant Bit or Least Significant Bit
OCR	Operation Conditions Register
PCMCIA	Personal Computer Memory Card International Association
PnP	Plug and Play a means to identify an SDIO device and optionally load applications and/or drivers without user intervention
R/O	Read Only

R/W	Read or Write
RAW	Read After Write
RCA	Relative Card Address register
Resume	Re-starting the temporarily halted data transfer
RFU	Reserved for Future Use. Normally Read-Only and set to 0
ROM	Read Only Memory
RWC	Read Wait Control
SCR	SD Configuration Register
SD	Secure Digital
SDA	SD Association
SDCLK	SD clock signal
SDIO	Secure Digital I/O
SDIO aware	A host designed to support the signals and protocol of SDIO devices
SPI	Serial Peripheral Interface
Stuff bit(s)	Filling bit(s) to ensure fixed length frames for commands and responses.
Suspend	Temporarily halting the transfer of data
TBD	To Be Determined (in the future)
Three-state driver	A driver stage which has three output driver states: HIGH, LOW and high impedance (which means that the interface does not have any influence on the interface level)
Tuple	Data blocks in a linked list or chain format
VDD	+ Power supply
VSS	Power supply ground
W/O	Write Only
WP	Write Protect

 ©Copyright SD Association, 2000,2001

Appendix A
(Normative)

A1 SD and SPI Command List

Table 27 and Table 28 show the commands that are supported by SD memory and SDIO devices in both SPI and SD modes. If a command is not identified as either mandatory or optional, then it is not supported by that device.

Supported Commands	Abbreviation	SDMEM System	SDIO System	Comments
CMD0	GO_IDLE_STATE	Mandatory	Mandatory	Used to change from SD to SPI mode
CMD2	ALL_SEND_CID	Mandatory		CID not supported by SDIO
CMD3	SEND_RELATIVE_ADDR	Mandatory	Mandatory	
CMD4	SET_DSR	Optional		DSR not supported by SDIO
CMD5	IO_SEND_OP_COND		Mandatory	
CMD7	SELECT/DESELECT_CARD	Mandatory	Mandatory	
CMD9	SEND_CSD	Mandatory		CSD not supported by SDIO
CMD10	SEND_CID	Mandatory		CID not supported by SDIO
CMD12	STOP_TRANSMISSION	Mandatory		
CMD13	SEND_STATUS	Mandatory		Card Status includes only SDMEM information
CMD15	GO_INACTIVE_STATE	Mandatory	Mandatory	
CMD16	SET_BLOCKLEN	Mandatory		
CMD17	READ_SINGLE_BLOCK	Mandatory		
CMD18	READ_MULTIPLE_BLOCK	Mandatory		
CMD24	WRITE_BLOCK	Mandatory		
CMD25	WRITE_MULTIPLE_BLOCK	Mandatory		
CMD27	PROGRAM_CSD	Mandatory		CSD not supported by SDIO
CMD28	SET_WRITE_PROT	Optional		
CMD29	CLR_WRITE_PROT	Optional		
CMD30	SEND_WRITE_PROT	Optional		
CMD32	ERASE_WR_BLK_START	Mandatory		
CMD33	ERASE_WR_BLK_END	Mandatory		
CMD38	ERASE	Mandatory		
CMD42	LOCK_UNLOCK	Optional		
CMD52	IO_RW_DIRECT		Mandatory	
CMD53	IO_RW_EXTENDED		Mandatory	Block mode is optional
CMD55	APP_CMD	Mandatory		
CMD56	GEN_CMD	Mandatory		
ACMD6	SET_BUS_WIDTH	Mandatory		
ACMD13	SD_STATUS	Mandatory		
ACMD22	SEND_NUM_WR_BLOCKS	Mandatory		
ACMD23	SET_WR_BLK_ERASE_COUNT	Mandatory		
ACMD41	SD_APP_OP_COND	Mandatory		
ACMD42	SET_CLR_CARD_DETECT	Mandatory		
ACMD51	SEND_SCR	Mandatory		SCR not supported by SDIO

Table 27 SD Mode Command List

Supported Commands	Abbreviation	SDMEM System	SDIO System	Comments
CMD0	GO_IDLE_STATE	Mandatory	Mandatory	Used to change from SD to SPI mode
CMD1	SEND_OP_COND	Mandatory		
CMD5	IO_SEND_OP_COND		Mandatory	
CMD9	SEND_CSD	Mandatory		CSD not supported by SDIO
CMD10	SEND_CID	Mandatory		CID not supported by SDIO
CMD12	STOP_TRANSMISSION	Mandatory		
CMD13	SEND_STATUS	Mandatory		Card Status includes only SDMEM information.
CMD16	SET_BLOCKLEN	Mandatory		
CMD17	READ_SINGLE_BLOCK	Mandatory		.
CMD18	READ_MULTIPLE_BLOCK	Mandatory		
CMD24	WRITE_BLOCK	Mandatory		
CMD25	WRITE_MULTIPLE_BLOCK	Mandatory		
CMD27	PROGRAM_CSD	Mandatory		CSD not supported by SDIO.
CMD28	SET_WRITE_PROT	Optional		
CMD29	CLR_WRITE_PROT	Optional		
CMD30	SEND_WRITE_PROT	Optional		
CMD32	ERASE_WR_BLK_START	Mandatory		
CMD33	ERASE_WR_BLK_END	Mandatory		
CMD38	ERASE	Mandatory		
CMD42	LOCK_UNLOCK	Optional		
CMD52	IO_RW_DIRECT		Mandatory	
CMD53	IO_RW_EXTENDED		Mandatory	Block mode is optional
CMD55	APP_CMD	Mandatory		
CMD56	GEN_CMD	Mandatory		
CMD58	READ_OCR	Mandatory		
CMD59	CRC_ON_OFF	Mandatory	Mandatory	
ACMD13	SD_STATUS	Mandatory		
ACMD22	SEND_NUM_WR_BLOCKS	Mandatory		
ACMD23	SET_WR_BLK_ERASE_COUNT	Mandatory		
ACMD41	SD_APP_OP_COND	Mandatory		
ACMD42	SET_CLR_CARD_DETECT	Mandatory		
ACMD51	SEND_SCR	Mandatory		SCR includes only SD-MEM information.

Table 2-8 SPI Mode Command List

Appendix B
(Normative)

B1. Normative References

The following documents are referenced by this specification. The reader is directed to the respective owners to obtain copies.

1) SD Memory Card Specifications
Part 1 PHYSICAL LAYER SPECIFICATION
Version 1.01 September 2000

2) SD Memory Card Specifications
Part 2 FILE SYSTEM SPECIFICATION
Version 1.0 February 2000

3) ISO/IEC9293:1994
Information technology - Volume and file structure of disk cartridges for information interchange

4) PC CARD STANDARD Release 7 1999
Volume 4 Metaformat Specification

5) The MultiMedia Card System Specification
MMCA Technical Committee Version 2.2

Appendix C
(Informative)

C1 Example SDIO Controller Design

Figure 39 shows an example of an SDIO controller design. In this example, two independent state machines are used. The first is the Bus State machine, which communicates with the host and maintains bus states. Figure 40 shows an example state table for this machine. The second machine is used to communicate and control the function(s) in the card. This machine maintains control of function states such as Exec, Ready and interrupts. An example state table for this machine can be seen in Figure 41.

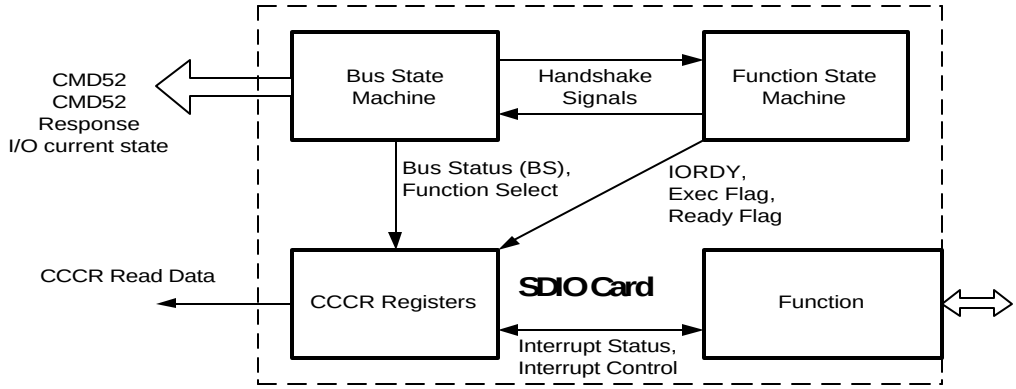


Figure 39 SDIO Internal State Machine example

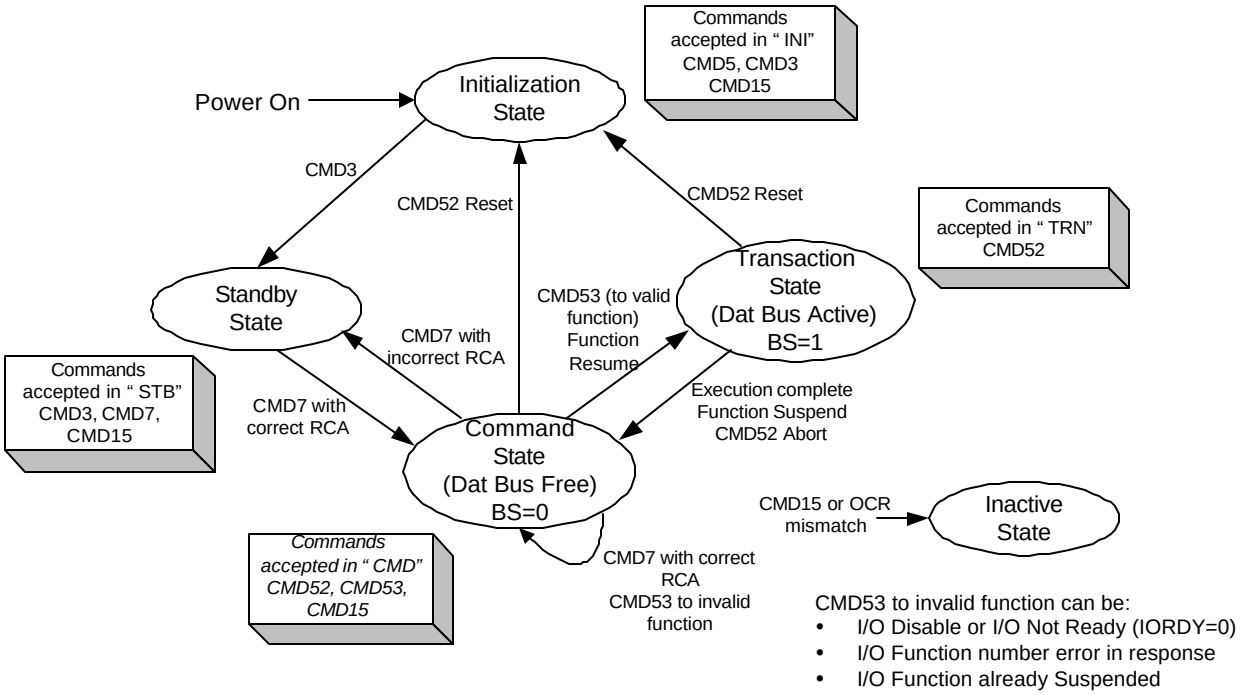


Figure 40 State Diagram for Bus State Machine

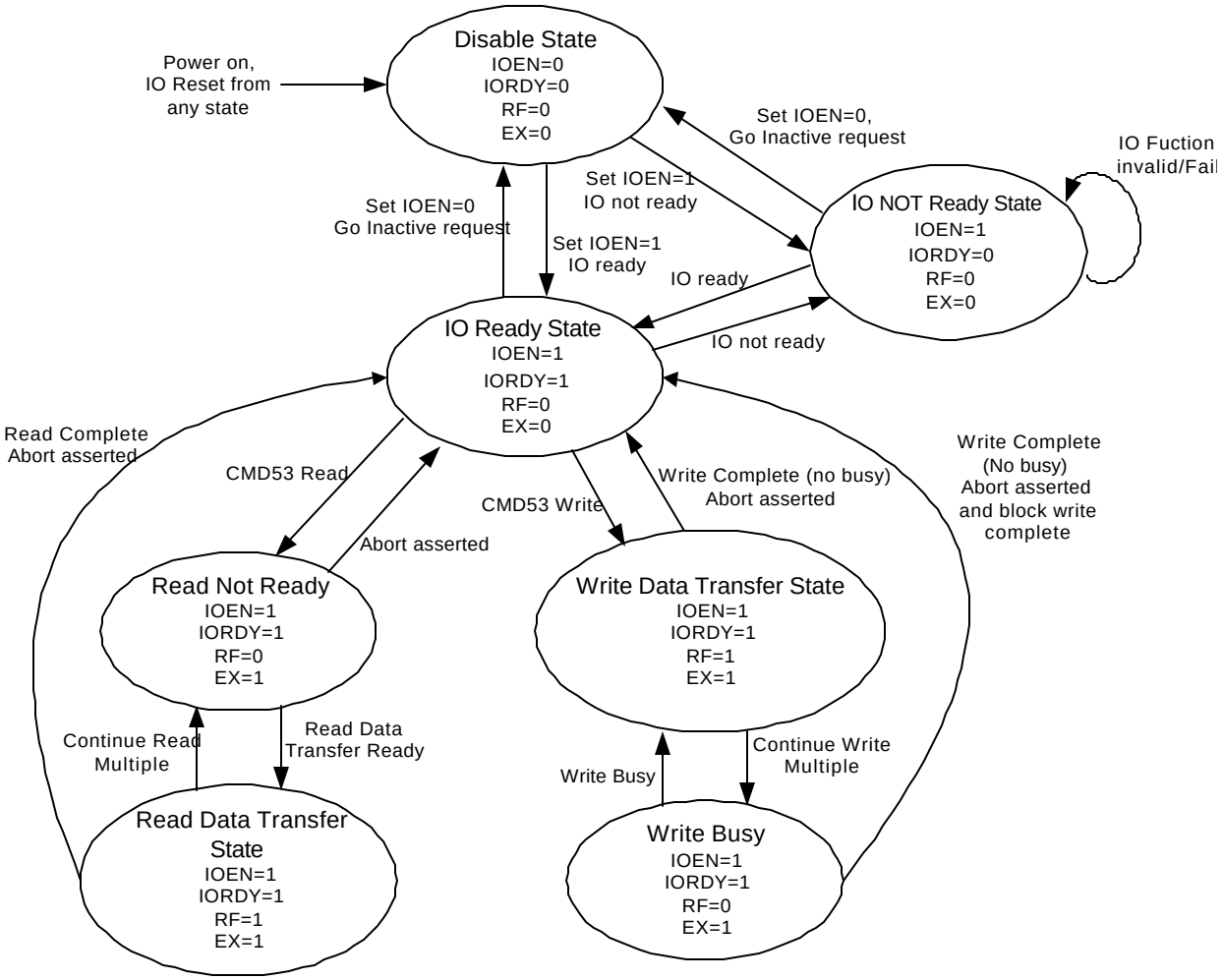


Figure 41 State Diagram for Function State Machine