High-Tech·Made in Switzerland

MCDISK

TECHNICAL REFERENCE MANUAL

GENERAL INFORMATION

The MCDISK (Memory Card DISK) is a card reader/writer device for PCMCIA/JEIDA compatible memory cards. These cards are widely used in mobile computers and are available from many manufacturers in various sizes and types. The PCMCIA/ JEIDA standard provides a means to handle all cards in a compatible way with little or no user intervention as it features automatic identification of the cards through the CIS (Card Information Structure).

The MCDISK contains sophisticated firmware that automatically analyses CIS and handles all card specific parameters and algorithms within the device. This allows the MCDISK to be used as a transparent mass storage unit using standard driver software on the host system. However, for applications where the host system needs precise control over every byte of the card, the MCDISK features a direct access mode.

The MCDISK can handle a wide variety of cards of any size: SRAM, ROM, OTP, FLASH, EEPROM, PC-ATA disks (mechanical/ silicon miniature disk drives) as well as I/O cards for special applications. All mechanical card types (I, II, III and IV/oversized) are supported.

The MCDISK has a standard SCSI interface and can be used like any removable hard disk device. Direct memory access (DMA) between the memory card and the SCSI port makes the data transfer very fast in comparison with other solutions.

TECHNICAL FEATURES

- Available in open-frame version with size and mounting equivalent to standard 3.5" slim-line floppy disk drive (MCDISK-E-1, MCDISK-G-1) or in desktop versions with 2 SCSI-II connectors and built-in power supply (MCDISK-G-5, MCDISK-E-5).
- Interfaces PCMCIA/JEIDA cards to any system with a SCSI interface.
- Handles all card capacities: 512 bytes up to 64 Megabytes for directly addressed memory cards, up to 4 Gigabytes for indexed access such as PC-ATA disk cards.
- Separate SCSI ID can be configured
- Local 68HC11 processor with 48 KByte of firmware.
- SCSI interface with a WD 33C93 SCSI controller.
- Built-in algorithms for transparent access to FLASH, EEPROM, OTP and PC-ATA cards.
- Direct mode allows accessing every single byte in common and attribute memory space as well as in I/O space.
- Standard SCSI commands.
- Low power consumption. (150 mA without card/terminators)
- Powerless card inserting of PCMCIA cards
- Fast 2Mbytes/s transferrate
- Switchable SCSI termination

References :

MCDISK-E-1 Open frame, double slot, +5V/+12V DC supply
MCDISK-E-5 Desktop case, double slot, 80-230VAC
MCDISK-G-1 Open frame, single slot, +5V/+12V DC supply
MCDISK-G-5 Desktop case, single slot, 80-230VAC

TABLE OF CONTENTS

1. INTRODUCTION

1.1 SCOPE OF THIS MANUAL

This technical reference manual contains information required to drive the MCDISK on the SCSI command level. It is intended for people who write drivers or special applications that directly interface the MCDISK.

For information on using the MCDISK with existing driver software, please refer to the driver's documentation.

For information on connecting the MCDISK to a host and general hints about usage, please refer to the appropriate MCDISK User's manuals.

This manual covers the entire line of MCDISK products (open frame and desktop versions). Functional differences between versions will be mentioned in the appropriate sections of this manual.

2. CONNECTORS

This section shows the pin assignments of the SCSI and PCMCIA connectors for technical reference. Please refer to the PCMCIA and SCSI standards documentation for detailed reference.

For information on parts location or physical outline of the connectors, please refer to the appropriate MCDISK user's manual.

2.1 SCSI CONNECTOR

Pin	Signal	Pin	Signal	Mnem
1	GND	2	Data bus bit 0	DB0*
3	GND	4	Data bus bit 1	DB1*
5	GND	6	Data bus bit 2	DB2*
7	GND	8	Data bus bit 3	DB3*
9	GND	10	Data bus bit 4	DB4*
11	GND	12	Data bus bit 5	DB5*
13	GND	14	Data bus bit 6	DB6*
15	GND	16	Data bus bit 7	DB7*
17	GND	18	Data bus Parity	DBP*
19	GND	20	nc	
21	GND	22	nc	
23	GND	24	nc	
25	nc	26	Terminator Power	
27	GND	28	nc	
29	GND	30	nc	
31	GND	32	Attention	ATN*
33	GND	34	nc	
35	GND	36	Busy	BSY*
37	GND	38	Acknowledge	ACK*
39	GND	40	Reset	RST*
41	GND	42	Message	MSG*
43	GND	44	Select	SEL*
45	GND	46	Command/Data	C/D*
47	GND	48	Request	REQ*
49	GND	50	Input/Output	I/O*

Table 2.1: SCSI connector

2.2 PCMCIA CARD CONNECTOR

Pin	Signal	Pin	Signal
1	GND1	2	D3
3	D4	4	D5
5	D6	6	D7
7	CE1*	8	A10
9	OE	10	A11
11	A9	12	A8
13	A13	14	A14
15	WE*	16	RDY
17	$V_{CC}1$	18	$V_{PP}1$
19	A16	20	A15
21	A12	22	A7
23	A6	24	A5
25	A4	26	A3
27	A2	28	A1
29	A0	30	D0
31	D1	32	D2
33	WP	34	GND2
35	GND3	36	CD1*
37	D11	38	D12
39	D13	40	D14
41	D15	42	CE2*
43	RFSH*	44	IORD*
45	IOWR*	46	A17
47	A18	48	A19
49	A20	50	A21
51	$V_{CC}2$	52	$V_{PP}2$
53	A22	54	A23
55	A24	56	A25
57	RFU3	58	RESET
59	WAIT*	60	INPACK*
61	REG*	62	BVD2
63	BVD1	64	D8
65	D9	66	D10
67	CD2*	68	GND

Table 2.2: PCMCIA connector

3. DIP-SWITCH/JUMPER CONFIGURATION

The different MCDISK versions are equipped with 4 or 8 DIP Switches (desktop versions) or Jumpers (open frame versions). For details please refer to the appropriate MCDISK user's manual.
This section summarizes the functions of the individual jumpers/switches for general overview.

Pos	Function
1 2 3	SCSI ID bit 0 (Switch ON/Jumper set=1) SCSI ID bit 1 (Switch ON/Jumper set=1) SCSI ID bit 2 (Switch ON/Jumper set=1)
4	Parity (Switch on/Jumper set=parity enabled)
5	SCSI Termination (Switch on=Termination on)
6	Dual SCSI ID mode (Switch on/Jumper set=Dual SCSI mode on)
7 8	CIS mode bit 0 (Switch ON/Jumper set=0) CIS mode bit 1 (Switch ON/Jumper set=0)

Note: Switch/Jumper positions 7 and 8 are used to set the default CIS mode. The CIS mode can be changed through MODE SELECT PAGE $32 (format parameters, see paragraph 6.6.14.7). For MCDISK versions that have no switch position 7 and 8, the default CIS mode is set to 11_2(=3), which is the normal case for most applications.
With an MCDISK with switches/jumpers 7 and 8 installed, ensure they are left OFF/open unless you have a very special application that needs a different CIS mode. Refer to the description of MODE SELECT PAGE $32 for a description of the 4 possible CIS modes.

Note: Some MCDISK models use a numeric thumbwheel switch instead of three single switches in position 1..3. Please refer to the appropriate User's Manual for details.

Note: Dual SCSI ID mode is implemented in Firmware Versions 3.3 and higher only (except for MCDISK-G which is a single slot unit). Leave this jumper open for units with firmware prior to V3.3!

Note: In some MCDISK models, jumpers 6 to 8 are not accessible from the outside. Refer to the appropriate User Manual for instructions.

4. INDICATORS

The MCDISK is equipped with two or four indicator LEDs:

One green LED (power LED) is lit whenever the MCDISK is powered up except in MCDISK-G where no power LED is available.
One red LED (operation status LED) is lit when the unit is not operable. This can be when no card is inserted correctly or when the card inserted could not be accessed properly for some reason (such as unknown card type, write attempt to read-only or write-protected card etc.). If the red LED is lit due to an access error with a card inserted, it will turn off again when a new command is executed successfully.
One yellow LED (access LED) for each slot is lit whenever a memory card is accessed and therefore the card should never be removed from the MCDISK while this LED is lit. It is possible to keep the access LED constantly lit using the PREVENT / ALLOW REMOVAL command, which is useful to prevent card extraction while a memory card is mounted as disk device by an operating system. Note that when a physical access takes place in one slot in double-slot MCDISKs, both access LEDs will be lit. This is because the two slots share some signal lines, and NEITHER card might be extracted or inserted when an access takes place.

During start-up (power up or SCSI bus reset), the MCDISK performs some internal self tests. When all internal testing is completed successfully, the yellow and red LED(s) will flash twice.
In case of an error, the LEDs will show a diagnostic status:

- Both yellow and red LED remain dark or the red LED is lit alone (without flashing first): Make sure that the unit is correctly connected to a powered-up SCSI bus. The unit cannot start up when the SCSI bus is not terminated. If the SCSI bus is OK, terminated and powered up and the indicators still remain dark, there may be a hardware problem, and the unit needs to be returned for repair.
- Both yellow and red LED flash only once and then remain dark: The internal RAM test has failed. The unit needs to be returned for repair.
- The red LED flashes repeatedly: There is a SCSI hardware problem.
- The red LED double-flashes repeatedly: This indicates a severe SCSI bus protocol error that prevents the MCDISK from continuing SCSI operation. The MCDISK needs to be reset to recover from this error. If the error is permanent, check SCSI cabling, terminators and SCSI ID's.
- The red LED triple-flashes repeatedly: The ROM contents are not OK. The unit needs to be returned for repair.
- The LEDs show some other pattern than those listed above: There is a severe hardware problem, and the unit needs to be returned to your local MCDISK supplier for repair.

5. THEORY OF OPERATION

5.1 GENERAL OVERVIEW

The MCDISK is a device that reads and writes PCMCIA/ JEIDA memory cards. To interface those cards to common equipment the MCDISK provides a SCSI interface. The SCSI interface is widely used for connecting mass storage devices like hard disks, CD-ROMs and WORMs to microcomputers.

For an application, where some type of memory card is used as mass storage for a computer system, the MCDISK provides *transparent* access to the card. This means that the SCSI host machine may access the card as if it was a removable hard disk device. In this *transparent* mode of operation, the MCDISK handles all issues that are specific to PCMCIA/ JEIDA memory cards (such as the Card Information Structure, CIS) internally and invisible to the SCSI host system. This is possible as most modern PCMCIA/JEIDA memory cards are self-identifying, i.e. they contain information about their type, size, data organisation etc. on the card, so there is no need for the host system to provide these parameters. But for the older PCMCIA/JEIDA memory cards, that cannot yet identify themselves, the MCDISK provides a means to set those parameters when the card is formatted. During the format process, the MCDISK creates a CIS itself and records it on the card. This makes the card self-identifying for future accesses. Therefore, when working with already formatted cards, the MCDISK works exactly like a SCSI hard disk, when older cards must be formatted, a few extra parameters must be provided by the formatting program.

In transparent mode, ROM and non writeable memory cards (it depends on the version of the MCDISK which types of memory are writeable) look like a write protected removable hard disk devices to a SCSI host, while SRAM, EEPROM and PC-ATA cards behave like fully writeable removable hard disks. OTP and FLASH memories are presented as WORM devices to the SCSI host. This is really true in case of OTP (writeable only once), and conceptionally correct in the case of FLASH (which is erasable in huge blocks, but between erase operations, each byte can be written only once).

In terms of the PCMCIA standard, the transparent mode of the MCDISK can be thought of as a built-in, but invisible version of Socket Services and Card Services.

There is a different type of application for the MCDISK, where the transparent access mode is not suitable: When the physical accesses done to the memory card need to be tightly controlled by the application. This is the case with special I/O or controller cards that perform user specific functions. In transparent mode, the MCDISK makes a lot of extra accesses to the card to analyse the CIS, find out the card size etc. These accesses might interfere with the user's intention. Therefore the MCDISK provides a *direct* mode of operation. In direct mode, only those accesses specified by the user through SCSI READ and WRITE commands are performed. Every byte of the entire PCMCIA 64M common memory and 32M attribute memory address ranges can be accessed without restriction in memory mode (OE and WE signals) or in I/O mode (IORD and IOWR signals).

5.2 ADRESSING MEMORY CARDS VIA SCSI LUNS

The MCDISK uses different SCSI LUNs (logical unit numbers) to address memory cards (or partitions thereof, see paragraph 5.5) in either *transparent* or *direct* access mode as follows:

LUNs 0 to 3 are used to access memory cards (or partitions thereof, see below) in *transparent mode*. The most common case is using LUN 0, which will give *transparent* access to the first partition of the memory card inserted. For MCDISK versions with two PCMCIA slots, LUN 0 accesses the *primary slot (see 5.3)*, while LUN 3 is used to access the first partition of the memory card inserted in the *secondary slot*. LUN 1 and 2 are dynamically assigned to additional partitions of the *primary* or *secondary slot* (see below).

LUNs 4 up to 7 are reserved for *direct mode* as follows:

- LUN 7 is used to access the common memory space of the *primary slot*.
- LUN 6 is used to access the attribute space of the *primary slot*.
- LUN 5 is used to access the common memory space of the *secondary slot* (for double-slot versions only)
- LUN 4 is used to access the attribute space of the *secondary slot* (for double-slot versions only)

Note: In older versions of the MCDISK firmware, LUN 4 and 5 were used for transparent mode, too.

5.3 ADRESSING SLOTS (DUAL-SLOT VERSIONS ONLY)

For MCDISK units with two PCMCIA slots there are two methods of accessing the upper slot:

Using a single SCSI ID for both slots. This is the default mode (and the only mode for units with Firmware prior to V3.3):

- The lower slot is called *primary slot* and can be accessed trough the respective LUNs as described in paragraph 5.2.
- The upper slot is called *secondary slot* and can be accessed trough the respective LUNs as described in paragraph 5.2.

Using a separate SCSI ID for each slot. This *Dual SCSI ID* feature is available only in firmware versions V3.3 and newer. To enable this feature, the Jumper position 6 (see paragraph 3) must be set:

- For the SCSI ID configured with jumpers 1-3 (see paragraph 3), the *primary slot* is the lower slot, and the *secondary slot* is the upper slot.
- With the *Dual SCSI ID* feature enabled, the MCDISK is accessible under a second SCSI ID, which is the configured SCSI ID + 1. For this second SCSI ID, the *primary slot* is the upper slot, and the *secondary slot* is the lower slot. Note that there is no second SCSI ID if the configured SCSI ID is set to 6 (as SCSI ID 7 is reserved for the host system and cannot be used by the MCDISK). If SCSI ID 6 is configured, the MCDISK behaves as if *Dual SCSI ID* mode was disabled).

This *Dual SCSI ID* feature allows MCDISK units with two PCMCIA slots to appear as two SCSI devices on the SCSI bus. This is useful when standard drivers are used that are not capable of addressing the upper slot through logical unit (LUN) number 3 instead of 0.

Note: The *Dual SCSI ID* feature works only if SCSI timeout are strictly held as defined in the SCSI-II standard. Some host adaptors have features like “Fast SCSI scan” that violate the SCSI timeout periods, because this is not relevant for many SCSI devices (including the MCDISK with *Dual SCSI ID* feature disabled). If you have SCSI problems, try disabling those “Fast SCSI scan” features.

The following table summarizes the addressing capabilities through LUNs and SCSI IDs of the MCDISK:

LUN	Configured SCSI ID	Second SCSI ID (=Config. ID+1) only if *Dual SCSI ID* feature is enabled and configured ID is < 6
0	Lower Slot transparent access	Upper Slot transparent access
1	Lower Slot 2nd device	Upper Slot 2nd device
2	Upper Slot 2nd device	Lower Slot 2nd device
3	Upper Slot transparent access	Lower Slot transparent access
4	Upper Slot attribute space direct access	Lower Slot attribute space direct access
5	Upper Slot common space direct access	Lower Slot common space direct access
6	Lower Slot attribute space direct access	Upper Slot attribute space direct access
7	Lower Slot common space direct access	Upper Slot common space direct access

Table 5.3:addressing capabilities through LUNs and SCSI IDs

5.4 BLOCKS

Most mass storage devices cannot be accessed on a byte per byte basis, as they are internally organized in sectors (*physical blocks*) consisting of several hundred bytes (most common are 512 bytes) which must be read or written as a whole. This is true for some PCMCIA memory cards, too (e.g. PC-ATA disks), but others (e.g. SRAM) allow accessing every byte independently.

Aside from this *physical block size* which is dependent on the card being used, there is the SCSI *logical block size*. A logical block is the smallest amount of data (most common is 512 bytes, too, for compatibility with most disk device's *physical block size*) that can be transferred with one SCSI READ or WRITE command. Although those two block sizes are normally set equal, the MCDISK offers additional flexibility and allows the *logical block size* to be set to any value suitable for an application in certain cases.

There are basically four cases which must be differentiated with regard to the *logical block size*.

- SRAM, ROM, EEPROM, FLASH and OTP cards without any PCMCIA-defined block checking enabled. This is the most common case for cards with capacities between a few Kilobytes and 4 Megabytes. As these cards are adressable on the byte level, the MCDISK allows setting any logical block size between 1 and 65535. Unlike most SCSI disks, all values are allowed and not only powers of 2.

- SRAM, ROM, EEPROM, FLASH and OTP cards with PCMCIA-defined checksums or CRC for block check. These cards, although physically accessible on a byte basis, are organized in a block structure (most common block size is also 512 bytes). Even for these cards, the MCDISK allows setting the logical block size independently in the range 1..65535. The MCDISK handles overlapping and partial access of the checksummed blocks automatically.

- PC-ATA disks (real mechanical disk cards or silicon disks). These cards can only be accessed in 512 byte blocks. The MCDISK allows setting the logical block size to any integer multiple of 512, but not to fractions of 512.

- When accessing the MCDISK in *direct mode* (through LUN 4..7), the logical block size can be set to any value between 1 and 65535.

By default, the MCDISK sets the logical block size to 512 bytes. It can be changed through the MODE SELECT command (see below).

5.5 PARTITIONS/MULTIPLE DEVICES

There are memory cards that have multiple types of memory, e.g. a PC-ATA disk with 256K of SRAM or a mixed RAM/ROM card (also sometimes called a “hybrid” card). In addition, the PCMCIA/JEIDA standard allows the concept of multiple partitions. Partitions are logical parts of the memory card, which are treated like separate devices. To handle such multi-device cards, the MCDISK automatically assigns the second and subsequent partitions to new LUNs as follows: The second partition of the card in the lower slot is assigned to LUN 1, the third to LUN 2. The second partition of the card in the upper slot is assigned to LUN 2, the third to LUN 1. Obviously, LUN 1..2 are shared between the lower and upper slot dynamically. To find out to which slot a device belongs, MODE SENSE PAGE $30 can be sensed (S1 bit set means device belongs to the card in the upper slot).

5.6 CIS HANDLING

In *transparent mode*, the MCDISK tries to identify the card through its CIS if available. If the card has no CIS, the MCDISK tries to identify the card through JEDEC ID or test procedures. When a card is formatted, the MCDISK creates new CIS for cards that did not have CIS before. The algorithm is designed such that, if possible, a card once formatted should be self-identifying from then on.

5.6.1 CIS MODES

Normally, the automatic procedures the MCDISK performs to create and analyse CIS work fine for all types of card without user intervention. However, for some special cases, it might be required to restrict the CIS handling in some way. The MCDISK differentiates 4 CIS modes:

- 3: Full CIS Mode: The MCDISK perform all of its CIS operations to detect and format cards automatically.

- 2: No CIS Level 1 writing into common memory space: The MCDISK does not try to write CIS Level 1, which contains information about the physical card specifications, into the common memory space. However, if the card has an attribute space, CIS Level 1 writing will take place.

- 1: No CIS formatting: No CIS Level 1 and no other CIS information will be written when the card is formatted.

- 0: No CIS mode: The MCDISK does not analyze the CIS and does not try to write CIS when a card is formatted.

Note:

Full CIS mode is the default and should be used whenever possible. Other CIS modes can be set through MODE SELECT PAGE $30 (see chapter 6.6.4) and through jumpers in some MCDISK units (see chapter 3.).

5.6.2 LEVEL 1 CIS (PHYSICAL CHARACTERISTICS)

For the MCDISK, this part of the CIS consists of the following tuples: CISTPL_DEV, CISTPL_DEVA (device type, speed and size), CISTPL_JEDEC, CISTPL_JEDECA (JEDEC ID of devices, used to determine write algorithm etc. for special chips) CISTPL_FUNC, CISTPL_FUNCE (function identification such as PC-ATA) and CISTPL_CONF (configuration information for I/O cards and sleep mode).
CIS Level 1 should be contained in a special attribute memory of the card (most probably a small ROM programmed by the card's manufacturer). Unfortunately, a number of memory cards on the market today don't support Level 1 CIS yet, some do but don't comply with the standard. Therefore, the MCDISK provides a means to recognize and handle cards with and without Level 1 CIS stored at different locations. There are the following variants:

- Cards of any type with correct attribute information supplied by the card manufacturer or written during card formatting (e.g. by the MCDISK or MCRW, see below): This is the best case, because the MCDISK will automatically recognize all relevant parameters of the card inserted, and can handle it correctly without further mode setup. This is the case for most modern FLASH and PC-ATA cards.

- Cards without CIS level 1 information and with write protection disabled: When the MCDISK does not find level 1 CIS, it performs JEDEC ID read attempts and / or RAM test procedures to find out size and type of the card. Again, no mode setup is required. When formatting such a card in full CIS mode, the MCDISK will create Level 1 CIS automatically. Thus, the card will be subsequently recognized as a card with Level 1 CIS (type and size known) by all card readers that comply with the PCMCIA/JEIDA standard.

- Write protected SRAM cards as well as all other card types without CIS Level 1 information cannot be recognized automatically by the MCDISK. These cards will be treated like 32 MByte ROM cards (not writeable). In order to make, e.g. a FLASH-EPROM card, writeable, the card type and size must be set up using the MODE SELECT PAGE $30 (type, size) and PAGE $36 (JEDEC ID, speed) before formatting. When formatting in full CIS mode, the MCDISK will create appropriate Level 1 CIS so that the card will be correctly recognized in subsequent accesses.

- Cards that have valid CIS information, but are not supported by the MCDISK's transparent mode, such as modem or other I/O peripheral cards. Accessing these cards in transparent mode will return the "unsupported feature" controller error. However, it is possible to access these cards in direct mode with appropriate driver software on the SCSI host computer.

- Cards that have invalid or incomplete CIS. These cards, when accessed in transparent mode, will cause a "bad card" controller error.

Notes:

- The PCMCIA/JEIDA standard provides storage of Level 1 CIS either in a separate memory area called attribute space or in the main memory of the card (common space). On formatting, the MCDISK tries to write Level 1 CIS into attribute space. If the card has no separate attribute space, the Level 1 CIS information will be written to the beginning of the card's common space (full CIS mode is enabled) This is fully compliant with the PCMCIA/JEIDA standard.

- Logically unformatted cards (that is, when the entire common space is directly available as data space) can only support Level 1 CIS if there is a separate attribute memory to store the CIS.

5.6.3 LEVEL 2 CIS (LOGICAL FORMATTING)

In addition to the basic physical characteristics of a memory card, the PCMCIA/JEIDA standard provides detailed information about the data organisation. The MCDISK supports the following Level 2 CIS features:

- Device geometry (CISTPL_DEVICEGEO). This tuple is required for FLASH memories and specifies erase/write parameters such as the erase block size.

- Format parameters (CISTPL_L2INFO, CISTPL_FORMAT and CISTPL_GEO). These affect the way data is organized on the card. By default, the MCDISK formats cards in a disk-like manner with a physical block size of 512 bytes, no error detection, no card test or fill on format. The format parameters can be set (depends on card type) and queried through MODE SELECT PAGEs $03,$04, $05 and $32.

- Format program vendor name and information strings. The MCDISK uses "MPL AG" and "MCDISK/MCRW" by default, but any other string (up to 24 characters) may be set. MODESELECT PAGE $31 is used to set/query these strings.

- Initialisation date. This date can be set or queried through MODE SELECT PAGE $33.

- Data organisation code and name. The MCDISK uses an organisation code of $00 (File system) and no organisation name by default. This information can be set or queried using MODE SELECT PAGE $34.

- Card information string (CISTPL_VERS_1). MODE SENSE PAGE $38 is used to query these strings. They cannot be modified.

- MCRW compatibility. The MCDISK is fully compatible with the MCRW (which is similar to the MCDISK except that it uses a serial interface instead of SCSI). Further, the MCRW's interface definition is upwards compatible with the older MCRW device for 34 pin Panasonic memory cards, offering special features like a variable block length text mode or time stamps on recording. Some information related to these special features is stored as vendor-specific PCMCIA tuple in the card's CIS. Although the MCDISK does not support these features directly, the information can be set (for formatting) or queried (to analyse cards recorded on a MCRW) through MODE SELECT PAGE $35.

5.7 GENERAL NOTES

- The MCDISK recognizes multiple partitions on a single device, but does not support creation of more than a single partition per device on formatting. However, using direct access mode (LUN 4..7) it is possible to create an appropriate CIS for partitioning directly.
- Be aware that the LUNs in the MCDISK do not represent entirely independent “units”, but only subdivisions of a single device, the memory card. So, most MODE SELECT commands affect multiple LUNs at once, as does formatting.

Tuple Name	ID	Description
CISTPL_NOP	$00	Null tuple (no function)
CISTPL_DEV	$01	Device information tuple (common space)
CISTPL_LINKA	$11	Long-link to attribute space
CISTPL_LINK	$12	Long-link to common space
CISTPL_LINKTARGET	$13	Link-target control tuple
CISTPL_NOLINK	$14	No-link control tuple
CISTPL_VERS_1	$15	Manufacturer and product information strings
CISTPL_DEVA	$17	Device information tuple (attribute space)
CISTPL_JEDEC	$18	JEDEC identifiers of common space devices
CISTPL_JEDECA	$19	JEDEC identifiers of attribute space devices
CISTPL_CONF	$1A	Configuration tuple
CISTPL_CE	$1B	Configuration entry tuple
CISTPL_DEVICEGEO	$1E	Device geometry (FLASH erase block size etc.)
CISTPL_FUNCID	$21	Function identification
CISTPL_FUNCE	$22	Function extension
CISTPL_L2INFO	$40	Level 2 information tuple
CISTPL_FORMAT	$41	Format tuple
CISTPL_GEO	$42	Geometry tuple
CISTPL_INIDATE	$44	Card initialisation date tuple
CISTPL_DATAORG	$46	Data organisation tuple
CISTPL_MPL	$80	MCRW compatibility tuple (MPL specific)
CISTPL_END	$FF	End-of-list tuple

Table 5.7, PCMCIA tuples recognized by MCDISK

6. SCSI PROGRAMMING GUIDELINES

This section explains the functional operation of the SCSI commands implemented on the MCDISK memory card reader/writer.

Phase	BSY	SEL	C/D, I/O MSG, REQ	ACK/ATN	Data Bus DB7..0,DBP
BUS FREE	Inactive	Inactive	Inactive	Inactive	Inactive
ARBITRATION	All arbitrating units	Winner of arbitration	Inactive	Inactive	Inactive
SELECTION	Initiator & Target	Initiator	Inactive	Initiator	Initiator
RESELECTION	Initiator & Target	Target	Target	Initiator	Target
COMMAND	Target	Inactive	Target	Initiator	Initiator
DATA IN	Target	Inactive	Target	Initator	Target
DATA OUT	Target	Inactive	Target	Initiator	Initator
STATUS	Target	Inactive	Target	Initiator	Target
MESSAGE IN	Target	Inactive	Target	Initiator	Target
MESSAGE OUT	Target	Inactive	Target	Initiator	Initiator

Table 6: SCSI Bus Phases

6.1 SCSI BUS PHASES

The SCSI bus protocol distinguishes eight phases. Table 5 list the possible bus phases and the corresponding SCSI bus signals.

6.1.1 BUS FREE PHASE

The BUS FREE phase is used to indicate that no device is actively using the SCSI bus and that it is available for subsequent users.

6.1.2 ARBITRATION PHASE

The ARBITRATION phase allows one SCSI device to gain control of the SCSI bus so it can assume the role of an initiator or target. ARBITRATION is optional in systems with a single initiator and no RESELECTION.

6.1.3 SELECTION PHASE

The SELECTION phase allows an initiator to select a target for sending commands to it. In single-initiator systems that do not arbitrate for the bus, the initiator may enter the SELECTION phase following a BUS FREE phase. In arbitrating systems, the device that wins ARBITRATION may enter the SELECTION phase subsequently.

6.1.4 RESELECTION PHASE

The RESELECTION phase allows a target to reconnect to an initiator after having disconnected from the bus while doing some time consuming internal operations (mechanical head movements).

6.1.5 INFORMATION TRANSFER PHASES

The COMMAND, STATUS, DATA IN, DATA OUT, MESSAGE IN and MESSAGE OUT phases are called information transfer phases. When the SELECTION phase has successfully completed, the selected target will enter the COMMAND phase to receive a command from the initiator. Depending on the command issued, the target enters other information transfer phases. The initiator can request a MESSAGE OUT phase by asserting ATN, but the target keeps control of the bus phases until it disconnects itself from the bus and returns to the BUS FREE phase.

6.1.6 SCSI BUS CONDITIONS

The SCSI Bus has two asynchronous conditions: the ATTENTION condition and the RESET condition.

6.1.7 ATTENTION CONDITION

The ATTENTION condition allows an initiator to inform a target that the initiator has a message ready. The target may get this message at its convenience by performing a MESSAGE OUT phase. The initiator can generate the ATTENTION condition by asserting the ATN signal.

6.1.8 RESET CONDITION

The RESET condition is used to disconnect all SCSI devices from the bus immediately. An SCSI device can generate the RESET condition by asserting the RST signal. The MCDISK will never generate a RESET condition, but it will respond to a RESET condition that is present on the bus.

6.2 MESSAGES

The MCDISK requires any SCSI device it communicates with to be able to receive and understand the COMMAND COMPLETE message. The MCDISK will not send any other messages during normal command processing. However, it will respond with messages when explicitly told so by the initiator (linked commands or messages sent by the initiator). Table 6.2 summarizes the messages understood and/or sent by the MCDISK.

Code	Description	Out (receieved)	In (sent)
$00	Command Complete		•
$05	Initiator Detected Error	•	
$06	Abort	•	
$07	Message Reject	•	•
$08	No Operation	•	
$0A	Linked Command Complete		•
$0B	Linked Command Complete (with Flag)		•
$0C	Bus device reset	•	
$8x	Identify	•	

Table 6.2: Messages

6.2.1 COMMAND COMPLETE ($00)

This message is sent by the MCDISK to indicate that the execution of a command (or a series of linked commands) has terminated and that valid status has been sent to the initiator. After sending this message, the MCDISK will disconnect from the bus by going to the BUS FREE phase.

6.2.2 INITIATOR DETECTED ERROR ($05)

When the MCDISK receives the INITIATOR DETECTED ERROR message during a transfer, it will abort the transfer with CHECK CONDITION.

6.2.3 ABORT ($06)

Upon reception of the ABORT message, the MCDISK will immediately abort the current command and go to the BUS FREE phase. The status for the initiator that sent the ABORT message will be cleared.

6.2.4 MESSAGE REJECT ($07)

When this message is sent to the MCDISK, it will simply be ignored. The MCDISK will itself send this message to the initiator whenever it receives an unknown message (i.e. one that is not listed in the “Out” column in table 6.2).

6.2.5 NO OPERATION ($08)

The MCDISK simply accepts this message when received from an initiator.

6.2.6 LINKED COMMAND COMPLETE ($0A)

The MCDISK sends this message to an initiator when a command with the LINK bit set has completed and the status has been sent to the initiator.

6.2.7 LINKED COMMAND COMPLETE (WITH FLAG) ($0B)

This message is like LINKED COMMAND COMPLETE, but will be sent only for those linked commands that have the FLAG bit set (in addition to the LINK bit).

6.2.8 BUS DEVICE RESET ($0C)

When the MCDISK receives this message, it will abort the current command, disconnect from the bus and perform an internal reset (clearing all statuses and reservations for all initiators).

6.2.9 IDENTIFY ($8X)

The IDENTIFY message is sent by the initiator after selection if the initiator supports disconnect/reconnect.

6.3 SCSI COMMAND DESCRIPTOR BLOCK

This section defines the SCSI commands supported by the MCDISK memory card reader/writer.

After being selected by an initiator, the MCDISK goes to the COMMAND phase to receive a series of command bytes called a Command Descriptor Block (CDB) from the initiator. The first byte of the CDB is the opcode of the command and determines the size of the CDB. The MCDISK implements commands with a CDB with 6 or 10 bytes (see Table 6.3.A and 6.3.B). After receiving the complete CDB, the MCDISK may, depending on the command, request data transfers by going to the DATA IN or DATA OUT phases. Upon completion of the command the MCDISK will send a status byte to the initiator, indicating success or failure of the command execution. In case of an error, the MCDISK has additional status information available which can be retrieved by the initiator through the REQUEST SENSE command.

After sending the status, the MCDISK will send either a COMMAND COMPLETE message and go to the BUS FREE phase (LINK bit not set) or it will send a LINKED COMMAND COMPLETE (with or without FLAG) message and go directly to the COMMAND phase (LINK bit set).

The following tables shows the 6 Byte and 10 Byte versions of the command descriptor block. Bits and Bytes marked as “reserved” are used for future expansion of the SCSI standard or the MCDISK command set and must be set to 0.

Bit: Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			Logical Block Address, MSB (if required)				
2	Logical Block Address (if required)							
3	Logical Block Address, LSB (if required)							
4	Transfer Length (if required)							
5	Reserved						Flag	Link

Table 6.3.A: 6 Byte Command Descriptor Block

Bit: Byte	7	6	5	4	3	2	1	0
0	Operation Code							
1	Logical Unit Number			reserved				
2	Logical Block Address, MSB (if required)							
3	Logical Block Address (if required)							
4	Logical Block Address (if required)							
5	Logical Block Address, LSB (if required)							
6	Reserved							
7	Transfer Length, MSB (if required)							
8	Transfer Length, LSB (if required)							
9	Reserved						Flag	Link

Table 6.3.B: 10 Byte Command Descriptor Block

6.3.1 OPERATION CODE

The MCDISK supports the following SCSI commands:

Code	Description	CDB Size
$00	TEST UNIT READY	6
$01	REZERO UNIT	6
$03	REQUEST SENSE	6
$04	FORMAT UNIT	6
$07	REASSIGN	6
$08	READ	6
$0A	WRITE	6
$0B	SEEK	6
$12	INQUIRY	6
$15	MODE SELECT	6
$16	RESERVE	6
$17	RELEASE	6
$1A	MODE SENSE	6
$1B	START/STOP UNIT	6
$1C	SEND DIAGNOSTIC COMMAND	6
$1D	RECEIVE DIAGNOSTIC RESULT	6
$1E	PREVENT/ALLOW REMOVAL	6
$25	READ CAPACITY	10
$28	READ EXTENDED	10
$2A	WRITE EXTENDED	10
$2B	SEEK EXTENDED	10
$2C	ERASE	10
$2E	WRITE VERIFY	10
$2F	VERIFY	10
$5A	MODE SENSE 10	10

Table 6.3.1: SCSI commands

6.3.2 LOGICAL UNIT NUMBER

The logical unit number (LUN) is used to address one of up to 8 logical devices within one physical SCSI device. In the MCDISK, the LUN is used to address different PCMCIA slots and multiple devices in a single memory card. The MCDISK assigns a separate LUN for each device (see paragraph 5). The most common case however is a memory card with a single device inserted into the lower slot (if two slots are available) of the MCDISK, which will be accessed through LUN 0.
LUN 0..3 are used for the *transparent mode*, while LUN 4..7 are used for *direct access mode*. Please refer to paragraph 5 for more information about transparent and direct modes of operation.

6.3.3 LOGICAL BLOCK ADDRESS

The logical block address (LBA) is used to specify a certain block on the logical unit. The range of the LBA starts at 0 and is contiguous up to the highest accessible block (which can be determined using the READ CAPACITY command). The number of bytes in a block can be set between 1 and 65535 bytes using the MODE SELECT command; the default value is 512 bytes. Note that this SCSI block size is absolutely independent from the size of the checksummed blocks on cards with logical formatting (see Chapter 4).

6.3.4 TRANSFER LENGTH

The transfer length is used to specify the number of blocks to be transferred (in data transfer commands like READ and WRITE) or the number of parameter/result bytes to be transferred (in commands like REQUEST SENSE or MODE SELECT).

If the transfer length in a 6 Byte CDB specifies the number of blocks as 0, the command will transfer 256 blocks of data (this is NOT the case in 10 Byte CDB's!).

6.3.5 LINK AND FLAG BITS

If the LINK bit is zero in a command, the MCDISK will send the COMMAND COMPLETE message after completion of the command (including status byte transfer) and will then go to BUS FREE phase. If the LINK bit is set and the FLAG bit is cleared, the MCDISK will send the LINKED COMMAND COMPLETE message and then go directly to the COMMAND phase again to receive the next command. The same happens when both LINK and FLAG bits are set, but the LINKED COMMAND COMPLETE (WITH FLAG) message will be sent instead.

6.4 STATUS BYTE

After completion (successful or unsuccessful) of a command, the MCDISK goes to the STATUS phase and transfers a single status byte to the initiator. The following list shows the different status codes that may be returned by the MCDISK:

Code	Description
$00	GOOD – This status is returned when a command has completed without error and there is no other error condition (such as an UNIT ATTENTION) pending.
$02	CHECK CONDITION – This status is returned when a command completes with an error condition. The initiator should issue a REQUEST SENSE command to retrieve additional status information.
$10	INTERMEDIATE/GOOD – This status is returned when a linked command has completed without error. If a linked command causes an error, the chain of linked commands will be broken and a CHECK CONDITION status is returned.
$18	RESERVATION CONFLICT – This status is returned when the MCDISK has been reserved for another initiator with the RESERVE command.

Table 6.4: Status Byte

6.5 UNIT ATTENTION CONDITION

The UNIT ATTENTION condition is used to inform the initiator that an external event has occurred that might affect the operation of subsequent commands.

When the MCDISK is in UNIT ATTENTION state and a command other than REQUEST SENSE or INQUIRY is issued, that command will not be executed, a CHECK CONDITION status will be returned and the UNIT attention status will be cleared. A subsequent REQUEST SENSE command will return the UNIT ATTENTION sense key.

When the MCDISK is in UNIT ATTENTION state and a REQUEST sense command is issued, it will be executed normally returning a UNIT ATTENTION sense key, and the UNIT ATTENTION status will be cleared.

The INQUIRY command will be executed returning a GOOD status regardless of a pending UNIT ATTENTION and it will not clear the UNIT ATTENTION status.

The following events cause the MCDISK to enter UNIT ATTENTION state:

- When a new card is inserted. This allows the initiator to detect card changes and reread card dependent information such as card capacity.
- A power-up sequence or a reset operation (by hardware or through RESET message).
- When a MODE SELECT command is issued, an UNIT ATTENTION is generated for all other initiators to notify them about the change in operating mode.

6.6 COMMAND DESCRIPTION

6.6.1 TEST UNIT READY ($00)

Bit: Byte	7	6	5	4	3	2	1	0
0	$00 (TEST UNIT READY)							
1	Logical Unit N°			reserved=0				
2	reserved=0							
3	reserved=0							
4	reserved=0							
5	reserved=0						Flag	Link

Table 6.6.1: Test unit ready command

This command checks if a logical unit is ready to be accessed. If so, it returns GOOD status, otherwise the status will be CHECK CONDITION.
In direct mode, the MCDISK returns not-ready status when no card is inserted.
In transparent mode, the MCDISK returns not-ready status if no card is inserted or if the card inserted signals busy status (with the BSY/RDY PCMCIA signal) or if there is no device or partition on the card for the LUN specified. If the card inserted is a PC-ATA disk, the MCDISK will also return not-ready status when the ATA controller is not ready. For cards that can be powered down with the STOP UNIT command, the MCDISK returns not-ready status if the card is powered down. Before accessing the card a START UNIT command should be issued.

6.6.2 REZERO UNIT ($01)

Bit: Byte	7	6	5	4	3	2	1	0
0	$01 (REZERO UNIT)							
1	Logical Unit N°			reserved=0				
2	reserved=0							
3	reserved=0							
4	reserved=0							
5	reserved=0						Flag	Link

Table 6.6.2: Rezero unit command

For PC-ATA cards, this command issues a RECALIBRATE command to the ATA drive, which causes the R/W heads to be restored to some initial position for mechanical drives. For all cards, REZERO UNIT will perform a PCMCIA hardware RESET and causes the card to be re-analysed (CIS reading, size testing, ATA controller initialisation), if it is accessed in transparent mode. In direct mode, the card will just be RESET.

6.6.3 REQUEST SENSE ($03)

Bit: Byte	7	6	5	4	3	2	1	0
0	$03 (REQUEST SENSE)							
1	Logical Unit N°			reserved=0				
2	reserved=0							
3	reserved=0							
4	Allocation Length (number of bytes)							
5	reserved=0						Flag	Link

Table 6.6.3.A: Request sense command

This command requests that the sense data (status information about last executed command) be sent to the initiator.
The ALLOCATION LENGTH parameter specifies the maximum number of sense data bytes to be sent to the initiator. The MCDISK will return at most 18 bytes of sense data, even if the ALLOCATION length is higher.

The MCDISK uses the extended data format as shown in the following table:

Bit: Byte	7	6	5	4	3	2	1	0
0	valid	Error class and code = $70						
1	$00							
2	0	0	0	0	sense key			
3	Information Byte MSB							
4	Information Byte							
5	Information Byte							
6	Information Byte LSB							
7	Additional sense length = 10							
8-11	reserved=0							
12	Additional Sense Code (ASC)							
13	Additional Sense Code Qualifier (ASCQ)							
14	$00							
15	FPV	C/D	0	0	0	Bit Pointer=0		
16	Field Pointer MSB							
17	Field Pointer LSB							

Table 6.6.3.B: Status information

VALID – This bit is set when the INFORMATION BYTES contain valid information.

ERROR CLASS AND CODE – Set to $70 to indicate extended status.

SENSE KEY – Values for sense key data are shown in table 6.6.3.C.

INFORMATION BYTES – If the VALID bit is set, these bytes are valid and contain the logical block address associated with the sense key.

ADDITIONAL SENSE LENGTH – The number of additional bytes to follow. This field is always = 10 for the MCDISK.

ADDITIONAL SENSE CODE (ASC) – The additional sense code is more precise error code (see table 6.6.3.C)

ADDITIONAL SENSE CODE QUALIFIER (ASCQ) – This field is used by the MCDISK to display internal, MPL specific error messages through the vendor specific codes $80…$FF. See table 6.6.3.D.

FPV – This bit is set when the FIELD POINTER is valid.

C/D – This bit is set when the FIELD POINTER is valid and is pointing to a byte of the CDB.

FIELD POINTER – If the FPV bit is set, these bytes contain a pointer to the byte that caused an ILLEGAL REQUEST condition. The failing byte is in the CDB when the C/D bit is set, else the FIELD POINTER refers to a byte in the data sent to the MCDISK.

Sense Key	ASC	Description
$0		NO SENSE – There is no sense key information to be reported. This sense key will be returned when a REQUEST SENSE command is issued after successful completion of a command (status GOOD).
	$00	No additional Information
$2		NOT READY – The MCDISK is not ready. This sense key will be returned when there is no memory card inserted (or has been removed during the access) or if there is no device on the card for the LUN that has been addressed in the command causing the CHECK CONDITION.
	$04	Drive not ready
	$3A	No media (no card inserted)
$3		MEDIUM ERROR – There is a problem with the memory card when tested during formatting or when accessed.
	$11	Data error (can not access PC-ATA block)
	$31	Bad card (during format)
$4		HARDWARE ERROR – There is a hardware problem detected by the MCDISK during power-up or operation.
	$03	Write fault. This can occur when writing to a non empty FLASH-EPROM location or when write to an ATA disk fails.
	$15	Seek error on ATA disk.
	$40	Diagnostic failure. This is relevant only when the MPL test card is used.
	$44	Internal controller error. This might be caused by various reasons. Check the ASCQ to get more information.

Sense Key	ASC	Description
$5		ILLEGAL REQUEST – There was an illegal opcode or parameter in the CDB or in the additional parameters supplied as data.
	$20	Invalid command opcode
	$21	Illegal logical block address
	$24	Illegal field in CDB
	$26	Illegal field in parameter data
$6		UNIT ATTENTION – The MCDISK has been reset or powered up, a new memory card has been inserted or another initiator has changed the operating mode of the MCDISK with the MODE SELECT command.
	$28	A new memory card has been inserted. Note that changing the write protect switch on a card while it remains inserted in the MCDISK is also considered as a card change. This is because conventional removable media cannot change their write-protect state without being removed from the drive.
	$29	The MCDISK has been reset or powered up.
	$2A	MODE SELECT parameters changed.
$7		DATA PROTECTED – Medium is protected, so write operations cannot be executed.
	$27	A write operation has been attempted on a non writeable card (read-only or write protected by switch, unknown card type or unknown error checking method)
$B		ABORTED COMMAND – The command has been aborted because of a communication error between the MCDISK and the initiator.
	$47	Parity error on SCSI bus.
	$48	Initiator detected error (initiator has sent INITIATOR DETECTED ERROR message).
	$49	Illegal message received.

Table 6.6.3.C: SENSE KEY and Additional Sense Code

ASCQ	meaning
$00	No special condition to report
$81	No card inserted
$82	Card extracted during access
$83	bad card (card test failed, ATA access problem)
$84	no appropriate CIS info found on card
$87	unknown (CIS lvl 2) partition type
$88	device is write protected
$89	Block check failure (ATA or memory card with checksum/CRC)
$8A	Insufficient card information for writing
$8B	write (FLASH) failed, unknown writing method or ATA write fault
$8C	card cannot be supported in transparent mode (e.g. modem I/O card)
$8D	card is inserted but not ready
$8E	ATA seek error (mechanical or logical)
$8F	ATA data error
$A1	card is not RAM (RAM test failed)
$A2	no such logical device
$A3	transfer address/size out of range
$A4	bad transfer attempt
$A5	unknown partition type
$A6	severe SCSI protocol problem
$A7	parity error occurred
$A8	initiator detected error
$AF	unimplemented feature selected
$91	bad card type/size specified
$BA	Missing or bad 12V supply for programming.
$BB	card attention: medium has changed
$BC	card attention: medium change request
$BD	Can not switch VCC to 6V for OTP programming
$C2	self-test error occurred.

Table 6.6.3.D: Additional Sense Code Qualifiers

Note: When reporting problems with a MCDISK, please always include Sense Key, Additional Sense Code (ASC) Additional Sense Code Qualifier (ASCQ) for error descriptions!

6.6.4 FORMAT UNIT ($04)

Bit: Byte	7	6	5	4	3	2	1	0
0	$04 (FORMAT UNIT)							
1	Logical Unit N°			FDT	CL	Defect List Fmt		
2	data pattern							
3	Interleave MSB							
4	Interleave LSB							
5	reserved=0						Flag	Link

Table 6.6.4: Format unit command

This command formats the logical unit. The card can be formatted either in raw mode (data blocks directly mapped to card addresses) or using CIS (card information structure) according to the PCMCIA/JEIDA standard. When using CIS, a checksum can be used to ensure data integrity for each memory block.

The operation performed at format depends on the card type and the parameters set in MODE SELECT PAGEs $03 and $32 (see description of MODE SELECT PAGES).
In addition, if a DEFECT LIST HEADER in the format command is present (FDT=1) , it may supersede the "card test" field of MODE SELECT PAGE $32 (see below).

Note also that the CIS mode (set through MODE SELECT PAGE $32 or by default through DIP switches/jumpers in those MCDISK versions that have 8 switches/jumpers, see paragraphs 3 and 4.4) is relevant for formatting.

FDT – Format Data. If this bit is set, it indicates that additional format data will be transferred in the data out phase (see below). If this bit is not set, no format data will be transferred.

CL – Complete List. This bit is not used by the MCDISK.

DEFECT LIST FMT – These three bits indicate the type of defect list. For the MCDISK, these bits must be set to 0.

DATA PATTERN – This field is provided for compatibility with real disk drives, but its value is not used in the MCDISK.

INTERLEAVE – This field is provided for compatibility with real disk drives, but its value is not used in the MCDISK.

If the FDT bit is set, the MCDISK expects a DEFECT LIST HEADER to be sent in the data out phase:

Bit: Byte	7	6	5	4	3	2	1	0
0	reserved=0							
1	X	X	DC	X	X	X	X	X
2	defect list length MSB = 0							
3	defect list length LSB = 0							

X – These bits are not used by the MCDISK.
DC – Disable medium certification. If this bit is set to 1, no card test will take place, even if card test was enabled before through MODE SELECT PAGE $32. If this bit is set to 0, and a card test method was set before in MODE SELECT PAGE $32, the card will be tested with the specified method. If no test was set with MODE SELECT PAGE $32, card test method 1 (non-destructive test) will be used.

Note: Please refer to the formatting notes (paragraph 7) for additional information.

6.6.5 REASSIGN BLOCKS ($07)

Bit: Byte	7	6	5	4	3	2	1	0
0	$07 (REASSIGN BLOCKS)							
1	Logical Unit N°			reserved=0				
2	reserved=0							
3	reserved=0							
4	reserved=0							
5	reserved=0						Flag	Link

Table 6.6.5.A: Reassign blocks command

This command is used to make defective blocks available for use again. For the MCDISK, a defective block is a block with a checksum error on a card that has checksum/CRC enabled. The REASSIGN BLOCKS commands as implemented in the MCDISK just updates the checksum of the blocks specified in the defect list, thus making them readable again. This might be useful for example to recover a card with checksums that was modified by a device that did not update the block checksums. Note that the REASSIGN BLOCKS command does not work with PC-ATA disks, as they don't support single block reallocation. To recover an ATA disk with bad blocks, use the FORMAT UNIT command with card test enabled (DC bit cleared in the DEFECT LIST HEADER). This does not work with all ATA disks, some are not able to map out bad blocks at all.

The REASSIGN BLOCK COMMAND expects a DEFECT LIST HEADER followed by 0 or more DEFECT DESCRIPTORS to be sent:

Bit: Byte	7	6	5	4	3	2	1	0
0,1	reserved=0							
2	defect list length MSB = 0							
3	defect list length LSB = 0							

Table 6.6.5.B: REASSIGN BLOCK defect list header

The DEFECT LIST LENGTH specifies the number of bytes used for the DEFECT DESCRIPTORS to follow (which is 4 times the number of defect descriptors, as they occupy 4 bytes each):

Bit: Byte	7	6	5	4	3	2	1	0
0	defect LBA (MSB)							
1	defect LBA							
2	defect LBA							
3	defect LBA (LSB)							

Table 6.6.5.C: REASSIGN BLOCK defect descriptor

The DEFECT LBA is the logical block address of the block to be updated. Note that due to the complete independency between logical block size and the actual block size on the card, REASSIGNing a single logical block might affect the checksum of multiple physical blocks.

6.6.6 READ ($08)

Bit: Byte	7	6	5	4	3	2	1	0
0	$08 (READ)							
1	Logical Unit N°			Logical Block Addr. MSB				
2	Logical Block Address							
3	Logical Block Address LSB							
4	Transfer Length (number of blocks)							
5	reserved=0						Flag	Link

Table 6.6.6: Read command

This command requests that one or a series of blocks starting at the specified address be transferred to the initiator.

LOGICAL BLOCK ADDRESS – This field specifies the number of the first block to be transferred. The first block on the logical unit has the LBA=0. The highest LBA that can be specified is the number of blocks on the device minus one. The highest possible LBA of the logical unit can be obtained using the READ CAPACITY command.

TRANSFER LENGTH – This field specifies the number of blocks to be transferred. A TRANSFER LENGTH of 0 will transfer 256 blocks of data (and NOT zero!)

If the memory card has been formatted in CIS mode with checksumming or CRC checking enabled, the MCDISK will check the data blocks for integrity before transferring them to the initiator. This will slow down the operation of the READ command. Block checking can be disabled by setting the RC (read continuous) bit in MODE SELECT page 1.

When a block check error occurs during the execution of the READ command and the DTE bit in MODE SELECT page 1 is not set (default), the MCDISK will exhaust the specified transfer length. If the DTE bit is set, the data transfer will be aborted at the next block boundary. The block in error will or will not be transferred depending on the state of the TB bit in MODE SELECT page 1. Note that the use of the TB bit makes sense only if the SCSI blocks size is an integer fraction of the card's block size (which is true when using the MCDISKs default settings, but may not be the case when custom block sizes are set using the MODE SELECT command).

6.6.7 WRITE ($0A)

Bit: Byte	7	6	5	4	3	2	1	0
0	$0A (WRITE)							
1	Logical Unit N°			Logical Block Addr. MSB				
2	Logical Block Address							
3	Logical Block Address LSB							
4	Transfer Length (number of blocks)							
5	reserved=0						Flag	Link

Table 6.6.7: Write command

This command requests that one or a series of blocks starting at the specified address be transferred from the initiator to the card.

LOGICAL BLOCK ADDRESS – This field specifies the number of the first block to be transferred. The first block on the logical unit has the LBA=0. The highest LBA that can be specified is the number of blocks on the device minus one. The highest possible LBA of the logical unit can be obtained using the READ CAPACITY command.

TRANSFER LENGTH – This field specifies the number of blocks to be transferred. A TRANSFER LENGTH of 0 will transfer 256 blocks of data (and NOT zero!)

If the memory card has been formatted in CIS mode with checksumming or CRC checking enabled, the MCDISK will update the block check of all physical data blocks affected by the write operation. When SCSI parity is enabled and a parity error is detected by the MCDISK on the data sent by the initiator, the MCDISK will stop writing immediately anything to the memory card. However, depending on the state of the DTE bit in MODE SELECT page 1 the MCDISK continues accepting data (but not storing it) until the next logical block boundary is reached or until the transfer length is exhausted.

The WRITE command will not execute and return a CHECK CONDITION with DATA PROTECT status when an attempt to write to a non-writeable card or a card that is write protected is made. See paragraph 5.6.2 to see under what conditions the MCDISK is able to recognize a card as writeable. If the card cannot be recognized, it will not be writeable in transparent mode. It will behave as a 64M ROM.
To write to a card that is not automatically recognized by the MCDISK, the card type must be specified using the MODE SELECT PAGE $30.

6.6.8 SEEK ($0B)

Bit: Byte	7	6	5	4	3	2	1	0
0	$0B (SEEK)							
1	Logical Unit N°			Logical Block Addr. MSB				
2	Logical Block Address							
3	Logical Block Address LSB							
4	reserved=0							
5	reserved=0						Flag	Link

Table 6.6.8: Seek command

This command is implemented in the MCDISK for compatibility only, and performs the same operation as the TEST UNIT READY command.

LOGICAL BLOCK ADDRESS – This field specifies the number of the first block to be seeked to. Because this command performs no action in the MCDISK, this field will not be used at all and may contain any value.

6.6.9 INQUIRY ($12)

Bit: Byte	7	6	5	4	3	2	1	0
0	$12 (INQUIRY)							
1	Logical Unit N°			reserved=0				
2	reserved=0							
3	reserved=0							
4	Allocation Length (number of bytes)							
5	reserved=0						Flag	Link

Table 6.6.9.A: Inquiry command

This command requests that the information regarding identification of the target be sent to the initiator

The ALOCATION LENGTH parameter specifies the maximum number of inquiry data bytes to be sent to the initiator. The MCDISK will return at most 120 bytes of inquiry data, even if ALLOCATION length is higher.

The INQUIRY data has the following format::

Bit: Byte	7	6	5	4	3	2	1	0
0	peripheral qualifier			peripheral device type				
1	RMB =1	device type modifier=0						
2	ISO Version=0		ECMA Version =0			ANSI Version =2		
3	Aenc	TrmIO	reserved =0		Response Data Format=2			
4	Additional list length							
5	reserved = 0							
6	reserved=0							
7	Radr	WB 32	WB 16	Sync	Linked	Res	CmdQ	SRes
8-15	Vendor Identification='MPL '							
16-31	Product Identification='MC-DISK-XX[X] '							
32-35	Firmware Revision Level='x.y '							
36-43	Firmware Date='yy/mm/dd'							
44-55	Serial Number space='DRV SER NUM'							
56-97	reserved=0							
98-119	vendor specific=0							

Table 6.6.9.B: Inquiry data

The PERIPHERAL QUALIFIER and the PERIPHERAL DEVICE TYPE fields identify the device currently connected to the logical unit.

A PERIPHERAL QUALIFIER of 0 specifies that the PERIPHERAL DEVICE TYPE is currently connected to this logical unit. A PERIPHERAL QUALIFIER of 1 specifies that the target is capable of supporting the PERIPHERAL DEVICE TYPE on this logical unit, but the physical device is not currently connected to this logical unit.

Code	Description
$00	Direct access device (SRAM,ATA)
$04	Write once device (Flash)
$1F	Unknown or no device type

Table 6.6.9.C: Peripheral device type code

Depending of the LUN, the MCDISK reports following PERIPHERAL QUALIFIER and PERIPHERAL DEVICE TYPE:

LUN	Qualifier	Device Type	Description
0 and 3	0	$00	Direct access card or no card inserted.
		$04	Write once card inserted (Flash, OTP)
1 and 2	0	$00	Additional partition of a direct access device.
		$04	Additional partition of write once device.
	1	$1F	No partition assigned to this LUN
4 to 7	1	$1F	Reports always unconnected, unknown or no device type on these LUNs

Table 6.6.9.D: Periperal device type codes for LUN

RMB - This bit is always set to indicate removable media.

DEVICE TYPE MODIFIER - Not supported by MCDISK, set to zero.

ISO VERSION - This field is set to 0 to indicate no compliance with ISO version of SCSI (ISO 9316).

ECMA VERSION - This field is set to 0 to indicate no compliance with European Computer Manufacturers Association's specification of SCSI.

ANSI VERSION - This field is set to 2 to indicate compliance with the ANSI specification of SCSI-2.

AENC - Asynchronous Event Notification Capability bit is set to zero.

TrmIO - Not supported by MCDISK, set to zero.

RESPONSE DATA FORMAT - This field is set to 2 to indicate compliance with the ANSI specification of SCSI-2.

ADDITIONAL LIST LENGTH - This field indicates the number of bytes of additional information that follow.

RAdr - Relative addressing not supported by MCDISK, set to zero.

WB32 - Wide bus 32 bit not supported by MCDISK, set to zero.

WB16 - Wide bus 16 bit not supported by MCDISK, set to zero.

Sync - Synchronous transfer not supported by MCDISK, set to zero.

Linked - Set to one to indicate the support of linked commands.

Res - Reserved set to zero.

CmdQ - Command queuing is not supported by MCDISK, set to zero.

SRes - Set to zero to indicate that the MCDISK responds to the RESET condition with the hard RESET alternative.

VENDOR IDENTIFICATION - Set to ‘MPL   ‘

PRODUCT IDENTIFICATION - This field shows the product name. Note that for compatibility with older versions the name is spelled with a hyphen (MC-DISK, not MCDISK). For example, the productname for the MCDISK-D-3 is ‘MC-DISK-D3’. Note that older firmware revisions of the MCDISK-D-1 and D-2 are the same and both identify as ‘MC-DISK-D’. If the product identification is used by software to identify the device as a MPL MCDISK, only the first 7 characters should be checked to cover all current and future versions.

For MCDISK units that have the Dual SCSI ID feature (see 5.3) the product identification string will be followed with a [0] for the configured SCSI ID and [1] for the second SCSI ID.

FIRMWARE REVISION LEVEL - This field shows the firmware revision. If the string reads for example ‘3.3’, the firmware is an officially released version. If the string reads for example ‘3.3c’ (letter at the end), the firmware is an intermediate beta officially release that may correct some bugs of the last release or add support for new devices. Note that older MCDISKs had a firmware revision string like ‘V3.0’.

6.6.10 MODE SELECT ($15)

Bit: Byte	7	6	5	4	3	2	1	0
0	$15 (MODE SELECT)							
1	Logical Unit N°			reserved=0				SP
2	reserved=0							
3	reserved=0							
4	Parameter List Length (number of bytes)							
5	reserved=0						Flag	Link

Table 6.6.10.A: Mode select command

This command is used to specify numerous operating parameters to the MCDISK. A MODE SELECT command will override any previous selection of device parameters, even from another initiator. Execution of a MODE SELECT command will generate a UNIT ATTENTION for all other initiators.

SP – Parameter saving is not implemented.

PARAMETER LIST LENGTH – This field specifies the number of parameter bytes that will be sent by the initiator in the DATA OUT phase. If this field is zero, no parameters will be transferred and no mode change will be done. This is not considered as an error.

The MODE SELECT parameter list contains a 4 Byte MODE SELECT HEADER, followed by zero or one BLOCK DESCRIPTOR and then zero or more MODE SELECT PAGES.

Bit: Byte	7	6	5	4	3	2	1	0
0	reserved=0							
1	medium type=0							
2	reserved=0							
3	Block Descriptor Length (Number of Bytes)							

6.6.10.B: Mode select header

MEDIUM TYPE – Set to zero for default (currently mounted medium type).

BLOCK DESCRIPTOR LENGTH – Specifies the number of bytes of all the BLOCK DESCRIPTORS that will follow. The MCDISK allows a single or no BLOCK DESCRIPTOR only, so this field must be 0 or 8.

Bit: Byte	7	6	5	4	3	2	1	0
0	density code=0							
1	Number of Blocks MSB=0							
2	Number of Blocks=0							
3	Number of Blocks LSB=0							
4	reserved=0							
5	reserved=0							
6	Block length MSB							
7	Block length LSB							

Table 6.6.10.C: Mode select block descriptor

DENSITY CODE – Set to zero, indicating the default density of the medium, because “density” is not defined for memory cards.

NUMBER OF BLOCKS – Set to zero to indicate that all logical blocks on the medium should be used.

BLOCK LENGTH – This field specifies the *logical block length* used to address and transfer data on the SCSI bus. It can be any value from 1 up to 65535 (see paragraph 5.2 about restrictions for the logical block size). The default block length is 512 bytes.

MODE SELECT PAGES

The MCDISK implements the following MODE SELECT PAGES defined in the SCSI standard: 1,3,4 and 5. In addition, the MCDISK implements 7 MPL specific mode select pages: $30, $31, $32, $33, $34, $35, and $36. See paragraph 6.6.14 for a detailed description of all mode select pages.

6.6.11 RESERVE ($16)

Bit: Byte	7	6	5	4	3	2	1	0
0	$16 (RESERVE)							
1	Logical Unit N° (don't care)			3rd PTY	3rd Party Device ID			0
2	reserved=0							
3	reserved=0							
4	reserved=0							
5	reserved=0						Flag	Link

Table 6.6.11: Reserve command

This command is used to reserve the MCDISK for exclusive use by a single initiator.

If the 3RD PTY bit is zero, the MCDISK will be reserved for the exclusive use of the initiator that issued the RESERVE command. Any attempt by another initiator to issue a command to the MCDISK will be rejected with a RESERVATION CONFLICT status. Only the initiator that issued the RESERVE command may release the reservation using the RELEASE command.
If the 3RD PTY bit is set, the MCDISK will be reserved for the initiator with the ID specified in the 3RD PARTY DEVICE ID field. Only this initiator and the initiator that issued the RESERVE COMMAND may release the reservation using the RELEASE command.

6.6.12 RELEASE ($17)

Bit: Byte	7	6	5	4	3	2	1	0
0	$17 (RELEASE)							
1	Logical Unit N° (don't care)			3rd PTY	3rd Party Device ID			0
2	reserved=0							
3	reserved=0							
4	reserved=0							
5	reserved=0						Flag	Link

Table 6.6.12: Release command

This command is used to release a reservation established with the RESERVE command before.

If the 3rd PTY bit is zero, the MCDISK will release the reservation (if any) for the initiator that issued the RELEASE command.

If the 3rd PTY bit is set, the reservation for the initiator with the ID specified in the 3RD PARTY DEVICE ID field will be released.

6.6.13 MODE SENSE ($1A)

Bit: Byte	7	6	5	4	3	2	1	0
0	$1A (MODE SENSE)							
1	Logical Unit N°			reserved=0				
2	PCF		Page Code					
3	reserved=0							
4	Allocation Length (number of bytes)							
5	reserved=0						Flag	Link

Table 6.6.13.A: Mode sense command

This command requests that operating mode settings (actual, default, changeable or saved) be transferred to the initiator. PCF – Page Control Field. This field defines the type of page information to be returned:

0 0	Report **current** values. The currently active settings will be returned.
0 1	Report **changeable** values. Those parameter bits that may be changed using MODE SELECT will be set in the returned information.
1 0	Report **default** values. The default settings will be returned. The default settings are stored in the MCDISKs firmware ROM and cannot be changed. However, for some parameters, there is no "default", as they completely depend on the type of card inserted. So, some "default" values are equal to the corresponding "current" values, if no card independent "default" value exists.
1 1	Report **saved** values. The MCDISK cannot save MODE SELECT parameters, so this values are always equal to the default values.

PAGE CODE - This field specifies the number of the page to be reported. If this field contains $3F, all pages implemented should be returned, but since all pages consist of more than 256 bytes, a CHECK CONDITION status will be returned. MODE SENSE(10) ($5F) should be used to retrieve all pages instead.

ALLOCATION LENGTH – This field specifies the maximum number of bytes to be returned by the MCDISK.

The MODE SENSE DATA always returns a MODE SENSE header and a MODE SENSE BLOCK DESCRIPTOR.

Bit: Byte	7	6	5	4	3	2	1	0
0	Sense data length							
1	medium type=0							
2	reserved=0							
3	Block Descriptor Length=8							

Table 6.6.13.B: Mode sense header

SENSE DATA LENGTH – Specifies the length in bytes of the following MODE SENSE data that is available. The SENSE DATA LENGTH does not include itself.

MEDIUM TYPE – Set to zero for default (currently mounted medium type).

BLOCK DESCRIPTOR LENGTH – Because a single BLOCK DESCRIPTOR will follow, this byte is 8.

BLOCK DESCRIPTOR:

Bit: Byte	7	6	5	4	3	2	1	0
0	density code=0							
1	Number of Blocks MSB=0							
2	Number of Blocks=0							
3	Number of Blocks LSB=0							
4	reserved=0							
5	reserved=0							
6	Block length MSB							
7	Block length LSB							

Table 6.6.13.C: Mode select block descriptor

DENSITY CODE – Set to zero, indicating the default density of the medium, because “density” is not defined for memory cards.

NUMBER OF BLOCKS – Set to zero to indicate that all logical blocks on the medium should be used.

BLOCK LENGTH – This field specifies the *logical block length* used to address and transfer data on the SCSI bus. The default block length is 512 bytes (see paragraph 5.2 for more about blocks).

MODE SENSE PAGES

The MCDISK implements the following MODE SENSE PAGES defined in the SCSI standard: 1, 3, 4 and 5. In addition, the MCDISK implements 7 MPL specific mode select pages: $30, $31, $32, $33, $34, $35, $36 and $38. See paragraph 6.6.14 for a detailed description of all mode select/ sense pages.

Note that when issued mode sense of all pages, page 4 (HARD DISK GEOMETRY) will only be returned if an ATA disk is inserted, otherwise page 5 (FLOPPY DISK GEOMETRY) will be returned. This is to prevent some host to get a floppy/ harddisk driver missmatch.

6.6.14 MODE SELECT/SENSE PAGES

6.6.14.1 PAGE $01: ERROR DETECTION

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $01					
1	page length = $06 (=6) bytes							
2	0	0	TB	RC	0	0	DTE	0
3	Retry count							
4..7	not used = 0							

Table 6.6.14.1: Page $01, error detection

TB (Transfer Block) – If this bit is set and the DTE bit is also set, the block that caused an error will be transferred before the transfer is aborted. This bit is 0 by default.

RC (Read Continuous) – If this bit is set, the MCDISK does not perform error detection on read. This speeds up data transfer greatly. However, to monitor data integrity, this bit must be 0, which is the default.

DTE (Disable transfer on error) – If this bit is cleared, a data transfer will always transmit the requested amount of data even if an error occurs in the middle. However, the data returned might be invalid. If this bit is set, the transfer aborts at the next block boundary if an error occurs. Depending on the TB bit, the failing block will or will not be transferred. By default, this bit is 0.

RETRY COUNT – This value is always returned as 1. The MCDISK does not use it internally, but it can be set to any value in MODESELECT PAGE $1 without causing errors.

6.6.14.2 PAGE $03: FORMAT PARAMETERS

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $03					
1	page length = $16 (=22) bytes							
2..9	unused = 0							
10	Sectors per track (MSB) = 0							
11	Sectors per track (LSB)							
12	Bytes per physical sectors (MSB)							
13	Bytes per physical sectors (LSB)							
14..19	unused = 0							
20	SSEC =1	HSEC =0	RMB =1	SURF =0	INS =0	reserved=0		
21..23	reserved=0							

Table 6.6.14.2: Page $03, Format parameters

SECTORS PER TRACK – This field’s contents, if <>0, will be stored in the geometry tuple of a card being formatted. For PC-ATA cards, this value cannot be changed. Its value has no influence on MCDISK operation. By default, this value is 0.

BYTES PER PHYSICAL SECTOR – This is the physical block size. Only powers of two (e.g. 128, 256, 512 …) are allowed. The block size will be stored in the JEIDA format tuple (ID=$41) of a card being formatted. For PC-ATA cards, this value is always 512 and can not be changed.

SSEC=1 means soft sectored, HSEC=0 means not hard sectored, RMB=1 means removable medium, SURF=0 means surfaces are switched before stepping to the next cylinder. Except RMB, these bits are meaningless for the MCDISK. See also special note about geometry changes in paragraph 6.6.14.4.

6.6.14.3 PAGE $04: HARD DISK GEOMETRY

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $04					
1	page length = $12 (=18) bytes							
2	Number of cylinders (MSB)							
3	Number of cylinders							
4	Number of cylinders (LSB)							
5	Number of heads (surfaces)							
6..19	unused=0							

Table 6.6.14.3: Page $04, Hard disk Geometry

This page is only included in mode sense all pages if an ATA disk is inserted to prevent some host to get a floppy/harddisk driver missmatch.

NUMBER OF CYLINDERS – This is the number of cylinders (=storage entity consisting of (sectors per track)*(Number of heads) blocks) on the card. This value may be set, but will be adjusted by the MCDISK so that (cylinders)*(heads)*(sectors per track)*(bytes per sector) matches the actual card capacity as close as possible. It will be stored in the geometry tuple of a card being formatted, if sectors per track (see 6.6.14.2) is <>0. For PC-ATA cards, this value shows the number of cylinders as reported by the ATA disk's IDENTIFY command, which is not changeable.

NUMBER OF HEADS – This is the number of heads. This value will be stored in the geometry tuple of a card being formatted, if sectors per track (see 6.6.14.2) is <>0. The value defaults to 0; for PC-ATA cards this value shows the number of heads as reported by the ATA disk's IDENTIFY command. It can not be changed. See also special note about geometry changes in paragraph 6.6.14.4.

6.6.14.4 PAGE $05: FLOPPY DISK GEOMETRY

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $05					
1	page length = $1E (=30) bytes							
2,3	unused=0							
4	Number of heads (sides)							
5	Sectors per track							
6	Bytes per sector (MSB)							
7	Bytes per sector (LSB)							
8	Number of cylinders (MSB)							
9	Number of cylinders (LSB)							
10..27	unused=0							
28..31	reserved=0							

Table 6.6.14.4: Page $05, Floppy Disk Geometry

This page is only included in mode sense all pages if NO ATA disk is inserted to prevent some host to get a floppy/harddisk driver missmatch.

NUMBER OF HEADS – This is the number of heads. This value will be stored in the geometry tuple of a card being formatted, if sectors per track (see 6.6.14.2) is 1[1]. The value defaults to 0. For PC-ATA cards, this value shows the number of heads as reported by the ATA disk's IDENTIFY command, which is not changeable.

SECTORS PER TRACK – This field’s contents will be stored in the geometry tuple of a card being formatted, if sectors per track (see 6.6.14.2) is <>0. By default, this value is 1[1]. For PC-ATA cards, this value shows the number of sectors per track as reported by the ATA disk's IDENTIFY command, which is not changeable.

BYTES PER SECTOR – This is the physical block size. Only powers of two (e.g. 128, 256, 512 …) are allowed. The block size will be stored in the JEIDA format tuple (ID=$41) of a card being formatted. For PC-ATA cards, this value is always 512 and can not be changed.

NUMBER OF CYLINDERS – This is the number of cylinders (=storage entity consisting of (sectors per track)*(Number of heads) blocks) on the card. This value may be set, but will be adjusted by the MCDISK so that (cylinders)*(heads)*(sectors per track)*(bytes per sector) matches the actual card capacity as close as possible. It will be stored in the geometry tuple of a card being formatted, if sectors per track (see 6.6.14.2) is <>0. For PC-ATA cards, this value shows the number of cylinders as reported by the ATA disk's IDENTIFY command, which is not changeable.

Special Note:

For page 3, 4 and 5: Note that changes of the disk geometry parameters (sectors, heads and cylinders) made through MODE SELECT PAGES 3, 4 or 5 will be immediately effective

---

[1] Note that these values are, unlike their representation in page 3 and 4, set to 1 rather than 0 if no real geometry is available. This is to prevent some hosts low level drivers from crashing with a divide by zero error.

for subsequent geometry-related operations such as MODE SENSE. However, the geometry will be reset to its previous values when the card is extracted and re-inserted. To make geometry changes permanent for a card, the card must be formatted using the FORMAT UNIT command after changing geometry through any of MODE SELECT PAGES 3, 4 or 5.

6.6.14.5 PAGE $30: PHYSICAL DEVICE SPECS

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $30					
1	page length = $06 (=6) bytes							
2	SF	WPA	NOC	I/O	Card Type			
3	Bsy	S1	IOS	WPS	0	0	Battery	
4	Device size in bytes (MSB)							
5	Device size in bytes							
6	Device size in bytes							
7	Device size in bytes (LSB)							

Table 6.6.14.5.A: Page $30, Physical device specs

SF - Special Function. If this bit is set, it indicates that the CARD TYPE field specifies not a simple memory type, but a special function (such as ATA). This field is changeable.

WPA - Write protect active. If this bit is set, the write protect switch is used to decide if the device assigned to the LUN is writeable or not. If this bit is set to zero, the assigned device is always writeable. This field is changeable.

NOC - No count mode. If this bit is set, the address counters of the MCDISK are disabled. This allows to read/write multiple bytes to a single location on the card with a single read/write command. This field is changeable.

I/O - If this bit is set, it indicates that the device is accessed in I/O mode (using IORD and IOWR signals instead of OE and WR). In direct mode (LUN 4..7), this bit can be modified using MODE SELECT to set the access mode. In transparent mode, this bit cannot be modified.

DEVICE TYPE - This field defines the memory type of the LUN being accessed. If the SF bit is set, it indicates the special function. If the SF bit is set, it indicates the special function. This field is changeable, however, changing the device type is required only for special cases as the MCDISK automatically detects card type in transparent mode. Great care must be applied when setting this field, because writing to a card with a wrong card type specification might damage the card. Therefore, ensure that the card type set here and the actual card type are the same. See Table 6.6.14.5.B for card type/special function definitions.

Code	Device Memory (SF=0)	SF=1
0	No Device	PC-ATA Disk
1	Mask ROM	reserved
2	OTPROM	reserved
3	EPROM	reserved
4	EEPROM	reserved
5	Flash-EPROM	reserved
6	RAM with battery backup	reserved
7	RAM without battery backup	reserved

Table 6.6.14.5.B: Card type/special function definition

BSY - This bit reflects the status of the PCMCIA RDY/BSY signal. If it signals busy status, the BSY bit is set to 1. This bit is read-only and must be set to 0 when page $30 is used in a MODE SELECT command.

S1 - Slot 1. If this bit is set, the LUN being accessed belongs to the upper (second) PCMCIA card slot. For single-slot versions of the MCDISK, this bit is always set to 0, indicating that the LUN belongs to the first and only slot.

IOS - This bit is set when the MCDISK interprets the WP, RDY/BSY and BVD1/2 signals in I/O mode as IOIS16, INTRQ, SPKR/STCHG.

WPS - Write Protect switch. This bit indicates the state of the write protect switch (if any) of the card in the addressed slot. Note that the MCDISK may refuse to write to a card even if the WP bit indicates "non-protected" due to unknown card type, bad parameters, missing programming voltage etc.

BATTERY - This field is read-only, and must be set to 0 when page $30 is used in a MODE SELECT command. It indicates the state of the cardís backup battery (if any) as follows: 2=good, 1=still working, but replacement needed, 0=bad.

DEVICE SIZE - This field defines the overall size (in bytes) of the device being accessed. Note that this is not necessarily equal to the data capacity of the device, because CIS and checksum occupy some space on CIS formatted cards. This field is changeable only if the MCDISK is unable to determine the size automatically, that is, if the card inserted is not unprotected RAM and does not identify itself (i.e. does not supply CIS level 1 information or JEDEC identifiers).

Note that the device type and/or the device size set through MODE SELECT PAGE $30 remain effective only until the card is changed or formatted. On formatting, the MCDISK will try to create Level 1 CIS so that the card will identify itself afterwards (i.e. making the settings permanent). However, if this is not possible (e.g. CIS mode<>3, raw formatting), the device type/size will be set back to the default values for unknown cards (64 Megabyte ROM) after formatting.

6.6.14.6 PAGE $31: LEVEL 2 INFORMATION

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $31					
1	page length = $32 (=50) bytes							
2..26	Level 2 formatting Program/Device name							
27..51	additional Level 2 information string							

Table 6.6.14.6: Page $31, Level 2 information

When this page is used in a MODE SET command, it does *not* affect the current card's CIS, but the new strings will be stored on the next card being formatted, if a new CIS is written on formatting. However, MODE SENSE will return the values set through MODE SET as long as the card is not extracted and re-inserted.

6.6.14.7 PAGE $32: FORMAT PARAMETERS

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $32					
1	page length = $08 (=8) bytes							
2	JEIDA Format type							
3	Error detection method							
4	Card test method							
5	0	0	0	0	0	0	Slw	Fill
6	Fill pattern (MSB)							
7	Fill pattern (LSB)							
8	CIS mode selection							
9	CIS size							

Table 6.6.14.7: Page $32, Format parameters

JEIDA FORMAT TYPE – This indicates the type of logical formatting, as stored in the Format Tuple ($41) of the card. Three types are defined: 0=disk-like, 1=memory-like, 255=raw format. The default value is 0.

ERROR DETECTION METHOD– This specifies the error detection method to be used: 0=none, 1=simple 8 bit checksum, 2=16 bit CRC. Note that CRC is not yet implemented in the MCDISK. The default value is 0.

CARD TEST METHOD – This specifies the type of card testing to be done on formatting: 0=no test, 1=non-destructive R/W test, 2=fully destructive address test. The default value is 0. For PC-ATA cards, 1 and 2 are synonymous and both cause the drive to be fully verified and, if possible, bad blocks to be mapped out. Note that the DC bit in the FORMAT UNIT DEFECT LIST HEADER will override the settings of this field (see paragraph 6.6.4)

FILL – if this bit is set, the card will be filled with the Fill Pattern (see below) on formatting. Note that this works only on fully rewriteable memory cards such as SRAM or EEPROM, but not with FLASH or PC-ATA.

FILL PATTERN – This pattern will be repeatedly written to the memory card on formatting when the Fill bit is set. The default value is $FF.

FILL PATTERN (COPY) – Firmware revisions before 3.0 had a 16-bit fill pattern. In firmware 3.0 and newer, the fill pattern is only 8 bit wide. This field is for compatibility with pre-3.0 versions of the MCDISK. When written, the value is ignored. When read, it shows the same value as the FILL PATTERN field.

CIS MODE – This field can be used to override the CIS mode set through jumper 4-7/8 (Note that not all MCDISK hardware have these jumpers. When the jumpers are not available, the MCDISK defaults to FULL CIS MODE. The CIS mode controls the operation of the MCDISK in transparent mode. It does not affect direct mode.

When this field is read, CIS MODE indicates the following CIS conditions:
0: no CIS
1: Level 1 CIS only
2: Level 2 CIS only
3: both Level 1 and 2 CIS

CIS MODE	JUMPERS 4-8	4-7	Description
3	Open	Open	FULL CIS MODE. In this mode, the MCDISK fully analyzes CIS and creates Level 1 and Level 2 CIS on formatting (if required).
2	Open	Set	NO LVL1 CIS WRITING MODE. This is like FULL CIS MODE, except that the MCDISK will not try to create Level 1 CIS on formatting.
1	Set	Open	NO COMMON SPACE CIS MODE. This is like NO LVL1 CIS WRITING MODE, except that the MCDISK will not write any CIS to the common memory space on formatting. If CIS exists in common memory space, it will be erased on formatting.
0	Set	Set	NO CIS MODE. In this mode, all CIS recognition and formatting is disabled. This mode might be useful to operate on a card with bad CIS information.

CIS SIZE: 0=default=512, $FF=auto, others=cissize=$2^n$. This can be useful to set the CIS size equal to an eraseable block size on FLASH cards without separate attribute memory such that the data blocks can be erased without erasing CIS information.

When this page is used in a MODE SET command, it does *not* affect the current card's CIS, but the new strings will be stored on the next card being formatted, if a new CIS is written on formatting. However, MODE SENSE will return the values set through MODE SET as long as the card is not extracted and re-inserted.

6.6.14.8 PAGE $33: INITIALISATION DATE

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $33					
1	page length = $06 (=6) bytes							
2	Year							
3	Month							
4	Day							
5	Hour							
6	Minute							
7	Second							

Table 6.6.14.8: Page $33, Initialisation date

This page is used to query/set the initialisation date as stored in the Initialisation Date Tuple (ID=$44).

When this page is used in a MODE SET command, it does *not* affect the current card's CIS, but the new date will be stored on the next card being formatted. Therefore, MODE SENSE will not return the date/time set through MODE SET as long as the card is not re-formatted.

6.6.14.9 PAGE $34: DATA ORGANISATION

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $34					
1	page length = $10 (=16) bytes							
2	Data organisation code							
3..17	Data organisation name							

Table 6.6.14.9: Page $34, Data organisation

DATA ORGANISATION CODE – This code specifies the type of data organisation: 0=file system, 1=application, 2=executable ROM code, $80=MCRW-B format. The default is $00 (Before firmware 3.0, the default was $80).

DATA ORGANISATION NAME – This is a null terminated string consisting of 14 characters max. By default, this string is empty. Note that firmware versions older than 3.0 of the MCDISK used a default string of "MPL MCRW-B".

When this page is used in a MODE SET command, it does *not* affect the current card's CIS, but the new strings will be stored on the next card being formatted, if a new CIS is written on formatting. However, MODE SENSE will return the values set through MODE SET as long as the card is not extracted and re-inserted.

6.6.14.10 PAGE $35: MCRW COMPATIBILITY

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $35					
1	page length = $22 (=34) bytes							
2	VAL	0	UPD	0	0	0	Text	RT
3	reserved=0							
4	Number of records (MSB)							
5	Number of records							
6	Number of records							
7	Number of records (LSB)							
8	Offset to first free byte (MSB)							
9	Offset to first free byte							
10	Offset to first free byte							
11	Offset to first free byte (LSB)							
12	Block size (MSB)							
13	Block size (LSB)							
14	Last Access (year)							
15	Last Access (month)							
16	Last Access (day)							
17	Last Access (hour)							
18	Last Access (minute)							
19	Last Access (second)							
20..34	Card name							
35	reserved=0							

Table 6.6.14.10: Page $35, MCRW compatibility

VAL – Valid MCRW information. This bit indicates valid MCRW compatibility information. For MODE SENSE, this means that the card has a MCRW Compatibility tuple (ID=$80), that is, the card is prepared to be read and/or written by a MCRW-B using all of its features. For MODE SELECT, this bit indicates if a MCRW compatibility tuple should be written to the next card being formatted.

UPD – This read-only bit is set when the card is not updatable (i.e. not writeable more than once, like FLASH-EPROM). This means that the Number of records field, the Offset to first free byte and the Last access date are not valid.

TEXT –This bit indicates MCRW text mode formatting.

RT – This read-only bit indicates that the card was formatted with a MCRW-B in real time mode.

NUMBER OF RECORDS – If the UPD bit is 0, this read-only field indicates the number of valid records on the card.

OFFSET TO FIRST FREE BYTES – If the UPD bit is 0, this read-only field contains an offset to the first byte of the first invalid record on the card.

BLOCK SIZE – This is the block size used by the MCRW-B in fixed block mode. Note that this block size is neither related to the SCSI block size nor the physical block size.

LAST ACCESS – If the RT bit is set, this read-only field contains the date of the last write access to the card done by a MCRW-B. Note that this date is *not* updated by the MCDISK.
CARD NAME – This is the name of the card., consisting of a null terminated ASCII-string with 12 characters max.

When this page is used in a MODE SET command, it does *not* affect the current card's CIS, but the new strings will be stored on the next card being formatted, if a new CIS is written on formatting. However, MODE SENSE will return the values set through MODE SET as long as the card is not extracted and re-inserted.

6.6.14.11 PAGE $36:DEVICE INFORMATION

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $36					
1	page length = $14 (=20) bytes							
2,3	JEDEC ID							
4,5	JEIDA Version							
6	bus width 2^{x-1}							
7	erase block size 2^{x-1} bus words							
8	parttition size 2^{x-1}eraseblock size							
9	hardware interleave 2^{x-1}							
10-13	erase block size in bytes							
14-17	offset in bytes from device start to first eraseable block							
18	Device speed (slowest)							
19	Devivice max speed							
20-21	reserved=0							

Table 6.6.14.11.A: Page $36, Device information

This page gives additional information about the devices and allows setting some device parameters. It is especially useful with FLASH memories to determine the erase block size. Note that before firmware version 3.0, this page was read-only and did not have device speed fields.

JEDEC ID - This shows the JEDEC ID of the memory devices used, if this information is contained in the card's CIS. $0000 means that the device has no JEDEC ID, i.e SRAM. When writing the page, $0000 means that the JEDEC ID should not be changed.

JEIDA VERSION - The first byte contains the major version number and the second byte contains the minor version number (full version is "major.minor", like 4.1). PCMCIA 2.0 is compatible with JEIDA 4.1. This field cannot be changed.

BUS WIDTH - Shows the bus width of the card in bytes = 2(x-1) bytes. It is normally =2, so 2(2-1) =2 Bytes (=16 bit bus). This field reflects the DGTPL_BUS in the CISTPL_DEVICEGEO and cannot be changed.

ERASE BLOCK SIZE - Shows the size of the smallest eraseable block size of 2(x-1) address increments of BUS WIDTH. This field reflects the DGTPL_EBS in the CISTPL_DEVICEGEO and cannot be changed.

PARTITION SIZE - Shows the minimum partition size in 2(x-1) number of erase blocks. x=1 where array partitioning at erase block boundaries is allowed. This field reflects the DGTPL_PART in the CISTPL_DEVICEGEO and cannot be changed.

HARDWARE INTERLEAVE - Shows the value for card employing hardware interleaved (i.e., "banks" of) memory arrays. Non-interleaved cards have values of x=1. This field reflects the DGTPL_HWIL in the CISTPL_DEVICEGEO and cannot be changed.

ERASE BLOCK SIZE IN BYTES - This is a more convenient way to inspect the erase block size. This field cannot be changed.

OFFSET IN BYTES - This field shows where the first eraseable block starts within the currently addressed logical unit. As there might be some space reserved for CIS on the first physical block of the card, the data area addressable in transparent mode might start not at the beginning, but in midst of the first physical block. This fields therefore shows the number of data bytes in the first physical block. If the first block addressed through the LUN is also at the beginning of an eraseable block, this value is 0. This field cannot be changed.

DEVICE SPEED - This field indicates the minimal speed of the device, i.e. how fast the device can be accessed if the WAIT signal is ignored. See below for the format of the speed specification. When writing the page, $00 means that the device speed should not be changed.

MAX DEVICE SPEED - This field indicates the maximal speed of the device, when the WAIT signal is monitored and is used to slow down accesses below the maximum speed if required. When writing the page, $00 means that the device speed should not be changed.

The speed is specified as follows:

Bit:	7	6	5	4	3	2	1	0
Speed	Exponent				Mantissa			

Intepretation of the Speed Field:

Code	Exponent meaning	Mantissa meaning
0	1 ns	reserved
1	10 ns	1.0
2	100ns	1.2
3	1 µs	1.3
4	10 µs	1.5
5	100 µs	2.0
6	1 ms	2.5
7	10 ms	3.0
8	reserved	3.5
9	reserved	4.0
A	reserved	4.5
B	reserved	5.0
C	reserved	5.5
D	reserved	6.0
E	reserved	7.0
F	reserved	9.0

6.6.14.12 PAGE $38: CARD INFORMATION

Bit: Byte	7	6	5	4	3	2	1	0
0	0		Page Code = $38					
1	page length = $6E (=110) bytes							
2-21	card manufacturer name							
22-41	product information (product name)							
42-71	additional product info 1							
72-111	additional product info 2							

Table 6.6.14.10: Page $38, card information

This page gives text information about the card for those cards that are equipped with a CISTPL_VERS1 tuple. This page is read-only.

CARD MANUFACTURER NAME – This string contains the name of the company that manufactured the card.

PRODUCT INFORMATION (PRODUCT NAME) – This string contains the name of the product.

ADDITIONAL PRODUCT INFO 1 – This string gives additional information about the product. PCMCIA suggests to provide the lot number here.

ADDITIONAL PRODUCT INFO 2 – This string gives even more additional information about the product. PCMCIA suggests to provide programming conditions here.

6.6.15 START/STOP UNIT ($1B)

Bit: Byte	7	6	5	4	3	2	1	0
0	$1B (START/STOP UNIT)							
1	Logical Unit N°			reserved=0				imm
2	reserved=0							
3	reserved=0							
4	reserved=0							start
5	reserved=0						Flag	Link

Table 6.6.15: Start / Stop unit command

If the START bit is set, cards that support the global powerdown bit, will be powered up. For PC-ATA cards that support IDLE/STANDBY commands, the card will be set into IDLE mode (mechanical drives will spin up the disks).

If the START bit is reset, cards that support the global powerdown bit, will be set into low power state. For PC-ATA cards, a STANDBY command will be issued (mechanical drives will spin down).

Note that if a card is powered down with STOP UNIT, it will return NOT READY status until the START UNIT command is issued, even if the card inserted has no power down features.

6.6.16 DIAGNOSTIC COMMANDS ($1C/$1D)

These command are implemented in the MCDISK, but are not used in normal operation (factory test only). Do not issue these commands.

6.6.17 PREVENT/ALLOW REMOVAL ($1E)

Bit: Byte	7	6	5	4	3	2	1	0
0	$1E (PREVENT/ALLOW REMOVAL)							
1	Logical Unit N°			reserved=0				imm
2	reserved=0							
3	reserved=0							
4	reserved=0							prev
5	reserved=0						Flag	Link

Table 6.6.17: Prevent/Allow Removal command

This command is used to prevent or allow removal of a card or a media inserted into the card.

If the PREV bit is set, it means that removal of the card or media should be prevented. If the card has no media that can be ejected, the MCDISK just turns on the yellow access led permanently to indicate to the user that the card should not be extracted now. However, there is no mechanical lock that would prevent card extraction. If the card has itself an ejectable media and supports locking, the MCDISK will lock the media.

If the PREV bit is cleared, it means that removal of the card or media is now allowed again. The MCDISK will turn off the yellow access LED and, if the card itself has an ejectable media, will unlock the media.

6.6.18 READ CAPACITY ($25)

Bit: Byte	7	6	5	4	3	2	1	0
0	$25 (READ CAPACITY)							
1	Logical Unit N°			reserved=0				0
2	0							
3	0							
4	0							
5	0							
6	reserved=0							
7	reserved=0							
8	reserved=0							
9	reserved=0						Flag	Link

Table 6.6.16.A: Read capacity command

This command requests that the capacity of the medium (memory card) be reported to the initiator. The result data consists of eight bytes as follows:

Bit: Byte	7	6	5	4	3	2	1	0
0	Last Block's Number, MSB							
1	Last Block's Number							
2	Last Block's Number							
3	Last Block's Number, LSB							
4	Block Size, MSB							
5	Block Size							
6	Block Size							
7	Block Size, LSB							

Table 6.6.16.B: Read capacity data

LAST BLOCK'S NUMBER – This is the number of the last logical block on the logical unit. Note that if the MCDISK cannot determine the card's capacity, it will return the size of the entire PCMCIA address space.

BLOCK SIZE – This is the current logical block size

6.6.19 READ EXTENDED ($28)

Bit:	7	6	5	4	3	2	1	0
Byte								
0	$28 (READ EXTENDED)							
1	Logical Unit N°			reserved=0				0
2	Logical Block Address, MSB							
3	Logical Block Address							
4	Logical Block Address							
5	Logical Block Address, LSB							
6	reserved=0							
7	Transfer Length, MSB							
8	Transfer Length, LSB							
9	reserved=0						Flag	Link

Table 6.6.17: Read extended command

This command is functionally equivalent to the READ command, except that it provides a wider range for both LOGICAL BLOCK ADDRESS and TRANSFER LENGTH values. If the TRANSFER LENGTH is zero, no data will be transferred.

6.6.20 WRITE EXTENDED ($2A)

Bit:	7	6	5	4	3	2	1	0
Byte								
0	$2A (WRITE EXTENDED)							
1	Logical Unit N°			reserved=0				0
2	Logical Block Address, MSB							
3	Logical Block Address							
4	Logical Block Address							
5	Logical Block Address, LSB							
6	reserved=0							
7	Transfer Length, MSB							
8	Transfer Length, LSB							
9	reserved=0						Flag	Link

Table 6.6.18: Write extended command

This command is functionally equivalent to the WRITE command, except that it provides a wider range for both LOGICAL BLOCK ADDRESS and TRANSFER LENGTH values. If the TRANSFER LENGTH is zero, no data will be transferred.

6.6.21 SEEK EXTENDED ($2B)

Bit:	7	6	5	4	3	2	1	0
Byte								
0	$2B (SEEK EXTENDED)							
1	Logical Unit N°			reserved=0				0
2	Logical Block Address, MSB							
3	Logical Block Address							
4	Logical Block Address							
5	Logical Block Address, LSB							
6	reserved=0							
7	reserved=0							
8	reserved=0							
9	reserved=0						Flag	Link

Table 6.6.21: Seek extended command

This command is implemented in the MCDISK for compatibility with mechanical storage devices only, and performs the same operation as the TEST UNIT READY command.

LOGICAL BLOCK ADDRESS – This field specifies the number of the first block to be seeked to. Because this command performs no action in the MCDISK, this field will not be used at all and may contain any value.

6.6.22 ERASE ($2C)

Bit:	7	6	5	4	3	2	1	0
Byte								
0	$2C (ERASE)							
1	Logical Unit N°			reserved=0				0
2	Logical Block Address, MSB							
3	Logical Block Address							
4	Logical Block Address							
5	Logical Block Address, LSB							
6	reserved=0							
7	Erase Length, MSB							
8	Erase Length, LSB							
9	reserved=0						Flag	Link

Table 6.6.22: Erase command

LOGICAL BLOCK ADDRESS – This field specifies the number of the first block to erase.

ERASE LENGTH – This field specifies the number of blocks to be erased.

Note that erasing is restricted to the physical property of the card. FLASH-EPROM cards can only be erased in blocks of 128K or even 512K at once. Thus, the LBA and EL (erase length) parameters must be chosen to cover an integer number of eraseable card blocks. Otherwise, ERASE will return a check condition with illegal block address status. To obtain the eraseable block size and the start of the first eraseable block within a device, examine MODE SENSE page $36.

6.6.23 WRITE VERIFY ($2E)

Bit: Byte	7	6	5	4	3	2	1	0
0	$2E (WRITE VERIFY)							
1	Logical Unit N°			reserved=0				0
2	Logical Block Address, MSB							
3	Logical Block Address							
4	Logical Block Address							
5	Logical Block Address, LSB							
6	reserved=0							
7	Transfer Length, MSB							
8	Transfer Length, LSB							
9	reserved=0						Flag	Link

Table 6.6.23: Write extended command

In the MCDISK, this command is equivalent to the WRITE EXTENDED command (no additional verification takes place. However, some types of memory such as FLASH will always be verified, even when WRITE or WRITE EXTENDED is used). WRITE VERIFY is implemented for compatibility only.

6.6.24 VERIFY ($2F)

Bit: Byte	7	6	5	4	3	2	1	0
0	$2F (VERIFY)							
1	Logical Unit N°			reserved=0			BC	0
2	Logical Block Address, MSB							
3	Logical Block Address							
4	Logical Block Address							
5	Logical Block Address, LSB							
6	reserved=0							
7	Transfer Length, MSB							
8	Transfer Length, LSB							
9	reserved=0						Flag	Link

Table 6.6.24: Verify command

If the BC bit is set to zero, the MCDISK will check the data integrity on the logical blocks specified. No data transfer takes place.

If the BC bit is set to 1, the MCDISK will do the same as with BC=0, but reference data will be transferred from the initiator to the MCDISK. Note however that the MCDISK does not perform a real verification of the data sent in the current version of the firmware.

6.6.25 MODE SENSE(10) ($5A)

Bit: Byte	7	6	5	4	3	2	1	0
0	$5A (MODE SENSE)							
1	Logical Unit N°			reserved=0				
2	PCF		Page Code					
3	reserved=0							
4	reserved=0							
5	reserved=0							
6	reserved=0							
7	Allocation Length in number of bytes (MSB)							
8	Allocation Length in number of bytes (LSB)							
9	reserved=0						Flag	Link

Table 6.6.25: Mode Sense(10) command

This command is used like the MODE SENSE ($1A) command when the requested pages requires more than 256 bytes. The MCDISK returns about 358 bytes if requesting all pages (PAGE CODE $3F). See the MODE SENSE ($1A) command for description of the fields in this command.
Please note that the MCDISK returns with a CHECK CONDITION status if requesting all pages with the command MODE SENSE ($1A).

7. FORMATTING NOTES

SRAM:

This type of memory is recognized automatically by the MCDISK. There is no need to set up parameters before formatting. Formatting is done within fractions of a second as long as no card test and no filling is set in MODE SELECT PAGE $32.

FLASH:

Most modern FLASH cards (such as Intel Series 2) are also recognized automatically by the MCDISK, as they have CIS information included. Firmware 3.0 and newer also detects size and type of some FLASH cards even if they have no CIS by reading the FLASH chip's JEDEC ID. Some other FLASH cards however will be initially treated as ROM cards of 64 Megabyte size, and cannot be formatted directly. The device type as well as the card size must be set through MODE SELECT PAGE $30 before such a card can be formatted for the first time. If the JEDEC identifier is known, it should be set trough MODE SELECT PAGE $36 (This is possible in firmware 3.0 and newer only, older versions do not allow writing PAGE $36). On first formatting (if not RAW formatted and the CIS-level is set to 3 (see MODE SELECT PAGE $32), CIS is written onto the card, so the card will be recognized automatically like a “modern” FLASH card afterwards.

Note that card test and fill options in MODE SELECT PAGE $32 makes no sense with FLASH EPROM.

FLASH formatting often takes several minutes for large capacity cards, especially if the card has been erased incorrectly before. If FLASH formatting takes a long time but ends with WRITE FAULT condition, the FLASH cells might have been “over-erased” some time. Over-Erasure leads to charge shifts within the FLASH memory cells and might make the device unusable. It might be possible to recover the problem by trying the FORMAT UNIT command many times until it reports GOOD status.

EEPROM:

Like flash cards, some older cards might not identify themselves properly, therefore it is necessary to set the device type (4) and size through MODE SELECT PAGE $30. Note that no standardized EEPROM programming algorithm exists, so the algorithm used by the MCDISK may not work with all EEPROM cards.

OTP:

Like EEPROM, there is no standardized programming algorithm defined for OTP yet, so the algorithm used by the MCDISK may not work with all OTP cards. Not also that OTP programming does not work with all hardware versions of the MCDISK.

PC-ATA:

These cards may take a very long time for formatting (up to 30 min. for some devices). Normally, formatting a PC-ATA card is not required, as it can only be used in raw mode with a fixed block size of 512, no matter how you try to format it. Formatting is therefore only required if the card needs a physical formatting, which is not the case very often.

COPYRIGHT AND REVISION HISTORY

Publication Date : October 1997
This manual reflects firmware version 3.8 of the MCDISK.

DISCLAIMER

The information contained herein is believed to be accurate as of the date of this publication, however, MPL AG will not be liable for any damages, including indirect or consequential, arising out of the application or use of any product, circuit or software described herein.
MPL AG reserves the right to make changes to any product herein to improve reliability, function or design.

FIRMWARE REVISION HISTORY

1.0	First release
1.3	Better FLASH erase
1.4	More support for Fujitsu - type FLASH cards
2.0	JEIDA 4.1 support (INTEL FLASH series 2)
2.1	Added EEPROM writing support, enhanced FLASH erasure
2.3	Added PC-ATA support. Added full PCMCIA Version 2.0 features. Corrected problems with SCSI-II hosts, corrected problems with SCSI transfer errors.
2.4	Enhanced ATA support, enhanced power management.
3.0	Complete redesign of the firmware. Supports two slot operation for MCDISK-D-1 and MCDISK-E, auto-detects FLASH cards without CIS, enhanced speed, allows accessing very slow devices. Supports ATMEL FLASH writing and is prepared for OTP programming.
3.2	Added support for Intel Series 2+ cards and Calluna Card ATA-Disks (only for MCDISK-E)
3.3	Added *Dual SCSI ID* feature, solved problem with Windows NT.
3.4	Added I/O access, Write protect override and nocount bits in Page $30. No RDY/BSY checking in direct access devices anymore
3.5	Added support for OEM Calluna drives. Removed NULLs from Inquiry string. MODE SENSE of all pages now returns only Page 4 if an ATA disk is inserted, otherwise page 5 will be returned, to prevent driver missmatch.
3.6	Added VCC switching support for MCDISK-G and modified MCDISK-E. Added 10-Byte Mode Sense command. Extended card analyzation for DOS formatted cards without CIS. Changed device type codes in inquiry command.
3.7	Limitted ATA transfer size to 255 blocks per command to prevent some ATA disk from hang.
3.8	Added AMD series C and D flash card support.

Our local distributor: