SanDisk
Application Note

Interfacing SanDisk ATA PC Cards and Flash ChipSets in Memory Mapped Mode

SanDisk Corporation
140 Caspian Court
Sunnyvale, CA 94089
TEL: 408-542-0500 FAX: 408-542-0503
URL: http://www.sandisk.com

Lit. No. 80-13-00103 Rev. 1 6/98 *Printed in U.S.A.*

1.0 Introduction

This application note presents design considerations for implementing SanDisk products using Memory Mapped Mode. Memory Mapped Mode is an alternate method of addressing the controller registers and can be used with any SanDisk ATA product including the Flash ChipSet. This mode is preferred in applications where the SanDisk product is replacing socket flash or for embedded designs that use a non-Intel microprocessor. PC Card ATA Memory Mapped Mode does not require an interface chip or socket and card services software for implementation. Please note that card registers names will be capitalized. For more information on how these registers function, please refer to the descriptions in the SanDisk Product Manuals.

Memory Mapped Mode Features:

- Hot swapping without accessing the card's attribute memory to configure the card. This is the product's power on default mode.
- 8 bit and 16 bit access to all card registers is only controlled by CE1 and CE2. True IDE Mode only allows 16 bit access to the data register. If True IDE Mode is selected for 8 bit hosts, a Set Features command must be issued to the product to enable 8 bit data transfers.
- Hardware select of data register with high order address line A10 for host string move execution, thus minimizing code required for data transfer.

2.0 Hardware Implementation

2.1 Required CPU Map

To utilize Memory Mapped Mode, the design must provide unique CE1 and CE2 signals to the controller which can be mapped into a specific address in the CPU's memory space. If 16 bit only mode is desired, CE1 and CE2 can be tied together. If A10 is used to select the data register, the required memory space is 2K bytes. If the data transfer is to be implemented using offset 0 or 8 & 9, the required memory map is only 16 bytes. (See Figure 2-1 Register Mapping.)

2.2 Required Signals

The following signals are the minimum required signals to implement Memory Mapped Mode. (See Figure 2-2 Schematic.)

D15-D0 — This is the data bus which can be either 8 bits or 16 bits depending on CE1 and CE2. All data and commands use this bus.

CE1, CE2 — CE2 always selects the odd byte of the word. CE1 will access the even byte or odd byte depending on A0 and CE2. For 8 bit systems, only CE1 should be used. For 16 bit systems, which access 16 bits always, CE1 and CE2 should be used concurrently. CE1 and CE2 should be decoded by host logic to determine memory window.

OE — This is the output enable strobe generated by the host. It is used to read data from the SanDisk product.

WE — This is the write enable strobe generated by the host. It is used to write data to the SanDisk product.

A3-A0 — Selects basic registers to communicate to the SanDisk product. This requires 16 bytes of host address space. A0 is optional if CE1 and CE2 are combined to enable 16 bit wide register access.

A10 (Optional) — Used to select the data register to accommodate systems with string move instructions. If A10 is high and other control signals select the product, then A3-A0 is ignored.

RDY/BSY (Optional) — This signal is driven low when the product is accessing memory. When it is high, register access is allowed. After a data transfer command is issued, this signal is used to signify that the host can transfer data.

RESET (Optional) — When this signal is high, the product is placed in a reset state. This signal is only valid at power on. If a reset of the product is required after power on, the device control register should be used to issue a soft reset.

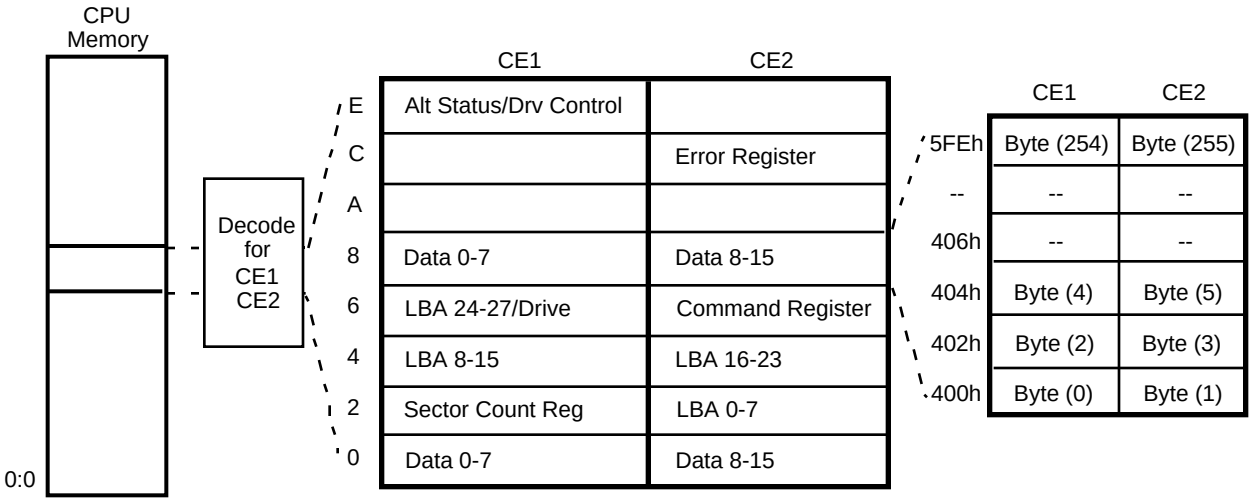

Figure 2-1 Register Mapping

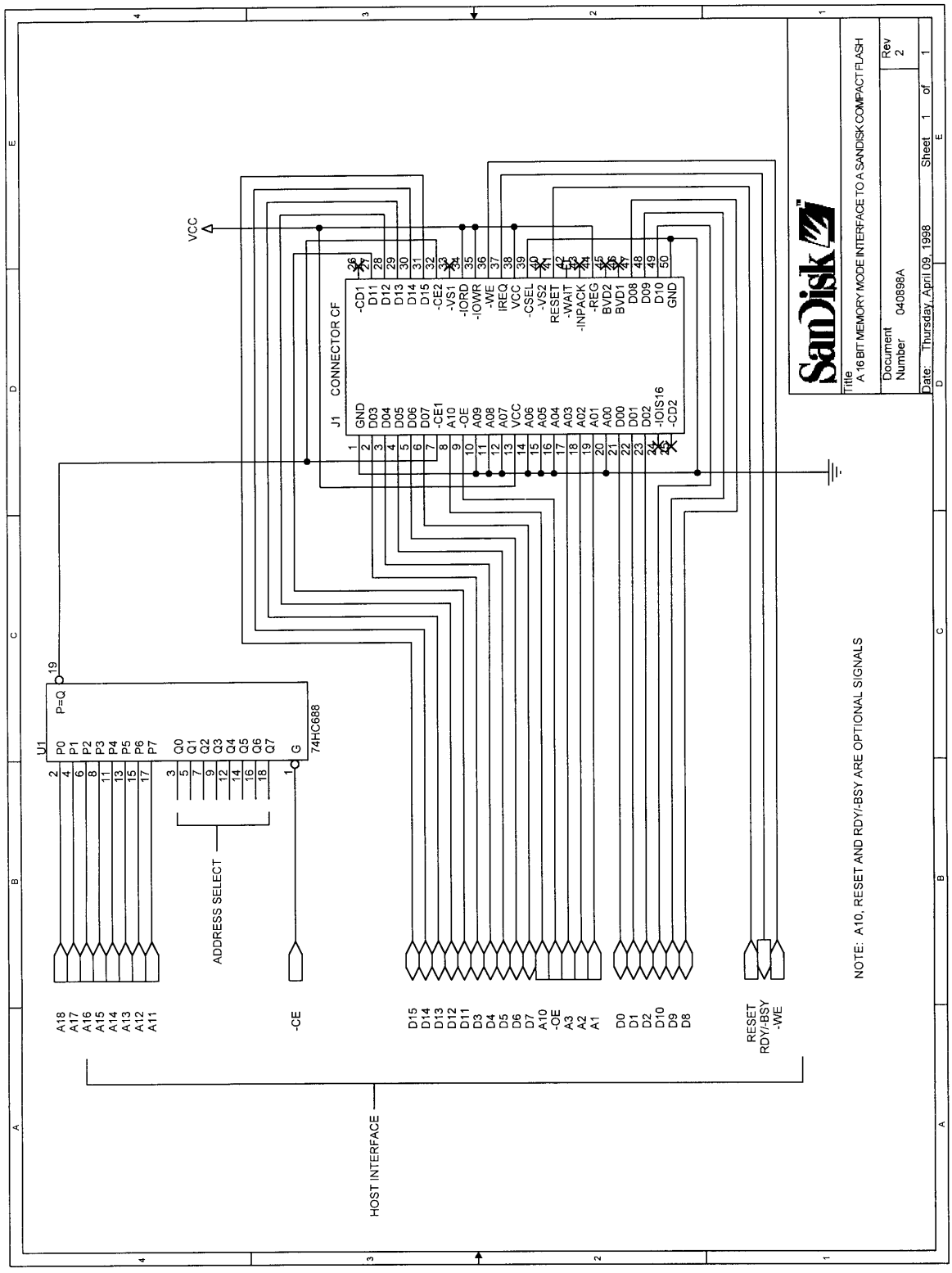

Figure 2-2 Schematic

3.0 Memory Mapped Mode Software Interface and Driver Issues

Memory Mapped Mode is not supported in some of the existing operating systems. Special software may be required to access the product in Memory Mapped Mode. SanDisk's Host Developer's Tool Kit (order number SDDK-01) supports this mode of access.

3.1 Host Memory versus Logical Block Addressing (LBA)

SanDisk products are block mode storage devices with the minimum block size of 512 bytes and the maximum block size of 128 Kbytes (256 sectors). This is normally referred to as a sector. Once a block transfer is started all 512 bytes must be transferred. During the transfer, the data can not be accessed randomly. To access a sector's data randomly, it must be loaded into the host's RAM.

SanDisk products support Logical Block Addressing (LBA) method, which is defined in the PC Card ATA specification. The LBA is an address pointer to the starting block within the SanDisk product's internal memory. The SECTOR COUNT REGISTER defines the number of blocks to transfer at the specified starting block address. The LBA consists of LBA 0 to LBA 27. This allows for 268 gigabytes of address space available. LBA 0-7 is determined by writing OFFSET 3. LBA 8-23 is determined by writing OFFSET 4 and 5. LBA-24-27 is located at OFFSET 6.

3.2 Error Register Handling in 16 Bit Mode

The PC Card ATA specification was derived from the ANSI ATA specification currently used in most x86 systems. The ANSI specification was based on the Intel I/O memory access. This is the same as our True IDE I/O Mode. In this mode, the DATA REGISTER is 16 bits wide (1F0h), and the next I/O address, the ERROR REGISTER (1F1h), is only 8 bits wide. There is an exception with the PC Card ATA, that is specified in our manuals. The primary concern is with systems that implement 16 bit wide access without A0 connected. In this configuration, the ERROR REGISTER is available at OFFSET Dh, instead of OFFSET 1h.

3.3 Data Transfer Sequence

CE1, CE2 and A0 are the signals used to determine how data is transferred to the host. Memory Mapped Mode offers more options for data transfer width compared to the True IDE Mode of operation. If a system only needs 8 bit transfers, then only CE1 is required to transfer on D0-D7 and A0 is used to determine ODD or EVEN byte. True IDE Mode requires a SET FEATURES command to be issued to the card before the data register can be accessed in 8 bit mode. A0 is not used if the host asserts both CE1 and CE2 for all accesses. See your product's SanDisk Product Manual for a detailed description of this relationship.

Once the width of access is determined, there are three different methods of accessing the DATA REGISTER on the card. (This is the ATA Register which is used to actually transfer the data to and from the host) The first method is at the register located at OFFSET 0. The second is to use the duplicate DATA REGISTER located at OFFSET 8 and 9. The third method is to use the optional signal A10, which selects the DATA REGISTER, and ignores A1-A3, only using the CE, and OE or WE signals to clock the data. This method is to allow the host to use a string move command instead of a move byte/word command repeated to transfer the data. (See Figures 3-1 through 3-3.)

TIME X-TO-O
SS.MMM.UUU.NNN
+42.883.965.700

STATE LISTING

RESET
REG#
CE2#
CE1#
IOWR#
IORD#
WE#
OE#
INPACK#
BVD1/STSCHG#
BVD2/SPKR#
RDY/IRQ#
WP/IO16#
WAIT#
USER1
USER2

X-MARK	O-MARK	X-TO-O	TRIG
000016	000128	+000112	NONE

STORE#	EVENT	ADDRESS	DATA HILO	DESCRIPTION	TIMESTAMP
X000016	NO VCC	0000000	0000		21.47 SECX
000017	NO VCC	0000000	0000		21.47 SEC
000018	RDY/IRQ/	000000E	7F50	STS=RDY/DSC	21.40 SEC
000019	COMMON RD	0000000	5000	ASCII=P	0.780 US
000020	COMMON WR	0000000	0000		1.520 US
000021	COMMON WR	0000002	0001	SEC COUNT	1.140 US
000022	COMMON WR	0000004	0000	CYL LO	1.100 US
000023	COMMON WR	0000006	ECE0	DRV/HD=LBA/DRV0/HD=0	1.160 US
000024	COMMON RD	0000008	848A		2.050 MS
000025	COMMON RD	0000008	00F5		0.540 US
000026	COMMON RD	0000008	0000		0.780 US
000027	COMMON RD	0000008	0002		0.380 US
000028	COMMON RD	0000008	0000		0.660 US
000029	COMMON RD	0000008	0240	ASCII= @	0.400 US
000030	COMMON RD	0000008	0020		0.660 US
000031	COMMON RD	0000008	0000		0.380 US
000032	COMMON RD	0000008	3D40	ASCII==@	0.660 US
000033	COMMON RD	0000008	0000		0.380 US
000034	COMMON RD	0000008	2020		0.660 US
000035	COMMON RD	0000008	2020		0.400 US
000036	COMMON RD	0000008	2020		0.660 US
000037	COMMON RD	0000008	2020		0.380 US
000038	COMMON RD	0000008	204D	ASCII= M	0.660 US
000039	COMMON RD	0000008	5A58	ASCII=ZX	0.380 US
000040	COMMON RD	0000008	3030	ASCII=00	0.660 US
000041	COMMON RD	0000008	3439	ASCII=49	0.400 US
000042	COMMON RD	0000008	3133	ASCII=13	0.660 US
000043	COMMON RD	0000008	3436	ASCII=46	0.380 US
000044	COMMON RD	0000008	0002		0.660 US
000045	COMMON RD	0000008	0002		0.380 US
000046	COMMON RD	0000008	0004		0.660 US
000047	COMMON RD	0000008	5265	ASCII=RE	0.400 US
000048	COMMON RD	0000008	7620	ASCII=V	0.660 US
000049	COMMON RD	0000008	322E	ASCII=2.	0.380 US
000050	COMMON RD	0000008	3030	ASCII=00	0.660 US
000051	COMMON RD	0000008	5375	ASCII=SU	0.380 US
000052	COMMON RD	0000008	6E44	ASCII=ND	0.660 US
000053	COMMON RD	0000008	6973	ASCII=IS	0.400 US
000054	COMMON RD	0000008	6B20	ASCII=K	0.660 US
000055	COMMON RD	0000008	5344	ASCII=SD	0.380 US
000056	COMMON RD	0000008	5033	ASCII=P3	0.660 US
000057	COMMON RD	0000008	422D	ASCII=B-	0.380 US
000058	COMMON RD	0000008	3820	ASCII=8	0.660 US
000059	COMMON RD	0000008	2020		0.400 US
000060	COMMON RD	0000008	2020		0.660 US
000061	COMMON RD	0000008	2020		0.380 US
000062	COMMON RD	0000008	2020		0.660 US
000063	COMMON RD	0000008	2020		0.380 US

Figure 3-1 Identify Drive Command State Listing

TIME X-TO-O
SS.MMM.UUU.NNN
+00.011.980.540

STATE LISTING

RESET
REG#
CE2#
CE1#
IOWR#
IORD#
WE#
OE#
INPACK#
BVD1/STSCHG#
BVD2/SPKR#
RDY/IRQ#
WP/IO16#
WAIT#
USER1
USER2

X-MARK	O-MARK	X-TO-O	TRIG
000282	000372	+000090	NONE

STORE#	EVENT	ADDRESS	DATA HILO	DESCRIPTION	TIMESTAMP
X000282	COMMON RD	0000000	0000		1.500 USX
000283	COMMON RD	0000006	50E0	DRV/HD=LBA/DRV0/HD=0	1.040 US
000284	COMMON RD	0000004	0000	CYL LO	0.600 US
000285	COMMON RD	0000004	0000	CYL LO	0.720 US
000286	COMMON RD	0000002	0000	SEC COUNT	0.720 US
000287	COMMON RD	0000006	50E0	DRV/HD=LBA/DRV0/HD=0	0.820 US
000288	COMMON RD	0000000	0000		0.840 US
000289	COMMON RD	000000C	7F50	ERR=	10.46 MS
000290	COMMON RD	0000006	50E0	DRV/HD=LBA/DRV0/HD=0	0.760 US
000291	COMMON WR	0000000	0000		1.520 US
000292	COMMON WR	0000002	0001	SEC COUNT	1.160 US
000293	COMMON WR	0000004	0000	CYL LO	1.100 US
000294	COMMON WR	0000006	20E0	DRV/HD=LBA/DRV0/HD=0	1.160 US
000295	COMMON RD	0000008	0000		1.465 MS
000296	COMMON RD	0000008	0100		0.540 US
000297	COMMON RD	0000008	0000		0.780 US
000298	COMMON RD	0000008	0100		0.380 US
000299	COMMON RD	0000008	0000		0.660 US
000300	COMMON RD	0000008	0100		0.400 US
000301	COMMON RD	0000008	0000		0.660 US
000302	COMMON RD	0000008	0100		0.380 US
000303	COMMON RD	0000008	0000		0.660 US
000304	COMMON RD	0000008	0100		0.380 US
000305	COMMON RD	0000008	0000		0.660 US
000306	COMMON RD	0000008	0100		0.400 US
000307	COMMON RD	0000008	0000		0.660 US
000308	COMMON RD	0000008	0100		0.380 US
000309	COMMON RD	0000008	0000		0.660 US
000310	COMMON RD	0000008	0100		0.380 US
000311	COMMON RD	0000008	0000		0.660 US
000312	COMMON RD	0000008	0100		0.400 US
000313	COMMON RD	0000008	0000		0.660 US
000314	COMMON RD	0000008	0100		0.380 US
000315	COMMON RD	0000008	0000		0.660 US
000316	COMMON RD	0000008	0100		0.380 US
000317	COMMON RD	0000008	0000		0.660 US
000318	COMMON RD	0000008	0100		0.400 US
000319	COMMON RD	0000008	0000		0.660 US
000320	COMMON RD	0000008	0100		0.380 US
000321	COMMON RD	0000008	0000		0.660 US
000322	COMMON RD	0000008	0100		0.380 US
000323	COMMON RD	0000008	0000		0.660 US
000324	COMMON RD	0000008	0100		0.400 US
000325	COMMON RD	0000008	0000		0.660 US
000326	COMMON RD	0000008	0100		0.380 US
000327	COMMON RD	0000008	0000		0.660 US
000328	COMMON RD	0000008	0100		0.380 US
000329	COMMON RD	0000008	0000		0.660 US

Figure 3-2 Read Sector Command State Listing

TIME X-TO-O
SS.MMM.UUU.NNN
+00.016.427.940

STATE LISTING

RESET
REG#
CE2#
CE1#
IOWR#
IORD#
WE#
OE#
INPACK#
BVD1/STSCHG#
BVD2/SPKR#
RDY/IRQ#
WP/IO16#
WAIT#
USER1
USER2

X-MARK	O-MARK	X-TO-O	TRIG
000261	000416	+000155	NONE

STORE#	EVENT	ADDRESS	DATA HILO	DESCRIPTION	TIMESTAMP
X000261	COMMON RD	0000008	0000		0.380 USX
000262	COMMON RD	0000008	0000		0.660 US
000263	COMMON RD	0000008	0000		0.380 US
000264	COMMON RD	000000E	7F50	STS=RDY/DSC	1.760 US
000265	COMMON RD	000000E	7F50	STS=RDY/DSC	0.780 US
000266	COMMON RD	0000002	0000	SEC COUNT	1.480 US
000267	COMMON RD	0000000	50E0	ASCII=P	1.060 US
000268	COMMON RD	0000004	0000	CYL LO	0.600 US
000269	COMMON RD	0000004	0000	CYL LO	0.720 US
000270	COMMON RD	0000002	0000	SEC COUNT	0.720 US
000271	COMMON RD	0000006	50E0	DRV/HD=LBA/DRV0/HD=0	0.820 US
000272	COMMON RD	0000000	FFFF		0.820 US
000273	COMMON RD	000000E	7F50	STS=RDY/DSC	225.8 US
000274	COMMON RD	0000000	50E0	ASCII=P	0.780 US
000275	COMMON WR	0000000	0000		1.500 US
000276	COMMON WR	0000002	0180	SEC COUNT	1.160 US
000277	COMMON WR	0000004	0000	CYL LO	1.100 US
000278	COMMON WR	0000006	30E0	DRV/HD=LBA/DRV0/HD=0	15.72 MS
000279					0.020 US
000280	RDY/IRQ\				271.1 US
000281	RDY/IRQ/	0000008	FFFF		134.7 US
000282	COMMON WR	0000008	FFFF		0.400 US
000283	COMMON WR	0000008	FFFF		0.380 US
000284	COMMON WR	0000008	FFFF		0.380 US
000285	COMMON WR	0000008	FFFF		0.400 US
000286	COMMON WR	0000008	FFFF		0.380 US
000287	COMMON WR	0000008	FFFF		0.380 US
000288	COMMON WR	0000008	FFFF		0.380 US
000289	COMMON WR	0000008	FFFF		0.400 US
000290	COMMON WR	0000008	FFFF		0.380 US
000291	COMMON WR	0000008	FFFF		0.380 US
000292	COMMON WR	0000008	FFFF		0.400 US
000293	COMMON WR	0000008	FFFF		0.380 US
000294	COMMON WR	0000008	FFFF		0.380 US
000295	COMMON WR	0000008	FFFF		0.380 US
000296	COMMON WR	0000008	FFFF		0.400 US
000297	COMMON WR	0000008	FFFF		0.380 US
000298	COMMON WR	0000008	FFFF		0.380 US
000299	COMMON WR	0000008	FFFF		0.400 US
000300	COMMON WR	0000008	FFFF		0.380 US
000301	COMMON WR	0000008	FFFF		0.380 US
000302	COMMON WR	0000008	FFFF		0.380 US
000303	COMMON WR	0000008	FFFF		0.400 US
000304	COMMON WR	0000008	FFFF		0.380 US
000305	COMMON WR	0000008	FFFF		0.380 US
000306	COMMON WR	0000008	FFFF		0.400 US
000307	COMMON WR	0000008	FFFF		0.380 US
000308	COMMON WR	0000008	FFFF		0.380 US

Figure 3-3 Write Sector Command State Listing

3.4 Memory Control Using A10 and RDY/BSY

A10 and RDY/BSY can be used by the host system to allow a state machine to directly transfer the card's buffer memory to the host's memory without host CPU intervention. The state machine would start after the command register is written with a data transfer command. The directional control would be determined on the command issued. The number of blocks to transfer would be loaded from a write to the sector count register.

The protocol, after a command is written, would be for the RDY/BSY signal output from the card to signal a DMA REQUEST to the host. The host would then assert A10 to the card for DMA ACKNOWLEDGE, and the state machine would then generate 256 cycles of CE, with either WE or OE depending on the direction of transfer. After 256 cycles the host would deassert A10 and the card's RDY/BSY signal will go BSY until the next sector transfer. When the sector count register is zero, the host is interrupted that the transfer has completed and the status of the transfer is determined.

3.5 Existing Driver Support

SanDisk's HDTK supports Memory Mapped Mode using 8 bit, 8/16 bit and 16 bit only access methods. This code is written in "C" and has been ported to many industry standard processors.

Microsoft Win CE has a driver to access our cards in Memory Mapped Mode. The HP 3xx series of HPCs uses Memory Mapped Mode for the CompactFlash™ slot.

Microsoft Win 95 and Win NT do not support Memory Mapped Mode for PC Card ATA. There are add on drivers that will support Memory Mapped Mode though.

4.0 ATA Command Set Implementation

For an embedded application, not all the ATA commands would be required. This application note discusses the minimum required commands to access data from the SanDisk device.

4.1 Minimum Required Commands

Identify Drive Command (ECh) — This command enables the host to receive device information such as total number of sectors available to the host.

Read Sector Command (20h) — This command transfers data from the device to the host. The transfer can be from 1 sector to 256 sectors of 512 bytes each.

Write Sector Command (30h) — This command transfers data from the host to the device. The transfer size is the same as the Read Sector Command transfer size.

Request Sense Command (03h) — An extended error code is provided when this command is issued after a normal ATA error.

4.2 Additional Commands

Execute Drive Diagnostics (90h) — This function is done when the device is powered on. For the host to check the device after power on, this command should be issued.

Translate Sector (87h) — This command is useful if the sector information, such as HotCount, is desired. HotCount is the number of writes that the sector has endured.

Power Commands — The power commands are not required in most systems. SanDisk devices will power down after every command, unless the power commands override this.

For application specific, embedded systems, all of the implemented ATA commands need not be supported in the system software. Most of the supported commands are only there for backwards software compatibility and are seldom used.

5.0 HDTK IDE Porting

5.1 IDE Porting Overview

The SanDisk Host Developer's Tool Kit (HDTK) provides a mechanism to access the ATA function in a system. Currently, the HDTK offers support of the FAT File System and several peripheral bus interfaces. The file system and the bus interface are enabled or disabled by just setting a few options. There are also many features built into the HDTK to allow you to take advantage of SanDisk products. The HDTK provides a high level of data management through its FAT File System or low level driver directly accessible to the storage devices. The HDTK works with or without the Interrupt Service routine. To access the hardware, the HDTK needs to know your system specific requirements. This is done through by configuring the file SDCONFIG.H.

5.2 SDCONFIG.H

To configure the HDTK, one must modify SDCONFIG.H. The SDCONFIG.H header file contains many options and system specific definitions that must be provided. Some of these options are compilation options that exist only during compilation to allow the compiler to select certain code. Others will be active at run-time. There are different sections for each peripheral bus interface such as IDE, PCMCIA, SPI and MMC in this file. Most of the time, for a selected configuration, the options are already set. You may need to modify a few options to match your platform for memory mapping or I/O mapping, interrupt driven or not, 16-bit or 8-bit peripheral bus.

There is only one peripheral bus interface selected at one time. The choices are:

- USE_TRUE_IDE
- USE_PCMCIA
- USE_SPI
- USE_MMC
- USE_SPI_EMULATION
- USE_MMC_EMULATION

To select IDE interface, the USE_TRUE_IDE option must be set. Set USE_TRUE_IDE to 1 to use the ATA protocol. Depending on the development platform, memory or I/O mapped mode should be set or cleared respectively.

The File System is enabled or disabled via the USE_FILE_SYSTEM option. Set USE_FILE_SYSTEM to 1 to enable the File System. Otherwise, set USE_FILE_SYSTEM to zero to disable the File System. The two examples below show use with an IDE interface and the File System. To select the IDE interface as a stand alone configuration in Memory Mapped Mode, the SDCONFIG.H must be modified as follows:

```
#defineN_CONTROLLERS     1        /* Use 1 IDE controller in the system */
#defineDRIVES_PER_CONTROLLER1        1        /* Number of drives on first controller */
#defineDRIVES_PER_CONTROLLER2        0        /* Number of drives on second controller */
#defineUSE_FILE_SYSTEM   0        /* Indicate there is no file system */
#defineUSE_TRUE_IDE              1        /* Indicate the IDE interface is selected */
#defineUSE_MEMODE                1        /* Use memory mapped mode */
#defineUSE_INTERRUPT             0        /* No interrupt service. Use polling technique */
#defineUSE_LBA_ONLY              1        /* Use Logical Block Address */
#defineWORD_ACCESS_ONLY          1        /* if 1 access registers as byte-pairs, 16-bit Bus */
#defineUSE_SET_FEATURES  0        /* Disable SanDisk Flash product feature */
#defineUSE_CONTIG_IO             1        /* Use 16-byte contiguous register address range */
```

To configure the IDE interface for use with the File System, the user must modify the SDCONFIG.H as follows:

```
#defineN_CONTROLLERS     1        /* Use 1 IDE controller in the system */
#defineDRIVES_PER_CONTROLLER1        1        /* Number of drives on first controller */
#defineDRIVES_PER_CONTROLLER2        0        /* Number of drives on second controller */
#defineUSE_FILE_SYSTEM   1        /* Indicate the FAT File System is in use */
#defineUSE_TRUE_IDE              1        /* Indicate the IDE interface is selected */
#defineUSE_MEMODE                1        /* Use memory mapped mode */
#defineUSE_INTERRUPT             0        /* No interrupt service.  Use polling technique */
#defineUSE_LBA_ONLY              1        /* Use Logical Block Address */
#defineWORD_ACCESS_ONLY          1        /* if 1 access registers as byte-pairs (16-bit Bus) */
#defineUSE_SET_FEATURES  1        /* Enable SanDisk flash product feature */
#defineUSE_CONTIG_IO             1        /* Use 16-byte contiguous register address range */
```

Other options should be configured to match your system requirements. Please consult the HDTK guide for more information.

For each selected peripheral bus there is a peripheral section to describe all hardware information such as number of IDE controllers, number of drives per controller, controller base address, etc.

In the IDE section, the user must provide the system specific hardware register definitions. The name of the registers and definitions below should not be modified because the code relies on these definitions. Only the values are allowed to change.

In Memory Mapped Mode, the base address of the IDE controller must be specified. Other options should be set to zero if not configured.

```
ATA_PRIMARY_MEM_ADDRESS          0xF0000          /* First memory base address */
ATA_SECONDARY_MEM_ADDRESS        0x00000          /* Second memory base address */
```

After configuring the SDCONFIG.H, the user must provide several routines related to the hardware initialization, interrupt and timer services.

5.3 System Specific Code

The HDTK IDE driver is based on the ATA (AT attachment) specification. Electrical signals and timings of the platform must meet the ATA specification requirement. Also, depending on the system hardware (memory or I/O), all timings related to the Flash device have to be implemented properly.

Most of the time, the HDTK will provide most of the code. Only the portions of the software related to your system need to be implemented. This system specific code is the only code that needs to be written for the specific platform. The HDTK does not provide this access in portable C code. Instead, the HDTK defines several function prototypes to simplify and make the porting easier.