Host Design Considerations: NAND MMC and SD-based Products

Application Note

Version 1.0
Document No. 80-11-00160
September 30, 2002

SanDisk Corporation
Corporate Headquarters • 140 Caspian Court • Sunnyvale, CA 94089
Phone (408) 542-0500 • Fax (408) 542-0503
www.sandisk.com

SanDisk® Corporation general policy does not recommend the use of its products in life support applications where in a failure or malfunction of the product may directly threaten life or injury. Per SanDisk Terms and Conditions of Sale, the user of SanDisk products in life support applications assumes all risk of such use and indemnifies SanDisk against all damages.

The information in this document is subject to change without notice.
SanDisk Corporation shall not be liable for technical or editorial errors or omissions contained herein; nor for incidental or consequential damages resulting from the furnishing, performance, or use of this material.

Lit. No. 80-11-00160 Rev. 1.0, Created on 9/26/2002 1:34 PM

Printed in U.S.A.

Revision History
- *Revision 1.0—initial release.*

Introduction

SanDisk's MultiMediaCard (MMC) and Secure Digital (SD) Card have been designed into a wide variety of consumer electronic products: MP3 players, cell phones, PDAs, digital still and video cameras, data loggers, and more. Although these cards were designed to support this wide range of products, there are many options an engineer needs to consider before designing a card slot into a product. Design considerations include how the end product handles timeout delays, bus type selection, block mode selection, and other options. These can have a major impact on the performance and compatibility of the product. This Application Note will review these options and provide recommendations on the optimum way to manage them.

Timing

There are important timing issues for the engineer to consider when designing products that integrate the MultiMediaCard and/or SD Card.

Timing specifications

Design engineers must meet the rise, fall, setup, hold, and other SD Card and MultiMediaCard bus timing specifications. If they want to support MultiMediaCards in their design, the clock speed should be controllable by the host. This is due to the MultiMediaCard's open-drain mode; the MultiMediaCard powers up in the open-drain mode and cannot handle a clock faster than 400 Khz. Once the MultiMediaCard completes the initialization process, the card switches to the push-pull mode. In the push-pull mode the MultiMediaCard can run at the maximum clock speed.

Refer to www.mmca.org and www.sdcard.org for timing specifications published by MultiMediaCard and SD Card Associations.

Read access and program times

Read access and program times are also very critical to the proper operation of a product design. If the time-out values for read access and program time are not met, data read from and written to the card may be incorrect or invalid. MultiMediaCard and SD Card manufacturers have different read and write time-out values, and the designer must ensure that the product time-out value is not set below the maximum specification.

The maximum read and write time-out values for the MultiMediaCard and SD Card are shown in Table 1.

Table 1. MultiMediaCard and SD Card Maximum Read/Write Time-out Values

Product	Time-out Values	
MultiMediaCard	**Typical**	**Maximum**
Read	(TAAC + NSAC)	10 * (TAAC + NSAC)
Write	(TAAC + NSAC) * R2W_FACTOR	(TAAC) + NSAC) * R2W_FACTOR * 10
SD Card		
Read	(TAAC + NSAC)	100ms
Write	(TAAC + NSAC) * R2W_FACTOR	250 ms

The factors used in calculating the values in Table 1—TAAC, NSAC, and R2W_FACTOR—can be read directly from the CSD register of the MultiMediaCard and SD Card.

The TAAC factor's unit is time, and the NSAC factor has units of 100 clocks. You can convert TAAC units to clock cycles by multiplying by the frequency of the clock and calculate the time-outs in units of clock cycles if desired. Alternatively, given the frequency of the clock, you can convert the NSAC units to time and calculate the time-outs in units of time.

The R2W_FACTOR is a read-to-write factor and has no units. A design engineer can use the time-out values derived from the CSD register to make the design compatible with all MultiMediaCards and SD cards regardless of customer brand.

Interface

The MultiMediaCard and SD Card support multiple busses. Both cards support the 1-bit SPI bus that includes bus pins DATin, DATout, CLK, and CS. The SPI bus is generally found on Motorola and other major MCU manufacturer products.

The SD Card also supports a 4-bit and a 1-bit SD bi-directional bus mode. SD bus pins are CLK, CMD, and DAT in 1-bit mode and CLK, CMD, and DAT[0:3] in 4-bit mode.

The MultiMediaCard also supports the 1-bit bi-directional MMC bus mode that has CLK, CMD, and DAT bus pins. The CMD and DAT pins are bi-directional on the SD 1-bit, SD 4-bit, and MMC 1-bit.

The maximum burst rate achievable with the SD Card and MultiMediaCard depends on the clock speed and bus mode. The burst rate is the data transfer rate between the card's buffer and host.

Table 2. MultiMediaCard and SD Card Clock Speed and Burst Rate

Product	Maximum Clock Speed and Burst Rate	
MultiMediaCard	**Clock Speed**	**Burst Rate**
SPI Bus mode	20 MHz	2.5 MB/s
MMC 1-bit mode	20 MHz	2.5 MB/s
SD Card		
SPI Bus mode	25 MHz	3.125 MB/s
SD 1-bit mode	25 MHz	3.125 MB/s
SD 4-bit mode	25 MHz	12.5 MB/s

The write and read throughput rates of the SD Card and MultiMediaCard are slower than the burst rate because each card includes the busy time to write data from the card's buffers to its internal Flash RAM, and busy time to read data from the internal Flash RAM to the card's buffer. Since most designs use this write and read busy time to complete other processes, choosing a 1- or 4-bit bus mode can have a 4x speed effect on the time spent servicing the SD Card.

The example in Table 3 shows the difference between moving 512 bytes of data to and from a MultiMediaCard or SD Card internal buffer using different bus modes.

Table 3. MultiMediaCard and SD Card Clock Speed and Transfer Time

Product	Maximum Clock Speed and Time Req. to move 512 bytes	
MultiMediaCard	**Clock Speed**	**Time**
SPI Bus mode	20 MHz	204.8 us
MMC 1-bit mode	20 MHz	204.8 us
SD Card		
SPI Bus mode	25 MHz	163.8 us
SD 1-bit mode	25 MHz	163.8 us
SD 4-bit mode	25 MHz	41 us

Read/Write Mode Selection

Another major MultiMediaCard and SD Card design consideration is the use of Singleblock or Multiblock command modes. **Singleblock** mode reads and writes data one block at a time; **Multiblock** mode reads and writes multiple blocks until a stop command is received.

Multiblock mode takes advantage of the multiple internal block buffers present in all MultiMediaCards or SD Cards. In Multiblock mode, when one block buffer gets full during write, the card gives the host access to the other empty block buffers to fill while programming the first block. The card does not enter a busy state until all block buffers are full.

In Singleblock mode, the card enters a busy state by forcing the DAT line low when the first block buffer is full and remains busy until the write process is complete. During the busy state, the host cannot send any additional data to the card because the card forces the DAT line low.

If speed is critical in a design, Multiblock mode is the faster and recommended mode. The more blocks that can be written in Multiblock mode the better the performance of the design. Therefore when planning the design, ensure that enough system RAM is designed in to support the multiblock capability. The performance gain will always outweigh the cost of the extra RAM. However, if speed is not critical—for example, a data-logger design that records only 512 bytes of data every minute—Singleblock mode is more than adequate.

Power and Clock Control

Power control should be considered when creating designs using the MultiMediaCard and/or SD Card. The ability to have software power control of the cards makes the design more flexible and robust. The host will have the ability to turn power to the card on or off independent of whether the card is inserted or removed. This can help with card initialization when there is contact bounce during card insertion. The host waits a specified time after the card is inserted before powering up the card and starting the initialization process. Also, if the card goes into an unknown state, the host can cycle the power and start the initialization process again. When card access is unnecessary, allowing the host to power-down the bus can reduce overall power consumption.

Clock control is another option that should be implemented in a MultiMediaCard or SD Card design. As mentioned in the ***Timing*** section, if the design needs to support the MultiMediaCard, the clock should be lowered to 400 kHz or less during initialization. When the initialization process is complete, the host can raise the clock speed to the card's maximum.

Initialization Algorithm

The initialization algorithm needs to be considered for products designed to support the MultiMediaCard and SD Card or SD Card only. An SD socket is physically thicker which allows both types of cards to be inserted. Therefore, the host needs to be able to detect which card is inserted into the socket.

When the SD initialization command is used first, it causes the MultiMediaCard to return an error that provides the host with an identification of the card type. If the host is supporting both the MultiMediaCard and SD Card, it can continue the initialization using the MMC commands. If the host does not support both cards, it issues an error message instructing the user to insert an SD Card.

If the design uses a MultiMediaCard socket, the host can start the initialization with the MMC command. The host does not need to detect which type of card is inserted because the SD Card will not physically fit into an MMC socket.

File System Support

If a design needs to support a file system, such as SanDisk's Host Developers Tool Kit (HDTK), additional considerations are necessary.

Reading and writing to an SD Card and MultiMediaCard is generally done in 512 byte blocks, however, erasing often occurs in much larger blocks. The NAND architecture used by SanDisk and other card vendors currently has Erase Block sizes of (32) or (64) 512 byte blocks, depending on card capacity. In order to re-write a single 512 byte block, all other blocks belonging to the same Erase Block will be simultaneously erased and need to be rewritten.

For example—writing a file to a design using a FAT file system takes three writes/updates of the system area of FAT and one write/update of the data area to complete the file write. First, the directory has to be updated with the new file name. Second, the actual file is written to the data area. Third, the FAT table is updated with the file data location. Finally, the directory is updated with the start location, length, date and time the file was modified. Therefore, when selecting the file size to write into a design, the size should be as large as possible and a multiple of the Erase Block size. This takes advantage of the architecture.

Some designs update the FAT table for every cluster of the data file written. This can slow the write performance, because the FAT table is constantly being erased and re-written. The best approach is to write all the file clusters then update the FAT table once to avoid the performance hit of erasing and re-writing all the blocks within the Erase Block multiple times.