# Supporting Two SD Devices with a Single SD Bus Host Controller

## Application Note

Version 1.0
Document No. 80-36-00458
October 18, 2005

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

*Lit. No. 80-36-00458 Rev. 1.0 10/05 Printed in U.S.A.*

*Revision History*
*October 2005 Revision 1.0—First draft of initial release*

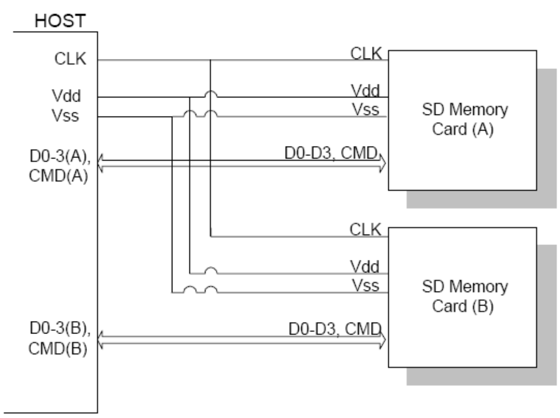
## INTRODUCTION

The SanDisk iNAND Module is a BGA-packaged embedded product with an SD interface that can be soldered directly to the PCB. There has been an increasing demand from host manufacturers that already support one SD device, to leverage their design and add another SD device. For instance, designers could add an SD slot to a host with embedded iNAND. Hosts containing SD slots could have the option of adding an embedded SD-based device such as a SanDisk iNAND Module. This application note will provide options for designers who wish to incorporate an iNAND Module when the host controller does not support two SD interfaces.

## SDA SPECIFICATIONS

According to the SDA specification, one SD bus can only support one SD device. The clock pin can be shared, but DAT [0-3] and CMD lines must be different for every SD device supported. See Figure 1 for more details.

**Figure 1 SD System Bus Topology**




The following section will show you how to support two SD devices using a controller with one bus and two chip-select pins or one bus and two GPIO pins.

## SD DEVICE SUPPORT FOR TWO

To support two SD devices on a single SD bus, multiplex the DAT [0-3] and CMD pins, the CMD pin only, or CLK pin only. The multiplexer must be able to support bidirectional I/O if used with CMD and DAT0-3 pins. Figure 2 shows how the DAT [0-3] and CMD pins can be multiplexed to support the two SD devices. The same results can be achieved by multiplexing only the CMD pin or CLK pin (Figure 3 and Figure 4). The host can access each SD device individually without affecting the state of the other when the multiplexer has separated the buses.

**Figure 2. Multiplexed DAT [0-3] and CMD Pins**

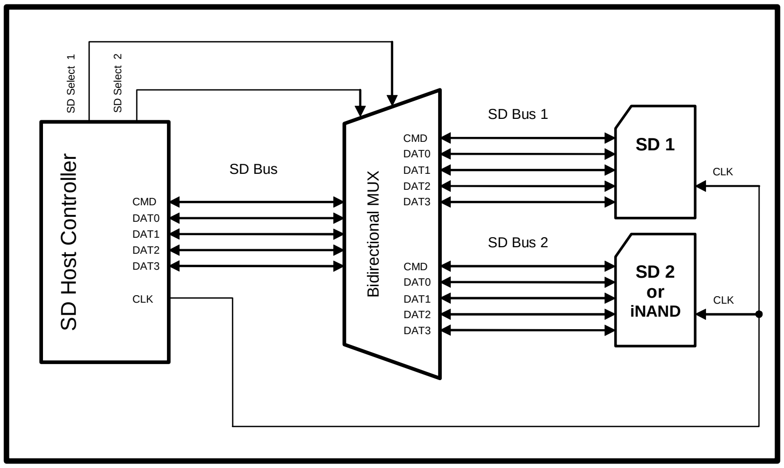



**Figure 3. Multiplexed CMD Pin Only**

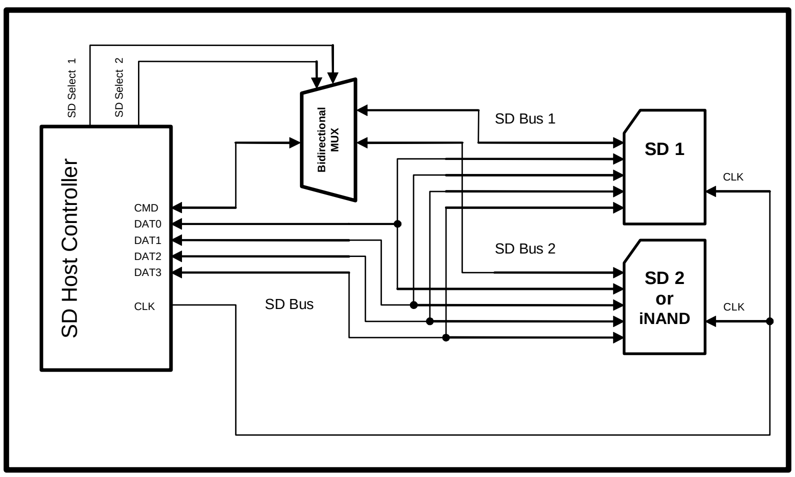
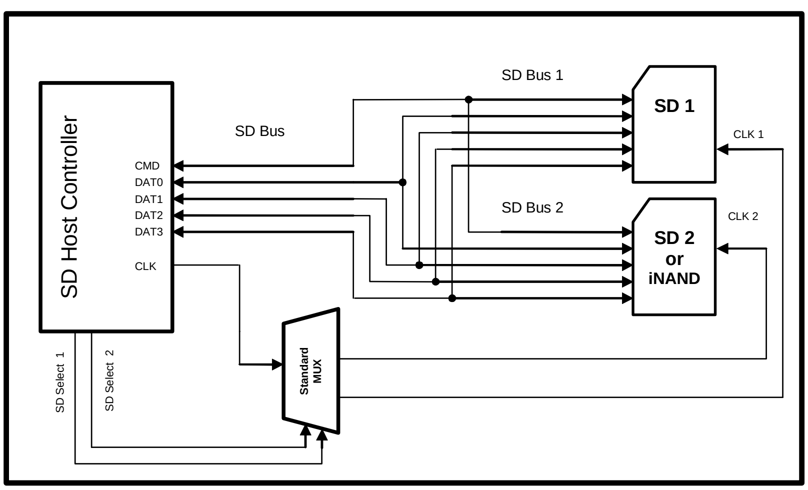



**Figure 4. Multiplexed CLK Pin Only**



## Relative Card Address

Hosts should use two different relative card addresses (RCA) when using a multiplexer to access two SD devices. The SD device's RCA can be changed by issuing CMD3. The host is also responsible for preventing any bus contention on the multiplexed bus when accessing two SD devices.

Before switching from one current SD device to the other, the host must complete the command sequence for the current SD device. The command sequence includes the command, response, and any data transactions if required.

When the sequence is completed, the SD device's CMD and DAT [0-3] pins go into a high impedance state (i.e., Hi Z). It is in this state that the host can access one SD device without impacting the other.

## Software Power Controls

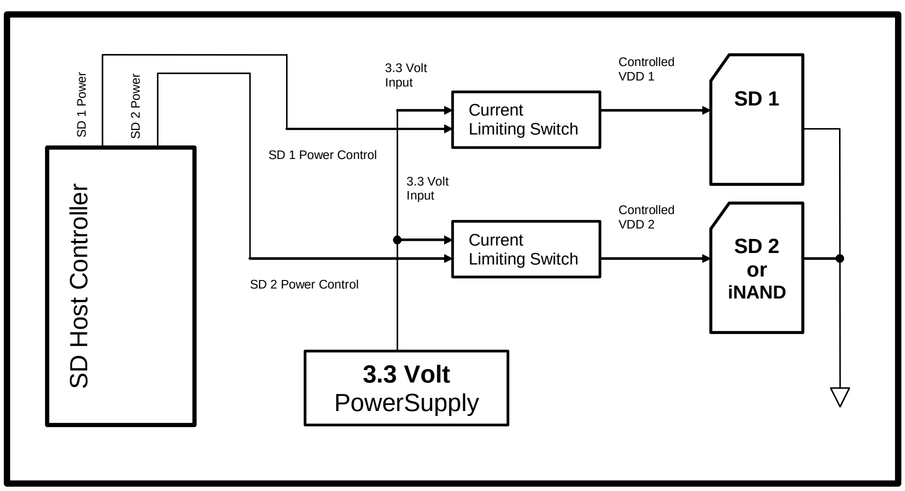
Software power controls can be added to each SD device to add robustness to the device design and ensure compliance with current SDA specifications. Creating a software power control can be accomplished by adding a switch controlled by a GPIO pin. Once implemented, an SD device that goes into an unknown state can be returned to a known state using the power control that enables the host to power cycle the SD device. Otherwise, the entire embedded SD device application must be power cycled or the user must remove and re-insert the SD Card.

### *Inrush Currents*

To comply with current SDA specifications, power must be controlled by software. However if not handled correctly, an **inrush current** may cause a power interruption in a removable SD device on a multiplexed bus. Power interruptions occur because the card's internal capacitance acts like a short and draws a large amount of inrush current while charging. If the current is not controlled, it may overburden the main power supply and cause voltage drops throughout the entire system.

An inrush current can be handled by adding a *Current Limiting Switch* (Figure 5) to control card power-up. If possible, set the switch to limit the inrush current and allow for a slow power-on process. Adding a current-limiting switch is the best way to handle inrush current while remaining within SDA guidelines.

**Figure 5. Adding a Current Limiting Switch**



## CRC ERRORS

Noise, EMI, and other interruptions on an SD bus can cause data errors.  Therefore, implementing the cyclic redundancy check (CRC), available in SD protocol, creates a more robust system less prone to errors.  The CRC enables the host to check all SD device command and data transactions.

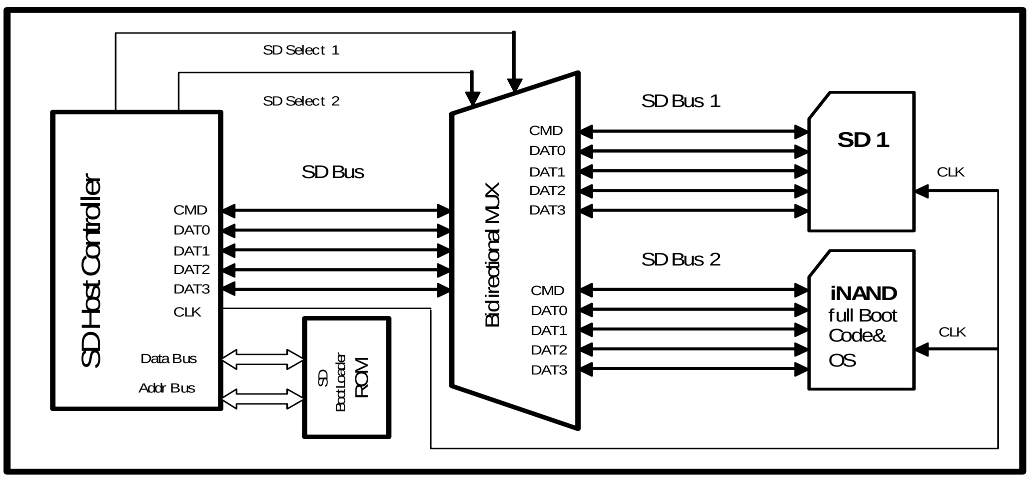
CRC errors may occur when a card is inserted or removed from a card slot. If an error occurs and is detected during a command or data transaction, the host is "advised" to repeat the transaction at least one or two times before power cycling and/or reinitializing the SD device.

## BOOTING FROM INAND

Initially, booting from iNAND will require a small, approximately 4 to 8 KBytes, boot loader ROM (Figure 6).  In the future, we plan to work with controller manufacturers to implement the SD device boot loader inside the controller. By doing this, designers will have the ability to boot directly from an SD device while simultaneously storing user data on the same SD device or external card slot.

The boot loader can be written for controllers that support an SD bus in a multiplexed configuration, or by itself.  Refer to Figure 1, 2, and 3.  Boot loader code using ROM as small as 4 to 8 KBytes can be written for specific controllers which will allow full system boots, including the operating system (OS).

**Figure 6. SD Boot Loader for iNAND**

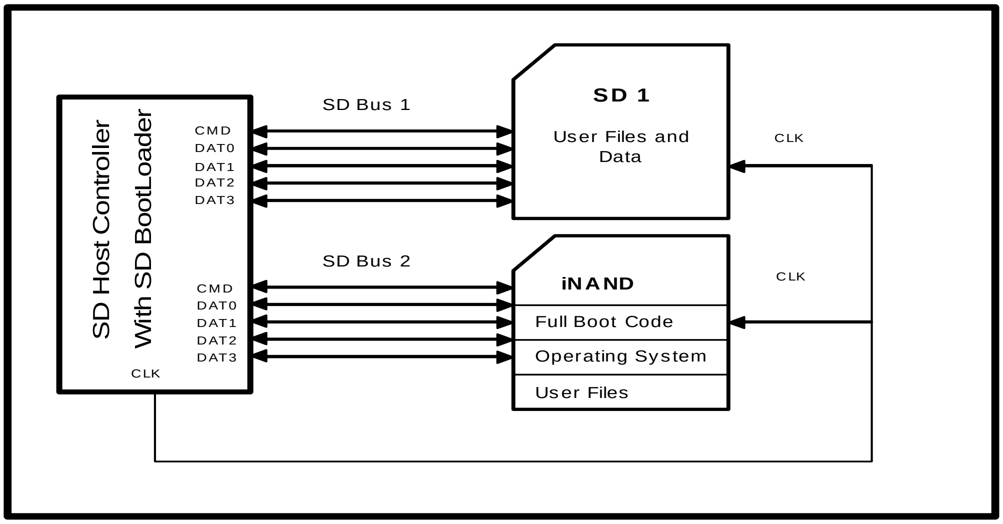



Boot loader code should also be written to initialize the controller and its SD bus; XIP memory; and any other necessary peripherals to facilitate a full boot. After initialization, the controller should have enough intelligence to initialize the SD device that is attached to the SD bus. After both the controller and SD device have been initialized, the boot loader should read the full boot code (including OS if required) from the SD device and move it to XIP memory.

After the full boot code has been moved, the boot loader returns control to the controller, and finishes booting from the XIP memory. Because iNAND and SD devices have a built-in controller to manage FLASH, it is unnecessary to load the controller with any FLASH management software for booting purposes.

Many controllers have internal boot loaders for different devices, including NAND. However, the boot loader described in this document can be considered another option for implementation in future controller designs.

**Figure 7. SD Host Controller with SD Boot Loader**




## SUMMARY

As more and more designers are implementing various SD devices in their products, the need will increase for ways to support more than one SD device with their controllers, and at the same time be able to boot from the embedded SD device, iNAND. This application note has explained how to multiplex the SD bus to support two SD devices, typically an SD card and iNAND. It has also outlined a boot loader solution that uses a small (about 4-8KBytes) of external ROM to accomplish the full boot process. As a long term solution, this same boot loader process can be implemented in future controller designs to accomplish the full boot with no external boot ROM required. This solution will give designers the flexibility to store the boot code and user files in one SD device or use an iNAND for boot code and an external card socket for user data.