Increasing Boot Options with Managed NAND

QuickLogic® White Paper

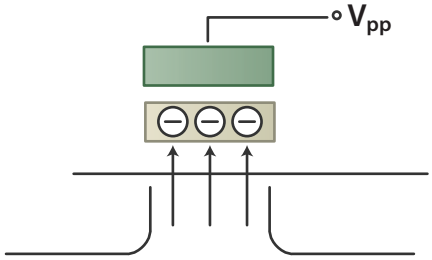
Non-volatile memory has many functions in a processor-based system, but one of the most critical is providing the initial firmware that the processor must use in its system initialization (boot) process. Both NAND Flash, with its interface challenges, and NOR Flash, with its relatively low density, are unsatisfactory for use as a single, non-volatile, bootable system memory solution. Managed NAND offers a third alternative, once the challenge of making it bootable has been solved.

Portable devices such as personal media players (PMPs) and portable navigation devices (PNDs) have at least two requirements for non-volatile semiconductor memory. One need, common to all processor-based systems, is non-volatile program storage. For many systems only the power-up initialization, or boot programs need to be in semiconductor memory. PMPs and PNDs, however, typically need their entire software package, including applications and operating systems, in semiconductor memory. This requires at least moderate memory capacity, from tens to hundreds of Megabytes. In all cases, the memory needs to be compatible with the processor's boot process.

The second need is for user data storage. In the case of PMPs, the data would be music and video files. For PNDs, the data includes maps, location markers and movement histories. What these data types all have in common is large size. User storage capacity of hundreds of Megabytes to Gigabytes is quickly becoming a consumer expectation. Unlike program storage, the data memory can be block-oriented rather than pure random access.

The non-volatile memory of choice in today's device designs is Flash memory. Flash memory stores its data as a charge imposed on a floating gate in a CMOS transistor, as shown in **Figure 1**. The presence of the charge determines the transistor's state, hence the memory cell's data value. Because the gate is floating, with no direct connection to any other circuits, any charge placed on the gate remains indefinitely unless deliberately removed.

Figure 1: Flash memory uses charge stored on a floating gate as the non-volatile storage mechanism.



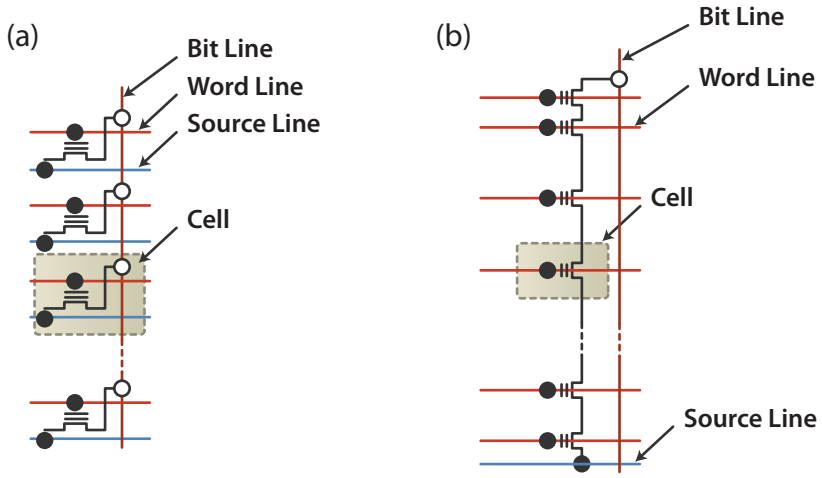
In order to charge or discharge the floating gate, the memory cell must impose a high-voltage signal on the transistor's other gate. Electrons then tunnel to or from the floating gate. This high voltage stresses the insulators and crystalline structures around the floating gate and can eventually damage those structures, rendering the memory cell useless. This wear-out mechanism is common to all forms of Flash memory.

Another attribute common to all Flash memory is the need to erase a memory cell before writing to it. Because a cell's state may not be known before attempting to store data into it, the control logic on Flash memory devices first puts the cell into a known (erased) state before configuring it with data. Thus, every time a cell is written to, it suffers some wear-out damage.

Two Types of Flash

Two distinct Flash memory architectures have arisen: NOR and NAND. The NOR Flash, shown in **Figure 2(a)** has an SRAM-like structure. Each individual bit in the memory connects to the word and bit lines of the device's internal addressing circuitry and the output level is normally high. If a memory cell has been programmed, activating it with the proper word line causes it to pull the bit line low, as with wired-NOR logic.

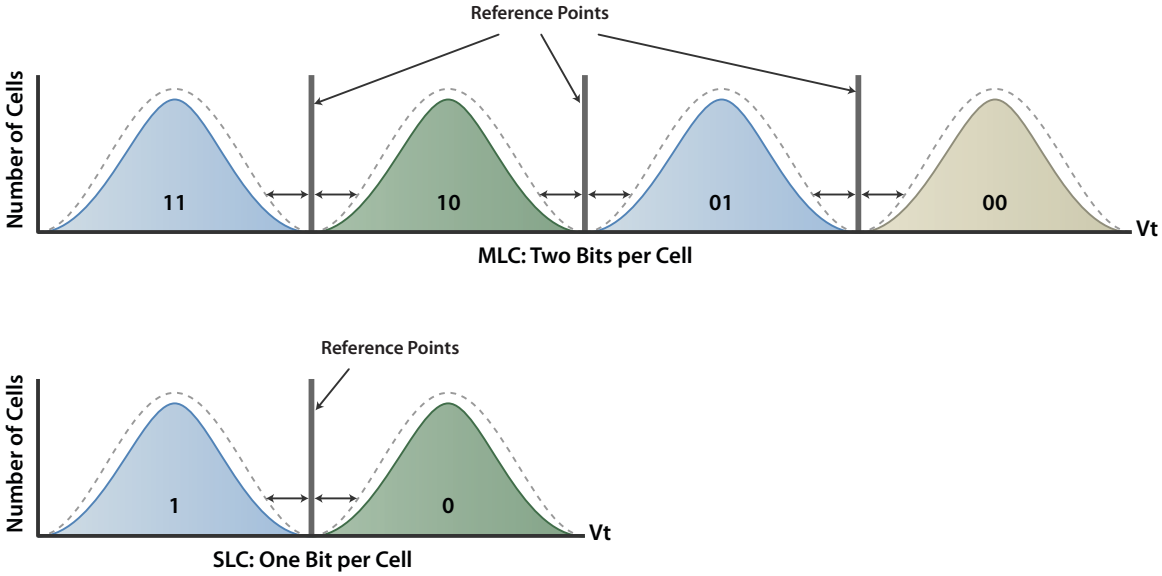
Figure 2: Flash memory comes in two configurations: NOR Flash (a) uses individual bit-line connections to each memory cell while the more compact NAND Flash (b) connects transistors in series to a common bit-line.



The individual bit connections give NOR Flash a relatively simple RAM-like external interface and high read speed. Erasing or writing to the memory, especially in large blocks, takes extra time however. Each bit in the block must be individually erased in order to prepare the block for writing, which can add up to a significant amount of time if the block is large.

The other Flash architecture – NAND – connects memory bits in series, as shown in **Figure 2(b)**. The transistors in the series are normally "on," keeping the bit line low. Activating a programmed transistor with the word line will make it turn off, breaking the series connection and allowing the bit line to go high. This configuration offers slower read speeds due to the impedance of the series connections. Eliminating most of the bit line connections, however, makes NAND cells more compact than NOR cells, allowing NAND memories to achieve higher densities. Another density boost for NAND Flash has appeared in the form of multi-level cell (MLC) circuit topology. In this approach, each bit location stores one of four voltage levels on the floating gate, allowing the cell to encode two bits of data instead of one, as shown in **Figure 3**.

Figure 3: NAND Flash is available in a high-density option that encodes two bits per cell using multi-level logic circuits.



The price for this increased density is a more complex interface. Used primarily for solid-state storage and file systems, NAND Flash uses a block-oriented storage structure rather than byte-oriented addressing. NAND Flash also uses a multiplexed address and data interface to reduce its I/O pin count. As a result, NAND Flash does not simply connect to a processor's external bus like SRAM. Often, it requires a specialized interface.

Regardless of the architecture, however, Flash memory devices all need some management in order to maximize their utility to designers. These functions include bad bit/block identification, error detection/correction and wear-leveling. Wear-leveling is especially important for maximizing the usable lifetime of a Flash memory device. If designers allow write/erase activity to be concentrated in one section of memory, that section will wear out, while the rest of the memory has plenty of life left. Wear-leveling algorithms shift that activity around in order to prevent any one address from wearing out quickly.

Because the wear-out mechanism is a function of how long the cell is exposed to high voltages, Flash vendors have also developed complex erase/write timing algorithms to ensure reliable programming and erasure with minimum exposure. These algorithms, along with all the other memory management functions, are unique to each vendor. As a result, developers typically implement Flash memory management in software.

Flash in PMP/PND Designs

Despite these complexities, Flash memory has earned a place in PMP/PND designs. These designs require high-capacity data storage for detailed large-area maps and multimedia files along with bootable program storage, both of which must be non-volatile. In addition, memory for these designs must be low-power and low-cost in order to meet the market demands for long battery life and reasonable pricing. Flash memory meets all of those needs.

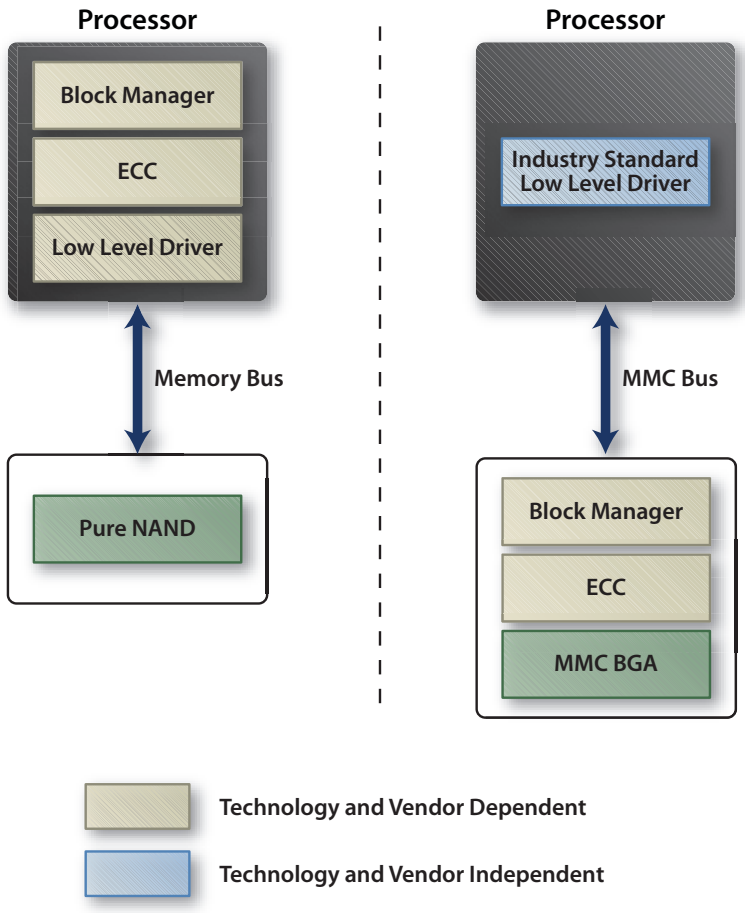
Unfortunately, no one type of Flash memory meets all of those needs well. The memory density of NOR Flash is too low, so the cost per byte is too high for it to serve as data storage and still meet product price requirements. The NAND Flash interface is complex, so NAND Flash cannot typically serve as bootable

program storage because the processor needs intelligence (in the form of software) in order to access the memory. The need for memory management further complicates the use of NAND Flash and MLC NAND makes things even worse.

As a result of these limitations, most PMP/PND designs use both types of Flash: the NOR for bootable program storage and the NAND for mass data storage. This practice adds extra chips, hence extra cost, to the design. Further, it requires some processor overhead for handling the NAND Flash interface and memory management functions.

While raw Flash memories do not meet all the design's requirements, there is an option that will: Managed NAND. Managed NAND devices include an on-chip controller that handles the vendor-specific algorithms for wear-leveling and write/erase timing, as well as the memory management functions of the NAND memory, as shown in **Figure 4**. This eliminates the software overhead associated with using NAND memory and allows Managed NAND devices to offer a standardized, vendor-agnostic interface to the system processor. Two interface standards are available: the Multimedia Memory Card (MMC) and the Secure Digital (SD) card interfaces.

Figure 4: While raw NAND Flash memory needs vendor-specific interfaces, Managed NAND uses a built-in controller to present an industry-standard interface to the processor.



The challenge with using Managed NAND in a PMP/PND device design is making it bootable. Because the standard interfaces are not SRAM-like, processors need some kind of adaptor in order to connect to them for memory access. Some processors have SD/MMC interfaces built in, but those that do have them do not have them located at the boot address. A purely software-based interface cannot be used for boot operations because the system needs to be able to access the memory in order to load the software. Any external hardware-based interface would need to be configured before the processor can boot as well.

Booting from Managed NAND

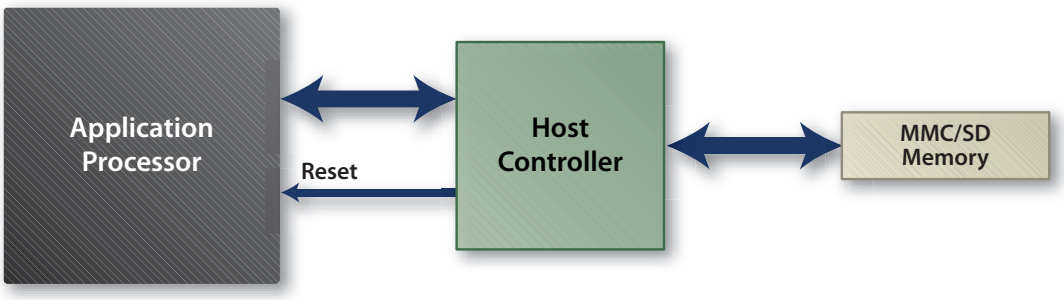
The solution to boot from Managed NAND thus needs two parts: a hardware component and a software component. The hardware component bridges the processor and the Managed NAND to make the memory look like simple ROM to the processor. The software component is an initial program loader (IPL) routine that performs minimal CPU initialization and transfers the boot loader code from the Managed NAND to the processor. Working together, these two components can take the system past the initial chicken-and-egg situation of needing software in order to load the software, allowing the Managed NAND to serve both as program and data storage.

QuickLogic has implemented a boot interface to Managed NAND using its SDIO/MMC Host Controller device. This device offers configurable interfaces, making it adaptable for use with most processor families, as well as for use with SD, SDHC and MMC Managed NAND memories. For processors already possessing these interfaces, the SDIO/MMC Host controller expands their capacity as well as offering boot capability. The configurable interfaces also allow the device to connect with a CE-ATA hard disk drive (HDD) using the same interface as for the memory card, giving designers an additional storage option. The SDIO/MMC Controller has on-chip RAM available, which allows it to hold the IPL code for the processor to access during the boot process.

The QuickLogic implementation is a tested, proven solution. It has demonstrated that it can perform an entire boot sequence under both Linux and Windows® CE operating systems using a Marvell® PXA270 processor, as well as on an TI OMAP platform. It has been tested with a variety of different SD, SDHC, and MMC Managed NAND devices and has built-in autodetection of the memory interface to perform the proper device initialization sequence. The solution is also portable to other application processors from companies such as Samsung, Freescale, Renesas and others.

Booting from Managed NAND imposes both hardware and software requirements on a design. The hardware requirements, shown in **Figure 5**, include a connection to the boot chip select line coming from the processor as well as control of the reset line. The reset line control allows the Host Controller to prevent the CPU from operating until the Managed NAND boot interface is ready. The boot chip select line's memory interface configuration is also reconfigured after power-up to support the use of a READY signal to gate access to memory.

Figure 5: QuickLogic's SDIO/MMC Host Controller controls the processor's reset line to give itself time to extract an IPL program from Managed NAND to begin the boot process.



One software requirement is the development of an IPL program specific to the processor being used. The RAM capacity of the SDIO/MMC Host Controller, 2048 bytes, dictates that the IPL program be limited to 512 instructions, assuming a 32-bit processor. The IPL code thus must be optimized for the CPU and architecture it is to support. While 512 instructions is a small program space, it is an adequate size, especially if the processor also has on-chip RAM available for holding stacks and variables. The processor's on-chip RAM is also useful for storing and running the boot loader program later in the process.

The IPL code is not the only software that must be adapted to the processor and architecture used in the design. The system's Boot Loader code must be adapted so that the processor executes the code out of its internal RAM rather than external ROM. It must also support reading the code through the Managed NAND memory interface. Similarly, the operating system will require drivers to be able to use the SDIO/MMC Host Controller to its full capability once the system has booted.

Three-Stage Boot Process

With both the software and hardware elements in place, the boot process follows a three step operation.

1. At power-up, the Host Controller keeps the processor in a reset state while it downloads the IPL code from Managed NAND into its internal RAM. It then maps its internal RAM to serve as SRAM to the processor under control of the processor's boot address chip select and releases the CPU from its reset state.
2. Once released, the CPU begins executing the IPL from the Host Controller's RAM. The IPL code uses the Host Controller to load the Boot Loader code into the processor's internal SRAM. The IPL then jumps to that SRAM address to begin the execution of the Boot Loader.
3. The Boot Loader initializes system SDRAM, and then loads the operating system from the Managed NAND into system SDRAM. Once the OS is loaded, program execution jumps to the OS, concluding the boot process.

After the boot process is complete, the remainder of the Managed NAND device (the portion not containing the boot software) can be accessed by the OS as a storage device within the file system. The amount of the Managed NAND device required to contain the boot software (the IPL, Boot Loader and OS image) is very small – on the order of 64 Mbytes. The Managed NAND device can be partitioned and formatted so that the remaining majority of the device can be accessed as user storage, while the 64 Mbyte boot portion of the Managed NAND device is protected and kept secure by design.

Using an approach such as this gives PMP/PND device designers many benefits. To begin with, it simplifies board design and saves parts cost. Using Managed NAND for both program and data storage eliminates the need for a separate NOR device. In addition, the interface to Managed NAND uses fewer I/O lines, requiring only 6 lines versus 16 or more, for a conventional memory interface. These I/O resources are then available for other uses in the design.

Since the Managed NAND offers standard, vendor-agnostic interfaces, designers have more supply options open to them. They can go with MMC or SD memory devices, both of which have JEDEC-standard pinouts, and have a wide range of suppliers from which to choose. Unlike raw NAND Flash, there is no vendor-specific software required in the design.

The QuickLogic SDIO/MMC Host Controller can also connect to multiple MMC- and/or SD-based Managed NAND devices simultaneously, as the controller can be customized to include a multiplexing structure for multiple SD devices or can be compatible with a multi-load MMC bus. This allows designers to mix and match memory suppliers and densities in their system designs to minimize bill of materials cost while precisely satisfying design requirements. For example, in a PND device, design requirements may call for 3 Gbytes of storage to accommodate map and point of interest data. To satisfy this requirement, a designer would typically choose a 4 Gbyte Managed NAND device, but with the QuickLogic SDIO/MMO Host Controller solution, the designer now has the option of choosing a less expensive two-chip approach of 2 Gbytes + 1 Gbyte.

Using Managed NAND also helps future-proof their design as increasing demands for memory capacity move storage requirements above 4 Gbytes. The standard interfaces of Managed NAND hide the complexity of multi-level circuits, allowing designers to double memory capacity simply by changing to the new technology. The configurable nature of the SDIO/MMC Host Controller also ensures that developers can move to SDHC (high capacity) memories as needed. Such a move would be complicated when using raw NAND Flash because the higher density devices use block-oriented addressing rather than the byte-oriented addressing that processors expect. File structures for high-density devices also move, from FAT16 to FAT32, further complicating adoption of high density. The QuickLogic Complete Connectivity Solution for Managed NAND boot accommodates all of these changes with the same hardware design, ensuring that designers can migrate to higher memory capacities with minimal effort.

Managed NAND thus makes an excellent match to the system requirements of PMP/PND device designs. It offers the high density benefits of NAND Flash with the boot capability of NOR Flash in a single memory device. It also provides a vendor-agnostic interface, increasing supply options, and reduces software overhead as compared to raw Flash memories.

QuickLogic's configurable SDIO/MMC Host Controller is an ideal solution for system developers as well. Its capacity and flexibility gives designers their choice of processor and memory bus interface types while providing additional I/O to the system along with the bootable Managed NAND function. It also helps future proof designs by providing access to higher-density memory with minimal system changes.

Booting from Managed NAND thus provides a compelling third option to the two types of Flash memory currently being used in PMP/PND designs. It offers cost savings, design simplification, and the capability of handling increasing memory densities. It also provides the ability to mix and match vendors, eliminating sole-source supply, and to mix and match memory densities using multiplexed SD or multiple-chip MMC designs. All these benefits translate into faster design times for current and future device generations, as well as reducing BOM costs in the highly dynamic and competitive mobile market.

Contact Information

Phone: (408) 990-4000 (US)
(905) 940-4149 (Canada)
+(44) 1932 57 9011 (Europe)
+(86) 21 6867 0273 (Asia – except Japan)
+(81) 45 470 5525 (Japan)
E-mail: info@quicklogic.com
Sales: www.quicklogic.com/sales
Support: www.quicklogic.com/support
Internet: www.quicklogic.com

Revision History

Revision	Date	Originator and Comments
A	May 2007	Judd Heape and Kathleen Murchek
B	June 2007	Judd Heape and Kathleen Murchek
C	July 2007	Judd Heape and Kathleen Murchek

Copyright and Trademark Information