

Introduction

Today's system designers must continually find new and creative ways to increase performance yet reduce space, power and cost. Non-volatile code storage, and in some cases, data storage has traditionally been handled by Parallel Flash memories. Emerging Serial Flash memories, with their low pin-count and small packaging, can provide designers with a superior alternative to ordinary Parallel Flash for code and data storage requirements. As a result, the use of Serial Flash is gaining momentum. Diverse applications such as graphics cards, hard drives, printers, wireless networking, set-top boxes, DVD drives, DSL modems and other applications are turning to Serial Flash to reduce controller pin count, board space, power consumption, noise, and overall system costs.

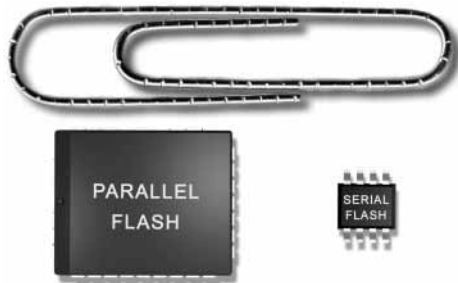


Fig 1. Serial Flash packages are less than 25% the size of a Parallel Flash

Pins and Packaging

The fundamental difference between Serial Flash and Parallel Flash memory has to do with interface pins and packaging. Serial Flash memories use the popular four-pin Serial Peripheral Interface (SPI) and are typically housed in cost effective eight-pin SOIC or MLP (QFN) packages that use less than 25 percent the space of a Parallel Flash package (see Figure 1). The MLP packages are 40 percent thinner than the SOIC and accommodate larger die sizes so a similar footprint can be maintained across higher densities. Serial Flash densities range from 512K-bit to 128M-bit but only the most advanced 0.18 micron technologies can fit higher densities (>4M-bit) into small eight-pin packaging. Although first generation Serial Flash packages varied between manufacturers, newer devices are standardizing with common pin-outs and, in some cases, instruction compatibility.

Parallel Flash memories typically require 22-44 address, data and control pins and are commonly packaged in 32-pin PLCC and 40 to 56-pin TSOPs. Due to historical reasons, the architectures of standard processors or microcontrollers must boot and execute code from a Parallel interface. Some controllers use embedded Flash to alleviate the external overhead. However, embedded Flash adds complexity and cost to IC fabrication and is rarely available on the most advanced logic processes. To address this dilemma, many system-on-a-chip ASIC controllers are moving away from the old Parallel Flash interface and are now using Serial Flash for code storage.

Serial Flash for Code Storage

Serial code storage, also referred to as code shadowing, is a design technique that initially appeared with complex graphics and hard drive controllers and has now expanded to a broad range of high performance applications. These controller-based systems were often downloading code upon power-up from Parallel Flash memory to faster RAM, a function that Serial Flash can handle more efficiently and cost effectively.

Figure 2a shows a block diagram of an application specific controller with a high-speed RAM (RAM, DRAM or SDRAM) interface and a Parallel Flash memory. Due to interface compatibility and performance issues the Parallel Flash is often separate from the RAM interface and requires 17 to 22 address lines, an 8-bit or 16-bit data bus and three to six control lines. This is a considerable overhead for most application specific controllers.

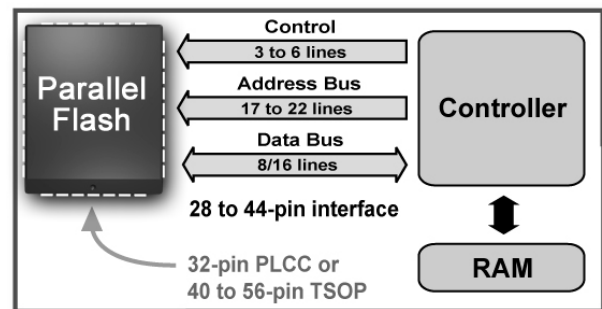


Fig 2a. Parallel Flash is a significant overhead for controllers

Figure 2b shows an application specific controller using a Serial Flash memory with a four-pin SPI interface for serial code storage. In this scenario the controller boots from internal ROM, or directly from the Serial Flash. Upon power-up the code is transferred from the Serial Flash to internal or external RAM using a single read command. Code is then executed from the

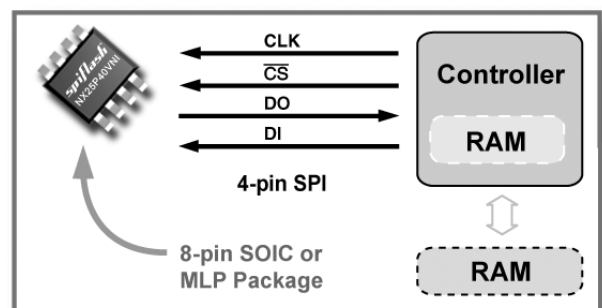


Fig 2b. Using a four-pin SPI Serial Flash, code can be downloaded to RAM at power-up.

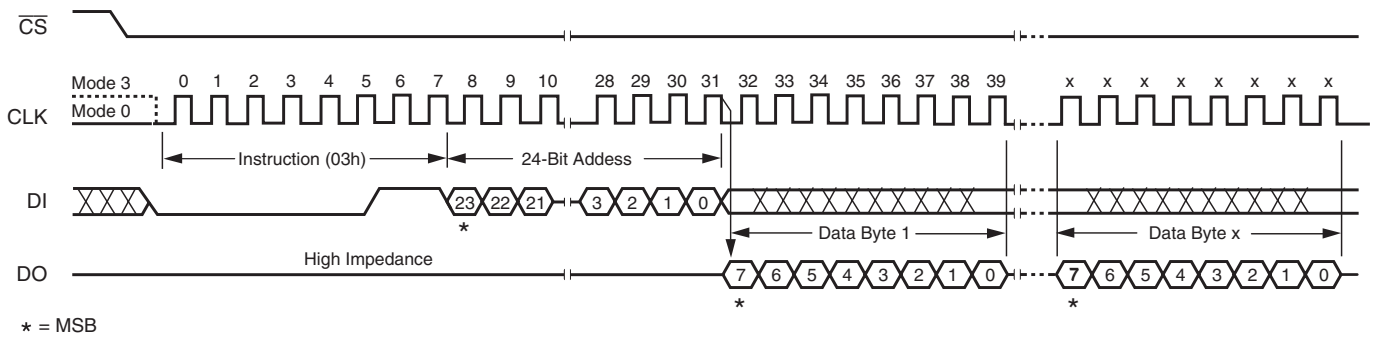


Fig 3. After a four-byte read instruction is issued, data from the SPI Serial Flash can be continuously clocked out of the device.

RAM. Code segments can also be transferred dynamically as needed after power-up. Considering that the system can not operate until the code is downloaded, a fast clock speed is critical. Some Serial Flash devices offer SPI read clock speeds as fast as 33MHz allowing 1M-bit of code to be transferred in 31 milliseconds.

Systems that use this technique can realize significant benefits. Removing the parallel Flash interface frees controller pins for other purposes or may enable the controller to fit into a smaller and lower-cost package. The tiny and low-pin-count Serial Flash packaging allows the PCB to be smaller with fewer traces, further reducing costs. Transient switching noise, common with parallel address and data busses, is eliminated. Additionally, Serial Flash tends to consume 25 to 50 percent less power during read than Parallel Flash and, once the code is downloaded, Serial Flash can be placed into low power (microamp) stand-by to further minimize supply requirements. In those cases where Parallel Flash was originally used for code execution, switching to serial code storage and executing out of RAM can also increase performance. Additionally, code can be compressed in the Serial Flash and decompressed when downloaded for greater storage capacity.

Data and Voice Storage

Besides serial code storage, Serial Flash memories are also suitable for data and voice storage applications, especially in microcontroller based systems. Most popular microcontrollers have an SPI port making it easy to interface to Serial Flash. Selecting the right Serial Flash device for data/voice storage depends on the how often the writing takes place and the size of the data to be written. For applications that are more read-intensive and with few erase/write events, a Serial Flash memory with sector erase (multiple pages) and page write (typically 256 bytes) can be a cost effective and space efficient solution. Examples of read intensive data/voice storage include look-up tables, text, displays, waveforms and pre-recorded voice prompts. If the erase/write event are frequent, and the data size is small, then a Serial Flash device with page erase or an SRAM buffer may be best. However, these features can increase the die size of the Serial Flash resulting in higher costs and larger packages.

FPGA Configuration

Another use for Serial Flash is FPGA configuration. Most FPGA manufacturers offer non-volatile serial configuration PROMs with an interface that is slightly different than an SPI Serial Flash. Although the FPGA configuration PROMs may offer features like cascading or JTAG, they tend to be considerably more expensive than standard Serial Flash memories. Using a small inexpensive PLD, the SPI interface can be converted to download to the FPGA upon power-up. The process requires a single four-byte instruction to be issued to the Serial Flash after which the data can be continuously clocked like an FPGA configuration PROM. Once the FPGA is configured, additional memory in the SPI device can be used for other application purposes within the system.

SPI Instructions

Controlling a Serial Flash memory is easy with simple instructions for read, erase, program, write enable/disable and other functions. All instructions are serial-shifted by means of the four SPI pins which include clock (CLK), chip select (\overline{CS}), data in (DI) and data out (DO). Optional write-protect and hold pins are also commonly provided. The write-protect pin is used for additional hardware protection of the memory. The hold pin allows the device to be suspended during an instruction, which can be useful if multiple devices are sharing the same SPI bus. These pins can simply be tied high if not used. Figure 3 shows a clocking diagram of a read instruction that is common among many Serial Flash memories. After chip select is asserted low, a single read instruction byte (03hex) is clocked into the device followed by a 24-bit address. Data can be continuously clocked out of the Serial Flash from the starting address until chip select is raised.

Conclusion

Serial Flash offers designers an attractive alternative to Parallel Flash and other non-volatile memory solutions. Four-pin SPI, small packaging and other features help reduce controller pin count, board space, power consumption, noise, and overall system costs. As more application specific controllers use Serial Flash for code storage, it is conceivable that standard processors and microcontrollers will also migrate towards this architecture in the future. For more information visit www.nexFlash.com and learn about NexFlash's spiFlash™ family of Serial Flash memories.