

Operating a NAND Flash Device Through an FPGA

You can use FPGAs to access mass storage with NAND Flash.



by Ryan Fisher
Applications Engineer
Micron Technology, Inc.
rfisher@micron.com

As product capabilities continue to expand, so does the demand for high-density static memory storage. NAND Flash is being used for media storage in a large number of systems, including digital cameras, USB stick drives, and portable music players. NAND Flash memory is prominently positioned to address these and other device needs and is evolving rapidly to meet the ever-growing demand.

The most direct approach for a host or system to use a NAND device is by using a NAND controller. The NAND controller can be internal; built into the application

processor or host; or incorporated in designs as an external, standalone NAND controller chip.

An alternative method involves utilizing FPGA resources that already exist in many systems. With FPGA resources, you can create a state machine to act as a NAND controller.

To demonstrate this principle, Micron Technology has developed a NAND controller using a Xilinx® Spartan™-3 FPGA. In this article, I'll focus on the high-level principles of how NAND Flash devices operate.

NAND Flash – The Basics

Discrete programmable non-volatile memory has traditionally been reserved for operating systems (and a few embedded

applications of these systems). NAND Flash devices have changed that by providing memory densities that are orders of magnitude greater than older discrete non-volatile memory. As such, NAND has one of the lowest prices per bit for discrete programmable non-volatile memory. This makes its storage resources available to other applications and functionalities that may have been otherwise too costly to implement using lower density non-volatile memory.

NAND Flash devices employ a state machine control structure to carry out various operations (READ, PROGRAM, ERASE, RESET, READ ID, READ STATUS) common throughout NAND Flash manufacturers (although some specialty commands exist). Because the base set of

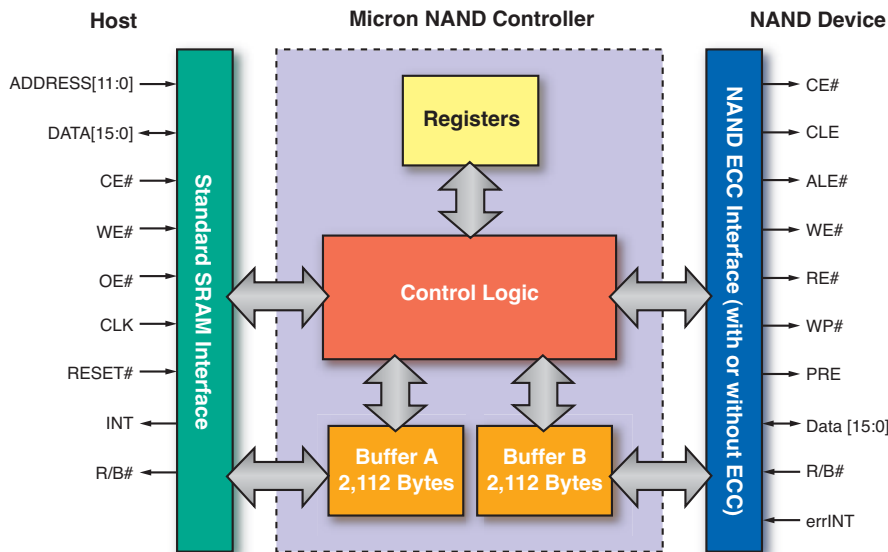


Figure 1 – Block diagram of FPGA NAND controller between host and NAND devices

commands are identical, you have the flexibility to use NAND devices from different manufacturers in the same product.

NAND Flash devices also have a highly multiplexed data/address structure that allows for a low active pin count. NAND devices have not changed their TSOP package pinouts since the introduction of the 64 Mb density, and with NAND devices now at the 8 Gb density range, this allows easy footprint migration from one memory density to another.

The combination of low price per bit, shared command structure, and signal pinouts makes NAND a powerful solution for mass storage use.

Implementing a NAND Controller with an FPGA

Micron has developed a NAND controller built with a Spartan-3 device that sits between the NAND Flash and the system host, which is illustrated in Figure 1. Most systems have an SRAM-type interface, which makes this setup of the FPGA between the NAND device and the host device easy to implement.

The host device, FPGA, and NAND device may be connected point-to-point. A pull-up resistor is needed on the R/B# line on the NAND side to provide the proper functionality for that signal. (See the 2 Gb

NAND Flash datasheet at www.micron.com/products/nand/massstorage/partlist.aspx for more details on this and other NAND Flash signals.)

Using the FPGA, we are able to perform READ, PROGRAM, ERASE, RESET,

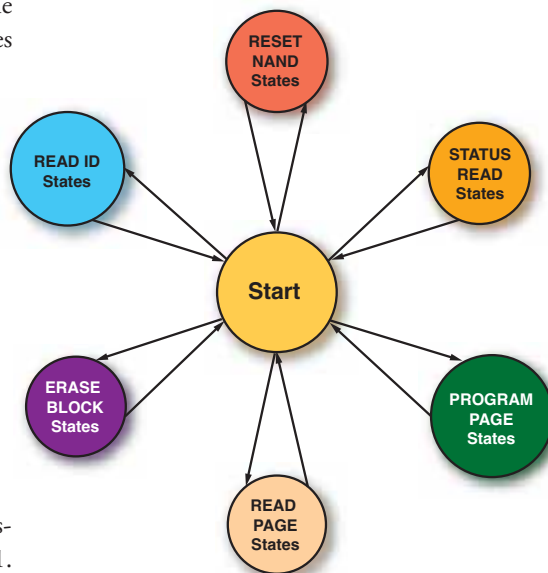


Figure 2 – FPGA NAND controller state machine diagram

READ ID, and READ STATUS operations to the NAND Flash device. Although we used a Spartan-3 FPGA for the implementation of the NAND controller, you could use any number of FPGAs in this process.

FPGA Controller State Machine

NAND Flash devices operate through an internal state machine that controls the various commands of the NAND device (READ, PROGRAM, ERASE, RESET, READ ID, READ STATUS). Many of these commands are common between NAND Flash devices from different manufacturers, allowing for part-to-part compatibility. This allowed us to set up a controller that had the flexibility to communicate with the internal state machine of the NAND Flash device, which could also be used across many NAND device densities independent of the manufacturer.

Figure 2 illustrates the state machine we created on the Xilinx Spartan-3 device to communicate with NAND Flash devices. Beginning with the “start” state, we branched off to the different functions of the NAND controller state machine to operate the NAND Flash. Using these functions, we operated the NAND Flash device as a mass storage device.

Conclusion

Mass storage capabilities are no longer considered “nice to have” in end products; they are becoming a necessary feature. NAND Flash devices are a useful solution to meet the demand for high-density static mass storage. Designs that do not have an embedded or external NAND Flash controller have the disadvantage of not being able to use this resource.

By using FPGA resources already available to many platforms that do not have NAND Flash controllers, you can gain access to this valuable resource by implementing an FPGA NAND Flash controller.

You can find more information about the FPGA NAND Flash controller discussed in this article, including VHDL code, in the technical note “Micron NAND Flash Controller via Xilinx Spartan-3 FPGA (TN-29-06),” available on the Micron website at www.micron.com/products/nand/massstorage/technote.html.

For more information about Micron’s NAND Flash products, visit www.micron.com/products/nand/massstorage/partlist.aspx.