Unlocking the Secrets of Bypass Mode

Application Note

July 2003

The following document refers to Spansion memory products that are now offered by both Advanced Micro Devices and Fujitsu. Although the document is marked with the name of the company that originally developed the specification, these products will be offered to customers of both AMD and Fujitsu.

Continuity of Specifications

There is no change to this document as a result of offering the device as a Spansion product. Any changes that have been made are the result of normal documentation improvements and are noted in the document revision summary, where supported. Future routine revisions will occur when appropriate, and changes will be noted in a revision summary.

Continuity of Ordering Part Numbers

AMD and Fujitsu continue to support existing part numbers beginning with "Am" and "MBM". To order these products, please use only the Ordering Part Numbers listed in this document.

For More Information

Please contact your local AMD or Fujitsu sales office for additional information about Spansion memory solutions.

Publication Number **22281** Revision **A** Amendment **0** Issue Date **November 1, 1998**

Unlocking the Secrets of Bypass Mode

Application Note

This document explains the functionality provided by Unlock Bypass Mode and demonstrates the advantages (and disadvantages) of using Bypass Mode to perform bulk programming operations.

AMD Flash Programming Operations

Programming of an AMD Flash memory device is accomplished through a 4-bus-cycle sequence. This sequence involves writing two unlock commands to the device. First, the data 0xAA (hex) is written to Flash offset 0xAAA. Second, the data 0x55 is written to offset 0x555. This two cycle unlock sequence enables the Flash state machine to accept the program command that follows.

The software driver then writes the program command 0xA0 to offset 0xAAA. The next bus cycle contains the data to program into the device, as well as the desired offset. The Flash state machine then writes the data using an Embedded Algorithm. Once complete, the Flash device returns to reading array data.

The purpose of Bypass Mode is to speed the programming operation by removing the unlock requirement before programming a byte (or word, depending on the operating mode of the Flash). When the Flash is placed into Bypass Mode, only the two bus cycles of the program command are required to program any location in the Flash, which reduces the total software/hardware overhead to perform a programming operation.

Bypass Mode is not available on all AMD flash devices. Only low voltage devices manufactured using AMD's 0.35 µm process technology (or smaller) contain this feature. Devices containing a "B" at the end of the density designation signify this process technology (for example, Am29LV800B).

Note: *The "B" designator should not be confused with the Bottom Boot designation. Older boot block devices will also contain a "B" at the end of the density designation, to signify that the device has a Bottom Boot sector organization. New 0.35 µm (or smaller) devices will contain an additional 'B' in addition to the Boot Sector organization (for example, Am29DL800BB or Am29DL800BT).*

Unlock Bypass Command

Bypass Mode is a three-bus-cycle operation. First, a two-bus-cycle unlock sequence is written to the part (detailed in the previous section), then the data 0x20 is written to the part (address is don't care). While in Bypass Mode, only the Unlock Bypass Program and Unlock Bypass Reset commands are valid. The Flash device will reject all other programming (or bus) sequences.

To program a byte (or word), the two bus cycle Unlock Bypass Program command is written. It consists of a single bus write cycle of 0xA0 (address don't care), followed by the program data and desired offset.

Overall system bus cycles are reduced from four cycles to two. From a system bus standpoint, overall byte (or word) programming bus traffic is reduced by 50%.

To exit from Bypass Mode, the system must issue the two bus cycle Unlock Bypass Reset command, performed by writing 0x90 (address don't care) to the device. Writing of data 0x00 (address don't care) to the device completes the Reset command, returning the part to reading array data.

Advantages and Disadvantages of Unlock Bypass Mode

Use of Bypass Mode is most appropriate whenever large amounts of data are bulk programmed into a Flash device. This occurs most often when the Flash device is first programmed during manufacturing. Using Bypass Mode can provide a measurable overall programming time reduction when the entire device is programmed.

To illustrate this principle, consider the following equation:

$$\text{Overall time to program a single byte/word} = (N * t_{bus} * \text{Cycles}_{write}) + t_{WHWH1}$$

where N = Number of bus cycles required by Flash Device (4 or 2, depending on Bypass Mode)

t_{bus} = Flash bus clock period

Cycles_{write} = Number of system/processor bus cycles required for a write operation

t_{WHWH1} = Time to perform Embedded Program Algorithm (listed in AMD datasheet)

For example, an Am29LV800BB-90EI (8 Mbit) device is examined, in conjunction with a 12 clock write cycle processor operating at 33 MHz. Using typical datasheet parameters, the theoretical time required to program a single byte (or word) programmed in-system would be:

(4 * 12 * 30 ns) + 9 µs = 10.44 µs/byte

Using Bypass Mode, the result becomes:

(2 * 12 * 30 ns) + 9 µs = 9.72 µs/byte

This results in a 6.9% overall theoretical reduction in programming time, or 720 ns per byte. Thus, for this 8 Mbit device, the overall time reduction is 0.75 seconds (or 750 ms).

Note: *The above calculations assume zero wait states, zero bus access latency, and zero system margin, cached instruction fetch).*

When bulk programming using a commercial programming device, typical bus cycle latency is much higher (this is due to the internal bus latency of a high voltage programming device, and the internal delay of its state machine). Given bus cycle timings in a microsecond resolution, the observed time savings can be significantly higher.

Assuming a 1 µs bus cycle (a bus operating with single cycle write timing, with a clock period of 1 µs), the above equations become:

(4 * 1 * 1 µs) + 9 µs = 13 µs/byte

Using Bypass Mode, the result becomes:

(2 * 1 * 1 µs) + 9 µs = 11 µs/byte

This results in an average 15% theoretical reduction in byte programming time, or 2 µs per byte. For an 8 Mbit device, the overall time reduction is 2.1 seconds.

Note: *Commercial programmers vary widely in operational characteristics. Individual programmer results will vary widely from theoretical calculations.*

The actual time reduction is dependent on several factors, including processor speed, wait states, instruction fetch cycles (which may be essentially zero for processors with cache), flash command write instruction execution cycles, bus speed when memory does not reside on the same processor bus, etc. In a typical system environment the actual time for programming an 8 Mbit Am29LV800BB-90EI Flash device was measured in both normal and byte unlock modes. The system environment was comprised of:

- AMD 486 DX4-120 processor with 16K cache enabled.
- 120 MHz processor core, 40 MHz local bus, 33 MHz PCI bus
- Wait state setting of 3-1-1-1

The Flash memory was located in PCI bus address space, and the Flash device was operating in 16-bit word mode.

Normal mode programming (in-system) of the entire 8Mbit device required 15.21 seconds and Bypass Mode required 15.1 seconds, yielding an overall savings of 1% (or 11 ms). Results on specific systems will vary widely given the above system considerations outlined above.

The same test was performed using a Data I/O PSX series programmer, in both normal and Bypass Mode. Time to program the same Am29LV800B device was reduced from 96 seconds to 56 seconds, resulting in a 27.4% decrease in overall programming time.

There are some minor disadvantages to operating in Bypass Mode. Since all command sequences other than the Unlock Bypass Reset and the Device Reset commands are disabled during this mode, it is necessary to exit the bypass mode before performing any other operation. This can complicate system programming, since commands such as the Autoselect sequences, Sector Erase, or Erase Suspend are no longer accepted. Care must be practiced when entering Bypass Mode—preparations such as possibly disabling interrupts, and performing all required Sector Erasures prior to Bypass Mode programming may be appropriate.

Conclusion

Unlock Bypass Mode can be used to significantly decrease bulk programming of AMD Flash devices, especially in high bus latency systems such as commercial programmers. In systems utilizing low latency, cached bus operations the time decrease is measurable, but less significant.

In general, as the number of required processor write cycles is increased, and as the speed of a write operation decreases, the benefit (in terms of decreased overall device bulk programming time) of Unlock Bypass Mode increases.