Atmel's Self-Programming Flash Microcontrollers

By Odd Jostein Svendsli, Marketing Manager

Summary

The third-generation Flash microcontrollers from Atmel are now available. These microcontrollers offer self-programming Flash. The microcontroller can write to its own program memory when running code from the boot sector. This creates new possibilities for the designers using self-programming memory-based microcontrollers. Atmel's microcontrollers with integrated Flash program memory provide increased system flexibility by enabling field programmability, data storage and remote upgrades.

Table of Contents

Introduction

Atmel's microcontrollers with integrated Flash program memory provide increased system flexibility by enabling field programmability, data storage and remote upgrades. Flash memory retains its content even after power is turned off, and can be electrically erased. Other common memory types are Mask ROM which is a fixed content memory that cannot be changed once the parts have been fabricated, and OTP EPROM technology that can be user-programmed one time using specialized external equipment.

Evolution of Flash Microcontrollers

Atmel introduced the first 8-bit Flash microcontroller in 1993. This first-generation Flash microcontroller is based on the industry-standard 8051 core, and uses a separate high voltage supply for programming. The user benefits from Flash microcontrollers are that the devices can be programmed just before system assembly, and if a software bug is found, or if a change is needed, the devices can be removed and reprogrammed. This eliminates the scrap of obsolete material associated with Mask ROM or OTP devices.

Atmel's second-generation of Flash microcontrollers were introduced in 1997. These microcontrollers are based on the AVR® architecture, and include In-System Programmable and single-voltage Flash memory. As a result, no additional programming voltages are needed. There are several additional benefits offered by this technology. It is now possible to assemble the printed circuit board and program the microcontroller before the software has received final approval. If a software bug is found, it is possible to reprogram the microcontroller without removing it from the printed circuit board. Some applications include a serial number that can be read from firmware. In-System Programmable Flash microcontrollers allow the serial number to be programmed into the Flash memory at the very last step on the production line. In-System Programmable Flash makes it possible to segment the program into verification and application software. The benefit of this approach is that the program space requirement decreases and the system can be assembled and tested before the application software is complete. This approach reduces development time and decreases time to market.

The third-generation Flash microcontrollers from Atmel are now available. These microcontrollers offer self-programming Flash. The microcontroller can write to its own program memory when running code from the boot sector. This creates new possibilities for the designers using self-programming memory-based microcontrollers. Here are a few of the benefits of self-programming memory. The scheme used to reprogram the Flash is flexible and can be adapted to the algorithm that best fits the test programming equipment. This feature allows more users to take advantage of the benefits of In-System programming. Remote upgrades are possible without any additional external components. If the communication channel is already a part of the system, updating the Flash program memory from a remote location can now be done without adding any extra components. Many applications need a microcontroller with external EEPROM to store large tables of data that are rarely modified. These applications can now use the self-programming memory to store the data, saving the cost and reducing board space by eliminating the external EEPROM.

Technology Alternatives for Self-Programming

There are several alternative approaches, which can be used to implement Flash-based microcontrollers with self-programming memory. The differences include the size of application and boot block, size of the sector that can be erased and reprogrammed, the design and configuration of the boot sector and the implementation of security to protect the code against and accidental reprogramming.

Impact of Sector Size

Determining the optimum sector size is a trade-off between programming time and flexibility. The benefit of a large sector size can be short programming time, for bulk erase and reprogramming since a large part of the memory is erased and reprogrammed in parallel. Small sector sizes are more efficient if a small part of the memory needs to be reprogrammed, or if a data array stored in Flash needs to be updated. A disadvantage of having a large sector size is that a larger RAM block is needed to store the data that will be programmed into the Flash sector. Larger RAM blocks yield larger die size adding cost to the microcontroller itself.

ROM or Flash Boot Block

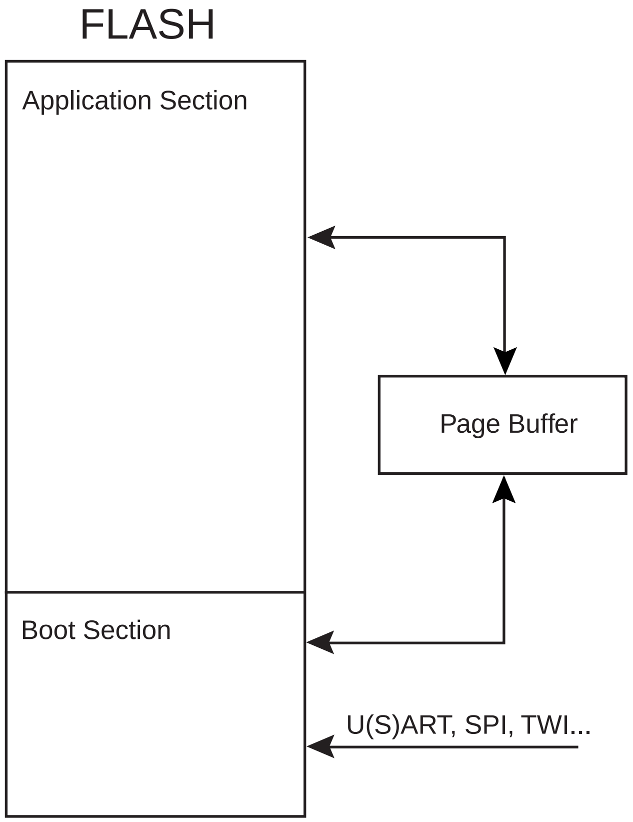
There are several approaches for implementing a secure boot block. Flash and Masked ROM are the two most common. Implementation of a boot block using Read-Only Memory (ROM) requires the memory to be programmed during wafer fabrication. The ROM boot block can be addressed by toggling external pins, or by entering the boot block from the application program. The advantage of ROM is that the block is defined in silicon and cannot accidentally be erased. This is true when the device uses multiple memory technology such as ROM for the boot block and OTP or Flash for program memory. The disadvantage of ROM is that the boot functions that are implemented during the manufacturing process cannot be changed or modified. A more flexible approach is to use Flash for the boot block. Configuration bits can be set so the user can select program execution to start in the boot block, or at the reset vector. The program can be written to jump to the boot block at any time. The advantage of using a Flash boot block is to increase user's flexibility. Several different communication channel options can be used for receiving data to write into the program memory. It is possible to use the boot-block to store application code if a part of the boot block is not used. If the vendor during the component test process programs the boot block, the device will have all the benefits of the ROM solution, but with more flexibility since it can be changed.

Security

Protecting against accidental reprogramming or unauthorized access to the program memory is accomplished by setting lock bits, which protect the application sector, boot block or both. Additionally, it is possible to implement encryption or decryption schemes in memory to provide a higher degree of protection. To protect against memory corruption due to an unstable power supply, Brown-out protection can be implemented using either the on-chip Brown-out reset protection circuit or by using discrete external devices.

Atmel's Approach

The recently announced megaAVR® series of microcontrollers incorporate the most advanced technology for self-programming memory.



Boot Block Size

The size of the boot block of the Atmel megaAVR series of microcontrollers is programmable between 256 bytes and 4 Kbytes. The smaller sizes are utilized when a simple self-programming scheme is used. The larger boot block sizes give the users the possibility for adding encryption/decryption algorithms. The DES algorithm will fit into a 2-Kbyte boot block in an AVR microcontroller.

Read while Write Programming

Atmel's megaAVR series of microcontrollers allows users to execute instructions from the boot block memory while doing a sector erase or writing in the application program memory. As a result, the microcontroller can continue responding to critical interrupts even during a programming event. To make this easier to use, the devices support the possibility to move the interrupt-vectors from the application section to the boot block section under software control. In some self-programming microcontrollers, the timing of the programming session has to be maintained under software control. Because of this, interrupts cannot be enabled during a programming session since an interrupt will change the timing causing the memory cell to be altered. In the megaAVR series of microcontrollers, the self-programming sessions are self-timed, eliminating this risk.

Conclusion

Self-programming memory simplifies and cost reduces designs that require software upgrade after the system has been installed. In the past, this function required the use of a second microcontroller, which was used to control reprogramming and to maintain critical functions during programming. Flash microcontrollers with self-programming memory eliminate the need for a second microcontroller. Reprogramming can now be accomplished using a single-chip solution. Self-programming memory can benefit applications that are difficult to access, such as air conditioning controls, deep-sea applications or even space applications such as satellites. Additionally applications with many microcontrollers work together in a large distributed system where it is not feasible to update each subsystem one-by-one such as fire-detectors in a large building will derive a major benefit from self-programming Flash-based microcontrollers.

References

1. ATmega128 Datasheet, Web: http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf

Editor's Notes

About Atmel Corporation

Founded in 1984, Atmel Corporation is headquartered in San Jose, California with manufacturing facilities in North America and Europe. Atmel designs, manufactures and markets worldwide, advanced logic, mixed-signal, nonvolatile memory and RF semiconductors. Atmel is also a leading provider of system-level integration semiconductor solutions using CMOS, BiCMOS, SiGe, and high-voltage BCDMOS process technologies.

Further information can be obtained from Atmel's Web site at www.atmel.com.

Contact: Odd Jostein Svendsli, Marketing Manager, Trondheim, Norway,

Tel: (+47) 72 88 43 88, e-mail: oj@atmel.com.