Microcontrollers—An Integral Part of Everyday Life
Look around and you can see microcontrollers everywhere—inside computers, cars, TV sets, mobile phones, MP3 players, washing machines, microwave ovens and phone cards.

Flash in a Pan? Not So
All of these microcontrollers have integrated memory modules, mostly Flash (a programmable, non-volatile memory), which enables field programmability, remote data storage and prototyping. The cost of Flash memory has decreased so much that it is now used in volume production and is not just for prototyping anymore. Flash has many benefits; Flash:
> Is electrically erasable and does not require ultraviolet light
> Provides end-of-line customizing for regional variations in consumer demands
> Provides software-enabled intelligence to address changing legislation
> Supports remote diagnostics and preventative maintenance
> Reduces code obsolescence, which saves on scrapped-product cost
> Standardizes platforms, which reduces product variability
> Eliminates sockets and rework with in-system programmable Flash
> Eliminates the need for external EEPROM by using EEPROM emulation
> Allows cost-effective programming changes and field software upgrades via in-application programmability and re-programmability
> Allows extremely fast programming times of Flash memory; reduces production programming costs through ultra-fast programming
> Allows programming across full operating supply voltage with no extra programming voltage; is useful in cost-effective, reprogrammable battery-powered applications
> Helps reduce time to market with application re-programmability
> Enables the user to define preferred settings with improved write/erase and data retention performance
> Helps to protect code from unauthorized reading and guards against unintentional erasing/writing of user-programmable segments of code with flexible block protection and security

Flash is here to stay.
Need for Security
These days hacking into hardware and software systems can be a full-time profession or hobby for some; therefore, there is an increasing demand for the security of application systems. A typical secure system has different levels of design with security features implemented at each level. The uppermost level is usually related to communications protocols and human interfaces. The applications software level supports the external interfaces and communications protocols and may also perform encryption and user authentication. This may be controlled by the software code program (for example, embedded inside the Flash of a microcontroller), which has built-in security features for authentication, encryption and protection of sensitive information.

Microcontrollers are versatile enough to be used not only for control, but also for protection purposes. Some applications that require secure code in microcontroller Flash memory include bank payment cards, mobile phones, laptops, industrial control applications, wireless communication applications, pay-TV applications and building access control applications. Additionally, secure code in Flash can also be used for blocking non-genuine and refilled cartridges for printers and ensuring servicing of home appliances only by authorized service centers or manufacturers. Security applications increasingly include electronic sensors and locks using Flash microcontrollers.

How Secure Is My Code Inside the Flash Memory?
Given enough time and resources, a determined hacker can break any protection. The overall security of a system is determined by the least secure element. When designing a secure system, a reasonable goal is to make the process of breaking the design expensive and time consuming. One of the first steps in any hardware design is choosing a microcontroller with secure Flash protection features.

Flash Security in Freescale 8-bit Microcontrollers
Freescale Semiconductor is the Flash microcontroller industry leader. Freescale’s 8-bit microcontrollers include circuitry to prevent unauthorized access to the contents of Flash and RAM memory, which store the application code. Following is a brief description of some of the salient security features of Freescale’s 8-bit microcontrollers.

Engaging/Disengaging Security
Security is engaged or disengaged based on the state of two nonvolatile register bits. Register bits states of 1:0 imply disengaged security, while all other combinations below mean that security is engaged in the microcontroller.

1:1 Engages Security
0:1 Engages Security
0:0 Engages Security

When security is engaged, Flash and RAM are considered secure resources. Some registers within the microcontroller (i.e., direct-page registers, high-page registers) and the background debug controller are considered unsecured resources. Programs executing within secure memory have normal access to any memory location and resources within the microcontroller. Attempts to access a secure memory location with a program executing from an unsecured memory space or through the background debug interface are blocked. That is, write commands are ignored and read commands return all 0s.

The on-chip debug module is designed so that it cannot be enabled while the MCU is in “secure” mode. The separate background debug controller can still be used for background memory access commands.

How Flash-Protected Blocks Add to Programming Safety
Block protection prevents the protected Flash region from accidental program or erase changes. Block protection is controlled through the Flash protection register.

In-application Flash programmability does not need two non-volatile elements. If the programming algorithm is contained in Flash, part of the algorithm could be to copy the algorithm itself into RAM. The programming algorithm could then be executed from RAM. This is a reliable method if the microcontroller can execute from internal RAM. But, in the event of a power brown-out or a complete power failure, the internal RAM will be completely erased. Once the programming algorithm operating from RAM has erased the Flash module, the window of liability is open until the programming of the Flash is complete.
Write-Protected Flash Blocks

The previous scenario of RAM corruption due to power supply problems can be overcome by having “protected blocks” in the Flash memory. You achieve this with a small “write-protected” block within the Flash array itself. This protected block can contain the programming algorithm for the microcontroller. Write-protection is provided by requiring an external high voltage source for this particular block. If the external high voltage is unavailable, it will not be possible to erase the protected block.

In the actual application, the high voltage would probably be absent. The remainder of the Flash module can use an internal charge pump to generate the required high voltage necessary for programming and erasing the Flash.

One use for block protection is to block protect an area of Flash memory for a boot loader program. This boot loader program can be used to erase the rest of the Flash memory and reprogram it. Because the boot loader is protected, it remains intact even if MCU power is lost in the middle of an erase and reprograms operation.

Vector Redirection

Whenever any block protection is enabled, the reset and interrupt vectors in the microcontroller will be protected. Vector redirection allows users to modify interrupt vector information without unprotecting bootloader and reset vector space. It also allows the user to reprogram the unprotected portion of the Flash with new program code, including new interrupt vector values, and leaving the protected area unchanged.

Block protects cannot be changed from user application programs. If the vector space is block protected, the backdoor security key mechanism (see the following section for more information) cannot permanently change the block protect, security settings or the backdoor key.

Protection Violation Flag

A protection violation flag bit is automatically set to 1 so as to register a command that attempts to erase or program a location in a protected block. (The erroneous command is ignored.)

Backdoor Security Key

A user can choose to allow or disallow a security unlocking mechanism through an 8-byte backdoor security key. The backdoor comparison key is located in Flash memory locations in the nonvolatile register space so users can program these locations exactly as they would program any other Flash memory location. The nonvolatile registers are in the same block of Flash as the reset and interrupt vectors, so block protecting that space also block protects the backdoor comparison key.

If security needs to be unlocked, the user inputs the key through an external interface, such as a keypad. The input key is compared to the key programmed or stored in the Flash memory. If the input key is same as the stored key, then security is disabled.

The backdoor key can be disabled through setting an appropriate bit to 0. When this is finished, there is no way to disengage security without completely erasing all Flash locations. There are, however, other ways of temporarily disengaging security. Security can always be disengaged through the background debug.

Conclusion

Freescale’s wide range of Flash-based, 8-bit microcontrollers (MC68HC908 and MC9S08 Families) have security features that lend protection for your code to help make them secure from tampering. Choose from a range of high performance devices with Flash memory, from 1 KB to 128 KB and pin counts from 8 to 80. Visit www.freescale.com/8bit for more information.

Meera Balakrishnan is a product marketing manager at Freescale.

Comments about this article?

e-Mail MCUideas@freescale.com.