AN2880
Rev. 1.0, 08/2005

Using the Backdoor Access Capability to Unsecure HCS12 MCUs

By Rogelio Reyna García
RTAC Americas
Mexico 2005

Overview

This document is a quick reference for an embedded engineer to implement the backdoor access capability for any HCS12 MCU. Basic knowledge about the functional description will give the user a better understanding on how the backdoor access works. This application note provides an example that demonstrates the use of backdoor access capability to unsecure an MCU in the HCS12 Family of microcontrollers. The example mentioned is intended to be modified to suit the specific needs for any application.

The example CodeWarrior project files are available as AN2880SW.zip from http://freescale.com.

Backdoor Access

The HCS12 Family has a security feature that enables the user to protect intellectual property by limiting the access to NVM (nonvolatile memory) for reading purposes. It is important to note that “secure” is different from “protect”, because “protect” is a feature of this family to manage write access to NVM. The idea behind “protect” is to ensure that the application code is not overwritten by mistake. The idea behind “secure” is to ensure that only the owner of the backdoor access keys will have access to the MCU code.

When enabled, secured operation has the following effects on the microcontroller[1]:

Normal Single Chip Mode

- Background debug module (BDM) operation is blocked.

Special Single Chip Mode

- BDM firmware commands are disabled.
- BDM hardware commands are restricted to the register space.
- Flash and EEPROM commands limited to MASS ERASE only.

Expanded Modes

- Internal Flash and EEPROM are disabled.
- BDM operations will be blocked.

There are two ways to unsecure the MCU: by mass-erasing the NVM or by using backdoor access keys[2]. It is important to note that when you unsecure a secured MCU by using backdoor access keys, it will be unsecured temporarily, and after next reset it will be secured again unless you program the appropriate Flash locations to unsecure the MCU permanently.

Registers and Flash locations

Before trying to unsecure an MCU, it would be good to ensure that it is really secured and that backdoor keys access has been enabled. This can be done by reading the FSEC register of the Flash module.

Base + 0x0101

	7	6	5	4	3	2	1	0
R	KEYEN1	KEYEN0	NV5	NV4	NV3	NV2	SEC1	SEC0
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 1. FSEC Register

If the bits SEC1:SEC0 are “10”, the MCU is unsecured.

If the bits KEYEN1: KEYEN0 are “10”, backdoor access is enabled[3].

1. For details on the operation of the MCU in the secured state, please refer to the device user guide or data sheet.
2. If you secure the MCU by mistake and you cannot unsecure it, you can use the “Unsecure…” command of CodeWarrior debugger. It will mass-erase the Flash for you and leave the MCU in the unsecured state.
3. Some versions of Flash have only one KEYEN bit. If this bit is set, then backdoor access is enabled.

You cannot write directly to this register because its value is copied in the reset sequence from the “Flash Options/Security” byte (located at address $FF0F). This location is writable and erasable as any other Flash location.

The location of the backdoor keys are:

- Key 1 — $FF00–$FF01
- Key 2 — $FF02–$FF03
- Key 3 — $FF04–$FF05
- Key 4 — $FF06–$FF07

However, you cannot use these locations as any other Flash locations. The next section describes how to interact with these locations.

Code and Explanation

The example code for this application note is available as a CodeWarrior project (AN2880SW.zip) from http://freescale.com.

At reset, the example software determines whether the MCU is secured or unsecured.

- If unsecured:
  - Display message to user by SCI communications at 9600 bps.
  - Save the backdoor keys. (In this example, they are in an array defined in the source code; in an end application, they must be obtained from an outside source.)
  - Backup of the Flash sector that contains the “Flash Options/Security” byte.
  - Program the “Flash Options/Security” to secured state.
  - Infinite loop.
- If secured:
  - Display message to user.
  - Unsecure the MCU by means of backdoor access.

To unsecure and secure the MCU, the application must be able to write to Flash. For more information about programming and erasing Flash, please refer to application notes AN2720, AN2204, and AN2400.

To unsecure the MCU, the backdoor key access sequence must be followed:

1. Set the KEYACC bit in the Flash Configuration register (FCNFG).
2. Write the first 16-bit word of the backdoor key to $FF00.
3. Write the second 16-bit word of the backdoor key to $FF02.
4. Write the third 16-bit word of the backdoor key to $FF04.
5. Write the fourth 16-bit word of the backdoor key to $FF06.
6. Clear the KEYACC bit in the Flash Configuration register (FCNFG).
7. If all four 16-bit words match the backdoor keys, the MCU is unsecured and bits SEC[1:0] in the FSEC register are forced to the unsecure state of “10”.

NOTE

Flash cannot be read while KEYACC is set. Therefore, the code for the backdoor key access sequence must execute from RAM.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the Flash security byte can be programmed to the unsecure state, if desired, to leave the MCU unsecured after next reset.

No word of the backdoor key is allowed to have the value $0000 or $FFFF.

It is important to emphasize the fact that the code that would unsecure the MCU must be executed from RAM locations. AN2720 shows a way to copy routines to the stack and execute them from there. This way, no resources are reserved for such a task.

The software of this application note (AN2880SW.zip) uses that concept to unsecure the MCU with minimum RAM overhead. Please refer to the source code for more details.

Considerations

The code for this project was developed in CodeWarrior 3.1 for S12. It was developed for and tested in an HCS12DP256B MCU in an Adapt9S12DP256 board (from Technological Arts).

To monitor the state of the MCU, a serial connection was used (9600 bps, 8 data bits, no parity, no flow control, 1 stop bit).

References

The following documents are also available from the Freescale Semiconductor website:

- AN2720: A Utility for Programming Single FLASH Array HCS12 MCUs, with Minimum RAM Overhead
- S12FTS32KV1: FTS32K Block User Guide
- S12FTS64KV1: FTS64K Block User Guide
- S12FTS128KV2: FTS128K Block User Guide
- HCS12FTS256KUG: FTS256K Block User Guide
- AN2400: HCS12 NVM Guidelines
- MC9S12DP256B: MC9S12DP256B Device User Guide

This page is intentionally blank.

This page is intentionally blank.

This page is intentionally blank.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

AN2880
Rev. 1.0, 08/2005