Freescale Semiconductor
Engineering Bulletin

Document Number: EB727
Rev. 0, 01/2010

Enabling and Disabling ECC on MC9S08DE60/MC9S08DE32 Microcontrollers

by: Philip Drake, 8-Bit Systems and Applications Engineering
Tom Lee, Product and Test Engineering
Austin, Texas

1 Introduction

Freescale Semiconductor's MC9S08DE60 and MC9S08DE32 microcontrollers have a unique feature that enables higher reliability and quality in an application. That feature is Error Correction Code (ECC) for the on-chip flash. There may be an issue with enabling the ECC that will most certainly cause problems. This procedure offers a tried and proven way to enable ECC in your application.

2 Problem statement

Let us explore the nature of this problem. This is in reference to information in the Freescale document MC9S08DE60, *MC9S08DE60/MC9S08DE32 Data Sheet,* section 4.5.10.1, "Enable ECC," that explains to customers how to correctly enable the flash ECC.

When the ECC is enabled, the NVOPT flash register that holds the security bits will become corrupted. This will

Contents

freescale™ semiconductor

cause the memory to be inadvertently secured. This device has a control option that allows the ECC to be enabled or disabled.

In the MC9S08DE60, when the ECC is disabled, the full 60K of flash (0x1080–0x13FF, 0x1900–0xFFFF) is available in the memory map. In the MC9S08DE32, when the ECC is disabled, the full 32K of flash (0x7C00–0xFFFF) is available. When the ECC is enabled, certain addresses of the flash block are allocated as ECC parity bits and thus removed from the memory map. There are four ECC parity bits for every eight ECC data bits. Therefore, when ECC is enabled, a third of the memory is used for parity. In the MC9S08DE60, when the ECC is enabled, only 43K of the flash (0x5400–0xFFFF) is available in the memory map. In the MC9S08DE32, when the ECC is enabled, only 22K of the flash (0xA800–0xFFFF) is available.

When the MC9S08DE60 flash is mass erased, the ECC is disabled by default. The NVOPT is initially programmed to 0xFE just like all other MC9S08 devices. When ECC is enabled, the NVOPT becomes corrupted so that it no longer reads 0xFE, and security is enabled. This corruption occurs because the ECC parity for NVOPT in the flash is 0xF, but it should be 0x9. Because of this ECC parity and data mismatch, the data is corrupted.

The solution is to program 0xFA into NVOPT instead of 0xFE. This prevents corruption of the NVOPT register. This works because a value of 0xFA along with an ECC parity of 0xF produces a NVOPT value of 0xFA from the ECC decoder. The end result is that 0xFA will keep the memory unsecured when ECC is enabled or disabled.

As ECC is enabled, there are other things that can cause problems. Other flash registers can be corrupted if they are not handled properly when ECC is enabled. If there are trim, write-protect, or backdoor key values that are present, those values will be corrupted as ECC is enabled, since the ECC parity bits in the flash are most likely incorrect for those flash registers.

To prevent these problems, two procedures shown below will bypass these issues.

3 Procedure to enable ECC

The steps shown here outline the recommended procedure to properly enable the ECC. Using a different procedure may cause inadvertent enabling of memory security or corruption of the flash data.

1. Take the data values that are stored in the following flash register locations and temporarily store them to a memory other than flash:

   0xFFAE (FTRIM)

   0xFFAF (TRIM)

   0xFFB0–0xFFB7 (NVBACKKEY)
2. Mass erase the flash.
3. Write 0x66 to NVECC.
4. Write 0xFA to NVOPT.

   Writing any value other than 0xFA may cause security to be enabled after a reset.
5. Assert reset.
6. Restore the following flash register locations using the temporarily stored values in step one:

0xFFAE (FTRIM)

0xFFAF (TRIM)

0xFFB0–0xFFB7 (NVBACKKEY)

7. Program the S-record into flash. The most direct way to accomplish this is through the expert programming interface available in CodeWarrior. Select the Multilink/Cyclone tab → Start Expert Mode Programmer. Select the correct programming algorithm and S19 record, then program the application code. Make sure the code does not include the location already programmed, the FTRIM, TRIM, and NVBACKKEY.
8. Write the block protect value to NVPROT following the ECC-on settings, if that is preferred.
9. NVOPT bits (other than bits 0 and 2) can be programmed to a different value, if that is preferred.

After programming any flash location with ECC enabled, that location should not be reprogrammed a second time with a different value. Reprogramming any flash location will cause the data to be corrupted. Progressively programming a bit one at a time into the same flash location will also cause data to be corrupted at that location. When ECC is enabled, each flash location should be programmed once and only once.

Transitioning from an ECC-enabled state to an ECC-disabled state (or vice versa) will cause the flash data to be corrupted. The flash should always be mass erased before making a change to the state of the ECC.

4 Procedure to disable ECC

The steps shown here outline the recommended procedure to properly disable the ECC. Using a different procedure may cause inadvertent enabling of memory security or corruption of the flash data.

1. Take the data values that are stored in the following flash register locations and temporarily store them to a memory other than flash:

   0xFFAE (FTRIM)

   0xFFAF (TRIM)

   0xFFB0–0xFFB7 (NVBACKKEY)
2. Mass erase the flash.
3. Write 0xFE to NVOPT to keep the memory unsecured after a reset.
4. Assert reset.
5. Restore the following flash register locations using the temporarily stored values in step one:

   0xFFAE (FTRIM)

   0xFFAF (TRIM)

   0xFFB0–0xFFB7 (NVBACKKEY)
6. Program the S-record into flash.
7. Write the block protect value to NVPROT following the ECC-off settings, if that is preferred.
8. NVOPT bits (other than bit 0) can be programmed to a different value, if that is preferred.

5 Command file scripts creation and usage

CodeWarrior has the ability to utilize command scripts. Anything you can do in the command window you can also do in a command script. Use the “call” command to select and run your script. To get a list of commands, type help. To get the syntax for a particular command, type that command and the tool will return the correct syntax.

Appendix A, “Enable ECC command script,” provides a command script that can be used to safely enable the ECC. Appendix B, “Disable ECC command script,” provides a command script to correctly disable the ECC.

6 Conclusion

By utilizing these procedures, you can enable or disable ECC and program your device without encountering data corruption issues.

7 References

- Freescale document MC9S08DE60, *MC9S08DE60/MC9S08DE32 Data Sheet*
- CodeWarrior for Microcontrollers version 6.3 IDE

Appendix A Enable ECC command script

```
// MC9S08DE60 & MC9S08DE32 ECC Enabling Command File

// These commands will:
//     1. mass erase the flash
//     2. program NVOPT to 0xFA
//     3. enable the flash ECC
//     4. restore the flash registers

// Evaluate the clock divider to set in FCDIV register:

// An average programming clock of 175 kHz is chosen.

// If the bus frequency is less than 10 MHz, the value to store
//     in FCDIV is equal to " bus frequency (kHz) / 175 ".

// If the bus frequency is higher than 10 MHz, the value to store
//     in FCDIV is equal to " bus frequency (kHz) / 1400  + 0x40
//     (to set PRDIV8 flag)".

// Data sheet values:
//
// Bus Frequency     FECDIV Value (decimal)
// -------------     ----------------------
//      20 MHz             76
//      10 MHz             49
//       8 MHz             39
//       4 MHz             19
//       2 MHz              9
```

```
//       1 MHz             4
//     200 kHz             0
//     150 kHz             0

// assert reset
reset
wait 10

define FCDIV 39 // 8 MHz bus frequency

// flash setup
wb 0x1802 0     // disable COP clearing SOPT1 register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0x1824 0xff  // set FPROT register to remove all flash protections
wb 0x1820 FCDIV // set clock divider in FCDIV register
                // see above to find how to evaluate this constant value

// temporarily store the data values in the following Flash register
//     locations to RAM:
//     0xFFAE        (FTRIM)
//     0xFFAF        (TRIM)
//     0xFFB0-0xFFB7 (NVBACKKEY)
copymem 0xffae..0xffb7 0x0080

//mass erase flash
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0xc000 0     // (dummy) write to flash array to buffer address and data
wb 0x1826 0x41  // write MASS ERASE command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 20

//blank check flash & unlock security
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0xc000 0     // (dummy) write to flash array to buffer address and data
wb 0x1826 0x5   // write BLANK CHECK command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 20

//write 0x66 to NVECC to prepare for enabling the ECC
wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0xffb8 0x66  // write NVECC register in flash array for ECC enable
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

//Write 0xFA to NVOPT.  Writing any value other than $FA may cause
//    security to be enabled after a reset.

//reprogram NVOPT to unsecure state with 0xFA value
wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0xffbf 0xfa  // write NVOPT register in flash array to UNSECURED state
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10
```

Enable ECC command script

```
// assert reset
reset
wait 10

// flash ECC is now enabled

// flash setup
wb 0x1802 0     // disable COP clearing SOPT1 register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0x1824 0xff  // set FPROT register to remove all flash protections
wb 0x1820 FCDIV // set clock divider in FCDIV register

// restore the following Flash register locations using the temporarily
//     stored values in RAM:
//     0xFFAE        (FTRIM)
//     0xFFAF        (TRIM)
//     0xFFB0-0xFFB7 (NVBACKKEY)

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0080..0x0080 0xffae // write FTRIM register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0081..0x0081 0xffaf // write TRIM register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0082..0x0082 0xffb0 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0083..0x0083 0xffb1 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0084..0x0084 0xffb2 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0085..0x0085 0xffb3 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
```

```
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0086..0x0086 0xffb4 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0087..0x0087 0xffb5 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0088..0x0088 0xffb6 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0089..0x0089 0xffb7 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10
```

Appendix B Disable ECC command script

```
// MC9S08DE60 & MC9S08DE32 ECC Disabling Command File

// These commands will:
//     1. mass erase the flash
//     2. program NVOPT to 0xFE
//     3. disable the flash ECC
//     4. restore the flash registers

// Evaluate the clock divider to set in FCDIV register:

// An average programming clock of 175 kHz is chosen.

// If the bus frequency is less than 10 MHz, the value to store
//     in FCDIV is equal to " bus frequency (kHz) / 175 ".

// If the bus frequency is higher than 10 MHz, the value to store
//     in FCDIV is equal to " bus frequency (kHz) / 1400  + 0x40
//     (to set PRDIV8 flag)".

// Data sheet values:
//
// Bus Frequency     FECDIV Value (decimal)
// -------------     ----------------------
```

Disable ECC command script

```
//      20 MHz            76
//      10 MHz            49
//       8 MHz            39
//       4 MHz            19
//       2 MHz             9
//       1 MHz             4
//     200 kHz             0
//     150 kHz             0

// assert reset
reset
wait 10

define FCDIV 39 // 8 MHz bus frequency

// flash setup
wb 0x1802 0     // disable COP clearing SOPT1 register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0x1824 0xff  // set FPROT register to remove all flash protections
wb 0x1820 FCDIV // set clock divider in FCDIV register
                // see above to find how to evaluate this constant value

// temporarily store the data values in the following Flash register
//     locations to RAM:
//     0xFFAE        (FTRIM)
//     0xFFAF        (TRIM)
//     0xFFB0-0xFFB7 (NVBACKKEY)
copymem 0xffae..0xffb7 0x0080

//mass erase flash
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0xc000 0     // (dummy) write to flash array to buffer address and data
wb 0x1826 0x41  // write MASS ERASE command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 20

//blank check flash & unlock security
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0xc000 0     // (dummy) write to flash array to buffer address and data
wb 0x1826 0x5   // write BLANK CHECK command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 20

//reprogram NVOPT to unsecure state with 0xFE value
wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0xffbf 0xfe  // write NVOPT register in flash array to UNSECURED state
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

// assert reset
reset
wait 10

// flash ECC is now disabled
```

```
// flash setup
wb 0x1802 0     // disable COP clearing SOPT1 register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
wb 0x1824 0xff  // set FPROT register to remove all flash protections
wb 0x1820 FCDIV // set clock divider in FCDIV register

// restore the following Flash register locations using the temporarily
//     stored values in RAM:
//     0xFFAE        (FTRIM)
//     0xFFAF        (TRIM)
//     0xFFB0-0xFFB7 (NVBACKKEY)

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0080..0x0080 0xffae // write FTRIM register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0081..0x0081 0xffaf // write TRIM register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0082..0x0082 0xffb0 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0083..0x0083 0xffb1 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0084..0x0084 0xffb2 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0085..0x0085 0xffb3 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0086..0x0086 0xffb4 // write NVBACKKEY register in flash array
```

Disable ECC command script

```
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0087..0x0087 0xffb5 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0088..0x0088 0xffb6 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10

wb 0x1820 FCDIV // set clock divider in FCDIV register
wb 0x1825 0x30  // clear FPVIOL and FACCERR in FSTAT register
copymem 0x0089..0x0089 0xffb7 // write NVBACKKEY register in flash array
wb 0x1826 0x20  // write BYTE PROGRAM command in FCMD register
wb 0x1825 0x80  // set FCBEF in FSTAT register to execute the command
wait 10
```

THIS PAGE IS INTENTIONALLY BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

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)
www.freescale.com/support

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 China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: EB727
Rev. 0
01/2010

freescale™
semiconductor