

MC9S08DZ60 Flash Usage Considerations

by: Andy McKeachan
Applications Engineer
East Kilbride

1 Introduction

Freescale's MC9S08DZ60 8-bit microcontroller is the centerpiece of the low cost high performance family.

The purpose of this engineering bulletin is to provide a short overview of the embedded Flash memory that is implemented on the MC9S08DZ60, explain some of the advantages associated with it, and how to use it. Example software for program and erase operations written in the C programming language is provided. This document primarily relates to the Flash memory on the MC9S08DZ60. It should be noted that the procedures required for programming and erasing the EEPROM memory are identical to those that are required for programming and erasing the Flash memory.

Contents

1	Introduction	1
2	MC9S08DZ60 Flash Memory Overview	2
3	How to Extend the Lifetime of Flash Memory	3
4	Software Examples	6
5	Summary	8
6	References	8

2 MC9S08DZ60 Flash Memory Overview

The MC9S08DZ60 microcontroller has a single non-continuous Flash memory array of 60032 bytes. There are 896 bytes of Flash memory located from 0x1080 to 0x13FF and 59136 bytes located from 0x1900 to 0xFFFF, including the interrupt vector table located from 0xFFC0 to 0xFFFF.

0x0000	Direct page registers
0X007F	128 bytes
0X0080	RAM 4096 bytes
0X107F	
0X1080	FLASH 896 bytes
0X13FF	
0X1400	EEPROM ¹ 2 x 1024 bytes
0X17FF	
0X1800	High page registers 256 bytes
0X18FF	
0X1900	
	FLASH 59136 bytes
0XFFFF	

MC9S08DZ60

Figure 1. MC9S08DZ60 Memory Map

The MC9S08DZ60 microcontroller incorporates advanced third generation 0.25 μ Flash technology making the Flash memory more affordable because its size approaches the size of ROM. The Flash memory is robust with a typical data retention of over 100 years. It can also be reliably used in low power applications. Therefore, reads and writes of the Flash memory are possible down to 2.7 V. This is a useful feature helping to reduce current consumption within the application.

The Flash memory can be erased and reprogrammed many times over (typically over 100 K write/erase cycles at 25° C) and is therefore ideally suited for the software development phase of a project. The Flash memory is also suitable for the production phase. Software changes can be implemented immediately, eliminating the cycle time and costs involved in generating a new ROM mask.

The MC9S08DZ60 microcontroller features a simplified command interface (CI) for programming, erasing, and blank checking the Flash array. The CI makes modifying the Flash memory easier from within the MCU application. This offers the user the significant advantage of being able to quickly reprogram the Flash memory in the field, therefore eliminating the need to replace the microcontroller in the application, if a software fix or upgrade is required.

Taking all of the above factors into account; it can be seen that over the product lifespan, Flash memory offers greater flexibility and significant potential cost savings in comparison to ROM.

The MC9S08DZ60 microcontroller features a burst programming mode that can be used to program sequential bytes of data in less time than normally required, if they were programmed individually. Burst programming is invoked by pipelining program commands for bytes in the same burst block. Burst programming reduces the programming time by keeping the high voltage generator switched on between program commands in the burst block. Burst programming is approximately twice as fast as single byte programming. When the burst programming feature is used, it is possible to program the full 60032 bytes of Flash memory on the MC9S08DZ60 in approximately three seconds. This faster programming time results in reduced production programming costs and reduced power consumption during programming.

The MC9S08DZ60 microcontroller offers the user the ability to perform a mass erase of the full 60032 bytes of Flash memory in approximately 100 ms, or the erasure of an individual 768 byte sector in approximately 20 ms. The MC9S08DZ60 Flash module also features a sector erase abort operation that allows the user to terminate an active sector erase operation. This makes other sectors available for read and program operations without having to wait for the sector erase operation to complete.

The MC9S08DZ60 microcontroller offers a useful block protection feature that prevents the contents of Flash addresses from being unintentionally programmed or erased in the application. The user has the ability to define the size of the memory being protected, ranging from 1.5 Kbytes located between 0xFA00–0xFFFF, up to the full range of Flash addresses. The user can implement a desired protection scheme by programming the relevant value into the Flash protect select (FPS) bits in the non-volatile Flash and EEPROM protection (NVPROT) register located at 0xFFBD. Protection is enabled after the next reset occurs. Note that the reset and interrupt vectors are protected if any area of the Flash memory is block protected. Therefore a vector redirection feature is available on the MC9S08DZ60 allowing the user to modify interrupt vector information without having to unprotect the bootloader and reset vector space. The MC9S08DZ60 microcontroller also has a security feature that prevents unauthorized read access to the contents of Flash, EEPROM, and RAM memory. Full details on the operation of block protection, vector redirection, and security implementation can be found in the device data sheet.

In summary, taking all of the above factors into account the MC9S08DZ60 offers a flexible, efficient, reliable, secure, and cost effective solution to the user who wishes to utilise a microcontroller with embedded Flash memory in their application.

3 How to Extend the Lifetime of Flash Memory

The Flash memory on the MC9S08DZ60 microcontroller is robust, however it does have a finite data retention lifetime. There are a number of factors that the user must take into consideration to maximise the data retention lifetime.

An important factor relates to the frequency of the internal FLASH and EEPROM clock derived from the microcontroller bus clock. The FLASH and EEPROM clock divider (FCDIV) register must be written with an appropriate value before programming or erasing operations commence. The value written to the register must be selected to ensure that the time-base is between 150 kHz and 200 kHz, which corresponds to the allowable program/erase timing pulse range of 5 – 6.7 μS.

The layout of the FCDIV register is shown in [Figure 2](#).

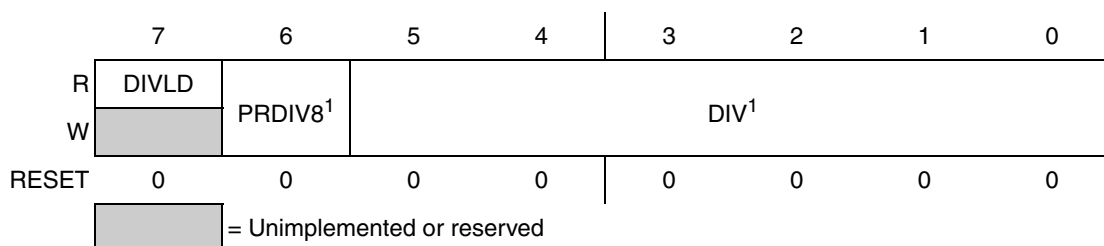


Figure 2. FLASH and EEPROM Clock Divider Register (FCDIV)

¹ PRDIV8 and DIV[5:0] are write once out of reset

It is the user’s responsibility to write the correct values to the PRDIV8 and DIV bits. These values can be calculated from the following equations:

if PRDIV8 = 0:

$$f_{FCLK} = (f_{BUS} / (DIV + 1)) \tag{Eqn. 1}$$

if PRDIV8 = 1:

$$f_{FCLK} = f_{BUS} / (8(DIV + 1)) \tag{Eqn. 2}$$

Example 1: Calculating the value to write to the FCDIV register when running with a bus frequency of 10 MHz. For this case PRDIV8 = 0 and the assumption is that target $f_{FCLK} = 200$ kHz.

$$f_{FCLK} = (f_{BUS} / (DIV + 1))$$

$$(DIV + 1) = f_{BUS} / f_{FCLK} = (10M) / (200K) = 50$$

$$DIV = 49$$

Therefore, for this configuration the value 0b00110001 or 0x31 must be written to the FCDIV register.

Example 2: Calculating the value to write to the FCDIV register when running with a bus frequency of 20 MHz. For this case PRDIV8 = 1, and the initial assumption is that target $f_{FCLK} = 200$ kHz.

$$f_{FCLK} = f_{BUS}/(8*(DIV + 1))$$

$$(8 \times (DIV + 1)) = f_{BUS}/f_{FCLK} = (20M)/(200K) = 100$$

$$(DIV + 1) = 12.5$$

$$DIV = 11.5$$

This value for DIV must be rounded up to 12 to ensure that the frequency of f_{FCLK} does not exceed its maximum allowable value of 200 KHz. Feeding this value of DIV back into Equation 2 yields the value of f_{FCLK} as below.

$$f_{FCLK} = f_{BUS}/(8*(DIV + 1))$$

$$f_{FCLK} = (20M)/(8*13)$$

$$f_{FCLK} = 192.3KHz$$

Therefore, for this configuration the value 0b01001100 or 0x4C must be written to the FCDIV register.

Table 1 shows the values for PRDIV8 and DIV that must be used for commonly selected bus frequencies.

Table 1. FLASH and EEPROM Clock Divider Settings

f_{Bus}	PRDIV8 (Binary)	DIV (Decimal)	f_{FCLK}	Program/Erase Timing Pulse (5 μ s Min, 6.7 μ s Max)
20 MHz	1	12	192.3 KHz	5.2 μ s
10 MHz	0	49	200 KHz	5 μ s
8 MHz	0	39	200 KHz	5 μ s
4 MHz	0	19	200 KHz	5 μ s
2 MHz	0	9	200 KHz	5 μ s
1 MHz	0	4	200 KHz	5 μ s
200 KHz	0	0	200 KHz	5 μ s
150 KHz	0	0	150 KHz	6.7 μ s

Applying a program/erase timing pulse that is too short can result in cells in the Flash array that do not accumulate the necessary charge required for the reliable read back of the cell status. There is also a minimal risk of data retention issues if a program/erase timing pulse of insufficient duration is used. This is due to an insufficiently charged cell losing further charge over a period of time. On the other hand, applying a program/erase timing pulse that is too long can result in damage to the Flash memory due to overstress. All of these potential hazards can be avoided if the correct program/erase timing pulse is used.

The lifetime of the Flash memory can be extended by keeping the program/erase cycles to a minimum. For this reason the sector erase abort command must be used sparingly because a sector erase operation that is aborted counts as a complete program/erase cycle. Although not practical for all applications, the lifetime of the Flash memory can also be extended by limiting its use to moderate temperatures (0 –70°C).

Proper handling and storage precautions must also be observed to avoid the possibility of any ESD damage being inflicted on the microcontroller.

4 Software Examples

This section describes the example software operation that accompanies this engineering bulletin. The software has been developed using the CodeWarrior Development Studio for Microcontrollers V6.1 available from Freescale at <http://www.freescale.com>.

The sample C software consists of the following files:

- main.c – contains the software for initialising the device (setting up the multi-purpose clock generator (MCG) module and so on)
- flash_drv.c – contains the mass erase, sector erase, sector erase abort, blank check, byte programming, and burst programming routines
- sci_drv.c – contains the SCI driver software

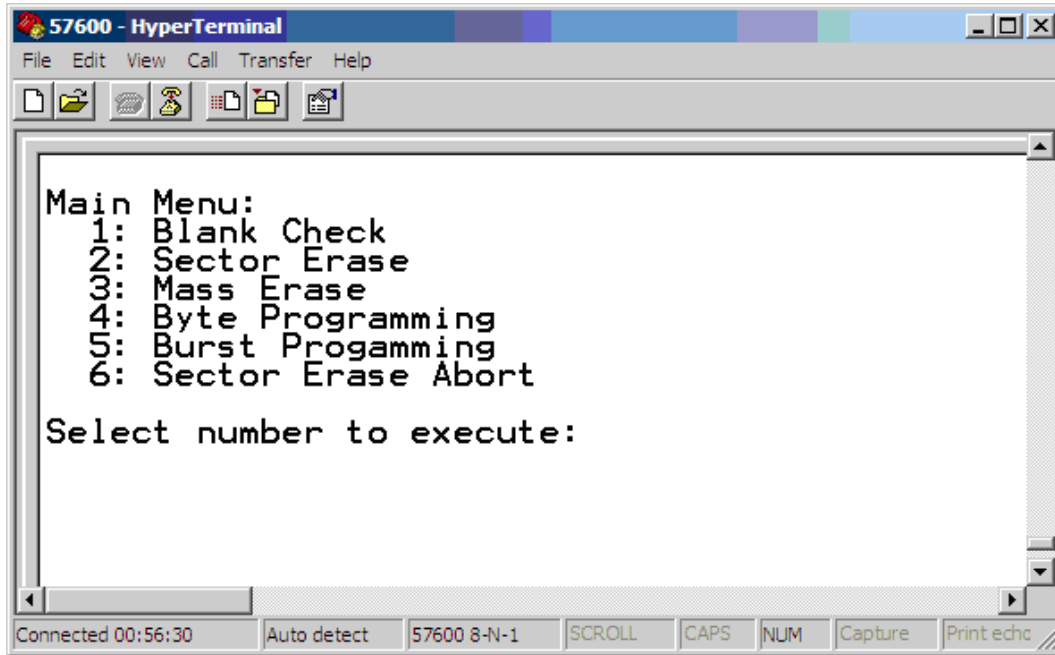
This software example configures the MCG to produce a 20 MHz bus clock from an 8 MHz external crystal. A value of 0x4C (refer to earlier calculation of this value) is written to the FCDIV register to ensure the program/erase timing pulse is in the correct range.

For actual operation, the compiled software is loaded into the microcontroller RAM via the BDM.

NOTE

The Flash architecture implemented on the MC9S08DZ60 does not allow instructions to be executed out of the FLASH memory while any program or erase operation is in progress.

The software example configures the SCI1 on the MC9S08DZ60 to communicate with a terminal program operating with a baud rate of 57600. When the software is executed, a menu appears on the terminal allowing the user to select the operation they would like to execute.



```
57600 - HyperTerminal
File Edit View Call Transfer Help
Main Menu:
 1: Blank Check
 2: Sector Erase
 3: Mass Erase
 4: Byte Programming
 5: Burst Programming
 6: Sector Erase Abort
Select number to execute:
Connected 00:56:30 Auto detect 57600 8-N-1 SCROLL CAPS NUM Capture Print echo
```

Figure 3. Terminal Program Main Menu

Figure 3 shows there are 6 possible options available.

1. Blank Check — When selected, this routine checks the status of the verified as all blank flag (FBLANK) in the Flash and EEPROM status (FSTAT) register. The message; the Flash memory is blank, is displayed on the terminal if the flag is set, otherwise the message; the Flash memory is not blank, is displayed.
2. Sector Erase — When selected, this routine erases the Flash sector located from address 0xD000 to 0xD2FF. The software performs a check to ensure that all locations within the sector read back as being blank. The message; procedure has been completed successfully, is displayed on the terminal if all of the locations within the sector are blank, otherwise the message; sector erase procedure has failed, is displayed.
3. Mass Erase — When selected, this routine erases the entire Flash memory. This includes the non-volatile options (NVOPT) register at location 0xFFBF that contains the bits for selecting the security state of the microcontroller. The mass erase procedure sets the SEC[1:0] bits to 1:1 that has the effect of securing the microcontroller after the next reset occurs. Information on how to unsecure the microcontroller can be found in the Security section of the Memory chapter in the data sheet.
4. Byte Programming — When selected, this routine programs the Flash memory starting at address 0xD000 with the contents of a 256 byte array (data[256]) using the byte programming feature. The software performs a check to ensure that all of the data in the programmed locations match the expected values. The message; procedure has been completed successfully, is displayed on the terminal if the data in all of the programmed locations match the expected values, otherwise the message; an error has occurred during programming procedure is displayed.

Summary

5. Burst Programming — When selected, this routine programs the Flash memory starting at address 0xD000 with the contents of a 256 byte array (datum[256]) using the burst programming feature. The software performs a check to ensure that all of the data in the programmed locations match the expected values. The message; procedure has been completed successfully, is displayed on the terminal if the data in all of the programmed locations match the expected values, otherwise the message; an error has occurred during programming procedure, is displayed.
6. Sector Erase Abort — When selected, this routine starts to erase the Flash sector located from the address 0xD000 to 0xD2FF. At a user defined stage in this process, the sector erase abort command is executed. After the operation has been completed, a check is made on the status of the flash access error (FACCERR) bit. If this bit is set the message; procedure has been completed successfully, is displayed on the terminal, otherwise the message; sector erase abort procedure has failed, is displayed.

NOTE

Before performing a programming operation on a particular byte in the Flash, the sector where that byte resides must be erased by a mass or sector erase operation. Reprogramming bits in an already programmed byte without first performing an erase operation may disturb data stored in the Flash memory.

5 Summary

The main objective of this engineering bulletin is to provide the reader with sample software examples written in the C programming language showing how to program and erase the Flash memory on the MC9S08DZ60. It provides examples showing how to perform a sector erase or a mass erase as well as how to perform byte and burst programming operations. It also provides a short overview of the Flash memory that is implemented on the MC9S08DZ60, outlines some of its advantages, and provides information on steps that can be taken to maximise its lifetime.

6 References

HCS08RMv1/D, HCS08 Family Reference Manual.

MC9S08DZ60 Data Sheet Revision 3.

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
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.



Document Number: EB695

Rev. 0

07/2008

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2008. All rights reserved.