NXP

Smoke Detector Application Hints Based on MC9S08QG4

by: Jerry Shi
      Freescale Semiconductor, Beijing

This application note introduces the smoke detector application based on MC9S08QG4. This application note:

- Provides the block diagram of the smoke detector system and details the functions of each block

- Lists the resource and features required for the smoke detector system

- Introduces the working modes of the MC9S08QG4

- Discusses the internal clock source (ICS) configuration and operation mode for low-power design

- Illustrates how to use the flash memory of MC9S08QG4 to emulate EEPROM

- Explains how to select an ADC clock for the fire detector system

Contents

freescale™
semiconductor

1 Smoke Detector System Architecture

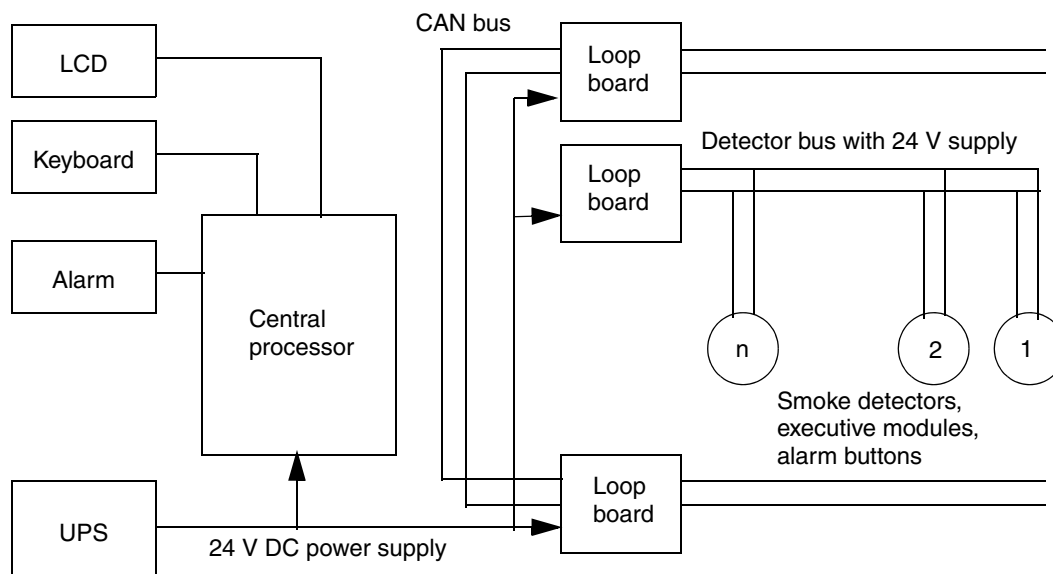Figure 1 shows a block diagram of a fire control system.

Figure 1. Fire Control System Block Diagram

This fire control system includes these blocks:

- Central processor: The central processor controls the system, periodically inquires the status of each smoke detector and alarm button, and sends commands to each executive module through loop boards. Also, it offers human interface with a TFT LCD and keyboard and alarms and displays locations of detected fire.

- Loop board: There are several loop boards in the fire control system. Loop boards are connected to the central processor. RS-485 was the most popular connectivity method between a loop board and central processor, but the CAN bus is used more frequently in modern designs.

  Each loop board has a detector bus interface with modulated data and 24 V DC supply on two wires. Up to 256 smoke detectors, alarm buttons and executive modules can be connected to a single loop, depending on specification of different manufacturers.

- Executive modules and alarm buttons: Executive modules execute actions such as turning on sprinklers in case of fire. Typically, the modules include one or more relays that are controlled by GPIOs of a MCU.

  Alarm buttons are usually installed in the corridor. In case of fire or emergency, you break the glass veil and push the button to raise alarm. Sometimes an even simpler MCU is adopted for lower cost.

- Power supply: Power of the entire fire control system is supplied by a UPS. 24 V DC voltage is delivered to central processor and all loop boards, which, in turn, deliver the 24 V DC power to all smoke detectors, executive modules, and alarm buttons through the detector bus. Communication data is modulated on the bus.

- Detector bus: The detector bus is a type of two wire based special control bus with 24 V DC supply and modulated data. Figure 2 shows a typical waveform on the bus; however, different manufacturers define different communication protocols.
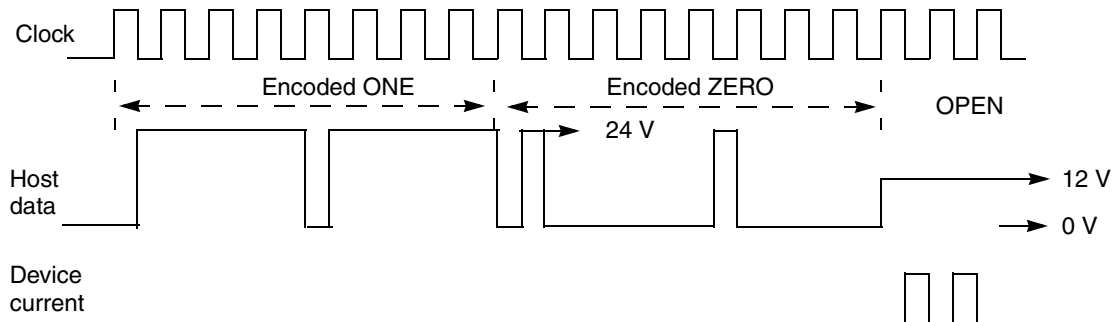

Figure 2. Detector Bus Timing

The timing determines requirement of bus MCU frequency. Two states—1 and 0 (24 V and 0 V)—are used to transfer data from the bus. The third state, OPEN, is used by the host to wait for a device to respond. At this state, the bus is set to a constant medium voltage (for example, 12 V), then the device can pull current from bus which works as a current loop. Change of bus current represents for logic 1 and 0 for communication.

- Smoke detector: Figure 3 shows a block diagram of a common smoke detector.

Each smoke detector in a loop has a unique address code. Inquiry commands are sent through the bus periodically (for example, every 3 seconds). An inquiry command is received by all smoke detectors and, if one of the detectors detects fire, it sends a message back to the controller.
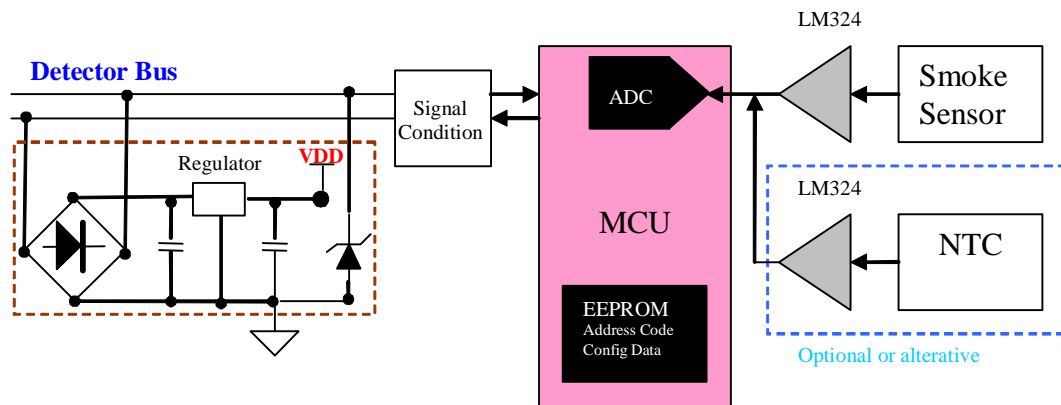

Figure 3. Smoke Detector Block Diagram

Smoke Detector Application Hints Based on MC9S08QG4, Rev. 0

2 Application Requirements and Features

These requirements are necessary for smoke detector applications:

- Very low-power consumption, less than 400 µA for the board
- One channel analog-to-digital converter (ADC)
- Seven GPIOs
- One 8-bit timer for periodical task handle
- Watchdog timer and low-voltage detection for system safe
- EEPROM to store configuration data such as address and configuration, 16 bytes
- 2 KB flash memory, 80 bytes of RAM

Features of MC9S08QG4 are:

- Supply voltage: 1.8 V–3.6 V, –40 °C to 125 °C
- Core: 16 MHz HCS08 core/ 8 MHz bus frequency
- Memory: 4 KB flash memory/ 256 bytes RAM
- Communications: ESCI, SPI, IIC
- 8 MHz internal at 1.8 V – 3.6 V
- Flash memory read/write at 1.8 V
- Internal osc (2% precision over temperature and frequency)
- On-chip ICE (DBG)
- Background debug controller (BDC)
- 2-ch, 16-bit, IC/ OC, or PWM; 8-bit MTIM
- COP, 10-bit ADC, ICS with FLL, LVI, RTI
- Up to 13 GPIOs
- Power saving modes
- On-chip temperature sensor
- Available packages: 16-ld QFN/ TSSOP/PDIP

Thus, the MC9S08QG4 matches the resource requirements for smoke detectors.

3 Working Modes for Low-Power Design

MC9S08QG4 has five modes of operation: run, wait, stop 3, stop 2, and stop 1.

- Wait mode:
  — CPU halts operation to conserve power
  — System clocks running
  — Full voltage regulation is maintained

- Stop modes: CPU and bus clocks stop
  - Stop 1: Full power down of internal circuits for maximum power savings
  - Stop 2: Partial power down of internal circuits; RAM contents retained
  - Stop 3: All internal circuits powered for fast recovery; RAM and register contents retained

After reset, the normal mode of operation is run mode in which the CPU is active and peripherals can be enabled. By executing a WAIT instruction, the MCU enters wait mode. In wait mode, power is reduced because the CPU is not clocked. To reduce power consumption further, stop mode can be used. When a STOP instruction is executed, one of three stop modes will be entered. Stop 1, stop 2, and stop 3 each provide different levels of operation that reduce power consumption.

Table 1 describes stop mode behaviors. Typical supply current of stop mode (3 V supply) is listed:

- Stop1 mode supply current, 475 nA
- Stop2 mode supply current, 600 nA
- Stop3 mode supply current, 750 nA
- RTI adder to stop3, 300 nA
- LVI adder to stop3 (LVDE = LVDSE = 1), 70 μA
- Oscillator adder to stop3, 5 μA

Table 1. Stop Mode Behaviors

Mode	CPU, Digital Peripherals, Flash	RAM	ICG	ATD	KBI	Regulator	I/O Pins	RTI
Stop 1	OFF	OFF	OFF	Disabled[1]	OFF	OFF	RESET	OFF
Stop 2	OFF	Standby	OFF	Disabled	OFF	Standby	States Held	Optionally On
Stop 3	Standby	Standby	Standby[2]	Disabled	Optionally On	Standby	States Held	Optionally On

[1] ATD stop mode or power-down mode depends on the state of ATDPU.
[2] Crystal oscillator can be configured to run stop3. Please see the ICG register.

Table 2 describes how to select each stop mode by configuring bits of several registers, as well as the exit condition of each stop mode.

Table 2. Stop Mode Selection and Exit Condition

STOPE	ENBDM	LVDE	LVDSE	PDC	PPDC	Stop Mode	Exit
0	x	x		x	x	Stop modes disabled	N/A
1	1	x		x	x	Stop 3 with BDM enabled	IRQ, RST, RTI, KBI
1	0	Both bits must be 1		x	x	Stop 3 with voltage regulator active	
1	0	Either bit can be 0		0	x	Stop 3	
1	0	Either bit can be 0		1	1	Stop 2	IRQ/RST/RTI
1	0	Either bit cam be 0		1	0	Stop 1	IRQ/RST

Smoke Detector Application Hints Based on MC9S08QG4, Rev. 0

When using stop 2 to ensure proper operation:

- The IRQ pin must be enabled or pulled up externally.
- The LVD must be disabled in stop (LVDSE = 0).
- If using the RTI, only the internal clock source functions in stop 2.
- The OSCSTEN bit has no effect in stop 2. This clock reference is always powered down.
- Only the RAM remains powered; all other I/O registers are reset upon wakeup.
- The PPDF flag must be cleared before the I/O pins can be modified from their stop 2 entry state.

Recovery from stop mode takes time. At 3 V supply, stop recovery time from STOP1 and STOP2 is approximately 50 μs. Recovery time from STOP3 depends on clock configuration. If an internal reference clock is used, the startup time is approximately 100 μs. If an external 32 kHz crystal is used, the start up time is 1.5 ms if the oscillator is enabled in stop mode (OSCSTEN is set), and 180 ms to 300 ms if oscillator is disabled in stop mode (OSCSTEN is clear).

Although using stop modes can save power, many smoke detectors cannot accept stop modes because the MCU must be ready to respond to the bus at all times. Recovery time from stop mode makes this difficult or even impossible.

4 ICS Configuration and Operation Mode for Low-Power Design

Specified by the typical bus timing, at least 100 kHz bus frequency is necessary for the fire detector system.

The MC9S08QG4 has six types of clock modes according to different configuration of ICS: FEI, FEE, FBI, FBILP, FBE and FBELP. Because FBE mode has the lowest power consumption and highest accuracy clock, it is recommended for smoke detector application. However, because of different bus timing among manufacturers, it may not work sometimes. For example, some bus timing requires 1 MHz bus frequency, MC9S08QG4 needs an external 2 MHz crystal for FBE mode, and many manufacturers do not use a 2 MHz crystal. Some bus specifications requires a very low bus frequency (such as 16 kHz), then FBI mode can be selected. Internal 32 kHz clock of MC9S08QG4 has a typical 0.2% accuracy, and 2% for the whole operation temperature.

A popular and low-cost, low-power configuration is to use a 455 kHz resonator for FBELP mode. The bus frequency is 227.5 kHz (can be divided lower, 113.75 kHz). If supply voltage is 3 V, the supply current of MC9S08QG4 is 240 μA at 227.5 kHz (BDIV = 0), 200 μA at 113.75 kHz (BDIV = 1), and 170 μA at 28.4 kHz (BDIV = 4).

Table 3 shows all the operation modes that are suitable for smoke detector application, considering power consumption and cost issues. These equations are used to calculate bus frequency of different modes:

- FEI: Fbus = Firc x 512 / (2 x bus divider)
- FBI: Fbus = Firc / (2 x bus divider)
- FBE: Fbus = Ferc / (2 x bus divider)

Table 3. Recommended Operation Modes for Smoke Detector

MODE	Fbus	Advantages	Disadvantages	Comments
FBELP	227.5 kHz	Low-cost, high-accuracy, low-supply current.	Cost higher than internal reference clock.	Recommended solution. Oscillator startup time is 5 ms (high range, low power), and supply current is 5 μA.
FBILP	16 kHz	Lowest cost, typical 0.2% and worst 2% accuracy. Quick startup.	Very low bus frequency, can be used for some special designs.	Only useful if bus timing accepts low Fbus. Startup of internal oscillator is 60 μs.
FEI	1 MHz	Lowest cost, typical 0.2% and worst 2% accuracy.	Roughly 2 mA supply current at 3 V 1 MHz.	Some bus specification requires 1 MHz Fbus for proper communication. Although Fbus can be switched to lower one with FLL bypassed and disabled, or even MCU can be stopped when bus is idle, the recovery time may not match the application. FLL startup time is 0.5-1 ms. FLL draws 210-300 μA current itself.

4.1 Working with External 455 kHz Resonator

When the MCU is working with an external 455 kHz resonator to get the best balance of power consumption, system cost, clock accuracy, and bus frequency, some considerations of component selection must be taken of the oscillator circuit to make it works reliably. Figure 4 shows a typical circuit for 455 kHz oscillator.
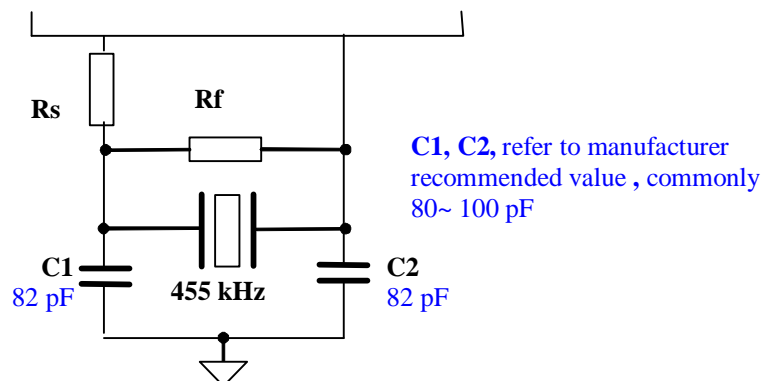
Figure 4. 455 kHz Oscillator Circuit

On the MC9S08QG4 data sheet, two types of characteristics are specified, low range 32 to 38.4 kHz, and high range, greater than 1 MHz. But MC9S08QG4 can work with a 455 kHz resonator, though it is not specified in the electrical characteristics. With such configuration, RANGE, bit 5 of ICSC2 register, must be set, which selects the high range mode of ICS.

Smoke Detector Application Hints Based on MC9S08QG4, Rev. 0

4.2   High Current Stage Upon Power-On Reset

Upon power-on reset (POR) of MC9S08QG4, the default mode of ICS is FEI, in which mode the bus frequency is 4 MHz, this consumes about 2 mA current from power supply, and is not acceptable for a smoke detector application. Set ICS to FBI mode immediately after POR. Although high current stage always exists upon POR, you can shorten the remaining time to several instruction cycles by doing this. After Tcsth, the crystal startup time (5 ms for high-range, low-power mode of oscillator), you can switch ICS to the final working mode, FBE for properly operation. During Tcsth, your program can accomplish the system initialization job, which is not critical for timing.

4.3   Supply Current of Used Internal Modules

Each internal mode draws a current from the power supply when working. Disabling some unused modules can save power, but you must consider the supply current of the used ones for system power consumption. The supply currents of some internal modules of MC9S08QG4 are:

- RTI: 300 nA
- LVD: 70 μA
- ACMP: 20 μA
- ADC: 120 μA (long sample time, low-power mode)
- OSC: 5 μA (low range, low-power mode)
- IRG: 100 μA
- FLL: 210 μA – 300 μA (depends on frequency and supply voltage)
- FLL + DCO: 0.5 – 2.5 mA (depends on frequency and supply voltage)

The I/O pin of MCU must be set properly to save power. All unused I/O must be set as one of these modes:

- Input with internal pull-up
- Output with the consideration of the level affection to the external circuit

5   EEPROM Emulation with Flash Memory

A smoke detector uses EEPROM to store data such as the address and system configuration. 16 bytes are usually sufficient to store those data. Flash memory of MC9S08QG4 can be used to emulate EEPROM. With an internal Vpp charge pump, no external Vpp is needed. Flash memory of MC9S08QG4 can be erased/programmed for 100,00 times, which is 10 times more than that of HC08 MCUs. Furthermore, some solutions are recommended to extend life of flash memory.

5.1 Flash Erasing/Programming Sample Codes

Sample codes for erasing/programming flash memory are in the example directory of CW5.x.

```
if (FSTAT&0x10){                                  //check to see if FACCERR is set
     FSTAT=FSTAT | 0x10;                          //write ad 1 to FACCERR to clear
}
(*((volatile unsigned char*)(Address)))=data;    //write to somewhere in flash
     FSTAT=FSTAT|0x80;                            //put FCBEF at 1
     _asm NOP;                                    //wait 4 cycles
     _asm NOP;
     _asm NOP;
     _asm NOP;
if(FSTAT&030){                                    //check to see if FACCERR or FVIOL
     return 0xFF;                                 //are set, if so, error
}
while((FSTAT&0x40)==0){                           //else wait for command to complete
 ;
}
```

Because it is not safe for the CPU to fetch when Vpp is present, the flash erase/program codes must be executed in RAM rather than in flash memory. To do so, define an array that contains binary codes compiled from the above C source code, and call this array in RAM as a subroutine:

```
//Array of opcode instructions of the erase/program function in the HCS08 family
volatile unsigned char PGM[59]={
0x87,0xC6,0x18,0x25,0xA5,0x10,0x27,0x08,0xC6,0x18,0x25,0xAA,0x10,0XC7,0x18,0x25,
0x9E,0xE6,0x01,0xF7,0xA6,0x20,0xC7,0x18,0x26,0c45,0x18, 0x25,0xF6,0xAA,0x80,0xF7,
0x9D,0x9D,0x9D,0x9D,0x45,0x18,0x25,0xF6,0xF7,0xF6,0xA5,0x30,0x27,0x04,0xA6,0xFF,
0x20,0x07,0xC6,0x18,0x25,0xA5,0x40,0x27,0xF9,0x8A,0x81};
```

PGM[21] is flash memory operation command. These macros can be defined to make it easier to call these codes from RAM as a subroutine:

```
#define Page_Erase PGM[21]=0x40; temp=((unsigned char(*)(unsigned int))(PGM))
#define Program_Byte PGM[21]=0x20; temp=((unsigned char(*)(unsigned int, unsigned char))(PGM))
```

This example shows how to call the functions:

```
unsigned char temp, data;
data=0x55;
temp=Page_Erase(0xF00);
if(temp==0FF){
   //error
}
temp = Program_Byte(0xF00, data);
if(temp==0xFF){
   //error
}
```

Before the flash memory can be properly erased or programmed, FCLK must be set to a correct value (Ffclk) between 150 kHz and 200 kHz.

5.2   Operates Properly at 227.5 kHz f$_{CLK}$

The f$_{CLK}$ characteristic of MC9S08QG4 is between 150 kHz and 200 kHz. The FCDIV register must be written with a proper value to set the right f$_{CLK}$, if PRDIV8, the sixth bit of this register is cleared.

$$f_{CLK} = f_{Bus} \text{ (DIV + 1)}$$ ***Eqn. 1***

But with an external 455 kHz FBE mode, at which the bus frequency is 227.5 kHz, f$_{CLK}$ can only be set to 227.5 kHz (not divided) or 113.75 kHz (divided by 2), which most close to the specified f$_{CLK}$.

For reliable operating, burst programming mode flash memory with 227.5 kHz is recommended. Burst program is different from byte program, so the source code must be re-written for burst mode.

On the other hand, power consumption when flash programming/erasing is larger, FLL adder of supply current can be ignored. Another recommendation is using FLL to get a higher bus frequency, then return to normal setting after flash memory operation is accomplished.

5.3   Elongate Endurance Cycle of Emulated EEPROM

The specified endurance cycle of flash memory of HCS08 MCU is 100,000 cycles. This is sufficient for most applications. For applications need to frequently update EPROM data, we can mathematically elongate the endurance cycle of flash memory.

Flash memory can only be erased by page. Page size of MC9S08QG4 flash memory is 512 bytes. Because there are only a few bytes used for a smoke detector, some solutions can be taken to elongate endurance cycle of the emulated EEPROM. One solution is to store data as records in the whole EEPROM (Figure 5).

Each record consists of n byte data and 1 byte flag which indicates whether this n+1 bytes allocation is used or not. If it is not used, user code can write new data to this location; if it is used, write data to next location. When reading the emulated EEPROM, user codes must check this flag first, and then read data from the last used location, which contains the updated up-to-date data. When all the (page size / record size) locations are used, the next write operation should be re-start from the first address of the page. Thus, the life of this page of flash can be extended by (page size / record size) times.
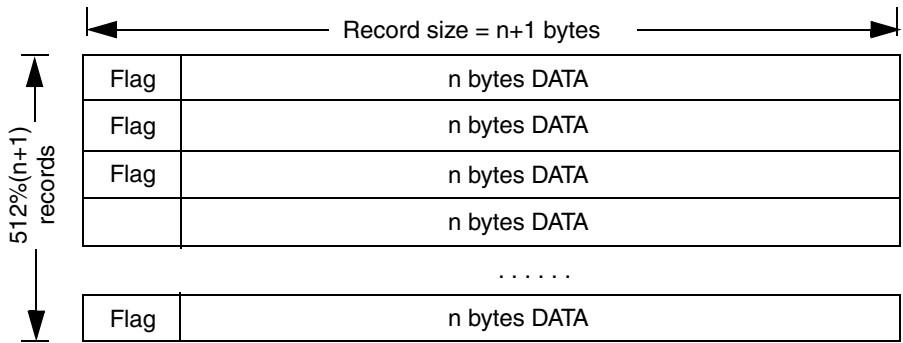

Figure 5. Store Configure Data as Records

If 15 bytes are required to store the address code and configuration data and an extra byte is used as flag, the record size is 16 bytes; therefore a page can be used by 512 / 16 = 32 times, and its life is extended to 32 x 100,000 = 3,200,000 cycles.

5.4 Supply Current when Flash Erasing/Programming

You must consider the supply current when erasing/programming flash memory. When Vdd = 3.2 V, Ffclk = 227.5 kHz, page erasing draws 5.78 mA, and takes 4,000 T_{fcyc}, i.e., 17.58 ms.

Table 4. Flash Programming/Erasing Time

		Spec FCLCK Range		Customer FCLK Range		Unit
Parameter	FCLK Cycles	200	150	227.5	113.75	kHz
Byte Program	9	45.00	60.00	39.56	79.12	µs
Byte Program (burst)	4	20.00	26.67	17.58	35.16	µs
Page Erase	4000	20.00	26.67	17.58	35.16	ms
Mass Erase	20000	100.00	133.33	87.91	175.82	ms

5.5 Flash Block Protection

Because these flash programing/erasing codes are included in your program, flash memory must be block protected to avoid unexpected memory operation caused by program run error. To block protect flash memory, set the NVPROT register with the address of the last unprotected memory. For example, to protect 0xFA00 through 0xFFFF, 0xF9FF is the last unprotected address, so 0xF8 must be written to NVPROT as bit 15–bit 9 (bit 8 ~ bit 0 are all logic 1) of the memory address.

If block protection is enabled, verify that the vectors are redirected or not. If the vectors are redirected by clearing FNORED bit in the NVOPT register, your program must be changed to allocate and access the new vector locations. If a bootloader program is used for in-system programming and updating and it is block protected, vectors must be redirected or the bootloader cannot program the default vector locations because those are also protected if only block protection is enabled.

Normally, an interrupt service routine is declared as:

```
interrupt 3 int lvi_isr();
//the 2nd entry (number 2) gets the address of func g().
interrupt 4 int f();        //OK
//third entry in vector points to f()
```

Number 3 or 4 indicates the absolute vector number of the ISR function. For example, vector 3 of MC9S08QG4 is low-voltage detection interrupt, and the vector location is 0xFFF8:0xFFF9.

This does not work if vector redirection is enabled because after redirecting, the analog comparator vector is moved to address 0xFDF8:0xFDF9 and entry interrupt service from 0xFFF8:0xFFF9 results in a fatal error. Then the interrupt service routine can be declared as:

```
interrupt void lvi_isr; //OK
//same as #pragma TRAP_PROC,
//please set the entry number in the prm-file

//In the PRM file, declare entry with its real address
VECTOR ADDRESS     0xFDD6     lvi_isr
```

Another way to declare the interrupt service routine is to define a function array locates at the start address of vector table.

```
/*Interrupt vector table*/
void (*const _vect[])()@0xFFD0 = {
        UNASSIGNED_ISR,              /* Int.no. 23 Vrti (at FFD0)        */
        UNASSIGNED_ISR,              /* Int.no. 22 Reserved2 (at FFD2)   */
        UNASSIGNED_ISR,              /* Int.no. 21 Reserved3 (at FFD4)   */
        UNASSIGNED_ISR,              /* Int.no. 20 Vacmp (at FFD6)       */
        UNASSIGNED_ISR,              /* Int.no. 19 Vadc (at FFD8)        */
        UNASSIGNED_ISR,              /* Int.no. 18 Vkeyboard (ad FFDA)   */
        UNASSIGNED_ISR,              /* Int.no. 17 Viic (at FFDC)        */
        UNASSIGNED_ISR,              /* Int.no. 16 Vscitx (at FFDE)      */
        UNASSIGNED_ISR,              /* Int.no. 15 Vscirx (at FFE0)      */
        UNASSIGNED_ISR,              /* Int.no. 14 Vscierr (at FFE2)     */
        UNASSIGNED_ISR,              /* Int.no. 13 Vspi (at FFEE4)       */
        UNASSIGNED_ISR,              /* Int.no. 12 Vmtim (at FFE6)       */
        UNASSIGNED_ISR,              /* Int.no. 11 Reserved13 (at FFF8)  */
        UNASSIGNED_ISR,              /* Int.no. 10 Reserved14 (at FFEA)  */
        UNASSIGNED_ISR,              /* Int.no. 9 Reserved15 (at FFEC)   */
        UNASSIGNED_ISR,              /* Int.no. 8 Reserved16 (at FFEE)   */
        UNASSIGNED_ISR,              /* Int.no. 7 Vtpmovf (at FFF0)      */
        UNASSIGNED_ISR,              /* Int.no. 6 Vtpmch1 (at FFF2)      */
        UNASSIGNED_ISR,              /* Int.no. 5 Vtpmch0 (at FFF4)      */
        UNASSIGNED_ISR,              /* Int.no. 4 Reserved20(at FFF6)    */
        lvi_isr,                     /* Int.no. 3 Vlvd (at FFF8)         */
        UNASSIGNED_ISR,              /* Int.no. 2 Virg (at FFFA)         */
        UNASSIGNED_ISR,              /* Int.no. 1 Vswi (at FFFC)         */
        _Startup                     /* Int.no. 0 Vreset (at FFEE) Reset vector*/
```

In case vectors are redirected, change the function array address to the new location:

```
void (*const _vect[])() @0xFDD0={
        UNASSIGNED_ISR,              /* Int.no. 23 Vrti (at FFD0)        */
        //
        lvi_isr,                     /* Int.no. 3 Vlvd (at FFF8)         */
        UNASSIGNED_ISR,              /* Int.no. 2 Virq (at FFFA)         */
        UNASSIGNED_ISR,              /* Int.no. 1 Vswi (at FFFC)         */
        UNASSIGNED_ISR,              /* Int.no. 0 Vreset (at FFFE) Reset vector*/}
```

In any case, the reset vector will not be redirected. It is always 0xFFFE:0xFFFF; therefore, the _Startup function address must not be changed. Define it in PRM file separately.

6 ADC Configuration

In configuring the ADC, you must select the right ADC clock.

The ADC can perform conversions by using the MCU bus clock, MCU bus clock divided by two, or the local asynchronous clock within the module. If the bus frequency is less than the f_{ADCK} frequency, precise sample time for continuous conversions cannot be guaranteed when short sample is enabled. If the bus frequency is less than 1/11th of the f_{ADCK} frequency, the precise sample time for continuous conversions cannot be guaranteed when a long sample is enabled.

As a 455 kHz resonator is used, and the bus frequency is 227.5 kHz, the asynchronous clock (ADACK) must be selected as clock source for ADC.

Appendix A References

MC9S08QG4 data sheet, Freescale Semiconductor

AN2493, MC9S08GB/GT Low Power Modes, Freescale Semiconductor

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 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

Document Number: AN3505
Rev. 0
08/2007

freescale™
semiconductor