Freescale Semiconductor
Application Note

Document Number: AN3605
Rev. 0, 05/2008

Interfacing Analog Accelerometer MMA73x0L in Mobile Applications

by: Vincent Ko
System and Application Engineering
Asia & Pacific Operation
Global Sales and Marketing Group

This document defines a low-g acceleration sensor system which might be used in mobile applications. The system is referred to as a LOW-G unit. This document defines the interface between the mobile system and the LOW-G unit, and also its modes of operation. This is a reference design that comes with software to use the MMA7360 for mobile applications.

Contents

1 System Implementation

This section discusses the system implementation needed to integrate the Freescale low-g acceleration sensor (low-g sensor) into a typical mobile communication platform. In most mobile communication platforms, IIC is the popular inter-communication interface among the onboard devices. Since a mobile processor normally does not come with an analog-to-digital converter (ADC), in this case a microcontroller (MCU) can act as a sensor interface. It is therefore more effective to include a low-cost (MCU) within the system to interface between the sensor and the baseband processor. A highly integrated device, the MC9S08QG4, is used for this purpose. In this document the system, including the MCU and the low-g acceleration sensor, is called the LOW-G unit.

The LOW-G unit has several advantages.

- All low-level control, such as ADC sampling, is managed by the MCU within the unit.
- The baseband processor interface has been simplified — it is now an IIC interface.
- The unit offloads the baseband processing bandwidth. A low-level algorithm on the sensor is managed by the MCU, and only high-level information is reported back to the baseband processor via the IIC interface. During this time, the baseband processor can be turned off completely to save power.

Figure 1 shows the integration system block diagram. The LOW-G unit is powered by the supply from the power management unit in the mobile platform. It can range from 2.2 V to 3.6 V. Five pins are involved in the communication between the baseband processor and the unit. They are:

- $\overline{\text{INT}}$ — Interrupt signal
- SDA — IIC data communication
- SCL — IIC clock communication
- $\overline{\text{CE}}$ — System enable signal
- $\overline{\text{RST}}$ — Master reset

The $\overline{\text{INT}}$ pin is a hardware interrupt pin and is logic high by default. The baseband processor and the LOW-G unit are operated under an interrupt-driven mechanism. When the unit requires service from the baseband processor, _$\overline{\text{INT}}$ is pulled low. The processor can read and clear the source of the interrupt through a dedicated IIC command. When the interrupt is cleared, the $\overline{\text{INT}}$ pin returns to logic high.

The IIC bus uses a serial data line (SDA) and a serial clock line (SCL) for data transfer. All devices connected to it must have open-drain or open-collector outputs. A logic AND function is exercised on both lines with external pullup resistors.

The $\overline{\text{CE}}$ pin is a hardware signal to put the complete system into stop mode. If stop mode is entered, the system drives minimum power. All functionality is stopped and no IIC communication is allowed.

The $\overline{\text{RST}}$ pin is the master reset to the unit. Master reset can be asserted in any software mode. During firmware upgrade mode, if the master reset is asserted before the firmware programming is completed, the unit will reset to firmware upgrade mode. Erasing and reprogramming the entire firmware is then required.

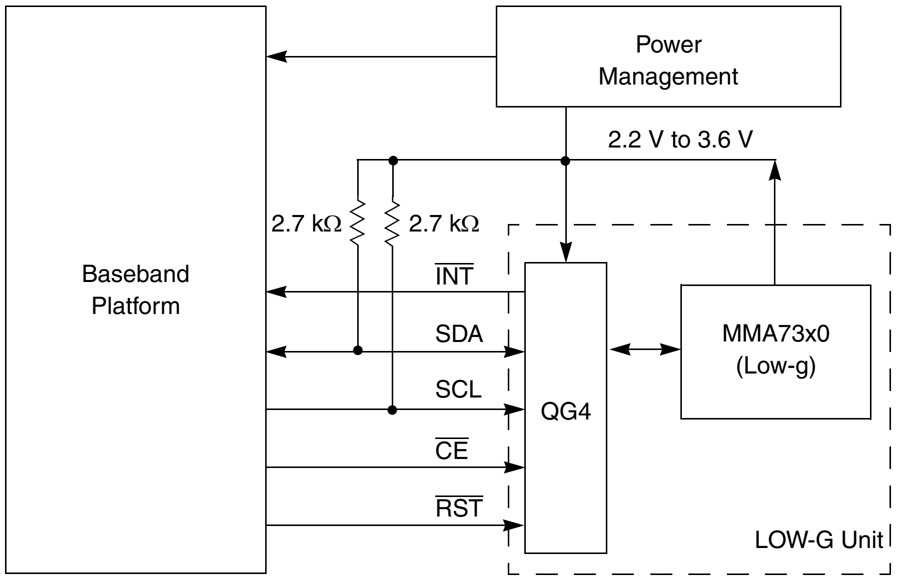


Figure 1. System Block Diagram

Figure 2 shows the schematic of the LOW-G unit. Two interfaces are shown:

- Master interface — connection to the baseband processor
- Background debug interface — for in-circuit debugging, and for MCU flash programming when the device is completely blank

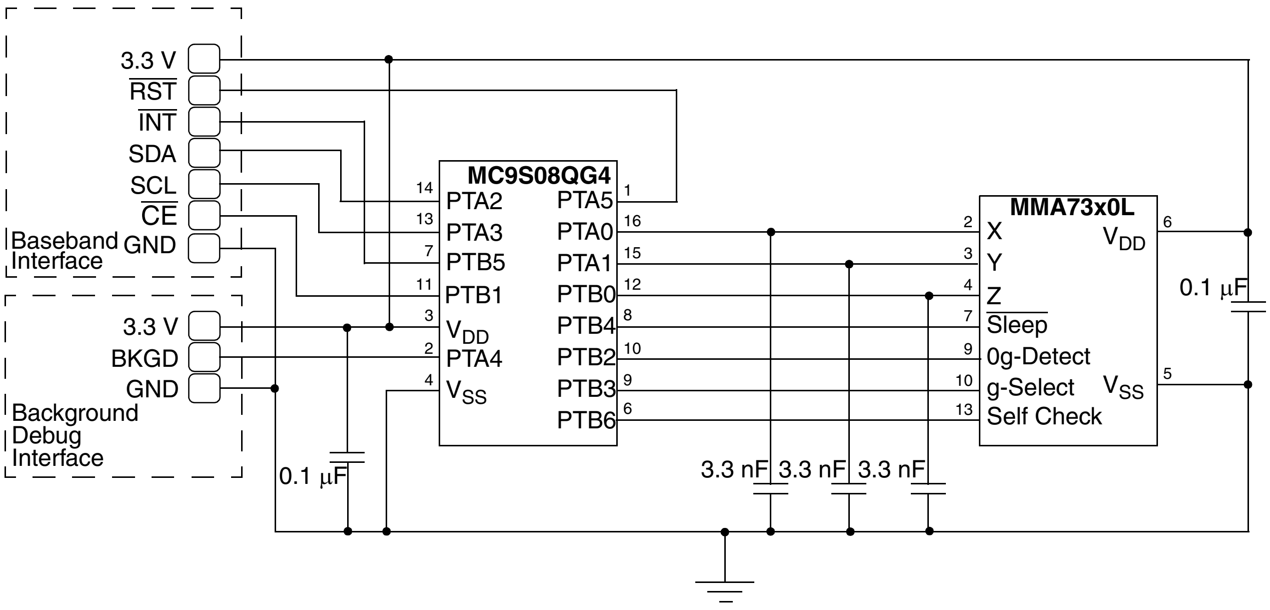


Figure 2. LOW-G Unit Schematic

2 IIC Interface

This section discusses the IIC communication format and its low-level protocol required for mobile communication.

2.1 Data Format

The data format is based on the IIC communication standard. The LOW-G unit interface is defined as slave mode only, and its maximum communication speed is 100 kbps.

Standard communication is composed of four parts:

- START signal
- Slave address
- Data transfer
- STOP signal

Figure 3 shows a typical single-byte communication format.

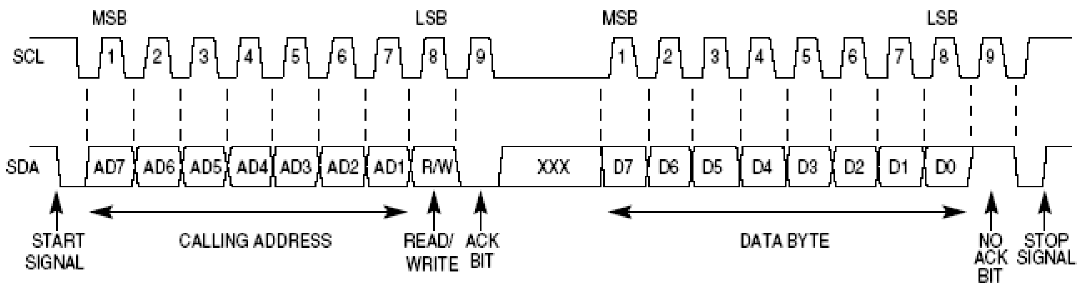


Figure 3. IIC Bus Data Format

Each data byte is eight bits long. Data may be changed only when SCL is low, and must be held stable when SCL is high. There is one clock pulse on SCL for each data bit, the MSB being transferred first. Each data byte is followed by a 9th (ACK) bit, which is signalled from the receiving device.

If the slave (LOW-G unit) receiver does not acknowledge the master the 9th bit time, the SDA line must be left high by the slave. The master interprets the failed acknowledge as an unsuccessful data transfer.

If the master (baseband processor) receiver does not acknowledge the slave transmitter after a data byte transmission, the slave interprets this as an end-of-data transfer and releases the SDA line.

In either case, the data transfer is aborted and the master does one of two things:

- Relinquish the bus by generating a STOP signal.
- Commence a new calling by generating a repeated START signal.

3 IIC Protocol

This section describes various IIC protocols. Figure 4 shows the bit field symbols definition. The addressing byte is used as an example.

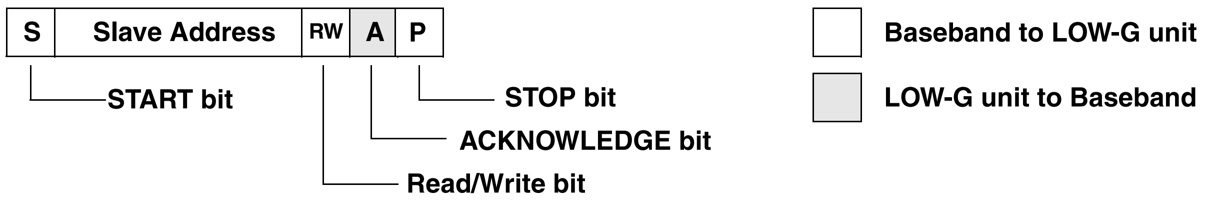


Figure 4. Protocol Symbol Definition

3.1 Single Byte Receiving Mode

A single byte is received from the master (baseband processor).

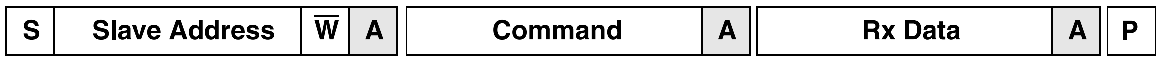


Figure 5. Single Byte Receiving

3.2 N Bytes Receiving Mode

N bytes are received from the master (baseband processor).

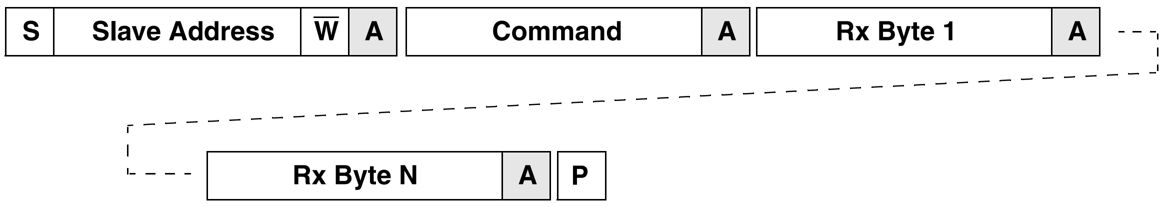


Figure 6. N Bytes Receiving

3.3 Single Byte Transmission Mode

A single byte is transmitted to the master (baseband processor).

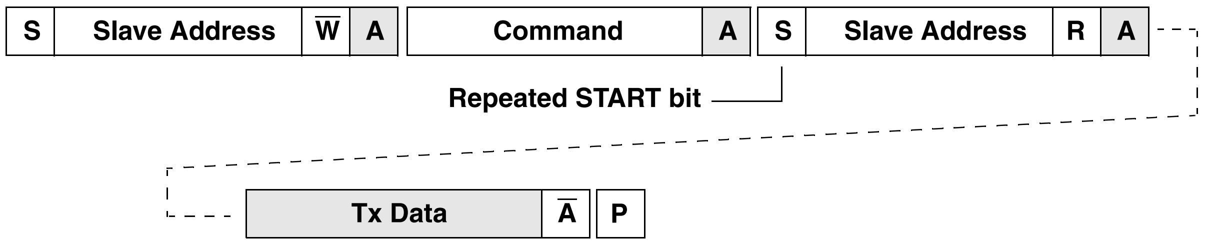


Figure 7. Single Byte Transmission

3.4 N Bytes Transmission Mode

N bytes are transmitted to the master (baseband processor).

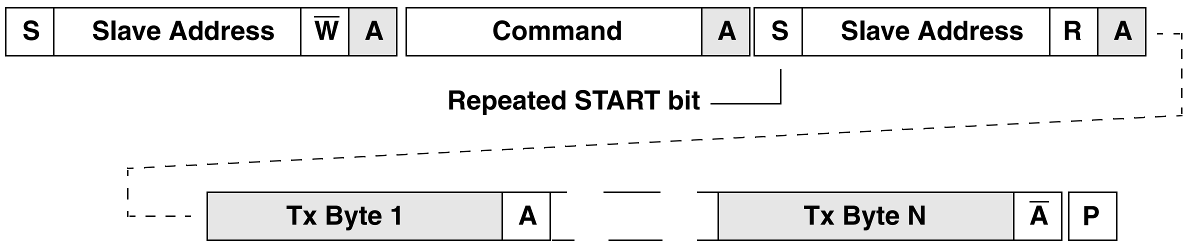


Figure 8. N Bytes Transmission

4 Modes of Operation

There are four generic modes of operation in the LOW-G unit.

- Stop mode
- Standby mode
- Default mode
- Firmware upgrade mode

4.1 Stop Mode

- All operations of the LOW-G unit are stopped.
- IIC communication is not allowed.
- It consumes minimum power from the supply.

4.2 Standby Mode

- The sensor unit is stopped when the MCU is in standby mode.
- IIC communication is allowed.
- The slave address for the sensor unit is 0x1E, where the last bit defines the communication direction.

4.3 Default Mode

In default mode, the LOW-G unit performs regular data sampling to the low-G sensor. The X, Y, and Z axis values can be polled by command Read_XYZ. The axis value read by the command Read_XYZ is deduced by Equation 1.

$$\text{Axis Value} = \text{Sensor Output} - \text{Calibration Value} - \text{Offset} \qquad \textit{Eqn. 1}$$

The calibration value is set by the command Unit_Cal. After it is set, the value cannot be modified. The offset value per axis is defined by the commands Chg_X_Offset, Chg_Y_Offset, and Chg_Z_Offset, which offset the calibration value. The offset range is from –32 to +32. The offset values return to zero after power is removed from the LOW-G unit.

When this mode is first entered, the sensor sampling rate is set at 64 ms, and its sensitivity level is by default set to the highest sensitivity range. For example, when MMA7360L is used, the default sensitivity level range is 1.5 g. The sampling rate can be changed by the command Unit_Rfrh. The sensitivity level can be changed by the command Unit_Range.

In this mode, threshold detection is allowed. The sensitivity level can be chosen by using the command Thrh_Sel. When this feature is enabled, the sampled sensor value is compared against the predefined threshold. If the sampled value is larger than the threshold, interrupt pin $\overline{\text{INT}}$ is asserted (output low) to notify the baseband system that sudden movement has been detected. This feature can be used as for theft detection or motion detection in the system. Interrupt pin $\overline{\text{INT}}$ is cleared by command Int_Ack. The $\overline{\text{INT}}$ pin returns to its initial state (output high).

4.4 Firmware Upgrade Mode

This mode can only be entered from default mode. It is used to re-flash the MCU firmware area. The MCU firmware has two portions — the firmware portion and the bootloader portion.

Executing the command FW_Erase erases the entire MCU firmware area.

The entire memory map of the MCU is separated into 64-byte blocks. Each of these blocks is referred to by an identity number, the Block_ID. Each Block_ID corresponds to a specific address block of the MCU memory. When firmware programming is performed, the Block_ID, followed by the 64 bytes of data to be programmed, must be specified, together with the command FW_Prog.

During the firmware upgrade process, the bootloader portion is untouched. Both the command FW_Erase and the command FW_Prog perform no operation on the bootloader portion of the MCU memory.

When the firmware upgrade is done, the user must issue the command Prog_Compl to re-initiate the LOW-G operation. The command Prog_Compl returns the unit to stop mode. The unit starts as it does from power-up.

5 Generic IIC Command Set

Table 1 describes the generic IIC command set summary used to control and enable the low-g sensor features. Please note that some commands are dedicated to a particular mode of operation. They will not respond to the host if they are not received in the defined mode of operation.

Table 1. Generic IIC Command Set

Command	Name	Direction	Mode	Description	Data Length	Data Format
0x80	Unit_Dis	M to S	Default	Put LOW-G unit in standby mode.	0	
0x81	Unit_Cal	M to S	Default	Sensor Calibration. Perform sensor calibration at the current g selection defined by command Unit_Range.	1	1 — Calibrate X axis at 0 g 2 — Calibrate Y axis at 0 g 4 — Calibrate Z axis at 0 g Other — No operation
0x82	Unit_Range	M to S	Default	Select g value from four sensitivity levels. Default is 2 g.	1	1 — 1.5 g 2 — 3 g 4 — 4 g 8 — 6 g 16 — 12 g 32 — 16 g Other — No operation Default setting is the highest sensitivity range of the sensor device used. If the selected range is not included in the sensor device, the operation will be ignored.
0x83	Unit_Rfrh	M to S	Default	Select the sensor refresh rate. Default is 64 ms.	1	1 — 8 ms 2 — 32 ms 3 — 64 ms (Default) 4 — 128 ms 5 — 256 ms 6 — 512 ms 7 — 102 ms
0x84	Thrh_Sel	M to S	Default	Enable threshold detection for X, Y, and Z. The threshold value is the low nibble (4-bit LSB) multiplied by 4.	1	Bit4 = 1 Select threshold for X axis Bit5 = 1 Select threshold for Y axis Bit6 = 1 Select threshold for Z axis Bit3:Bit0 Value: 0 — Disable threshold detection Threshold = (Bit3:Bit0) x 4 For example, 0x16 = set X-axis threshold as value 24.
0x12	Int_Ack	M to S	All[1]	Clear all pending interrupt flags and release $\overline{\text{INT}}$ pin.	0	
0x85	Read_XYZ	S to M	Default, free fall (free fall not used)	Read X, Y, and Z axis value.	6	Byte 1 — High byte of X axis value Byte 2 — Low byte of X axis value Byte 3 — High byte of Y axis value Byte 4 — Low byte of Y axis value Byte 5 — High byte of Z axis value Byte 6 — Low byte of Z axis value

Table 1. Generic IIC Command Set (continued)

Command	Name	Direction	Mode	Description	Data Length	Data Format
0x86	Chg_X_Offset	M to S	All[1]	Change calibration offset for X, Y, and Z axis. The offset range is +32 to –32.	1	Byte 1 — Signed offset value (–32 to +32).
0x87	Chg_Y_Offset	M to S	All[1]	Change calibration offset for X, Y, and Z axis. The offset range is +32 to –32.	1	Byte 1 — Signed offset value (–32 to +32).
0x88	Chg_Z_Offset	M to S	All[1]	Change calibration offset for X, Y, and Z axis. The offset range is +32 to –32.	1	Byte 1 — Signed offset value (–32 to +32).
0x10	Unit_Mod	M to S	All[1]	Select mode.	1	0 — Return to default mode 1 — Pedometer enable 2 — Free fall mode enable (not used) 4 — Rolling dice mode enable 8 — Swing detection mode enable 16 — Rotate detection mode enable 32 — Wave messaging mode enable 128 — Firmware upgrade enable Other — No operation
0x11	FW_ID	S to M	All[1]	Return device firmware ID.	2	Byte 1 — ID1 Byte 2 — ID2
0x89	FW_Status	S to M	All[1]	Read pending interrupt flag.	2	Byte 1: Bit0 — X threshold flag is pending Bit1 — Y threshold flag is pending Bit2 — Z threshold flag is pending Bit3 — Free fall flag is pending (not used) Bit4 — Pedometer upcounting flag Bit5 — Dice start rolling flag Bit6 — Dice stop rolling flag Bit7 — Swing detection flag Byte 2: Bit0 — Orientation change flag Other bits are not used.

Table 1. Generic IIC Command Set (continued)

Command	Name	Direction	Mode	Description	Data Length	Data Format
0x20	FW_Erase	M to S	Firmware upgrade	Erase entire firmware area.	0	
0x21	FW_Prog	M to S	Firmware upgrade	Program flash block indicated by Block_ID.	65	Byte 1 — Block_ID Byte 2 — Data[0] Byte 3 — Data[1] Byte 4 — Data[2] ...... Byte 65 — Data[63]
0x22	Prog_Cmpl	M to S	Firmware upgrade	Firmware upgrade completed, return to stop mode.	0	
0x23	Ready_Chk	M to S	Firmware upgrade	Check if the previous programming is successful.	1	55 — Successful Other — Not successful

1 All modes except stop mode.

6 Additional Software Components

In addition to the generic modes, there are several software components that can be chosen based on the application requirements. With a 4K flash programming space, any two of the software components can be selected. Compilation is required for any change in software component selection.

- Pedometer mode
- Rolling dice game mode
- Swing detection mode
- Rotate detection mode
- Wave messaging mode[1]

6.1 Pedometer Mode

This mode is entered by using the command Unit_Mod. When entering this mode, all required sensor settings are automatically pre-defined and preset. The user simply performs START, STOP, and CLEAR functions by using the command Pedo_Ctrl. On each pedometer count, the increment interrupt asserts. The interrupt pin $\overline{\text{INT}}$ is cleared by the command Int_Ack, and the $\overline{\text{INT}}$ pin returns to its initial state (output high). The baseband host can read the pedometer counter value by using Pedo_Read.

If the unit leaves pedometer mode while the pedometer counter is running, the counter will be stopped. When the unit re-enters pedometer mode, counting resumes.

1. When using wave messaging mode, a different MCU, other than MC9S08QG4, must be used to accommodate the additional port that drives the LEDs. See Section 6.5, "Wave Messaging Mode," for further information.

Table 2. Command Set for Pedometer Mode

Command	Name	Direction	Mode	Description	Data Length	Data Format
0x40	Pedo_Ctrl	M to S	Pedometer	Pedometer control	1	0 — Clear counter 1 — Start counting 2 — Stop counting Other — no operation
0x41	Pedo_Read	S to M	Pedometer	Read pedometer count	4	Byte 1 — Pedometer count (high byte) Byte 2 — Pedometer count (3rd byte) Byte 3 — Pedometer count (2nd byte) Byte 4 — Pedometer count (low byte)

6.2 Rolling Dice Game Mode

This mode selects the algorithm for six dice[2] rolling together based on the sensor movement. Sensor movement in directions X, Y, and Z will flip the die face accordingly. This algorithm presumes six dice rolling at the same time. Each die experiences a slightly different rotational force, such that the rotational results of the dice are different over time.

When this mode is entered, the low-G sensor is enabled. It continuously monitors the movement of the unit. When a dramatic change in sensor movement is detected, the dice rolling algorithm starts and interrupt pin $\overline{\text{INT}}$ is asserted. When the sensor movement is stopped, the dice rolling algorithm will continue to run for a a few seconds and then stop. Interrupt pin $\overline{\text{INT}}$ is asserted again. The algorithm stopping time depends on the time taken to roll the dice. The longer it is, the more delay there is before the algorithm stops. The longest delay is about four seconds. The corresponding interrupt flag can be read by command FW_Status to understand the source of the interrupt. The interrupt condition is cleared by the command Int_Ack.

The rolling dice result can be read by command Dice_Read1, Dice_Read2, Dice_Read3, Dice_Read4, Dice_Read5, and Dice_Read6, according to the die number to be read.

Before entering rolling dice mode, the individual die value can be set by command Dice_Init. The high nibble value bit[6:4] of the parameter defines the die number to be set and the low nibble bit[3:0] defines the initial value of the die.

2. In English, the word dice is plural and the singular form of the word is die. For convenience, not all commands and names that are used here make this distinction.

Table 3. Command Set for Rolling Dice Mode

Command	Name	Direction	Mode	Description	Data Length	Data Format
0x42	Dice_Init	M to S	All[1]	Define initial value of die.	1	Bit[6:4] — Die number Dice 1 = 0 Dice 2 = 1 .... Dice 6 = 5 Bit[3:0] — Die initial value Byte 1 = 0, Set all dice to value 1
0x43	Dice_Read1	S to M	Rolling dice	Read die 1 value.	1	Byte 1 — Die value
0x44	Dice_Read2	S to M	Rolling dice	Read die 2 value.	1	Byte 1 — Die value
0x45	Dice_Read3	S to M	Rolling dice	Read die 3 value.	1	Byte 1 — Die value
0x46	Dice_Read4	S to M	Rolling dice	Read die 4 value.	1	Byte 1 — Die value
0x47	Dice_Read5	S to M	Rolling dice	Read die 5 value.	1	Byte 1 — Die value
0x48	Dice_Read6	S to M	Rolling dice	Read die 6 value.	1	Byte 1 — Die value

6.3 Swing Detection Mode

This mode detects swing motion in six basic directions. The motion detection is triggered by a roughly 1.5 g swing movement. When a valid motion is detected, interrupt asserts and the $\overline{\text{INT}}$ pin is pulled low. Issue the command Int_Ack to clear the interrupt flag and the $\overline{\text{INT}}$ pin returns to logic high. The swing motion that has been detected can be read by command Swing_Read.

Table 4. Command Set for Swing Detection Mode

Command	Name	Direction	Mode	Description	Data Length	Data Format
0x49	Swing_Read	S to M	Swing detection	Read swing direction	2	Byte 1 — Swing Direction X Positive = 1 X Negative = 2 Y Positive = 3 Y Negative = 4 Z Positive = 5 Z Negative = 6 Byte 2 — Swing Magnitude Range is from 1 to 255, where 1 is the lightest and 255 is the heaviest.
0x4B	Swing_Sen	M to S	Swing detection	Select swing detection sensitivity	1	Byte 1 — sensitivity 1 = Most sensitive ... 10 = Least sensitive Default value = 5

6.4 Orientation Detection Mode

This mode detects the orientation of the sensor unit. There are four elements that can be detected by this mode. X_UP defines the X axis as facing up; X_DOWN defines the X axis as facing down; and so on. Given the orientation result, the system can define its landscape and portrait position. Current orientation of the system can be read by command Rotate_Read.

Table 5. Command Set for Orientation Detection

Command	Name	Direction	Mode	Description	Data Length	Data Format
0x4A	Rotate_Read	S to M	Orientation detection	Read orientation.	1	Byte 1 — Orientation Unknown = 0 X Up = 1 X Down = 2 Y Up = 3 Y Down = 4

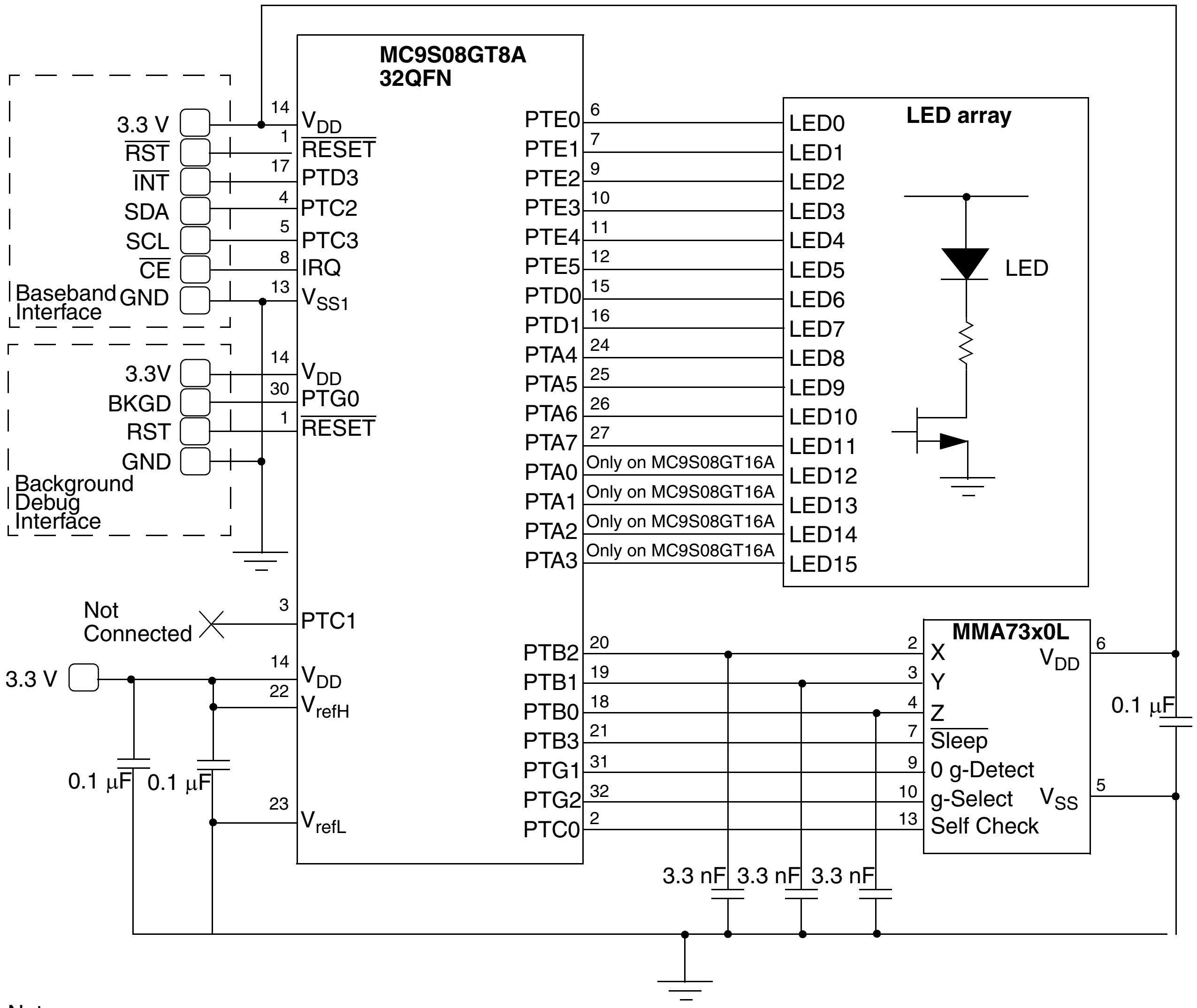
NOTE

When swing detection and orientation detection are both chosen during compilation, swing detection mode and orientation detection mode can be enabled at the same time when bit 3 and bit 4 are set for command Unit_Mod. The user can identify the source of an interrupt through the corresponding bits set in the FW_Status command.

6.5 Wave Messaging Mode

This mode detects hand swing motion and uses a column of LEDs to display the pre-defined characters or picture accordingly. The detection algorithm adapts to the swinging speed and magnitude. The display window will be adjusted so that the picture or all characters that display will adapt to the window width.

When using wave messaging mode, a different system schematic must be used. The MCU is replaced by MC9S08GTxA. The additional column of LEDs is driven by the MCU general-purpose input/output ports (GPIO). MC9S08GT8A can support up to 12 LEDs; MC9S08GT16A can support up to 16 LEDs.



Note:
Pin numbers here are based on the MC9S08GT8A 32QFN package.

Figure 9. Schematic When Using Wave Messaging Mode

6.5.1 Display Window

The display window can support up to 16 LEDs in a column. The user is allowed to cut down the number of LEDs based on the application need, or on the characters or picture type to be displayed. Each column in a display window is represented by a word, or two bytes — the entire display RAM is arranged as a sequential word. Each bit in a word represents a particular pixel. Figure 10 shows the bit arrangement for the display window.

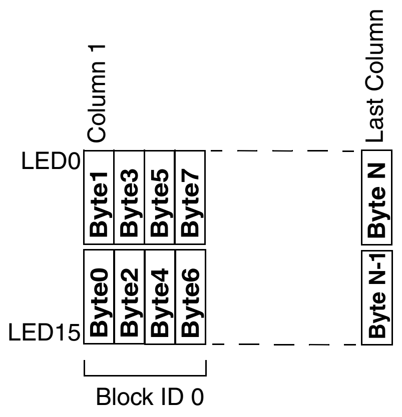


Figure 10. Bit Arrangement of a Display Window

The display window size is defined during the firmware compilation time. The default value is 76 columns, which is four 16 x 16 Chinese characters plus 4 columns of space in between. An example of the display is shown in Figure 11. In this case, byte 1 through byte 32 represent the first Chinese character. An English character display can be arranged in a similar manner. An example is shown in Figure 12 — byte 1 through byte 12 represent the first letter, which is "F."

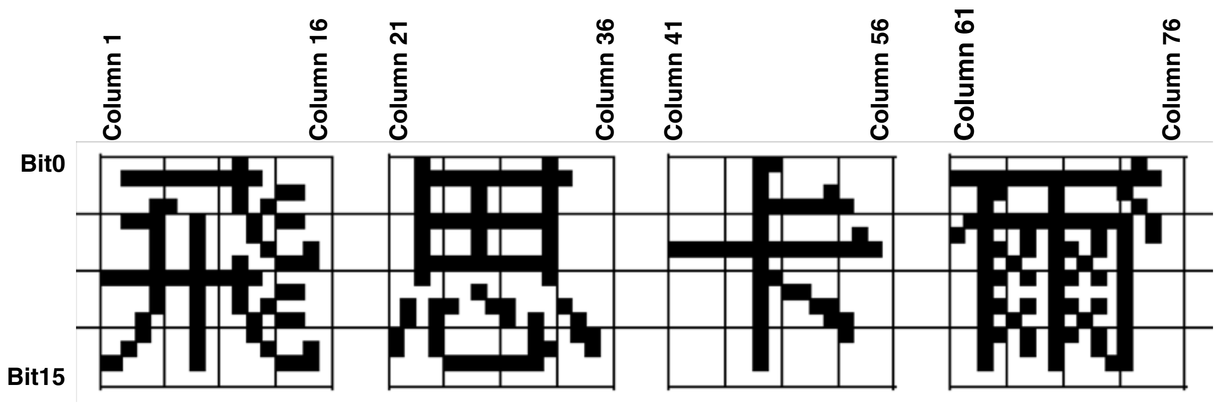


Figure 11. An Example of a Display of Chinese Characters

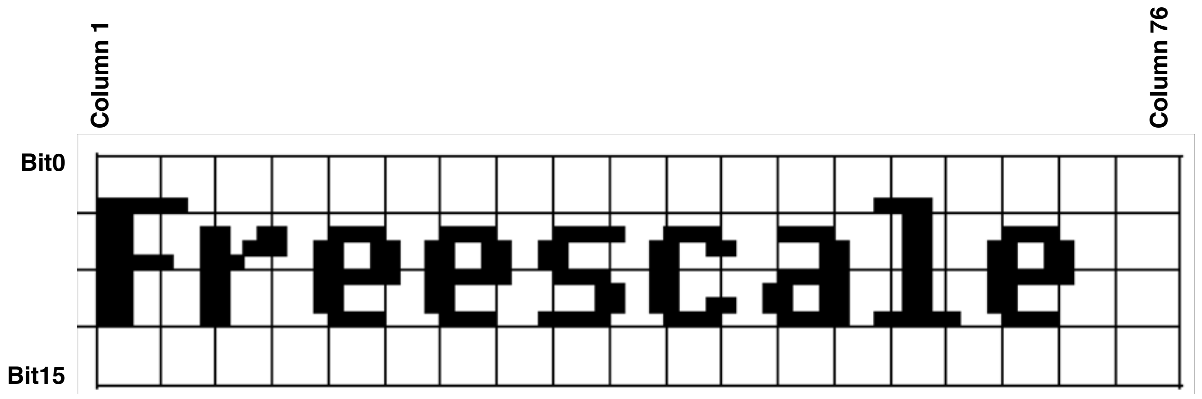


Figure 12. An Example of a Display of English Characters

6.5.2 Command Set for Wave Messaging

The display window is separated into a set of 4-column blocks. For example, column 1 to column 4 belong to one block — column 4 to column 8 belong to another block — and so on. Each block contains eight bytes of memory and can be updated through the dedicated IIC command (0x15 Prog_Disp_Content). The command defines the block ID as the first byte of data sent. Block ID 0 implies the display RAM content as column 1 to column 4 or display RAM byte 0 to byte 7. Similarly, block ID 1 implies the display RAM content for column 5 to column 8 or display RAM byte 8 to byte 15, and so on.

Table 6. Additional Command Set for Wave Messaging Mode

Command	Name	Direction	Mode	Description	Data Length	Data Format
0x14	Flush_Display	M to S	Wave messaging	Program the entire display RAM with a fixed value.	1	Byte 1 — Value to be programmed to the display RAM
0x15	Prog_Disp_Content	M to S	Wave messaging	Program the display content with a specific value.	9	Byte 1 — Display block ID Byte 2 — Display RAM[offset+0] Byte 3 — Display RAM[offset+1] ... Byte 9 — Display RAM[offset+7] Where offset = Block_ID*8

7 Calibration

From device to device, the low-g sensor output may have up to ±10% variation. Calibration should be performed on a device-to-device basis at least once during manufacturing. The LOW-G unit includes a calibration command, Unit_Cal. It can calibrate the 0-g value of the individual axis at its selected sensitivity level — X, Y, and Z — and stores the values in the LOW-G internal flash. For example, if a 1.5 g range is selected when issuing the calibration command, the 0-g value at 1.5 g range will be stored.

The user can use the commands Chg_X_Offset, Chg_Y_Offset, and Chg_Z_Offset to offset the saved calibration value and fine-tune the performance of the sensor. After power-on reset, the offset value returns to zero. However, the saved calibration value remains unchanged. After the firmware upgrade, the calibration process does not need to be performed again. The saved calibration value is not altered by the firmware upgrade process.

The calibration process for each axis can be done only seven times — after that the new calibration value will not be saved to the unit's internal flash. The seventh calibration value will continue to be used.

8 State Diagram

Figure 13 shows the state diagram of the LOW-G unit.

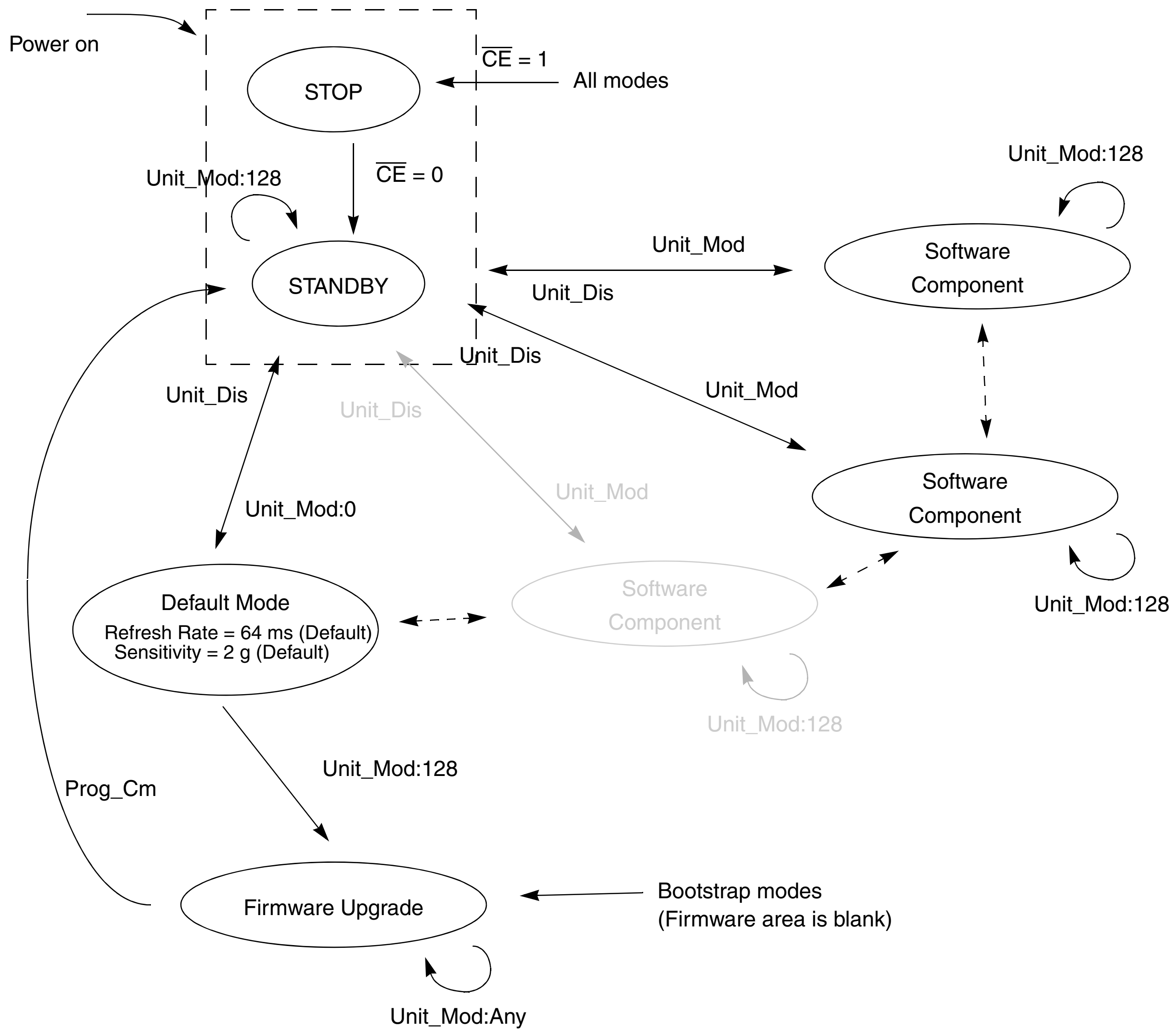


Figure 13. LOW-G Unit State Diagram

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: AN3605
Rev. 0
05/2008