

Wireless Sensing Triple Axis Reference design (ZSTAR) Designer Reference Manual

ZSTARRM
Rev. 0.9
8/2006

freescale.com



USA:

**This device complies with Part 15 of the FCC Rules.
Operation is subject to the following two conditions:
(1) this device may not cause harmful interference,
and (2) this device must accept any interference**

Canada:

**This Class [*] digital apparatus complies with Canadian ICES-003.
Cet appareil numérique de la classe [*] est conforme à la norme NMB-003 du Canada.**

Europe:

TBD

Warning:

Any changes or modifications to this device not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

Wireless Sensing Triple Axis Reference design

Designer Reference Manual

by: Pavel Lajšner and Radomír Kozub
Freescale Czech Systems Laboratories
Rožnov pod Radhoštěm, Czech Republic

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:

<http://www.freescale.com>

The following revision history table summarizes changes contained in this document. For your convenience, the page number designators have been linked to the appropriate location.

Revision History

Date	Revision Level	Description	Page Number(s)
May, 2006	0.02	First draft	N/A
July 2006	0.9	Language correction, minor updates	N/A

Revision History

Table of Contents

Chapter 1 Introduction

1.1	Introduction	9
1.2	MMA7260Q 3-axis Accelerometer Sensor	9

Chapter 2 Wireless Sensing Triple Axis Reference design introduction

2.1	Introduction	11
2.2	Featured products	12
2.2.1	Triple Axis Accelerometer MMA7260Q	12
2.2.2	Microcontroller MC9S08QG8	12
2.2.3	MC13191 2.4 GHz ISM Band Low Power Transceiver	13
2.2.4	Microcontroller MCHC908JW32	13

Chapter 3 Sensor Board description

3.1	Board overview	15
3.2	A/D conversion of XYZ levels	17
3.2.1	ADC module init	17
3.2.2	ADC measurement	18
3.3	Power management	18
3.3.1	MC13191 power management features	20
3.4	ZSTAR Sensor Board hardware overview	21
3.4.1	Analog connections	21
3.4.2	g-select connections	21
3.4.3	BDM (Background Debug Mode) connections	21
3.4.4	Sensor Board schematics	22
3.4.5	Button connections	23
3.4.6	MC13191 to MC9S08QG8 microcontroller interface	23
3.4.7	MC13191 RF interface	23
3.4.8	Clocking options of MC9S08QG8	24
3.4.9	LED indicators connections	24
3.4.10	Power supply	25
3.5	Bill of Materials	26

Chapter 4 USB stick board description

4.1	Board overview	27
4.2	ZSTAR USB stick Board hardware overview	30
4.2.1	USB connections	30
4.2.2	Power supply	30
4.2.2.1	Fixed voltage regulators	30

Table of Contents

4.2.3	MC13191 to MCHC908JW32 microcontroller interface	31
4.2.4	Oscillator and clocking options	31
4.2.5	LED indicators connections	31
4.2.6	Button connection	32
4.2.7	MON08 interface	32
4.2.8	Optional serial interface	32
4.2.9	USB stick schematics	33
4.3	Bill of Materials	34

Chapter 5 Software Design

5.1	Introduction	35
5.2	SMAC (Simple Media Access Controller)	35
5.2.1	SMAC Features	35
5.2.2	Modifications of SMAC for ZSTAR demo	35
5.2.2.1	MC9S08QG8 SMAC modifications (Sensor Board)	36
5.2.2.2	MCHC908JW32 SMAC modifications (USB stick)	36
5.2.2.3	Generic SMAC modifications (USB stick + Sensor Board)	37
5.3	ZSTAR RF protocol	38
5.3.1	Zpacket format	38
5.3.1.1	Network number	39
5.3.1.2	RX strength	39
5.3.1.3	Zcommand	39
5.3.1.4	Zdata	39
5.3.2	ZSTAR protocol Zcommand description	40
5.3.2.1	ZSTAR_BROADCAST	40
5.3.2.2	ZSTAR_CONNECT	40
5.3.2.3	ZSTAR_DATA	40
5.3.2.4	ZSTAR_ACK	41
5.3.2.5	ZSTAR_CALIB	41
5.3.2.6	ZSTAR_STATUS	41
5.4	STAR protocol and ZSTAR extensions (over USB)	42
5.4.1	Communication handshake 'R' (0x52)	42
5.4.1.1	Extended Communication handshake 'r' (0x72)	42
5.4.2	Accelerometer data transfer 'V' (0x56)	42
5.4.2.1	Extended Accelerometer data transfer 'v' (0x76)	43
5.4.3	Calibration data 'K' (0x4B)	43
5.4.4	Calibration process 'k' (0x6B)	44
5.4.4.1	Remaining STAR demo commands	44
5.4.5	Additional ZSTAR commands	44
5.4.5.1	g-select reading 'G' (0x47)	44
5.4.5.2	g-select setting 'g' (0x67)	45
5.4.5.3	Info 'I' (0x49)	45
5.4.5.4	Debug on 'U' (0x55) and Debug off 'u' (0x75)	45
5.4.6	Further debug and test commands	46
5.4.6.1	Forced channel number selection	46
5.4.6.2	Semiautomatic self-calibration	46
5.5	Bootloader	47

5.5.1	Bootloading procedure	48
5.5.2	Dualboot guidelines	51
5.5.2.1	Dualboot applications switching	52

Chapter 6
Application Setup

6.1	ZSTAR Installation Procedure	53
6.1.1	USB stick installation	53
6.1.2	AN2295 Bootloader Drivers installation	60

Chapter 1 Introduction

1.1 Introduction

This paper describes the design of a Wireless Sensing Triple Axis Reference design (ZSTAR), a demo for wireless demonstration of the 3-axis accelerometer MMA7260Q sensors from Freescale.

The reference design will enable you to see how Freescale's accelerometers can add additional functionality to applications in various industries. The accelerometer measurements can be grouped into 6 sensing functions - Fall, Tilt, Motion, Positioning, Shock and Vibration - for multifunctional applications.

The RD3152MMA7260Q development tool offers robust wireless communication using the powerful, easy-to-use 2.4GHz frequency MC13191 transceiver. Minor changes can be made with pin to pin compatibility allowing implementation of the MC13192 and MC13193 for ZigBee™ wireless applications.

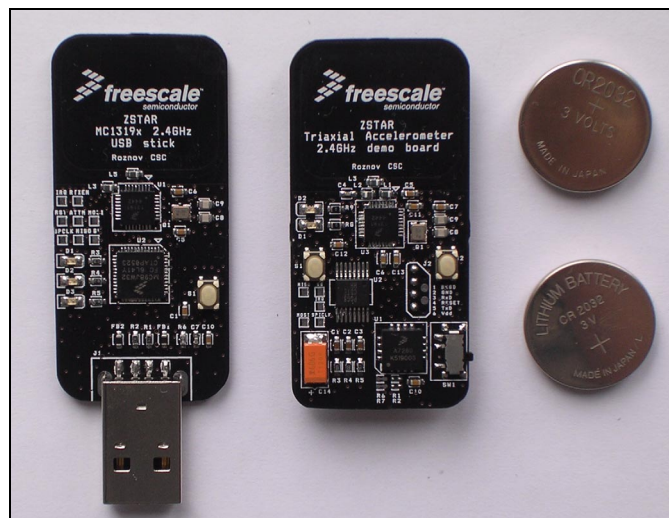


Figure 1-1 ZSTAR Demo photo (CR2032 batteries for comparison)

1.2 MMA7260Q 3-axis Accelerometer Sensor

The MMA7260Q low cost capacitive micromachined accelerometer features signal conditioning, a 1-pole low pass filter and temperature compensation, and g-Select, which allows a selection from 4 sensitivities. Zero-g offset full scale span and filter cut-offs are factory set and require no external devices. This device includes a sleep mode making it ideal for handheld battery powered electronics.

Chapter 2

Wireless Sensing Triple Axis Reference design introduction

2.1 Introduction

The Wireless Sensing Triple Axis Reference design (ZSTAR) has been designed as a wireless complement to the previous STAR (Sensing Triple Axis Reference design) RD3112MMA7260Q demo. A 2.4GHz radio-frequency (RF) link based on the low-cost MC13191 family is used for connection from the sensor to PC, allowing the visualization of key accelerometer applications.

Figure 2-1ZSTAR demo overview



The demo consists of the two boards:

- Sensor Board (or remote board) containing the MMA7260Q 3-axis accelerometer, S08 family MC9S08QG8 8-bit microcontroller and the 2.4GHz RF chip MC13191 for wireless communication.
- USB stick, again with the MC13191 RF front-end, and the HC08 family MCHC908JW32 for the USB communication.

Both sides communicate over the RF medium utilizing the freely available software stack SMAC from Freescale.

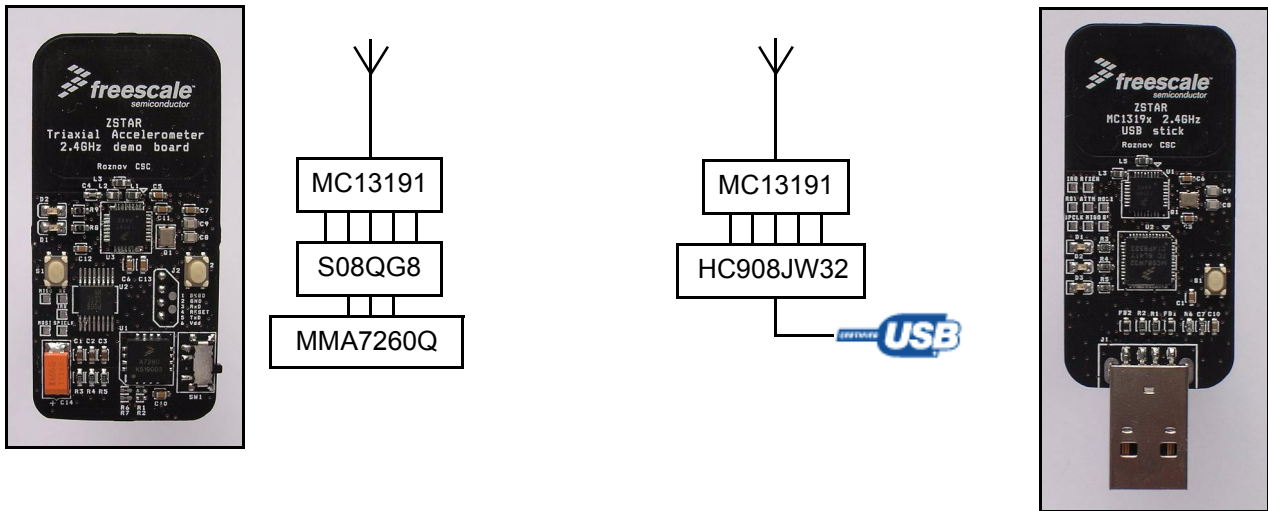


Figure 2-2ZSTAR Block diagram

2.2 Featured products

This demo consist of several Freescale products whose main features are listed below.

2.2.1 Triple Axis Accelerometer MMA7260Q

The ZSTAR board is a demonstration tool for the MMA7260Q, a 3-Axis Low-g accelerometer. The MMA7260Q has many unique features that make it an ideal solution for many consumer applications, such as freefall protection for laptops and MP3 players, tilt detection for e-compass compensation and mobile phone scrolling, motion detection for handheld games and game controllers, position sensing for g-mice, shock detection for warranty monitors, and vibration for out of balance detection.

Features such as low power, low current, and a sleep mode with a quick turn on time, allow the battery life to be extended in end applications. The 3-axis sensing in a small QFN package requires only a 6mm x 6mm board space, with a profile of 1.45mm, allowing for easy integration into many small handheld electronics.

There are several other derivatives of the MMA7260Q:

- **MMA7261Q** with a selectable 2.5g to 10g range
- **MMA6270Q** is an XY dual axis accelerometer
- **MMA6280Q** is an XZ dual axis accelerometer

All members of this sensor family are footprint (QFN package) compatible which simplifies evaluation and design of the target application.

2.2.2 Microcontroller MC9S08QG8

The MC9S08QG8 is a highly integrated member of Freescale's 8-bit family of microcontrollers based on the high-performance, low-power consumption HCS08 core. Integrating features normally found in larger, more expensive components, the MC9S08QG8 MCU includes a **background debugging system** and on-chip in-circuit emulation (ICE) with real-time bus capture, providing a single-wire debugging and

emulation interface. It also features a programmable 16-bit timer/pulse-width modulation (PWM) module (TPM), that is one of the most flexible and cost-effective of its kind.

The compact, tightly integrated MC9S08QG8 delivers a versatile combination, from wealth of Freescale peripherals and the advanced features of the HCS08 core, including **extended battery life** with a maximum performance down to 1.8V, industry-leading Flash and innovative development support. The MC9S08QG8 is an excellent solution for power and size-sensitive applications, such as wireless communications and handheld devices, small appliances, Simple Media Access Controller (SMAC)-based applications and toys.

- MC9S08QG8 Features
 - Up to 20 MHz operating frequencies at >2.1 volts and 16 MHz at <2.1 volts
 - 8 K Flash and 512 bytes RAM
 - Support for up to 32 interrupt/reset sources
 - 8-bit modulo timer module with 8-bit prescaler
 - Enhanced 8-channel, 10-bit analog-to-digital converter (ADC)
 - Analog comparator module
 - Three communication interfaces: SCI, SPI and IIC

2.2.3 MC13191 2.4 GHz ISM Band Low Power Transceiver

The MC13191 is a short range, low power, 2.4 GHz Industrial, Scientific, and Medical (ISM) band transceivers. The MC13191 contains a complete packet data modem which is compliant with the IEEE® 802.15.4 Standard PHY (Physical) layer. This allows the development of proprietary point-to-point and star networks based on the 802.15.4 packet structure and modulation format. For full 802.15.4 compliance, the MC13192 and Freescale 802.15.4 MAC software are required.

When combined with an appropriate microcontroller (MCU), the MC13191 provides a cost-effective solution for short-range data links and networks. Interface with the MCU is accomplished using a four wire serial peripheral interface (SPI) connection and an interrupt request output, which allows the use of a variety of processors. The software and processor can be scaled to fit applications ranging from simple point-to-point to star networks.

2.2.4 Microcontroller MCHC908JW32

The MCHC908JW32 is a member of the low-cost, high-performance M68HC08 Family of 8-bit microcontroller units (MCU's). All MCU's in the family use the enhanced M68HC08 central processor unit (CPU08) and are available in a variety of modules, memory sizes and types, and package types.

- MCHC908JW32 Features
 - Maximum internal bus frequency: 8-MHz at 3.5-5V operating voltage
 - Oscillators:
 - 4-MHz crystal oscillator clock input with 32MHz internal phase-lock loop
 - Internal 88-kHz RC oscillator for timebase wakeup
 - 32,768 bytes user program FLASH memory with security feature
 - 1,024 bytes of on-chip RAM
 - 29 general-purpose input/output (I/O) ports:
 - 8 keyboard interrupt with internal pull-up
 - 3 pins with direct LED drive
 - 2 pins with 10mA current drive for PS/2 connection

Wireless Sensing Triple Axis Reference design introduction

- 16-bit, 2-channel timer interface module (TIM) with selectable input capture, output compare, PWM capability on each channel, and external clock input option
- Timebase module
- PS/2 clock generator module
- Serial Peripheral Interface Module (SPI)
- Universal Serial Bus (USB) 2.0 Full Speed functions:
 - 12 Mbps data rate
 - Endpoint 0 with an 8-byte transmit buffer and an 8-byte receive buffer
 - 64 bytes endpoint buffer to share amongst endpoints 1–4

Chapter 3 Sensor Board description

3.1 Board overview

The Sensor Board utilizes a small footprint size dual-layer printed circuit board (PCB) containing all the necessary circuitry for MMA7260Q accelerometer sensing and transferring data over a radio frequency (RF).

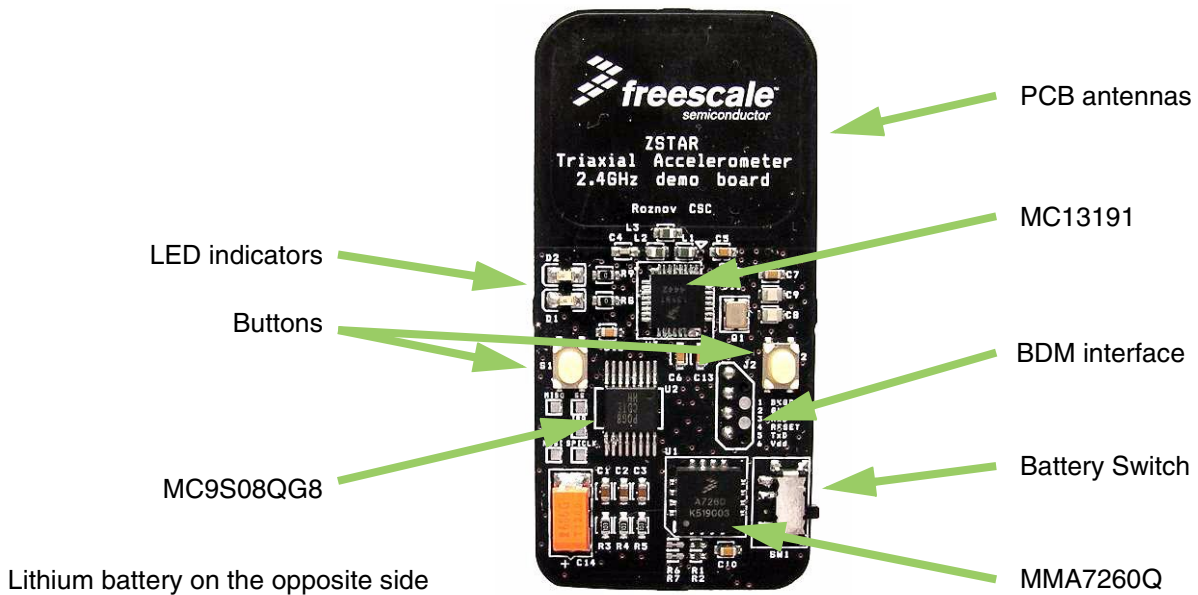


Figure 3-1 Sensor Board overview

The board is powered by a Lithium coin-sized CR2032 battery with provisions also made for the larger capacity CR2477 size. The block diagram of the board is as follows:

Sensor Board description

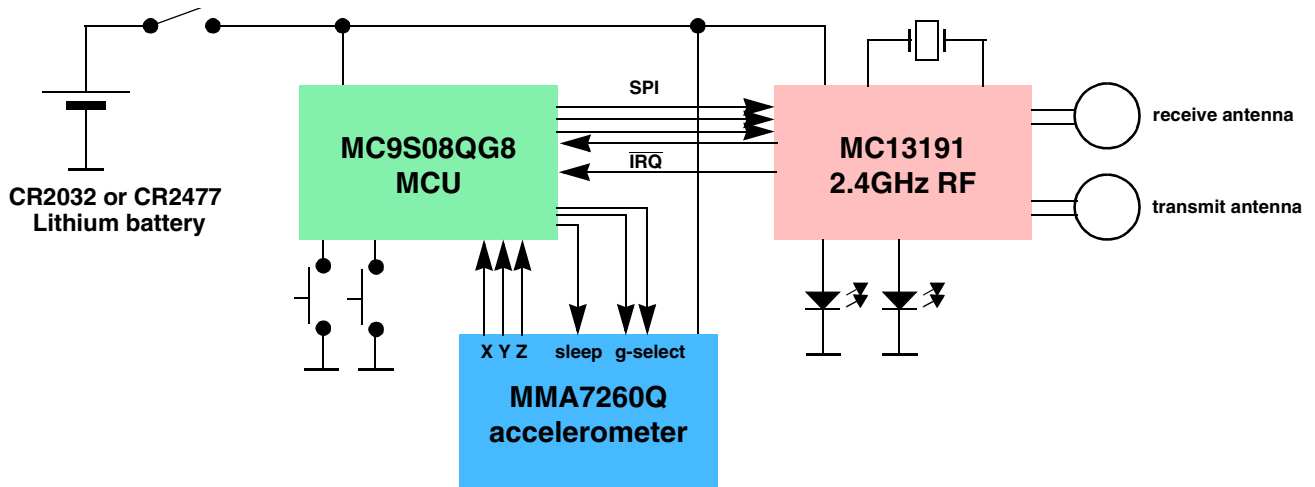


Figure 3-2 Sensor board block diagram

Figure 3-3 shows in more detail, how different software and hardware modules co-operate with each other. The main task of the Sensor board is to:

- periodically wake-up from power saving mode
- measure all three XYZ acceleration values from the sensor
- compose a data frame using simple [ZSTAR RF protocol](#)
- use [SMAC \(Simple Media Access Controller\)](#) to send this data frame over the RF link
- wait for an acknowledgment from the other end (here, the USB stick)
- go to sleep

This basic loop repeats roughly 20 times per second providing nearly a real-time response from the sensor.

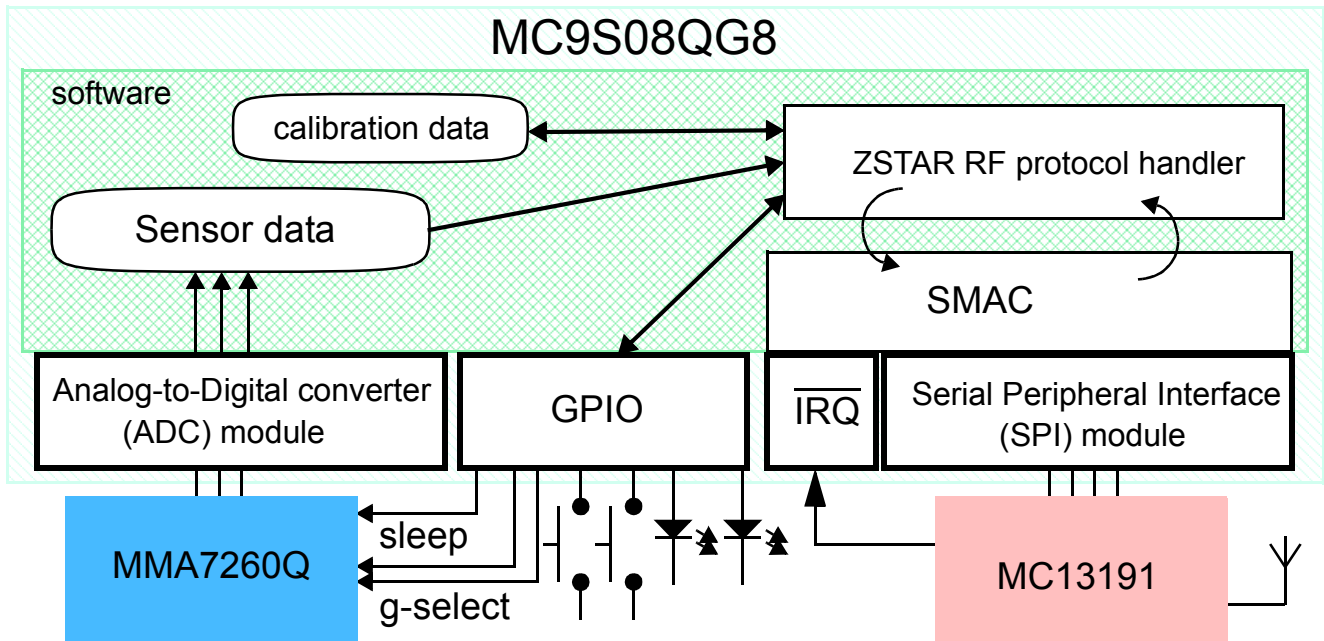


Figure 3-3ZSTAR Sensor board software overview

For the Sensor board operation, several of the MC9S08QG8's hardware modules are used: Analog to Digital Converter (ADC), Synchronous Peripheral Interface (SPI), External Interrupt Request module (IRQ) and General Purpose Input/Output (GPIO).

3.2 A/D conversion of XYZ levels

The 3-axis accelerometer sensor MMA7260Q provides three separate analog levels for the X, Y and Z axis. These outputs are ratiometric which means that the output offset voltage and sensitivity will scale linearly with applied supply voltage. This is a key feature when interfacing to a microcontroller with A/D converter reference levels tied to a power supply, because it provides system level cancellation of supply induced errors in the analog to digital conversion process.

During the analog-to-digital conversion in the microcontroller, 8-bit resolution is used. MC9S08QG8 A/D channels 0, 1 and 2 are connected to X (channel 1), Y (channel 2) and Z (channel 0) outputs of the MMA7260Q. The microcontroller's APCTL1 register enables these ADC channels for pin I/O control by the ADC module.

The ADCCFG register controls the selected mode of operation, clock source, clock divide, and configuration for low power or long sample time.

3.2.1 ADC module init:

```
APCTL1 = 0b00000111; /* 0,1,2 channels are ADC */
ADCCFG = 0b01100000; /* set prescale to 8, ADICLK=BUS, 8-bit, high speed */
```

Actual ADC measurements are done in the main software loop. There is a macro (called POWSUM) that allows configuration of measurement to take several measurements of each channel during one loop. E.g. changing POWSUM to 3, $2^3 = 8$, each channel will be measured 8 times, with POWSUM 7, each channel

Sensor Board description

is measured 128 times. By default, POWSUM is 0, for 1 measurement of each channel. Before result values are provided, the accumulated values are scaled back to the 8-bit range and inverted where necessary (may be required depending on the physical MMA7260Q device orientation relative to the Earth gravity).

Raw (i.e. not calibrated) values are actually sent, the calibration and calculation of an exact g value is done internally in the PC software.

3.2.2 ADC measurement

The following routine is used for accelerometer measurement:

```
unsigned int xx = 0;
unsigned int yy = 0;
unsigned int zz = 0;
unsigned char xxx, yyy, zzz;

#define POWSUM 0

for (i = 0; i < (1 << POWSUM); i++)
{
    ADCSC1 = 0x01; //read X channel
    while(!ADCSC1_COCO);
    xx += ADCR;

    ADCSC1 = 0x02; //read Y channel
    while(!ADCSC1_COCO);
    yy += ADCR;

    ADCSC1 = 0x00; //read Z channel
    while(!ADCSC1_COCO);
    zz += ADCR;
}

xxx = ~(unsigned char)(xx >> POWSUM);
yyy = ~(unsigned char)(yy >> POWSUM);
zzz = (unsigned char)(zz >> POWSUM);
```

3.3 Power management

A CR2032 (or CR2477) Lithium battery provides a fairly limited charge for such a realtime-like demo that demands frequent transmissions. Some sort of power management has to be implemented in order to keep the current consumption at a reasonable level.

Typically, current consumptions of Sensor board components are as follows:

- 2.4GHz transceiver MC13191
 - in Hibernate mode, 2.3 μ A
 - in Doze mode, 35 μ A
 - in Idle mode, 500 μ A
 - in Transmit mode, 30mA
 - in Receive mode, 37mA

- 8-bit microcontroller MC9S08QG8
 - in Stop mode, 750nA
 - in Wait mode, 1mA
 - in Run mode, 3.5mA
- low-g triaxial sensor MMA7260Q
 - in Sleep mode, 3 μ A
 - in Normal mode, 500 μ A

It is obvious that in a battery operated application care must be taken to ensure the lowest possible current consumption, especially when the maximum current (provided by the battery) is somehow limited. A CR2032 Lithium battery cannot provide current in the range of 40mA for long periods of time. To alleviate high current surges, an additional large capacitor has been designed - see [3.4.10 Power supply](#).

For transmission and reception using the MC13191, a specific scheme has been used to ensure the battery is not depleted or overloaded. Targeting a 20 samples per second (50ms period) transmission rate, the following scheme for one transmission/sleep cycle is used for the data transfer:

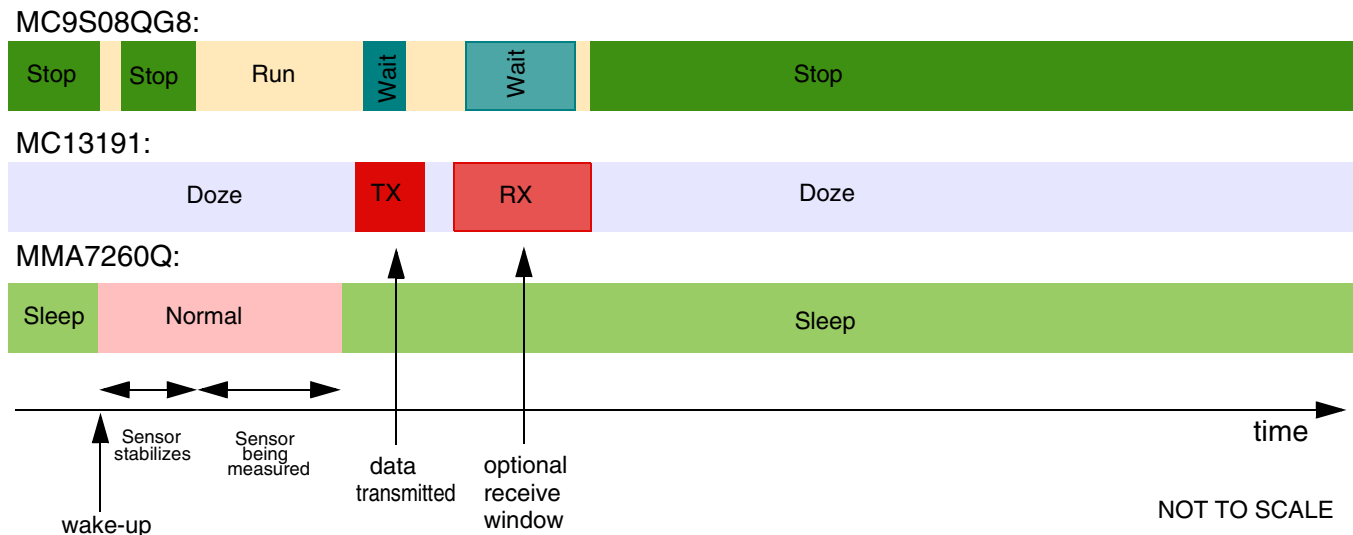


Figure 3-4 Transmission/sleep cycle details

As shown on the previous diagram, all parts of the Sensor board remain most of the time in Sleep/Doze/Stop modes, in which the total current consumption is below 10 μ A.

During each loop, once the data has been acquired from sensor, transmission over the MC13191 transceiver is initiated. The current consumption of the transmitter is ~30mA at that time, but only for a short period of time (typically ~600 μ s).

In order to keep the sensor board informed on the status of connection (for example, if the data-receiving side - USB stick - is out of range, disconnected, etc.), the reception has to be turned on after the data has been transmitted. This is not really required within each loop cycle, and in the actual implementation only on every 8th loop the receive window opens (receiver is enabled to receive the acknowledgment). More in the [5.3 ZSTAR RF protocol](#) description.

Sensor Board description

The reception window is larger to fit any incoming receive data and the current consumption is also higher during reception, so this portion of current consumption would be one of the largest if the acknowledgment was received in every loop cycle.

The “optional receive” feature allows huge power savings, still keeping the reception of acknowledgment data from the data-receiving side.

Some further savings might be incorporated by utilizing the timer-triggered transceiver events that are described in the MC13191 Reference Manual. The MC13191, for example, latches a so-called time-stamp of each received frame. The data-receiving side may read this value and trigger the acknowledgment to be sent at exactly specified time after reception (also, a start of data frame transmission can be programmed as timer-triggered). The sensor board might then narrow its own receive window to perfectly match the expected time of the acknowledgment frame. For the simplicity of code, this has not been implemented in the current version of ZSTAR firmware.

3.3.1 MC13191 power management features

MC13191 provides several power saving modes. One of them is called **Doze mode** in which the MC13191 crystal oscillator remains active. An internal timer comparator is functional too, providing a power efficient and accurately timed way of waking-up the application after a specified time.

This feature is fully utilized within the Sensor board. The microcontroller calculates the time period for which the application should be in power saving mode, then fills in the timer comparator registers in the MC13191, and the microcontroller goes into Stop mode (MC13191 into Doze mode).

Once the timer reaches the pre-programmed time (a timer compare occurs), the MC13191's IRQ signal is asserted which brings the microcontroller out of the Stop mode. There are various scaling possibilities that allow periods from a few μs up to 1073 seconds (~17 minutes) to be programmed, without intervention of the microcontroller.

3.4 ZSTAR Sensor Board hardware overview

This section describes the Sensor board in terms of the hardware design. The MC9S08QG8 microcontroller drives both the MMA7260Q sensor and the MC13191 RF transceiver.

3.4.1 Analog connections

The MMA7260Q sensor is connected to AD0, AD1, and AD2 inputs to analog-to-digital converter via RC filters formed by R3, C3, R4, C2, R5, C1. These are recommended to minimize clock noise from the switched capacitor filter circuit inside the sensor. Once the software filtering (also described in [3.2 A/D conversion of XYZ levels](#)) is employed, these RC filters may be completely omitted.

3.4.2 g-select connections

R1, R2, R6 and R7 components are made on the PCB. R1 and R2 are just footprints with no components assembled, while R6 and R7 are connected with copper trace allowing the user to disconnect (cut) these lines. By default, g-sel1 and g-sel2 MMA7260Q sensor input pins (used to select the acceleration range) are connected to pins PTB0 and PTB1 of the microcontroller. The range can be controlled by software.

If user does not want to use this feature, the g-range can be selected by placing 0R resistors in the R1 and/or R2 positions. If no resistors are assembled, MMA7260Q internal pull-down resistors will automatically select the 1.5g range (both g-sel inputs low).

Once R6 and R7 are cut, PTB0 and PTB1 (or their alternate SCI functionality of RxD1 and TxD1, or KBI or AD inputs) may be used. These signals are also routed to BDM connector, pins 3 and 5.

3.4.3 BDM (Background Debug Mode) connections

A J2 connector is a non-standard footprint primarily intended for in-factory programming and testing via “spring-needle” type of connections. The J2 connector carries all standard signals for Background Debug Mode communication so if required, one may solder wires and a standard 2x3 pins 2.54mm (100mil) pitch header for regular BDM re-programming. The pin numbering is shown on [Figure 3-5](#).

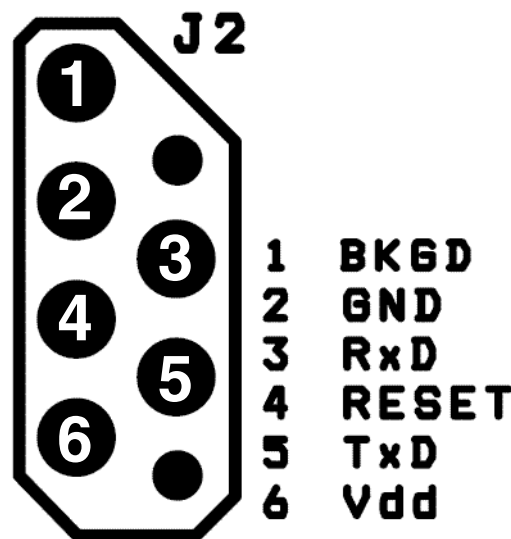
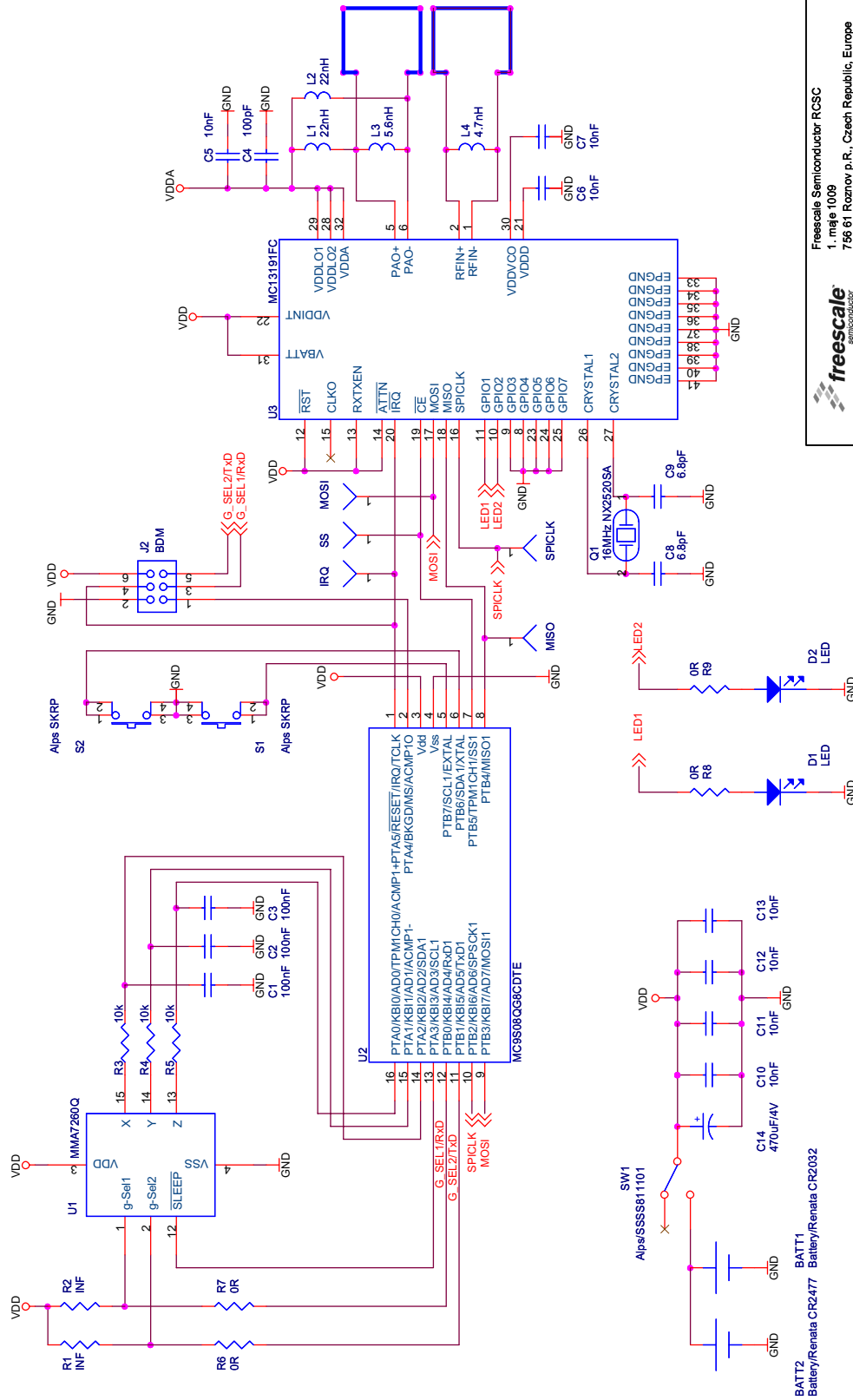


Figure 3-5 BDM connector layout

3.4.4 Sensor Board schematics



Freescale Semiconductor RCSC	
1. mge 1009	
756 61 Roznov p.R., Czech Republic, Europe	
Title Low-cost 2.4GHz Triax Board	
Author: Radomír Kozub & Pavel Lelšner	
Design Name: LOW-COST 2.4GHZ AND XYZ-ACCELEROMETER DEMO HW02300230.DSN	
Schematic Name: SCHEMATIC1	
Modify Date: Monday, January 30, 2006	Sheet 1
Copyright: Freescale	POPI Status: General Business Information

Figure 3-6 Sensor board schematics

3.4.5 Button connections

Two buttons (S1 and S2) are connected directly to pins PTB6 and PTB7. Both have internal pull-up resistors, but are not part of the Keyboard interrupt module, therefore don't allow a direct microcontroller wake-up from the Stop modes.

3.4.6 MC13191 to MC9S08QG8 microcontroller interface

In order to fit all the necessary circuitry onto a 16-pin microcontroller, the full recommended MC13191 interface has had to be reduced. The full interface includes the following connections:

- 4-wire Synchronous Peripheral Interface (SPI) connection (MISO, MOSI, SPICLK, \overline{CE})
- Interrupt Request signal (\overline{IRQ})
- Attention (\overline{ATTN}) wake-up signal
- Receive/Transmit Enable (RXTXEN) signal
- External Reset (\overline{RST}) signal

SPI and **\overline{IRQ}** are vital for the communication and configuration of the MC13191. SPI is connected to the MC9S08QG8 SPI module (pins PTB4/MISO1, PTB3/MOSI1, PTB2/SPSCK1, and GPIO pin PTB5 for \overline{CE}).

Interrupt Request (\overline{IRQ}) is connected to the microcontroller \overline{IRQ} pin sharing its alternate \overline{RESET} function when BDM communication is active.

Attention (\overline{ATTN}) signal is intended to externally wake-up the MC13191 from Doze and Hibernate modes. Since this feature is not used and exit from the Doze mode is done using a timer compare event, The \overline{ATTN} pin is not routed to the microcontroller and needs to be connected to V_{dd} .

Receive/Transmit Enable (RXTXEN) signal is used to control transitions to/from receive and transmit modes. Since this can be accomplished just by software programming and/or timer compare events, this connection to the microcontroller may also be omitted, saving an additional pin. RXTXEN is connected to V_{dd} .

External Reset (\overline{RST}) signal places the transceiver in a complete reset condition (Off mode and power down). Alternative Software reset is also possible and since Off mode (the one with the lowest possible power consumption) is not required too, \overline{RST} is connected to V_{dd} too.

3.4.7 MC13191 RF interface

The RF interface (antennas) were designed with the cost and board size in mind. Among several designs, the PCB layout antennas were in the main consideration (cost). Of several PCB antenna designs available for the 2.4GHz band (F-antenna, dipole, loop), the loop antenna has been selected mainly because of the size required on the PCB.

The MC13191 transceiver is designed with separated RF IN (receive) and PA OUT (transmit) paths. To avoid the need for an antenna switch, two separate antennas need to be used. Both ZSTAR boards (USB stick and Sensor Board) use the same antenna layout, there are two antennas on the PCB, just on the opposite sides of the PCB.

The antenna is designed as a rectangle, 20x24mm (780x940mils), made of 1.25mm (50mils) wide trace of copper. The corners are rounded with a 3.8mm (150mils) radius.

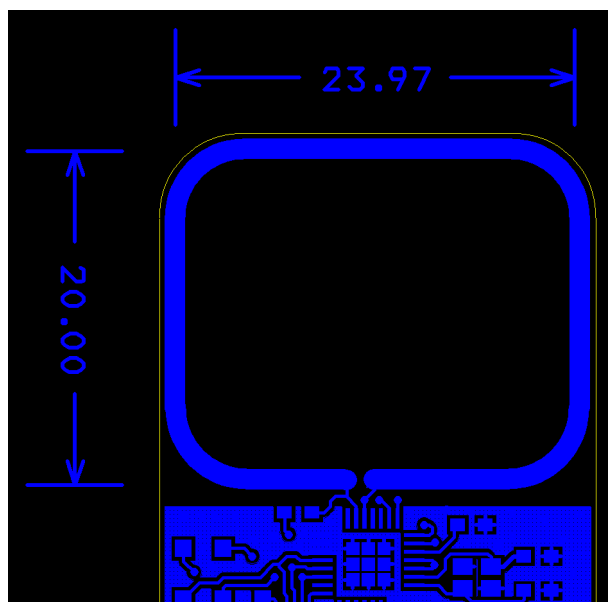


Figure 3-7ZSTAR antenna layout

The matching is provided by L3 (transmit antenna) and L4 (receive antenna) coils. L1 and L2 coils bias the transmitter output transistors to the V_{DDA} level.

The inductors used in this design are from TDK:

L3 (5.6nH) MLG1608B5N6DT

L4 (4.7nH) MLG1608B4N7ST

and L1, L2 (22nH) MLG1608B22NJT.

3.4.8 Clocking options of MC9S08QG8

Due to the availability of accurate timing provided by the MC13191 transceiver, an internal oscillator (ICG) in the MC9S08QG8 is used as the main clock source for the microcontroller. The protocol related timing is derived from MC13191 timers, the microcontroller itself is clocked from an internal oscillator, leaving the oscillator pins as GPIO. This is highly beneficial to the limited pin count microcontroller.

3.4.9 LED indicators connections

The MC13191 allows extension to the number of general I/O pins by 7 additional GPIO connections. Two of these (GPIO1 and GPIO2) are used for LED indicators. R8 and R9 are their current limiting resistors, and in the actual design orange LED's are used, with a threshold voltage around 2.0-2.5V.

The remaining unused GPIO3-GPIO7 signals are connected to ground, improving the physical PCB layout of the MC13191.

3.4.10 Power supply

The Sensor board is powered by a Lithium coin-sized battery. The primary choice was the popular CR2032, with a PCB layout provision made for the CR2477 size. This bigger battery holds roughly 4 times more charge (~1000mAh), but it is not as popular as CR2032 size.

A surface mounted SMTU series battery holder from Renata™ is placed on the underside of the PCB. The SMTU series holders provide (by mechanical construction) battery reverse protection, so no additional circuitry is required. Slide switch SW1 disconnects the battery from the application when not in use.

A large tantalum capacitor (C14, 470 μ F/4V) improves the response of the power supply to current peaks caused by reception or transmission. Coin-sized Lithium CR2032 batteries are targeted at a maximum continuous discharge current in the range of 3mA. Such a large capacitor helps to supply enough current to the MC13191 during a receive/transmit without significant V_{dd} voltage drops.

3.5 Bill of Materials

Table 3-1. Sensor board bill of materials

Item	Quantity	Reference	Part	Manufacturer	Manufacturer order code
1	1	BATT1	battery holder CR2032	Renata	SMTU 2032-1
2	3	C1,C2,C3	100nF	TDK	C1608JB1H104K
3	1	C4	100pF	TDK	C1608CH1H101J
4	7	C5,C6,C7,C10, C11,C12,C13	10nF	TDK	C1608CH1E103J
5	2	C8,C9	6.8pF	TDK	C1608CH1H070D
6	1	L3	5.6nH	TDK	MLG1608B5N6DT
7	1	L4	4.7nH	TDK	MLG1608B4N7ST
8	2	D1,D2	Kingbright KP-1608SEC	Kingbright	KP-1608SEC
9	1	J2	BDM + serial	N/A	
10	2	L1,L2	22nH	TDK	MLG1608B22NJT
11	1	Q1	16MHz NX2520SA	NDK	NX2520SA 16MHz EXS00A-02940 Specification n° EXS10B-07228
12	2	R1,R2	N/A	N/A	
13	3	R3,R4,R5	10k	resistor 0603 package	
14	2	R6,R7	N/A	N/A	
15	2	R8,R9	0R	resistor 0603 package	
16	1	SW1	slide switch Alps/SSSS811101	Alps	SSSS811101 (or SKRSPACE010 or SKRPABE010)
17	2	S1,S2	switch SKRP	Alps	SKRPADE010 (or SKRSPACE010 or SKRPABE010)
18	1	U1	MMA7260Q	Freescale	MMA7260Q (MMA7260QR2 for tape and reel)
19	1	U2	MC9S08QG8CDTE	Freescale	MC9S08QG8CDTE
20	1	U3	MC13191FC	Freescale	MC13191FC (MC13191FCR2 for tape and reel)
21	1	C14	470uF/4V	Vishay	594D477X9004C2T

Chapter 4 USB stick board description

4.1 Board overview

The USB stick board utilizes the same small footprint as Sensor Board is also a dual-layer printed circuit board (PCB). It contains the minimalistic design of the MC13191 RF transceiver connected through an 8-bit MCHC908JW32 microcontroller to the USB. It's main task is to receive data from the Sensor Board and transfer it to the PC over the USB link.

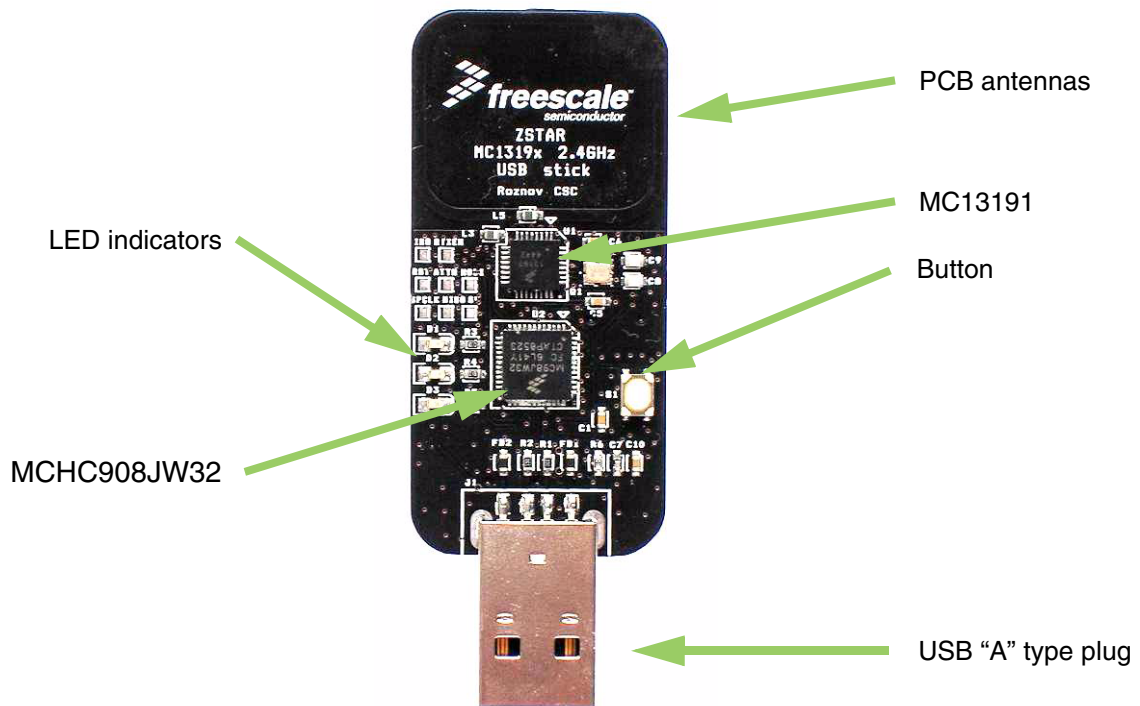


Figure 4-1 USB stick board overview

The USB stick board is powered from the USB. The block diagram of the board is as follows:

USB stick board description

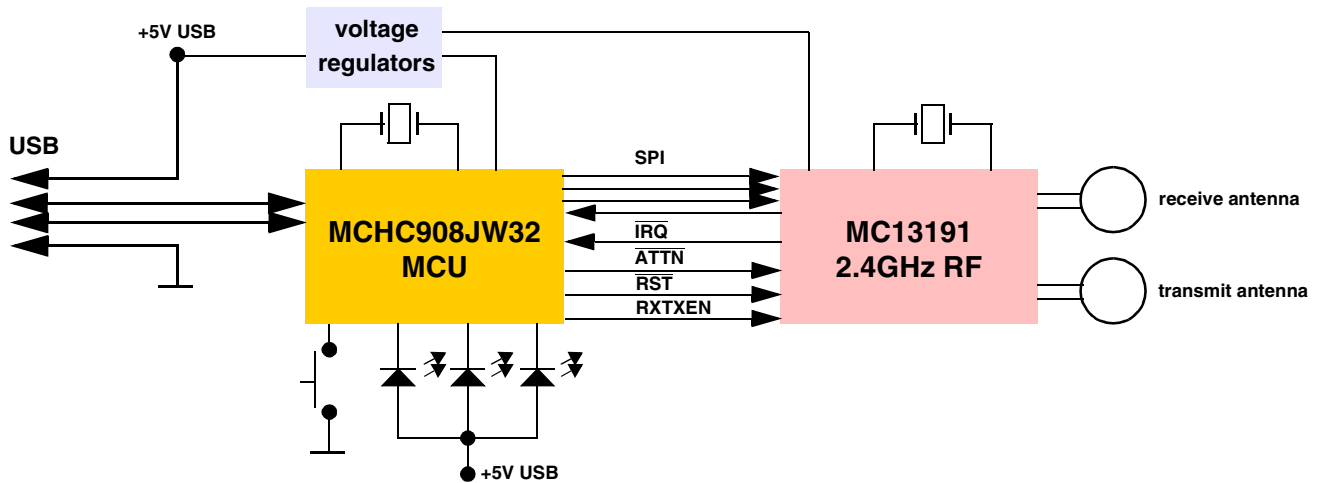


Figure 4-2 USB stick board block diagram

Figure 4-3 shows, in more detail, how different software and hardware modules co-operate with each other. There are two main tasks of the USB stick board:

- receive the data from the MC13191 transceiver and store it in RAM buffer
- handle the USB module communication, decode and provide the data from the RAM buffer

These two are somewhat independent and the only common point between them is the accelerometer and button data buffer in RAM. The RF software communicates with the Sensor Board and retrieves the latest accelerometer data. This is stored in RAM and can be independently read by the PC application via the USB link. The protocol employed on the PC side is just a subset of the simple STAR protocol used in the original RD3112MMA7260Q demo. The protocol is described in chapter [5.4 STAR protocol and ZSTAR extensions \(over USB\)](#).

MCHC908JW32

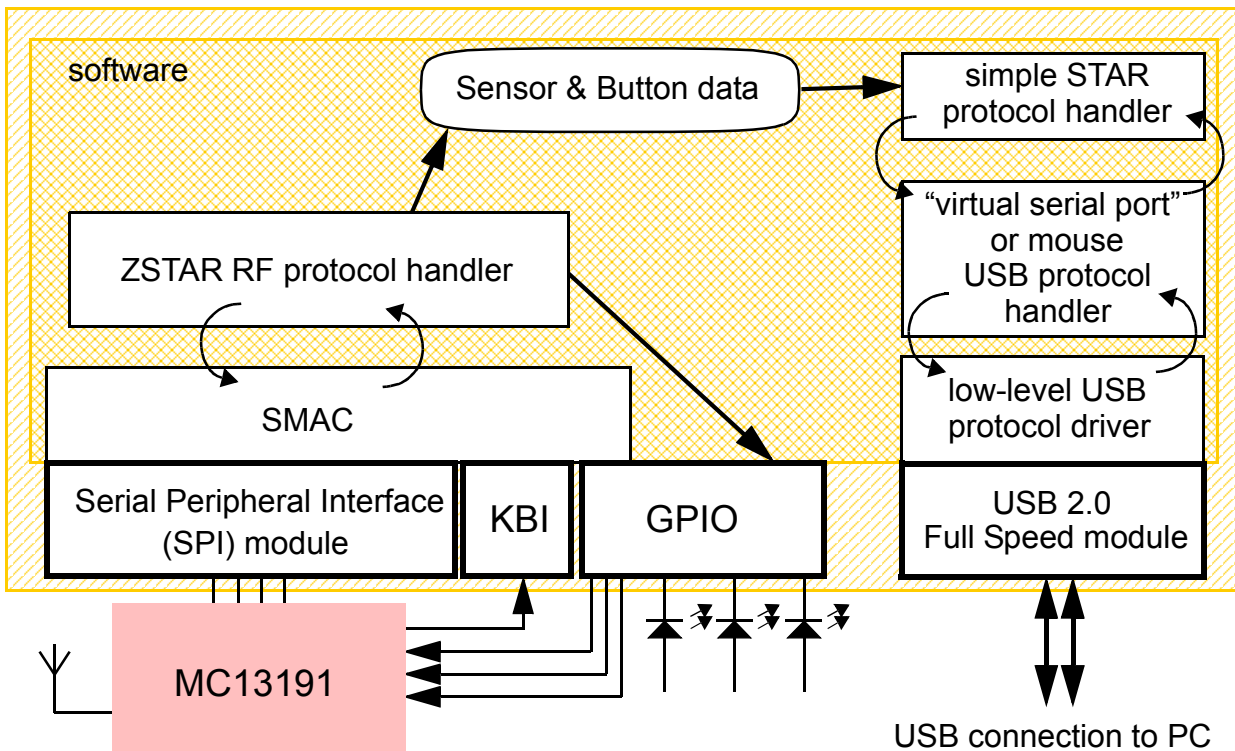


Figure 4-3ZSTAR USB stick board software overview

For the USB stick board operation, several MCHC908JW32 hardware modules are used: USB 2.0 Full-speed module (USB), Synchronous Peripheral Interface (SPI), Keyboard Interrupt module (KBI) and a General Purpose Input/Output (GPIO).

4.2 ZSTAR USB stick Board hardware overview

This section describes the USB stick board in terms of the hardware design. The MCHC908JW32 microcontroller drives the MC13191 RF transceiver and communicates over USB with PC.

4.2.1 USB connections

Two USB communication lines are connected directly via R1 to PTE2/D+ and R2 to PTE3/D- microcontroller pins. There, the R1 and R2 resistors define the output impedance of both drivers (Z_{DRV} as per chapter 7 of the USB 2.0 specifications).

Terminating the D+ line with a 1.5k Ω pull-up resistor (required for Full-speed signalling) is internal in the MCHC908JW32.

A USB “A” type SMT Plug is designed at the edge of the USB stick board allowing the stick to be connected directly into a USB hub without the need for a cable.

4.2.2 Power supply

The USB stick board is a Low-power Bus-powered Function (as defined in chapter 7.2.1.3 of USB 2.0 specifications). This means that a maximum of one unit load (100mA) may be drawn by the USB Stick board. Ferrite beads are included on the VBUS and GND USB connections to minimize EMI. The recommended type is a **GLF1608T100M** or similar from TDK.

V_{BUS} voltage is defined as a minimum 4.4V and a maximum 5.25V on a Low-power Bus-powered Function.

The MC13191 RF transceiver requires a maximum power supply voltage of 3.4V and the MCHC908JW32 microcontroller could not guarantee an internal 3.3V regulator working at such a low power supply. Therefore, two separate voltage regulators need to be implemented, and in addition, the voltage levels have to be close enough to avoid the need for level shifters (for the [MC13191 to MCHC908JW32 microcontroller interface](#)).

4.2.2.1 Fixed voltage regulators

Two voltage levels (3.3V for MC13191 and 3.6V for MCHC908JW32) were selected. For these levels, a low-cost, small footprint fixed regulator exists. The **NCP502/A** series regulators from ON SemiconductorTM were successfully implemented.

The **NCP502/A** series voltage regulator is an 80mA CMOS fixed linear regulator designed primarily for handheld communication equipment and portable battery powered applications which require a low quiescent current.

Each device contains a voltage reference unit, an error amplifier, a PMOS power transistor, resistors for setting the output voltage, current limit, and temperature limit protection circuits. The **NCP502/A** has been designed to be used with low cost ceramic capacitors. The device is housed in a micro-miniature SC70-5 surface mount package. Standard voltage versions are 1.5 V, 1.8 V, 2.5 V, 2.7 V, 2.8 V, 3.0 V, 3.3 V, 3.5 V, 3.6 V and 5.0 V. Other voltages are available in 100 mV steps.

Typically, a low-cost 1 μ F ceramic capacitor is recommended for input and output decoupling. 0603-sized SMD TDK capacitor **C1608X5R1A105K** was used.

Enable Operation - Enable pin of 3.3V regulator (for MC13191) is connected to PTC3 pin of the microcontroller. This way, the microcontroller may completely turn off the RF part of the application to minimize power consumption in USB suspend modes.

Alternatively, power down of the MC13191 RF transceiver may be done by forcing it into Off mode by pulling the $\overline{\text{RST}}$ pin low.

4.2.3 MC13191 to MCHC908JW32 microcontroller interface

On the USB stick board the full recommended MC13191 interface has been used. This includes the following connections:

- 4-wire Synchronous Peripheral Interface (SPI) connection (MISO, MOSI, SPICLK, $\overline{\text{CE}}$)
- Interrupt Request signal ($\overline{\text{IRQ}}$)
- Attention ($\overline{\text{ATTN}}$) wake-up signal
- Receive/Transmit Enable (RXTXEN) signal
- External Reset ($\overline{\text{RST}}$) signal

The SPI connection is connected to the MCHC908JW32 SPI module signals (MISO, MOSI, SPCLK, $\overline{\text{SS}}$).

The $\overline{\text{IRQ}}$ signal is routed to the PTA3/ $\overline{\text{KBA3}}$ Keyboard interrupt module pin instead of the MCHC908JW32 $\overline{\text{IRQ}}$ pin, which is left for the [MON08 interface](#) and the [Button connection](#). The reason for re-routing this signal is that V_{TST} (1.5x V_{DD} , up to 8V) is applied to the microcontroller's $\overline{\text{IRQ}}$ during programming, therefore some additional jumper configuration would be required to disconnect this voltage from the MC13191. Here, the $\overline{\text{IRQ}}$'s MON08 function is only shared with the button under the condition that the button is not pressed during programming.

The remaining three signals ($\overline{\text{ATTN}}$, RXTXEN and $\overline{\text{RST}}$) are connected to GPIO signals of port D (PTD0, PTD2 and PTD1).

4.2.4 Oscillator and clocking options

The MCHC908JW32 microcontroller requires a stable clock, mainly for the Full-speed USB module operations. USB specifications define an overall 2500ppm (0.25%) accuracy. Basically, any generic 4MHz crystal is sufficient for such accuracy. The main issue with 4MHz crystals are their physical size. Due to the nature of crystal resonating elements, the 4MHz crystals are simply far too big for the USB stick in the ZSTAR demo.

Another option is a SAW resonator (e.g. **CERALOCK™** series from Murata). These are usually sorted and selected by the manufacturer to fit the USB 2.0 Full-speed accuracy required. Today, only 6, 12, 24 and 48MHz versions are available from Murata. A 6MHz version (manufacturer order code **CSTCR6M00G15**) has been used in the USB stick design, although the 6MHz frequency is outside the MCHC908JW32 microcontroller specifications.

Provision is also made on the PCB (Q3 component) for an Epson **SG-310** series (or compatible) Crystal Oscillator (active output). Here, a 4MHz version oscillator is contained in a small 3.2x2.5mm package.

4.2.5 LED indicators connections

The MCHC908JW32 microcontroller allows a direct drive of LED's on its three pins. PTB0, PTB1 and PTB5 are high-current open-drain outputs, so the LED's D1, D2 and D3 are connected to these high-current outputs.

4.2.6 Button connection

One button is implemented on the USB stick board. It is connected to the $\overline{\text{IRQ}}$ microcontroller pin that has internal pull-up and allows an easy software interrupt.

4.2.7 MON08 interface

For MCHC908JW32 in-circuit programming, a MON08 interface is required. Several pins must be connected to specific voltage levels in order for the MCHC908JW32 to enter the Monitor mode. The details are described in the MCHC908JW32 datasheet, Chapter 7 Monitor ROM (MON).

To minimize the number of MON08 connections, several pins are hardwired to specific voltage levels directly on the USB stick board. Namely, PTA1 to V_{dd} , PTA2 and PTC1 to GND.

Pins PTA0, $\overline{\text{RST}}$, $\overline{\text{IRQ}}$ and OSC1, together with the power supply lines, are routed to PCB pads MON08 connector (J3).

There is no standard physical connector to be soldered onto the J3 footprint. The J3 connector pads are used during manufacturing for the initial in-circuit programming. Further re-programming of the USB stick maybe done using an [AN2295 Bootloader](#) as described in [chapter 5.5](#).

4.2.8 Optional serial interface

For the purpose of evaluating the USB functions of the MCHC908JW32 microcontroller, a few other pins were routed to an additional PCB pads connector (J2). The two TIM timer pins are connected to J2 allowing emulation of SCI, IIC or such like serial interfaces in software. A simple example of a USB to UART converter software is a part of the [AN3153 Application note - Using the Full-Speed USB Module on the MCHC908JW32](#).

4.2.9 USB stick schematics

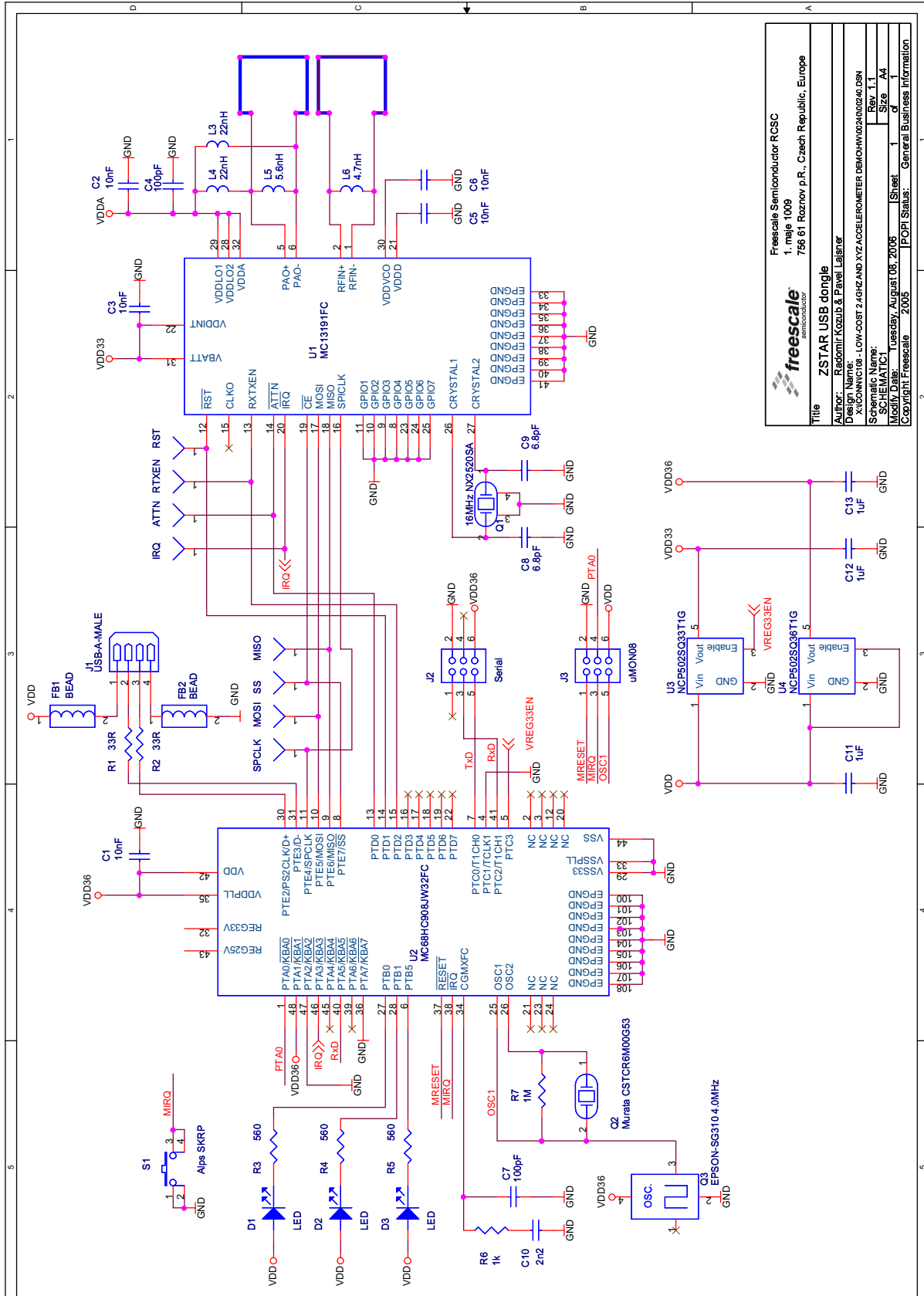


Figure 4-4USB stick schematics

4.3 Bill of Materials

Table 4-1 USB stick bill of materials

Item	Quantity	Reference	Part	Manufacturer	Manufacturer order code
1	5	C1,C2,C3,C5,C6	10nF	TDK	C1608CH1E103J
2	2	C4,C7	100pF	TDK	C1608CH1H101J
3	2	C8,C9	6.8pF	TDK	C1608CH1H070D
4	1	C10	2n2	TDK	C1608CH1H222J
5	3	C11,C12,C13	1uF	TDK	C1608JB1C105K
6	3	D1,D2,D3	Kingbright KP-1608SEC	Kingbright	KP-1608SEC
7	2	FB1,FB2	Ferrite bead	TDK	GLF1608T100M
8	1	J1	USB-A-MALE	Molex	48037-1000
9	1	J2	Serial	N/A	
10	2	L3,L4	22nH	TDK	MLG1608B22NJT
11	1	L6	4.7nH	TDK	MLG1608B4N7ST
12	1	L5	5.6nH	TDK	MLG1608B5N6DT
13	1	Q1	16MHz NX2520SA	NDK	NX2520SA 16MHz EXS00A-02940 Specification n° EXS10B-07228
14	1	Q2	Murata CSTCR6M00G15	Murata	CSTCR6M00G15
15	2	R1,R2	33R	resistor 0603 package	
16	3	R3,R4,R5	560R	resistor 0603 package	
17	1	R6	1k	resistor 0603 package	
18	1	R7	1M	resistor 0603 package	
19	1	U1	MC13191FC	Freescale	MC13191FC
20	1	U2	MCHC908JW32FC	Freescale	MCHC908JW32FC
21	1	U3	NCP502SQ33T1G	ON Semi	NCP502SQ33T1G
22	1	U4	NCP502SQ36T1G	ON Semi	NCP502SQ36T1G
23	1	S1	switch SKRP	Alps	SKRPADE010 (or SKRSPACE010 or SKRPABE010)

Chapter 5 Software Design

5.1 Introduction

This section describes the design of the ZSTAR software blocks. The software description comprises these topics:

- [SMAC \(Simple Media Access Controller\)](#) modifications description
- 'Air' [ZSTAR RF protocol](#) protocol description
- Serial [STAR protocol and ZSTAR extensions \(over USB\)](#) protocol description
- AN2295 [Bootloader](#) (over USB) implementation notes

5.2 SMAC (Simple Media Access Controller)

The SMAC is a simple ANSI C based code stack available as sample source code which can be used to develop proprietary RF transceiver applications using the MC13191.

5.2.1 SMAC Features

- Compact footprint:
 - 2K FLASH
 - 10 bytes (+ maximum packet length) RAM
 - As low as 16kHz bus clock
- Can be used to demonstrate coin cell operation for a remote control
- MC13191 compatible
- Very-low power, proprietary, bi-directional RF communication link
- ANSI C source code targeted at the HCS08 core and portable to almost any CPU core (including 4-bit)
- Low priority IRQ
- Sample application included, extremely easy to use
- Liberally commented
- Metrowerks CodeWarrior Experimental edition for support

5.2.2 Modifications of SMAC for ZSTAR demo

The development of the ZSTAR software is based on the free SMAC stack available from Freescale. The SMAC version used was 4.1a. Two sorts of modifications were made since the original version did not support the HC08 family or the MC9S08QG8 derivative of the 9S08 family. All changes are made using conditional compile options, using `ZSTARQG8` and `ZSTARJW32` definitions.

A fully detailed description of the SMAC is in the SMAC Reference Manual ([SMACRM.pdf](#)), available together with SMAC source code.

5.2.2.1 MC9S08QG8 SMAC modifications (Sensor Board)

Here the modifications of the SMAC are very minimal, since the core, peripherals and naming conventions are the same as in the MC9S08GB/GT code (originally in the SMAC 4.1a code).

The main changes are listed below:

drivers.c:

- void MC13192Wake (void) function not implemented, $\overline{\text{ATTN}}$ pin not connected to the microcontroller.
- void RTXENDeAssert(void) and void RTXENAssert(void) functions not implemented, RXTXEN pin not connected to the microcontroller.

mcu_hw_config.c:

- A set of functions
void SetGPIO(unsigned char gpio);
void ClearGPIO(unsigned char gpio);
void ToggleGPIO(unsigned char gpio);
added for the purpose of driving LED's connected to the MC13191 GPIO pins.
- void UseExternalClock(void) and void UseMcuClock(void) functions not implemented, no external clock available to the microcontroller.
- $\overline{\text{RESET}}$ pin handling in void MC13192Restart(void) and void MC13192ContReset(void) functions omitted since the $\overline{\text{RESET}}$ pin is not connected to the microcontroller.
- $\overline{\text{RESET}}$, $\overline{\text{ATTN}}$ and RXTXEN pins handling in void GPIOInit(void) and void MCUInit(void) functions omitted since these pins are not connected to the microcontroller.
- LED toggling added into void MCUInit(void) during the waiting for MC13191 to initialize.

device_header.h:

- Several SPI, TPM and SOPT definitions added at the top of the standard <mc9s08qg8.h> header file.

created port_config_ZSTARQG8.h file with target specific defines (GPIO assignments, etc.)

5.2.2.2 MCHC908JW32 SMAC modifications (USB stick)

There are several modifications of SMAC required to

1. compile for the HC08 family member MCHC908JW32
2. reflect that the MC13191 connects to the microcontroller in a slightly different way (as described in [chapter 4.2.3 MC13191 to MCHC908JW32 microcontroller interface](#)) - namely, MC13191's $\overline{\text{IRQ}}$ pin.

The 9S08 to HC908 porting required slight changes in the following files:

mcu_spi_config.c:

- void SPIInit(void) function modified to initialize the HC908 SPI module.

mcu_spi_config.h:

- SPIWaitTransferDone(), SPIClearRecieveStatReg(), SPIClearRecieveDataReg(), SPISendChar(u8Char) and SPIRead() macros changed to work with the HC908 SPI module.

Further changes are relevant to the ZSTAR JW32 platform and specific connections:

`drivers.h`:

- `CLEAR_IRQ_FLAG` macro changed to reflect KBI module serving IRQ requests from MC13191.

`mcu_hw_config.c`:

- `void UseExternalClock(void)` and `void UseMcuClock(void)` functions no implemented, no external clock available to the microcontroller.
- LED toggling added into `void MCUInit(void)` during waiting for MC13191 to initialize.
- `UINT8 IRQPinLow(void)` function returns the state of the pin PTA3/KBI3 instead of the IRQ pin.

`mcu_hw_config.h`:

- `IRQFLAG`, `IRQACK()`, `IRQInit()` and `IRQPinEnable()` macros changed to reflect KBI module serving IRQ requests from the MC13191.

`device_header.h`:

- Several KBI definitions added at the top of the standard `<mc68hc908jw32.h>` header file.

created `port_config_ZSTARJW32.h` file with target specific defines (GPIO assignments, etc.)

5.2.2.3 Generic SMAC modifications (USB stick + Sensor Board)

Several modifications of SMAC have been made in order to improve the behavior with some older MC13191 silicon versions. Namely a software time-out functions (using microcontroller's TIM/TPM timer functions) have been added into `UINT8 PDDataRequest(tTxPacket *psPacket)` and `UINT8 PLMEEnergyDetect(void)` functions in `simple_phy.c` file.

Both functions wait for the `gu8RTxMode` variable to change to `IDLE_MODE`. This variable should change in the `void interrupt IRQIsr(void)` function once the execution of a specified task (in MC13191) finishes. Under some rare circumstances, an IRQ event (and also an `IRQIsr()` interrupt) does not occur, so this software workaround has been implemented to avoid a software lock-up.

5.3 ZSTAR RF protocol

The ZSTAR demo uses very simple protocol to transfer the accelerometer, button and calibration data between the Sensor Board and the USB stick over the RF medium. The protocol is built on top of **SMAC (Simple Media Access Controller)** drivers that are available for the MC13191 transceivers family. The protocol is bidirectional allowing the set up of independent connections amongst numerous pairs of ZSTAR demos.

All data is transferred in so-called Zpackets. This protocol is primarily targeted at simple demo purposes, allowing a fast transfer of the accelerometer data in short packets with minimum overheads and with minimum battery loads (most of the receive windows eliminated, short transmit packets, etc.).

5.3.1 Zpacket format

The ZSTAR Zpacket is contained inside the MC13191 standard packet structure, which is consistent with the IEEE 802.15.4 Standard. The SMAC library transparently adds a 16 bit Packet control field (see chapter 7.2.1.1 of IEEE 802.15.4 Standard specifications) to differentiate packets from ZigBee and other standards.

The Zpacket becomes a payload data for the SMAC standard packet and contains the following fields:

- Network number
- RX strength
- Zcommand
- Zdata

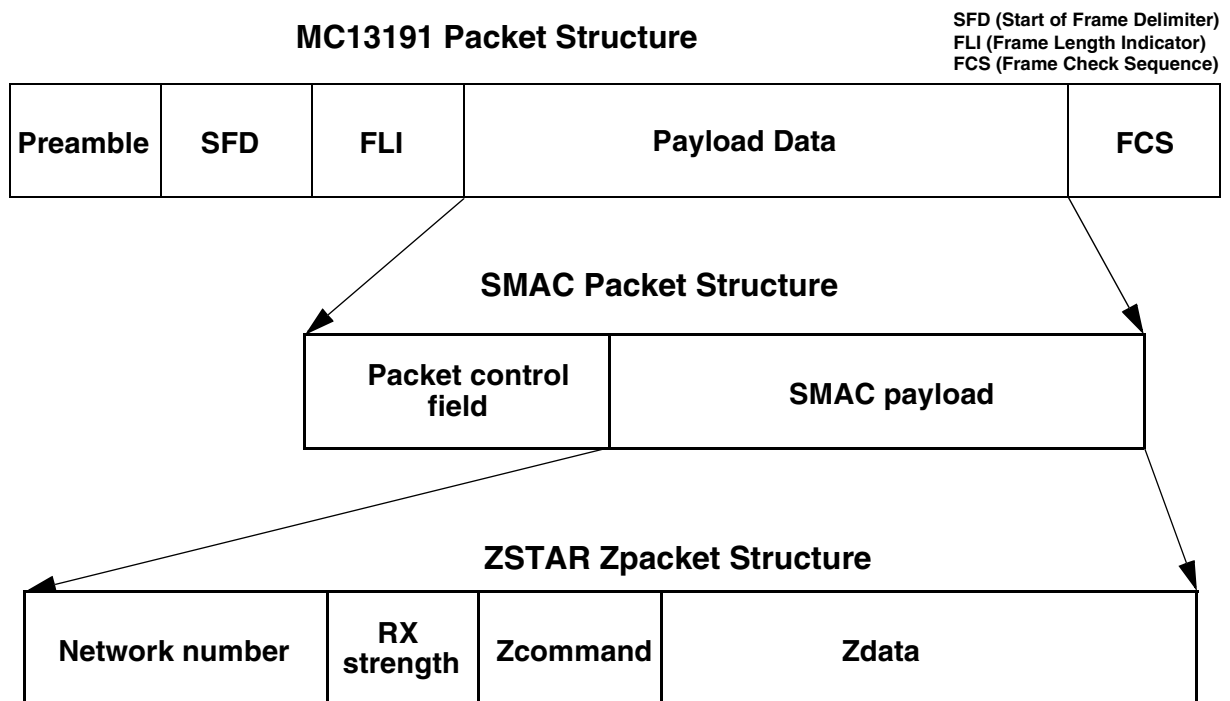


Figure 5-1Zpacket format

5.3.1.1 Network number

The network number is randomly generated at the beginning of the connection between the USB stick and the Sensor Board. It is used to determine between various connections. Packets with different Network numbers are simply ignored.

This field is 16 bits long.

5.3.1.2 RX strength

This field reports the strength of the last received packet on the other end of the connection. This value simply tells us how well the other side receives 'our packets'. This can be used by transmission power management functions to change the transmission power if the other party receives packets with enough strength.

The values reported are retrieved using the `MLMELinkQuality()` SMAC primitive.

This field is 8 bits long.

5.3.1.3 Zcommand

The ZSTAR demo protocol uses a few simple commands to establish and maintain the data flow between the Sensor Board and USB stick.

The command is carried in **Zcommand** field and is 8 bits long. The commands are defined as listed in [Table 5-1](#).

Table 5-1 ZSTAR Zcommand List

ZCommand	ZCommand code	Direction	Zdata
ZSTAR_BROADCAST	'B' (0x42)	USB stick to Sensor Board	none
ZSTAR_ACK	'A' (0x41)	USB stick to Sensor Board	none
ZSTAR_CALIB	'K' (0x4B)	USB stick to Sensor Board	calibration data to Sensor Board
ZSTAR_STATUS	'S' (0x53)	USB stick to Sensor Board	g-range selection data to Sensor Board
ZSTAR_CONNECT	'C' (0x43)	Sensor Board to USB stick	calibration data from Sensor Board
ZSTAR_DATA	'D' (0x44)	Sensor Board to USB stick	accelerometer values, button levels, g-range selection

5.3.1.4 Zdata

The **Zdata** field follows the **Zcommand** field and may be empty if the actual command doesn't require any additional data. The data format is dependent on the **Zcommand**. A detailed description is in the next chapter.

5.3.2 ZSTAR protocol Zcommand description

5.3.2.1 ZSTAR_BROADCAST

This command is sent when the USB stick tries to establish connection with the Sensor Board. The USB stick first generates a new random network number which is then ‘broadcast’ to any Sensor Board that is not yet connected to a USB stick. The USB stick transmits this command on a free channel, while the Sensor Board searches all available channels. Once a Sensor Board receives this command, it responds with a [ZSTAR_CONNECT](#).

5.3.2.2 ZSTAR_CONNECT

This command is a reply to [ZSTAR_BROADCAST](#), [ZSTAR_CALIB](#) or [ZSTAR_STATUS](#) commands sent by the USB stick. The [Zdata](#) field contains the calibration data stored in the Sensor Board, in the following format:

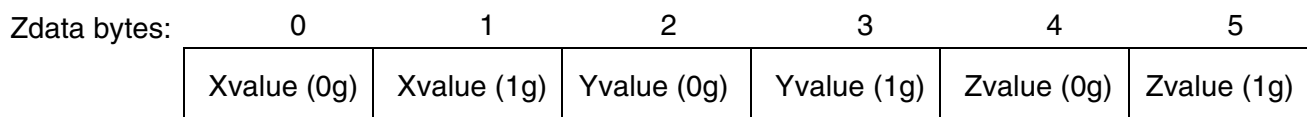


Figure 5-2 [ZSTAR_CONNECT](#) Zdata format

5.3.2.3 ZSTAR_DATA

Once the connection is established, the Sensor Board starts to periodically transmit accelerometer, button and g-range status data towards the USB stick.

The [Zdata](#) field contains 4 bytes; the actual X, Y and Z accelerometer values and a byte with status information.

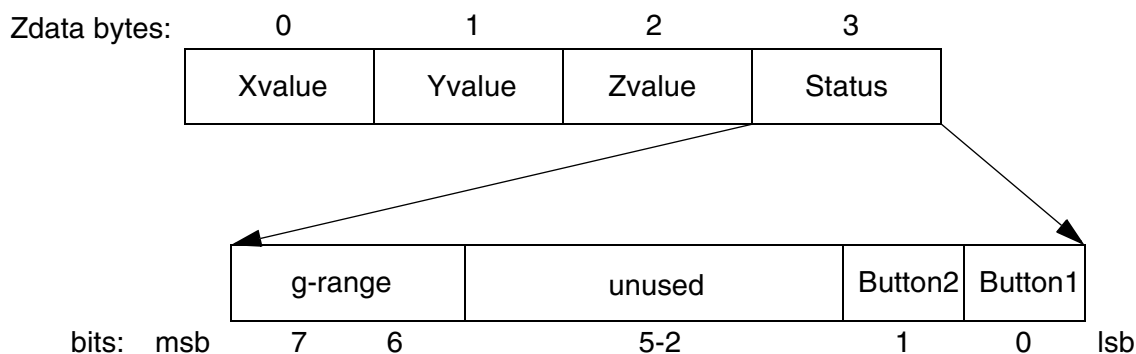


Figure 5-3 [ZSTAR_DATA](#) Zdata format

Each transmission of a [ZSTAR_DATA](#) packet is acknowledged by a [ZSTAR_ACK](#) packet from the USB stick, although the Sensor Board does not always open the reception window to receive this acknowledgement, in order to save the battery charge.

5.3.2.4 ZSTAR_ACK

This command is sent as the data acknowledgement so the Sensor Board board knows that the connection is still alive. If the receive window is opened by the Sensor Board and the **ZSTAR_ACK** has not been received, the operation (periodic transmission of a **ZSTAR_DATA** packet) continues but the Sensor Board will try to receive an acknowledgement more frequently. If the acknowledgement is not received several times, the connection is dropped and the Sensor Board will try to re-establish the connection again. The USB stick will recognize this situation once several **ZSTAR_DATA** packets have not been received.

5.3.2.5 ZSTAR_CALIB

This command carries the calibration data provided by the USB stick for the Sensor Board and is sent instead of a **ZSTAR_ACK** packet. The calibration data is intended to be stored in Flash memory of the Sensor Board. Since the Sensor Board does not receive after every **ZSTAR_DATA** packet, the USB stick keeps sending a **ZSTAR_CALIB** until the Sensor Board confirms reception using a new **ZSTAR_CONNECT** packet.

5.3.2.6 ZSTAR_STATUS

This command carries the g-range data provided by the USB stick for the Sensor Board and is sent instead of a **ZSTAR_ACK** packet. The g-range data is intended to switch the g-range of accelerometer sensor. Since the Sensor Board does not receive after every **ZSTAR_DATA** packet, the USB stick keeps sending a **ZSTAR_STATUS** until the Sensor Board confirms reception using a new **ZSTAR_CONNECT** packet.

5.4 STAR protocol and ZSTAR extensions (over USB)

The ZSTAR demo uses a subset of the original STAR demo protocol commands. This way, most of the software originally developed for the RD3112 (STAR) is also usable with the ZSTAR.

The STAR demo communicates over the RS232 serial line with a simple text-based protocol. The same protocol is used in ZSTAR for communication between the USB stick and a PC (over a virtual serial port). The PC application sees the same interface (serial port) and the same protocol as if a STAR demo was connected.

5.4.1 Communication handshake 'R' (0x52)

Initially, a handshake (commands needed to test/establish the connection between the PC and the ZSTAR demo) is conducted. This is accomplished by the PC sending an 'R' command, the ZSTAR demo responding with 'N'. In this way, the PC application sees that the demo is ready for communication. Communication is reset, and any debug or system modes are disabled.

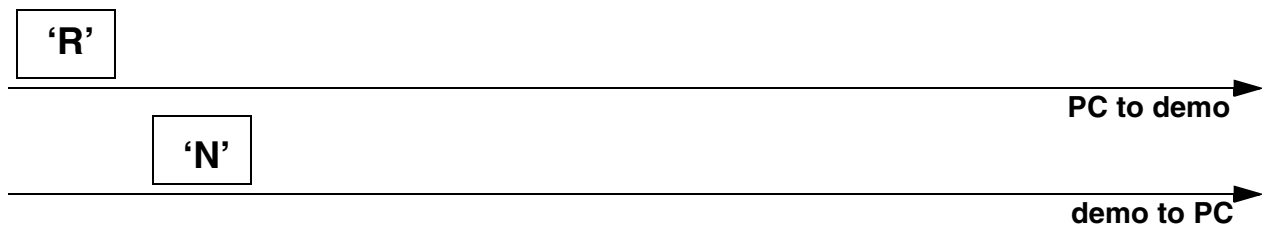


Figure 5-4 Communication handshake 'R' (0x52)

5.4.1.1 Extended Communication handshake 'r' (0x72)

To determine whether a ZSTAR or STAR demo is connected, Only the ZSTAR demo implements an Extended Handshake Communication command. Once the PC sends the 'r' command, the ZSTAR demo responds with a 'Z'.

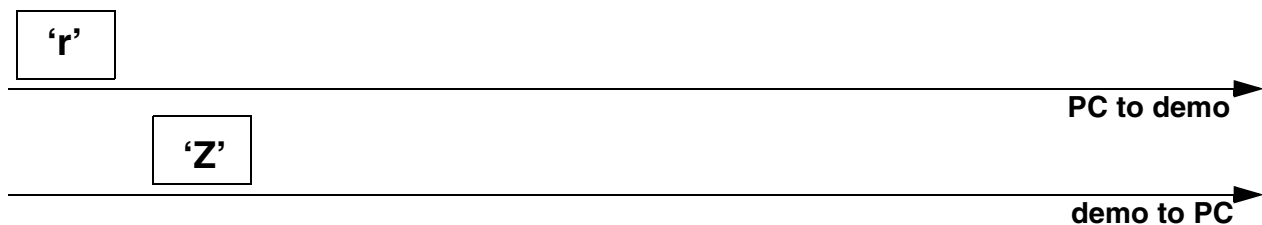


Figure 5-5 Extended Communication handshake 'r' (0x72)

5.4.2 Accelerometer data transfer 'V' (0x56)

The PC sends the Values 'V' command, the demo responds with 6 bytes in the following sequence:

'x', X-value, 'y', Y-value, 'z', Z-value, simply an 'x' character followed by the actual X-axis accelerometer binary value, then a 'y' followed by the actual Y-axis accelerometer binary value and a 'z' followed by the actual Z-axis accelerometer binary value. Since the ZSTAR demo caches the last (over the air) transmitted values, these values are immediately provided to the PC.

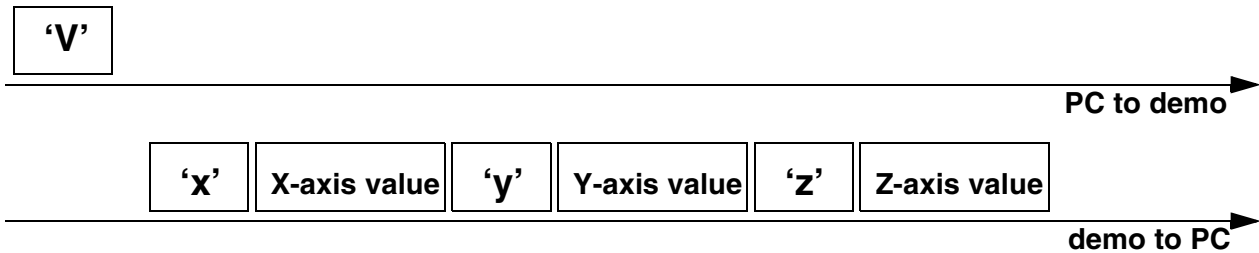


Figure 5-6 Accelerometer data transfer 'V' (0x56)

5.4.2.1 Extended Accelerometer data transfer 'v' (0x76)

The ZSTAR demo has also two buttons designed on the Sensor Board. To acquire the actual state of these buttons, the original 'V' command has been extended to a 'v' command, that provides the same information, followed by a 'b' character and a binary byte containing the actual state. The least two significant bits are used, the others are reserved. If a button is pressed, the actual bit is set to '1', and if depressed, the bit is '0'.

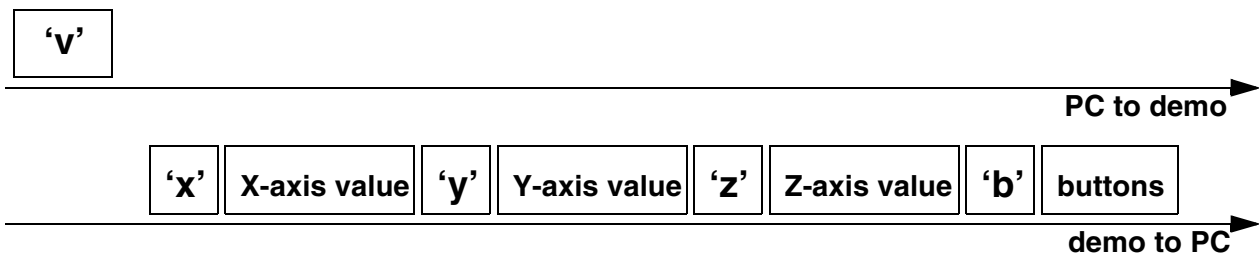


Figure 5-7 Extended Accelerometer data transfer 'v' (0x76)

5.4.3 Calibration data 'K' (0x4B)

The calibration data is the accelerometer values for specific g (acceleration) levels. The values for 0g and 1g (Earth gravity) are provided for each axis. The values are stored in the Flash memory of the Sensor Board and are transferred to the USB stick once the air connection is established (as described in chapter 5.3.2.2 ZSTAR_CONNECT). These values are stored in the USB stick for retrieval by the PC using the 'K' command.

The PC sends the Calibration data 'K' command, the demo responds with 9 bytes in the following sequence:

'X', Xval0, Xval1, 'Y', Yval0, Yval1, 'Z', Zval0, Zval1, simply an 'X' character followed by the 0g and 1g X-axis calibration accelerometer binary values, and the same for Y- and Z-axis.

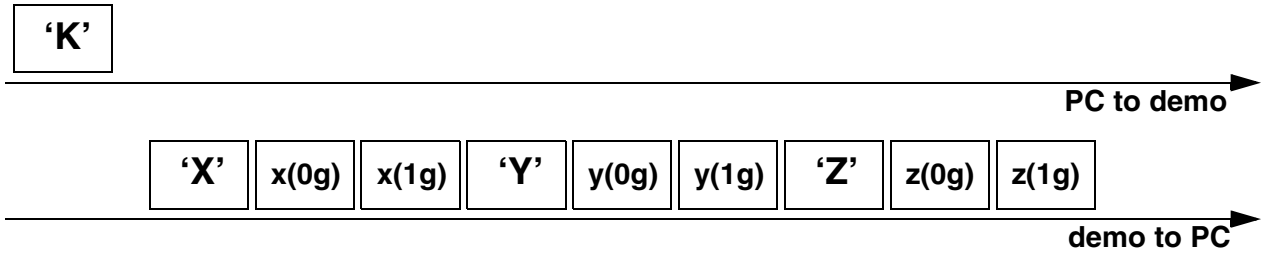


Figure 5-8 Calibration data 'K' (0x4B)

5.4.4 Calibration process 'k' (0x6B)

The calibration process is initiated by a 'k' command from the PC, followed by 6 bytes of calibration data. These are to be stored in the Flash memory of the Sensor Board being used. More in chapter [5.3.2.5 ZSTAR_CALIB](#).

The calibration data is just 6 bytes in the following sequence:

Xval0, Xval1, Yval0, Yval1, Zval0, Zval1 - 0g and 1g calibration accelerometer binary values for X-, Y- and Z-axis. No response from the demo is provided. Verification of the calibration data stored can be done using the [Calibration data 'K' \(0x4B\)](#) command.

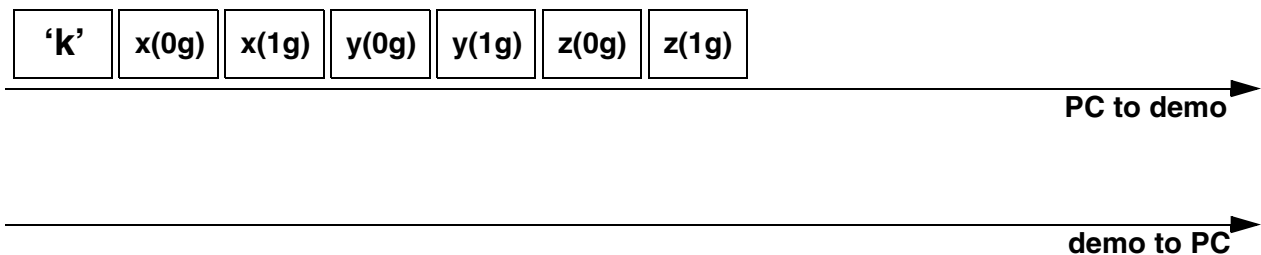


Figure 5-9 Calibration process 'k' (0x6B)

5.4.4.1 Remaining STAR demo commands

The remaining STAR commands, such as 'F', 'G', 'H', '0', '1', '2', '3' and 'E' are not implemented in the ZSTAR demo.

5.4.5 Additional ZSTAR commands

5.4.5.1 g-select reading 'G' (0x47)

The ZSTAR demo allows dynamic selection of the g-range for the accelerometer sensor (see details in chapter [3.4.2 g-select connections](#)), thus additional commands are implemented to read and change the g-range.

When the PC issues a 'G' command, the ZSTAR demo responds with the g-range actually selected. A '0', '1', '2' or '3' character is returned where '0' is for the 1.5g range, '1' for 2.0g, '2' for 4.0g and '3' for the 6.0g range. If a different sensor (e.g. MMA7261Q) is implemented on the Sensor Board, the g-ranges are 2.5g, 3.3g, 6.7g and 10g respectively.

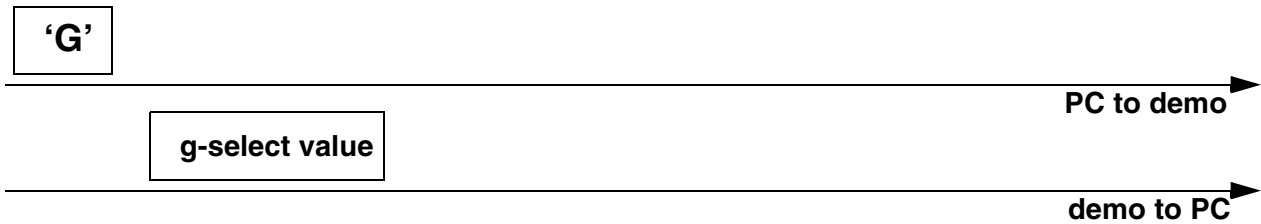


Figure 5-10 g-select reading 'G' (0x47)

5.4.5.2 g-select setting 'g' (0x67)

To select the g-range of the sensor on the ZSTAR Sensor Board, a 'g' command is issued. It needs to be followed by the required g-range ('0', '1', '2' or '3'). The USB stick board then communicates this selection to the Sensor Board over the air (see more in [5.3.2.6 ZSTAR_STATUS](#)).

No response from the demo is provided. To verify which g-range has been selected, the [g-select reading 'G' \(0x47\)](#) command can be used.

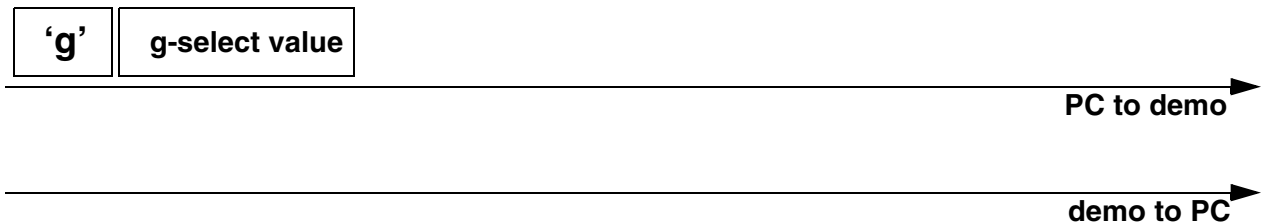


Figure 5-11 g-select setting 'g' (0x67)

5.4.5.3 Info 'I' (0x49)

The Info command 'I' has only been added to determine which version, compile date, and author of the USB stick software has been implemented. The demo returns a plain text information, and this command is usually issued over terminal (e.g. HyperTerminal) software.

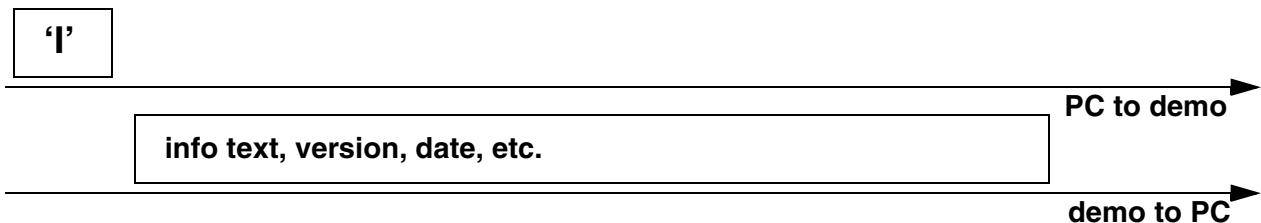


Figure 5-12 Info 'I' (0x49)

5.4.5.4 Debug on 'U' (0x55) and Debug off 'u' (0x75)

Various debug information can be observed after issuing a 'U' command, usually in terminal (e.g. HyperTerminal) software. Mainly, information on the air protocol is displayed in text, information on the detected channel energy during the surveying for a free channel is shown, channel numbers, and, once the connection is established, the network number. The received level of packets from the Sensor Board

and the reported USB stick packet level are shown, as well as command names, etc. This can be useful in determining the communication range between the USB stick and the Sensor Board.

The debug information is no longer displayed after issuing a 'u' command or [Communication handshake 'R' \(0x52\)](#).

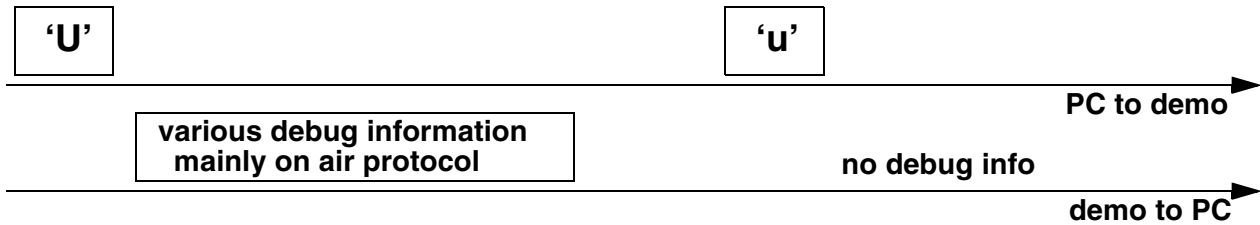


Figure 5-13 Debug on 'U' (0x55) and Debug off 'u' (0x75)

5.4.6 Further debug and test commands

5.4.6.1 Forced channel number selection

In order to allow effective testing and debugging, few additional commands have been added. If, before a connection between the USB stick and Sensor Board is established, any hexadecimal number command ('0' through '9' or 'A' through 'F') is sent, the connection will be established on this specific channel number (0 to 15). Any new channel can be selected, sending the new channel number command. The selection becomes effective during the new connection is being established.

The return to the automatic mode (where a random channel with the minimum energy is selected) can be forced only by a complete software reset (ie. removing the USB stick from the USB slot).

5.4.6.2 Semiautomatic self-calibration

For the purpose of easier semiautomatic calibration of the ZSTAR demo, additional Calibration command 'Q' (0x51) has been added. This command is usually issued over terminal (e.g. HyperTerminal) software.

A user is required to place the Sensor Board into 3 specific positions, in which the Earth gravity will induce a maximum acceleration in each of X, Y, and Z axes. Before issuing the first 'Q' command, the Sensor Board must be placed flat, ie. with Z-axis aiming toward the Earth core. For the second issue of 'Q' calibration command, the board's X-axis has to aim toward the Earth core. The board should 'stay' on its right edge. Next, the Y-axis is calibrated, with the board 'staying' on its top edge. Finally, after issuing the fourth 'Q' command, the calibration data are sent to the Sensor Board, actually using [ZSTAR_CALIB](#) command. The text help is provided during the self-calibration process.

5.5 Bootloader

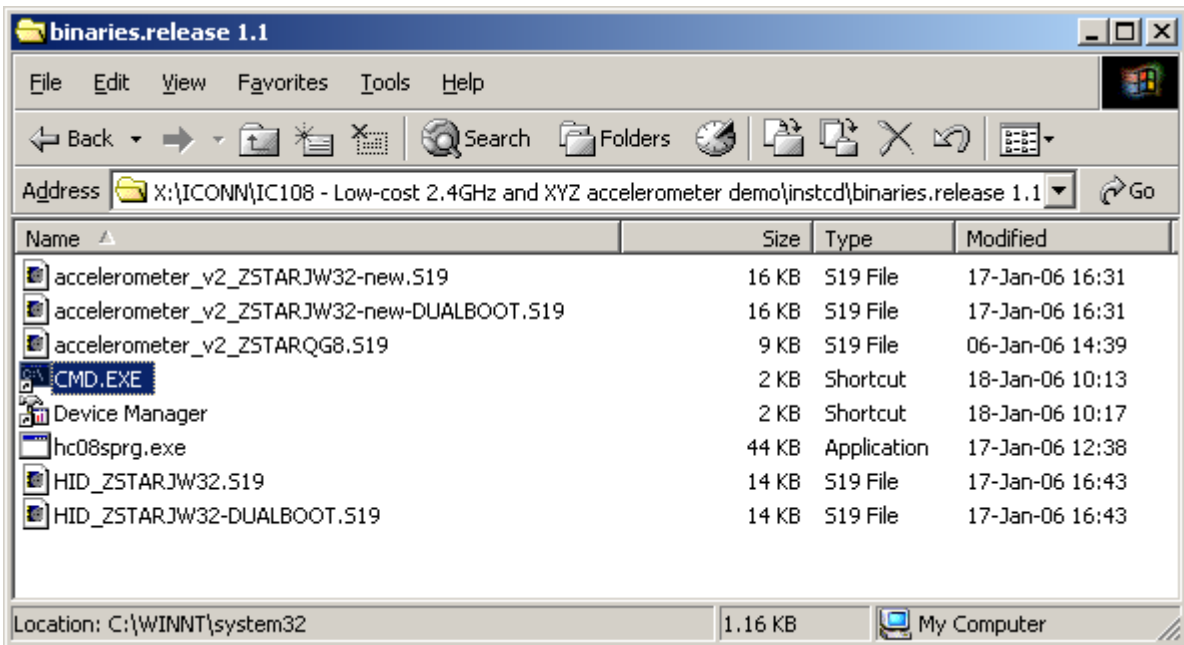
There's bootloader software implemented in MCHC908JW32 microcontroller. The bootloader is based on [AN2295 Application note - Developer's Serial Bootloader for M68HC08 and HCS08 MCUs](#) and AN2295SW accompanied software. The original AN2295 bootloader targets serial connections between the PC and applications, and since the MCHC908JW32 implements a virtual serial port application, the USB version of the AN2295 bootloader has been created to allow reprogramming of Flash memory in the USB stick.

The USB virtual serial port software is fully described in [AN3153 Application note - Using the Full-Speed USB Module on the MCHC908JW32](#); the MCHC908JW32 bootloader implements the same virtual serial port but under a different PID (the PC sees that serial port as a different application from ZSTAR).

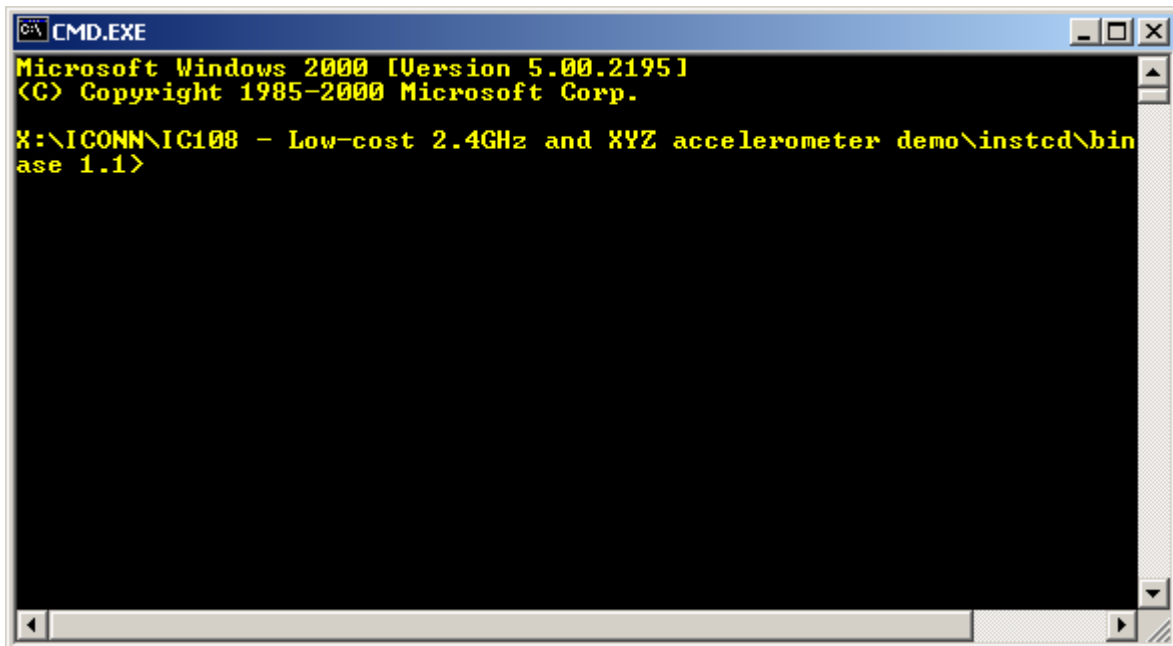
The bootloader drivers installation guide can be found in chapter [6.1.2 AN2295 Bootloader Drivers installation](#).

5.5.1 Bootloading procedure

1. Find on the installation CD the folder with binaries:

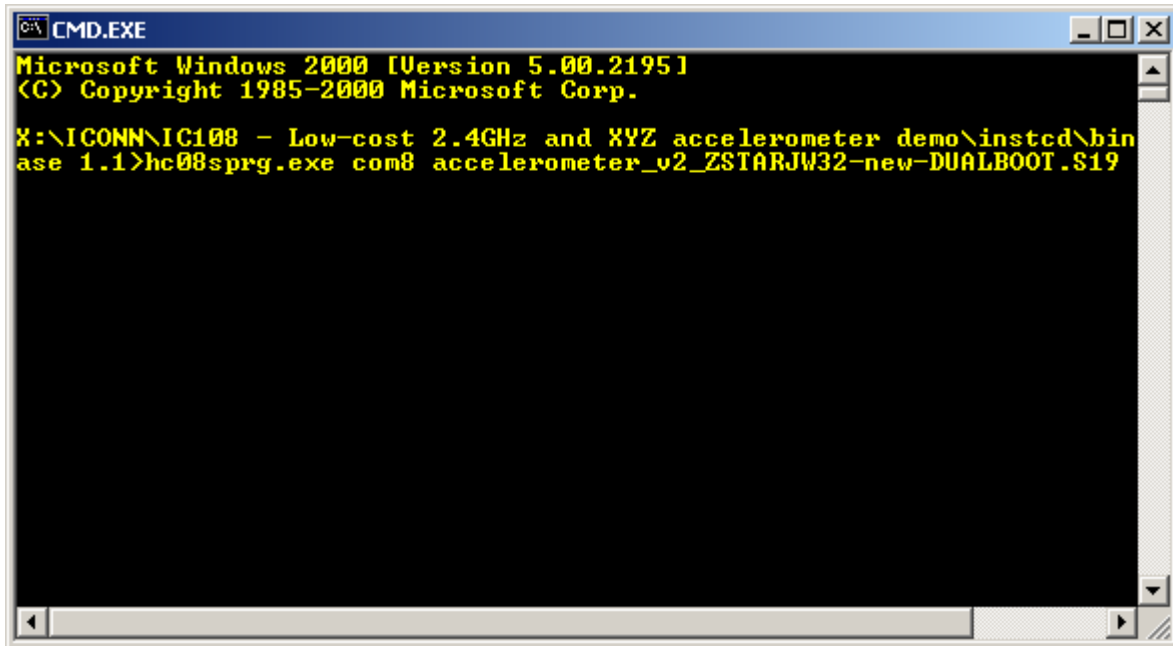


2. Start (double-click) the CMD.EXE shortcut, a command line window should appear:



- Now type: hc08sprg [bootloader com port number] [binary (S file) that you want to bootload], just like this:

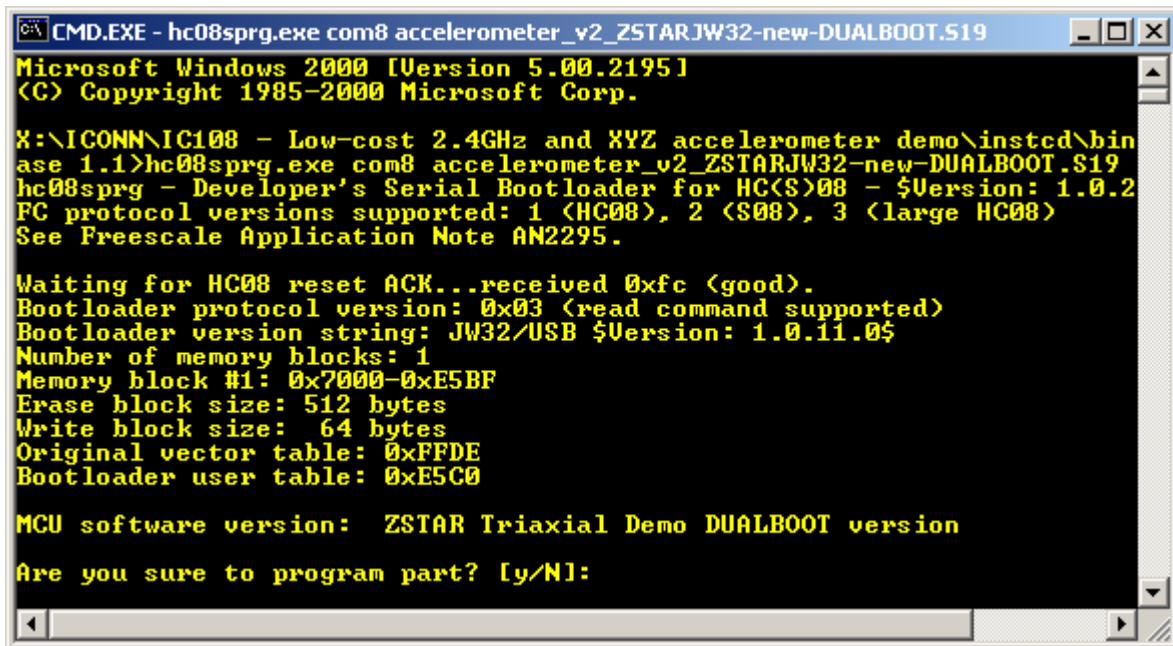
```
hc08sprg.exe com8 accelerometer_v2_ZSTARJW32-new-DUALBOOT.S19
```



```
CMD.EXE
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

X:\ICONN\IC108 - Low-cost 2.4GHz and XYZ accelerometer demo\instcd\bin
ase 1.1>hc08sprg.exe com8 accelerometer_v2_ZSTARJW32-new-DUALBOOT.S19
```

- Press ENTER and initial bootloader communication will start:



```
CMD.EXE - hc08sprg.exe com8 accelerometer_v2_ZSTARJW32-new-DUALBOOT.S19
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

X:\ICONN\IC108 - Low-cost 2.4GHz and XYZ accelerometer demo\instcd\bin
ase 1.1>hc08sprg.exe com8 accelerometer_v2_ZSTARJW32-new-DUALBOOT.S19
hc08sprg - Developer's Serial Bootloader for HC(S)08 - $Version: 1.0.2
FC protocol versions supported: 1 (HC08), 2 (S08), 3 (large HC08)
See Freescale Application Note AN2295.

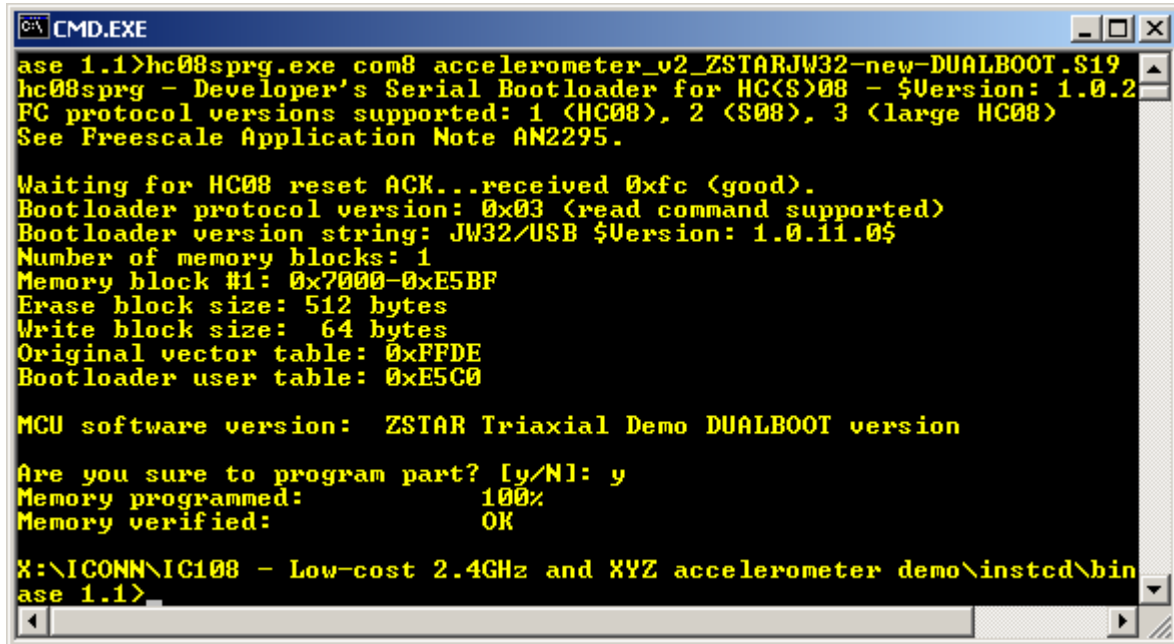
Waiting for HC08 reset ACK...received 0xfc (good).
Bootloader protocol version: 0x03 (read command supported)
Bootloader version string: JW32/USB $Version: 1.0.11.0$
Number of memory blocks: 1
Memory block #1: 0x7000-0xE5BF
Erase block size: 512 bytes
Write block size: 64 bytes
Original vector table: 0xFFDE
Bootloader user table: 0xE5C0

MCU software version: ZSTAR Triaxial Demo DUALBOOT version

Are you sure to program part? [y/N]:
```

If this screen does not appear, remove the USB stick and start from the beginning.

5. Just confirm with Y, and the binary will be loaded onto the USB stick:



```
CMD.EXE
ase 1.1>hc08sprg.exe com8 accelerometer_v2_ZSTARJW32-new-DUALBOOT.S19
hc08sprg - Developer's Serial Bootloader for HC(S)08 - $Version: 1.0.2
FC protocol versions supported: 1 (HC08), 2 (S08), 3 (large HC08)
See Freescale Application Note AN2295.

Waiting for HC08 reset ACK...received 0xfc (good).
Bootloader protocol version: 0x03 (read command supported)
Bootloader version string: JW32/USB $Version: 1.0.11.0$
Number of memory blocks: 1
Memory block #1: 0x7000-0xE5BF
Erase block size: 512 bytes
Write block size: 64 bytes
Original vector table: 0xFFDE
Bootloader user table: 0xE5C0

MCU software version: ZSTAR Triaxial Demo DUALBOOT version

Are you sure to program part? [y/N]: y
Memory programmed: 100%
Memory verified: OK

X:\ICONN\IC108 - Low-cost 2.4GHz and XYZ accelerometer demo\instcd\bin
ase 1.1>
```

The bootloader disappears (in Device Manager) and the newly loaded software starts to execute. Using this procedure the software in the USB stick can be changed anytime.

5.5.2 Dualboot guidelines

NOTE: The USB stick already comes from factory with two dualboot-aware applications pre-programmed.

USB stick and AN2295 Bootloader software provide a way of having two different software (devices) in one USB stick. In order to do this, two dualboot-aware versions of the software needs to be consecutively bootloaded onto the USB stick:

Follow the sequence of instructions in the 5.5.1 Bootloading procedure for two dualboot versions of software:

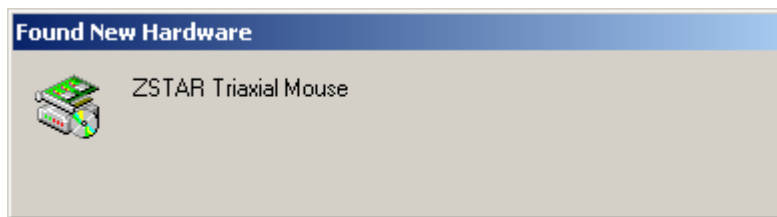
1. First bootload `accelerometer_v2_ZSTARJW32-new-DUALBOOT.S19`.

After bootloading, ZSTAR Triaxial Demo (COM1) should appear in Device Manager.

2. Next, remove the USB stick again, press the button, re-insert the stick and bootload

`HID_ZSTARJW32-DUALBOOT.S19` software in.

After bootloading, a new device (ZSTAR Triaxial Mouse) should appear:



The 'tilt' mouse will install automatically and also appear in the Device Manager:



5.5.2.1 Dualboot applications switching

Having both dualboot-aware applications programmed in the USB stick, they can be switched just by quickly pressing the button (having the USB stick inserted into the USB slot). The applications will appear and disappear accordingly.

The 'tilt' mouse application in order to work must have sensor board calibrated correctly (e.g. using RD3152MMA7260Q_SW.exe or [5.4.6.2 Semiautomatic self-calibration](#) procedure).

Chapter 6

Application Setup

6.1 ZSTAR Installation Procedure

6.1.1 USB stick installation

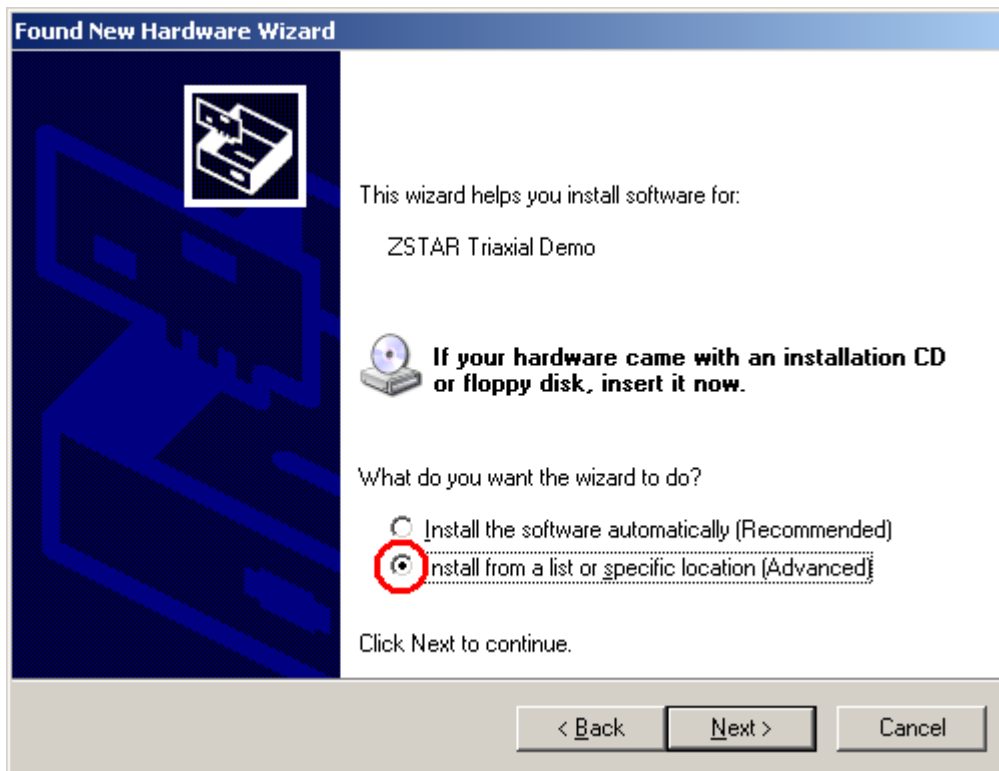
First of the all, we have to install the USB stick to your PC. Please follow the next steps.

1. Plug the USB stick into a USB slot. The 'Found New Hardware' announcement should appear:

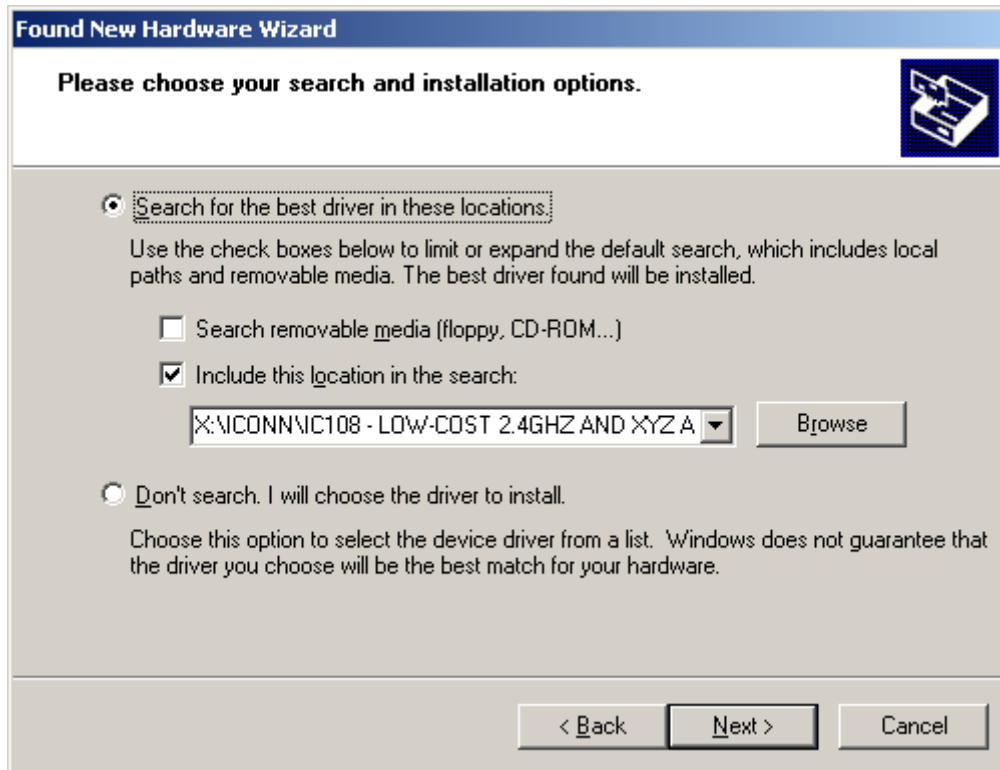


Application Setup

2. Then the installation wizard starts for new hardware. Choose “Install from a list or special location”

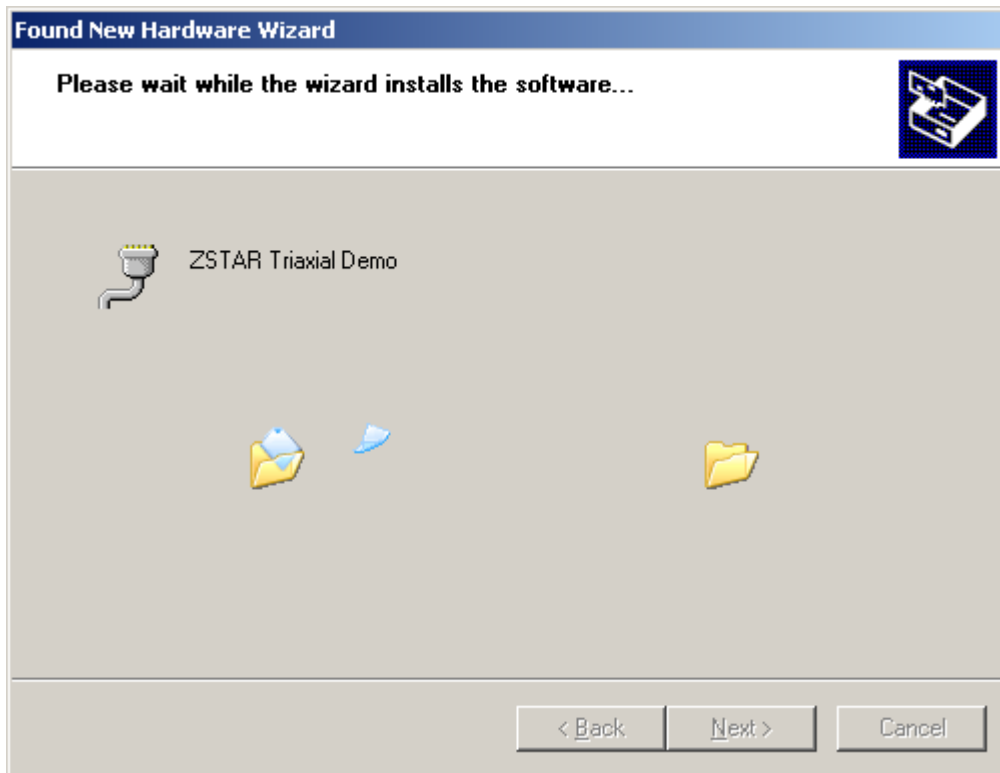


3. Point to the Installation CD as the driver path:

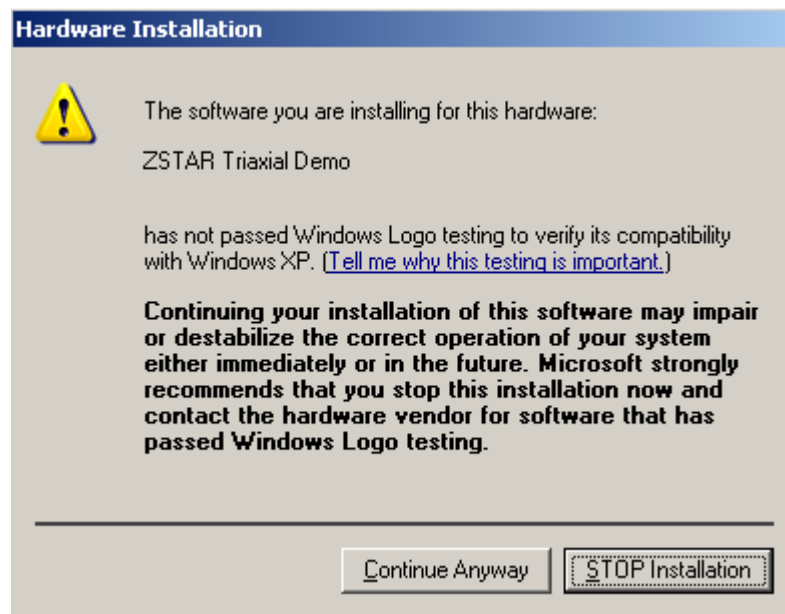


Application Setup

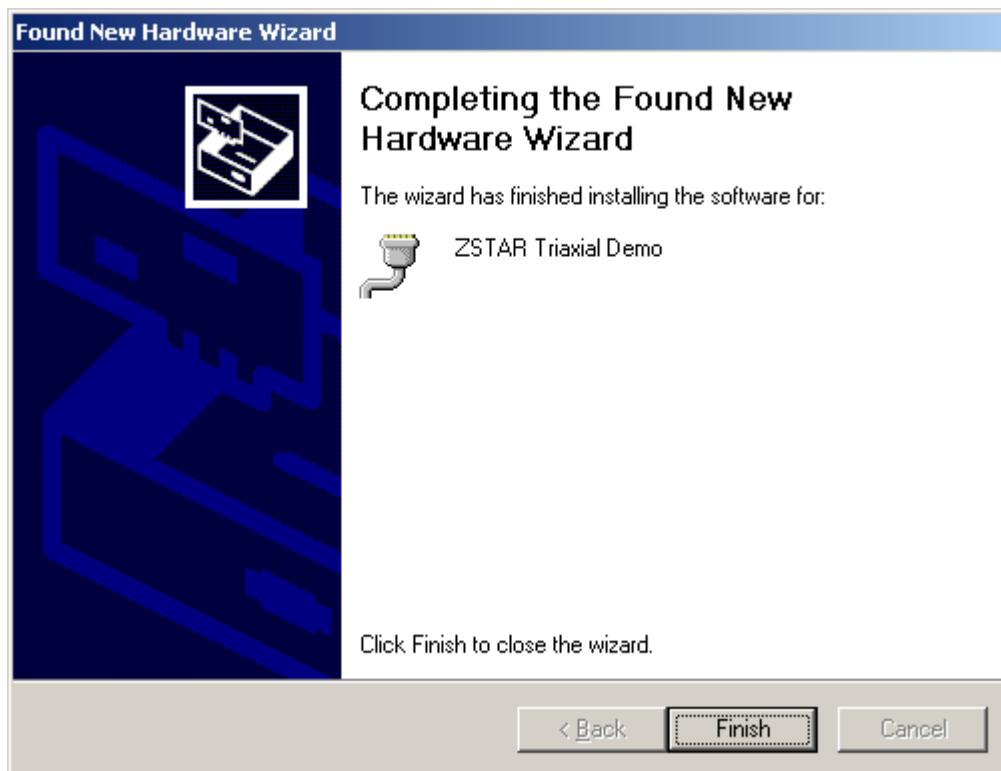
4. Installation should continue:



- If you are using Windows XP SP2, you will be asked to stop or continue installation because the drivers are not certified by Microsoft. Select the “Continue Anyway” button

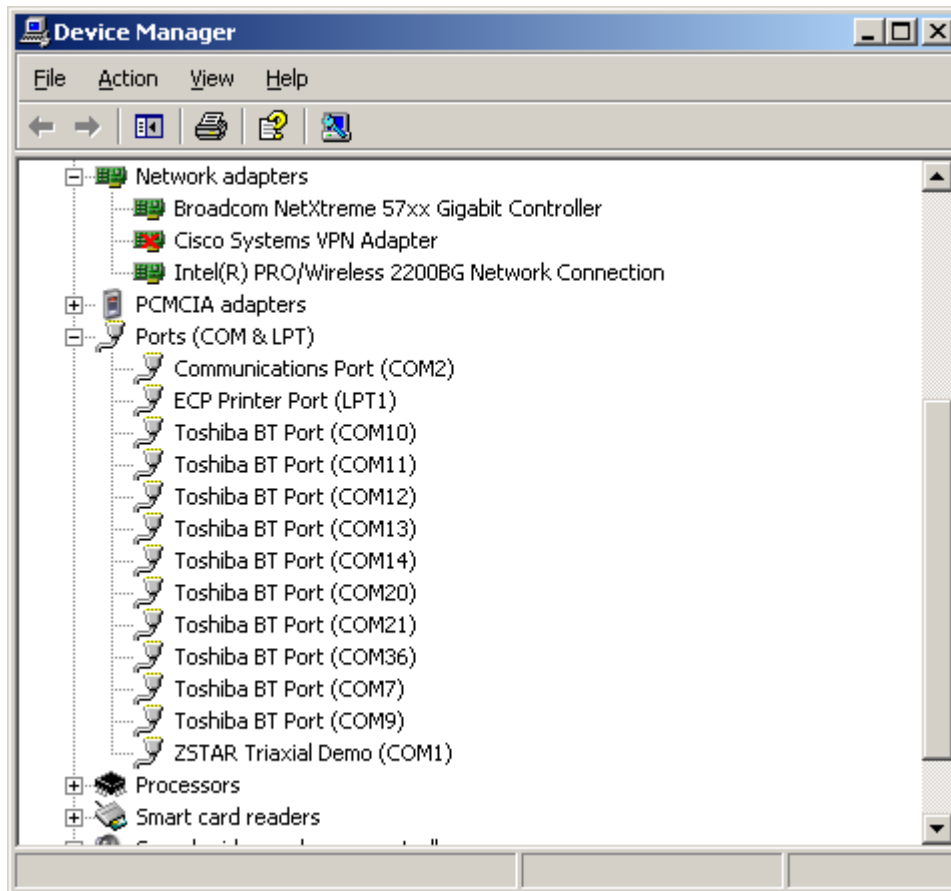


- Installation should successfully finish.



Application Setup

7. Check whether a new serial port (ZSTAR Triaxial Demo) has appeared in your Device Manager (My Computer, right click, Manage, Device Manager):



8. If required, the ZSTAR Triaxial Demo COM port maybe renumbered using the standard procedure in Windows operating system:

Right click for **Properties**, **Port Settings** tab, **Advanced** button and change the COM port number accordingly.

9. Launch the RD3152 software "RD3152MMA7260QSW.exe" and select the COM port number which you may have assigned in Device Manager.

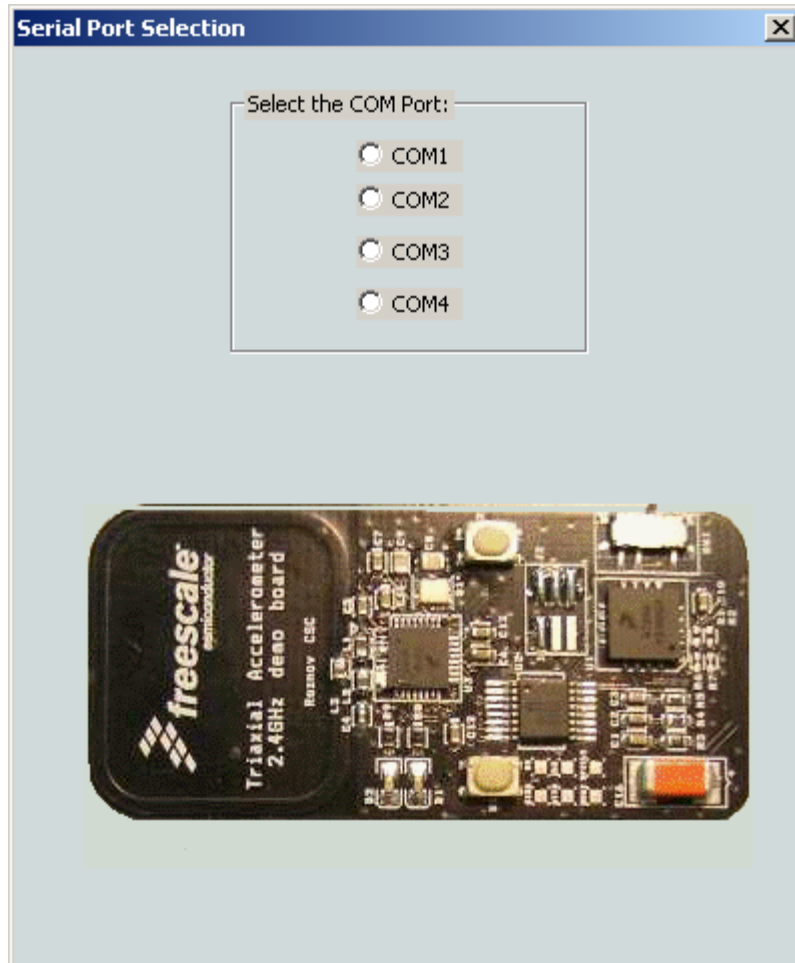


Figure 6-1

If no error message appears, the COM port is opened correctly and software communicates with the USB stick.

10. Now lets go and check data from the ZSTAR Sensor Board.

Raw data, 2D/3D screen, or Scope work should be used for this purpose. While moving (turning, shaking) with a ZSTAR Sensor Board, data from the separate axis should change accordingly.

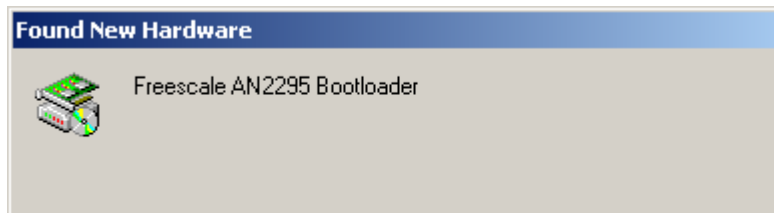
The RD3152MMA7260QSW calibration screen can be used for sensor calibration (calibration data is stored in the Sensor Board).

6.1.2 AN2295 Bootloader Drivers installation

This procedure assumes that ZSTAR Demo drivers are already installed. The drivers are also common for the bootloader (= are already present in Windows folders). If not, the procedure will be identical to the ZSTAR drivers installation.

1. Press the Button on the USB stick and insert it into a USB connector (keeping the button pressed when inserted).

The following window appears:



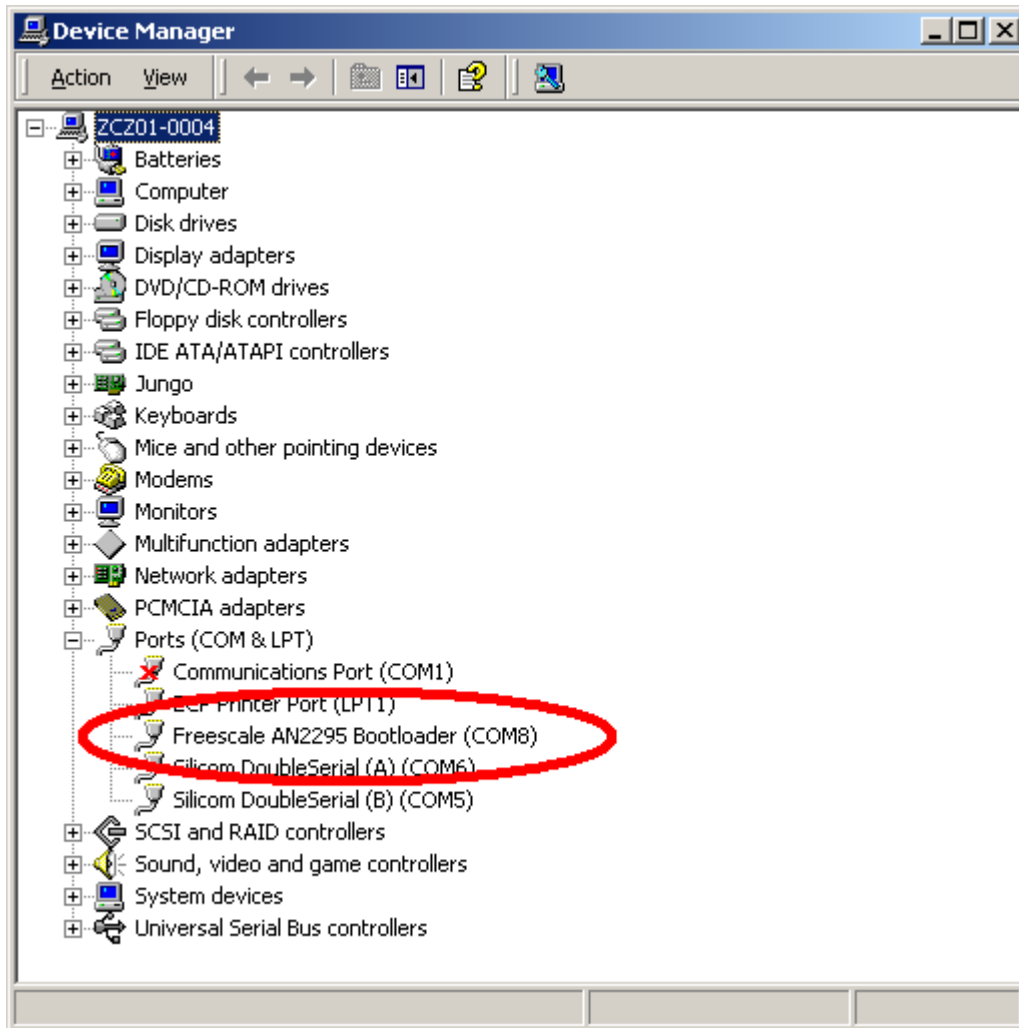
- The PC searches for an appropriate driver (as the ZSTAR Demo, in some instances a folder with drivers (`zstar.inf` and `usbser.sys`) needs to be selected), then the following window should appear:



- Just click Yes, and the bootloader port will be installed (as seen in the Device manager):
- Right click **My computer** on the Desktop > **Properties**, **Hardware** tab, **Device Manager** button.

Application Setup

5. A similar setup should be observed:



6. Note down the COM port number (here, COM8); this is the port number of the Bootloader

Once the software in the USB stick needs to be updated, the Bootloader can be invoked anytime, just by pressing the button while inserting the USB stick into the USB slot.

Appendix A

References

The following documents can be found on the Freescale web site: <http://www.freescale.com>.

1. AN2295 Application note - Developer's Serial Bootloader for M68HC08 and HCS08 MCUs
2. AN3153 Application note - Using the Full-Speed USB Module on the MCHC908JW32
3. MC9S08QG8 Datasheet
4. MCHC908JW32 Datasheet
5. MMA7260Q Datasheet
6. MC13191 Datasheet



How to Reach Us:

Home Page:

www.freescale.com

E-mail:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:

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

Japan:

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

Asia/Pacific:

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

For Literature Requests Only:

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

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics of their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see <http://www.freescale.com> or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to <http://www.freescale.com/epp>.

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.

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. 2006. All rights reserved.

