CHALMERS

Design of a ZigBee Magnetic Sensor Node

Master of Science Thesis

SÉGOLÈNE ARRIGAULT
VAIA ZACHARAKI

Department of Signals and Systems
Division of Communication Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden, 2007
Report No. EX084/2007

Chalmers University of Technology

Master Thesis (Report's number: EX084/2007)

"Design of a ZigBee Magnetic Sensor Node"

Ségolène Arrigault and Vaia Zacharaki

Supervisor: Professor Erik G. Ström

2007-09-18

Contents

List of Tables

List of Figures

Acronyms

ACK Acknowledge

ADC Analog-to-Digital Converter

AGC Automatic Gain Control

AMR Anisotropic Magneto-Resistive

ANSI American National Standards Institute

API Application Programming Interface

BDM Background Debug Mode

BPSK Binary Phase Shift Keying

CAD Computer-Aided Design

CCA Clear Channel Assessment

CE Chip Enable

CFM ColdFire Flash Module

CPU Central Processing Unit

CRC Cyclic Redundancy Check

CSMA-CA Carrier Sense Multiple Access with Collision Avoidance

DCD Differential Chip Detection

DIV divider

DMA Direct-Memory Access

DTIM DMA Timer

DSSS Direct Sequence Spread Spectrum

ED Energy Detection

EPORT Edge Port

EVB Evaluation Board

FCS Frame Check Sequence

FFD Full-Function Device

FIFO First-In First-Out

FLI Frame Length Indicator

GPIO General Purpose I/O

GPT General-Purpose Timer

GTS Guaranteed Time Slots

I^2C Inter-integrated Circuit

Id Identification number

IDE Integrated Development Environment

IF Intermediate Frequency

INTC Interrupt Controller

I/O Input/Output

I-phase In-phase

IRQ Interrupt Request

ISM Industrial, Scientific and Medical

JTAG Joint Test Action Group

LQFP Leaded Quad Flat Package

LQ Link Quality

LQI Link Quality Indication

LVD Low Voltage Detector

MAC Multiply Accumulate Unit, also Media Access Controller

MCU Microcontroller Unit

MFD Multiplication Factor Divider

MFR MAC Footer

MHR MAC Header

MISO Master In Slave Out

MLME MAC subLayer Management Entity

MOSFET Metal Oxide Semiconductor Field Effect Transistor

MOSI Master Out Slave In

MPDU MAC Protocol Data Unit

MSb Most Significant bit

MSDU MAC Service Data Unit

NACK "Not Acknowledge"

NVM Non-Volatile Memory

O-QPSK Offset Quadrature Phase-Shift Keying

OSI Open Systems Interconnection

PA Power Amplifier

PAN Personal Area Network

PC Personal Computer

PCB Printed Circuit Board

PER Packet Error Rate

PHY Physical

PHR Physical Header

PIT Programmable Interval Timer Module

PLL Phase-Locked Loop

PLME Physical Layer Management Entity

PN Pseudo-random Noise

POR Power-On Reset

PPDU Physical Protocol Data Unit

PSDU Physical Service Data Unit

PWM Pulse Width Modulation

Q-phase Quadrature-phase

QSPI Queued Serial Peripheral Interface

RAM Random Access Memory

RF Radio Frequency

RFD Reduced-Function Device also Reduced Frequency Divider

RISC Reduced Instruction Set Computing

RST Reset

RX Receive

SAP Service Access Point

SCL Serial Clock

SDA Serial Data

SFD Start-of-Frame Delimiter

S/H Sample and Hold

SHR Synchronization Header

SMAC Simple Media Access Controller

SPI Serial Peripheral Interface

S/R Set/Reset

SRAM Static RAM

SSCS Service Specific Convergence Sublayer

TC1 Timer Comparator

TX Transmit

UART Universal Asynchronous/Synchronous Receiver/Transmitter

USB Universal Serial Bus

VCO Voltage Controlled Oscillator

WPAN Wireless Personal Area Network

Abstract

The goal of this project was the creation of an efficient tool in order to study the road traffic as to find future solutions to control the traffic and reduce both the travel time and negative effects on the environment. This tool should provide cost-efficient and accurate estimation of traffic parameters, so it has been decided to base the estimation on data collected from wireless sensor network, based on ZigBee Magnetic sensor nodes placed next to the road.

This project was divided into three main tasks: a hardware part including the design of the tool and its creation, a software part including the on-board programming as well as the design of the network and finally a test and verification part.

Future enhancements can already be foreseen: the implementation of the ZigBee protocol can be proposed on top of the IEEE 802.15.4 protocol which is already implemented.

One important remaining problem is the power consumption of the nodes, which is still too high. Some duty cycles should be implemented so as to reduce the power consumption of the nodes. The main idea would be that each node sleeps while it is not sensing or transmitting information.

Part I

Introduction

Nowadays, road traffic is a very important aspect of our society, since it is part of our everyday life. Unfortunately, it has strong negative impacts on human everyday life (congestion, noise annoyance) as well as on the environment (atmospheric pollution). Therefore the issue of making traffic flow more efficiently is of great importance.

The goal of this project is to provide an efficient tool to study the traffic, so that future solutions can be found to control the traffic and reduce both travel time and negative effects on the environment. This tool should provide cost-efficient and accurate estimation of traffic parameters, such as the number of vehicles passing a certain point per unit time, the type of vehicles (cars, trucks, motorcycles, etc.), the direction of their movement and their current speed. For this purpose, it has been decided to base the estimation on data collected from wireless sensor network based on ZigBee. Magnetic sensor nodes will be placed close to the road and will create a wireless network so as to exchange information and supply accurate traffic estimations.

This project includes the whole process of design, manufacturing, and verification of a ZigBee network with magnetic sensors. The design includes both hardware and software design. The hardware consists of the design and manufacturing of a functional ZigBee node with a magnetic sensor. The software design includes the programming of each specific node to assure the communication within the node, as well as to allow all nodes to communicate with each other and form a ZigBee network. Finally, a verification phase is conducted so as to characterize the designed node in means of power consumption, radiated power spectrum, antenna radiation pattern and radio sensitivity.

The project has been initiated by Geveko Industry Holding AB. It is conducted by different companies and institutions working in partnership:

- Qamcom Technology AB: a technology company that develops products and systems in which communication technology plays a major part.
- IMEGO: the institute of micro and nanotechnology in Gothenburg, expert company within sensor system development.
- Geveko: an investment company focused on Industrial Operations and Management of Securities.
- Signal and System Institution at Chalmers University of Technology in Gothenburg

The work described in the rest of this report has been achieved by Ségolène Arrigault and Vaia Zacharaki at Qamcom Technology AB so as to perform their In-

ternational Master thesis for the Department of Signals and Systems at Chalmers University of Technology.

Part II

Presentation ZigBee

The first step of this project was for us to get familiar with the IEEE 802.15.4 and the ZigBee standards. Therefore, we read articles and documents that we found on internet and that are listed in references [1], [2] and [3].

The IEEE 802.15.4 and ZigBee standards are used in wireless applications (Wireless Personal Area Network (WPAN) technology: from industrial monitoring and control, home automation, sensor networks to gaming, medical and automotive solutions...) which require modest (small) data transmission but still need reliable and secure communication using simple low-cost and low-power radio systems. To respond to these needs, these standards provide reliable, interoperable, secure (wireless low data rate control and monitoring applications), low-power consumption and low-cost flexible wireless network technology in short-range communications (1 to 75 meters range).

The IEEE 802.15.4 standard specifies the Physical (PHY) and Media Access Control (MAC) layers at the 868 MHz (Europe), 915 MHz (North America) and 2.4 GHz (Worldwide) ISM bands, enabling global or regional deployment, at a data rate of 20 kbps, 40 kbps and 250 kbps respectively. The air interface is Direct Sequence Spread Spectrum (DSSS) using Binary Phase Shift Keying (BPSK) for 868 MHz and 915 MHz and Offset Quadrature Phase-Shift Keying (O-QPSK) for 2.4 GHz. The access method in IEEE 802.15.4-enabled networks is Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA), implemented either with a non-beacon or beacon mode. The IEEE 802.15.4 standard employs 64-bit IEEE and 16-bit short addresses to support theoretically more than 65,000 nodes per network. [2]

The ZigBee standard is published by the ZigBee Alliance, an industry group of over 50 companies (Ember, Freescale, Chipcon, Invensys, Mitsubishi, CompXs, AMI Semiconductors, Figure 8 Wireless, ENQ Semiconductor). It defines the network, security and application layers of the system. [3]

Chapter 1

Network topologies

The ZigBee network layer supports three networking topologies: Star, mesh (peer-to-peer) and cluster-tree (hybrid star/mesh). [2]

Star Topology: In this configuration (Figure 1.1), one device acts as the central node of communication (Personal Area Network (PAN) coordinator) through which all the information are exchanged. In fact this device is the only one capable of communicating with the other devices on the network and is described as a FFD. The coordinator should always be in receiving mode when not transmitting so as to be able to listen to all the other nodes. The other devices in the network are either Reduced-Function Devices (RFDs: low-cost and small memory micro controllers) or FFDs. They are not able to communicate with any other node except for the coordinator device. The RFDs receive and transmit for short periods of time, achieving low-power consumption. Consequently, the RFDs can operate using only batteries while the coordinator and the FFDs should probably be mains powered. [3]

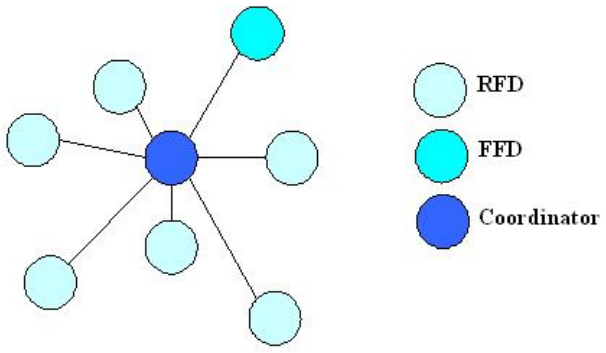


Figure 1.1: ZigBee Star Topology

Cluster Tree Topology: The cluster tree topology (Figure 1.2) is very similar to the star topology. The difference is that other nodes in addition to the coordinator can communicate with each other so that more RFD/FFDs can be connected to non-coordinator FFDs. The advantage of this topology is the possible geographical network expansion. [3]

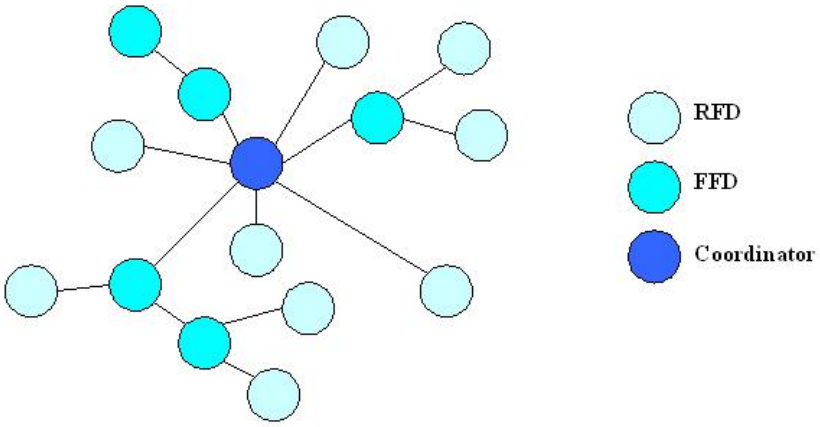


Figure 1.2: ZigBee Cluster Tree Topology

Mesh Topology: In this configuration (Figure 1.3), all FFDs are directly connected between each other and one of them acts as the coordinator. RFDs can also be included in the network but they can only be connected to their parent FFD. The advantage of this topology is that it provides a reliable routing network. [3]

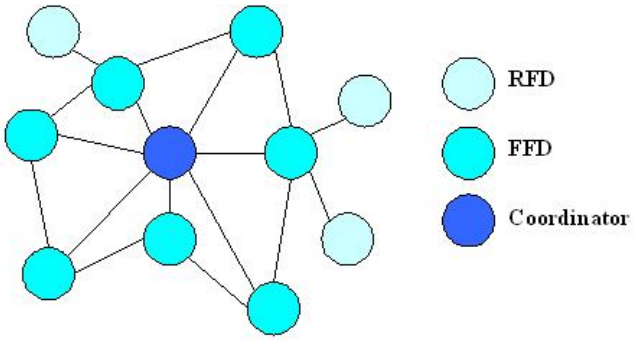


Figure 1.3: ZigBee Mesh Topology

Chapter 2

Architecture

The network architecture is based on the Open Systems Interconnection (OSI) seven-layer model, where each layer is responsible for one part of the standard and offers services to the higher layers. Some interfaces define the logical links between the different layers. Therefore, the different layers are the PHY layer that defines all the electrical and physical specifications for the device (includes the Radio Frequency (RF) transceiver and its low-level control mechanism: connection to a communication medium, channel access, modulation), a MAC sublayer providing access to the PHY for all types of transfer and the upper layers. The upper layers consist of a network layer providing network configuration, manipulation and message routing and an application layer implementing the function of the device. Figure 2.1 shows the architecture model of 802.15.4 and ZigBee standards. [1] and [2]

2.1 PHY layer

The PHY provides two services: the PHY data service and the PHY management service interfacing to the Physical Layer Management Entity (PLME). The PHY data service enables the transmission and reception of Physical Protocol Data Unit (PPDU)s across the physical radio channel. The features of the PHY are activation and deactivation of the radio transceiver, Energy Detection (ED) within the current channel, Link Quality Indication (LQI) for received packets, channel frequency selection, Clear Channel Assessment (CCA) for CSMA-CA, and transmitting as well as receiving packets across the physical medium. [1]

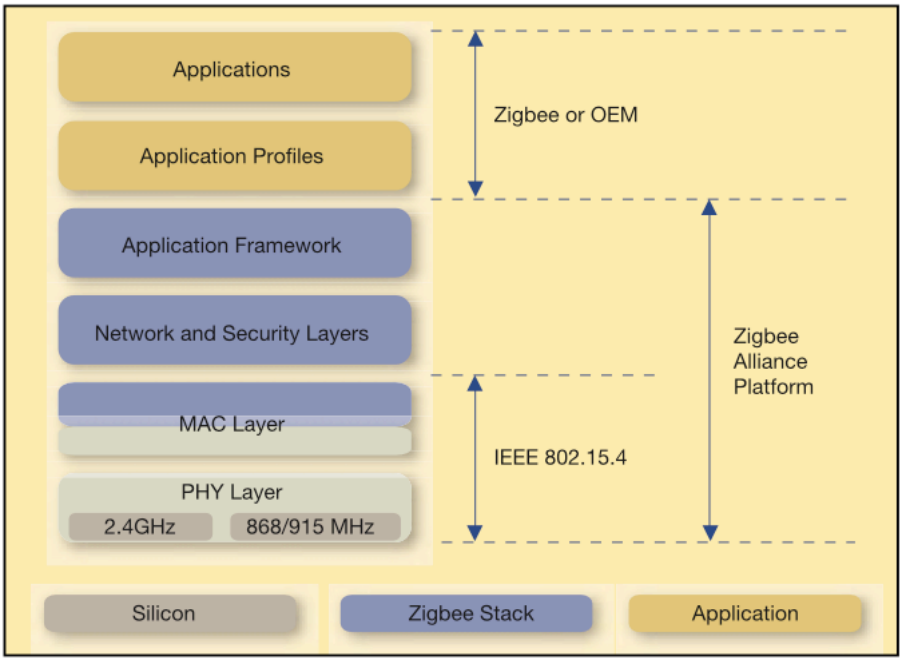


Figure 2.1: Architecture Model

2.2 Operating frequency range

Table 2.1 shows one or several frequency bands along with the modulation and spreading formats for the successful operation of the device. The IEEE 802.15.4 standard fits the regulations in Europe, Japan, Canada and the United States. A total of 27 channels (numbered from 0 to 26) are allocated in the three frequency bands. The channels' assignment as well as the formulas that determine the center frequency of these channels are also shown in Table 2.1.

Table 2.1: Frequency bands, data rates, channel assignments and numbering

PHY (MHz)	**Frequency Band (MHz)**	**Center Frequency (MHz)**	**Number of Channels**	**Spreading parameters**		**Data parameters**		
				Chip rate (kchip/s)	**Modulation**	**Bit rate (kb/s)**	**Symbol rate (ksymbol/s)**	**Symbols**
868/915	868 - 868.6	$F_c = 868.3$, for $k = 0$	1	300	BPSK	20	20	Binary
	902 - 928	$F_c = 906 + 2(k-1)$, for $k = 1, 2, ..., 10$	10	600	BPSK	40	40	Binary
2450	2400 - 2483.5	$F_c = 2405 + 5(k-11)$, for $k = 11, 12, ..., 26$	16	2000	O-QPSK	250	62.5	16-ary Orthogonal

In this project, we used the frequency 2.4 GHz as the operating frequency (requirement of the project). The data rate of the IEEE 802.15.4 PHY at 2450 MHz is 250 kb/s. The 2450 MHz PHY employs a 16-ary quasi-orthogonal modulation technique. During each data symbol period, four information bits are used to select one of 16 nearly orthogonal Pseudo-random Noise (PN) sequences to be transmitted. The PN sequences for successive data symbols are concatenated, and the aggregate chip sequence is modulated onto the carrier using O-QPSK.

The functional block diagram of the modulation and spreading functions of the 2450 MHz PHY is presented in Figure 2.2.

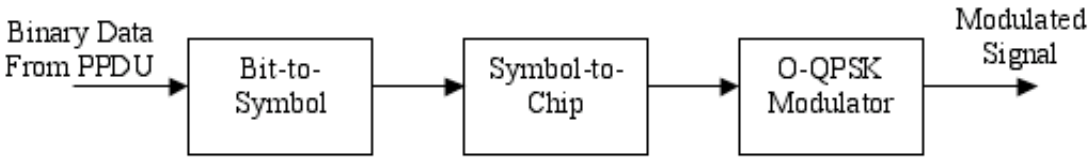


Figure 2.2: Modulation and spreading functions

- Bit-to-symbol mapping: In this stage, the bit sequence (PPDU) is mapped into a symbol sequence by representing every consecutive four bits with one symbol. The 2450 MHz PHY symbol rate shall be 62.5 ksymbol/s ±40 ppm.
- Symbol-to-chip mapping: Each data symbol is mapped into a 32-chip PN sequence.
- O-QPSK modulation: The chip sequences representing each data symbol are modulated onto the carrier using O-QPSK with halfsine pulse shaping. Even-indexed chips are modulated onto the In-phase (I-phase) carrier and odd-indexed chips are modulated onto the Quadrature-phase (Q-phase) carrier. Because each data symbol is represented by a 32-chip sequence, the chip rate (nominally 2.0 Mchip/s) is 32 times the symbol rate. To form the offset between I-phase and Q-phase chip modulation, the Q-phase chips shall be delayed by $\mathcal{T}_c$ with respect to the I-phase chips, where $\mathcal{T}_c$ is the inverse of the chip rate. The half-sine pulse shape used to represent each baseband chip is described by Equation 2.1:

$$p(t) = \begin{cases} \sin(\pi \frac{t}{2\mathcal{T}_c}), & 0 \leq t \leq 2\mathcal{T}_c \\ 0, & otherwise \end{cases} \tag{2.1}$$

Figure 2.3 shows a sample baseband chip sequence with half-sine pulse shaping.

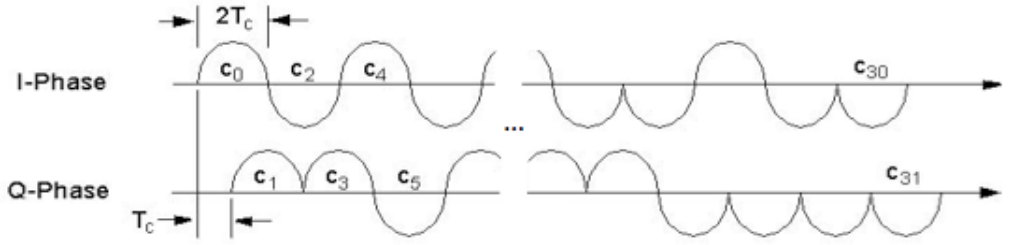


Figure 2.3: Sample baseband chip sequences with pulse shaping

During each symbol period the least significant chip, $\mathcal{C}_0$, is transmitted first and the most significant chip, $\mathcal{C}_{31}$, is transmitted last. [1]

2.3 MAC sublayer

The MAC sublayer provides an interface between the Service Specific Convergence Sublayer (SSCS) and the PHY. The MAC sublayer provides two services: the MAC data service and the MAC management service interfacing to the MAC subLayer Management Entity (MLME) Service Access Point (SAP) (known as MLME-SAP). The MAC data service enables the transmission and reception of MAC Protocol Data Units (MPDUs) across the PHY data service. The features of the MAC sublayer are beacon management, channel access (CSMA-CA mechanism), Guaranteed Time Slots (GTS) management, frame validation, acknowledged frame delivery, PAN association and disassociation. In addition, the MAC sublayer provides hooks for implementing application appropriate security mechanisms. [1]

Chapter 3

Functional overview

In order to describe the general network functionality, information on the data transfer model, the frame structure, the robustness and power consumption considerations are detailed below.

3.1 Data transfer model

In this project, a star topology was chosen, where either the devices transfer data to the coordinator or the coordinator transfers data to the network devices.

The data transfer transactions between the coordinator device and the RFD nodes follow the scheme below and are shown in Figure 3.1.

- The communication is initiated by the coordinator that sends a data request to a network device.
- The network device replies by transferring data to the coordinator.

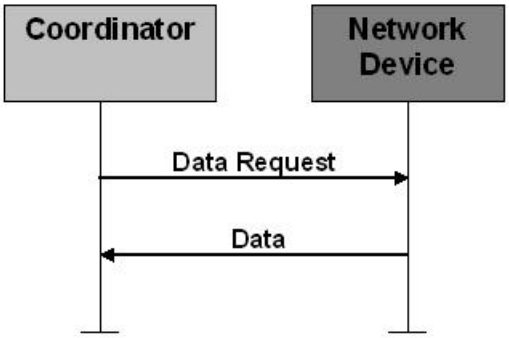


Figure 3.1: Communication to a coordinator in a network

3.2 Frame structure

The design of the frame structures is of simple conception while being robust enough for transmission on a noisy channel. Each layer adds to the frame its own specific headers and footers. The standard defines four frame structures: a beacon frame, a data frame, an acknowledge frame and a MAC command frame. In this project only the data frame is being used and detailed. The data frame is used for all transfers of data. [1]

3.3 Data frame

The structure of the data frame, which originates from the upper layers is shown in Figure 3.2.

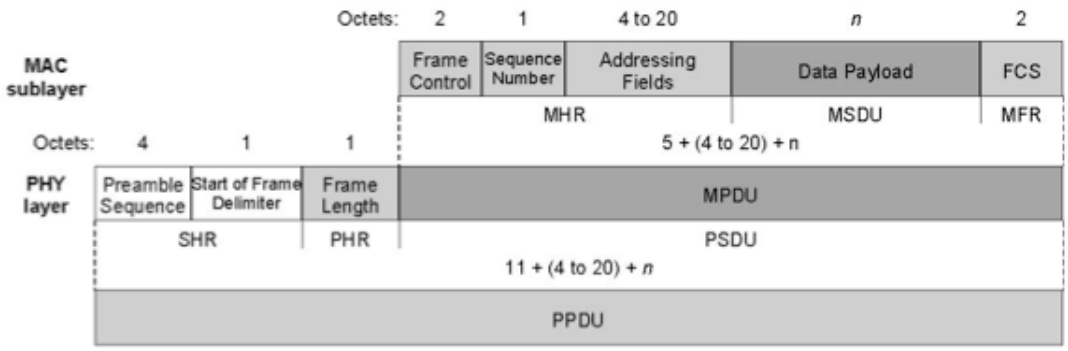


Figure 3.2: Schematic view of the data frame

The data payload is passed to the MAC sublayer and is referred to as the MAC Service Data Unit (MSDU). The MSDU is prefixed with an MAC Header (MHR) and appended with an MAC Footer (MFR). The MFR contains the frame control, sequence number, and addressing information fields. The MFR is composed of a 16 bit Frame Check Sequence (FCS). The MHR, MSDU, and MFR together form the MAC data frame, (i.e., MAC Protocol Data Unit (MPDU)). The MPDU is passed to the PHY as the PHY data frame payload, (i.e., Physical Service Data Unit (PSDU)).

The PSDU is prefixed with an Synchronization Header (SHR), containing the preamble sequence (for synchronization) and Start-of-Frame Delimiter (SFD) fields, and a Physical Header (PHR) containing the length of the PSDU in octets. The SHR, PHR, and PSDU together form the PHY data packet, (i.e., PPDU). [1]

3.4 Robustness

The IEEE 802.15.4 and ZigBee standards use different mechanisms for the data transmission to be robust, such as the CSMA-CA, the frame acknowledgment and the data verification mechanisms (FCS mechanism using a 16 bit International Telecommunication Union-Telecommunication Standardization Sector (ITU-T) Cyclic Redundancy Check (CRC)). [1]

3.5 Power consumption considerations

In this project, all the devices are battery powered except for the coordinator node which can be either mains-powered or battery powered. Unfortunately, it is impractical to replace or recharge the batteries of those devices regularly (due to their application and geographical allocation), so that the power consumption is of major importance. The IEEE 802.15.4 and ZigBee standards were developed taking into consideration the significant aspect of low-power consumption and limited power supply.

As to reduce the devices' power consumption, a duty cycle can be implemented (future improvement). The RFDs would mostly be in a sleeping mode and would only wake up when needed to transmit. The coordinator, however, should listen to the channel continuously. [1]

Part III

Selection of the components

The next step, after getting familiar with the ZigBee standard is to decide the different components which will be necessary for the creation of the node. The main components of the node are a magnetic sensor, a ZigBee transceiver and a microcontroller that permits and controls the communication between the different units.

Chapter 4

Survey on ZigBee chipsets

We first concentrated on the choice of the ZigBee chipset to be used for the creation of the node. A wireless application is based on three components: radio, microcontroller and firmware. Therefore, in order to select these components, we performed a survey, based on the project requirements, on the different ZigBee chipsets available on the market, browsing on internet. The documents used for making the survey are listed in reference [4].

First of all, the transceiver (radio) should fulfill the following requirements:

- Short range data links and networks
- Low power consumption (required since the nodes are battery-powered)
- Low cost
- Compliant with the IEEE 802.15.4 standard
- Operating at 2.4 GHz ISM band (this frequency complies with worldwide regulations and allows to achieve a data rate of 250 kbps)
- Supporting star topology
- Compatible with any Microcontroller Unit (MCU)
- Compatible with ZigBee networking
- Reliable

Table 4.1 presents a comparison between different IEEE 802.15.4 compliant transceivers, operating at 2.4 GHz. [5]

Table 4.1: Transceiver Market Survey

Part Number	**Texas Instruments (Chipcon) CC2420**	**Atmel AT86RF230**	**FreeScale MC13191/ MC13192/ MC13193**	**FreeScale MC13201/ MC13202/ MC13203**	**Microchip MRF24J40**	**Oki ML7065**
Frequency Band (MHz)	2400 to 2483.5	2405 to 2480	2405 to 2480	2405 to 2480	2405 to 2480	2400 to 2483.5
Operating Voltage range (V)	2.1 to 3.6	1.8 to 3.6	2.0 to 3.4	2.0 to 3.4	2.4 to 3.6	2.0 to 2.75
Modulation Techniques	O-QPSK DSSS	O-QPSK	O-QPSK DSSS	O-QPSK DSSS	O-QPSK	O-QPSK DSSS
Current Consumption (RX) (mA)	19.7	16	37	37	22	57
Current Consumption (TX) (mA)	17.4	17 (max. PTX)	30	30	18	56
Nominal Output Power (dBm)	0	3	0	0	0	-3 to 3
Programmable Output Power Ranging From (dBm)	-25 to 0	-17 to 3	-27 to 4	-27 to 3		
Operating temperature range (∘C)	-40 to 85	-40 to 85	-40 to 85	-40 to 85	-40 to 85	-25 to 70
Data Rate (max) (Mbps)	0.25	0.25	0.25	0.25		0.25
Data Buffering (bytes)	128	128	125	125	122	
Receiver Sensitivity (dBm)	-95	-101	-92	-91/-92/-92	-91	-90
Crystal oscillator frequency (MHz)	16	16	16	16	20	10
IEEE 802.15.4 Compliant	Yes	Yes	Yes	Yes	Yes	Yes
Approx. 1KU Price (US$)	3.30	6.13	2.06/2.57/2.94	2.24/2.61/3.13	2.99	

After conducting the survey and discussing with Qamcom Technology AB, the FreeScale MC13192 transceiver [6] was chosen. In fact, this transceiver follows the project specifications, as listed below:

- Short range data links and networks: it is capable of reaching 650 to 1000 meters without the use of an on-board LNA [7]
- Low power consumption: it has three power down modes for power conservation (off, hibernate and doze modes)
- Low cost
- Compliant with the IEEE 802.15.4 standard: it supports 250 kbps O-QPSK data in 5 MHz channels with full spread-spectrum encode/decode and contains complete 802.15.4 PHY modem
- Operating at 2.4 GHz Industrial, Scientific and Medical (ISM) band
- Supporting star and mesh topologies
- Compatible with any MCU: the interface with the MCU is accomplished utilizing a four wire SPI connection which allows for the use of a variety of processors
- Compatible with ZigBee networking: the software and processor can be scaled to fit the application from simple point to point proprietary systems to ZigBee networking
- Reliable: the receiver sensitivity is of -92 dBm (Typical) at 1.0 % PER. The transceiver also supports DSSS coding and decoding as well as check mechanisms like FCS generating a CRC.

As far as the selection of the microcontroller is concerned, any microcontroller can be used for the creation of the node, provided that it fulfills the project requirements as well. These requirements for the MCU are listed below:

- Low power consumption (required since the nodes are battery-powered)
- Low cost
- High speed
- Large memory
- Serial communication interfaces: to be used for the communication with the sensors (see section 11.3 on page 70) as well as for debug purposes

- Fast ADC: to be used for converting the sensor's analog measurements into digital values for transmission
- A QSPI: to be used for the communication with the transceiver

Table 4.2 presents a comparison between different microcontrollers that seem to be more efficient than others when being used with a compliant IEEE 802.15.4 transceiver. Table 4.2 also presents the Integrated Development Environment (IDE) used to program the corresponding microcontroller. [8]

Table 4.2: Microcontroller Market Survey

Part Number	**Texas Instruments MSP430F149**	**Atmel AVR Atmega64L/ Atmega128L**	**Microchip PIC18F4620**	**FreeScale MCF5212**	**FreeScale MC9S08GB/GT family**
Frequency (MHz)	8	8	40	66/80	40
Flash (KB)	60	64/128	64	256	16/32/60
RAM(KB)	2	4	3.968	32	1/2/4
Power supply range (V)	1.8 to 3.6	2.7 to 5.5	2.0 to 5.5	3.0 to 3.6	1.8 to 3.6
Operating temperature range (∘C)	-40 to 85	-40 to 85	-40 to 85	-40 to 85	-40 to 85
GPIO	48	53	36	44	34-36-39/34-36-39-56
ADC	12-bit, 8-channel	10-bit, 8-ch	10-bit, 13-ch	12-bit, 8-ch	10-bit, 8-ch
Other Integrated Peripherals	Analog Comparator, Hardware Multiplier	AC, HM	AC, HM	HM	
Interface	2 USART (SPI or UART)	2 USART, Master/Slave SPI Serial Interface	Enhanced USART, I2C, Master Synchronous Serial Port (MSSP) module supporting 3-wire SPI	2/3 UARTs with DMA capabilities, I2C, QSPI	2 USART, SPI, Inter-Integrated Circuit (IIC), 2 Serial Communications Interface (SCI)
Timers	1 Watchdog/ Interval, 1 16-bit (3CCR), 1 16-bit (7CCR)	1 Watchdog, 2 8-bit, 2 16-bit	1 Watchdog, 4 timers	1 Watchdog, 4 16-bit, 4 32-bit timers	1 Watchdog, 2 2-ch, 16-bit/1 Watchdog, 2 2-ch, 16-bit, 1 3-ch and 1 5-ch 16-bit
Software Integrated Development Environment (IDE)	IAR Embedded Workbench, TI Code Composer Essentials Professional	AVR Studio 4, Mature AVR Studio 3.5	Application Maestro Software (MPLAB®IDE)	FreeScale Metrowerks CodeWarrior for ColdFire	FreeScale CodeWarrior
Pin/Package	64LQFP, 64QFN, 64TQFP	64-lead TQFP and 64-pad QFN/MLF	40 PDIP/44 TQFP/44 QFN	64 LQFP/ 81 MAPBGA	48 QFN, 44 QFP, 42 SDIP/48 QFN, 44 QFP, 42 SDIP, 64 LQFP
Approx. 1KU Price (US$)	6.05	10.68/12.6	5.44/5.47/5.72	6.650	1.75, 1.7, 1.7/ 2.75, 2.65, 2.65,2.95/ 3.75, 3.65, 3.9, 3.95

After conducting the survey and discussing with Qamcom Technology AB, the FreeScale MCF5212 microcontroller [9] was chosen. It is ideal for networking since it provides high performance for cost-sensitive applications that require significant control processing for connectivity, data buffering, and user interface, as well as signal processing. The MCF5212 is a 32-bit highly integrated implementation of Coldfire family Reduced Instruction Set Computing (RISC) microcontrollers. In fact, this microcontroller follows the project specifications, as listed below:

- Low power consumption: it has three power down modes for power conservation (stop, doze and wait modes).
- Low cost
- High speed: it has a system clock of up to 66 MHz.
- Large memory: it has 256 KBytes of Flash memory and 32 KBytes of Random Access Memory (RAM).
- Serial communication interfaces: the MCF5212 has:
  1. 2 Universal Asynchronous/Synchronous Receiver/Transmitter (UART)s, with DMA capabilities, used for debugging and communicating the measurements of the sensors to a computer with a screen for further process.
  2. an I^2C interface used for communicating with the temperature sensor.
  3. a high-speed serial command interface to which a Background Debug Mode (BDM) port is connected to flash the microcontroller with C code and to provide real-time access to all hardware, peripherals and memory on the board.
- Fast ADC: it has a 8-channel, 12-bit resolution ADC with a clock running up to 5.0 MHz, used for converting the sensor's analog measurements into digital values for transmission.
- A QSPI: it has a full-duplex, three-wire synchronous transfers QSPI with programmable bit rates up to half the Central Processing Unit (CPU) clock frequency, used for the communication with the transceiver.

Finally, the IDE used to program the MCF5212 is the FreeScale Metrowerks CodeWarrior for ColdFire IDE, as shown in Table 4.2.

Moreover, in order to create and program the node, we studied the FreeScale M5213EVB (Evaluation Board (EVB)). The purpose of the development board is to assist the user in quickly developing an application with a known working

environment as well as to provide an evaluation platform. The board consists, among others, of the MCF5213 CPU (100-LQFP (Leaded Quad Flat Package (LQFP))), the 2.4 GHz ZigBee capable RF transceiver MC13192 and a BDM port through which we flash the microcontroller with the C code (developed with the FreeScale Metrowerks CodeWarrior for Coldfire IDE), using the P&E's Universal Serial Bus (USB) ColdFire Multilink Interface. This provides real-time access to all hardware, peripherals and memory on the board. The M5213EVB is also provided with example codes implementing simple PHY and simple MAC layers for the IEEE 802.15.4 standard.

Chapter 5

Choice of the Sensors

Now that we selected which transceiver and microcontroller to use, we still have to choose the magnetic sensor. The sensors should be carefully chosen since the accuracy of the measurements is one of our first concerns. For this purpose, the company IMEGO, expert within sensor system development, conveyed a survey [10] on the different commercial sensitive solid state magnetic sensors available on the market. The survey was based on the project requirements which are:

- High enough field sensitivity in order to make measurements of the small disturbances in the earth magnetic field
- High field resolution
- Simplicity of the magnetic sensor
- Low cost
- Low energy consumption

The survey included only magnetic sensors that are relevant for traffic detection regarding all of the previous requirements, so that only sensors from Honeywell, Nonvolatile Electronics (NVE) and Philips were compared.

IMEGO presented their results during a meeting, that we attended, to Qamcom Technology AB and Geveko. It has been decided that 3-dimension (x, y and z axis) measurements are needed for accurate and reliable car detection (The choice of one, two, or three-axis magnetic field sensing is a trade off in performance versus cost). However, this requirement can be achieved in two different ways: either by using a 2-axis sensor combined with a 1-axis sensor or by using directly a 3-axis sensor. Since at this phase of the project, it is not clear which solution gives the best performance, all three sensors will be mounted on the board and

tested. Therefore according to IMEGO's survey and Qamcom Technology AB's decision, the Honeywell Corporation sensors were selected. The 1-axis and 2-axis sensors are the HMC1001 and HMC1002 [11], respectively, and the 3-axis sensor is the HMC1043 [12], since they offer a small, low cost, high sensitivity and high reliability solution for low field magnetic sensing, as shown in Table 5.1.

Table 5.1: Features of Honeywell sensors HMC1001, HMC1002 and HMC1043

Sensor	**#Axis**	**Sensitivity (mV/V/ (mT)**	**Power Consumption (mW) at supply voltage Vcc=5V**	**Power Supply (V) Typical value**	**Field Range (mT)**	**Field Resolution (gauss)**	**BW (MHz)**	**Operational Temperature (∘C)**	**Price ($)**
HMC 1001	1	32		5	±0.2	27	5	-55 to 150	<11:$17 <1001:$10 <10001:$9
HMC 1002	2	32		5	±0.2	27	5	-55 to 150	<11:$19 <1001:$15 <10001:$12
HMC 1043	3	10	75	3	±0.6	120 (for 50Hz BW, Vbridge=5V)	5 5 5	-40 to 125 -40 to 125 -40 to 125	>500:$12 >3000:$9 >10000:$4 to $5

Moreover, it has also been decided that for assuring good accuracy (sensitivity) of the magnetic sensor's measurements, a temperature sensor is needed. Ideally, we would like the resistance of the sensors (and therefore the voltage output) to change only in response to the applied magnetic field. However, the resistance of the sensors, since they are made of nickel-iron (permalloy), also responds to changes in temperature (Figure 5.1). Temperature related effects are the most common causes of error in sensor measurements. Therefore the goal of the temperature sensor is to provide us with the temperature value for each magnetic sensor measurement, so that the error introduced by the temperature can be corrected later on, in software. For this purpose, it has been decided with Qamcom Technology AB to use the DS1631 high-precision digital thermometer and thermostat from Maxim Integrated Products and Dallas Semiconductor [13].

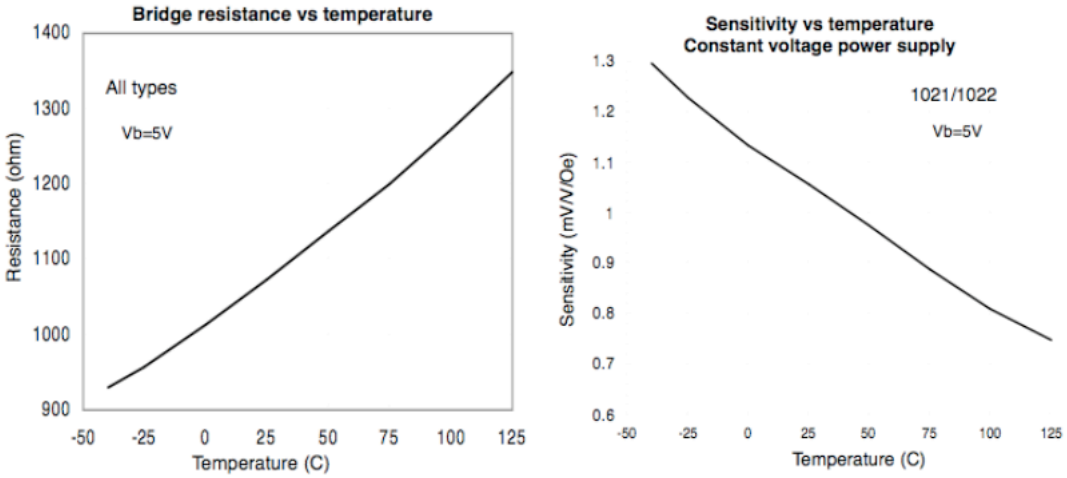


Figure 5.1: Sensors' Bridge Resistance and Sensitivity vs Temperature

Part IV

Creation and Description of the board

At this stage of the project, since we have selected the main components constituting the node, we started the design of the board. For this purpose, we familiarized first with a Computer-Aided Design (CAD) software and then with each specific component.

The CAD software used is Protel 99 SE, which is a complete board-level design system for Windows 2000/NT/98/95. Protel 99 SE provides a completely integrated suite of design tools permitting to take the designs from concept through to the final board layout. We used Protel 99 SE for drawing the schematic (diagram of the future board using graphic symbols) of the board and for designing the Printed Circuit Board (PCB). To familiarize with this CAD tool, we studied the step-by-step introductory tutorial Exploring Protel 99 SE [14]. The final schematics and PCB of the board are presented in Appendix A and Appendix B, respectively.

In order to get familiar with the main components of the board as well as to decide how to link them together, we studied the corresponding component datasheets. A brief description of each component along with the way they interface to the other components on the board is presented in Chapters 6, 7, 8 and 9 on pages 32, 36, 37 and 42 respectively.

Chapter 6

Magnetic sensors

In this project, we used magnetic sensors for vehicle detection, since almost all road vehicles have significant amounts of ferrous metals in their chassis (iron, steel, nickel, cobalt, etc.). It is known that the earth's magnetic field (in the range of 50 μT) permeates everything between the south and north magnetic poles. Therefore, "low field" magnetic sensors can be used to detect the earth's magnetic field disturbances created by vehicles with ferrous metals. As the lines of magnetic flux group together (concentrate) or spread out (deconcentrate), a magnetic sensor placed nearby will be under the same magnetic influence the vehicle creates to the earth's field.

The three Honeywell sensors chosen are Anisotropic Magneto-Resistive (AMR) sensors. They are simple 4-element (made of permalloy thin films) resistive wheatstone bridge devices (Figure 6.1) that only require a supply voltage to measure magnetic fields. In the presence of an applied magnetic field, a change in the bridge resistance causes a corresponding change in voltage output. The sensors are capable of sensing magnetic fields as low as 30 μgauss and each of them provides only an amplitude response to magnetic fields in its sensitive axis with an excellent linearity.

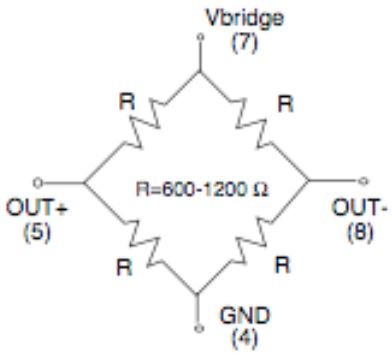


Figure 6.1: On-Chip components (HMC1001)

Because the AMR sensor outputs are in small millivolt levels given earth's magnetic field strengths, these wheatstone bridge sensors require follow-on amplification to make vehicle induced field changes easier to detect. With the differential output of the sensors, each sensor requires an amplifier stage compatible with the sensor output voltages and the sensor bridge supply voltage [15]. For this purpose, we used the ST Microelectronics LM224D quad operational amplifier.

In addition to the bridge circuit, the sensor has two on-chip magnetically coupled straps, the OFFSET strap and the Set/Reset strap [11]. The OFFSET strap is intended for creating a "magnetic offset" field and is not used in this project. The reasons to perform a set or reset on an AMR sensor are to optimize the magnetic domains for most sensitive performance, to flip the domains for extraction of bridge offset under changing temperature conditions and to re-align the magnetic domains of the permalloy thin film with the easy axis, after excessive magnetic fields (in excess of ± 10 gauss at the bridges) upset the sensor magnetic domain direction. This re-alignment is needed, since the exposure of the sensor to excessive magnetic fields can result in reduced sensitivity, or no change in sensor voltages (stuck sensors). Therefore, following such an upset field, a strong restoring magnetic field must be applied periodically to restore, or set, the sensor characteristics. The permalloy thin film magnetic domains get re-aligned in the easy axis directions and the memory of the upset field direction is erased. This effect will be referred to as applying a set pulse or reset pulse. The frequency of performing the set/reset pulses is entirely application dependant and can occur many times per second for more precision [16].

"Set" and "reset" pulses are defined as pulsed currents that enter the positive and negative pins of the set/reset strap, respectively. These set and reset pulses are shown in Figure 6.2 as dampened exponential pulse waveforms because the most popular method of generating these relatively high current, short duration pulses

is via a capacitive "charge and dump" type of circuit. In a typical charge and dump circuit for set and reset applications, each capacitor charge will store a fixed amount of energy which will be dissipated or "dumped" into a load resistance composed of the set/reset strap and pulse circuit components [16].

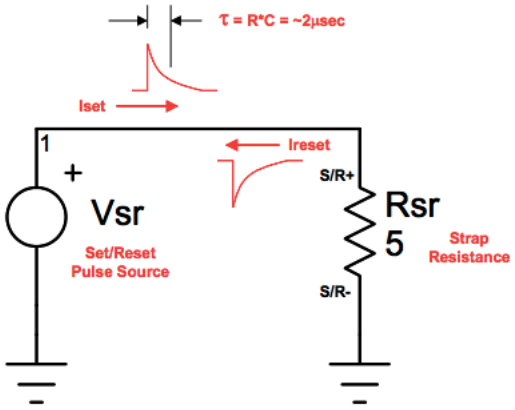


Figure 6.2: Set/Reset Circuit

The output response curves shown in Figure 6.3 illustrate the effects of the Set/Reset (S/R) pulse. When a SET current pulse (Iset) is driven into the SR+ pin, the output response follow the curve with the positive slope. When a RESET current pulse (Ireset) is driven into the SR- pin, the output response follow the curve with the negative slope. These curves are mirror images about the origin except for a bridge offset effect [11].

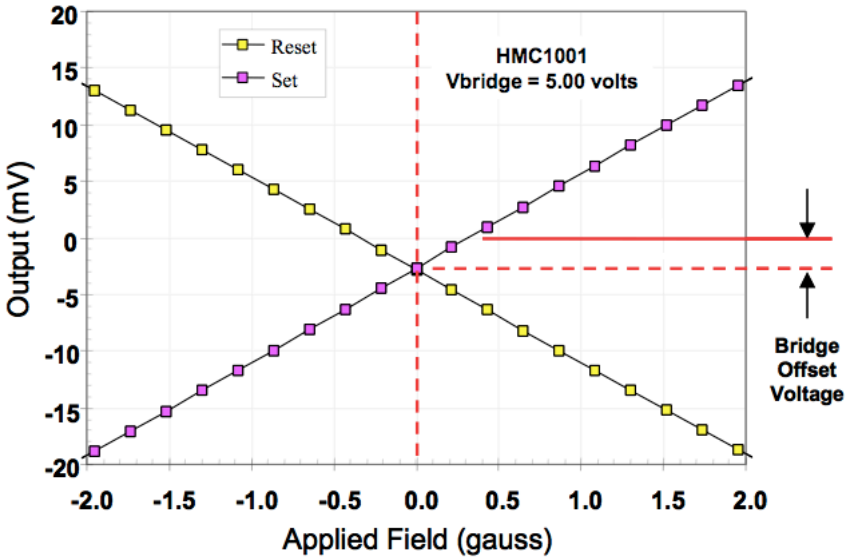


Figure 6.3: Offset/Sensitivity Graph

A bridge offset voltage is equal to the difference in volts of the output nodes, Vo+ and Vo-. A voltage output is expected on a wheatstone bridge across the output nodes, but an offset voltage is undesired output that does not change value or polarity if the sensor stimulus varies. In the case of AMR sensors, the bridge offset voltage is caused by in-exact resistive values of each sensor element (during the manufacture process) and may cause reduced performance of the sensor system if not compensated for, since it shifts the operating point of the output voltages [17]. However the good news is that this offset is fixed for the part's life with the remaining drift due to temperature changing the resistance of the bridge elements [15]. Several candidate methods of bridge offset reduction (offset trimmed to zero) exist. These are listed as 1) Shunt Resistance Method, 2) Amplifier Bias Nulling, 3) Switching Feedback, 4) Offset Strap Current and 5) Digital Subtraction. The method used in this project is the Digital Subtraction one, since it requires no change in hardware [17]. This method is fully detailed in section 11.3.1 on page 72.

The sensors are linked to the analog-to-digital pins of the MCU for communicating the magnetic measurements and to GPIO pins for generating the Set/Reset pulses. The connection of the sensors to the MCU is detailed in MCU part.

However because the sensor is not intimate to the surface or interior of the vehicle, it does not get the same fidelity of concentration or deconcentration. And with increasing standoff distance from the vehicle, the amount of flux density changes with vehicle presence drops of at an exponential rate.

Chapter 7

Temperature Sensor DS1631

The DS1631 measures temperature over a -55°C to +125°C range, using bandgap-based temperature sensors. A delta-sigma ADC converts the measured temperature to a 9-, 10-, 11-, or 12- bit (user-configurable) digital value in 750ms (maximum). After each conversion, the digital temperature is stored as a 16-bit (2-byte) two's complement number. The DS1631 thermometer accuracy is ±0.5°C from 0°C to +70°C with wide power-supply range $3.0V \leq \mathcal{V}_{DD} \leq 5.5V$.

The DS1631 communicates with the MCU over a bidirectional 2-wire serial data bus that consists of a Serial Clock (SCL) signal and Serial Data (SDA) signal. The DS1631 interfaces to the bus through its SCL input pin and open-drain SDA Input/Output (I/O) pin. Three address pins allow up to eight devices to be multi-dropped on the same 2-wire bus. However in this project, only one temperature sensor was used. [13]

Chapter 8

MC13192 Transceiver

The MC13192 is a short-range, low power, 2.4 GHz band transceiver (16 selectable channels). We chose this transceiver because, combined with our microcontroller, it provides a cost-effective solution for short-range data links and networks. Furthermore, the MC13192 includes the simple PHY/MAC layers, which support point-to-point and star network configurations, to be used with the MCF5213 family of MCUs.

The transceiver includes a low noise amplifier, 1.0 mW Power Amplifier (PA), Phase-Locked Loop (PLL) with internal Voltage Controlled Oscillator (VCO), on-board power supply regulation, and full spread-spectrum encoding and decoding. The device supports 250 kbps O-QPSK data in 2.0 MHz channels with 5.0 MHz channel spacing per the 802.15.4 Standard. The SPI port and interrupt request output are used for Receive (RX) and Transmit (TX) data transfer and control. The transmit and receive data packets are buffered for simplified use with low cost MCUs. The interface between the transceiver and the MCU is done by using a four wire SPI connection and an interrupt request output.

8.1 Data Transfer Modes

The MC13192 can support two modes of data transfer: the packet mode, in which the data is buffered in on-chip RAM and the streaming mode, in which the data is processed word-by-word. In our case, we are using the packet mode so that only this mode will be discussed in the rest of this chapter.

8.1.1 The packet structure of the MC13192

The packet structure of the MC13192 is shown in Figure 8.1. The maximum Payload Data supported is up to 125 bytes. The transceiver adds a preamble of 4 bytes as well as a SFD of 1 byte and a Frame Length Indicator (FLI) of 1 byte. Finally, a FCS of 2 bytes is calculated and added at the end of the data.

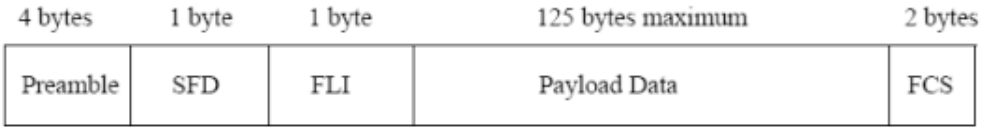


Figure 8.1: The packet structure of the MC13192

8.1.2 Receive Path Description

In the receive signal path, the RF input is converted to low Intermediate Frequency (IF) I-phase and Q-phase (I & Q) signals through two down-conversion stages. First of all, a CCA can be performed based upon the baseband energy integrated over a specific time interval. In our case, the ED algorithm of CCA is performed by measuring the channel energy and giving an indication of the measured strength. The digital back end performs Differential Chip Detection (DCD) and the correlator "de-spreads" the DSSS O-QPSK signal, determines the symbols and packets, and detects the data. (See section 2.2 on page 9) The preamble, the SFD and the FLI are used to detect the payload data and the FCS, which are stored in RAM. A two-byte FCS is calculated on the received data and is compared to the FCS value appended to the transmitted data, which generates a CRC result. Link quality is measured over a 64 μs period after the packet preamble and is stored in RAM. Since the MC13192 is in packet mode, the data is processed as an entire packet and the MCU is notified that an entire packet has been received via an interrupt.

8.1.3 Transmit Path Description

For the transmit path, the TX data that was previously stored in RAM is retrieved, formed into packets per the 802.15.4 PHY, spread, and then up-converted to the transmit frequency. The data is first loaded into the TX buffer. The MCU

then requests that the MC13192 transmit the data. The MCU is notified via an interrupt when the whole packet has successfully been transmitted.

8.2 MC13192 Operational Modes

The MC13192 has a number of operational modes that allow for low-current operation:

- **Off:** All IC functions Off, Leakage only. $\overline{RST}$ asserted. Digital outputs are tri-stated including $\overline{IRQ}$.
- **Hibernate:** Crystal Reference Oscillator Off. (SPI not functional.) IC responds to $\overline{ATTN}$. Data is retained.
- **Doze:** Crystal Reference Oscillator On but CLKO output available only if Register 7, Bit 9 = 1 for frequencies of 1 MHz or less. (SPI not functional.) Responds to $\overline{ATTN}$ and can be programmed to enter Idle Mode through an internal timer comparator.
- **Idle:** Crystal Reference Oscillator On with CLKO output available. SPI active.
- **Receive:** Crystal Reference Oscillator On. Receiver On.
- **Transmit:** Crystal Reference Oscillator On. Transmitter On.
- **CCA/ED:** Crystal Reference Oscillator On. Receiver On.

8.3 Connection with the MCU

The host microcontroller directs the MC13192, checks its status, and reads/writes data to the device through the 4-wire SPI port. The transceiver operates as a SPI slave device only and the host microcontroller supplies the interface clock and acts as SPI master. A transaction between the host and the MC13192 occurs as multiple 8-bit bursts on the SPI.

The SPI connections to the MCU include $\overline{CE}$, MOSI, MISO and SPICLK signals:

1. **Chip Enable $\overline{CE}$:** A transaction on the SPI port is framed by the active low $\overline{CE}$ input signal. A transaction is a minimum of 3 SPI bursts and can extend to a greater number of bursts.

2. **SPI Clock (SPICLK):** The host drives the SPICLK input to the MC13192. Data is clocked into the master or slave on the leading (rising) edge of the return-to-zero SPICLK and data out changes state on the trailing (falling) edge of SPICLK.
3. **Master Out/Slave In (MOSI):** Incoming data from the host is presented on the MOSI input.
4. **Master In/Slave Out (MISO):** The MC13192 presents data to the master on the MISO output.

A typical interconnection to a microcontroller is shown in Figure 8.2.

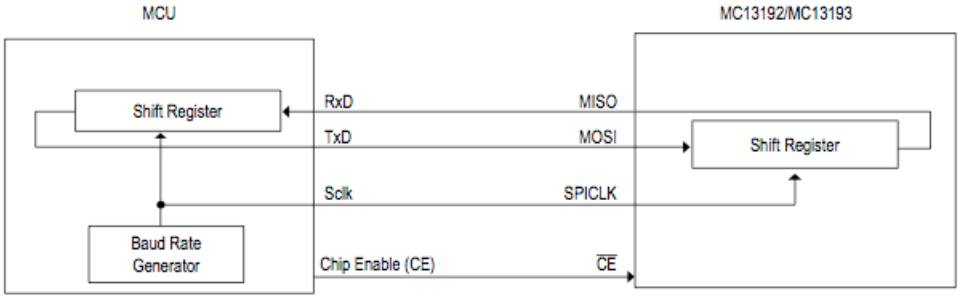


Figure 8.2: SPI Interface

The SPI can run at a frequency of 8 MHz or less. Although the SPI port is fully static, internal memory, timer and interrupt arbiters require an internal clock (CLKcore), derived from the crystal reference oscillator (16 MHz), to communicate from the SPI registers to internal registers and memory.

8.4 Antenna

The MC13192 has differential RF inputs and outputs that are well suited to balanced printed wire antenna structures. Alternatively, a printed wire antenna, a chip antenna, or other single-ended structures can be used with commercially available chip baluns or microstrip equivalents. In this project we used two printed wire antenna structures, one for transmitting and the other one for receiving. For the receive side, the RX antenna is ac-coupled to the differential RFIN inputs. For the transmit side, the TX antenna is connected to the differential PAO outputs. This dual antennae approach is a low cost option as the antennae can be printed wire devices. (See Appendix A)

8.5 Other signals

Other important pins of the transceiver are described below along with their connections with the MCU (See Appendix A):

- The RFIN+ and RFIN- are receive inputs only.
- PAO+ and PAO- are the differential PA output pins. These signals support a full differential dual port radio interface.
- The $\overline{ATTN}$ line can be driven by a GPIO from the MCU or can also be controlled by a switch or other hardware. The latter approach allows the MCU to be put into a sleep mode and then awakened by CLKO when the $\overline{ATTN}$ line wakes up the MC13192.
- RXTXEN is used to initiate receive, transmit or CCA/ED sequences under MCU control. RXTXEN must be controlled by an MCU GPIO.
- Device reset ($\overline{RST}$) is controlled through a connection to an MCU GPIO.
- The modem interrupt request $\overline{IRQ}$ is an active low open drain output that is asserted when an interrupt request is pending. Interrupts provide a way for the MC13192 to inform the host microcontroller (MCU) of onboard events without requiring the MCU to constantly query MC13192 status.
- Optionally, CLKO can provide a clock to the MCU (through the EXTAL and XTAL pins of the MCU). The CLKO frequency is programmable via the SPI and has a default of 32.786+ kHz (16 MHz / 488). [18] and [6]

Chapter 9

MCF5212 microcontroller

In order to develop the magnetic sensor node in this project, we used the MCF5212 microprocessor, which belongs to the MCF5213 family of microprocessors. The MCF5213 family (that also includes the MCF5211 and MCF5212) is a highly integrated implementation of the ColdFire family of Reduced Instruction Set Computing (RISC) microcontrollers. This 32-bit device is based on the Version 2 (V2) ColdFire Reduced Instruction Set Computing (RISC) core with a Multiply-Accumulate (MAC) Unit and hardware divider (DIV) providing 76 Dhrystone 2.1 MIPS at a frequency up to 66 MHz (for a 64 LQFP package) from internal Flash. It also offers high performance and low power consumption. On-chip memories connected tightly to the processor core include 256 Kbytes of Flash memory and 32 Kbytes of internal SRAM. The configuration of the MCF5212 is shown in Table 9.1 and described below.

Table 9.1: MCF5212 Configurations

Module	**5212**
ColdFire Version 2 Core with MAC (Multiply-Accumulate Unit)	
System Clock	66 MHz
Clock module with 8 MHz on-chip relaxation oscillator and integrated Phase Locked Loop (PLL)	8 MHz
Performance (Dhrystone 2.1 MIPS)	Up to 76
Flash / Static RAM (SRAM)	256/32 Kbytes
Interrupt Controller (INTC) capable of handling up to 63 interrupt sources	
Eight-channel 12-bit Fast Analog-to-Digital Converter (ADC)	
Four-channel, 32-bit Direct-Memory Access (DMA) controller	
16-bit periodic Programmable Interval Timer Module (PITs)	2
Four-Channel General-Purpose Timer (GPT) capable of input capture/output compare, Pulse Width Modulation (PWM), and pulse accumulation	3
Four-channel, 32-bit input capture/output compare timers with optional DMA support (DTIM)	4
Queued Serial Peripheral Interface (QSPI) module	
Universal Asynchronous/Synchronous Receiver/Transmitters (UARTs)	3
An Inter-integrated Circuit (I^2C) bus controller	
Eight-channel/Four-channel, 8-bit/16-bit Pulse Width Modulation Timer (PWM)	8
General Purpose I/O Module (GPIO)	
Chip Configuration and Reset Controller Module	
Background Debug Mode (BDM)	
JTAG IEEE 1149.1 Test Access Port	(Reduced Debug Interface)
Package	64 Leaded Quad Flat Package (LQFP)

9.1 Clock Module

The clock module allows the device to be configured for one of several clocking methods. Clocking modes include internal PLL clocking with either an external clock reference or an external crystal reference (8 MHz) supported by an internal crystal amplifier. The PLL can also be disabled and an external oscillator can be used to clock the device directly.

The clock module can be operated in:

- **Normal PLL mode:** This is the default mode of operation. In normal PLL mode, the PLL is fully programmable. It can synthesize frequencies ranging from 4x to 18x the reference frequency and has a post divider capable of reducing this synthesized frequency without disturbing the PLL. The PLL reference can be either a crystal oscillator or an external clock. (In normal PLL mode, $f_{sys} = f_{ref} * 2(\text{MFD} + 2)/(2\text{RFD})$ with f_{ref} = input reference frequency, f_{sys} = CLKOUT frequency, Multiplication Factor Divider (MFD) ranging from 0 to 7 and Reduced Frequency Divider (RFD) ranging from 0 to 7).
- **1:1 PLL mode:** In this mode, the PLL synthesizes a frequency equal to the external clock input reference frequency ($f_{sys} = f_{ref}$). The post divider is not active.
- **External clock mode (PLL disabled):** In external clock mode, the PLL is bypassed, and the external clock is applied to EXTAL. The resulting operating frequency is equal to the external clock frequency ($f_{sys} = f_{ref}$).

The clock module can operate in four different low-power modes:

- **The Wait mode:** In this mode, the system's clocks to the peripherals are enabled and the clocks to the CPU and SRAM are stopped (clocks sent to peripheral modules only).
- **The Doze mode:** Similar to the Wait mode.
- **The Stop mode:** In this mode, all system clocks are disabled. The PLL can be disabled in stop mode, but requires a wakeup period before it can relock. The oscillator can also be disabled during stop mode, but requires a wakeup period to restart. When the PLL is enabled in stop mode, the external CLKOUT signal can support systems using CLKOUT as the clock source.
- **The Halted mode:** This mode corresponds to the normal clock operation.

The clock module consists of five signals which are summarized in Table 9.2.

Table 9.2: Signal Properties

Name	Function
EXTAL	Oscillator or clock input
XTAL	Oscillator output
CLKOUT	System clock output
CLKMOD[1:0]	Clock mode select inputs
$\overline{\text{RSTO}}$	Reset signal from reset controller

In this project, an external crystal of 8 MHz is connected to EXTAL and XTAL pins, providing the reference frequency for the internal clock generation. The CLKOUT signal is the system clock output and is used to provide the clock for the BDM (TCLK/CLKOUT).

9.2 On-Chip Memories

9.2.1 Static RAM (SRAM)

The dual-ported Static RAM (SRAM) module provides a general-purpose 32-Kbyte memory block that the ColdFire core can access in a single cycle. The location of the memory block can be set to any 32-Kbyte boundary within the 4-Gbyte address space. This memory is ideal for storing critical code or data structures and for use as the system stack. Because the SRAM module is physically connected to the processor's high-speed local bus, it can quickly service core-initiated accesses or memory-referencing commands from the debug module. The SRAM module is also accessible by the DMA. The dual-ported nature of the SRAM makes it ideal for implementing applications with double-buffer schemes, where the processor and a DMA device operate in alternate regions of the SRAM to maximize system performance.

9.2.2 Flash

The ColdFire Flash Module (CFM) is a Non-Volatile Memory (NVM) module that connects to the processor's high-speed local bus. The CFM is constructed with four banks of 32K x 16-bit Flash arrays to generate 256 Kbytes of 32-bit Flash memory. These arrays serve as electrically erasable and programmable, non-volatile program and data memory. The Flash memory is ideal for program and

data storage for single-chip applications, allowing for field reprogramming without requiring an external high voltage source.

9.3 Power Management

The MCF5212 incorporates several low power modes of operation which are entered under program control and exited by several external trigger events. There are four possible power modes:

- The **Run mode** is the normal system operating mode. Current consumption in this mode is related directly to the system clock frequency.
- The **Wait mode** is intended to be used to stop only the CPU and memory clocks until a wakeup event is detected. In this mode, peripherals may be programmed to continue operating and can generate interrupts, which cause the CPU to exit from wait mode.
- The **Doze mode** affects the CPU in the same manner as wait mode, except that each peripheral defines individual operational characteristics in doze mode. Peripherals which continue to run and have the capability of producing interrupts may cause the CPU to exit the doze mode and return to run mode. Peripherals which are stopped will restart operation on exit from doze mode as defined for each peripheral.
- The **Stop mode** affects the CPU in the same manner as the wait and doze modes, except that all clocks to the system are stopped and the peripherals cease operation. Stop mode must be entered in a controlled manner to ensure that any current operation is properly terminated. When exiting stop mode, most peripherals retain their pre-stop status and resume operation.

Entry into either the wait, doze or stop mode idles the CPU with no cycles active, powers down the system and stops all internal clocks appropriately. A wakeup event is required to exit a low-power mode and return to run mode. Wakeup events consist of either any type of reset or any valid, enabled interrupt request.

Moreover, an integrated Power-On Reset (POR) circuit monitors the input supply and forces an MCU reset as the supply voltage rises. The Low Voltage Detector (LVD) monitors the supply voltage and is configurable to force a reset or interrupt condition if it falls below the LVD trip point. The RAM standby switch provides power to RAM when the supply voltage to the chip falls below the standby battery voltage.

9.4 Interrupt Controller (INTC)

The MCF5212 has a single interrupt controller that supports up to 63 interrupt sources. There are 56 programmable sources, 49 of which are assigned to unique peripheral interrupt requests. The remaining 7 sources are unassigned and may be used for software interrupt requests. The interrupt architecture of ColdFire has a 3-bit encoded interrupt priority level sent from the interrupt controller to the core, providing 7 levels of interrupt requests.

9.5 Edge Port (EPORT)

The Edge Port (EPORT) module has three external interrupt pins, $\overline{IRQ7}$, $\overline{IRQ4}$ and $\overline{IRQ1}$. Each pin can be configured individually as a level-sensitive interrupt pin, an edge-detecting interrupt pin (rising edge, falling edge, or both), or a general-purpose I/O pin.

In this project, only the $\overline{IRQ1}$ pin is used for its primary function (interrupt pin) and is connected to the $\overline{IRQ}$ pin of the transceiver so that the transceiver can send interrupt requests to the MCU. $\overline{IRQ4}$ and $\overline{IRQ7}$ are configured as General Purpose Input/Output signals and are linked to the GPIO signals of the transceiver.

9.6 Fast Analog-to-Digital Converter (ADC)

The Fast Analog-to-Digital Converter (ADC) consists of an eight-channel input select multiplexer and two independent Sample and Hold (S/H) circuits feeding separate 12-bit ADCs. The two separate converters store their results in accessible buffers for further processing. The converters share a common voltage reference and common digital control module.

The ADC consists of a cyclic, algorithmic architecture using two recursive sub-ranging sections. Each sub-ranging section resolves a single bit for each conversion clock, resulting in an overall conversion rate of two bits per clock cycle. Each sub-ranging section is designed to run at a maximum clock speed of 5.0 MHz. Thus a complete 12-bit conversion takes 6 ADC clocks (1.2 μs), not including sample or post processing time.

The ADC has two external clock inputs used to drive two clock domains within the ADC module:

- The peripheral clock (= system clock): it corresponds to half of the core clock (PLL output divided by 2 if PLL enabled or oscillator clock divided by 2 when PLL disabled.)
- The ADC 8 MHz Clock: External crystal oscillator of 8 MHz which provides 8 MHz for auto standby power saving mode.

The ADC supports five power modes. The power modes, in order of highest to lowest power utilization at the expense of increased conversion latency and/or startup delay, are:

- The **Normal power mode** which operates when at least one ADC converter is powered up, when both auto power-down and auto standby modes are disabled and when the ADC's clock is enabled. In this mode, the ADC uses the conversion clock as the ADC clock source both when active or idle. To minimize conversion latency, it is recommended the conversion clock be configured to 5.0 MHz. No startup delay is imposed.
- The **Auto power-down mode** which operates when at least one ADC converter is powered up, when auto power-down mode is enabled and when the ADC's clock is enabled. Auto power-down and standby modes can be used together. This hybrid mode converts at an ADC clock rate of 100 kHz using standby current mode when active, and gates off the ADC clock and powers down the converters when idle. A startup delay is executed at the start of all scans while the ADC engages the conversion clock and the ADC powers up, stabilizing in the standby current mode. This provides the lowest possible power configuration for ADC operation.
- The **Auto standby mode** which operates when at least one ADC converter is powered up, when auto power-down is disabled, when auto standby is enabled, when the ADC's clock is enabled and when either the relaxation oscillator is enabled for 8-MHz operation or the external oscillator clock runs at 8 MHz. In auto standby mode, the ADC uses the conversion clock when active and the 100 kHz standby clock when idle. The standby (low current) state automatically engages when the ADC is idle. The ADC will execute a startup delay at the start of all scans, allowing the ADC to switch to the Conversion clock and to revert from standby to normal current mode. It is recommended the conversion clock be configured at or near 5.0 MHz to minimize conversion latency when active.
- The **Standby mode**, similar to the auto standby mode. In this mode,

the ADC uses the conversion clock when active and gates off the conversion clock and powers down the converters when idle. A startup delay is executed at the start of all scans, allowing the ADC to stabilize when switching to normal current mode from a completely powered off condition. This mode uses less power than normal and more power than auto standby. It requires more startup latency than auto standby when leaving the idle state to start a scan.

- The **Power-down mode** which operates when both ADC converters are powered down and when the ADC's clock is disabled. In this configuration, the clock trees to the ADC and all of its analog components are shut down and the ADC uses no power.

In this project, the ADC signals are used to establish the communication between the MCU and the magnetic sensors in order to retrieve and convert the analog sensor's measurements into digital values for further process (to be sent to the transceiver). However, two of the ADC signals are configured as General Purpose I/O signals, since they are used for the Set/Reset (S/R) purposes of the magnetic sensors.

9.7 Direct-Memory Access (DMA) Controller

The Direct-Memory Access (DMA) controller provides an efficient way to move blocks of data with minimal processor intervention. It has four channels that allow byte, word, longword, or 16-byte burst line transfers. These transfers are triggered by software or by the occurrence of certain UART or DMA timer events. Each channel supports dual-address transfers. Dual-address transfers consist of a source data read and a destination data write. The DMA controller module begins a dual-address transfer sequence during a DMA request. Figure 9.1 illustrates the dual-address transfer.

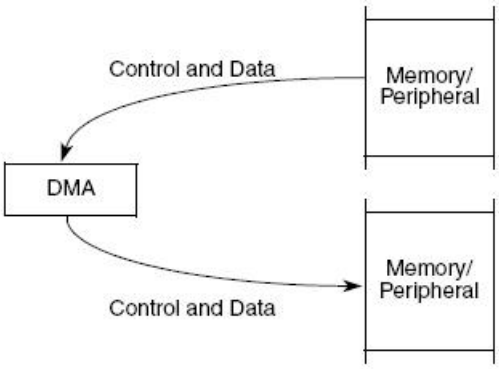


Figure 9.1: Dual–Address Transfer

9.8 DMA Timers (DTIM0-DTIM3)

There are four independent, DMA transfer capable 32-bit timers (DTIM0, DTIM1, DTIM2, and DTIM3) on the MCF5212. Each module incorporates a 32-bit timer with a separate register set for configuration and control. The timers can be configured to operate from the system clock or from an external clock source using one of the DTINx signals. If the system clock is selected, it can be divided by 16 or 1. The input clock is further divided by a user-programmable 8-bit prescaler which clocks the actual timer counter register. Each of these timers can be configured for input capture or reference (output) compare mode. Timer events may optionally cause interrupt requests or DMA transfers.

In this project, only one timer is used for general timing purposes so that the DTIM signals of the three other timers are configured as General Purpose I/O signals and are used for the S/R purposes of the magnetic sensors.

The formula that determines time-out periods (T) for various reference values is:

$$T = (\frac{1}{CLF}) * (1\text{or}16) * (\text{DTMR}n[\text{PS}] + 1) * (\text{DTRR}n[\text{REF}] + 1) \quad (9.1)$$

where CLF is the clock frequency, the DTMRn[PS] value yields the prescaler value (DTMRn[PS] = 0x00 yields a prescaler of 1, and DTMRn[PS] = 0xFF yields a prescaler of 256) and DTRRn[REF] is the reference value.

9.9 Queued Serial Peripheral Interface (QSPI)

The MCF5212 also has a Queued Serial Peripheral Interface (QSPI). This module provides a synchronous serial peripheral interface with queued transfer capability. It allows up to 16 (pre-programmed) transfers to be queued at once, eliminating CPU intervention between transfers. There are two transfer length options. The user can choose a value of 8 to 16 bits and the data is transferred with the Most Significant bit (MSb) first. Data transfer is synchronized with the internally generated QSPI_CLK clock. The maximum QSPI clock frequency is one-fourth the clock frequency of the internal bus clock (f_{sys} = 66 MHz so 12.9 kHz $\leq$ QSPI_CLK $\leq$ 16.5 MHz). The QSPI module only operates in master mode, so in every case this module should be configured to operate in master mode in order to function properly. Otherwise, the QSPI activity will be indeterminate. The QSPI can initiate serial transfers but cannot respond to transfers initiated by other QSPI masters. Transfer RAM in the QSPI is indirectly accessible using address and data registers. The user initiates QSPI operation by loading a queue of commands in command RAM, writing transmit data into transmit RAM, and then enabling the QSPI data transfer.

The QSPI uses a dedicated 80-byte block of static RAM accessible to both the module and CPU to perform queued operations. This RAM does not appear in the device memory map, because it can only be accessed by the user indirectly (through QSPI registers), as 48 separate locations that comprise 16 words of transmit data, 16 words of receive data, and 16 bytes of commands. The RAM is divided into three segments with 16 addresses each:

- Receive data RAM (32 receive data bytes), the initial destination for all incoming data.
- Transmit data RAM (32 transmit data bytes), a buffer for all out-bound data.
- Command RAM (16 command control bytes), where commands are loaded.

The RAM is organized so that 1 byte of command control data, 1 word of transmit data, and 1 word of receive data comprise 1 of the 16 queue entries. The module has four signals:

- QSPI Data Output (QSPI_DOUT): Serial data output from QSPI.
- QSPI Data Input (QSPI_DIN): Serial data input to QSPI.
- Serial Clock (QSPI_CLK): Clock output from QSPI.
- Peripheral Chip Selects (QSPI_CS0).

In this project, the QSPI module is used to establish the communication between the MCU and the MC13192 transceiver. The QSPI_DOUT and the QSPI_DIN pins are linked with the Master Out Slave In (MOSI) and the Master In Slave Out (MISO) pins of the transceiver, respectively. They assure the data transfer between the two units. The QSPI_CLK is connected to the SPICLK of the transceiver to synchronize the data transfer. The QSPI_CLK is equal to 1/4 of the internal bus clock. Finally, the QSPI_CS0 is configured as a General Purpose I/O signal and is linked to the nCE pin of the transceiver to enable SPI transfers. (For a detailed description of the communication between the MCU and the transceiver, see section 11.5 on page 77.)

9.10 UARTs

The MCF5212 has two full-duplex Universal Asynchronous/Synchronous Receivers/ Transmitters (UARTs) that function independently. The two UARTs can be clocked by the system bus clock, eliminating the need for an external clock source. Each UART module interfaces directly to the CPU and consists of a serial communication channel, programmable clock generation, interrupt control logic and DMA request logic and internal channel control logic.

The four UART module signals are:

- Transmitter Serial Data Output (UnTXD)
- Receiver Serial Data Input (UnRXD)
- Clear-to-Send $\overline{UnCTS}$: This input can generate an interrupt on a change of state.
- Request-to-Send $\overline{UnRTS}$: This output can be programmed to be negated or asserted automatically by either the receiver or the transmitter. When connected to a transmitter's $\overline{UnCTS}$, $\overline{UnRTS}$ can control serial data flow.

The transmitter converts parallel data from the CPU to a serial bit stream, inserting appropriate start, stop, and parity bits. It outputs the resulting stream on the transmitter serial data output (UnTXD).

The receiver converts serial data from the receiver serial data input (UnRXD) to parallel format, checks for a start, stop, and parity bits, or break conditions, and transfers the assembled character onto the bus during read operations. The receiver may be polled, interrupt driven, or use DMA requests for servicing.

Figure 9.2 shows a signal configuration for a UART/RS-232 interface.

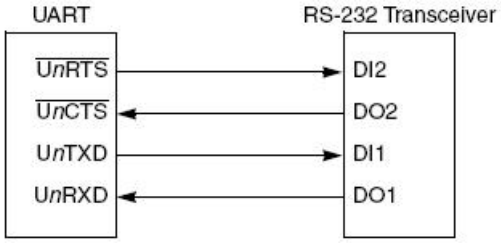


Figure 9.2: UART/RS-232 Interface

In the project, UTXD0 and URXD0 are connected to the transmitting and receiving line of the serial port, respectively. They are used for their primary function (UART signals), as described above. However, all the other UART module signals are used as General Purpose I/O signals.

9.11 Inter-integrated Circuit (I^2C) Bus

The MCF5212 also has an Inter-integrated Circuit (I^2C) Interface. I^2C is a two-wire, bidirectional serial bus that provides a simple, efficient method of data exchange, minimizing the interconnection between devices. This bus is suitable for applications that require occasional communication between many devices over a short distance. The flexible I^2C bus allows additional devices to be connected to the bus for expansion and system development.

The interface is designed to operate up to 100 Kbps with maximum bus loading and timing. The device is capable of operating at higher baud rates, up to a maximum of the internal bus clock divided by 20, with reduced bus loading. The maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF.

The I^2C module uses a serial data line (I^2C_SDA) and a serial clock line (I^2C_SCL) for data transfer. These pins are connected to the corresponding pins (SDA and SCL pins, respectively) of the temperature sensor so as to retrieve the information from the sensor (temperature measurements). Normally, a standard communication is composed of four parts: START signal, slave address transmission, data transfer, and STOP signal. (For a detailed description of the communication between the MCU and the sensor, see section 11.4 on page 74.)

9.12 General Purpose I/O (GPIO)

The MCF5212 also has a General Purpose I/O (GPIO) Interface. Most of the pins on the MCF5212 that are associated with the external interface may have general purpose I/O capability, when they are not used for their primary function. The MCF5212 ports module includes these distinctive features: Control of primary function use on all ports and Digital I/O support for all ports; registers for: storing output pin data, controlling pin data direction, reading current pin state, and setting and clearing output pin data registers. The digital I/O pins are grouped into 8-bit ports. Some ports do not use all 8 bits. The MCF5212 ports module controls the configuration for the following external pins: External bus accesses, Chip selects, Debug data, Processor status, I^2C serial control, QSPI, UART transmit/receive, 32-bit DMA timers. A block diagram of the MCF5212 ports is shown in Figure 9.3.

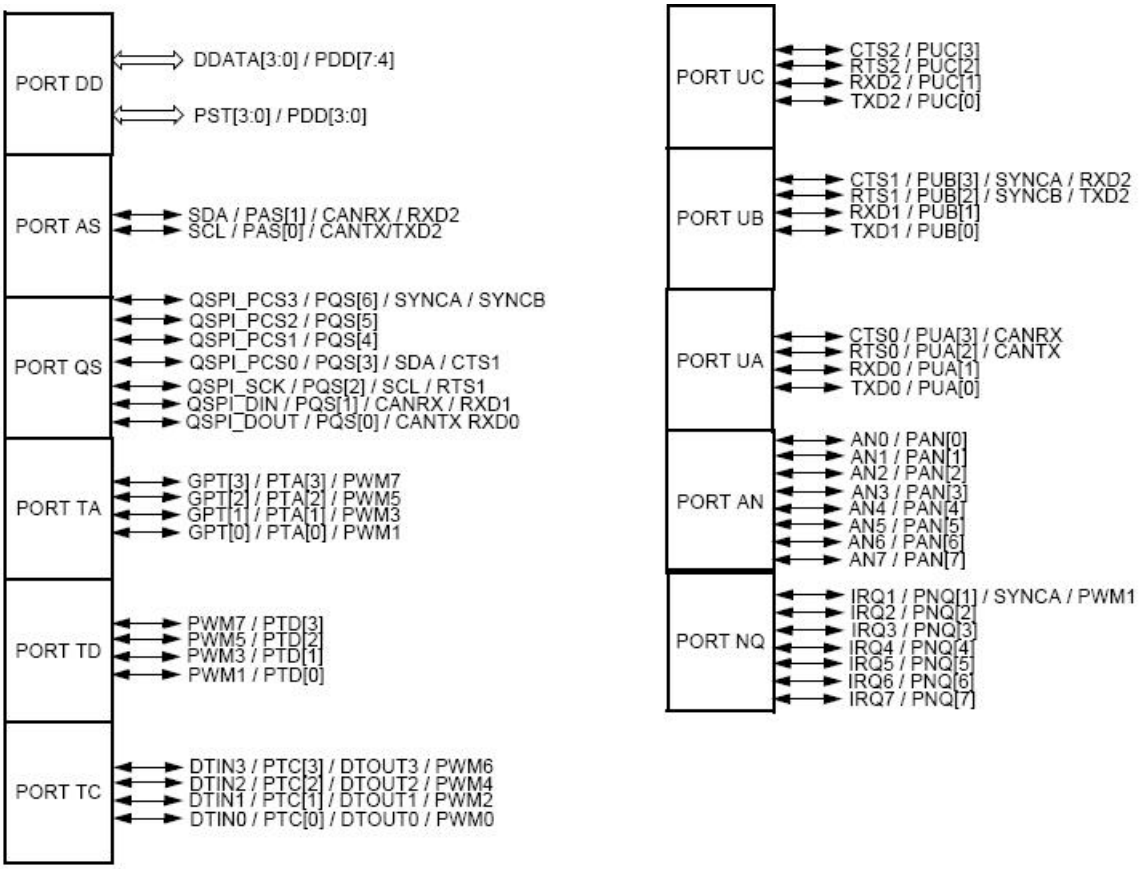


Figure 9.3: General Purpose I/O (GPIO) Module Block Diagram

9.13 Reset

The reset controller is provided to determine the cause of reset, assert the appropriate reset signals to the system, and keep a history of what caused the reset. There are seven sources of reset:

- External reset input
- Power-On Reset (POR)
- Watchdog timer
- Phase-Locked Loop (PLL) loss of lock
- PLL loss of clock

- Software
- Low Voltage Detector (LVD)

There are two reset controller signals: an output signal $\overline{RSTO}$ and an input signal $\overline{RSTI}$.

- **$\overline{\text{RSTO}}$**: This active-low output signal is driven low when the internal reset controller module resets the chip. When $\overline{RSTO}$ is active, the user can drive override options on the data bus.
- **$\overline{\text{RSTI}}$**: Asserting the external $\overline{RSTI}$ for at least four rising CLKOUT edges causes the external reset request to be recognized and latched.

The nRSTI signal is used, in this project, to reset the MCU and the BDM. The pin is linked with a RESET switch on the board as well as with the nRSTI pin of the BDM.

9.14 Integrated Debug Module

The ColdFire processor core debug interface is provided to support system debugging in conjunction with low-cost debug and emulator development tools. Through a standard debug interface, users can access debug information, without the need for costly in-circuit emulators.

The ColdFire family allows the implementation of either a low-level system debugger, Background Debug Mode (BDM), or of a Joint Test Action Group (JTAG), which is a dedicated user-accessible test logic that complies with the IEEE 1149.1 standard for boundary-scan testability. In this project, the BDM has been implemented in the microprocessor in a dedicated hardware module. Communication with the development system is handled through a dedicated, high-speed serial command interface. The debug module implements a synchronous serial protocol using two inputs (DSCLK and DSI) and one output (DSO). The development system serves as the serial communication channel master and must generate the clock DSCLK. The serial channel operates at a frequency from DC to 1/5 of the PSTCLK frequency. The channel uses full-duplex mode, where data is sent and received simultaneously by both master and slave devices. [9]

Part V

Ordering the board

At this stage of the project, the schematics and the PCB have been designed, so that the board was ordered to a company expert in board mounting. The final board is presented in Appendix C.

Part VI

Software

Now that the board has been designed and ordered, the second main part of this project is to design the software in order to create the wireless network consisting of all the boards as well as to assure the communication between the different components on each board. The software has been designed in C language within the FreeScale Metrowerks CodeWarrior for ColdFire IDE.

First of all, we need to find a protocol describing the communication between the different boards, so as to create the network.

Our protocol is based on the SMAC provided by Freescale to complement the MC13192 hardware and to implement the Physical and MAC layers of the network stack. The SMAC provides simple proprietary wireless connectivity and can support point-to-point and star network configurations. Then, we implemented the network layer, supporting a star networking topology in which one device acts as the central node of communication (PAN coordinator) to which all the other devices procure sensing information.

Chapter 10

Networking Topology

10.1 Freescale SMAC

The Freescale SMAC is a simple American National Standards Institute (ANSI) C based code stack available as sample source code. SMAC is used for developing proprietary RF transceiver applications using a Freescale 802.15.4 transceiver (MC1319x, MC1320x, MC1321x). SMAC was built to work with any HCS08 MCU with an SPI, but it can easily be adapted to almost any processor core. [19]

10.1.1 Software Architecture

The SMAC architecture includes the MCU and modem (transceiver) drivers as well as the physical and MAC layers. The SMAC block diagram is shown in Figure 10.1.

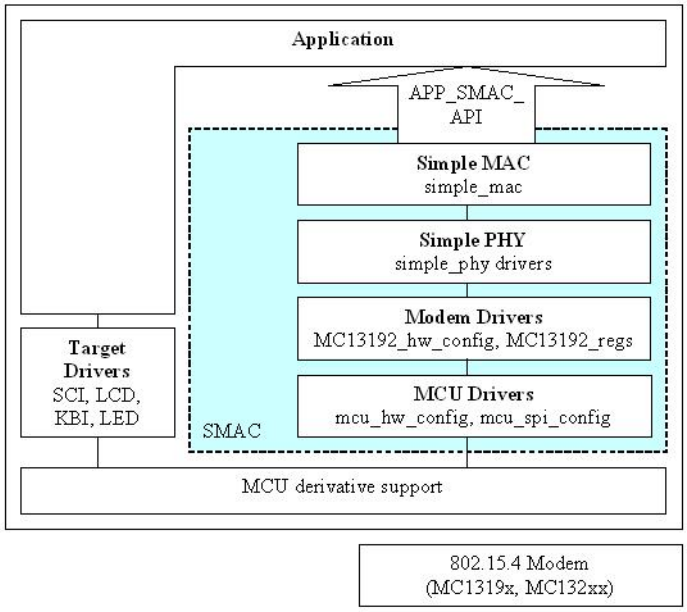


Figure 10.1: SMAC Block Diagram

The MCU Drivers include "mcu_hw_config" and "mcu_spi_config" which contain drivers for the GPIO pins.

The Modem Drivers include "MC13192_regs", which contains the MC13192 transceiver register map, and "MC13192_hw_config" which contains the MC13192 transceiver hardware interconnections.

The Simple PHY includes "drivers", which defines all externals, prototypes and MC13192 status mask bits used by the actual C driver, and "simple_phy", which is the physical layer file for the MCU and MC13192 transceiver. The SMAC PHY is the second lowest layer of C code.

The Simple MAC includes "simple_mac", which is the MAC layer file for the MCU and MC13192 transceiver. The SMAC MAC is the highest layer of C code for the SMAC.

Finally, an Application Programming Interface (API) is implemented in the SMAC as a C header file (.h) and makes references to the required functions within the SMAC and provides the application with SMAC functionality. [19]

10.1.2 SMAC Primitives

Table 10.1 and Table 10.2 provide a detailed description of SMAC primitives associated with the SMAC Application API. Table 10.1 describes the primitives handled by the simple PHY layer while Table 10.2 describes the ones handled by the simple MAC layer.

Table 10.1: SMAC primitives handled by the simple PHY layer

PHY Primitive	**Use**
PDDataRequest	Transmits data packet
PDDataIndication	Receives data packet indication
PLMEHibernateRequest	Hibernates the MC13192 (very low current, no CLKO)
PLMEDozeRequest	Dozes the MC13192 (Low current, CLKO $\leq$ 1MHz)
PLMEWakeRequest	Wakes the MC13192 from Hibernate or Doze
PLMESetChannelRequest	Sets the MC13192 operating channel
PLMESetTrxStateRequest	Sets the MC13192 transceive operation
PLMEEnergyDetect	Measures channel energy
PLMELinkQuality	Reports energy from last successful RX packet
PLMEGetTimeRequest	Gets MC13192 timer value
PLMESetMC13192ClockRate	Sets MC13192 CLKo frequency
PLMESetMC13192TmrPrescale	Sets MC13192 timer frequency
PLMESetTimeRequest	Sets MC13192 timer value (i.e. initialize)
PLMEEnableMC13192Timer1	Sets MC13192 timer compare value
PLMEDisableMC13192Timer1	Disables MC13192 timer comparator TC1
PLMEMC13192ResetIndication	Indicates a MC13192 reset condition
PLMEMC13192SoftReset	Forces the MC13192 into a soft reset condition
PLMEMC13192XtalAdjust	Adjusts the MC13192s crystal trim value
PLMEMC13192FEGainAdjust	Adjusts the MC13192s gain compensator
PLMEMC13192PAOutputAdjust	Adjusts the MC13192s Output power
PLMEGetRficVersion	Returns the RFIC version number
PLMELoadPRBS9	Loads the transmit RAM with a PRBS9 data pattern

Table 10.2: SMAC primitives handled by the simple MAC layer

MAC Primitive	Use
MCPSDataRequest	Used to send a packet
MLMERXEnableRequest	Places the radio into receive mode on the channel selected by MLMESetChannelRequest
MLMERXDisableRequest	Returns the radio to idle mode from receive mode
MLMEHibernateRequest	Places the radio into Hibernate mode
MLMEDozeRequest	Allows the user to put the radio either in Normal Doze Mode (without CLKO but with automatic wakeup) or Acoma Mode (with CLKout, but without timeout)
MLMEWakeRequest	Brings the radio out of low power mode
MLMESetChannelRequest	Sets the actual frequency that the radio transmits and receives on
MLMESetMC13192ClockRate	Sets the desired clock out from radio
MLMESetMC13192TmrPrescale	Changes the rate at which the radio timers operate
MLMEMC13192XtalAdjust	Adjusts the radio reference clock by a trim value
MLMEEnergyDetect	Starts an energy detect (ED)/ Clear Channel Assessment (CCA) cycle and returns the energy value (-power/2) via the returned argument. For example, if the Energy Detect returns 80 then the interpreted value is -80/2 or -40 dBm.
MLMEMC13192SoftReset	Performs a soft reset to the radio
MLMELinkQuality	Returns an integer value that is the link quality from the last received packet of the form: dBm = (-Link Quality/2)
MLMEMC13192FEGainAdjust	Compensates for the energy detection and Automatic Gain Control (AGC). The default value should be placed into the MC13192Init. But in the event users need to calibrate the readings due to a specific application like an external low noise amplifier, this is where users would set the offset.
MLMEMC13192PAOutputAdjust	Adjusts the output power of the transmitter
MLMEGetRficVersion	Reads the version number of the radio
MLMETestMode	By employing this function, users can execute a test of the radio. Some basic test modes are necessary to assist SMAC users evaluate their hardware.

10.2 Upper Layers

Since we have the PHY and MAC layers, we now need to implement the upper layers as to complete the network design. As mentioned above, we implemented the network layer with a star topology, in which one device acts as the central node of communication (coordinator) to which all the other devices procure sensing information. The coordinator is the only one capable of communicating with the other devices on the network and is referred to as a FFD. The other devices in the network are referred to as Reduced-Function Devices (RFDs) (RFDs: low-cost and small memory microcontrollers) and they are not able to communicate with any other node except for the coordinator device. In our case, since we designed only one board, we use the same board for the FFD and RFD devices. However, it is possible that, in the future, the FFD can be replaced with another board without sensors and with a more powerful microcontroller (larger memory space).

Our network was designed so that it can support up to 20 RFDs, a limitation implied by the memory space of each RFD. Each RFD device operating on the network has its own address (2 bytes), so that it can be distinguished from the other RFDs, and the FFD has a list containing all the RFDs' addresses. Any communication between the FFD and the RFDs is only initiated by the FFD. The FFD contacts all the RFDs, one after the other, in a loop, in order to get the sensor data provided by each sensor node (RFD).

The communication protocol between the FFD and one RFD is described below:

- The FFD broadcasts a known data sequence to all RFDs. This sequence, composed of 1 byte of Packet Number, 2 bytes of PAN Id and 2 bytes containing the Destination Address of the specific RFD that the coordinator wants to contact and finally 4 predefined bytes ("SEND"), is known by all the RFDs as a sequence indicating that the FFD is ready to receive the sensor data of the specified RFD. This sequence will be referred to as the Start_TX sequence (Figure 10.2) in the rest of the report.

Figure 10.2: Packet Structure of FFD's Start_TX sequence

- When a RFD receives data, it checks if it is a Start_TX sequence. For this, it first checks if the PAN Id and Destination Address in the packet correspond to the Id of the PAN it belongs to and to its own address respectively. In the case they do not match, the RFD does not pursue further actions. However, if they match, the RFD further checks if the four following bytes are equal to the sequence "SEND". If they do not match, it means that the sequence it receives is noise and no further action is taken. However, if the four next bytes are equal to "SEND", it means that the RFD received a Start_TX sequence containing its address so that it begins transmitting sensor data to the FFD. However, in the case that the RFD does not have any available sensor data to transmit, it sends back to the FFD a packet containing the sequence "EMPTY" as payload.
- The RFD transmits a packet of 123 bytes, containing a payload of 110 bytes. This payload consists of 110 bytes of sensor data (108 bytes of magnetic sensor data and 2 bytes of temperature sensor data). The transmission rate of the MC13192 transceiver is 250 kbps, meaning that we can transmit 123 bytes (one packet) in around 4ms. However, according to a more realistic scenario and taking into account various processing delays (for example, processing the sensor data from the RFD's MCU to the RFD's transceiver), we considered that the effective transmission rate is only 24 kbps. Therefore, the time required to transmit one packet (123 data bytes) is equal to 41 ms. The protocol has been designed in such a way that each RFD transmits only one packet to the FFD whenever the RFD receives the Start_TX sequence.
- When the FFD has sent a Start_TX sequence with a specific RFD address, it waits for 41 ms before sending a new Start_TX sequence with the next RFD address in its list. The FFD waits for 41 ms due to the fact that this time corresponds to the time needed for the RFD to transmit one data packet, as mentioned above. During this time, the FFD either receives one packet of data from the corresponding RFD, that is further checked and considered as correct or erroneous packet or does not receive any packet. Therefore the expected packet is considered as a lost packet. If the packet is considered correct, it is printed on a Personal Computer (PC) screen through the serial port RS232.

Therefore the FFD collects the sensor data of all RFDs existing in the network after $41 * 20 = 820$ ms.

Chapter 11

Communication on a board

11.1 Reduced Function Device (RFD) Board

The software design for the RFD board consists of establishing the communication between the MCU and the other components: the magnetic and temperature sensors as well as the transceiver.

The magnetic sensors (1, 2 and 3-axis) of the RFD are sensing continuously the traffic. We need to stock the sensor measurements until the RFD is able to send them to the FFD. Therefore we have 10 buffers of 108 bytes each, which are filled one after the other with the sensor measurements as they are collected by the sensors. Each buffer contains 9 values of every axis measurement (2-byte measurement) of each sensor. Since all three sensors get 6 measurements at a time, each value in the buffers has a total length of 12 bytes ($= 6 * 2$ bytes) and therefore each buffer contains 108 bytes ($= 12 * 9$).

Since the magnetic sensors outputs are linked to the analog-to-digital converter of the MCU, we stock into the buffers the digital outputs of the ADC. The protocol of communication between the magnetic sensors and the MCU is described in section 11.3 on page 70. The digital outputs of the ADC (samples) are stocked into buffers in the MCU at a rate of 100 Hz. We chose this rate because it is a good trade-off between memory space in the MCU and the number of samples needed for accurate traffic estimation: for example, lets consider the car "SMART", which has a length of 2.5 meters. If the car runs at a speed of 130 km/h ($= 36$ m/s), the sensor will see the car during 0.07 s ($= \frac{2.5m}{36m/s}$) at least. If the sensors sample at 100 Hz (0.01 s), the sensors will get at least 7 samples of the car. This number of samples is good enough to detect accurately the car without overflowing the

memory space of the MCU.

The RFD is continuously filling the buffers with the sensors measurements. The buffers are periodically emptied when being sent to the FFD so that the RFD can refill them with new sensor measurements. Therefore, 10 buffers provide enough memory space to stock the sensor measurements from the moment they are captured to the moment they are transmitted, without losing information.

When the RFD receives a Start_TX sequence from the FFD, it has to transmit back one packet (Figure 11.1) with sensor measurements. For this purpose, it creates a packet consisting of 1 byte of the Packet Number, 2 bytes with the PAN Id, 2 bytes indicating the Source Address, 108 bytes of magnetic sensor measurements and 2 bytes of temperature sensor measurements.

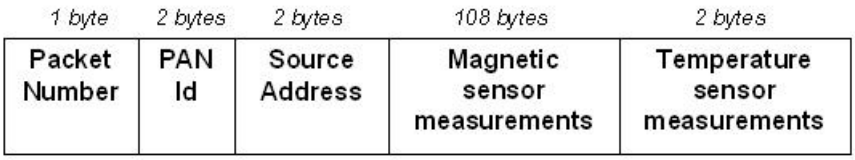


Figure 11.1: RFD Packet Structure

- As mentioned before, the magnetic sensor measurements transmitted within one packet correspond to the values of one entire buffer. Once the values of one buffer have been transmitted, the RFD can overwrite these values with new sensor measurements. The 10 buffers act as one First-In First-Out (FIFO) buffer, so that the buffers are transmitted in the chronological order in which they were filled.
- Finally, we transmit 2 bytes of temperature sensor measurements corresponding to one temperature value. For this purpose, we trigger once the temperature sensor when creating a packet. For a detailed description of the way the temperature sensor interfaces with the MCU see section section 11.4 on page 74. Since, we consider that the temperature does not change between the moment the buffer is filled and the moment it is sent (less than 1s elapses between those two moments), we trigger the temperature sensor only when creating the packet and not for every magnetic sensor measurement.

Finally, when the packet is created, the RFD directly sends it to the FFD. For this purpose, the MCU communicates the packet to the transceiver, so that it can be sent to the FFD. For a detailed description of the way the transceiver interfaces with the MCU see section 11.5 on page 77.

11.2 Full-Function Device (FFD) Board

The software design for the FFD board consists of establishing the communication between the MCU and the transceiver. The FFD board only transmits a sequence to each RFD, indicating that the RFD can transmit its sensors data, and receives these data. Therefore, the only communication on the board occurs between the MCU and the transceiver. For a detailed description of the way the transceiver interfaces with the MCU see section 11.5 on page 77.

11.3 Communication between the Magnetic Sensors and the Microcontroller MCF5212

In this part, the communication between the magnetic sensors and the microcontroller MCF5212 is described. As mentioned before, the ADC signals are used to establish the communication between the MCU and the magnetic sensors in order to retrieve and convert the analog sensors measurements into digital values for further process (to be sent to the transceiver). Although the ADC has 8 input signals, only 6 of them are used to get the measurements of the magnetic sensors. This is done because, as mentioned before, we included on the same board three different magnetic sensors (a 1-axis, a 2-axis and a 3-axis sensor), that provide 6 different measurements in total. As mentioned in section 9.6 on page 47, the Fast ADC consists of two separate 12-bit analog-to-digital converters, which store their results in accessible buffers for further processing.

"The different conversion types and scanning modes of the ADC are described below, along with the ADC configuration chosen for this project.

Conversion types: The inputs of the ADC can be configured for either single-ended or differential conversions:

- **The Single-ended mode:** The ADC measures the voltage of the selected analog input and compares it against the (VREFH - VREFL) reference voltage range.
- **The Differential mode:** In differential mode, the ADC measures the voltage difference between two analog inputs and compares that against the (VREFH - VREFL) voltage range.

In this project the single-ended conversion mode was selected.

Scanning modes: The ADC can be configured to perform a single scan and halt

(once mode), perform a scan whenever triggered (triggered mode), or perform a programmed scan sequence repeatedly until manually stopped (loop mode: they automatically restart a scan as soon as the previous scan completes). The single scan (once mode) differs from the triggered mode only in that it must be re-armed after each use. This arming can occur anytime, even while the scan it initiated is still in process.

In either case the scanning mode can either be sequential scan mode or parallel scan mode, which can be simultaneous or non-simultaneous.

In *sequential scans*, up to 8 SAMPLE slots are sampled one at a time in the order SAMPLE 0-7 corresponding to any of the 8 analog inputs (AN0-7). A scan ends when the first disabled sample slot is encountered.

Parallel scans differ in that converter A collects up to 4 samples (SAMPLE 0-3) in parallel to converter B collecting up to 4 samples (SAMPLE 4-7). During *simultaneous scan*, the scans in the two converters are done simultaneously and always result in simultaneous pairs of conversions, one by converter A and one by converter B. In *non-simultaneous scan*, the parallel scans in the two converters are achieved independently.

Therefore, the following different scan modes can exist:

- **Once sequential:** When the ADCs conversion process is initiated, samples are taken one at a time starting with SAMPLE0 until a first disabled sample is encountered. If no disabled sample is encountered, conversion concludes after SAMPLE7.
- **Once parallel:** When the ADCs conversion process is initiated, converter A will capture samples 0-3 and converter B will capture samples 4-7. By default, samples are taken simultaneously (synchronously), and scanning stops when either converter encounters a disabled sample or both converters complete all 4 samples. When in non–simultaneous mode, samples are taken asynchronously, and scanning stops when each converter encounters a disabled sample or completes all 4 samples.
- **Loop sequential:** In the loop sequential mode, up to 8 samples are captured in each loop, and the next scan starts immediately after the completion of the previous scan. The process repeats until the user decides to end it.
- **Loop parallel:** In loop parallel scan modes, both converters restart together if in simultaneous mode and restart independently if in non-simultaneous mode. The process repeats until the user decides to end it.
- **Triggered sequential:** Similar to once sequential except that re-arming is

not necessary after each use.

- **Triggered parallel (default):** Similar to once parallel except that re-arming is not necessary after each use.

In this project, the once parallel mode was selected." [9]

In our design two of the ADC signals are configured as GPIO signals and they are used for the S/R purposes of the magnetic sensors. However, one more signal is needed for generating the S/R pulses, therefore a DMA Timer (DTIM) signal is configured as a GPIO signal, as mentioned above. "Set" and "reset" pulses are defined as pulsed currents that enter the positive (S/R+) and negative (S/R−) pins of the set/reset strap of the sensors, respectively. The "Set" and "Reset" pulses are generated from the MCU by setting the pins of the MCU linked with the S/R pins of the sensors to high (1) and low (0) alternatively with a period of 0.02 s. However, the pulses are converted into dampened exponential pulse waveforms (alternating between a maximum positive and negative value, $\pm V$), since a dual Metal Oxide Semiconductor Field Effect Transistor (MOSFET) (N-channel and P-channel), followed by a capacitor and a resistance, is connected in between the MCU pins and the S/R sensors' pins.

The magnetic sensor measurements are taken just after (10 $\mu seconds$ after) a set or reset pulse. The S/R is used among others for bridge offset reduction. The method used in this project in order to cancel the bridge offset is the Digital Subtraction method, as mentioned above.

11.3.1 Digital Subtraction method

This method is the most popular as it requires no change in hardware. There are at least two ways to come up with the subtraction of the bridge offset:

The first one would be to determine the natural bridge offset at the factory with no magnetic field applied, by testing in a shielded fixture or helmholtz coil fixture to counter the earth's field.

The second method is based on the following idea:

An offset value should be subtracted from or added to every sensor measurement value:

$$sc[n] = \begin{cases} s[n] - of[n], & \text{set} \\ -s[n] + of[n], & \text{reset} \end{cases} \tag{11.1}$$

Where $sc[n]$ corresponds to the final value of each sensor measurement (bridge offset compensated), $s[n]$ corresponds to each magnetic sensor measurement sample and $of[n]$ corresponds to each bridge offset value.

The bridge offset value is found by applying a low-pass filter to the sensor measurements following the equation below:

$$of[n] = c * of[n-1] + (1-c) * o[n] \tag{11.2}$$

Where c is a constant value ($0 \leq c \leq 1$) and $o[n]$ corresponds to the following equation:

$$o[n] = \frac{s'[n-1] + s[n]}{2} \tag{11.3}$$

where $s'[n-1]$ corresponds to each magnetic sensor measurement that is taken 10 $\mu seconds$ before applying every set or reset pulse and is used just for canceling the bridge offset and $s[n]$ corresponds to each magnetic sensor measurement that is taken 10 $\mu seconds$ after applying every set or reset pulse. The Digital Subtraction method is shown in Figure 11.2.

Although, the bridge offset voltages theoretically stay the same over the whole useful life of the sensors, thermal drifts can change the bridge offset value so that the previous procedure to compensate for the bridge offset is repeated periodically.

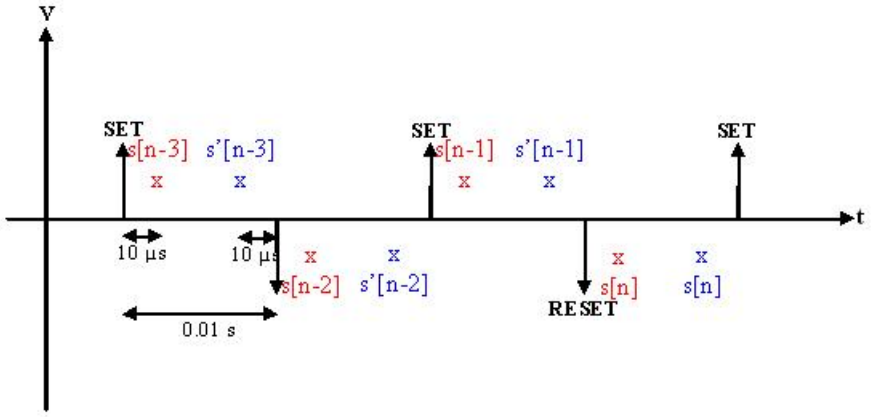


Figure 11.2: Bridge offset cancellation: Digital Subtraction method

11.4 Communication between the Temperature Sensor DS1631 and the Microcontroller MCF5212 through the I^2C Interface

In this part, the communication between the temperature sensor DS1631 and the microcontroller MCF5212 is described. As mentioned before, the DS1631 communicates with the MCU over a bidirectional 2-wire serial data bus (I^2C interface) that consists of a SCL signal and SDA signal.

"The DS1631 can be programmed to perform continuous consecutive conversions (continuous-conversion mode) or to perform single conversions on command (one-shot mode). The conversions (either consecutive or single) are initiated by a Start Conversions command. Consecutive conversions continue to be performed until a Stop Conversions command is issued, at which time the device goes into a low-power idle state. In the case of the one-shot mode, when the conversion is complete the device enters a low-power idle state immediately. The DS1631 always powers-up in a low-power idle state. In this project the one-shot mode was used, since one temperature measurement was required whenever magnetic sensor data was to be transmitted.

In order to initiate 2-wire communication, the microcontroller (master) generates a START condition signal (by pulling SDA from high to low while SCL is high) followed by a control byte containing the DS1631 slave address (in case that more than one temperature sensors share the bus) and indicating that the master next

writes a command byte. The DS1631 responds with an Acknowledge (ACK) after receiving the control byte. When a device is acting as a receiver, it must generate an ACK on the SDA line after receiving every byte of data, by pulling the SDA line low for an entire SCL period, and the transmitting device must release the SDA, during the ACK clock cycle.

Then the master sends a command byte, which indicates the type of operation to be performed (start temperature conversions, stop temperature conversions, write to configure the sensor (1-byte)/read to see the sensor configuration (1-byte) or read the last converted temperature value (2-byte)). The DS1631 again responds with an ACK after receiving the command byte. The master can either write data to the DS1631 or read data from the DS1631.

It writes data to the DS1631 in order to configure the sensor (select either continuous-conversion mode or one-shot mode, choose output digital temperature data resolution: in this project, the output digital temperature data of the sensor was configured to 12 bits of resolution). In this case, the master must send one byte of data, after receiving an ACK from the DS1631 in response to the command byte. After receiving each data byte, the DS1631 responds with an ACK, and the transaction is finished with a STOP condition signal from the master (the master transitions SDA from low to high while SCL is high and releases the bus to its idle state, in which both SDA and SCL remain high).

It reads data from the sensor in order to get the temperature values. In this case, after receiving an ACK in response to the command byte, the master must generate a repeated START followed by a control byte with the same slave address as the first control byte, but indicating this time that the master next reads from the DS1631. After the DS1631 sends an ACK in response to this control byte, it begins transmitting the requested data on the next clock cycle. For two-byte reads (the temperature values are 2-byte values), the master must respond to the first data byte with an ACK and to the second byte with a "Not Acknowledge" (NACK) followed by a STOP. A NACK signal is a variation on the ACK signal. When the master device is acting as a receiver, it uses a NACK (by leaving the SDA line high during the ACK clock cycle) instead of an ACK after the last data byte to indicate that it is finished receiving data.

Some general 2-wire communication information is provided below:

- All data is transmitted MSb first over the 2-wire bus.
- One bit of data is transmitted on the 2-wire bus each SCL period.
- A pull-up resistor is required on the SDA line, and SCL must either be forced high by the master (if the SCL output is push-pull) or pulled high by a pull-

up resistor (if the SCL output is open-drain). When the bus is idle, both SDA and SCL must remain in a logic-high state.

- After every 8-bit (1-byte) transfer, the receiving device must answer with an ACK (or NACK), which takes one SCL period. Therefore, nine clocks are required for every one-byte data transfer.
- All bus communication must be initiated with a START condition and terminated with a STOP condition signal. The only time when the SDA is allowed to change states while SCL is high is during a START or STOP signal. At all other times, changes on the SDA line can only occur when SCL is low: SDA must remain stable when SCL is high." [13]

Figure 11.3, Figure 11.4 and Figure 11.5 show the 2-wire interface timing for the different operations to be performed as indicated by the command byte sent by the microcontroller (master). More specifically, the possible operations are: the start/stop of temperature conversions, the configuration of the temperature sensor and the reading of the temperature measurements.

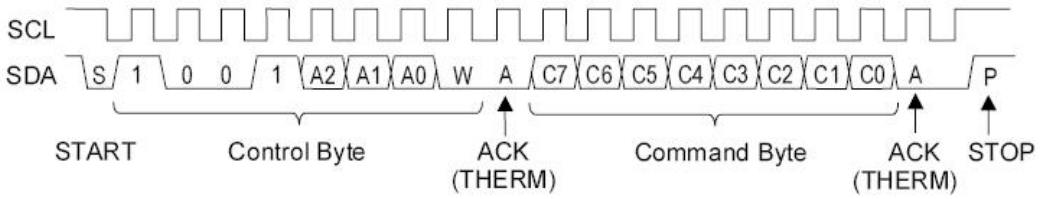


Figure 11.3: Issue a "Start Convert T" or "Stop Convert T" Command

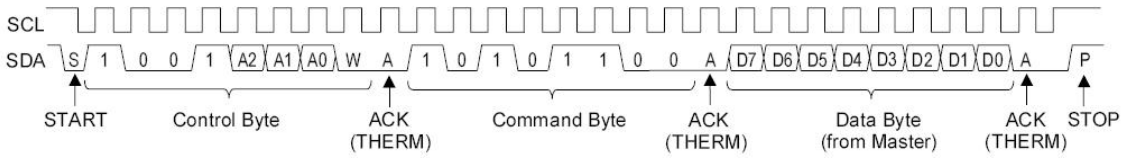


Figure 11.4: Write to the Configuration Register

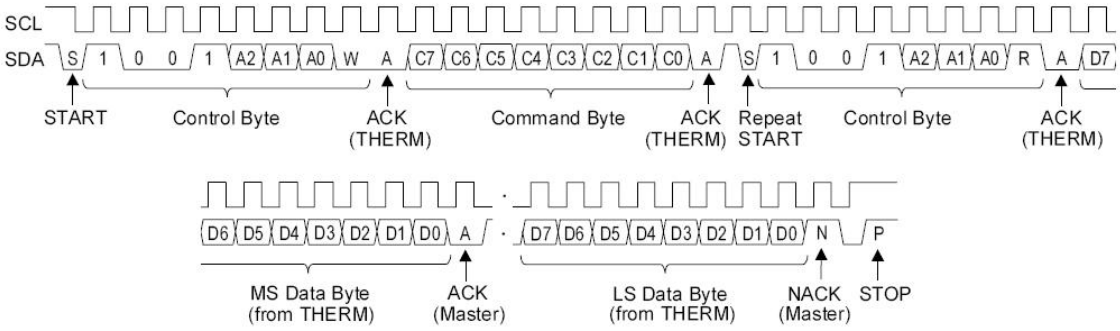


Figure 11.5: Read from the Temperature Register

11.5 Communication between the Transceiver MC13192 and the Microcontroller MCF5212 through the SPI Interface

All control, reading of status, writing of data, and reading of data is done through the MC13192 SPI port. The host microcontroller accesses the transceiver through SPI "transactions" in which multiple bursts of byte-long data are transmitted on the SPI bus. Each transaction is three or more bursts long depending on the transaction type (singular or recursive), since a complete SPI transaction should be framed by Chip Enable $\overline{CE}$. The SPI port of an MCU transfers data in bursts

of 8 bits with most significant bit (MSb) first. The master (MCU) can send a byte to the slave (transceiver) on the MOSI line and the slave can send a byte to the master on the MISO line. Although an MC13192 transaction is three or more SPI bursts long, the timing of a single SPI burst is shown in Figure 11.6.

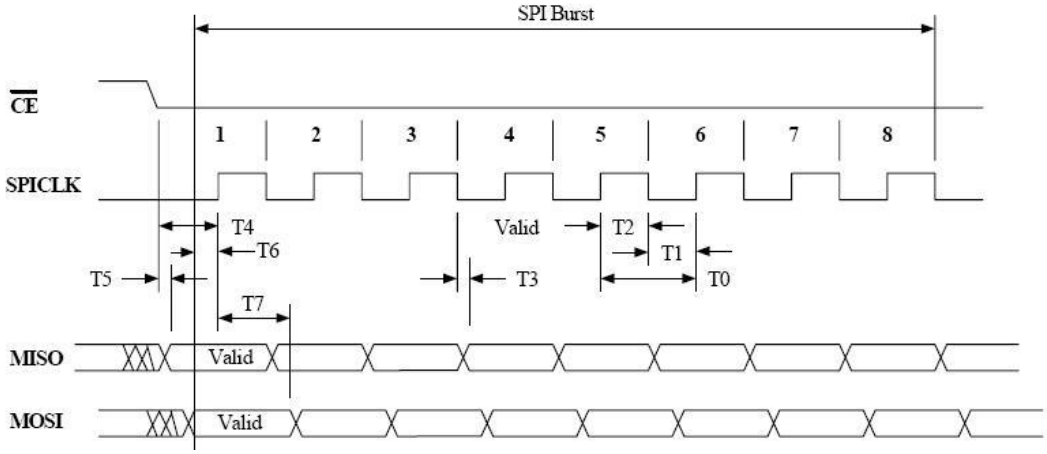


Figure 11.6: SPI Burst Timing Diagram

The assertion of $\overline{CE}$ to low signals the start of a transaction. The first SPI burst is a write of an 8-bit header to the transceiver (MOSI is valid) that defines a 6-bit address of the internal resource being accessed and identifies the access as being a read or write operation. In this context, a write is data written to the MC13192 and a read is data written to the SPI master. The following SPI bursts will be either the write data (MOSI is valid) to the transceiver or read data from the transceiver (MISO is valid). Although the SPI bus is capable of sending data simultaneously between master and slave, the MC13192 never uses this mode. The number of data bytes (payload) will be a minimum of 2 bytes and can extend to a larger number depending on the type of access. After the final SPI burst, $\overline{CE}$ is negated to high to signal the end of the transaction. Figure 11.7 shows a read access transaction and Figure 11.8 shows a write access transaction. [6]

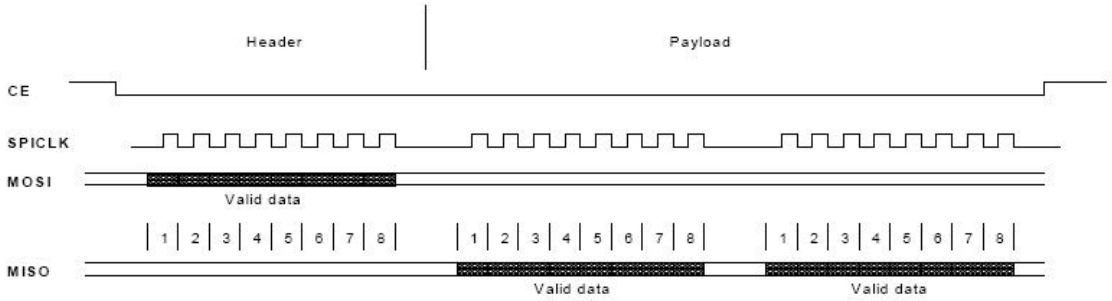


Figure 11.7: Singular Read Access

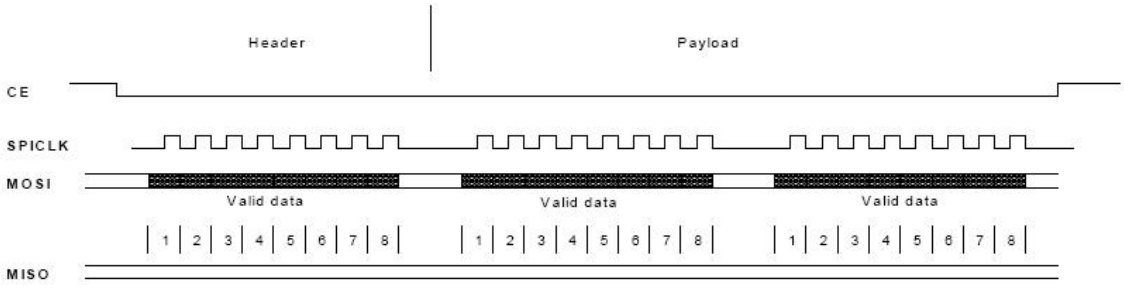


Figure 11.8: Singular Write Access

Part VII

Tests

The last step of the project is to perform tests as to evaluate the performances of our board, communication protocols and network. The tests will be performed following the Packet Error Rate (PER). We will consider that the system (our design) is viable as long as the PER stays under 5%.

We have four leds on-board and we will light them following the PER, as shown as in Table 11.1, so as to measure the PER while the boards are communicating:

Table 11.1: PER following the number of leds lightened on-board

Number of Leds Lightened On-board	PER
0	$\leq$1%
1	1%<PER$\leq$2%
2	2%<PER$\leq$5%
3	5%<PER$\leq$10%
4	>10%

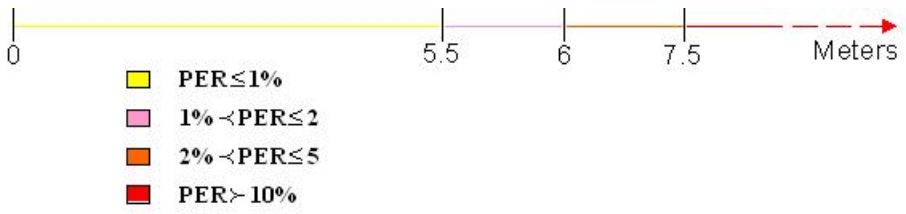
The first test consists of evaluating the communication between two boards (point-to-point communication), one playing the role of a FFD and the other one being a RFD.

For this test, we created various scenarios to estimate the performances of our system in different conditions:

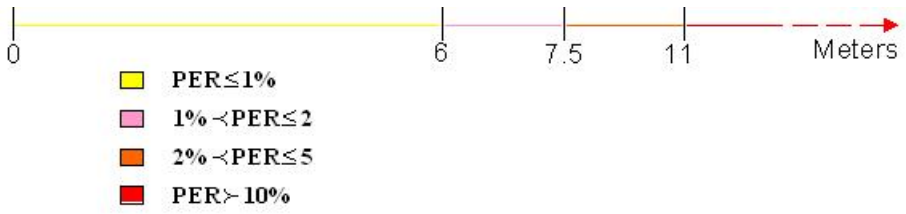
1. **The first scenario is to place the two boards at different distances indoor:**

   Up to 20 meters, we got a PER less than 1%. Indoor we didn't test over 20 meters.

2. **The second scenario is to place the two boards at different distances outdoor:**

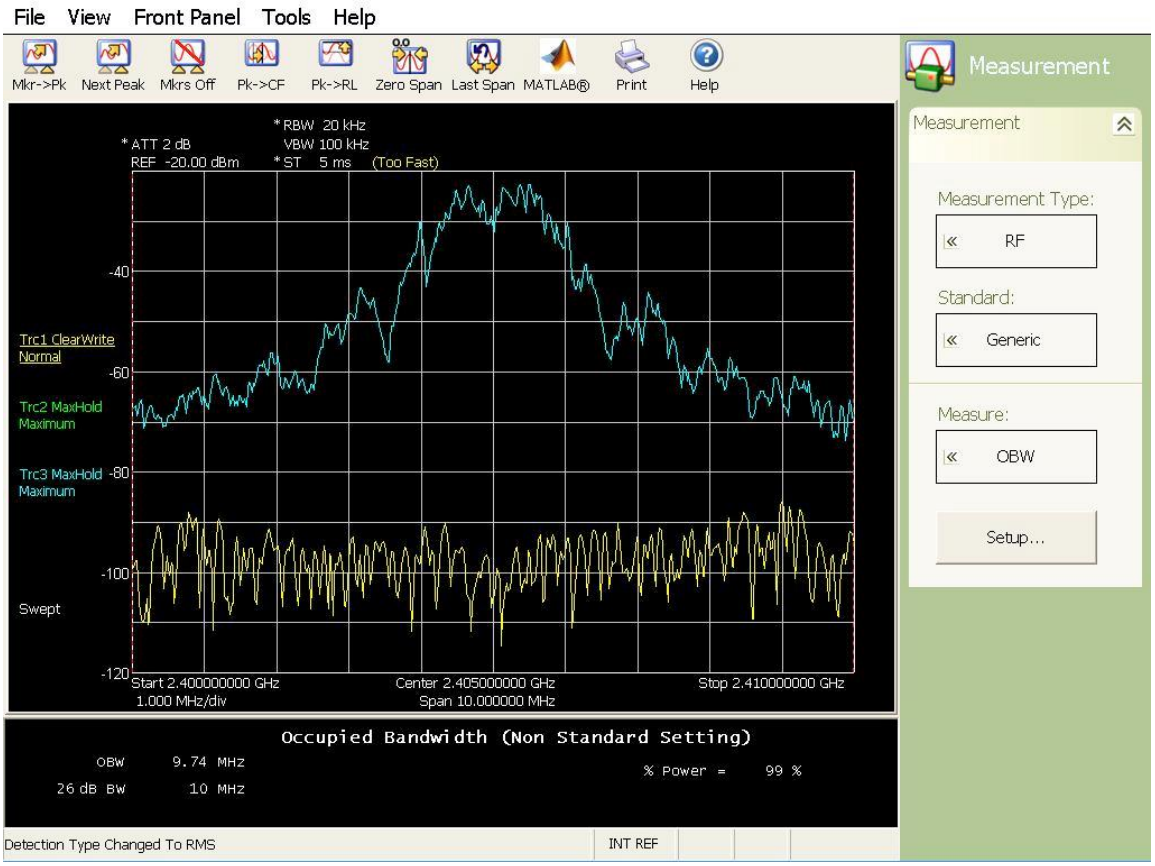


3. **The third scenario is to place the two boards at different heights outdoor (2 meters of difference between the two boards) and again to change the distances between the two boards:**



Finally, we tested the communication between three boards, as to test a small network, with small distances between the boards. The PER was less than 1%.

We also caught on the oscilloscope the power spectrum of the transmitted signal:



Conclusions:

We can notice that the tests indoor give much better results than the one conducted outdoor. We noticed that as long as we don't have any obstacles between the two boards we get much better results (so that we get better results when we have a difference of heights between the boards).

The results with magnets are shown in Appendix D. These graphs are the result of turning around a vertical axis, a magnet on top of/close to the sensors. The goal of another thesis is to explain these results.

The problem encountered while making the tests is the fact that it was not possible to use float values in the program (not supported by the microcontroller) so that we have big ranges of percentages for the PER.

We could also notice that the power consumption on-board is currently too high since we had to change the batteries of the nodes while making the tests.

Part VIII

Conclusion

The goal of this project was the creation of an efficient tool in order to study the traffic, since traffic estimations deriving from this tool, could be used so that future solutions could be found to control traffic and reduce both travel time and negative effects on the environment.

In this project, the work accomplished towards the creation of the tool can be distinguished in three different parts. The design of the tool included both hardware and software design as well as a testing and verification part. More specifically, the hardware design resulted in the creation of a functional wireless magnetic sensor node. The basic components of the board are the FreeScale MCF5212 microcontroller unit, the FreeScale MC13192 transceiver and the Honeywell HMC1001, HMC1002 and HMC1043 magnetic sensors. The software part included the programming of each specific node so as to assure the communication within the node and allow all nodes to communicate with each other as well. Therefore, a wireless network, based on the IEEE 802.15.4 Standard, was successfully created. Finally, a testing and verification phase was conducted so as to characterize the designed node in means of power consumption, radiated power spectrum, and distance range between the nodes in which the traffic estimation measurements are considered correct.

During the project, many problems were encountered, that were finally successfully solved. However, there was one problem noticed that could not be solved. During the testing and verification phase, we realized that it was not possible for a node to transmit data in the network and convert magnetic sensor measurements at the same time, in the sense that the transmission of a packet could influence the simultaneous conversion of the measurements so that erroneous packets could be created.

Finally, as a future enhancement of the project, the implementation of the ZigBee protocol can be proposed so that a wireless ZigBee magnetic sensor network can be designed. In order to do this only the software part needs to be adapted to follow the ZigBee protocol. This is because in the hardware part all the components were carefully chosen so that they can allow for the ZigBee implementation. Moreover, another suggestion for further improving the efficiency of the tool is related with the nodes' power consumption. A duty cycle can be implemented so as to reduce the power consumption. The Reduced-function devices (RFDs) will mostly be in a sleeping mode and will only wake up when needed to transmit as well as every time the need to sense. The coordinator, however, should listen to the channel continuously. Another basic future implementation, that was not included in the goal of this project, is the signal processing of the magnetic sensor measurements so that the resulting estimations can allow for future traffic control and reduction of travel time and traffic's negative effects on the environment as well.

Part IX

Thanks to

Qamcom Technology AB and more specially Fredrik Malmsten and Patrik Bohlin, for welcoming us in the company and helping us out through the project.

Professor Erik G. Ström, supervisor of the thesis, as well as the Signal and System Institution at Chalmers University of Technology for their help and support during the project.

Imego AB and more specially Max Erlandsson, Christian Jonasson and Christer Johansson for helping us out with the sensors.

Geveko for initiating this project.

Finally, all our friends and family for their precious support all along the thesis.

Appendix A

Schematics

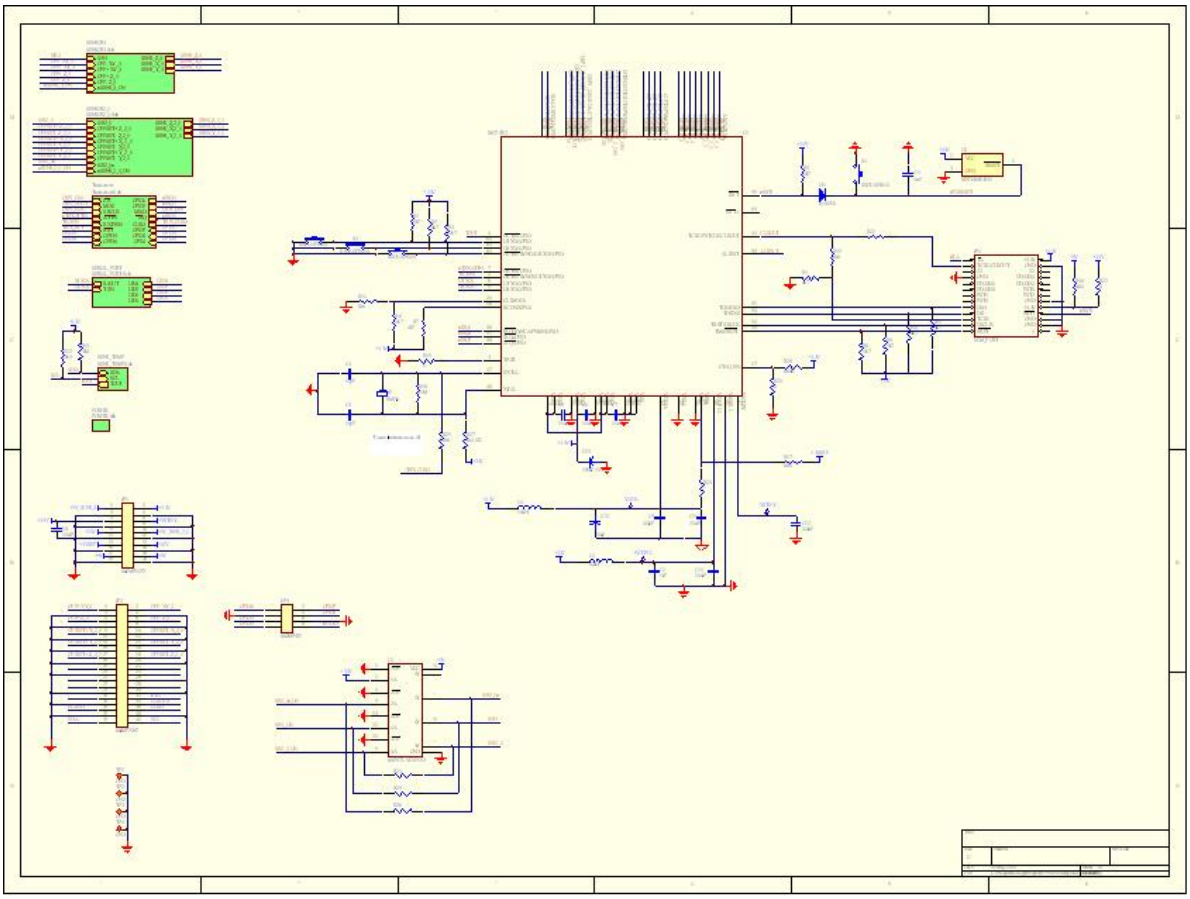


Figure A.1: MCU

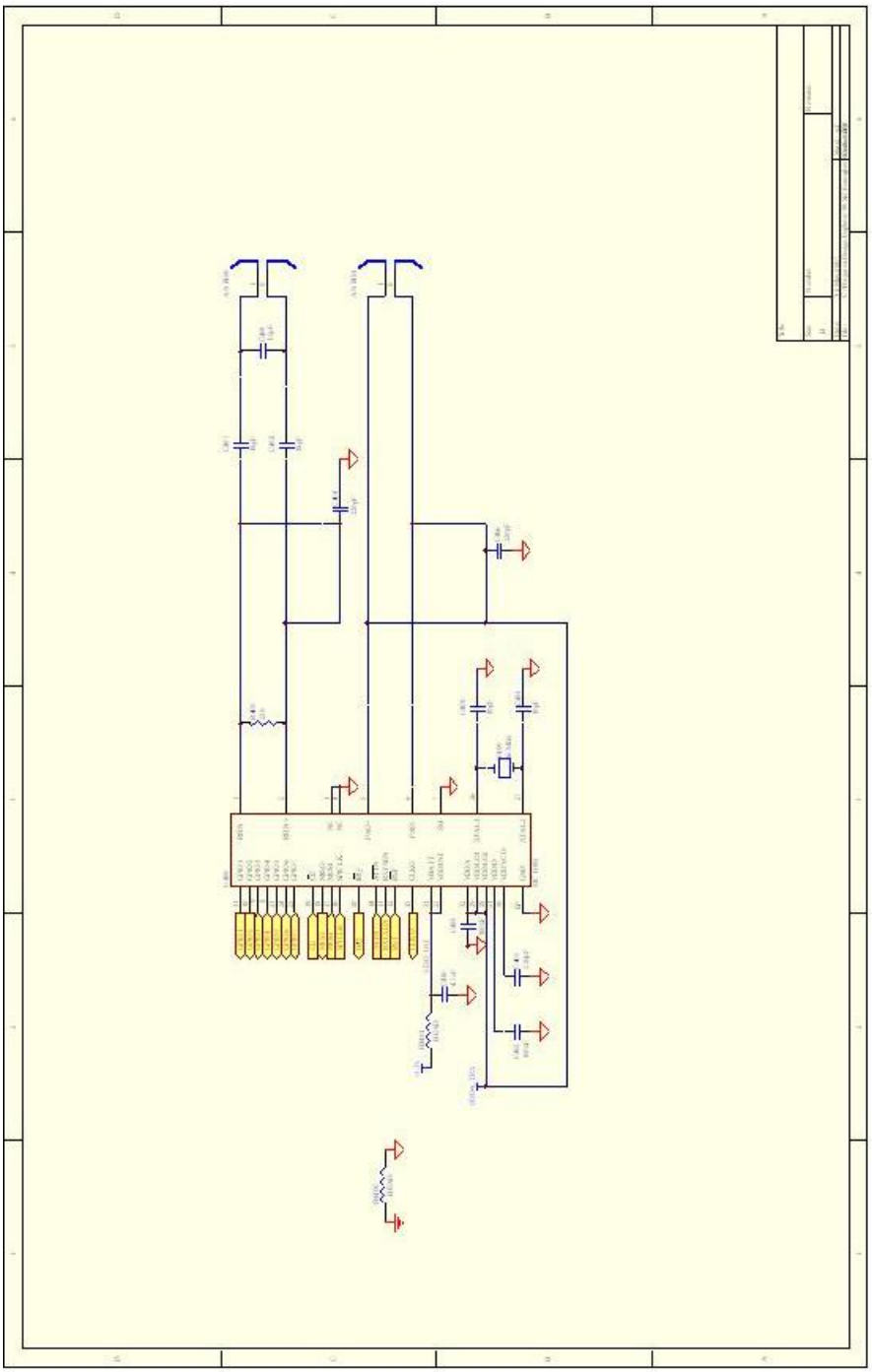


Figure A.2: Tranceiver

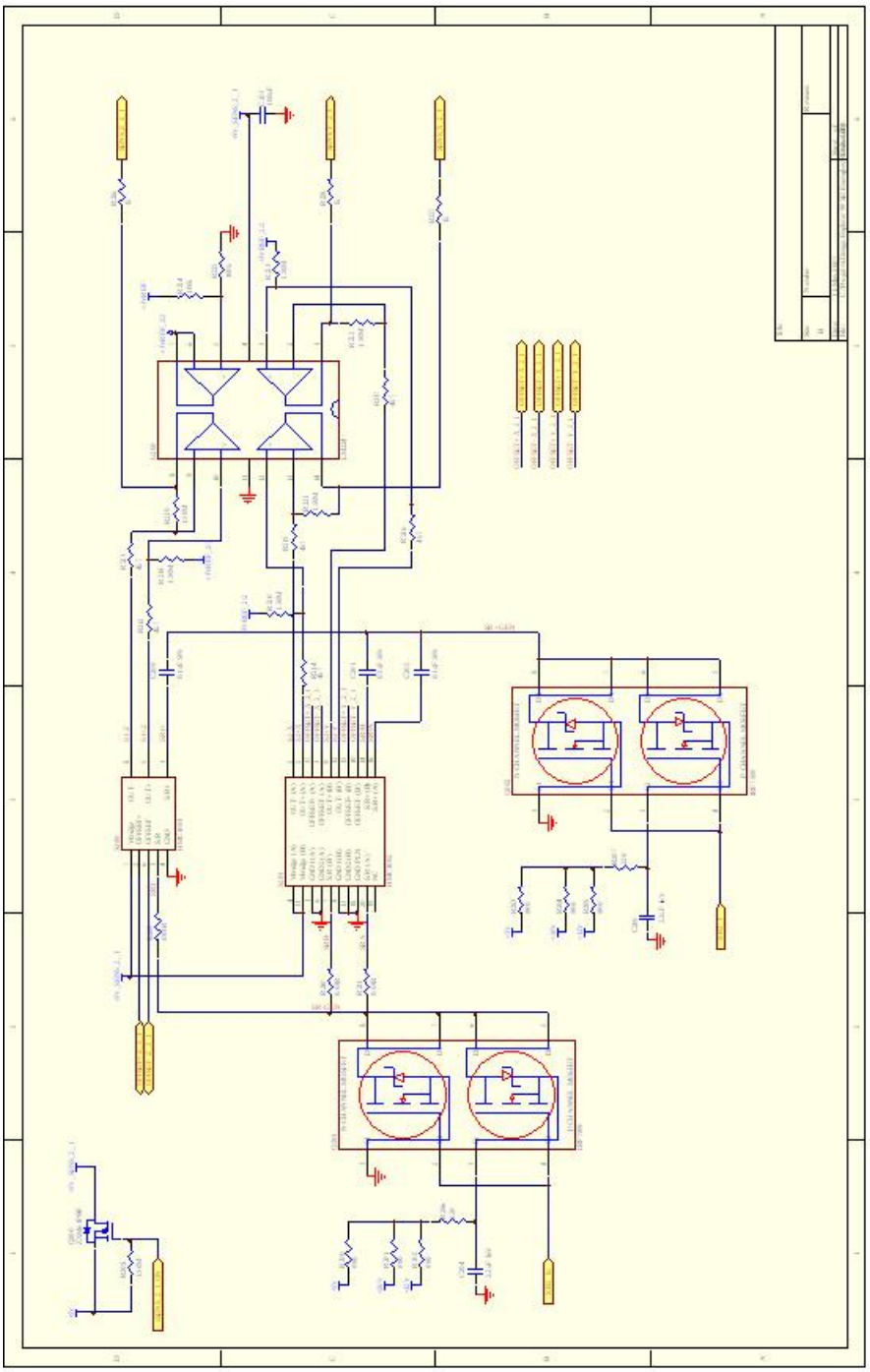


Figure A.3: 1-axis and 2-axis Magnetic Sensors

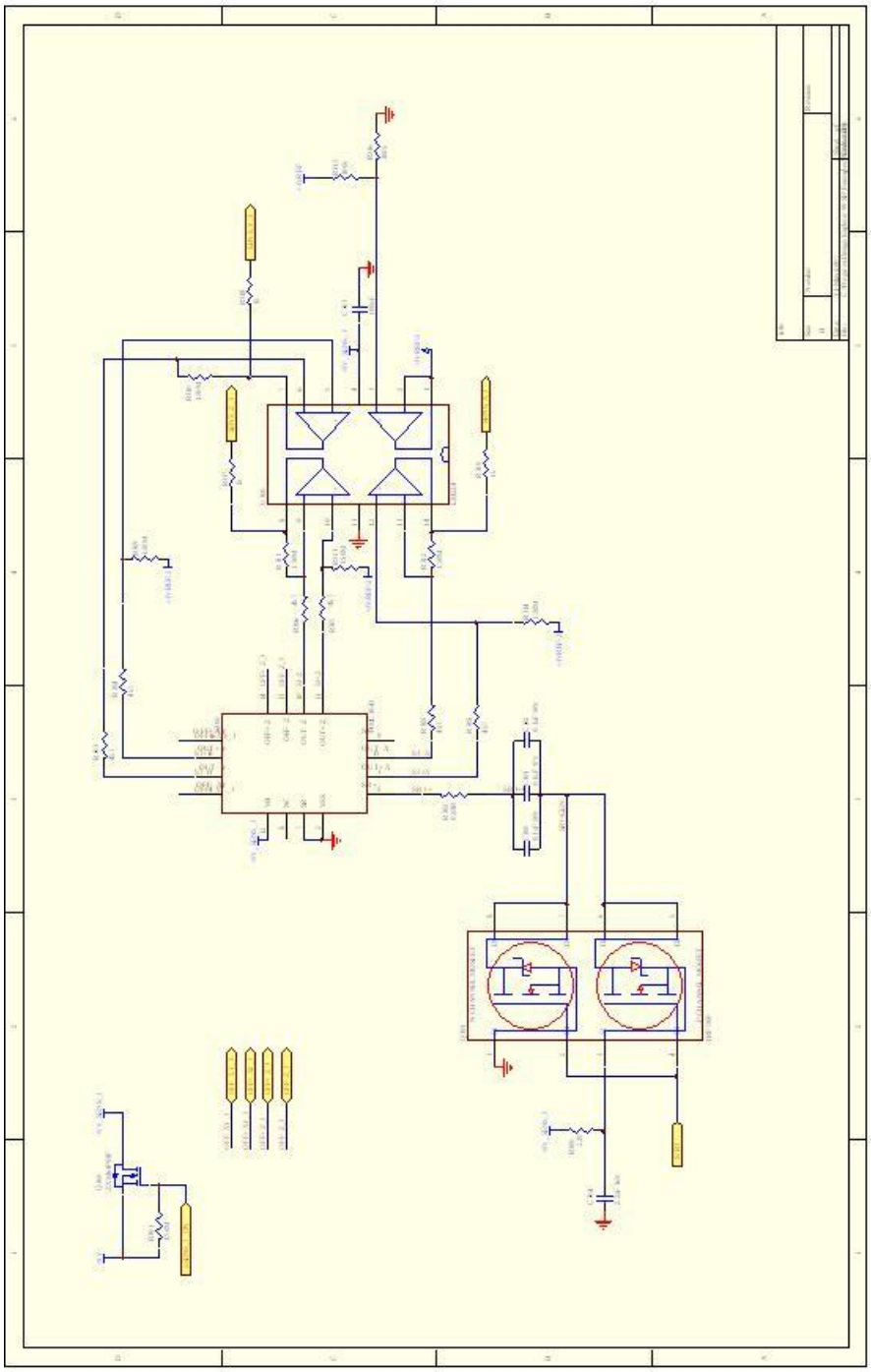


Figure A.4: 3-axis Magnetic Sensor

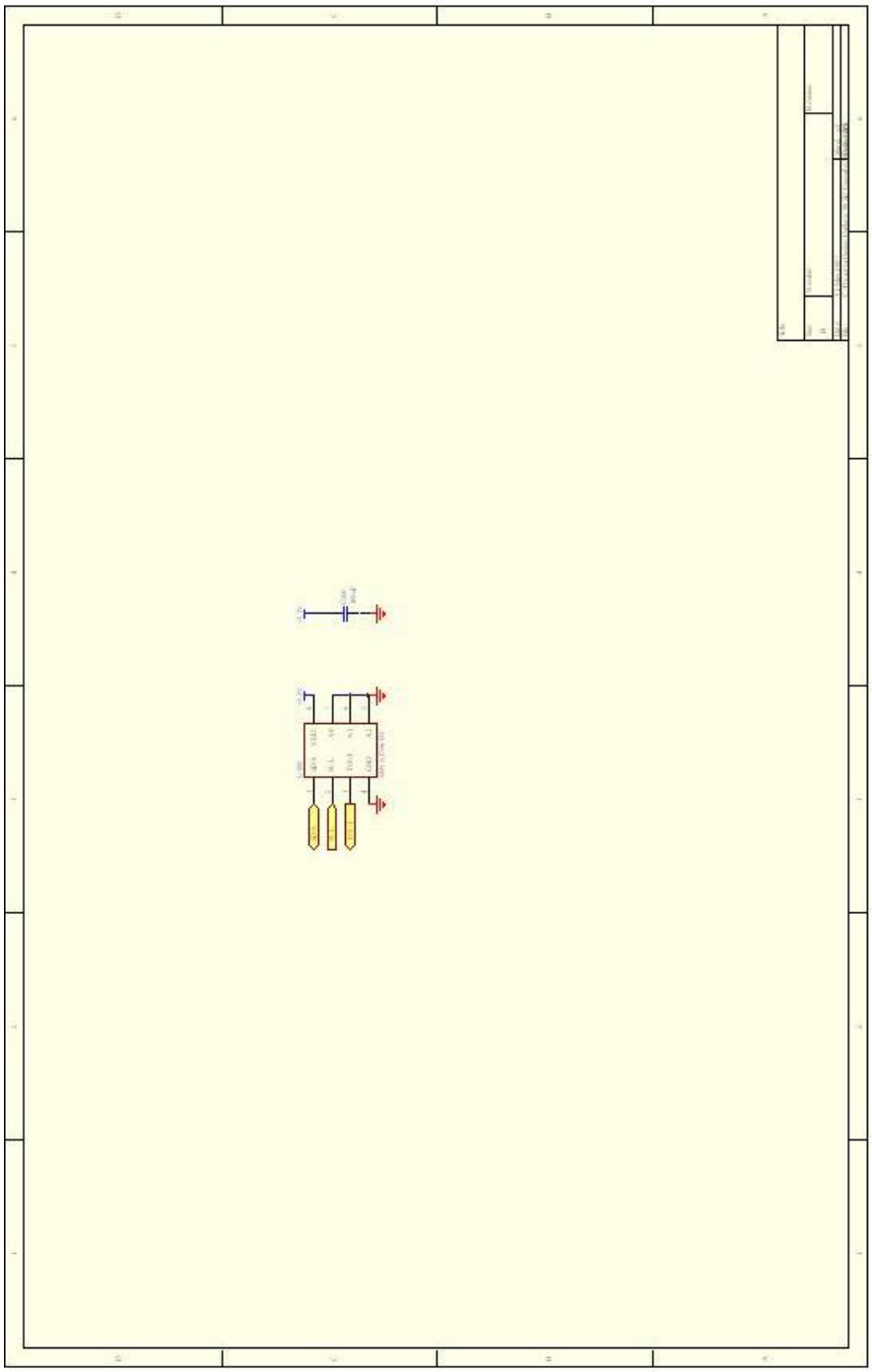


Figure A.5: Temperature Sensor

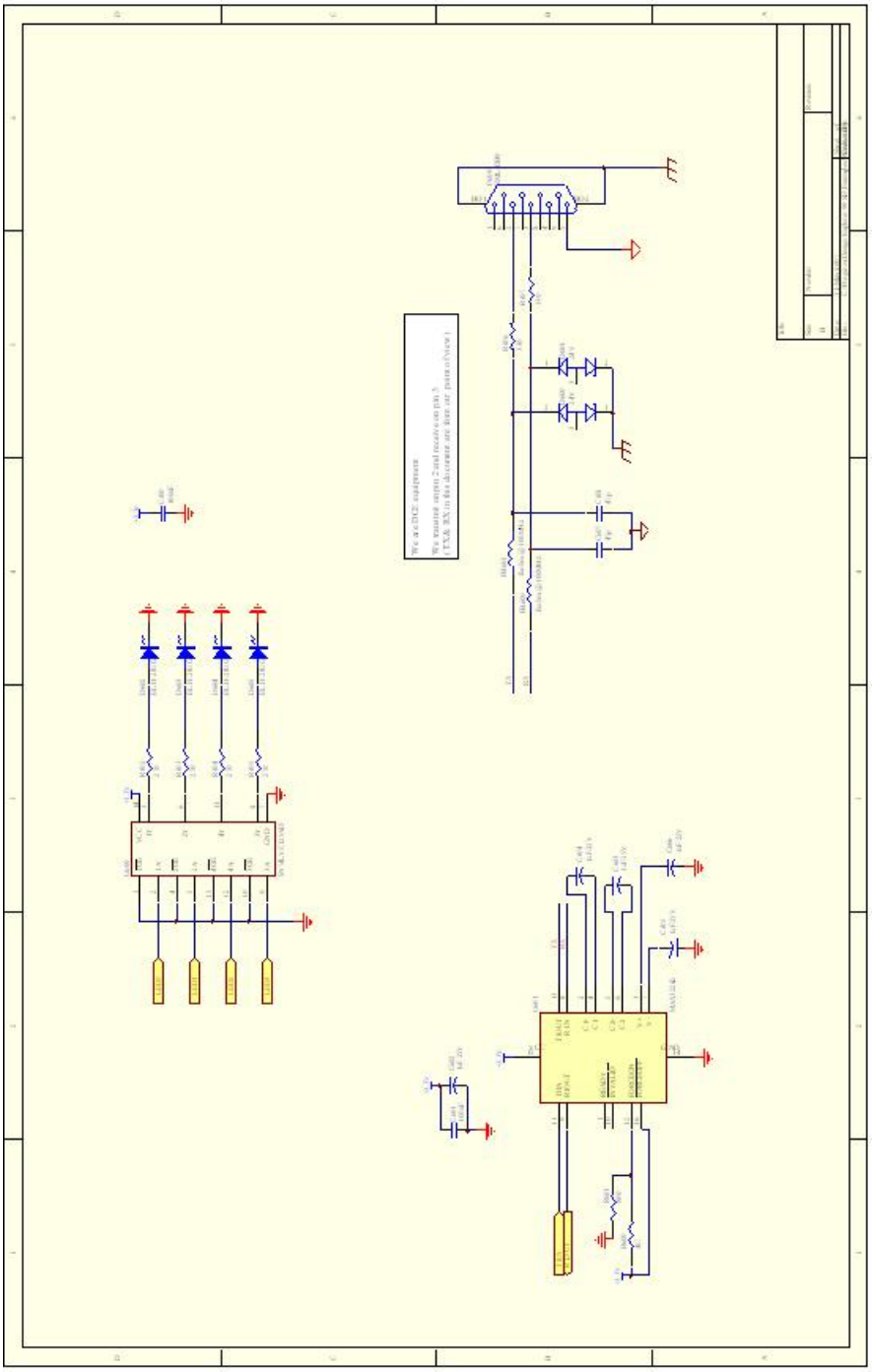


Figure A.6: Serial Port

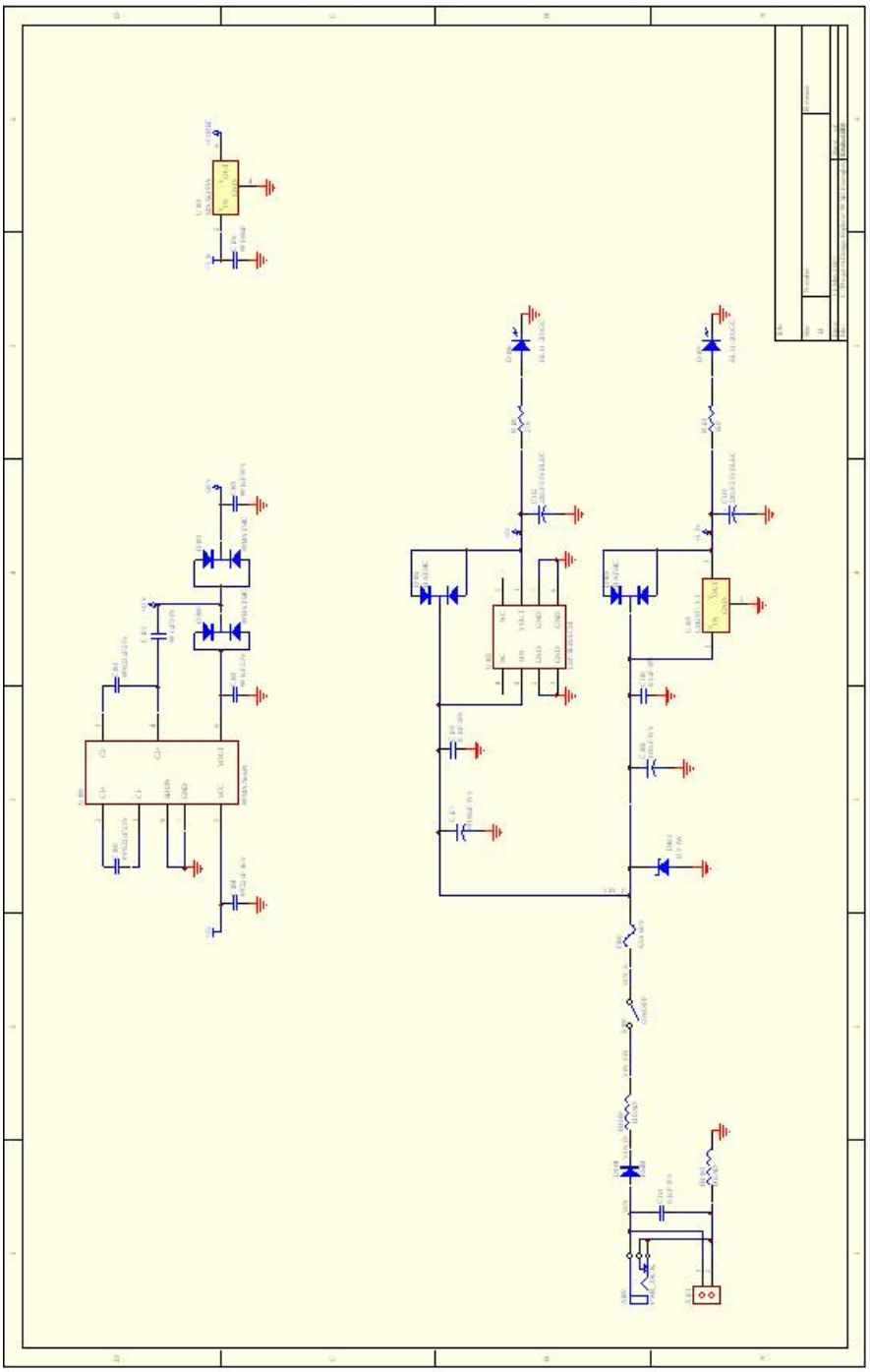


Figure A.7: POWER

Appendix B

PCB of the board

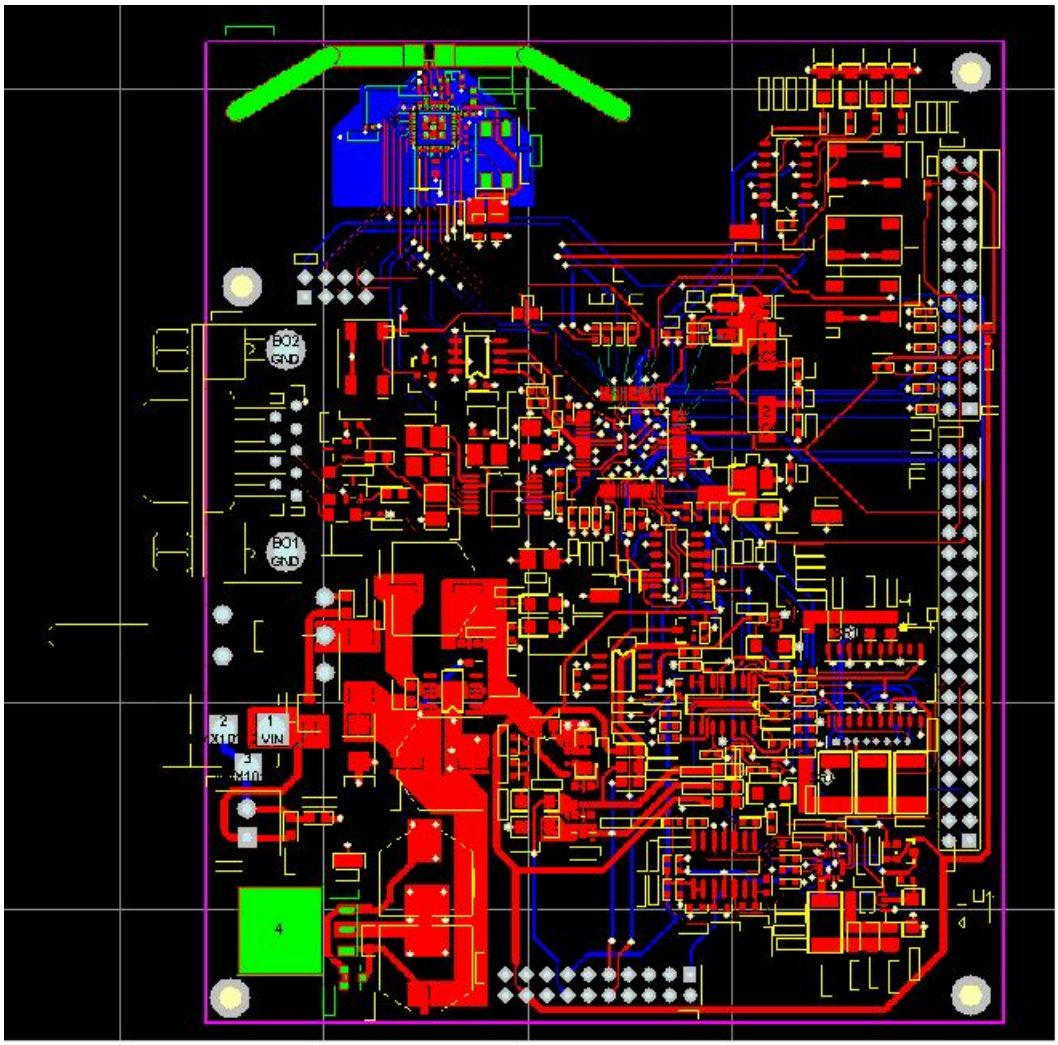
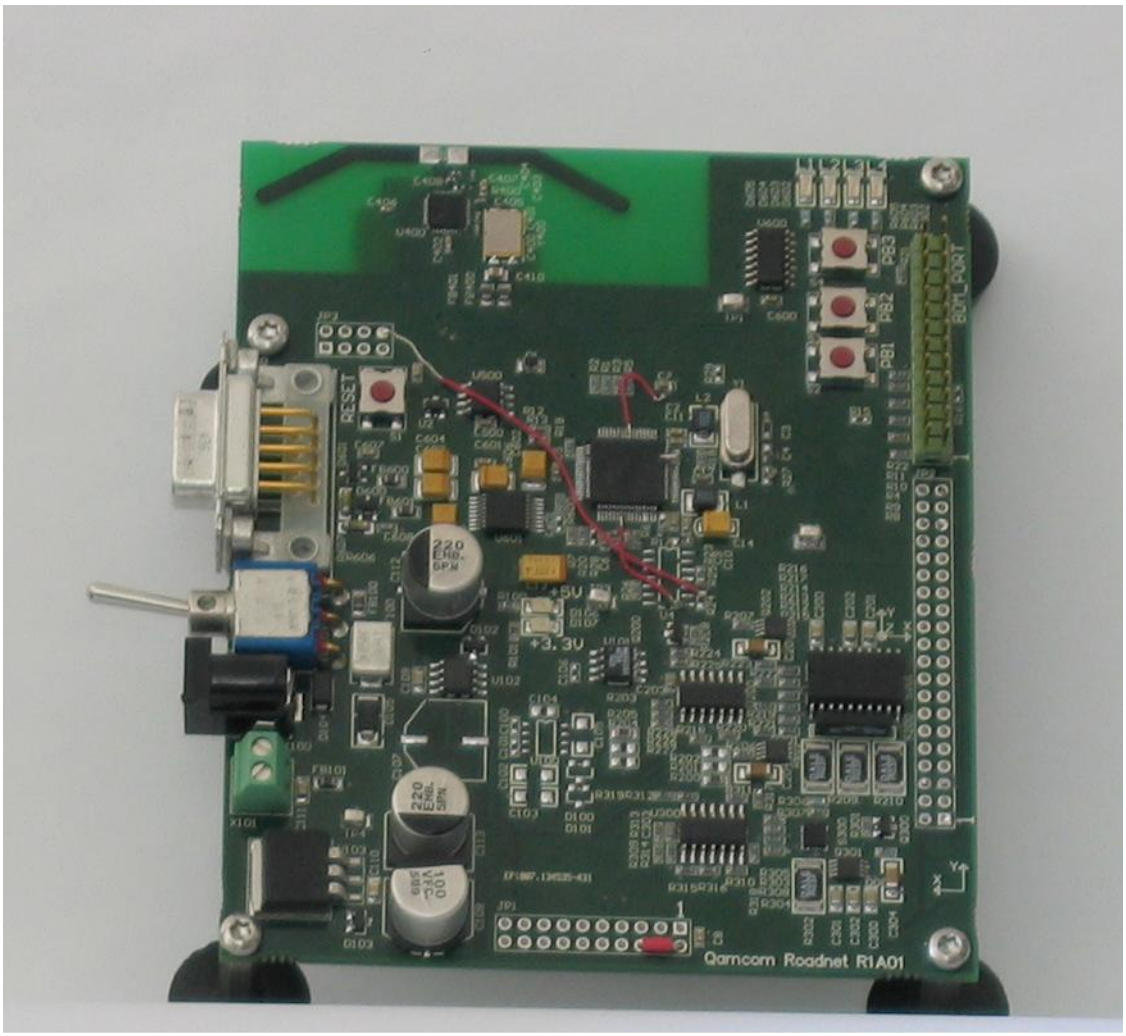


Figure B.1: Printed Circuit Board (PCB)

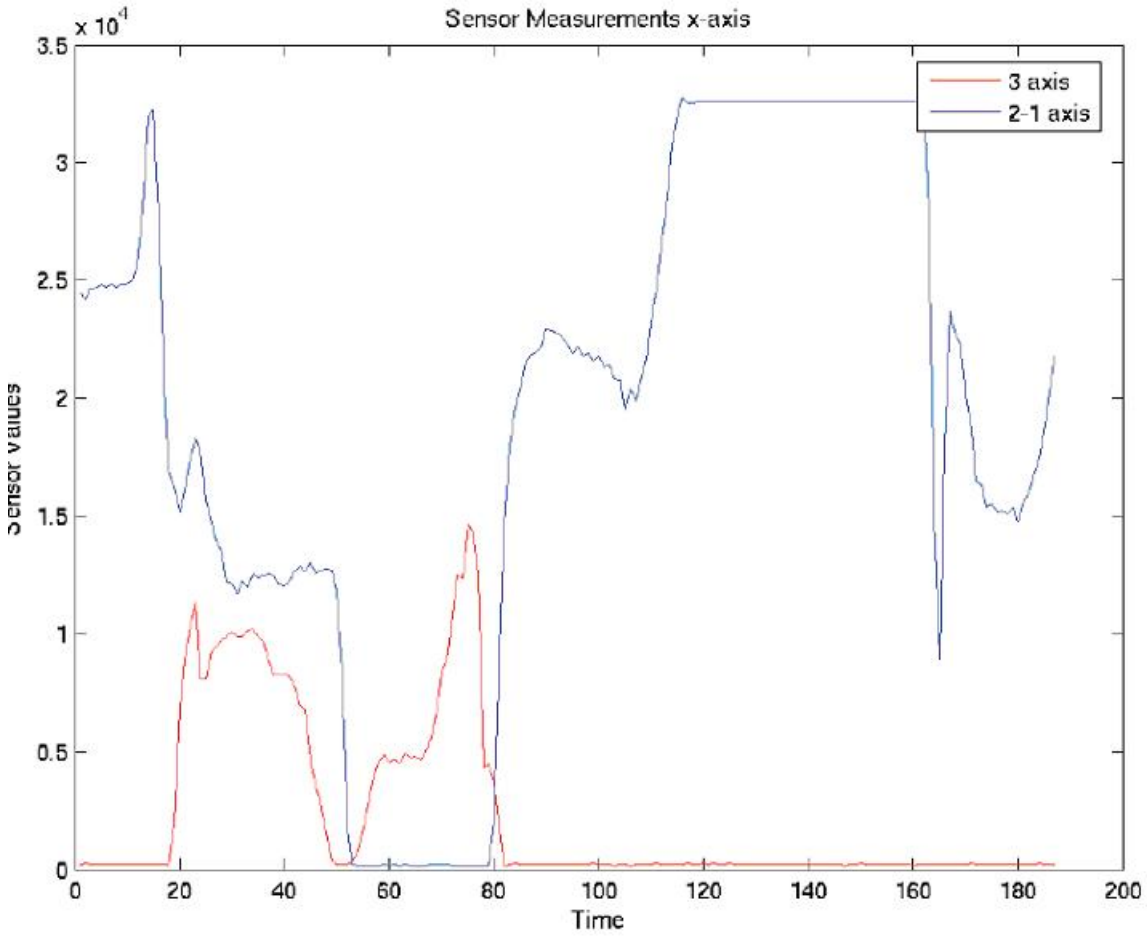
Appendix C

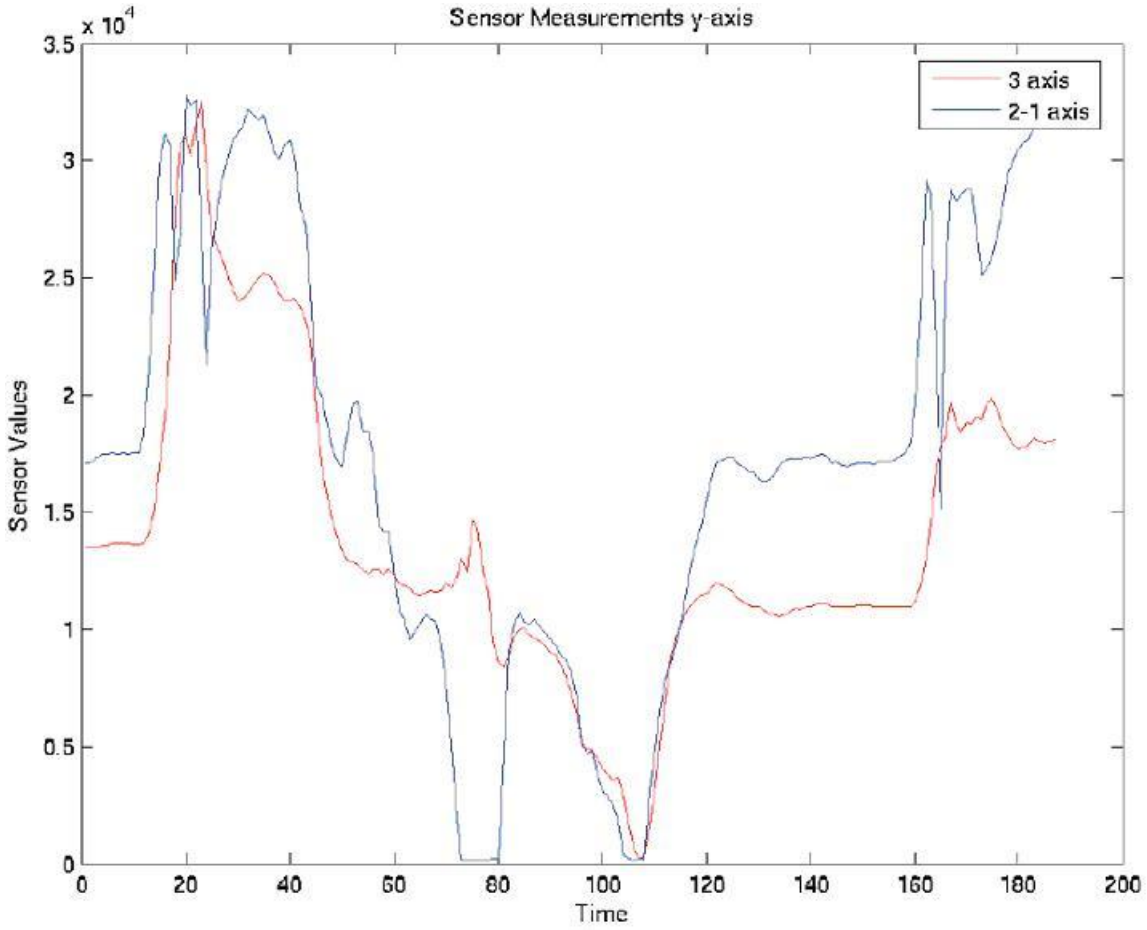
Final Board Layout

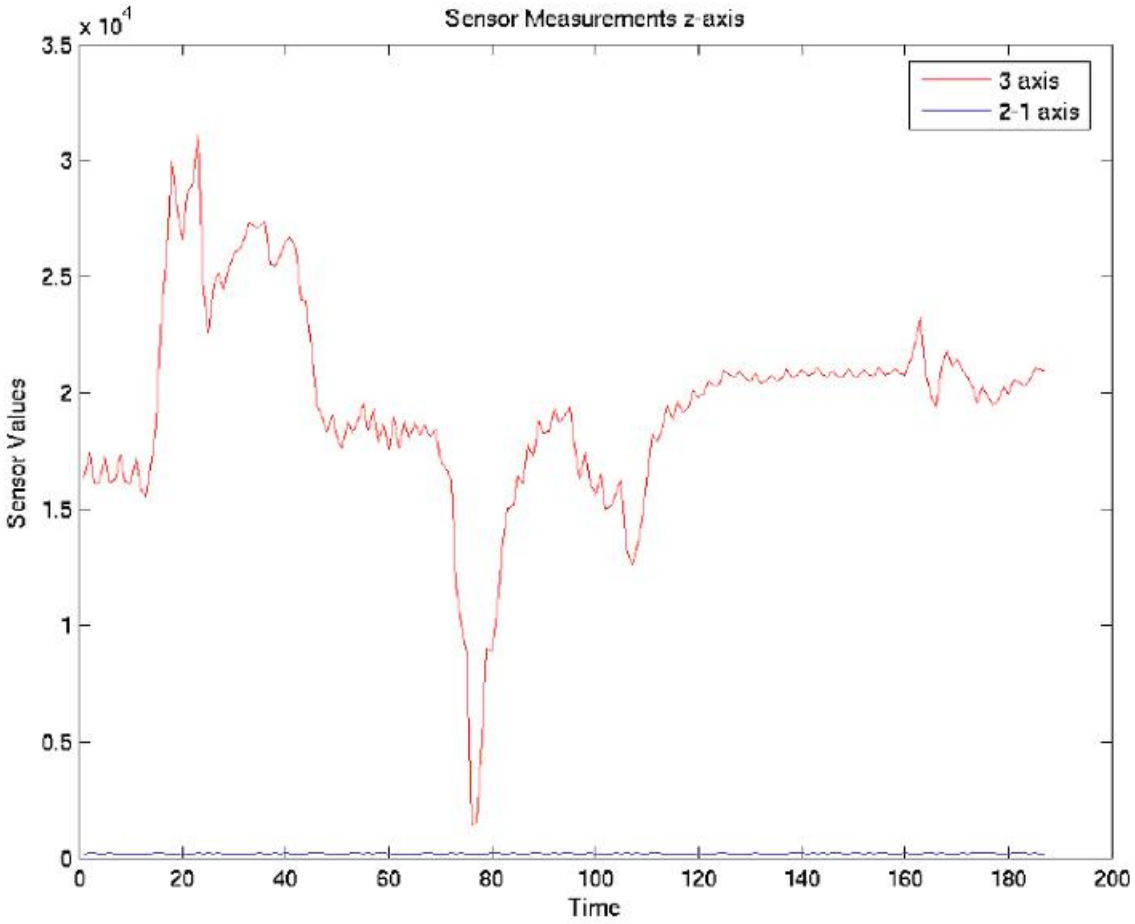

RESET
+5V
+3.3V
Qarncorn Roadnet R1A01

Appendix D

Results of turning around a vertical axis, a magnet on top of/close to the sensors


Sensor Measurements x-axis
x 10^4
3.5
3
2.5
2
1.5
1
0.5
0
Sensor values
0
20
40
60
80
100
120
140
160
180
200
Time
3 axis
2-1 axis


Sensor Measurements y-axis
x 10^4
3.5
3
2.5
2
1.5
1
0.5
0
Sensor Values
0
20
40
60
80
100
120
140
160
180
200
Time
3 axis
2-1 axis


Sensor Measurements z-axis
x 10^4
3.5
3
2.5
2
1.5
1
0.5
0
Sensor Values
0
20
40
60
80
100
120
140
160
180
200
Time
3 axis
2-1 axis

Bibliography

[1] IEEE Computer Society. *802.15.4TM IEEE Standard for Information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*. The Institute of Electrical and Electronics Engineers, Inc., 10/2003. Downloadable from: http://www.compel.ru/images/catalog/31/802.15.4-2003.pdf.

[2] Khanh Tuan Le. *Designing a ZigBee-ready IEEE 802.15.4-compliant radio transceiver*. Chipcon, 11/2004. Downloadable from: http://www.chipcon.com/files/411rfdf4.pdf.

[3] Malcolm Streeton and Roke Manor Research. *ZigBee Technical Datasheet*.

[4] *Internet Market Surveys on ZigBee Chipsets.*
ZigBee and IEEE 802.15.4 Developer Tools, Chips, Chipsets, Protocols:
http://www.palowireless.com/zigbee/devtools.asp,
Building Up ZigBee and 802.15.4 Chipsets and Applications:
http://www.electronics.ca/reports/multimedia/802.15.4_forecasts.html,
ZigBee: Technology for Wireless Sensor Networks:
http://www.mindbranch.com/listing/product/R606-38.html,
ZigBee Emerging Technology Report:
http://www.researchandmarkets.com/reports/347927/zigbee_emerging_technology_report.htm.

[5] *Documentation for the Transceiver Market Survey.*
Texas Instuments (Chipcon) CC2420:
http://focus.ti.com/docs/prod/folders/print/cc2420.html,
Atmel AT86RF230:
http://www.atmel.com/dyn/resources/prod_documents/doc5131.pdf,
Freescale:

http://www.freescale.com/webapp/sps/site/homepage.jsp?nodeId=01J4Fs2565,
Microchip:
http://ww1.microchip.com/downloads/en/DeviceDoc/39776a.pdf,
Oki:
http://www2.okisemi.com/site/productscatalog/mcumpu/availabledocs/Intro-5434.html.

[6] Freescale Semiconductor. *MC13192/MC13193 2.4 GHz Low Power Transceiver for the IEEE®802.15.4 Standard Reference Manual, Document Number: MC13192RM, Rev. 1.5.* Freescale Semiconductor, 10/2006. Downloadable from: http://www.freescale.com/files/rf_if/doc/ref_manual/MC13192RM.pdf.

[7] L. Roshak A. Asmussen, R. Rodriguez. *MC1319x Range Performance.* Freescale Semiconductor, 10/2005. Downloadable from: http://www.freescale.com/files/rf_if/doc/app_note/AN2902.pdf.

[8] *Documentation for the MCU Market Survey.*
MCU Texas Instuments MSP430F149:
http://focus.ti.com/lit/ds/symlink/msp430f149.pdf,
Atmel AVR Atmega64L:
http://www.atmel.com/dyn/resources/prod_documents/2490S.pdf,
Microchip PIC18F4620:
http://ww1.microchip.com/downloads/en/DeviceDoc/39626C.pdf,
Freescale MCF5212:
http://www.freescale.com/files/32bit/doc/data_sheet/MCF5213EC.pdf,
Freescale MC9S08GB/GT family:
http://www.freescale.com/files/microcontrollers/doc//data_sheet/MC9S08GB60.pdf.

[9] Freescale Semiconductor. *MCF5213 ColdFire®Integrated Microcontroller Reference Manual, Devices Supported: MCF5211, MCF5212, MCF5213, Document Number: MCF5213RM, Rev. 2.* Freescale Semiconductor, 10/2006. Downloadable from: http://www.freescale.com/files/32bit/doc/ref_manual/MCF5213RM.pdf.

[10] Max Erlandsson Christian Jonasson and Christer Johansson. *IMEGO AB Magnetic sensors for traffic detection 060817 A pre-study performed at Imego AB.*

[11] Honeywell. *1- and 2-Axis Magnetic Sensors.* Downloadable from:

http://www.cse.ohio-state.edu/siefast/nest/nest_webpage/datasheet/Honeywell%20-%20Magnetometer%20-%20HMC1002.pdf.

[12] Honeywell. *3-Axis Magnetic Sensor HMC1043*. Downloadable from: http://www.ssec.honeywell.com/magnetic/datasheets/hmc1043.pdf.

[13] MAXIM Dallas Semiconductor. *DS1631/DS1631A/DS1731 High-Precision Digital Thermometer and Thermostat.* MAXIM - Dallas Semiconductor. Downloadable from: http://www.tranzistoare.ro/datasheets2/24/249334_1.pdf.

[14] *Introductory Tutorial: Exploring Protel 99 SE*. Downloadable from: http://www.mil.ufl.edu/imdl/handouts/exploringp99se.pdf.

[15] Honeywell. *Vehicle Detection Using AMR Sensors*. Downloadable from: http://www.ssec.honeywell.com/magnetic/datasheets/an218.pdf.

[16] Honeywell. *Set/Reset Function for Magnetic Sensors*. Downloadable from: http://www.ssec.honeywell.com/magnetic/datasheets/an213.pdf.

[17] Honeywell. *Handling Sensor Bridge Offset*. Downloadable from: http://www.ssec.honeywell.com/magnetic/datasheets/an212.pdf.

[18] Freescale Semiconductor. *MC13192/MC13193 2.4 GHz Low Power Transceiver for the IEEE®802.15.4 Standard, Document Number: MC13192, Rev. 3.1*. Freescale Semiconductor, 03/2007. Downloadable from: http://www.freescale.com/files/rf_if/doc/data_sheet/MC13192.pdf.

[19] Freescale Semiconductor. *Simple Media Access Controller (SMAC) User's Guide, Document Number: SMACRM, Rev. 1.4*. Freescale Semiconductor, 10/2006. Downloadable from: http://www.freescale.com/files/rf_if/doc/user_guide/SMACRM.pdf.