Wireless Low Power Data Acquisition Device
Embedded design and application
Wireless Low Power Data Acquisition Device
Embedded Design and Application

Bachelor Thesis
2012 May

Author: Özgun Ayaz
Supervisor: Hans-Erik Eldemark
Examiner: Kenneth Nilsson

School of Information Science, Computer and Electrical Engineering
Halmstad University
PO Box 823, SE-301 18 HALMSTAD, Sweden
Description of cover page picture: Block diagram of the device designed in the project. The diagram depicts the relations between different components on the system, and further information is available in Chapter 4.
Preface

Embedded systems in the modern world are becoming one of the most important areas of digital electronics. One specific application where embedded microcontroller systems are most proficient is data acquisition. Today, whilst lots of different products are available on the market for the purpose of gathering data from sensor networks, most of these are basically built for industrial and large-scale commercial applications. Contrary to these markets, the consumer market remains largely unpopulated, and that was exactly the main reason as to why I decided to implement an easy-to-use, inexpensive and reliable data acquisition module for this market.

The end result is a data acquisition platform which runs a full featured CGI-based dynamic web server and a highly customizable data acquisition platform which could, in a short time, be customized to any consumer’s needs. I looked for a device which I could just take out of its box and start using, and when I couldn’t find one I built one.

I would like to thank Hans-Erik Eldemark and HASI Electronics for their extensive support during all phases of the project, particularly in the matter of exploring smart-grid applications, and Mikaela Grimsberg for putting up with my occasional nervous breakdowns and helping me through moments of frustration.

-Özgun Ayaz
Abstract

Embedded systems are utilized in various modern-day applications in order to ease routine tasks that are otherwise demanding in manpower; an example of which is data sensing and acquisition. Modern sensor applications usually involve one or more embedded microcomputer systems with inputs from various sensors and a means of connection to transmit the acquitted data to its point of interest. While many sensor and acquisition applications and standards exist for industrial use, the consumer/small business markets for such devices seem to be fairly bleak in comparison.

In this project, various data communication standards for transmitting information between sensors and acquisition devices are investigated. Wired, such as RS232, I²C, SPI and 1-Wire, as well as wireless protocols like Wi-Fi and ZigBee were studied. Then, a generic wireless data acquisition platform is designed and realized. The reference implementation connects to a wireless LAN network over Wi-Fi, starts a web server and serves HTML5 web pages and a CSS3 style sheet from SPI flash memory. The sensor data is collected from integrated sensor circuits over an I²C link. Device can run on 3 AA batteries for a combined uptime of at least 3 weeks.

Lastly, possible expansion options for the implemented device are discussed and supplemented with real-world examples where possible. Feasible consumer applications for such a platform are then given to conclude.

**Keywords:** data acquisition system, sensor network, embedded system, wireless communication
# Contents

1 Introduction .................................................................................................................. 1

2 The MSP430 Microcontroller Platform ........................................................................ 3
   2.1 FRAM Technology ................................................................................................. 4
   2.2 Clock System ......................................................................................................... 5
   2.3 Low Power Step-down Operation .......................................................................... 6
   2.4 Universal Serial Communication Interface ......................................................... 6
   2.5 Timers and Interrupts ............................................................................................ 7

3 Sensor Communication Protocols ................................................................................. 9
   3.1 Serial Peripheral Interface ................................................................................... 9
   3.2 Inter Integrated Circuit (I²C) ................................................................................ 10
   3.3 RS232/RS485 Asynchronous Serial Protocols .................................................... 11
   3.4 ZigBee .................................................................................................................. 13
   3.5 Wireless LAN (Wi-Fi) ........................................................................................... 14

4 Design and Implementation .......................................................................................... 15
   4.1 MSP430 FRAM Microcontroller ........................................................................... 16
   4.2 Temperature and Humidity Sensor ....................................................................... 17
   4.3 Light Sensor .......................................................................................................... 18
   4.4 External Flash Storage ......................................................................................... 19
   4.5 Software Implementation ...................................................................................... 20
   4.6 First Time Configuration ....................................................................................... 21

5 Results and Conclusion ............................................................................................... 23

6 Further Discussion ....................................................................................................... 25
   6.1 ZigBee Expansion ............................................................................................... 25
   6.2 RS232/RS485 Expansion ..................................................................................... 25
   6.3 Twitter Expansion ............................................................................................... 26
   6.4 Compliance with Universal Standards .................................................................. 27
   6.5 Estimated Cost of Manufacturing ....................................................................... 27
   6.6 Market Comparison ............................................................................................. 28

References ....................................................................................................................... 31
Figures

Figure 1: The integrated circuit from an Intel 8742, an 8-bit microcontroller that includes a CPU running at 12 MHz, 128 bytes of RAM, 2048 byte of program memory, and I/O in the same chip. (Picture courtesy of Wikipedia) ...

Figure 2: An FRAM cell. (Picture courtesy of Wikipedia) ...

Figure 3: FRAM versus Flash comparison on MSP430 devices (image courtesy of Texas Instruments) ...

Figure 4: Block diagram of the MSP430 FRAM microcontroller (image courtesy of Texas Instruments) ...

Figure 5: Simple connection between a SPI master and a slave ...

Figure 6: Typical I2C transfer logic diagram ...

Figure 7: Transmission of the ASCII uppercase “K” character over an RS232 link as seen on the oscilloscope ...

Figure 8: Example ZigBee network operation in mesh topology ...

Figure 9: Wireless LAN module for embedded systems, compared to an U.S. one cent coin ...

Figure 10: The designed device on the development platform. The main microcontroller, sensors, external flash memory, Wi-Fi module and batteries are clearly visible ...

Figure 11: Block diagram of the data acquisition device ...

Figure 12: Detailed block diagram of the embedded system ...

Figure 13: Close-up of the expansion card built for the project. Chips: (from left to right) TSL2550D, HH6131 and 25VF080B. Bypass capacitors are added to supply rails to smooth out the supply voltages. The circuit is wired on an IC breakout board ...

Figure 14: Bus Pirate universal debugging tool used to program the SPI flash memory. The Bus Pirate is an all-in-one tool to analyze popular serial protocols such as SPI, I2C, 1-Wire, asynchronous serial, MIDI and JTAG ...

Figure 15: Implementation diagram of main microcontroller serial ports ...

Figure 16: Web interface screenshot of the device ...

Figure 17: Digital household power meter with RS485 connectivity ...

Figure 18: Similar device on market, with half the features and seven times the price ...
Tables

Table 1: Power consumption comparison between low power MSP430 Flash and FRAM series microcontrollers. F2 data referenced from [10]. FRAM data is based on actual readings.

Table 2: Description of MSP430 low power steps.

Table 3: Component costs for the designed platform. All prices are in SEK from Farnell Sweden. Prices are current as of 05/13/12. Tax not included. Unit price for 100 pieces.

Table 4: Differences between the designed product and a similar product found on the market.
1 Introduction

An embedded system is a tightly integrated computer system with all parts highly optimized and cooperating towards one task and one task only [1, 2]. In the modern day, nearly all appliances that have digital interfaces—microwaves, cell phones, cars—have an embedded system at the heart of it. Embedded systems usually consist of a microprocessor, read-only memory (ROM) for program space and random-access memory (RAM) to be used as scratch area [2]. Usually, all components in an embedded system are tightly integrated for seamless operation and highly specialized for their purpose, hence the term “embedded.” Some embedded systems might have real time operating systems (RTOS) to help them achieve mission-critical timings, but the vast majority is so specialized for its field that it is more feasible to implement the entire digital logic in one program [3]. Today, embedded systems are utilized in nearly all fields of digital electronics one could imagine, from ATMs and electronic door locks, to ignition timers in motor vehicles and refrigerators [1, 4].

Whilst an embedded system might consist of discrete processor, ROM and RAM parts, it is usually more feasible to implement all three in a single part that would be called a microcontroller (MCU). Typical MCUs consist of processor, program and scratch memory, as well as basic general input output peripherals (GPIO). By reducing the size (and therefore, cost) compared to a design that utilizes discrete processor, memory, and GPIO devices, microcontrollers make it more economical to digitally control devices and processes by combining all these in one integrated circuit, as depicted in Figure 1. Mixed signal microcontrollers are also common, integrating analog components needed to control non-digital electronic systems. Major microcontroller platforms on the market today include Microchip’s PIC, ATMEL’s AVR as well as Texas Instruments’ MSP430 and other microcontrollers based on the ARM Cortex model [4].

One of the areas of digital electronics where embedded systems are the most proficient surely would be sensor data acquisition, which stands for sampling signals that measure real-world physical conditions into numeric values that can be worked on by a digital computer system [2]. Historically this has been done manually by data acquisition operators, entering data by hand from sensor readouts to computer systems to process. Today, this situation is almost nonexistent. Data acquisition from digital sensor devices via digital communication channels is a task that can be easily fulfilled by even the simplest of microcontrollers that exist on the market today. Most microcontrollers are also proficient in performing basic automation capabilities based on the captured data.
With that being said, it should be clear that sensor data acquisition in modern microcontrollers is a fairly straightforward task. However, the real challenge at this point becomes making them as power efficient as they could probably be while keeping the balance with cost efficiency and environmental concerns.

For one data acquisition system to rise to such a challenge, in-depth understanding of microcontroller platforms and data communication systems is a must. Chapters 2 and 3 will provide an overview of the MSP430 microcontroller platform and various data communication models utilized in the project. Chapter 4 will provide detailed information of how the reference platform is implemented, and Chapter 5 will supplement the documentation with possible expansion options and real world application cases.
2 The MSP430 Microcontroller Platform

With more microcontrollers available in market today than one could possibly count, the MSP430 platform was utilized in this project. The MSP430 is a mixed-signal microcontroller family from Texas Instruments [5]. Built around a 16-bit CPU, the MSP430 is designed for low cost and, specifically, low power embedded applications. On account of this design attitude, active power consumption in MSP430 microcontrollers can be as low as a few micro amperes, while the data retention consumption can be in the order of just a few nanoamperes; which is what makes this MCU family extremely suitable for this application.

The top CPU speed of the family is 25 MHz, with the possibility of throttling it further down for extremely low power appliances [6]. The MCU family also comes with a power control unit that can step the microcontroller down in 8 power stages. These low power steps allow the main processor unit to sleep while letting its peripherals keep working and send an interrupt pulse to the processor to wake it up when needed, thus keeping the overall power consumption at a low level. Additionally, the MSP430 is capable of wake-up times in the order of nanoseconds, which allows for keeping the processor in sleep for a longer time and bringing it up swiftly when it is needed.

Every member of the MSP430 microcontroller family comes featuring quite a lot of peripherals: internal oscillator, timers, PWM, watchdog which allows resolving race conditions, Universal Serial Communication Interfaces (USCI), analog-to-digital converters, ADCs, and brownout reset circuitry [6]. Some less usual peripheral options include comparator, built in operational amplifier for signal conditioning, digital-to-analog converter, liquid crystal display (LCD) screen driver, hardware multiplier, Universal Serial Bus (USB), and direct memory access (DMA) controller [6]. Apart from some mask ROM devices designed for high volume production, all of the devices are in-system programmable via JTAG or a built in bootstrap loader using RS-232, which allows for easy prototyping without having to manufacture expensive printed circuit boards (PCB) or factory-programmed chips.

Starting from May 2011, MSP430 chips which feature Ferroelectric Random Access Memory (FRAM) technology are also available in the market [7]. FRAM is a bleeding-edge development in memory technology, with MSP430 FRAM being the first experimentation platform available in this field. FRAM technology aims to bridge the gap between flash memory and RAM. In parallel to the recent availability of the first FRAM-based MSP430 batch, a microcontroller from this product line was chosen. More information about the MSP430 in general will be given in the following sections.
2.1 FRAM Technology

As mentioned before, ferroelectric random access memory (FRAM) is a recent development in the engineering field which aims to bridge the gap between RAM and flash memory [8]. FRAM is a type of RAM that retains the contents even when the power is turned off. It works faster and can be write cycled far more times than flash memory thus becomes possible to design electronics that are less dependent on being in stand-by mode (see Fig. 3). The memory circuit that is used in the microcontroller has been in development since 2001 by Texas Instruments and Ramtron and produced in a modified 130 nm process [9].

A FRAM memory cell basically works just like a dynamic RAM (DRAM) cell, but instead of using a dielectric layer, it uses a ferroelectric layer to achieve non-volatility [8]. This way, it basically breaks down the line between non-volatile flash memory and fast but volatile RAM memory. Due to this non-volatile nature, FRAM memory on the MSP430 microcontroller could be configured as either code or data memory (see Figure 3). A simple FRAM cell is depicted in Figure 2.

Another advantage of using FRAM memory versus flash memory is its extremely low power consumption. Due to FRAM’s inner workings being similar to a dynamic RAM cell it can work with relatively low power consumption, and due to it being non-volatile and doesn’t require refreshing like DRAM memory, the power consumption gets even lower. Table 1 shows power consumption differences between MSP430 FRAM series and low-power flash series microcontrollers. As could be seen from the table, FRAM is a promising technology and could be used to manufacture products that could perform with next to no available power.

Figure 2: An FRAM cell. (Picture courtesy of Wikipedia)
### Table 1: Power consumption comparison between low power MSP430 Flash and FRAM series microcontrollers. F2 data referenced from [10]. FRAM data is based on actual readings.

<table>
<thead>
<tr>
<th>Mode</th>
<th>MSP430 F2 Series</th>
<th>MSP430 FRAM Series</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Active Mode</strong></td>
<td>365 µA/MHz</td>
<td>81.4 µA/MHz</td>
</tr>
<tr>
<td><strong>Standby</strong></td>
<td>0.5 µA</td>
<td>0.32 µA</td>
</tr>
<tr>
<td><strong>Shutdown</strong> (Data retention)</td>
<td>0.1 µA</td>
<td>No power needed</td>
</tr>
</tbody>
</table>

### 2.2 Clock System

MSP430 has 3 different clock options. The first clock is the master clock (MCLK) and used to provide clock signals to the main processor and memory systems. The second clock generator is the auxiliary clock (ACLK) and can be software selectable by individual peripheral modules as their clock sources. Both MCLK and ACLK can be divided by 1, 2, 4, 8, 16 or 32.

The third clock option is the subsystem master clock (SMCLK). Just like ACLK, this clock signal can be used to provide clock pulses to peripheral modules, with the difference being that it cannot be divided. Some peripheral modules have a fourth clock option called the module clock (MODCLK) that can be only used by them.

All clock signals can be sourced from the low power low frequency oscillator (LFO), the internal digitally controlled high frequency oscillator (DCO) or one of the two external clock inputs.

![Figure 3: FRAM versus Flash comparison on MSP430 devices (image courtesy of Texas Instruments)](image-url)
<table>
<thead>
<tr>
<th>Mode</th>
<th>Disabled peripherals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Nothing</td>
</tr>
<tr>
<td>LPM0</td>
<td>CPU, MCLK</td>
</tr>
<tr>
<td>LPM1</td>
<td>CPU, MCLK, DCO</td>
</tr>
<tr>
<td>LPM2</td>
<td>CPU, MCLK, DCO, SMCLK (optional)</td>
</tr>
<tr>
<td>LPM3</td>
<td>CPU, MCLK, DCO, SMCLK</td>
</tr>
<tr>
<td>LPM4</td>
<td>CPU, MCLK, DCO, SMCLK, ACLK</td>
</tr>
<tr>
<td>LPM3.5</td>
<td>Everything except GPIO pads</td>
</tr>
<tr>
<td>LPM4.5</td>
<td>Everything</td>
</tr>
</tbody>
</table>

Table 2: Description of MSP430 low power steps

This flexibility in the clock system allows different peripherals on the processor to be sourced from different clocks, therefore enabling the option of halting individual peripherals via the means of low power step-down modes, which is further discussed in 2.3.

### 2.3 Low Power Step-down Operation

As indicated before, the MSP430 is capable of stepping down to eight different power modes. These power modes allow the microcontroller to power down unneeded peripherals while keeping the necessary ones working, thus saving power while continuing uninterrupted operation. The power modes range from active (with everything powered and working) to shutdown (power completely cut off) with 6 middle steps in between.

This capability of low power operation allows the MSP430 FRAM device to work for even longer periods of time under low power availability conditions. It takes just a write to the MSP430 status register to go into and out of any of the power stages. Detailed information about MSP430 low power operation is given in Table 2.

### 2.4 Universal Serial Communication Interface

It is very typical for modern microcontrollers to have several peripherals to communicate with external devices. These can include digital inter-chip connectivity protocols such as Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I²C) and asynchronous serial protocols such as RS232 and RS485.

In the MSP430 design, instead of providing all these in discrete peripheral modules, there are unified serial interfaces which could be configured as any of the serial protocols as needed, which are called Universal Serial Communication interfaces (USCI). This unified operation grants the developer a big amount of flexibility in designing inter-chip communication interfaces. In one instance, it allows re-using the same I/O pads for different devices that use different protocols and aren’t communicated with at the same time. This is a typical design case and such an
optimization without USCI modules is impossible to achieve. In other cases, it allows the use of a single peripheral module for all communication protocols, therefore eliminating the need for separate modules for each protocol, thus saving power.

Each USCI module has embedded transmit and receive buffers, and capable of generating interrupts on incoming data and completed transfers, which allows for programming the microcontroller to attend to other tasks while waiting for data transfer to complete.

In MSP430 microcontrollers, there are 2 different types of USCI modules, dubbed USCI-A and USCI-B. The A-type can be used for SPI, I²C, infrared data (IrDA), RS232 and RS485, whilst the B-type can only be used for SPI and I²C. Detailed information about these communication protocols will be given in the next chapter.

2.5 Timers and Interrupts

The MSP430 has 5 timer modules and the main processor is able to receive interrupts from timer peripherals. All timers can be software configured to counting up, down or hitting in intervals. The main processor is also able to receive interrupts from all other peripherals, which include USCI modules, GPIO pads and other undocumented subsystems. It must be noted that tandem utilization of interrupts with low power stages are key in designing power efficient embedded applications. A block diagram showing these components in detail is depicted in Figure 4.
3 Sensor Communication Protocols

In embedded systems, data communication between components of the system is achieved with the use of several different communication protocols. For seamless and error-free operation of the system, it is a must for one to possess a thorough understanding of all communication protocols one intends to utilize.

It is common in embedded applications to communicate with external systems via wired protocols such as Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I²C, pronounced I-Square-C), and asynchronous serial such as RS232/RS485 [1, 4, 11, 12].

The most common wireless protocols are ZigBee and Wi-Fi with both being approved as standards under the Institute of Electrical and Electronics Engineers (IEEE, pronounced I-Triple-E) [1, 4]. ZigBee is the most common form of wireless communication between sensor nodes, allowing for sufficient data rates and long ranges. Wi-Fi allows for higher data rates and more sophisticated error correction at the cost of increased power consumption, and as a result utilized more often at sensor network endpoints rather than nodes themselves [13].

In this chapter, the most common protocols used in sensor networks will be discussed.

3.1 Serial Peripheral Interface

Serial Peripheral Interface (SPI) protocol is a full duplex synchronous serial data protocol created by Motorola used by digital systems for communicating with one or more peripheral devices over short distances [12]. In an SPI connection, there is one master device (usually the microcontroller or main component of the system) which controls the slave (peripheral) devices. SPI is a synchronous standard, which means the clock pulses must be transmitted along with the data.

In the standard SPI implementation, 3 wires are used. These wires are:

- MISO – Master in, slave out: Data from slave to master
- MOSI – Master out, slave in: Data from master to slave
- SCLK – Serial clock: Clock signal to be associated with the data

Figure 5: Simple connection between a SPI master and a slave.
Usually, SPI systems implement a fourth line called the “Chip Select” (CS) line – if present, the voltage level on this line indicates if the master is talking to the slave or not. In SPI systems with multiple slaves, each slave must have their own CS line and the master indicates which slave it is talking to by asserting the CS line of the relevant slave. If a slave is not being talked to, it must disregard the data on its MOSI line and not drive the MISO line. SCLK is provided by the master and slaves must never manipulate this line. The master can hold the clock line low or high at idle as supported by the slaves. A simple connection diagram can be seen in Figure 5.

As mentioned before, SPI is a full duplex protocol, which means data must be transmitted in and out of a device at the same time. In the SPI implementation of this, one edge of the clock signal (usually the rising edge) is used to send data from the master to the slave, and the other (falling) edge is used by the master to receive data from the slave. That means, for every byte that is being transmitted from the master, a byte is read from the slave. With this method, SPI allows for up to 80 Mbps of full duplex transfers.

SPI is a fairly easy protocol to implement in microelectronic devices. On embedded microcontroller systems, it can be implemented in software with just a few bytes of code, or due to it being the main method of communicating between a microcontroller and peripheral devices such as communication or memory modules, it can be achieved using dedicated communication modules available in nearly every microcontroller on the market today [12].

### 3.2 Inter Integrated Circuit (I²C)

The Inter-Integrated Circuit interface was created by NXP Semiconductors (then Philips) in 1982 as a method of communicating between various controller chips [2, 11]. Commonly written as “I²C”, the protocol allows communication of data between devices over two wires. It sends serial synchronized information using one line for data (SDA) and one for clock (SCL) and, similar to SPI, allows multiple devices on the same data bus and allows for transfers of up to 5 Mbps.

There are two main differences between I²C and SPI. The first is that instead of two data lines, there is only one; which facilitates the need of direction arbitration on the data line. This is achieved by using an open drain configuration on the line with external resistors to pull it to a high state, which ensures that no two outputs are connected together at one time. The second is that the data is transmitted in a half-duplex fashion, which means that incoming and outgoing transfers cannot take place at the same time.
Since there is no select line, I²C uses 7-bit device addresses each being unique to every slave to address different chips on the same bus. Read/write operations are selected with the 8th bit on the address transaction byte – if the master transmits a logical high after the 7-bit address, this indicates that a read will commence, and if a low is transmitted, it means that the master intends to send data. The device receiving the data should respond with an acknowledgement (logic low) after every byte, and if no more data will be transmitted or can be accepted, a non-acknowledgement (logic high) should be sent to indicate that the transfer session should end. Timing diagram is further explained in Figure 6.

As it can be seen, transfers are clock driven and writes and reads cannot take place in the same session. Therefore, the data line should never change when the clock line is high. There are two exceptions to this, however; the start and stop conditions. If the data line goes high to low while the clock line is high, it indicates that an address byte will be sent by the master and all slaves should be listening to it in case the transmission is for them. Similarly, a stop condition is indicated by a low-to-high transition while clock is high, which means that a transfer has ended and all devices should release the bus and return to idle state.

Although I²C controllers are available on most modern microcontrollers, it might be more feasible to implement the protocol in software as it is fairly easy to do so.

3.3 RS232/RS485 Asynchronous Serial Protocols

RS-232 is an ANSI standard for serial data communication defined in the ITU-T V.24 standard [14, 15]. RS stands for Recommended Standard. This standard describes the signaling between computers (DTE) and modems (DCE). Over the years, V.24/RS-232C began to be used in other contexts than the computer or other equipment on the one hand and a modem on the other.
RS-232 defines voltages from +3 to +12 V as a logical zero and -3 to -12 V as a logical one (see Fig. 7). In communications 3, 5, 8 or 9 conductors can be used. Three wires are used as data transmit (TxD); receive (RxD) and the common signal ground (SG). Although three wires are all that is needed to transmit data over the protocol, two additional conductors (RTS and CTS) can be utilized to indicate that data is waiting in the DTE transmit buffer and the DCE device is ready to receive data, thereby eliminating the need to send in-band control signals. The remaining four wires are used to show that the devices are ready to communicate (DTR and DSR) that the modem has established contact with the other end of the line (DCD) and the phone is ringing (RI). However, in embedded systems, signaling lines other than the original three are rarely utilized. Maximum RS232 data rate is 921.6 Kbps, although this is rarely utilized and 115.2 Kbps is used instead.

EIA-485, also known as TIA/EIA-485 or RS-485, is a standard that defines the electrical characteristics of transmitters and receivers used in balanced digital multi point systems (multi-drop) [16]. The main difference between RS232 and RS485 is the voltage levels and physical signaling. Since RS485 transmits data on differential pairs, it is much more noise resistant; therefore data communication over longer distances can be achieved. RS485 is commonly utilized in remote monitoring and automatic control applications. The maximum distance that can be achieved over a RS232 line is 300 meters, and that requires high-quality low capacitance cables. On the other hand, an RS485 line can easily achieve communication over 1200 meters.

Since the protocols don’t include a clock line, data must be carefully timed in order for the communication to be successful. Therefore, the protocol should be implemented via a dedicated controller instead of being done in software at the microcontroller end [12].

Figure 7: Transmission of the ASCII uppercase “K” character over an RS232 link as seen on the oscilloscope.
3.4 ZigBee

ZigBee is a high level communication protocol specification which uses low-power low-footprint digital transceivers [13]. On the physical layer, ZigBee highly conforms to the IEEE 802.15.4 standard. Applications include electrical meters with in-home displays, wireless light switches, home automation systems, agricultural sensors and other consumer and industrial equipment that require wireless transfer of data at relatively low bit rates. The technology defined by the ZigBee specification is intended to be simpler, less expensive and more power efficient compared to other wireless personal area networks such as Bluetooth. ZigBee is aimed at wireless communication applications that require a long-range low-power data transfer and comes at the cost of low data rate. The ZigBee Alliance is the organization behind the platform.

The platform supports mesh network, star network and tree structure and mixed variants of these, with the "mesh" technology being the most commonly used (see Fig. 8). In all network topologies, the ZigBee protocol aims to minimize the time period that the radio is powered to reduce power use. Nodes only need to be active while data is being transmitted or received. This allows for extremely low power operation over considerably long distances, with bit rates as high as 250 Kbits per second.

With these qualities, ZigBee is an important protocol for sensor networks and therefore is frequently utilized in data sensing and acquisition systems. Therefore, there are a lot of commercially available ZigBee modules for desktop and embedded systems. Usage of such a module can make an embedded system ZigBee-enabled in next to no time lost in development.

Figure 8: Example ZigBee network operation in mesh topology.
3.5 Wireless LAN (Wi-Fi)

Wi-Fi is a technology for wireless networks [17, 18]. Products working in this standard can work using radio frequencies in the 2.4 GHz and/or 5.0 GHz band without requiring a license, and allows for data transfer rates of up to 125 Mbps.

Wi-Fi is originally a brand name launched by the industry association Wi-Fi Alliance to describe the technology based on standards in the IEEE 802.11 family. Nowadays, often used as a more popular and powerful means to describe wireless LAN ("WLAN") over the IEEE 802.11 standard, Wi-Fi is a certification label (logo) for wireless data networking products that work according to the IEEE 802.11 international standard. In daily practice, Wi-Fi is increasingly used as a synonym for “wireless home network.”

Wi-Fi defines two different topologies: ad hoc and infrastructure. In ad-hoc, a client communicates directly with another client. The maximum distance between these stations is thus automatically limited to the range of the two transmitters / receivers (depending on many factors, however, usually up to about 30 meters). The infrastructure mode works with 802.11 access points or “base stations.” The base stations are connected by another infrastructure and mobile stations can switch from one to another access point ("roaming"), without losing connection to the network, just like cellular mobile phone networks.

Many public locations like airports, hotels and libraries install base stations which the mobile computer user has Internet access at these locations and can make use of information services of the organization. Such (semi-)public base stations are also called access points or hotspots.

The bandwidth and reach of Wi-Fi will be greater than that of Bluetooth. For these reasons, Wi-Fi is, without doubt, the most common access method for ubiquitous wireless Internet. A downside of Wi-Fi, especially compared to ZigBee is the relatively high energy consumption. This is not usually a problem for computers; however this should be kept in mind when developing devices that will be powered with batteries.
In the context of this project, a reference data acquisition platform was built with the utilization of the standards, communication protocols and microcontroller peripheral mechanisms described in the previous chapter. This chapter will focus on the design and implementation aspects of the wireless data acquisition device that was built in context of the work.

The designed data acquisition device uses a Texas Instruments MSP430FR5739 microcontroller as the main embedded processor. The device gathers temperature and humidity data from the onboard Honeywell HIH6131 chip. Likewise, light level data is gathered from Texas Advanced Optoelectronics Solutions’ TSL2550D integrated circuit. All sensors communicate with the main processor over an I²C bus.

For Wi-Fi connectivity, the TiWi-SL wireless LAN module made by LS Research was used (see Fig. 9). The module provides great range while allowing for low power operation and is connected over the SPI bus [19]. An HTTP web server is implemented which serves sensor data and relevant configuration parameters in the form of HTML5 web pages styled with CSS3 style sheets. Some JavaScript was used to offload memory-heavy string operations to the client browser, freeing code space at the microcontroller.

Figure 10: The designed device on the development platform. The main microcontroller, sensors, external flash memory, Wi-Fi module and batteries are clearly visible.

4 Design and Implementation

In the context of this project, a reference data acquisition platform was built with the utilization of the standards, communication protocols and microcontroller peripheral mechanisms described in the previous chapter. This chapter will focus on the design and implementation aspects of the wireless data acquisition device that was built in context of the work.

The designed data acquisition device uses a Texas Instruments MSP430FR5739 microcontroller as the main embedded processor. The device gathers temperature and humidity data from the onboard Honeywell HIH6131 chip. Likewise, light level data is gathered from Texas Advanced Optoelectronics Solutions’ TSL2550D integrated circuit. All sensors communicate with the main processor over an I²C bus.

For Wi-Fi connectivity, the TiWi-SL wireless LAN module made by LS Research was used (see Fig. 9). The module provides great range while allowing for low power operation and is connected over the SPI bus [19]. An HTTP web server is implemented which serves sensor data and relevant configuration parameters in the form of HTML5 web pages styled with CSS3 style sheets. Some JavaScript was used to offload memory-heavy string operations to the client browser, freeing code space at the microcontroller.
The web pages are stored on SST25VF080B flash memory chips manufactured by ST Microelectronics. The chip provides 1 MB in storage space and connects to the microcontroller over the SPI bus. External flash memory allowed fitting a full-featured web server in a small code space while retaining the power management features. The device also includes a 3-slot AA battery pack for true cordless operation. All components on the system run off 3.3 volts. 3 AA batteries give off 4.5 volts and this voltage is regulated down to the required amount with a power regulator. The regulator allows for any supply voltage from 2.1 to 24 volts.

The system was designed, prototyped and developed using Texas Instruments’ FRAM Experimentation Board, which can be observed in Figure 10. Sensors and the flash chip were soldered to an external pin expander and attached to the main board via headers and cables (see Fig.13). The test kit for the wireless module was used for connectivity; which has an onboard antenna and headers for direct fitting onto the wireless expansion slot on the FRAM board. A block diagram is given in Figures 11 and 12; detailed information about the individual components and operation of the device will be given in the following sections.

4.1 MSP430 FRAM Microcontroller

As the main processor of the designed embedded system, a MSP430 FRAM series microcontroller was chosen; MSP430FR5739 to be particular. The following were the main reasons behind this choice:

- Designed for low power applications
- FRAM memory allows for fast and efficient memory operation
- Memory can be configured for code or data as needed
- Wide variety of GPIO pads and USCI peripherals to choose from
- Industry standard Eclipse-based development environment
The MSP430FR5739 has 16 KB of FRAM memory and 1 KB of static RAM memory. Runtime variables which don't require non-volatility were assigned space on the SRAM memory, while system parameters that should be preserved between power cycles and buffers not small enough to fit the SRAM were assigned on the FRAM memory.

Microcontroller has 3 USCI modules; 2 being A-type and the other being B-type. The B-type USCI is used for the Wi-Fi module in SPI configuration. One A-type module is used for connection to the SPI flash memory, while the other is left empty for future asynchronous serial expansion options. Since the sensor data is just a few bytes per second, \( \text{I}^2\text{C} \) communication was implemented in software, in order to leave USCI modules empty for future expansion options (see Fig. 15).

The microcontroller is normally kept at the LPM3 power stage. A timer is configured to wake the processor up 5 times per second to fetch data from the sensor chips. Another interrupt is assigned to the USCI-B module to wake the processor in case a client connects to serve the client. The interrupt driven mechanism of the system allows the main processor to spend just a few microseconds every second in the active stage, therefore saving as much battery power as possible.

4.2 Temperature and Humidity Sensor

Temperature and humidity data readings are provided from the HIH6131 chip which is an \( \text{I}^2\text{C} \) humidity sensor with an optional temperature feature and is manufactured by Honeywell. Both readings are provided in 14 bits sensitivity. The chip is truly \( \text{I}^2\text{C} \) compliant and allows for transfer rates up to 400 Kbps.
The sensor is highly accurate, and is particularly advertised as the most accurate sensor in the market. Humidity readings are accurate up to 5% and automatic temperature correction is possible thanks to the onboard temperature sensor.

While other humidity sensors require a rehydration process after soldering, the HIH6131 can use temperature readings to compensate for dehydration occurred during soldering, thus not requiring such a process and thereby removing one production step from the assembly line [20].

4.3 Light Sensor

The light sensor used in the device is the TSL2550D, manufactured by Texas Advanced Optoelectronic Solutions. It is intended for use in automatic brightness adjusting displays in laptop computers and cell phones.

The sensor communicates through the SMBus interface, which is built on top of the I²C interface, thus being compatible with it. It can measure light levels as ADC count values at 14 bit sensitivity up to a maximum value of 1846 lumens. It takes 800 milliseconds to complete a light reading. It also has an extended range option, increasing the maximum readout limit to over 9000 lumens and lowers the reading time to 160 milliseconds, at the cost of the lower sensitivity of 8 bits.

Due to the sensor transmitting the readings in ADC counts, floating point math operations are necessary to convert the value into human-readable lumens. Although floating point operations require tremendous amounts of memory and processing power, pre-calculating these values into lookup tables eliminates this problem and makes the device suitable for embedded system operation [21].

Figure 13: Close-up of the expansion card built for the project. Chips: (from left to right) TSL2550D, HIH6131 and 25VF080B. Bypass capacitors are added to supply rails to smooth out the supply voltages. The circuit is wired on an IC breakout board.
4.4 External Flash Storage

As of this writing, the storage space on MSP430 FRAM devices are fairly limited – the highest being 16 KB on the MSP430FR5739 which is used in the project. As a result of this, it’s not possible to integrate eye-catching HTML5 content into the microcontroller. To solve this problem, an external flash memory with 1 MB of space was added to the project. All HTML and CSS data are stored on this external storage and streamed from the flash memory as required by the embedded web server.

The flash memory utilized in the project is manufactured by ST Microelectronics and is codenamed 25VF080B [22]. It communicates via the SPI bus. Despite this particular flash memory being utilized in the project, SPI flash memory interfaces are de-facto standards and any SPI flash memory could be used. With the reference implementation using a mere 21 KB of HTML and CSS data, nearly any flash can be thrown in as a replacement.

The flash memory is able to address each individual byte, with addresses ranging from 0x000000 to 0x1FFFFF hexadecimal values. To retrieve a byte, after asserting the select line for the flash memory, the address value is entered, after which the flash memory streams all bytes in order starting from the pointed address until the chip is de-asserted. The web server directly talks to the flash memory via the SPI bus and reads HTML and CSS files out in binary format by reading from their offsets in memory. There is no file system utilized, which reduces code complexity.

Power line for the microcontroller is sourced from a GPIO pin. This allows for keeping the flash memory powered down and only letting it consume power when a web page must be served. The microcontroller takes no part in flashing the SPI memory as well; flash must be programmed with the means of an external SPI flasher. In the project, the Bus Pirate universal serial protocol analyzer built by Sparkfun Electronics was used to emulate an SPI programmer (see Fig. 14).

Figure 14: Bus Pirate universal debugging tool used to program the SPI flash memory. The Bus Pirate is an all-in-one tool to analyze popular serial protocols such as SPI, I2C, 1-Wire, asynchronous serial, MIDI and JTAG.
4.5 Software Implementation

The designed embedded system contains a total of 12288 bytes of code which was mostly written in the C programming language. The software consists of a total of 5821 lines of C code, with the I²C library being implemented in assembly for speed concerns and consisting of 121 ASM instructions. Raw implementation of the I²C driver negates the speed penalty of implementing it in software.

The software consists of the following parts:

- SPI Wireless LAN driver
- SPI bus arbitrator
- HTTP Web server
- Dynamic HTML parser
- CGI function parser
- First time configuration tool
- Sensor data readout library
- Error detection
- Main code to tie it all together

The software is completely interrupt-driven, making it possible to keep the device in standby state and only leaving the state when it needs to read sensor data, check alarm conditions or serve a connecting client; which comprises of only a few microseconds in each second. Allowing the device to stay in standby mode for extended periods of time drastically reduces power footprint without having to sacrifice from functionality.
4.6 First Time Configuration

Upon pulling the device out of its box, the device will look for a specific wireless connection and try to connect to it with pre-defined parameters. These parameters can be reconfigured in software if desired. Default parameters are;

- **Network name:** SensorNetwork
- **Security:** WEP
- **Key:** (last 5 digits of the WLAN module MAC address)

The wireless LAN card driver allows for pulling the factory-configured MAC address of the wireless device. Therefore different devices can be issued different keys without the need of developing code for each individual device.

After a wireless connection has been established, the user can use the web interface to configure the module for his/her own needs (see Fig. 16).
(This page intentionally left blank.)
5 Results and Conclusion

Embedded systems are widely used in various applications that are demanding in manpower without automation with a computer system; an example of which is data sensing and acquisition. Sensor applications are usually done in the way of sensor networks, with lots of data gatherer nodes and a base station that does all the data gathering and reporting.

It was found that while many base stations exist in market for heavy duty and industrial purposes, there are no devices that truly cater to the consumer/small business markets, with the key to success in these markets being low cost and ease of use.

In this project, after investigating various protocols such as RS232, I²C, SPI, Wi-Fi and ZigBee, a generic wireless data acquisition platform is designed and realized.

The implemented device connects to a wireless LAN network over Wi-Fi, starts a web server and serves HTML5 web pages and a CSS3 style sheet from SPI flash memory. The sensor data is collected over an I²C link. Configuration can be done in a truly wireless, cable-free way. Device can run on DC power as well as 3 AA batteries for more than 3 weeks.

The design process yielded an extreme amount of embedded system design and development experience, particularly in the field of low-power design applications. Knowledge gained in several years of college experience was successfully applied and a data acquisition platform was successfully designed and applied.

With 2 serial buses and a whole lot of future expansion options, the device can be implemented in nearly any area of life, with the most promising areas of application being home monitoring, wireless area monitoring and power aware smart grid applications. With proper R&D and integration of universal standards once they emerge, this reference implementation can turn into a successful consumer product which will greatly improve living quality in households for the greater good. Just with small changes to the implementation and brief testing, this product can bring life to a whole another level for science.
6 Further Discussion

As it can be seen in the previous chapters, the project yielded the design of an embedded system-based communication platform. Rather than providing the end user with an all-out data acquisition solution, what was done in this project was implementing a reference platform with basic data sensing functionalities and means of further expansion and improvisation. In this chapter, what was done and what could be done in the future will be discussed and supported with real world usage scenarios and comparisons with related products available on the market as needed.

6.1 ZigBee Expansion

ZigBee is a high level communication protocol specification which uses low-power low-footprint IEEE 802.15.4 compliant digital transceivers. The platform supports mesh network, star network and tree structure and mixed variants of these, with the "mesh" technology being the most commonly used.

In all network topologies, the ZigBee protocol aims to minimize the time period that the radio is powered to reduce power use. Nodes only need to be active while data is being transmitted or received. This allows for extremely low power operation over considerably long distances, with bit rates as high as 250 Kbits per second [13, 23].

With these qualities, ZigBee is an important protocol for sensor networks and therefore is frequently utilized in data sensing and acquisition systems; and as a result, is utilized in many sensor nodes.

The serial bus on the device which is used to connect to the flash memory is rarely utilized, and therefore can be considered unused in the case of connecting a ZigBee transceiver to the device. Due to the ZigBee protocol being power optimized already, addition of a ZigBee module would allow the device to communicate with the vast majority of wireless sensor nodes on the market while not having to cut into the battery life.

6.2 RS232/RS485 Expansion

RS232 or similar asynchronous standards are utilized extensively in remote sensor monitoring applications where transferring a clock signal along with data is not feasible, due to cabling cost and ringing/glitching issues on the received signal. Such an interface is usually used in digital power meters to transmit readings [24] (see Fig. 17). In the design, a USCI serial port was left empty for the sole purpose of implementing this feature. Via this bus, the base station can communicate with sensors that are located several kilometers away [16]. Two-way communication can be achieved with only 3 wires.
A real world example of where this would prove useful would be adjusting a home’s power usage based on readings from a power meter. By downloading the power prices from an Internet-based remote server, the device can be configured to adjust power usage in a household depending on the price of power and necessity of appliances. If an appliance – i.e. a dishwasher – is not absolutely critical to run at that specific time, the device can delay the operation to a later time, thereby reducing power costs to the user.

6.3 Twitter Expansion

Twitter is an online social networking service which lets its users express themselves in messages that cannot exceed 140 characters [25]. According to their own statistics, Twitter has more than 140 million active users by the time of this writing [26]. Integration of the alarm mechanism with Twitter to enable the device to send alarm messages via Twitter will be an appealing feature for the users of the platform.

Twitter uses the OAuth Framework to allow third parties to post tweets on their users’ behalf. OAuth is an authentication framework that allows content owners to grant third parties access to their content without revealing their credentials [27]. In Twitter’s case, this would allow users to be informed of their sensor data via a regular Twitter client.

After authenticating users, Twitter uses the REST API to transfer data. Representational State Transfer (REST) is an IT architecture concept that describes how services for machine to machine communication can be provided [28]. The term is derived from a dissertation by Roy Fielding, one of the authors of the HTTP specification, and has been a proliferation in system area.
REST uses traditional HTTP verbs such as GET, POST, PUT and DELETE; and therefore can be implemented into the HTTP module already built in the device. Such an implementation is currently a work in progress as of this writing, and will be available in a future firmware upgrade.

6.4 Compliance with Universal Standards

As of this writing, every sensor module manufacturer seems to come up with a new standard, with some hundred standards being available on the market. This is an unfortunate situation for base station manufacturers, as the market is oversaturated with similar performing sensor modules with different communication protocols.

The IPSO Alliance is the leading proponent of IP network devices for use in energy, consumer, medical and industrial applications. Based in the European Union, IPSO Alliance is a nonprofit association of more than 60 members from leading technology, communications and energy companies around the world [29].

The alliance maintains promising work in the field of sensor device communication and a standard published would be in favor of many sensor manufacturers around the world. Such a standard can be implemented for broad compatibility once one has been made publicly available.

6.5 Estimated Cost of Manufacturing

As it has been stated before, the device was manufactured with cost kept in mind. All parts for the device can be easily found from major electronic component suppliers. Due to the WLAN card having the baseband and TCP/IP stack implemented in
firmware [19], devices with small memory sizes can be utilized to further reduce the manufacturing costs.

Other aspects that reduce the cost are, the humidity sensor not requiring a rehydration process after soldering. Regular humidity sensors become dehydrated after passing through a soldering machine and need to be rehydrated in order to provide correct values. With Honeywell’s sensor technology, this process is not required [20], therefore bringing down the manufacturing cost.

Another aspect that brings down the cost is the software. Due to the software’s modular nature, it can poll for configuration values from peripheral devices and configure each device with unique values without the need to intervene with the programming process in the assembly line.

With these key design points, the device is extremely simple to manufacture and PCBs can be assembled in a regular 2-step process consisting of fabrication and soldering. Due to the low number of components, the PCB can be made extremely small, therefore reducing the cost even further. Cost estimation is given in Table 3.

<table>
<thead>
<tr>
<th>Component</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSP430 FRAM</td>
<td>43.42</td>
</tr>
<tr>
<td>Wi-Fi Module</td>
<td>179.93</td>
</tr>
<tr>
<td>1MB Flash</td>
<td>9.26</td>
</tr>
<tr>
<td>Temp.&amp;Humid.Sensor</td>
<td>122.70</td>
</tr>
<tr>
<td>Light Sensor</td>
<td>39.17</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>394.48</strong></td>
</tr>
</tbody>
</table>

**Table 3**: Component costs for the designed platform. All prices are in SEK from Farnell Sweden. Prices are current as of 05/13/12. Tax not included. Unit price for 100 pieces.

6.6 Market Comparison

As mentioned before, there are a lot of wireless data acquisition stations available on the market. But, most of these are built for industrial and other heavy duty applications where cost is not a concern. Industrial sites also have access power most, if not all, of the time; therefore these devices are not designed with energy efficiency either. As a result, a similar product found on the market, which can be seen in Figure 18, retails for more than seven times the price of the device designed in this project and have neither battery backup nor expandability options. A detailed comparison is given in Table 4.
The only difference found between this device and the one designed was an Ethernet output; however, given the availability of wireless network in consumer households where the device was designed for, and the first-time configuration module included in the device, a seamless out-of-the-box experience can be achieved without investing in a costly Ethernet port.

**Table 4:** Differences between the designed product and a similar product found on the market.
(This page intentionally left blank.)
References


(This page intentionally left blank.)
Presentation of the author

Özgun Ayaz

Özgun Ayaz is an undergraduate student in Halmstad University School of Information Science, Computer and Electrical Engineering. Born in 1989, he worked with many engineering companies all over the world, including Sweden, Turkey, the United Kingdom and the United States. Apart from embedded systems development in which he is driven by passion, he is interested in hunting and spearfishing; and is a contributor in the Android Open Source Project.