WINNOVATIONS Wireless Innovations For Security
School of Engineering Science ♦ Burnaby, BC
E-mail: ensc440-security@sfu.ca

Friday, March 11, 2005

School of Engineering Science
Simon Fraser University
8888 University Drive
Burnaby, British Columbia
V5A 1S6

Re: ENSC 440 Design Specification for a Wireless Home Security System

Dear Mr. One,

The enclosed document, Design Specification for a Wireless Home Security System, outlines the system design details and description of the technologies used in implementing this solution. Our goal is to design a wireless home security solution that notifies the homeowners via their computer or a cell phone, in case of fire, flood, or break-ins.

Our company, WInnovations, is comprised of four 5th year SFU engineering students. If you have any questions please feel free to contact us at ensc440-security@sfu.ca.

Sincerely,

Gavin Lee
CEO
WInnovations

Enclosure: Design Specification for a Wireless Home Security System

# Design Specification for a Wireless Home Security System

***Project Team*:** Jason Fong, Alex Hsiao, Gavin Lee, Gaurav Magoon

***Contact email:*** ensc440- security@sfu.ca

***Submitted to:*** Lucky One, Mike Sjoerdsma, Scott Logie, Tony Ottaviani, Amir Masoud Niroumand

***Issued date:*** March 11th, 2005

## Executive Summary

In this era of terrorism and increased reporting of break-ins and home invasions, personal security has never been more paramount to society than it is today. In addition, some of today's existing security systems may be insufficient in meeting popular demands. For example, many systems have two popular methods of notification: a siren, and an emergency call to authorities. But what happens when no one can hear the siren? What happens if the phone lines happen to be down in the first place?

In addition, what happens if a fire starts in the house when nobody is home? What happens if flooding somehow occurred within the home? Should the homeowner have to wait until s/he gets home before being aware of what is going on?

At WInnovations, we are carrying out pre-emptive actions to tackle these concerns. Our Wireless Home Security System (WHSS) will feature user notification via e-mail and/or Short Message Service (SMS) messages. The WHSS also touts the ability to sense motion, extreme temperatures, and flooding, making our product stand out among the rest.

This is the third of four documents regarding our project that will be released. The project proposal was released on January 25, 2005. The functional specification, which described the features and possible uses of the Wireless Home Security System, was released on February 22, 2005. The post-mortem will be released shortly after presentation of the project, the latter of which is scheduled for April 13, 2005.

WInnovations was formed by four 5th-year Simon Fraser University engineering students, whose specialties range from computer programming to electronics system design. Put together with effective time management and organizational skills, WInnovations foresees being able to complete this project on time and within budget.

Implementations of the appropriate circuitry and relevant computer programs are underway. Your first opportunity to see our project will be on March 21, 2005, when we expect to demonstrate some working individual models.

This design specification describes some of the inner workings of the Wireless Home Security System. This document is not intended for reading by the customer, although those with appropriate technical knowledge may understand its material. The hardware, software, and communications between them will be discussed in brief technical detail.

We hope that this document will show the progress we have accomplished to date, and in about a month, have a functioning prototype on the way to improving the convenience of your future home security system.

# Table of Contents



# List of Figures

## List of Tables

# 1 Introduction

## 1.1 Scope

This design document outlines the detailed design and approach to implementing the Wireless Home Security System. The details of why the approach was used, how the separate components will be implemented and what parts were chosen will provide sufficient information for the implementation phase of this project.

## 1.2 Acronyms

Table 1 shows the acronyms that will be used in this document:

**Table 1: Acronyms**

| Term | Definition |
|---|---|
| SMS | Short Message Service |
| WHSS | Wireless Home Security System |
| API | Application Program Interface |
| PIR | Pyroelectric Infrared |

# 2 System Overview

In the wireless home security system, sensors will be placed in various settings in a home. Their status will be constantly monitored and if the sensor goes off or has detected a state of danger (such as smoke detector, flood detector, home security, etc), a wireless signal will be sent to a PC via a wireless transmitter that is located in a different location. Upon user preference, the information can be sent from the PC to the user via e-mail or SMS for notification with the assumption that the PC is turned on. Figure 1 shows the conceptual overview of the security system [2].

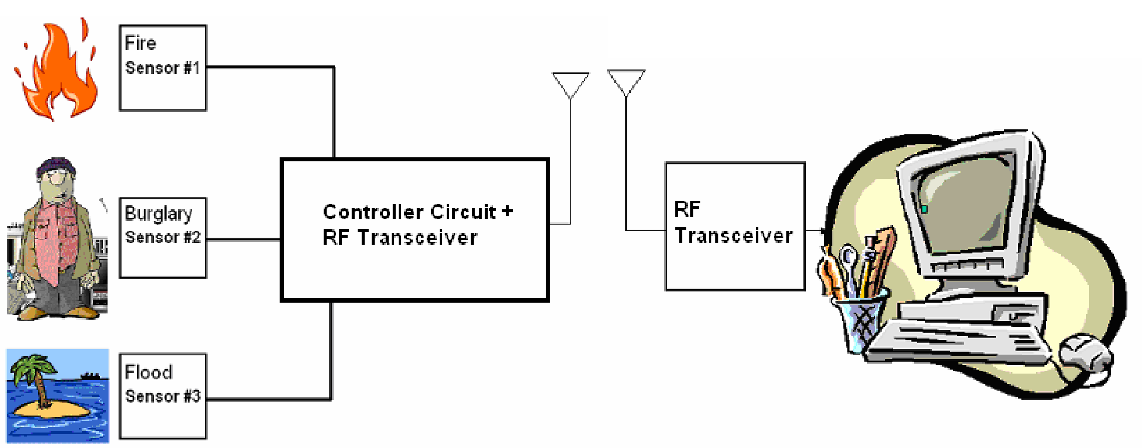


**Figure 1: Conceptual Overview**

# 3 Hardware

## 3.1 Hardware Overview

The design of the hardware sensor system consists of several components. A diagram of the hardware system is shown below in Figure 2.

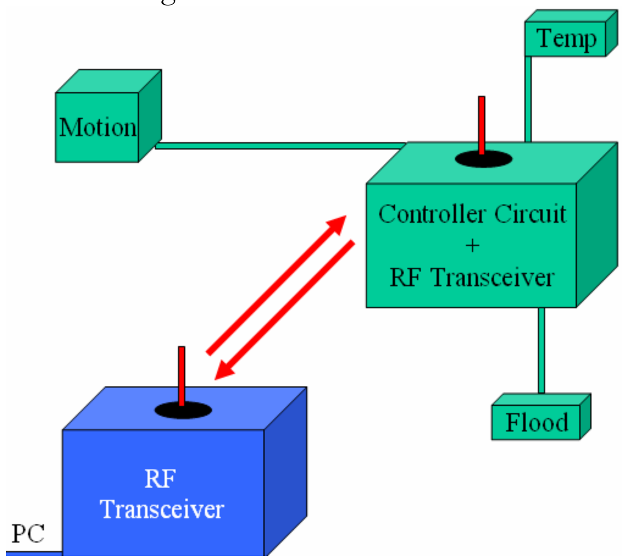


**Figure 2: A physical diagram of the hardware system**

These components are the microcontroller, the RF transceiver, and several sensor modules. First the selection criteria for the microcontroller will be discussed and then the various modules will be discussed. Lastly a system block diagram of the hardware system shall be shown.

## 3.2 Microcontroller

For the purpose of our project, we selected the PIC16F877 microcontroller from Microchip. This chip was selected because we wanted a microcontroller which satisfied the following technical features:

- Had the ability to store a large program in its memory.
- Had the flexibility of allowing more user low level control over the program.
- Consisted of several I/O pins including A/D converters, I2C/SPI, and UART all on the chip.
- It allowed for a range of the frequencies of crystal oscillators that could be used to drive the microcontroller.
- Multiple sources of interrupts including external interrupts.
- Several timer modules which be essential for time bases.
- Ability of In-circuit serial programming.

The PIC16F877 satisfies these requirements by having the following features:

- Only 35 single word instructions to learn.
- Clock speed is from DC – 20 MHz.
- 8K x 14 words of FLASH Program Memory.
- 368 x 8 bytes of Data Memory (RAM).
- 256 x 8 bytes of EEPROM Data Memory.
- Up to 14 sources of interrupts.
- Synchronous Serial Port with SPI and I2C.
- Universal Synchronous Receiver Transmitter (UART/SCI).
- Parallel Slave Port.
- 5 I/O Ports A, B, C, D and E.

Apart from the technical reasons for selecting the PIC16F877 as our microcontroller, the other reasons were:

- Easy to program. Programming the microcontroller should not take too much time.
- Can be expandable when required to add additional sensors.
- This microcontroller also utilizes the PicStart Plus PIC programmer which is an available programming device at the labs in the School of Engineering Science.
- The design of our device is based on portability. Our security device can be placed at any location in a home. This means the device should be small and portable. By using the PIC we are conforming to this requirement.
- This microcontroller is an affordable part for the usage of student projects. If an alternative microcontroller such as the HC12 would have been selected for the project, the size of the microcontroller may have posed some problems in the long run.
- The power supply that will be required for the PIC16F877 would be a standard 5V supply.
- The device has been used previously in projects and there is help and documentation available on the device on the Internet and in books.

## 3.3 Motion Sensor Module

### 3.3.1 Types of Motion Sensors

For motion sensors, the two most widely used types of motion sensors are optical (infrared) and ultrasonic (sound) [5]. The following outlines the key properties of each of the types of motion sensors:

- Optical (pyroelectric infrared or passive infrared) [1]
  - Passive (waits for activity)
  - Dependent on lens used (most common – fresnel lens)
  - Relies on detecting heat energy emitted by humans
  - Optimized to detect at optimal 10 microns wave length (peak wavelength humans emit)
  - Line of sight device (cannot move around corners, etc.)
- Ultrasonic (sound)

- Active (sends signals to look for activity)
- More sensitive than optical sensors
- Susceptible to false alarms
- No blind spots or gaps in area of coverage
- Operate at frequencies above 20Hz (human movement)
  - 25, 30, 40 KHz
- Hybrid (infrared and ultrasonic)
  - Minimizes false alarms

### 3.3.2 Determining Type of Motion Sensor

Having looked at the various attributes of each of the motion sensors, the following table summarizes each of the possible types of motion sensors WInnovations could possibly use.

**Table 2: Type of Motion Sensors**

| Requirements (in order of priority) | PIR (pyroelectric infrared) | Ultrasonic | Hybrid (PIR + Ultrasonic) |
|---|---|---|---|
| Viewing Angle | Depending on PIR sensor as well as objects in the path of the sensor | Can provide a much larger viewing angle since it can be used around corners | Can provide a much larger viewing angle since it can be used around corners |
| Sensitivity | Only sensitive to sources emitting between 5-14 micrometer wavelengths of infrared light (10 micrometers is peak for humans) | Highly sensitive to changes in the external environment and can often result in false alarms | Uses a combination of infrared and ultrasonic to minimize the potential for false alarms |
| Cost (CDN dollars) | ~$2 to $12 | ~$100 | $2 + $100 = $102 |
| Temperature (external environment) | Sensitive to external environment with increasing temperatures | Is not sensitive to temperature in the external environment | Combination of the two |
| Power Supply | Consumes ~5V (depending on voltage to power PIR and associated circuit) | Consumes ~12V to power ultrasonic source | Combination of the two |
| Humidity | Can lead to lower reliability | Can lead to lower reliability | Can lead to lower reliability |

Based on the functional requirements, the best solution that strikes a balance between cost and the functionality for the security system is the optical pyroelectric infrared (PIR) sensor. It provides a low cost solution that does not compromise the primary functionality of detecting motion. Since the main objective of the security system is to prevent intruders from breaking and entering residential areas, the primarily concern is with detecting human activity. Since the pyroelectric infrared sensor is effective in detecting human activity, is provides a sufficient solution that meets our requirements. The other two viable solutions, on the other hand, are over budget and are more prone to false alarms, which is not desirable for our solution.

### 3.3.3 Sensor Evaluation

Having narrowed down the type of motion sensor to be used for the motion sensor module in the previous section, the focus of this section is to evaluate the details of different PIR sensors. A quick glimpse of the various sensors based on their features, is shown in Table 2.

**Table 3: Different PIR Sensors**

| Specifications | RE200B pyroelectric infrared sensor [8] | PIR325 [7] | Model 442-3 IR-EYE |
|---|---|---|---|
| Spectral Response (infrared wavelength) | 5-14 µm | 5-14 µm | N/A |
| Noise (µVpp) | 0.3 | 20 | 0.36 $mV/Hz^{1/2}$ |
| Supply Voltage | 2 to 10 volts | 2 to 15 volts | 5 to 15 volts |
| Operating Temperature (ºC) | -30 to 70 | -30 to 70 | -40 to 70 |
| Composition | N/A | Crystalline material | Lithium Tantalate |
| Element type | Dual | Dual | Dual |
| Viewing Angle | N/A | 95 degrees | N/A |
| Cost | $1.76 US | N/A | $60.00 US |

The dual element PIR sensors are low cost and generic to provide multipurpose use in a wide variety of applications. Although these PIR sensors can be used in different applications, its spectral response of 5 to 14 micrometers is specific for detecting human movement. Human heat emissions peak at 10 micrometers which is within the spectral response range for the dual element PIR sensor.

### 3.3.4 Fresnel Lens

To achieve a wide viewing angle with minimal space required, a Fresnel lens is ideal for this application. The Fresnel lens is a sheet of material with fine grooves that approximates the divergence of light characteristics of the plano-convex lens. In addition, the cost of creating a plano-convex lens is much higher compared with the Fresnel lens as it minimizes the amount of lens material used. Figure 3 shows a visual comparison of the two lenses.

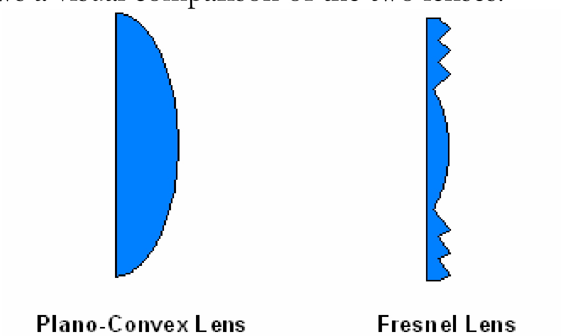


**Figure 3: Lens Comparison**

A comparison of the different Fresnel lenses with varying viewing angles and distances is shown in Table 3. [8].

**Table 4: Various Fresnel Lenses**

| Lens Model No. | Viewing Angle | Maximum Distance (m) | Distance to sensor |
|---|---|---|---|

| | | | |
|---|---|---|---|
| MST-101 | 101º | 33 | 1.2” |
| MST-101-EW1 | 141º | 50 | 1.2” |
| MST-101-LR1 | 12-27º | 100 | 1.2” |

As the requirements specification for the WHSS specifies, a minimum of 100º viewing angle at a minimum distance of 50 meters. Based on the table of the various lenses, the MST-101-EW1 Fresnel lens best matches the requirements for the motion sensor module.

### 3.3.5 Sensor Module Design

Having narrowed down the type of sensor, the design of the sensor is the main focus. The output of the PIR sensor is a low amplitude voltage signal representing the change in the detection of infrared light. Therefore, to have a signal that is seldom affected by noise, the signal needs to be filtered and amplified. A higher level diagram of the schematic is shown in Figure 5.

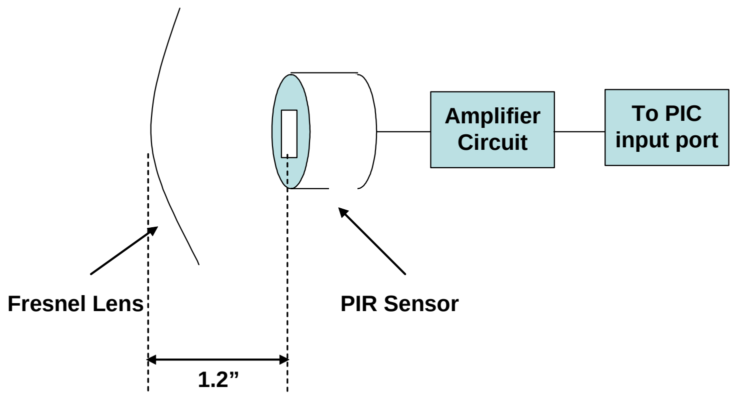


**Figure 4: Higher Level Motion Diagram of Motion Sensor Module**

The sensor module will be powered by a 5 volt centralized power source shared among the other sensor modules and the main processor, the PIC chip. The output of the PIR sensor is a 0.077 volt DC signal. When motion is detected based on the infrared light between 5 and 14 µm, the output is a 0.079 volts DC signal, which translates into a 2 mV change. As a result, the amplifier will need to be designed to amplify a non-inverted signal by a factor of at least 5 volts / 0.077 volts = 63.29 to receive an output of at 5 volts. Figure 5 illustrates the detailed schematic of the motion sensor module.

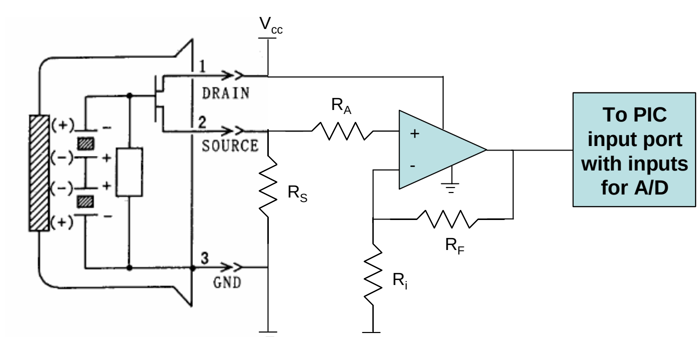


**Figure 5: Schematic Diagram of Motion Sensor Module**

The non-inverting amplifier shown in Figure 6 will consist of an LM324 amplifier. The LM324 amplifier was primarily chosen for its single supply in the range of 3 to 32 volts [9], which meets the requirement for a 5 volt source to power the sensor module.

### 3.4 Temperature Sensor Module

The selection of the temperature sensor was based on a set of criteria. These requirements are mentioned as follows:

- Resolution/Sensitivity
- Rate of measurement
- Temperature range
- Environmental weather conditions
- Environmental positioning
- Distance range
- Sampling rate
- Interface
- Low Power Consumption
- Portability
- Size
- Weight
- Digital Output
- Usable with Microcontroller
- Linear
- Availability of Samples

A set of devices were compared for selection based on these requirements. Table 4 shows the comparison between these devices.

**Table 5: Temperature sensor selection table**

| Criteria | Choice #1 | Choice #2 | Choice #3 | Choice #4 |
|---|---|---|---|---|
| IC Name | MAX6625 | AD7416 | LM75 | MAX7500 |
| Resolution/Sensitivity | 9 bits | 10 bits | 9 bits | 9 bits |
| Temperature Range (C) | -55 to 125 | -40 to 125 | -55 to 125 | -55 to 125 |
| Positioning Conditions | NA | NA | NA | NA |
| Distance Range | I2C Dependent | I2C Dependent | I2C Dependent | I2C Dependent |
| Size | NA | NA | NA | NA |
| Accuracy (+/- C) | 4 | 1 | 2 | 2 |
| Weight | NA | NA | NA | NA |
| Interface Type | I2C | I2C | I2C | I2C |
| Type: IC/Device | IC | IC | IC | IC |
| Package Type | SOT23 | SOIC | MINI SOP-8 | 8 SO |
| Supply Voltage Range | 3.0 to 5.5 | 2.7 to 5.5 | 3.0 to 5.5 | 3.0 to 5.5 |
| Linear | Yes | Yes | Yes | Yes |
| Low Power Consumption | Yes | Yes | No | No |

| | | | | |
|---|---|---|---|---|
| Usable with Microcontroller | Yes | Yes | Yes | Yes |
| Digital Output | Yes | Yes | Yes | Yes |
| Sample Availability | Yes | Yes | Yes | Yes |

Based on Table 4, the selected choice for the temperature sensor was the AD7416 IC. This device is based on the I2C protocol. This is the protocol used to communicate with the microcontroller PIC16F877 as selected previously.

### 3.5 Flood Sensor Module

The approach to developing a flood sensor is to use the buoyancy effect to cause an output trigger signal. This signal to the PIC processor will notify the user through the transceiver modules. Figure 6 shows a diagram of the flood sensor. The casing of the flood sensor will be a clear casing to allow for a photo sensor receiving light from a LED. When water enters the bottom of the casing, the buoyant object (represented by the blue block) will move upwards. This movement will block the light emitted from the LED to reach the photo sensor, thus triggering a change in voltage of the circuit. Figure 7 is a diagram of the dimensions of the sensor module's clear plastic casing. Figure 8 shows the flood sensor schematic with the signal amplified.

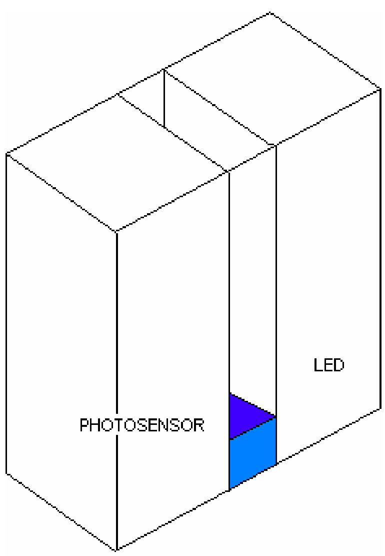


**Figure 6: Flood Sensor Diagram**

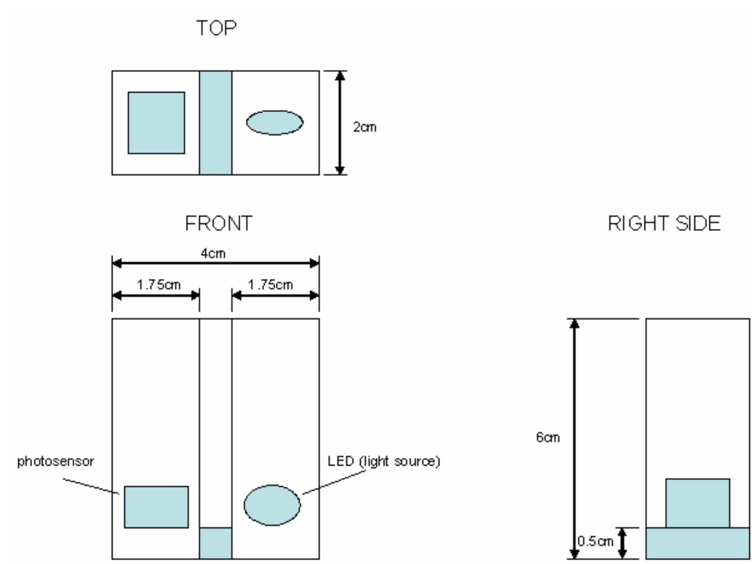


**Figure 7: Flood Sensor Dimensions**

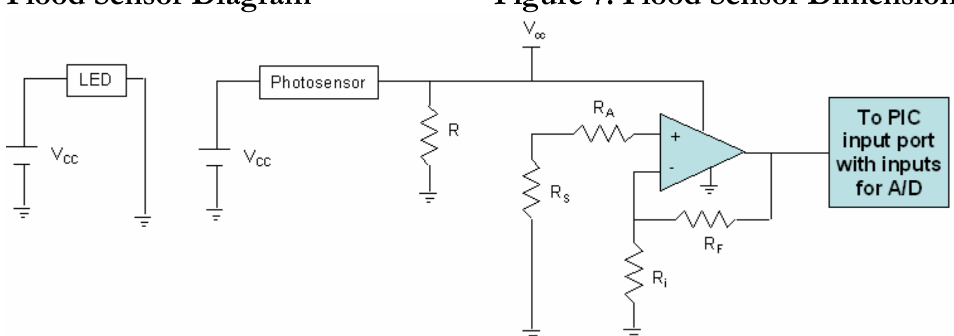


**Figure 8: Flood Sensor Schematic**

### 3.5.1 Design Options

The flood sensor is an interesting sensor in that there are no real commercially available solutions that are widely used in industry. An important consideration with the design of the flood sensor module is that it must be waterproof despite having the possibility of having it exposed to water.

## 3.6 Other hardware components

Figure 10 shows the hardware system block diagram of the microcontroller with the rest of the hardware components as well as the RF transceiver module.

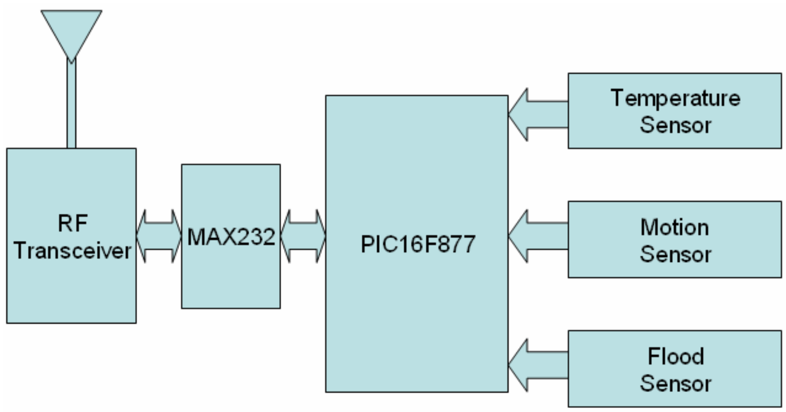


**Figure 9: Hardware system block diagram**

For the communication between the RF transceiver and the microcontroller, the serial port was used because RS-232 is quite reliable. In RS-232, data is reliable because it is represented using high voltage signals (+/- 12 Volts). Higher voltage levels tend to degrade less over longer distances.

A popular chip called Max 232 from Maxim exists which allows for the conversion of microcontroller level signals to RS-232 format signals. The data is transmitted serially.

This chip was selected based on the following reasons:

- Usage of the chip is very simple
- It is affordable
- Can be interfaced to a microcontroller quite easily

Figure 10 below shows the schematic for the microcontroller and the Max 232 IC.

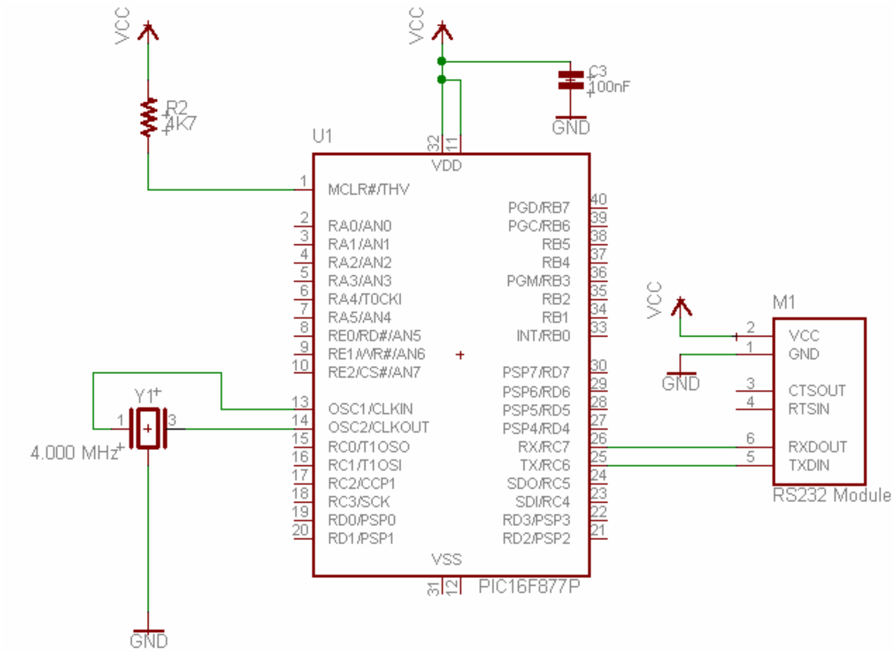


**Figure 10: A schematic showing the PIC and MAX232 (RS232 Module)**

### 3.6.1 Other hardware modules: RF Transceiver

There are two RF transceivers that are part of a wireless communications kit that come with ML2724, a RF kit from Micro Linear. These RF transceivers operate at 2.4 GHz and operate through the serial port of a computer.

The usage of the RF transceiver is based on a few commands. These commands have been mentioned below in Table 5 along with the descriptions.

**Table 6: RF Transceiver commands**

| Commands | Description |
|---|---|
| at+xcen=N | Enables the transceiver |
| at+rxon=N | Selects transmit or receive mode |
| at+dest=NN | Sets the destination address |
| at+chat=S | Sends the message S |
| at+cvr=N | Sets the strength of the reception signal |

These commands, also known as radio modem commands are used to configure the RF transceiver. These commands will be called from the PIC16F877 to configure the wireless connection.

# 4 Firmware Design

## 4.1 Overview

The software design of the PIC chip is crucial for interfacing with the sensors, A/D conversion, interrupt handling, message encoding/decoding, and serial communication for message passing. In the following sections we will explore the design of the overall PIC system as well as the design for individual modules.

### 4.2 Overall System

Figure 11 shows the block diagram for the PIC system. The major I/O devices, namely, the three sensors and the transceiver, are to communicate with internal modules on the PIC chip. The Timer2 module is designed to generate time delays to read sensor data periodically by using timer interrupts. When a desired time delay is reached, it calls the I2C module or the A/D converter to start reading data from the temperature sensor and the motion sensor respectively. Since the flood sensor is designed to use an output relay, the external interrupt will be serviced when the sensor output goes high.

After sensor data is collected, the message encoder is called to encode a message according to the message encoding scheme. The message will then be transmitted via the serial communication module (UART) to the transceiver. The outgoing messages contain sensor, event, and data information. The host side (PC) will decode and process the messages when they are received.

The serial communication module can also receive incoming messages from the host computer. When a message is received, the message decode will be called to decode and carry out necessary actions. The incoming messages from the PC are for the purpose of resetting, disabling, or re-enabling the sensors or the PIC chip.

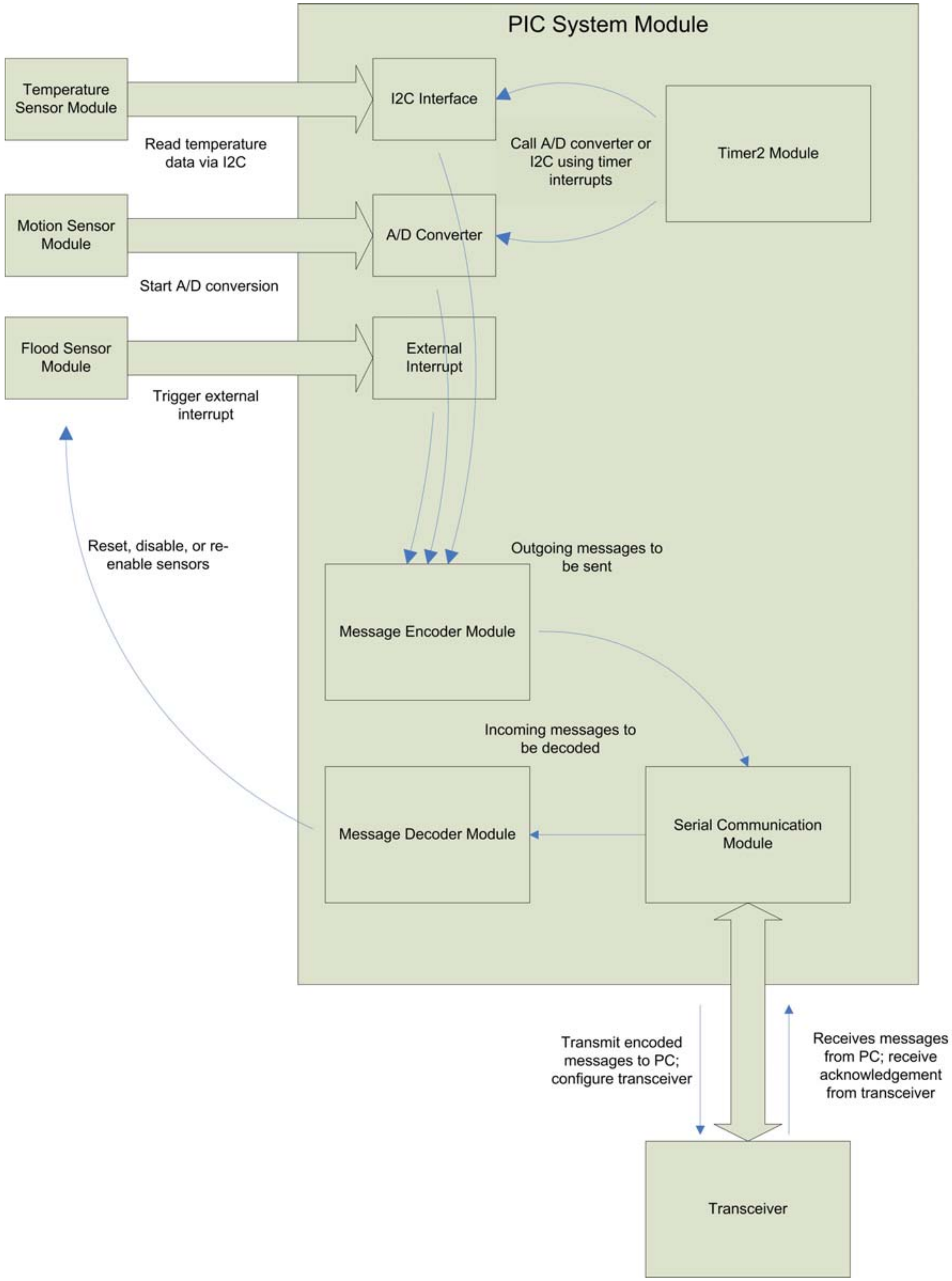


**Figure 11: PIC System Block Diagram**

### 4.3 Timer2 Module

The PIC16F877 chip has 3 timer modules, however, only Timer2 module has a period register and a comparator to generate longer delays by reducing the overhead of servicing frequent timer interrupts. The purpose of the time delay is to periodically read motion sensor data very 0.2 second and temperature data every 1 second. The following calculation shows how the time delays can be achieved.

- The external oscillator operates at $FOSC = 3.6864MHz$
- Timer2 divides the clock frequency by 4 by default and we further slow down the clock by using a prescaler setting at 1:16: $FOSC/4/16 = 57.6kHz$
- The timer register increments every $1/57.6kHz = 17.3611\mu s$
- Timer2 interrupt is triggered every $17.3611 \cdot (2^8 - 1) = 4.43ms$
- A counter preloaded with a value of 45 can generate $4.43ms \cdot 45 \approx 200ms = 0.2s$ delay
- With another counter preloaded with a value of 5 can generate $0.2s \cdot 5 = 1s$ delay

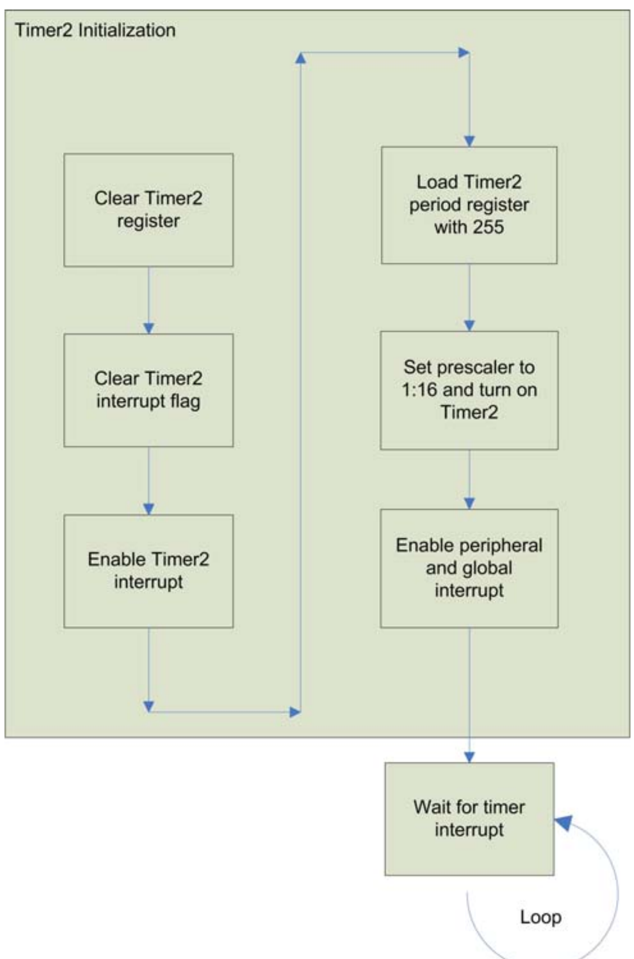


**Figure 12: Flow Diagram for Timer2 Initialization**

Figure 13 shows the sequence of how Timer2 module is configured. After the timer has been initialized, it simply waits for interrupts to occur. Figure 11 shows how the timer interrupt is serviced. Counter1 is used to generate a 0.2s delay, and Counter2 is used to generate a 1s delay.

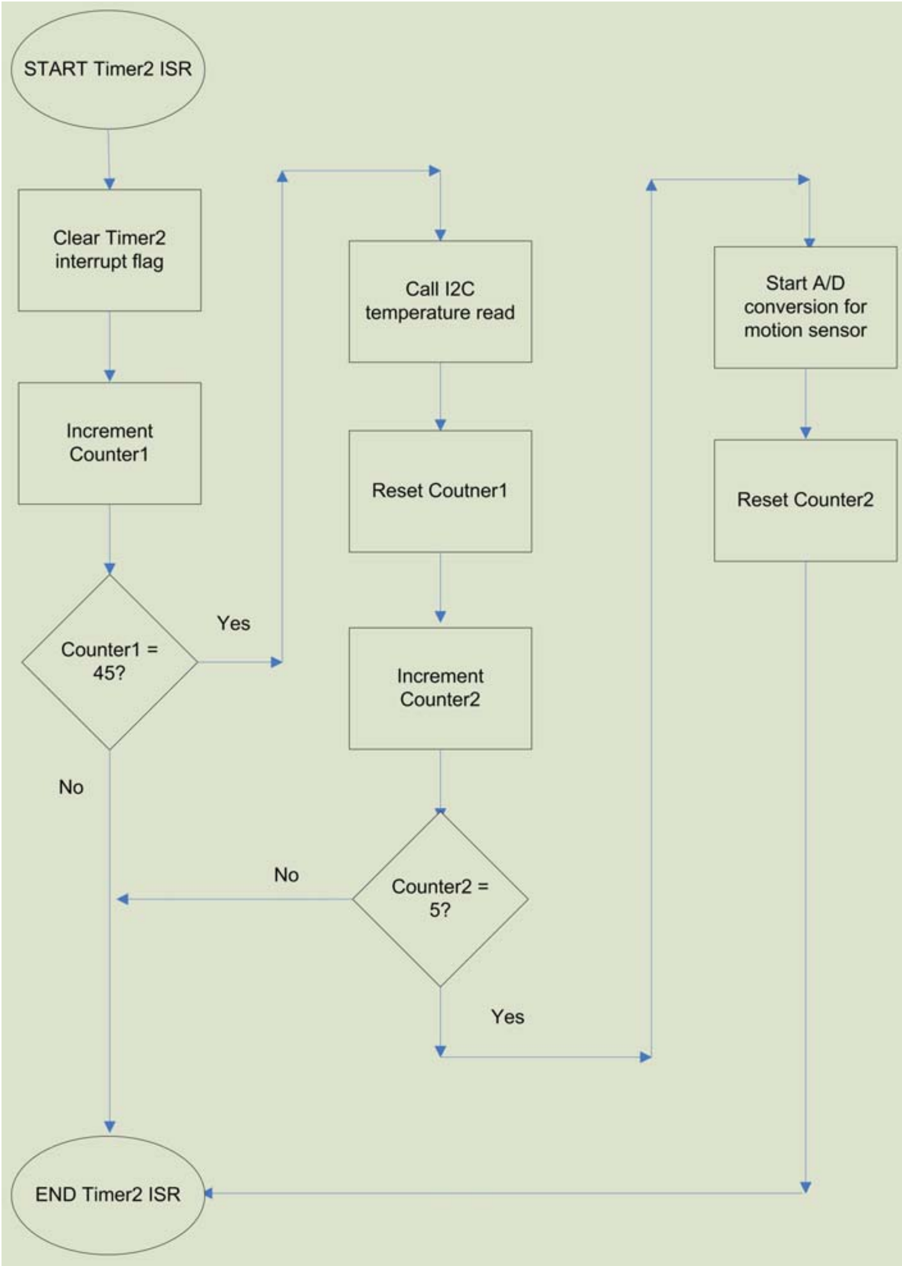


**Figure 13: Timer2 Interrupt Service Routine Flow Diagram**

### 4.4 Message Encoder

After sensor data has been retrieved via I2C or A/D converter, the message encoder is called to encode the data into a message with standard format. Table 6-9 shows the message encoding scheme.

**Table 7: Message High Byte**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Message | | | | | | | | |
| Description | Sensor ID | | | Event ID | | | 10-bit Data | |

**Table 8: Message Low Byte**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Message | | | | | | | | |
| Description | 10-bit Data | | | | | | | |

**Table 9: Sensor ID**

| Bits | | | Sensor |
|---|---|---|---|
| 0 | 0 | 1 | Temperature sensor |
| 0 | 1 | 0 | Motion sensor |
| 0 | 1 | 1 | Flood sensor |
| Others | | | Reserved |

**Table 10: Event ID**

| Bits | | | Sensor |
|---|---|---|---|
| 0 | 0 | 0 | No event |
| 0 | 0 | 1 | Alarm triggered |
| Others | | | Reserved |

Since the A/D conversion will generate a 10-bit data, and the $I^2C$ interface will generate a 9-bit data, we reserve a 10-bit data field within a 16-bit message. To encode a message for the flood sensor, we keep all the 10 bits of the data field 0 since there is no data to be transmitted. For example, an encoded 2-byte message for a flood sensor should be '01100100' and '00000000' in binary.

### 4.5 Message Decoder

The message decoder is called upon receiving messages from the host (PC). The standard incoming message format for our system is set to be 1 byte in length. Table 10-12 shows the incoming message format.

**Table 11: Incoming Message**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Message | | | | | | | | |
| Description | Sensor ID | | | | Event ID | | | |

**Table 12: Sensor ID**

| Bits | | | | Sensor |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | All sensors |
| 0 | 0 | 0 | 1 | Temperature sensor |
| 0 | 0 | 1 | 0 | Motion sensor |
| 0 | 0 | 1 | 1 | Flood sensor |
| Others | | | | Reserved |

**Table 13: Event ID**

| Bits | | | | Event |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Reset/Enable |
| 0 | 0 | 0 | 1 | Disable |
| Others | | | | Reserved |

After a message is decoded, the corresponding action will be carried out immediately. In case of disabling sensors, "sensor enable bit" will be cleared so no data will be transmitted until it is re-enabled. To re-enable a sensor, the message with event ID '0000' should be used to clear the message buffer and set the sensor enable bit to 1.

## 4.6 Serial Communication Module & RF Transceiver

The serial communication module uses the UART (universal asynchronous receiver transmitter) on the PIC16F877 chip. The UART is used to transmit encoded messages to the transceiver, as well as receiving incoming messages sent from the host or acknowledgements from the transceiver.

When the PIC is first initialized, the UART is used to transmit necessary commands to turn on and configure the RF transceiver (See Section 3.6.1 for detail). During idle state, the PIC must configure the RF transceiver to operate in "receiving mode" in order to receive any incoming messages from the host side. Before an encoded message is transmitted, the transceiver must be set to "Transmit mode" and then transmit the message, and toggled back to "Receive mode".

Figure 14 shows how the UART Receive interrupt is handled.

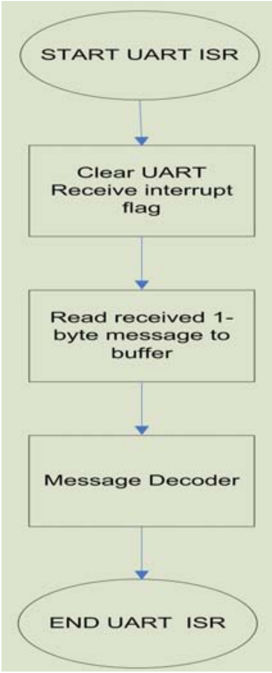


**Figure 14: UART Interrupt Service Routine Flow Diagram**

# 5 Software Design

### 5.1 Software Overview

The main purpose of the PC software is to act as a friendly, graphical user interface between the user and the WHSS. It is capable of displaying sensor statuses, current temperature, and an event log (which will be stored in database format – see 5.2: Database Design).

One of the purposes of the PC software is to receive signals sent from the sensors (via the PIC). The PC software will decode the message, in the process determining which sensor the signal pertains to, whether an alarm was triggered, and the temperature value. If an alarm is triggered, the PC software is capable of sending e-mails or SMS messages. See section 5.4: Communications, for more info on how this is done. It will also be possible to send data to the PIC, but this will happen far less frequently than vice versa. The only reasons the PIC will need to receive data from the PC software are to reset or disable a sensor. Figure 15 illustrates an overview of the main sections of the software in relation to the functionality described.

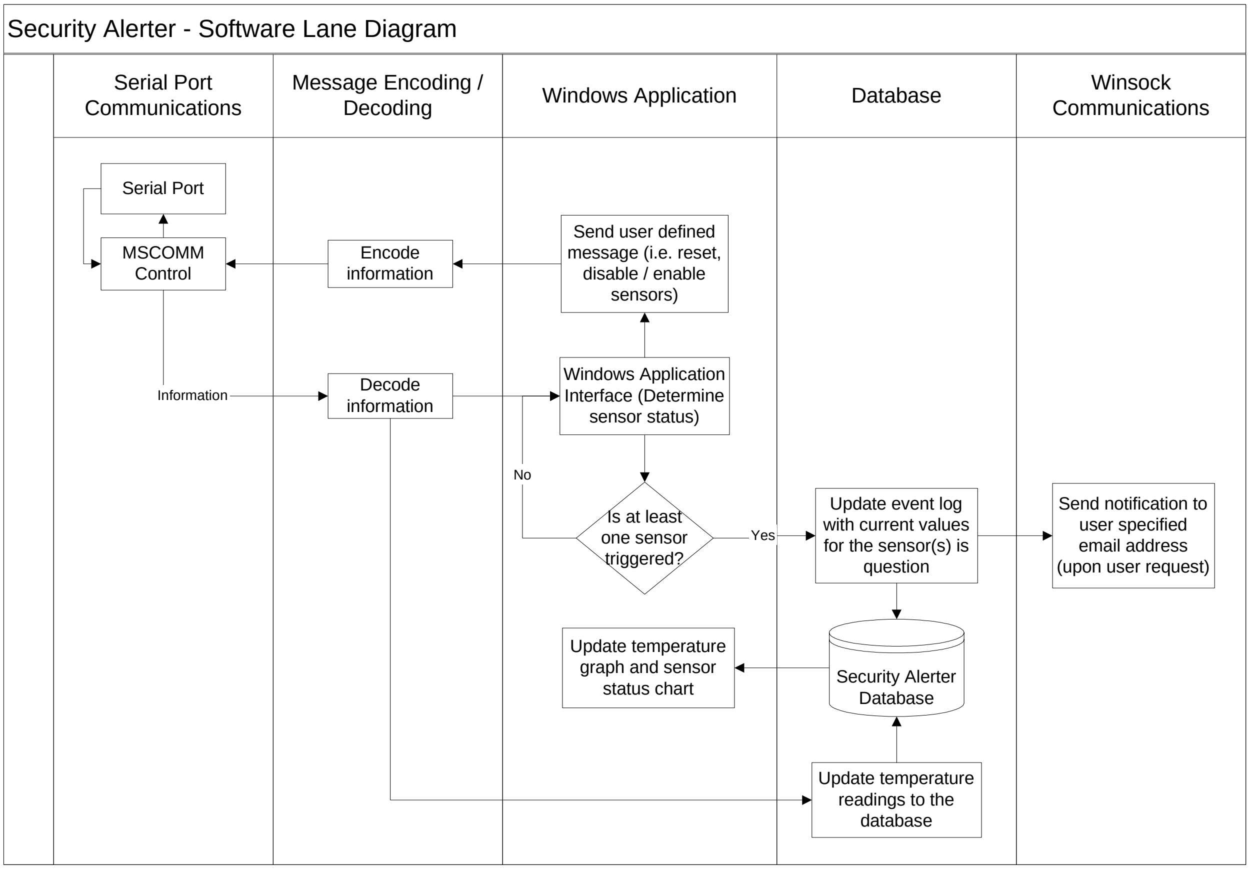


**Figure 15: Security Alerter - Software Lane Diagram**

A detailed description of each of the main software sections are described in the subsequent subsections.

## 5.2 IDE (Integrated Development Environment)

Visual Basic 6.0 is the development platform used for development of the PC software due to ease of learning and convenience with serial port communications. Visual Basic provides the following an outline of the components and features that are applicable to the software application requirements specified in the functional specifications document:

- Provides compatibility with all PCs with Windows OS (using existing Windows libraries)
- Contains COM Serial port component access using Win32 API
- Allows for ADO (ActiveX Data Objects) compatibility with data sources for database access
- Uses Winsock control to facilitate sending emails
- Provides an interactive user interface with event based functionality

## 5.3 Data Storage Options

Storage of information such as user profiles, data logs and application configuration settings is critical for security systems. Table 13 shows the possible data storage options that can be used to store data.

**Table 14: Data Storage Options**

| Criteria | Database | Text File |
|---|---|---|
| **Ease of Implementation** | ADO (ActiveX Data Objects) built into the IDE | Develop own algorithm and standards to store and retrieve data |
| **Password security** | More secure as there are 2 levels of authentication:<br>1. Database authentication<br>2. Application authentication | Password readily exposed in the text file (unless encrypted) |
| **Organization** | Well organized and structured and can be modified to accommodate future changes to the data structure | Hard coded and often difficult to make changes to its already defined structure |
| **Speed** | Data can be queried very quickly based on certain criteria | Search for certain records may be difficult and may involve traversing through text data which is not very efficient |
| **Expandability** | Limitless (dependent on hard disk space) | Limitless (dependent on hard disk space) |

Based on the storage options and the criteria listed in Table 3, the choice of using a database to store information is advantageous. The organization and ease of implementation using databases are particularly important in developing a sustainable application.

### 5.3.1 Database Design

There will be one main Microsoft Access 97 database to store all relevant data – it is named SecurityAlerter. Note that this database may not be directly editable in other versions of Access. This database consists of eight database tables, namely LogEvent, Param, SecurityAlerter, Sensor, SensorStatus, TempData, User, and UserRights. Below are a series of tables describing each column in each database table.

**Table 15: LogEvent – all logged events are stored here**

| Column | Description |
|---|---|
| **DateTime** | Date and time of event |
| **Description** | What happened |
| **LogEventID** | The event ID number |
| **SensorID** | The sensor ID number |
| **Notification** | Description of notification sent to user |

**Table 16: Param – stores serial port communication parameters**

| Column | Description |
|---|---|
| **ParamID** | The parameter ID number |
| **ParamName** | Short description of the parameter |
| **ParamValue** | The value of the parameter associated with *ParamName* |
| **RowLastUpdated** | The date of the last update of the value. |

**Table 17: Sensor – stores sensor information**

| Column | Description |
|---|---|
| **SensorID** | The sensor ID number |
| **SensorName** | Short description of the sensor |
| **ThresholdUpper** | Upper limit of CurrentValue before sensor status changes* |
| **ThresholdLower** | Lower limit of CurrentValue before sensor status changes* |
| **CurrentValue** | The current value of the sensor* |
| **Triggered** | Set if the sensor is triggered |
| **SensorStatusID** | The current status of the sensor |

* - only for the temperature sensor

**Table 18: SensorStatus – stores statuses of the sensor**

| Column | Description |
|---|---|
| **StatusID** | The status ID number |
| **State** | Short description of the sensor |

**Table 19: TempData – stores temperature readings**

| Column | Description |
|---|---|
| **DateTime** | The date and time of the temperature reading |
| **Temperature** | Actual temperature reading |

**Table 20: User – stores registered users and their access rights**

| Column | Description |
|---|---|
| **UserID** | The user ID number |
| **Username** | User login ID |
| **Password** | Password for login ID |
| **Description** | Comments for this user |
| **UserRightsID** | The rights ID number<br>(what rights does this user have) |

Figure 16 illustrates the one-to-many relationship of the database tables and the associated fields, primary keys and foreign keys respective to each of the tables.

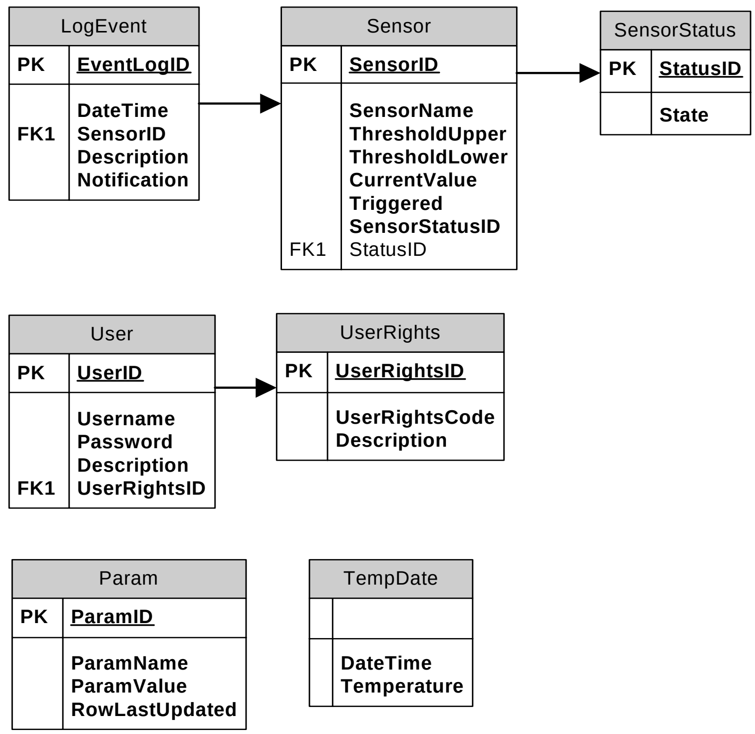


**Figure 16: Entity Relationship Diagram**

The primary key represents the record identifier associated with the table and the foreign key is the record identifier for a linked table. By defining tables based on encapsulating the different objects in the WHSS, a more organized structure is presented for ease of designing and coding for the Visual Basic application.

## 5.4 Application Interface

The WHSS software will be run as a Windows program. It is possible to minimize it to the system tray. The main screen will have two tabs: one for the event log and one for security information (consisting of the sensors, their states, and a graph of recent temperatures – the time period can be determined by the user).

A menu system toward the top of the application sports more advanced features. They include:

- Event log database file location
- User administration
- Temperature reading frequency
- E-mail and SMS settings
- Sensor testing

### 5.4.1 Application Interface Design

Figure 17 shows the sensor information along with a graph to read temperatures as they are received by the temperature sensor module. The information shown in "System Overview" is obtained from the database which stores the current states of the sensors.

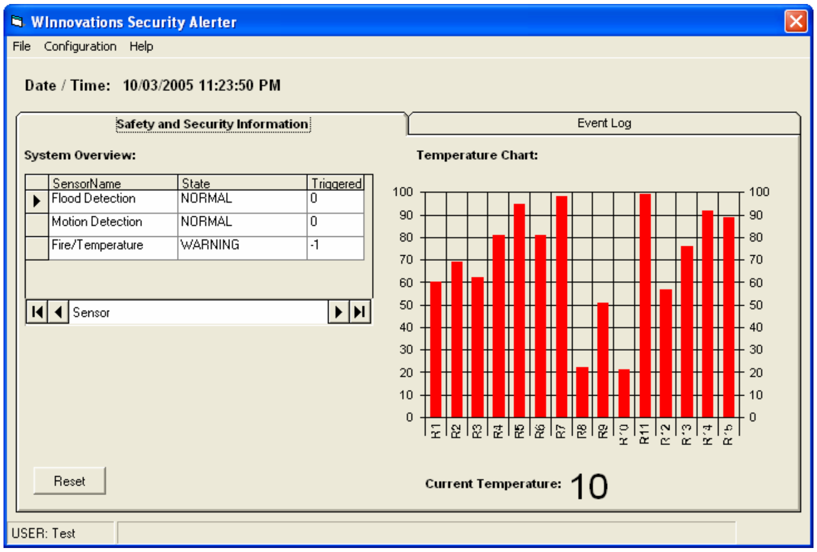


Figure 17: Safety and Security Information

The event logs are a critical part of the security system by allowing the user to look at previous events that have occurred in the system. By allowing the user to view historical sensor information based on a date range and sensor type, the user view critical events with ease. Figure 18 illustrates a mock-up of the user interface for displaying the exporting the event logs to a file.

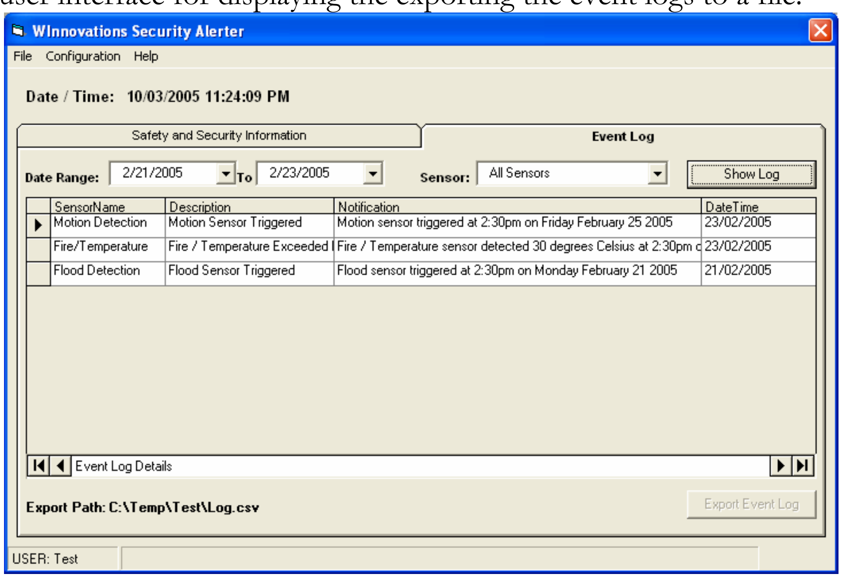


Figure 18: Event Log

## 5.5 Message Encoding / Decoding

Message encoding is converting a string of binary digits to a string of ASCII characters. Message decoding is converting a string of ASCII characters to a string of binary digits. There are functions that can perform both of these tasks.

Data is received from the PIC via the serial port in ASCII format. All characters will be encoded into binary format, which corresponds to a sensor, an event, and some data. The host program will decide what to do with it.

If the host program sends data back to the PIC (e.g. disabling or resetting a sensor), then message decoding will be required. This data, now in ASCII, will be sent via the serial port to the PIC for processing there.

### 5.5.1 Form walkthrough

This section is a walkthrough of all the forms in the WHSS software. It is subject to change, but this is generally what the software will look like to the end user.

When the WHSS software is run, a main form will be displayed with two tabs (the Visual Basic object SSTab "Safety and Security Information" tab is shown).

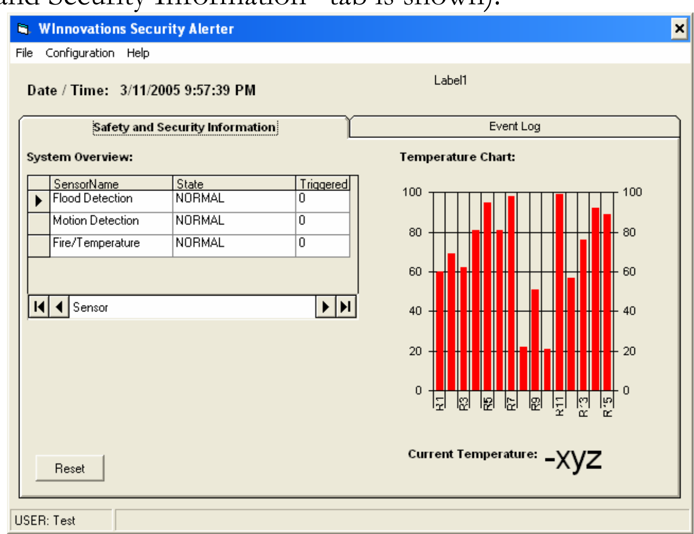


**Figure 19: Main Form with "Safety and Security Information" Tab Open**

In the "Safety and Security Information" tab, the user can check the status of each of the connected sensors. In addition, it is also possible to toggle between sensors using the horizontal scroll, and "reset" the sensor by pressing the reset button, located on the bottom left portion of the form.

On the right hand side, there is a graph of recent temperature readings. Every one second, temperature data is fed in, resulting in a new entry with the date and time of the reading in addition to the sensed temperature.

The WHSS will check for "old" data. The database will be checked for such data when the program first loads, and also at a scheduled daily time (this way if the user runs the program continuously for several days, the database doesn't get too large). "Old" data is defined as temperature readings that are older than three days.

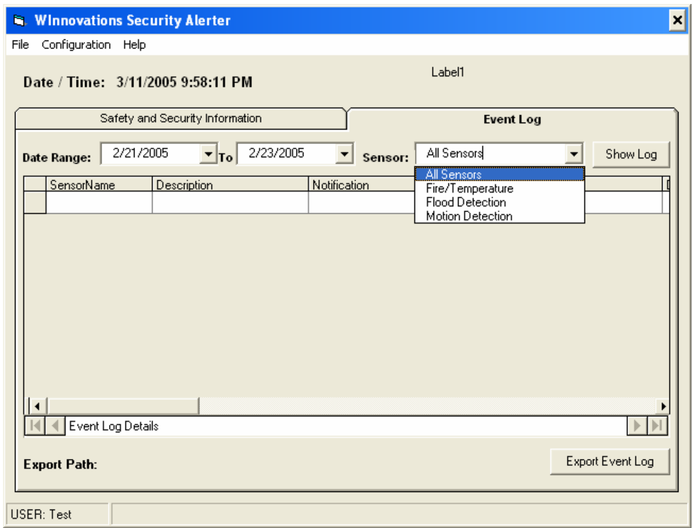


**Figure 20: Main Form with "Event Log" Tab Open**

The "event log" tab allows the user to access past events that have occurred. As can be seen toward the top of the tab, it is possible to filter the database of events to fit inside a date range, and whether or not to select a sensor. It is also possible to display events happening to any of the sensors.

When the "Show Log" button is pressed, a list of events extracted from the database is shown in the main window. Each event is represented by one table row, with information such as sensor name, description of the event, and details of any notification sent to the user. In addition, the "Export Event Log" button opens another common dialog that allows a plain text file of the events to be written to the target file destination.

Now we'll run through the menus. File → Exit, obviously, quits the WHSS program.
Under Configuration → Alerter Settings, an Alert Form pops up with three tabs, as shown below.

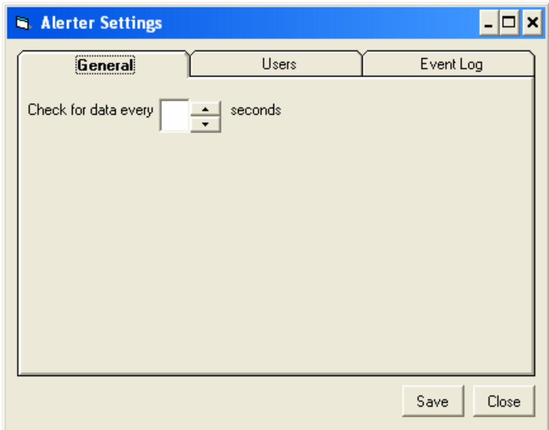


Figure 21: Alert Form with "General" Tab

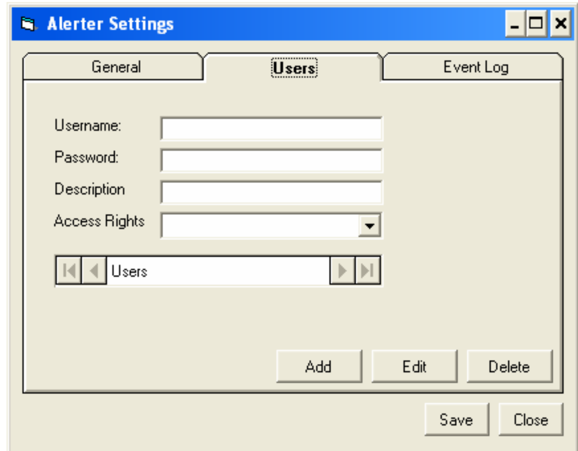


Figure 22: Alert Form with "Users" Tab

In this tab, it is possible to set how often sensor updates are sent to the software. Any changes in status, and all temperature readings, are fed into their respective places in the database. The "save" button must be pressed for any changes to be saved.

The "Users" tab is shown in Figure 22. In this form, it is possible to toggle between existing users using the horizontal scroller. By clicking "Add", a new user can be created with the user name, password, description, and access rights as specified on the form. The "Edit" button updates the current user's info with the parameters described above. As expected, the "Delete" button removes the current user from the database.

All these changes will have no effect until the "Save" button is pressed. The "Close" button closes the entire form. The last tab is the "Event Log" tab, as shown below.

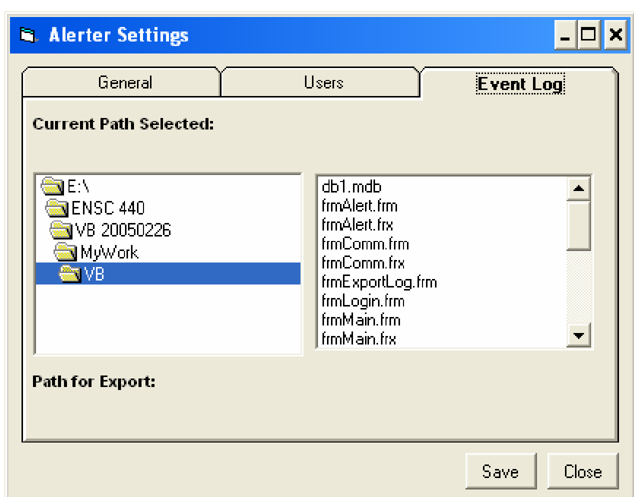


Figure 23: Alert Form with "Event Log" Tab Open

The destination path for the exportation of the event log is set here, and of course the "Save" button must be pressed for any changes to be saved.

### 5.6 Serial Port Communications

Serial port communication is established through the use of the MSCOMM control available in the Visual Basic IDE. Table 20 shows the properties associated with the MSCOMM control [10].

**Table 21: MSCOMM Control Communications**

| Properties | Description |
|---|---|
| **CommPort** | Sets and returns the communications port number. |
| **Settings** | Sets and returns the baud rate, parity, data bits, and stop bits as a string. |
| **PortOpen** | Sets and returns the state of a communications port. Also opens and closes a port. |
| **Input** | Returns and removes characters from the receive buffer. |
| **Output** | Writes a string of characters to the transmit buffer. |

The OnComm event associated with the MSCOMM object will be used to receive data when data arrives through the transceiver connected to the PC.

### 5.7 Winsock Communications

Under Configuration → Communication Settings, a Communication Form pops up with two tabs, as shown below. The "Notification" tab will be discussed first.

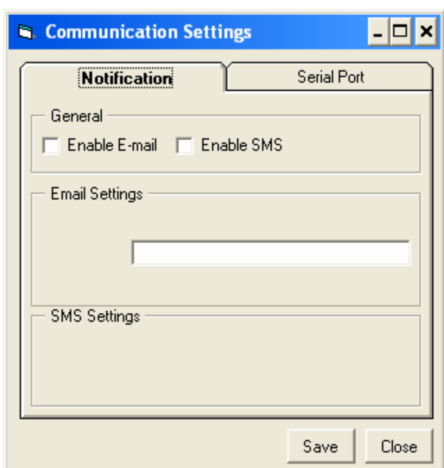


**Figure 24: Communication Form with "Notification" Tab Open**

It is possible to enable or disable E-mail and/or SMS notifications in this section. It will be possible to change settings for either of these notification methods, and its full design will be implemented by mid-April 2005. When an unexpected security or safety event occurs and the user is not home, the alternative to notify the user is to send e-mails. E-mails will be sent using an SMTP (Simple Mail Transfer Protocol) server. Sending and receiving email messages is used by port 25 of the PC.

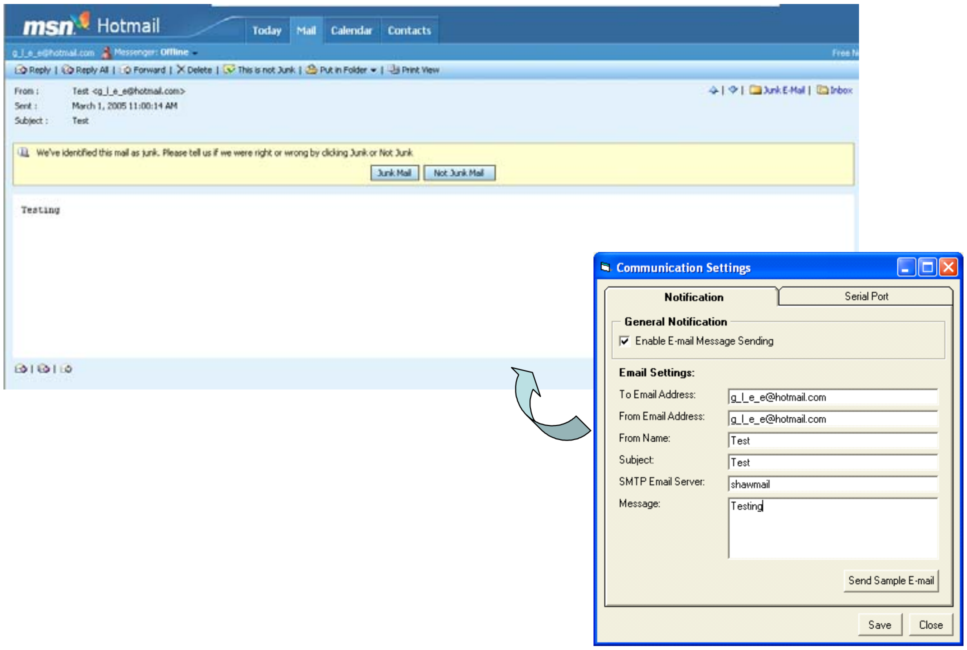


**Figure 25: A screenshot of the email received after a message is sent out**

# 6 Conclusion

The design specifications described in this document pertaining to the WInnovations Wireless Home Security System have been outlined in detail and provide an accurate as possible representation of the product. The specifications will cover as much of the technical requirements regarding this product.

# 7 References

[1] http://www.lightsearch.com/resources/lightguides/sensors.html
[2] http://www.cma.science.uva.nl/english/download/pdf/manuals/d03517.pdf
[3] http://www.lighting.com/content.cfm?id=516&sid=21&page=/
[4] http://content.honeywell.com/sensing/prodinfo/ultrasonic/technical/98ib80.pdf
[5] http://www.cs.sunysb.edu/~tony/392/sensors/sensors.html
[6] http://home.howstuffworks.com/question238.htm
[7] http://www.glolab.com/pirparts/infrared.html
[8] http://mstmicro.com/sensor/sensor.html
[9] http://cache.national.com/ds/LM/LM124.pdf
[10] http://www.vbip.com/winsock/winsock_control.asp
[11] http://msdn.microsoft.com