

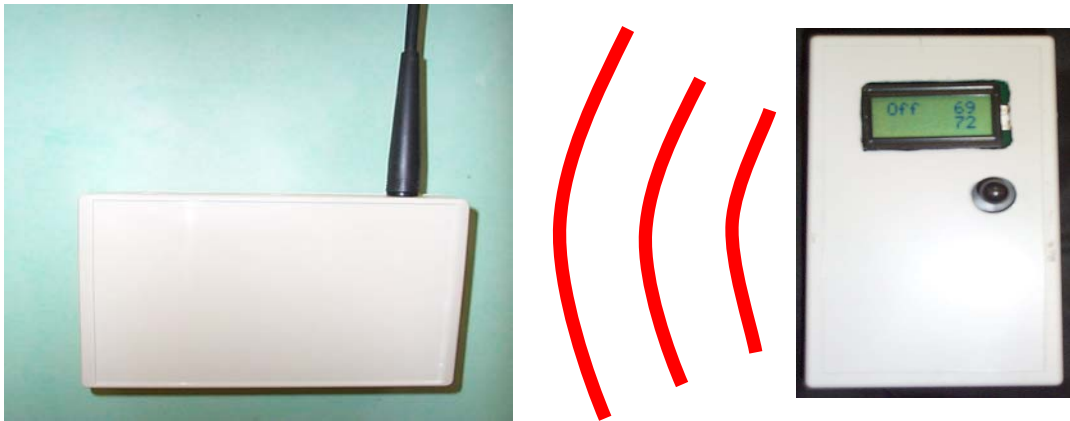
design document for

# Portable Thermostat

submitted to:

Professor Joseph Picone  
ECE4522: Senior Design I  
Department of Electrical and Computer Engineering  
Mississippi State University  
Mississippi State, MS 39762

December 4, 2003



submitted by:

J. Baugh, B. Chapman, A. Mulkana, G. Williams  
Faculty Advisor: Professor Marion Hagler  
Department of Electrical and Computer Engineering  
Mississippi State University  
Box 9571  
Mississippi State University, MS 39762  
Tel: 662-325-3912, Fax: 662-325-2298  
email: {jab24, bac7, amm18, gmw4}@msstate.edu

TABLE OF CONTENTS

1. PROBLEM..... 1

2. OBJECTIVES..... 2

    2.1. Temperature Sensing..... 3

    2.2. Connectivity..... 4

    2.3. Anti-Short-Cycle Control..... 4

    2.4. Remote Unit’s Physical Characteristics..... 4

    2.5. Cost..... 4

3. APPROACH ..... 5

    3.1. External Base Unit..... 5

    3.2. PC Software ..... 7

    3.3. Remote Unit ..... 11

        3.3.1. Remote Hardware Approach..... 12

        3.3.2. Remote Software Approach..... 13

    3.4. Temperature Sensor..... 17

        3.4.1. Theory of Operation..... 17

        3.4.2. Temperature Sensing Components..... 17

        3.4.3. Sensing Resistance Changes..... 19

        3.4.4. Signal Amplification. .... 19

    3.4. HVAC Interface..... 20

    3.5 Power ..... 21

4. TEST SPECIFICATIONS ..... 22

    4.1. Temperature Sensing..... 22

    4.2. Packaging..... 23

    4.3. RS-232E ..... 23

    4.4. LCD Output ..... 24

    4.5. Anticipator ..... 26

    4.6. Over-Current ..... 26

    4.7. HVAC Connectivity ..... 27

5. TEST CERTIFICATION ..... 28

    5.1. Temperature Sensing..... 28

5.2.	Size and Weight .....	29
5.3.	RS-232E .....	29
5.4.	LCD Output .....	29
5.5.	Anticipator .....	29
5.6.	Over-Current .....	29
5.7.	HVAC Connectivity .....	30
6.	SUMMARY AND FUTURE WORK.....	30
7.	ACKNOWLEDGMENTS .....	31
8.	REFERENCES.....	31

## EXECUTIVE SUMMARY

People have always looked for ways to add convenience and comfort to their lives. Home heating, ventilation, and air conditioning (HVAC) is one way that this has been achieved. It adds comfort by heating or cooling the air to the desired temperature and is convenient because it requires minimal oversight from the user. The main disadvantage of traditional heating and cooling systems is that the thermostat is physically located in only one place. The ambient temperature measured by the thermostat may not be the ambient temperature in other areas of the home. The temperature in the rooms of a house can vary depending on such factors as airflow, number of windows, and insulation quality. A portable thermostat would be able to better adjust the HVAC accordingly based on the temperature it measures at its location.

The portable thermostat system can be divided into two main parts, the remote and base. The HVAC system is electronically controlled by the base, which consists of a personal computer (PC) and a base hardware unit. The base hardware unit controls the flow of data to and from the PC. Additionally, it controls the base temperature sensor, HVAC system, and sending the received data from the remote to the PC. The remote will consist of a PIC, RF transmitter, temperature sensor and user interface. The user interface consists of an LCD and two push buttons that allow the user to set the desired temperature. The LCD will display the ambient temperature, desired temperature and the mode of operation the HVAC system is in (AC, heat, fan, off). The PIC will poll the temperature sensor for the ambient temperature and compare it to the desired temperature. Depending on the mode of operation the HVAC is set to, the remote will transmit to the base telling it to turn the HVAC on, off or no change.

Several design constraints have been defined in order to help finish this product successfully. The temperature-sensing device will measure temperatures from 50°F to 90°F. The user will be able to adjust the temperature between 55°F and 85°F. Heating and cooling systems have a tendency to “overshoot” the desired temperature. To correct this, an anticipator will be used to turn off the HVAC system before the desired temperature is reached. Generally, most PC's have at least one RS232 interface. Therefore, the PC will connect to the base hardware through EIA standard RS232E. The portable thermostat will also interface with standard 5-wire thermostats. Our unit will provide AC over-current protection. A delay time of 5 minutes will be defined in order to keep the HVAC system from drawing high currents, which could damage the system.

For the temperature-sensing device, we wanted to have a very accurate thermometer for our system. Therefore we chose to use the thermistor for our temperature circuit because of its sensitivity. MATLAB and PSpice simulations were used to find the best circuit for our portable thermostat. Both the anticipator and the AC over-current protection are implemented through software. We feel that our product will be very marketable and make HVAC systems perform to a higher standard.

## 1. PROBLEM

Home heating and air conditioning allows homeowners to control the ambient temperature inside the home. In order to increase or decrease the temperature in a home, hot or cold air is circulated throughout the home via heating and air conditioning. A thermostat is a device that automates the process of turning the heating and air conditioning on or off. However, traditional thermostats fall short for one specific reason. The problem is that the thermostat is physically located at one place in the home. This does not address the problem of temperature gradients throughout the home and increases heating and cooling costs.

In the most basic form, the thermostat's user interface allows the user to select a desired ambient temperature. Through a feedback loop, the ambient air temperature is measured and then compared to the desired temperature. When the ambient temperature deviates from the desired temperature by a pre-defined threshold, the thermostat turns on the heating or air conditioning unit until the desired temperature is achieved. Once the desired temperature is obtained, the thermostat turns off the units and repeats the process.

During the design of the home, architects and mechanical engineers attempt to locate the heating and cooling vents and thermostat to theoretically produce the same temperature throughout the home. However, due to cost constraints, bad designs, contractor modifications, and environmental factors, there is usually a large temperature gradient throughout the finished home. In these real situations, the traditional thermostat, immobile and physically attached to a wall, is inadequate.

InComfort's (Intelligent Comfort, Inc.) product will provide a solution to these problems. Our product will be a portable thermostat with its own digital temperature gauge that can be operated from any location in a household. The portable unit will read the temperature at its current location and relay this through a wireless link to the base thermostat. This way, the adjustments made to heating/air conditioning unit will be based on the temperature measured at the current location of the portable thermostat. There are several examples of when our product would be very beneficial to the public.

One of these instances involves persons living in apartment complexes or townhouses. Most townhouses have a standard layout with a living area/kitchen downstairs and two or three bedrooms upstairs. The thermostat in these buildings is usually located downstairs. The main problem with this setup is that hot air rises due to Charles' Law. This often causes temperatures to be hotter in the upper level during the winter and summer. Given that this is half the rented property and the location of the sleeping quarters, tenants will spend a significant amount of time on this level. The portable thermostat would remedy this problem by allowing the user to take the portable unit with him or her and set it on a nightstand while they slept. With a portable thermostat the user would sleep comfortably and not have to continually walk downstairs to adjust the temperature.

Other homes have problems at different times of the day due to environmental factors. The sun may heat an east facing room in the morning and a west facing room in the afternoon. Because the thermostat is located in one, it is difficult to address the problem of differing room

temperatures. It is often inconvenient for a user to keep changing the thermostat when they go to a different level or the sun has changed position. This is especially annoying for people that spend a large amount of time in one room, such as people working at home. A room that functions as an office may change temperature several times a day. This could be due to natural factors such as the sun coming in through a window, outdoor temperature changes, or man-made factors such as heat from lighting or electronic components in computers and stereos.

Families spending time together at night also have a problem as they move from room to room. The family may eat dinner in the kitchen where the temperature is comfortable, but then move to a living room which is always a little too cool and then finally to the bedrooms which are always too warm. Each time the family changes rooms, the thermostat would have to be adjusted. Although not an excessive annoyance to stop a task, adjusting the thermostat and then return to the task that was at hand, it is still time consuming and bothersome.

Also, many people sleep with their doors to their rooms closed for different reasons. However, this causes the circulation of air to be poor in a household. This would cause a temperature difference between the temperature in your room where you are sleeping and the temperature at the base thermostat. Once again, our product would allow the user to set our portable unit on a nightstand next to their bed and sleep easily.

Besides that, if the user were having a small group of people over to their house the group would spend the majority of their time in one room (den, living room, etc.). When a lot of people are placed in one room the temperature will rise. The portable thermostat would allow the user to sit the unit in that room and keep the whole room at a comfortable temperature. The unit could also be used in the kitchen. Ovens, stoves, and other kitchen equipment can heat up the kitchen and even other rooms considerably. This would be a perfect situation to use the portable thermostat.

The portable thermostat solves many of the problems of traditional thermostats. As our thermostat is moved from room to room, it quickly determines the new temperature and turns the heating and air conditioning on or off. Our thermostat will automatically adjust the temperature at the user's location as he or she moves throughout the household. With a traditional thermostat, the user would have to continually change the thermostat as he or she moved about the house. Our product will give users the freedom to keep a consistent temperature wherever they are within their homes.

## 2. OBJECTIVES

The purpose of this project is to create a portable and PC-based thermostat. The remote can be taken to a room and the HVAC will operate until the desired temperature is obtained at the remote. The HVAC system can also be operated from a PC that acts as a fully functional thermostat and a base station for the remote. The design constraints for the Portable Thermostat project include the following five technical constraints and five real world constraints.

1. **Temperature Sensing:** The temperature sensor will measure temperatures from 50°F to 90°F. The user will be able to adjust the temperature between 55°F and 85°F.

2. **Computer Connectivity:** The EIA RS232E standard will be used to connect the base hardware unit with the PC.
3. **HVAC Connectivity:** The base hardware unit will connect to standard 5-wire HVAC systems. Some 5-wire systems contain a sixth wire, which is a common. HVAC units that have the extra common wire will operate with this thermostat. This thermostat is not compatible with heat pump systems.
4. **Anticipator:** The anticipator will allow the user to adjust cutoff temperature by minus 3°F when mode is set to heat and plus 3°F when mode is set to AC.
5. **AC over Current Protection:** The base will contain a trip time to keep the HVAC system from cycling on and off too quickly. This can cause undue stress to the HVAC unit and excessively high currents to the compressor.
6. **LCD output:** The remote unit will display the ambient temperature, desired temperature, and mode of operation (Heat, AC, Off) on a character LCD that contains 2 lines and 8 characters.
7. **Cost:** The target price is \$200 manufactured. This does not include the RF module used in the prototype. The RF cost will be determined by estimated costs of similarly sized transmitters as if one were specially made for this application.
8. **Operating Systems:** The PC software will be compatible with Windows 95, Windows 98, Windows ME and Windows XP operating systems.
9. **Remote Weight:** The remote's weight will be no more than 10 ounces.
10. **Base Unit Mounting:** The base unit will be wall-mountable with the option of connecting the RS232 and power cable to the back or bottom of the unit. The HVAC wires must be connected through the back.

## 2.1. Temperature Sensing

The temperature sensor will measure temperatures from 50°F to 90°F. The user will have the ability to adjust the temperature between 55°F and 85°F. According to guidelines published by the California Department of Energy, the lowest recommended temperature is 55°F [1] during winter nights or when no one is home for long periods of time. The highest recommended temperature is 85°F during the summer. Many factors such as humidity and airflow affect the desired temperature of a room. The human comfort zone is between 69.8°F and 75.2°F [2]. By choosing 55°F to 85°F the full range of temperatures can be taken into account.

Heating and cooling systems have a tendency to “overshoot” the desired temperature. To correct this, an anticipator is used to turn off the HVAC system before the desired temperature is reached. The anticipator value is normally set during installation. For example, suppose the user sets the thermostat to “Heat” and the temperature to “70°F”. If the anticipator is set to -1°F for the heating system, the system will operate until the measured temperature is 69°F.

## 2.2. Connectivity

The PC will connect to the base hardware through EIA standard RS232E. This was chosen because most PC's have at least one RS232 interface [3]. The base hardware unit will connect to standard 5-wire HVAC systems. Some 5-wire systems contain a sixth wire, which is a common. HVAC units that have the extra common wire will operate with this thermostat. No formal standard has been found that specifies this [4]. This thermostat is not compatible with heat pump systems.

## 2.3. Anti-Short-Cycle Control (AC Over-current Protection)

When the air conditioning compressor turns off, the refrigerant is temporarily at a high pressure that slowly decreases over a period of a few minutes. If the unit is turned back on during this interval, the compressor has to perform a cold start against high backpressure, causing the compressor to draw very high current [5]. We will provide a delay where the refrigerant has time to depressurize before turning the unit back on. A delay of 1 minute will be used as commonly used in industry [4].

## 2.4. Physical Characteristics

The remote unit will weigh no more than 10oz to allow for easy mobility while carrying the remote. This designated weight was chosen because it is comparable to other handheld devices such as cell phones and graphing calculators.

The base unit will be wall-mountable. This will allow the HVAC wires to remain inside the wall eliminating the need for a wall jack and wire harness. The HVAC wires will connect to the unit through the wall side only. The user will have two options when connecting the RS232 and power supply to the unit. Connectors will be provided on the back and bottom of the unit. The connectors on the back of the unit will allow the user to install RS232 and power in the wall creating a "clean" look around the unit. If the user does not wish to install either or both wires in the wall, then the user can connect to the unit on the bottom.

## 2.5. Base Unit

The target price is \$200 after manufacturing. This is very competitive with similar units that list for \$219 and \$246. The price does not include the RF module. The RF module used in the prototype is not necessarily what would be used for production. If the thermostat were to go into production a custom RF module would be designed for this application. The RF cost will be determined by estimated costs of similarly sized transmitters as if one were specially made for this application. [6]

### 3. APPROACH

The portable thermostat can be divided into three main parts, remote, base hardware, and the PC. The PC coordinates the HVAC with the other components of the system. Many choices had to be made during the design to insure that all parts would interact correctly and efficiently.

#### 3.1. External Base Unit

A logic unit is required within the external portion of the base. Without it, there is no device to control data flow between the computer, remote, and local temperature sensor. In Senior Design I, we had two options: FPGA's or microcontrollers. We chose microcontrollers because they usually have built-in analog-to-digital converters (ADC) and UART's. However, after examining our design, we encountered a problem. For the microcontroller to serve our purposes, we found that it had to have an ADC and support for two UART's. Additionally to ease its implementation, the microcontroller should come in a DIP package to make it possible to prototype on a solderless breadboard. After searching by these criteria, the remaining microcontrollers were the Cypress CY8C26443, and the Microchip PIC16F877. Their descriptions were as follows:

	CY8C26443	PIC16F877
ADC	12 channels, 8-12 bits	8 channels, 10-bits
Available serial busses	UART's (2), SPI	UART (1), SPI, SCI, I2C
I/O Pins	24	33
Programmer	Must be bought or borrowed	Provided by department
Programming software	Must be bought or borrowed	Free downloads / packages provided by department

Table 1. Comparison between the CY8C26433 and the PIC16F877.

The PIC already included the ADC, but it had only one UART. Therefore, we had to find a way to connect an additional UART with it. After more searching, we found the MAX3100, a UART manufactured by Maxim. This 14-pin UART is especially designed to interface with microcontrollers and communicates with them through the Serial Peripherals Interface (SPI). The nearest competitors we found used a 40-pin package and would have required several pins to operate. Instead, by using the MAX3100, we were able to reduce the number of required pins to 3 for I/O and 3 for control.

Ultimately, after checking our facilities, we chose the PIC16F877. This decision was made mostly because the university has already provided us with PIC programmers, programming resources, and software.

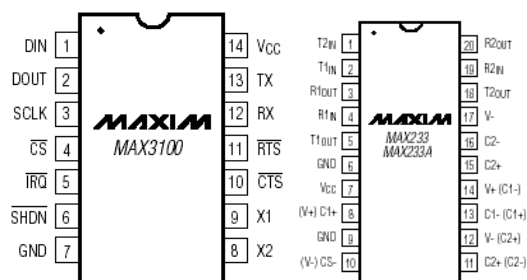


Figure 2. Pin diagrams of the MAX3100 (left) and MAX233 (right).

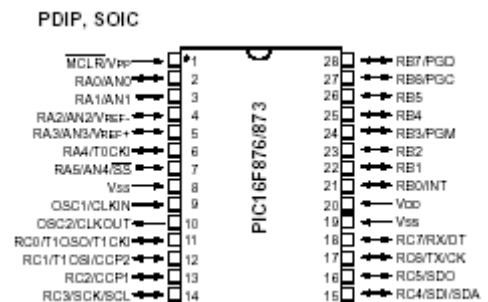


Figure1. Pin diagram of the PIC16F873.

Furthermore, we were more familiar with the PIC processor than the Cypress processor, and we were

already using a PIC16F877 on the remote, making the sharing of similar source code possible. However, the 16F877 had more output pins than we needed, and while this may be great for debugging during prototyping, it is neither cost-effective nor suitable for a packaged product. For this reason, in Senior Design II we decided to use the PIC16F873. The comparison is as follows:

	PIC16F877	PIC16F873
ADC	8 channels, 10-bits	8 channels, 10-bits
Available serial busses	UART (1), SPI, SCI, I2C	UART (1), SPI, SCI, I2C
I/O Pins	33	22
Programmer	Provided by department	Provided by department
Programming software	Free downloads / packages provided by department	Free downloads / packages provided by department
Cost	\$5.11 per unit (at 100 units)	\$4.33 per unit (at 100 units)

Table 2. Comparison between the PIC16F877 and the PIC16F873.

Additionally, because the output from the PIC to the PC is sent over a RS-232 link, the voltages must be converted from CMOS to RS-232 levels and vice versa. A common method of carrying this out is to use the MAX233. This chip utilizes internal capacitors and “charge pumps” to create the +12V and -12V needed for RS-232 [7].

When operating, the base attempts to establish a connection with the PC. Following this, the base waits for a prescribed period for the remote to send a signal. If the signal is received, the remote’s incoming command is applied to the HVAC system and relayed to the PC, regardless of whether PC is present or running its Graphical User Interface. However, when no signal is received for the specified interval, the base switches to local mode and begins to measure local temperature. At this point, the base relays temperatures to the PC and allows it to control the HVAC based upon the local settings. If the PC and remote both are absent for a specified amount of time, however, the base shuts down the HVAC and waits for a connection from either the remote or PC before it resumes operation. An outline of a typical operation is as follows:

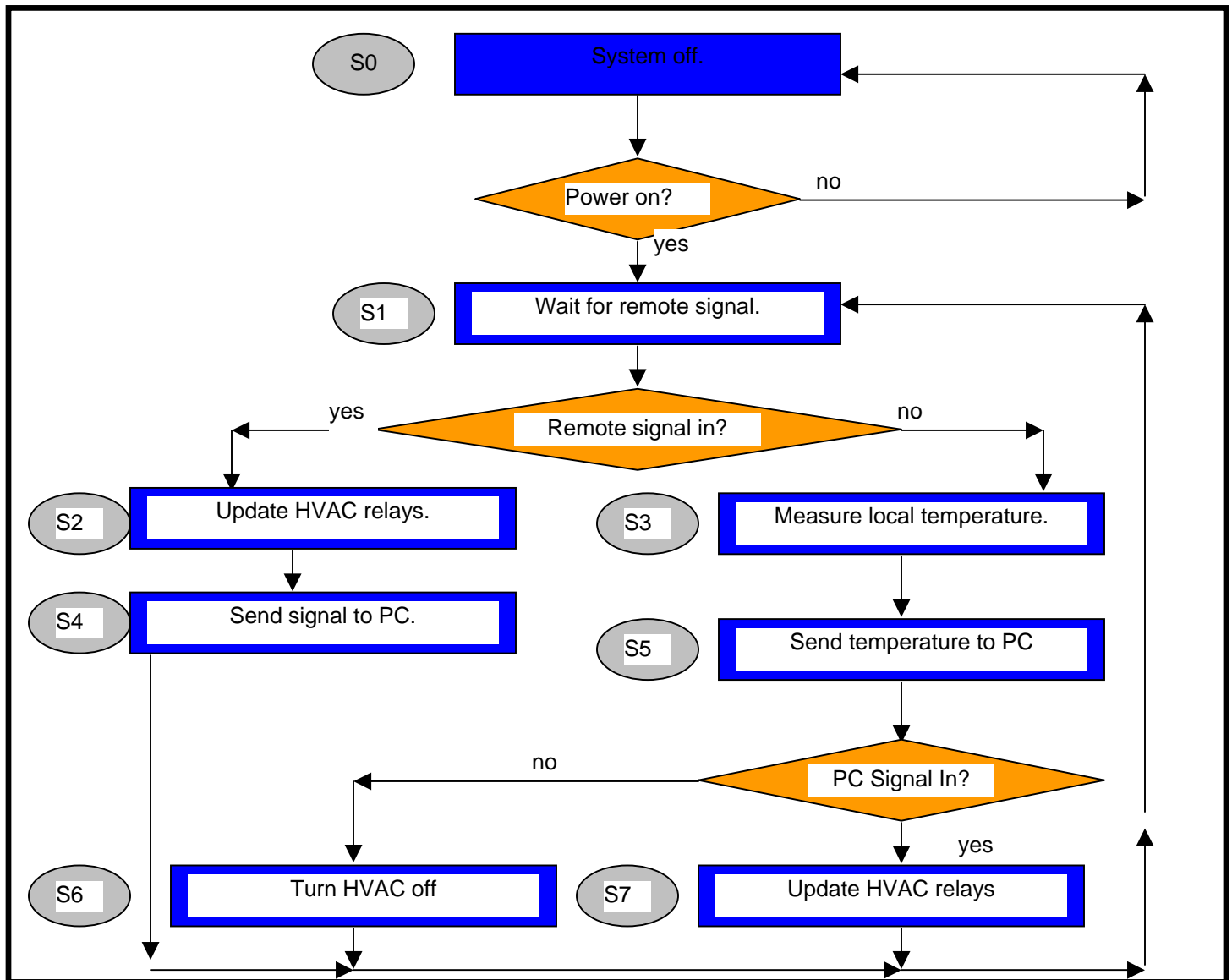


Figure 3. ASM Chart of the PIC16F873's execution.

### 3.2. PC Software

The base unit for the system is made up of several components. It contains a base PIC (Microchip's 16F873) to handle the incoming information from the local temperature sensor and the remote unit, a GUI on the PC to allow the user to set current thermostat settings when the system is in the base mode, and an HVAC interface to select the correct mode (i.e. A/C, Heat, etc.). The communication between the PC and the base PIC is made over an RS-232 connection.

Visual Basic 5.0 was chosen for the GUI interface so the user can easily view and change the HVAC settings. Figure 4 below shows the form of the base unit in Visual Basic.

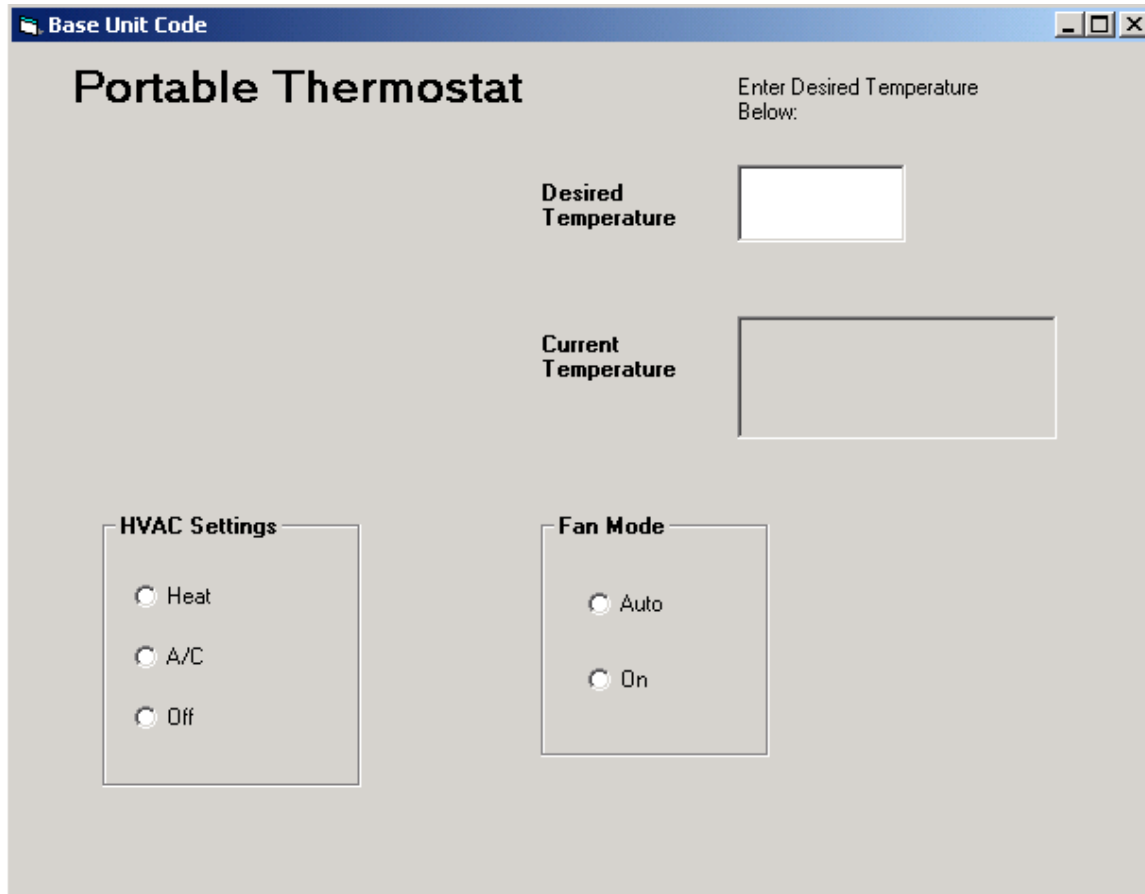


Figure 4. PC User Interface

The code is designed to function differently for two different scenarios. When the remote unit has control of the system several items on the form will disappear. These items include the desired temperature label, the text box to the right of the previous label, and the directions above both of these instructing the user to enter a desired temperature. Since the remote unit has control and the user will only be able to enter the temperature on the remote unit, these items disappear because they could potentially confuse the user. As soon as the mode switches to local mode these items reappear so the user can enter the desired temperature [11].

Another reason for choosing Visual Basic was because of its built in communication functions with the serial port. The Visual Basic Professional edition has OnComm, which allows programming with the serial port. The OnComm event is similar to a command box or text box that you place on a Visual Basic form. However, the OnComm event does not show up when the program is in operation. The OnComm event is shown below in Figure 5 along with its properties that can be set before the program is run or during a program's operation. These properties were setup to allow communication over RS-232 to the base PIC [3,10].

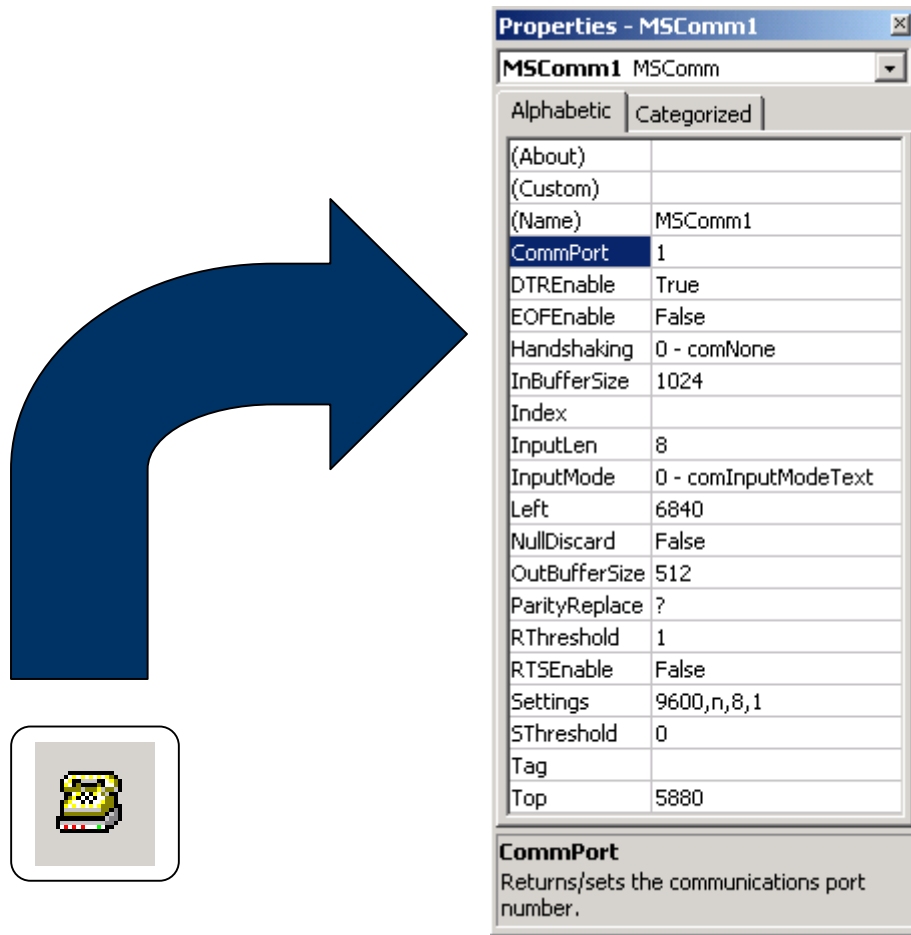


Figure 5. Example of the MSComm1 event and the MSComm1 settings in Visual Basic.

One of these properties was specifically used to solve the problem with overcurrent described in the design constraints. The Rthreshold property has two different settings [3,10]. It can be set to one allowing Visual Basic to receive information through the serial port or it can be set to zero which disables the receiving ability of Visual Basic at the serial port. This property combined with the timer function included in Visual Basic allowed a nice way to implement overcurrent protection. The timer function is similar to the OnComm icon shown above (except it looks like a clock). This icon is placed onto the form and then the time interval is set to a specified time within its properties. When the timer function is called in the main program, it waits for the specified time interval before it runs the code inside of the function [8]. The procedure is shown below:

```
MSComm1.Output = buffer1
MSComm1.RThreshold = 0
Call Timer1 Timer

Private Sub Timer1_Timer()
    MSComm1.RThreshold = 1
End Sub
```

Figure 6. Example of the MSComm1 coding.

Immediately after buffer 1 is output to the HVAC system in the main program (which tells it to turn the AC on) the Rthreshold is set to zero. Next the timer procedure is called and this waits one minute until it sets Rthreshold back to a value of one. This means that the serial port will not accept information from the base PIC for one minute and therefore keeps the system from repeatedly turning on and off and drawing high currents.

There were several changes made in Senior Design II. Upon testing our software for different operation systems (Windows 95, Windows 98, XP, ME, and 2000) a VB executable was created and tested on these systems. However, the need for a serial port selection screen was needed. The user may have his/her serial port set to a different port for the computer system. Therefore a new screen was needed to allow the user to make the correct choice. The selection screen is shown below in figure 7.

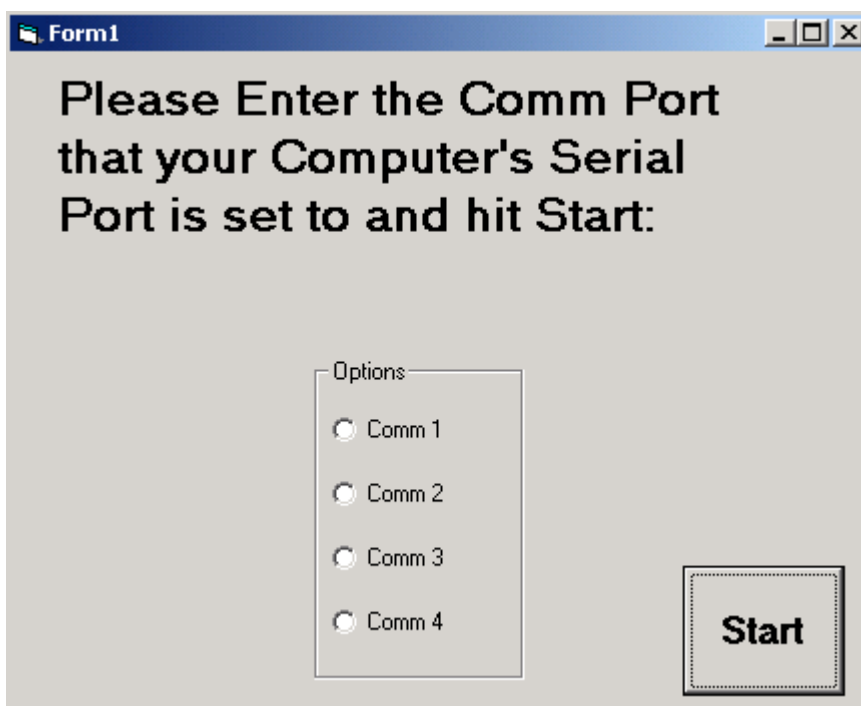


Figure 7. Base Com Port Selection Screen

This screen appears at the very beginning of program operation. It must be completed before the base thermostat program will run.

Other changes to the PC code were simple modifications that make the program easier to operate for the user. For example, a default temperature of 75 was selected to automatically be the desired temperatures starting point. Also, two command buttons (up and down) were placed to the right of the desired temperature so the user can make a single click to change the temperature up one degree or down one degree. This way the user does not have to use the keyboard to enter in an exact temperature. The new screen is shown below in figure 8.

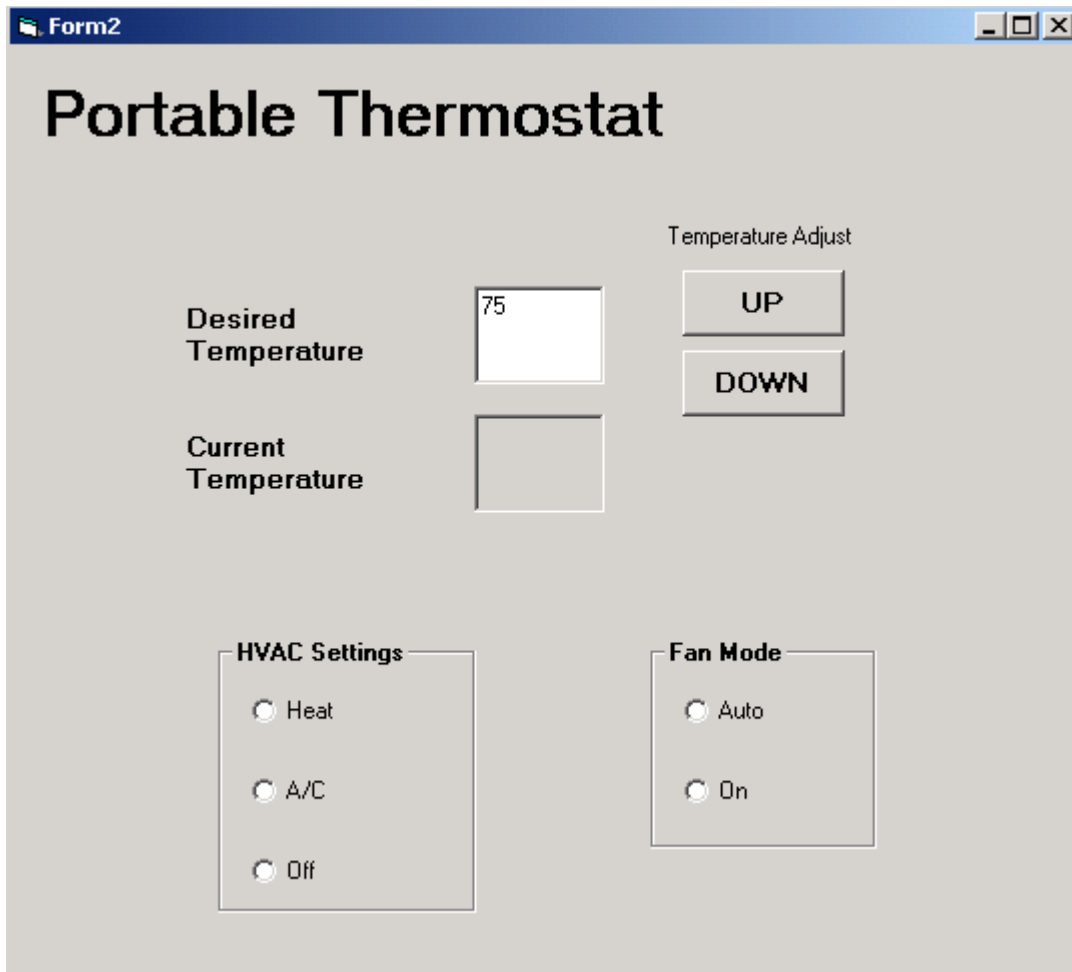


Figure 8. PC Base Thermostat (Updated Version)

### 3.3. Remote Unit

Software and hardware design in the remote unit will be accomplished using the Microchip 16F873. This chip was chosen for reasons of speed, analog to digital conversion ability, memory storage, and number of I/O ports. The Microchip 16F873 chip operates at 20Mhz, features a 10-bit multi-channel ADC converter, has 128 x 4 bytes of EEPROM data memory, and has 22 I/O pins.

### 3.3.1. Remote Hardware Approach

Initially, the Microchip 16F877 was chosen as the processor for the remote. The large number of I/O pins was thought to be necessary for all the modules that must connect to the microchip. Figure 2 shows the pin out of the 16F877 as it was in the remote unit. Eleven pins were allocated for the LCD display, seven pins for remote buttons, four pins for the anticipator settings, two pins for voltage reference, two pins for the RF unit, and one pin for the temperature sensor. This layout also left a small number of pins left over to allow for an upgrade or a new feature to be added in the future.

Several other chips were also looked at before this chip was chosen. One other chip given careful consideration was the 87C652 by Philips Semiconductor. This chip shared features with the Microchip 16F877 such as speed, memory, a necessary number of I/O pins, and a sleep-mode allowing for the conservation of power. The 87C652 does not support analog to digital conversion; however, and has a larger assignment instruction set to learn, 100 instructions as opposed to only 35.

The 16F877 PIC worked without problem and successfully performed all functions required for the remote; however, the 16F877 had unused pins which caused the remote to be larger. The smaller and cheaper 16F873 PIC was found to provide all the same functionality as the larger chip and required minimal changes to the source code. For this reason, during the packaging stages of development, the 16F873 was chosen as the final processor to drive the remote. Much of the layout remained similar as it was with the larger PIC. The temperature sensor circuit was connected to pin0 on portA. This pin was setup for analog to digital conversion. The output voltage read across the thermistor in the temperature sensor circuit is used to determine the room's current temperature. Also connected to the microchip is the Hitachi 44870 LCD. Of the LCD's 16 pins, 11 are connected to the microchip. Those not connected directly to the microchip

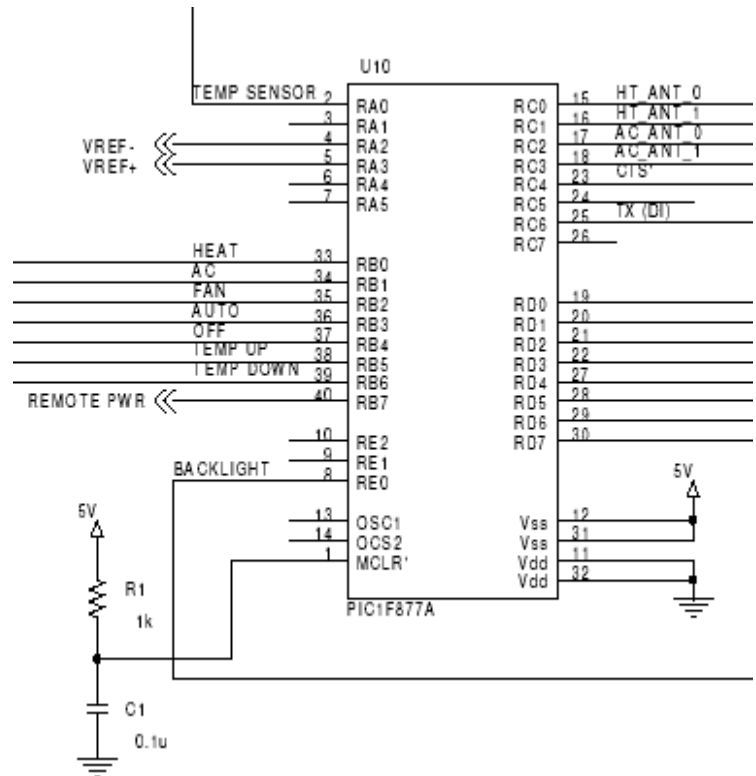


Figure 9. Pin assignments for the 16F877 PIC.

are the pins for Vdd, Vss, backlight, and contrast. The contrast pin is tied in parallel with a 4K-ohm resistor. This resistor value was chosen after experimenting with several other values achieved by using a Decade Resistor Box. A 330-ohm resistor caused the LCD display to become extremely dark and a 4.7K-ohm resistor created a very faint contrast. Pin 1 on the microchip has the MCLR function assigned to it. It resets the chip upon each power up. It is tied to five volts with a 10K ohm resistor in between. The chip's oscillator is running at 3.57 MHZ. At this speed, the oscillator type 'XT' was chosen since 'XT' is suggested for 4 MHZ [12]. Also suggested was to use two 20pF capacitors in parallel with the 3.57 MHZ crystal; however, two 33pF capacitors was chosen to provide a more stable oscillation. Both of these capacitors also connect to Vss (ground). The only additional hardware component necessary to achieve the software requirements is the RF module. From a hardware aspect, connecting the RF unit was very simple. PortC is the only port that can be configured for Universal Asynchronous Receiver/Transmitter (UART) operations. Pin6 portC is the pin connected to the RF module's transmitter line [9].

### 3.3.2. Remote Software Approach

Input into the Microchip 16F873 will come from either the user through use of a four-position joystick or from the temperature sensor unit. These input items will be processed by the microchip, displayed to the user, and sent to the transmitter. The possible interactions of the remote's microchip with other modules are shown in Figure 8 below.

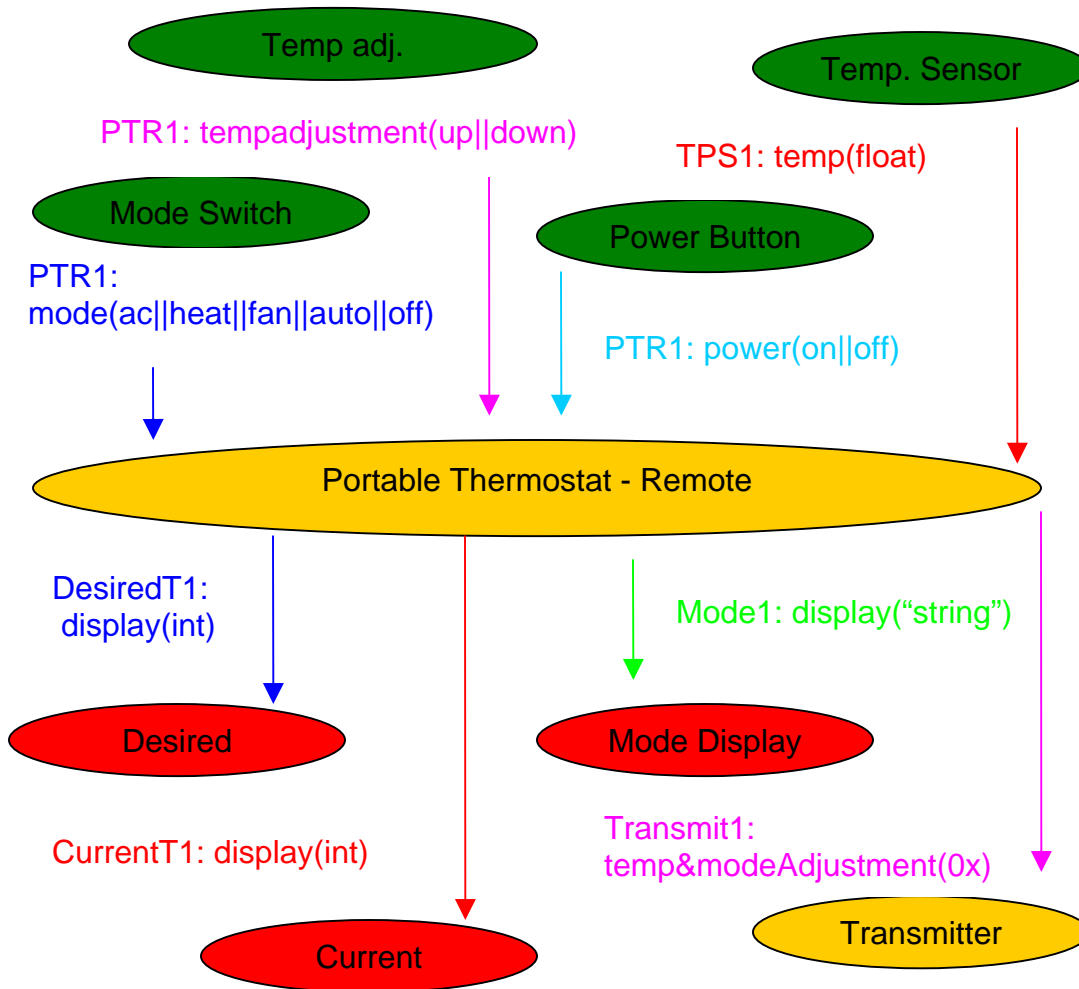


Figure 10. Object communication model for the remote module.

The program on the 16F873 chip was written in C++ Language using PCW’s PIC C Compiler. The main needs of the program are to be able to read an analog value from the temperature sensor, display HVAC operations onto the LCD display, process user inputs, and send commands to the base unit via the RF unit. Pin0 of portA on the microchip was setup for analog to digital conversion. The microchip will read the output voltage across the thermistor on pin0. Through the use of a table coded onto the 16F873, the current room temperature will be realized by comparison of the output voltage to other attainable voltages. These readings will be refreshed to the LCD display approximately every 30 seconds.

Input from the user will be accepted by method of the joystick. The up and down positions of the joystick will increase and decrease the desired temperatures respectfully. The left position will work as the on/off power switch to the remote and the right position will operate as the mode selector. The temperature up and down positions of the joystick will be asserted high when being pushed. The 16F873 microchip’s program will continuously run through a loop to check for assertion of either of these two positions. When one of these positions are pushed, 5 volts is sent to the corresponding pin that that position of the joystick is connected to. This will change the value of the pin to high, causing the program to realize that the user has made a request via the

joystick. For instance, if the user pushes the joystick up, then “if (Temp\_Up == 1)” becomes true and the loop is entered. The loop will then process the information, change the desired temperature, and send updates to the LCD display so that the user may verify their change.

The mode selection behaves similar to the buttons; however, when one mode is asserted, a high value is assigned to that mode until the mode position of the joystick is asserted once again. Unlike the temperature up and down adjustments which operate independently of one another, the future mode selected depends on the present mode. Since only the right position of the joystick is used for mode selection, a circular array of modes was coded to allow the user to cycle through all the modes. When the user first turns the remote on, the mode is defaulted to ‘off’. The user may then push the joystick right in a desire to select AC as the mode. When the right position of the joystick is asserted, 5 volts is sent to an input into the PIC. Upon receiving a high value to that input, the program will check its current location in the array, increment to the next position, and chose the mode in that position as the new HVAC mode. Once the new HVAC mode of operation has been selected, the LCD display is refreshed to correspond with any changes, and the information is also sent to the RF unit.

The RF unit that will conduct the communications between the remote and the base unit is the 900MHz MaxStream. Although this model supports two-way communication, only one-way transmission is needed since the remote will not need to receive any information from the base. The remote will only need to have one pin connecting it to the RF unit. This pin will be pin6 on portC. Serial transmission will be needed for the microchip to communicate with the RF unit [13]. Since the 16F877 chip has a built in UART (Universal Asynchronous Receive/Transmit), an external UART chip will not be needed. PortC has to be used to connect to the RF unit for the reason that it is the only port that can be setup for asynchronous serial data communications.

The only information being sent to the base via the RF unit will be the HVAC modes of operation. They will be sent as alphanumeric values. Table 3 below shows the corresponding alphanumeric value for each possible HVAC setting.

Fan Only	A
AC and Fan	C
Heat and Fan	E
Off	Z

Table 3. HVAC mode alphanumeric transmissions

These transmissions will be sent approximately every 20 seconds from within the portion of the program that deals with the HVAC mode switches. Mode ‘Off’ represents when the remote is still on but the thermostat is to be set to off. To ensure that the value sent by the remote is received in the correct format of what the PC is expecting, the start bits, stop bits, and data rate must be setup just it is on the PC. As shown in Figure 9 below, the start bit begins transmission, and then is followed by the data bits, and lastly, the stop bits. The parity bit is an option that was unneeded and, therefore, left out.

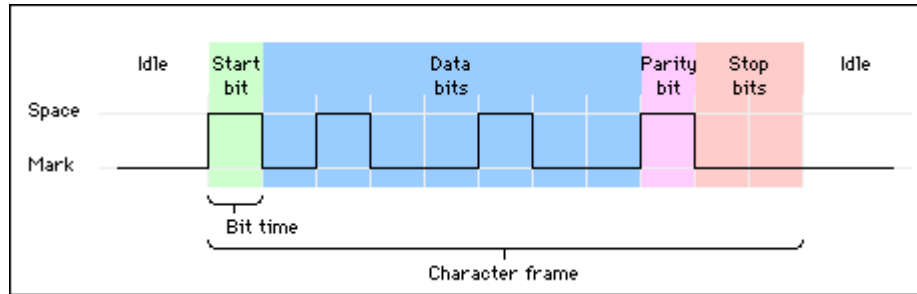


Figure 11. Data transmission example for 6Dh[9,10]

When transmission is idle, the serial output line will stay at logic-0 level. When data is ready to be sent, a start bit is sent first. This bit is used to synchronize the transmitter and the receiver. At this point, the PC will know that data is ready to be sent when it receives the start bit at logic-1 level. The data bits are sent next and are sent least significant to most significant bit and is framed between the start bit and the stop bit.

The user’s thermostat adjustments are also output to the LCD display. A Hitachi 44870 display is used with the remote. It has an 8 x 1 screen size. The HVAC mode will consume the two places to the far left of the remote, followed by a space, then two places for the desired temperature, followed by a space, and then the last two spaces for the current (ambient) temperature.

Figure 4 shows a sample printout to the LCD display.

Each time a user makes a change with either of the push buttons, the LCD display will be refreshed. The items on the LCD will be printed from left to right using the C function ‘printf( )’[14]. Compatible with the printf( ) function is an LCD specific function, lcd\_putc( ). Lcd\_putc( ) allows the user to print integers and strings to the

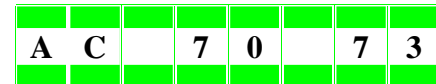


Figure 12. LCD Output

LCD. An example of printing an integer to the LCD would be: printf(lcd\_putc, “%d”, int\_variable);. The one other LCD specific function used was the ‘lcd\_gotoxy(byte x, byte y)’ function. It allows for the program to move the cursor to a specific position on the LCD. This function was used for positioning the curser before each print command. Byte x is a byte to represent which column to move the cursor to, and byte y is a byte to represent which row to move the cursor to. Byte y was always one since the LCD display only has one line. ASM ‘nop’ statements were used to add short delays to avoid timing issues when printing to the LCD [15]. ASM code was inserted into the C code by use of #asm and #endasm calls.

The entire program for the remote consists of one file called work.c. The C file was created using PCW’s PIC C Compiler. The file was typed and compiled using this program. Once all the errors were eliminated and the file compiled correctly, a HEX file was created. This HEX file could then be opened using MPLAB 5.7. From MPLAB, Microchip’s PICSTART Programmer was initialized. The proper device was selected next, the PIC16F873. Configuration bits were set to represent the proper settings such as XT for a 4MHz crystal oscillator and no code protecting which would only allow the chip to be burned a final time. The microchip is placed into the PICSTART Programmer hardware unit and the metal latch is lifted up to securely lock the microchip into place. From the software, ‘program’ is selected. The programming of the chip

takes approximately 2 minutes as it goes through all the addresses on the microchip. After completing programming, a quick verification was utilized with the PICSTART program. Afterwards, the chip would be reinserted into the breadboard containing the rest of the remote hardware. Once power was supplied to the remote unit, it could be fully tested. Upon any undesired occurrence with the remote's operation, the file would have to be edited, recompiled and re-burned. This was a time-consuming yet essential part of the process to develop a desired program for the remote unit.

### 3.4. Temperature Sensor

The temperature sensor is a vital part of the thermostat and is used to determine the ambient temperature. A sensor is located in both the base and the remote. The temperature measured by the sensor is displayed to the user and is used in determining whether the HVAC system should be turned on, off, or not changed.

The temperature sensor must be able to measure between 50°F and 90°F as stated in the design constraints. Additionally, the final signal containing the temperature information must be in digital form for calculations and displaying the ambient temperature on the LCD.

#### 3.4.1. Temperature Sensing Component

The temperature-sensing component must be able to take an analog signal that represents the temperature and convert it to a digital value. The temperature sensor must be able to measure the temperature between 50°F and 90°F as stated in the design constraints.

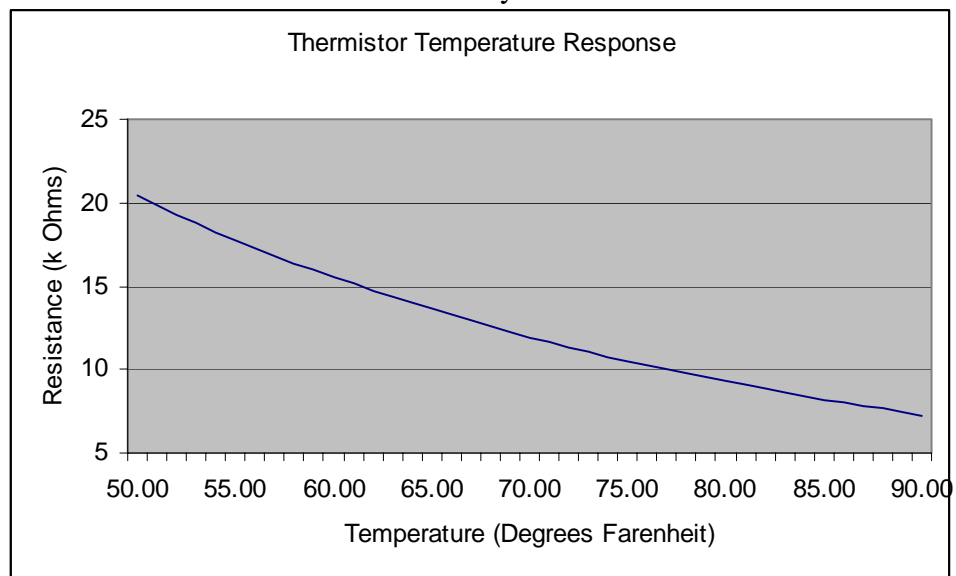
In order to measure the ambient temperature, a component was needed that would react predictably to a change in temperature over the specified temperature range. Additionally, the device's reaction had to be large enough to measure using available amplification and measurement techniques. Through research it was determined that thermocouples, RTDs, negative temperature coefficient (NTC) thermistors and positive temperature coefficient (PTC) thermistors were possible choices. The RTD, PTC thermistor, and NTC thermistor all operate on the principle of changing resistance with a change in temperature. Each type has a different reaction to temperature due to the material it is manufactured from. For example, the NTC thermistor is composed of ceramics and exhibits a decrease in resistance with an increase in temperature while an RTD is composed of thin platinum wire wound around a cylindrical core. The resistance of the RTD increases with an increase in temperature but is far less sensitive than the NTC thermistor. The thermocouple is composed of two wire leads of differing materials. As the temperature changes a voltage is induced between the two leads. All advantages and disadvantages had to be compared to determine the best temperature-sensing component for the application.

Due to its large change in resistance over the specified temperature range, the NTC thermistor was chosen. Additionally, the NTC thermistor is relatively low in cost (approximately \$1.30 in quantities of 1) and reliable. The NTC thermistor is composed of ceramic materials whose resistance decreases as the temperature increases. Several methods can be used to determine the resistance/temperature curve of a NTC thermistor. The method used depends on the data

available from the manufacturer and the intended application of use. The Steinhart – Hart method uses three measured temperature/resistance points to create a third degree polynomial equation of the thermistor’s resistance/temperature response. This method produces the most reliable curve over the full temperature range of the thermistor. In order to use this method the Steinhart – Hart method, coefficients must be provided by the manufacturer. Another method of determining the resistance/temperature curve of a NTC thermistor involves the sensitivity index value ( $\beta$ ). This parameter is provided by the manufacturer at a specified temperature. This method is valid over small temperature ranges around the temperature that  $\beta$  is defined at. The resistance ( $R_o$ ) and temperature ( $T_o$ ) at which the beta value was measured must also be known. Equation (1) below allows the temperature/resistance curve to be calculated if  $\beta$ ,  $R_o$ , and  $T_o$  are known. All temperature values in (1) are in degrees Kelvin.

$$R(T) = R_o * e^{\beta \left( \frac{1}{T} - \frac{1}{T_o} \right)} \quad (1)$$

The resistance response is nonlinear but can be corrected by several different means. Since a microcontroller will be used in the design, any nonlinearities could be removed after the analog to digital conversion process. Other methods use analog circuitry to correct the nonlinearity but it was determined that they would not be needed since a microcontroller



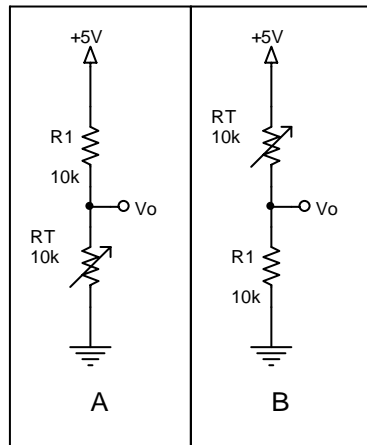
would be used.

Figure 13. Plot of thermistor temperature response.

The selected thermistor was a BC Components 2322 640 55103. This particular thermistor was chosen because of its tight tolerance values and availability. It has a  $R_o$  value of 10k $\Omega$  at 25°C and a  $\beta$  value of 3977. The  $R_o$  tolerance is +/- 1% and the  $\beta$  tolerance is +/- 0.75%. Using Equation 1, the resistance of the thermistor over the desired temperature range can be determined. A plot of the chosen thermistor’s response to temperature is shown in Figure 11.

### 3.4.2. Sensing Resistance Change

In order to determine the ambient temperature a method of sensing the thermistor's resistance had to be created. A simple voltage divider was studied first as shown in Figure 11. By



measuring the voltage across RT in Figure 2A or the voltage across R1 in Figure 2B, the temperature could be determined. The disadvantage of this method is that Vo is not capable of spanning the full 0V to 5V. It is important to span the full range so that the ADC can obtain the highest bit resolution. Various other methods were studied but the selected method of implementing the thermistor was in a wheatstone bridge. This method was chosen because minimal components were needed for its implementation, it removed the DC offset of the voltage divider and it has a long history of reliability.

Figure 14. Voltage divider circuits.

A wheatstone bridge is a common sensing circuit used in many applications requiring the measurement of a change in resistance [17]. Strain gauges, load cells, and temperature measurement are just a few of the applications where wheatstone bridges are used.

As shown in Figure 3, the wheatstone bridge consists of 4 resistive elements. At least one of the resistors must be the sensing element. Depending on the application, the bridge may contain 2, 3 or 4 sensing elements. By varying the number of elements in the bridge and the placement of them within the bridge, the transfer function will change.

If a voltage source is used to drive the bridge then the sum of the voltage drops across the resistors in each leg will equal the voltage source. As the varying element changes resistance the voltage drop across it will change. R3 and R4 are often chosen so that the voltage drop across R4 is equal to the lowest voltage drop across RT. This effectively creates a voltage at Vo that goes from 0V when RT equals R4 to a maximum when the voltage drop across RT minus R4. A current source can also be used to drive the bridge. The circuit must be resolved to take the current source into consideration.

A single thermistor in the bridge was chosen for this application. Calculations and testing indicated that the selected thermistor would produce an acceptable output range voltage. This would also help reduce costs since only a single thermistor would be needed in temperature circuit. The bridge is driven by a 5V source since a voltage source is readily available. The source is also regulated enough to provide acceptable output. If a current source were to be used more parts would be needed to implement the source increasing costs and complexity.

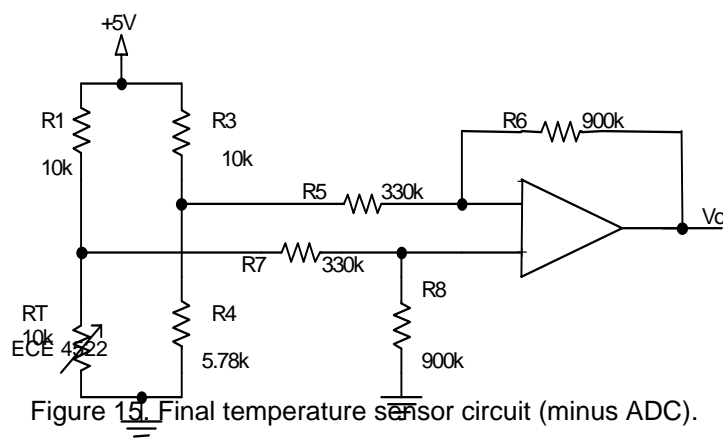


Figure 15. Final temperature sensor circuit (minus ADC).

### 3.4.3. Signal Amplification

Based on the chosen resistor values for R1, R3, and R4 as well as the thermistor, Vo ranges from 0V to 1.66V. This was still not an

acceptable value for the ADC. In order to amplify the signal to 5V an opamp configured in differential mode was used. Instrumentation amplifiers were considered if the opamp would not work, but the opamp, in a differential configuration, amplifies the difference between the input signals. This configuration was chosen because it would take advantage of the differential signal coming from the bridge network. An additional advantage is that it would reduce common mode noise. More circuitry would be required if a separate filter circuit were to be implemented [17, 18, 19].

Since the other components in the remote circuit and most of the components in the base operate at 5V, the opamp power rails would ideally operate at 5V. Initial testing of the opamp circuit involved a u741. This opamp would not work on a single supply rail and would not produce rail-to-rail output voltages. After researching and testing several opamps, the OPA340 was selected. It can operate on a single supply of 5V. The output is capable of getting within 1mV of the supply rails. This allows the ADC input to span the full 0V to 5V range over the temperature range of 50°F to 90°F creating a high digital resolution.

### 3.5. HVAC Interface

The most popular HVAC control configurations use four or five wires [5]. The uses of these wires are:

- RC – Transformer Cooling (24 VAC, 1.2A)
- RH – Transformer Heating (24 VAC, 1.2A)
- G – Fan
- W – Heat
- Y – Cool

In a typical five-wire system, RC is connected to Y to activate the cooling system or G to activate the fan motor. In contrast, RH is connected to W to activate the heating system. The sole difference between the four-wire and five-wire systems

is that four-wire systems use only one transformer to activate the systems. This configuration can be utilized with a five-wire thermostat by shorting the RC and RH terminals and connected the “R” wire to either terminal. Due to the compatibility of a five-wire thermostat with a four-wire system, we are designing our thermostat for the five-wire system [4].

A method was needed to connect the correct wires to turn the HVAC system on and off. Additionally, isolation was needed between the low voltage control side of the PIC and the higher voltage, high current side of the HVAC system. Both relays and optocouplers were examined as possible solutions. A basic optocoupler consists of an LED and phototransistor inside an enclosed case. When current flows through the LED, the LED emits light. The light is detected by the phototransistor, which allows current to flow through device. The main disadvantage of the optocoupler is that it does not handle transients very well. Large transients can easily destroy the device. A more robust solution was needed. Relays were considered next. A relay is an electromechanical device that also switches high voltages and currents with a low

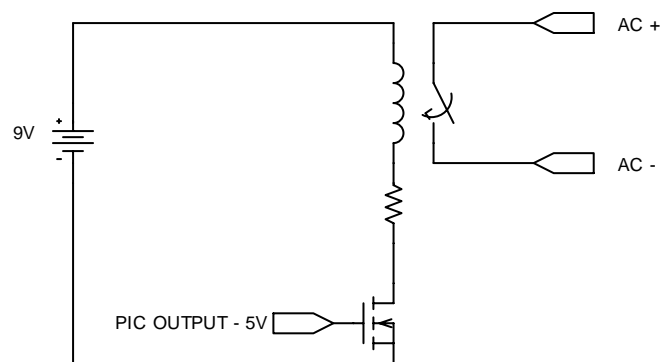


Figure 16. PIC and relay interface circuit.

voltage and current control signal. When a small current flows through the coil an electromagnet is created. This magnet mechanically closes the contacts allowing current to flow on the high voltage side. Due to possible transients in the HVAC system and the inductive load of the transformer, a relay was chosen. The selected relay requires 9V and 25mA to energize the contacts. The contacts are rated at 120VAC and 5A. In the final design, this relay will most likely be replaced with a smaller relay rated at 2 or 3 amps.

Under the intended wiring convention, three outputs are used; therefore, we need at most three relays to connect the output lines with the transformer inputs. Three pins on the PIC16F873 will be assigned the responsibility of controlling these relays; B5 for the air-conditioning relay, B4 for the heating relay, and B2 for the fan relay.

The PIC's output is rated at 25mA max while the relay coil requires 95mA to energize. Some type of interface circuit was needed between the PIC and the relay coils. A power MOSFET was chosen to handle this task [20]. The MOSFET's gate threshold is 3V-4V, which can be provided by the PIC.

The HVAC interface's connection with the microcontroller will be:

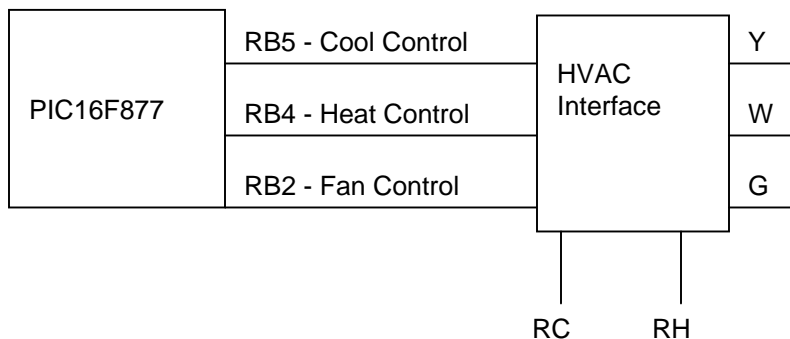


Figure 17. Diagram of the PIC's connection with the HVAC interface

### 3.5 Power

Although power is not considered in the design constraints, it is an important part of the system. The voltage tolerance of the remote and base is specified at  $\pm 0.25V$ . This limit is due to the wireless modules used. The input voltage range as well as the current consumption is very different for the two circuits.

The base is powered from a regulated wall cube power supply. The chosen supply is a 5V switching supply capable of delivering 2A of current. The tolerance rating of the supply is  $\pm 0.25V$ , which meets the needed supply tolerance. A linear regulator could be added to reduce the cost of the wall power supply. If a linear regulator were implemented, an unregulated 9V or 12V DC supply could be used.

The remote's power supply is more complicated than the base. The remote's power must be supplied from batteries since it is a portable device. Additionally, because batteries are being used, the input voltage varies with use and load. To reduce weight and increase consumer appeal, the number of batteries used needed to be minimized. Different power regulators were studied such as boost converters, charge pumps, and linear regulators. The boost converters were efficient at either low or high current, but not both. The charge pumps could not supply the needed current. The Texas Instruments TPS61092 was the only converter we were able to find that could operate efficiently at low and high currents. We attempted to use this chip, but were unable to successfully prototype with it because of its surface mount package.

A linear regulator was used in the final design. The Texas Instruments REG102-5 was used. This regulator was selected because it could operate at voltages less than 5V. This is very beneficial for our application because it guarantees that the battery supply voltage will stay at or below 5V. It will also allow the circuitry to continue to operate after the voltage drops below 5V. Our system will operate on a supply voltage of 5.25V to 4.75V.

#### 4. TEST SPECIFICATION

The following tests will allow an individual to determine if the constraints of the design have been met. The tests consist of a procedure for performing the test and the expected results of the test. If the expected and measured values agree then the design has met the constraints. If the test results do not agree with the expected results then the design constraints have not been met.

##### 4.1. Temperature Sensing

The full measurement range will be tested by adjusting the ambient temperature between 49°F and 91°F. Values read from the ADC will be compared with the expected values to determine if they are within the specified range.

Remote Test Procedure:

1. Load modified software into the remote's PIC that will display the measured temperature over the full measurement range. The modified software will be the same as the normal software with the exception that the LCD display will be able to display the full measurement range and not just the user adjustable range.
2. Place the remote in an environment that has an ambient temperature of 49°F. Once the remote is in the environment, allow 3 minutes for the remote's temperature to stabilize. Record the ambient temperature as measured by a second, reference thermometer and the temperature as displayed on the remote unit.
3. Increase the ambient temperature in 1°F increments. After an adjustment is made, wait until the environment has reached the increased temperature. Once the environment is at the desired temperature wait 3 minutes and record the temperature measured on the display. Additionally, record the temperature indicated on the reference thermometer. Continue this process through 91°F.

4. Compare the temperatures recorded from the remote with those obtained by the reference thermometer. At 49°F the expected temperature measured by the remote is 50°F. Between 50°F-90°F the expected temperature measured by the remote is +/- 1°F of the reference thermometer's values. At 91°F the expected temperature measured by the remote is 90°F.

#### Base Test Procedure:

5. Load modified software on the PC that will display the measured temperature over the full measurement range. The modified software will be the same as the normal software with the exception that the screen will have the ability to display the full measurement range and not just the user adjustable range.
6. Place the base hardware unit in an environment that has an ambient temperature of 49°F. Once the base is in the environment, allow 3 minutes for the bases temperature to stabilize. Record the ambient temperature as measured by a reference thermometer and the temperature as displayed on the PC.
7. Increase the ambient temperature in 1°F increments. After an adjustment is made, wait until the environment has reached the increased temperature. Once the environment is at the desired temperature wait 3 minutes and record the temperature measured on the PC. Additionally, record the temperature indicated on the reference thermometer. Continue this process through 91°F.
8. Compare the temperatures recorded from the PC with those obtained by the reference thermometer. At 49°F the expected temperature measured by the PC is 50°F. Between 50°F-90°F the expected temperature measured by the PC is +/- 1°F of the reference thermometer's values. At 91°F the expected temperature measured by the PC is 90°F.

## 4.2. Packaging

The unit will also be placed on a scale to determine that its weight is less than the 10-ounce specification. The finished packaging will be mounted on a piece of dry wall to verify that it can be wall mounted. The HVAC wires, a RS232 cable, and a power cable will be connected to the back of the unit through the drywall. The unit will then be turned on. The PC will send data to the base unit asking the base unit to turn on the heating and cooling systems. This will verify that the connections are working properly. The RS232 and power wires will then be removed and connected through the bottom of the unit. Once again, the PC will ask the base unit to turn on the heating and cooling systems to verify that the RS232 and power connections are working properly on the bottom of the unit.

## 4.3. RS-232E

The computer will communicate with the base hardware unit by RS232. Several components will be controlled from this port. The input data will include a local temperature sensor and the RF data coming from the portable unit. The output data will be the information sent to the HVAC unit. A RS232 connection will be used to control the flow of data between these components. The diagram below shows the flow of data for the base unit.

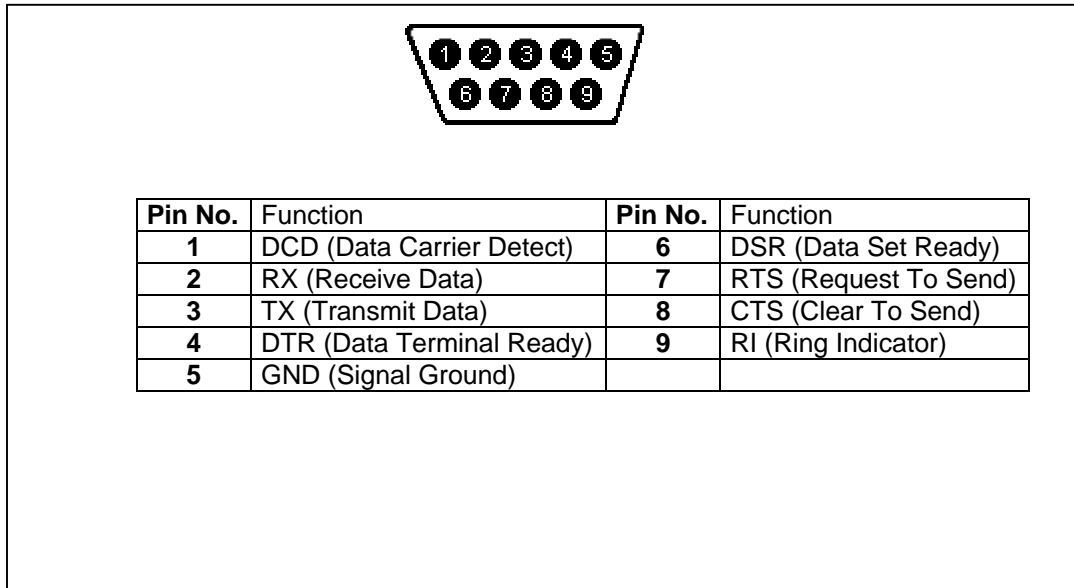


Figure 18. Pin-out diagram for a 9-pin serial port.

The RS232 connection consists of data and handshaking signals. To test the data and handshaking signals, a bit sequence will be sent from the PC to the base hardware unit over the RS232 connection (the actual value of the bit sequence will be determined once more progress has been made on the hardware). The data will be interpreted by the base hardware unit and a reply bit sequence will be sent to the PC (the actual value of the bit sequence will be determined once more progress has been made on the hardware). The received data will be displayed on the PC and compared with the expected received data. If the received and expected data are the same this will indicate that the RS232 data and handshaking paths are working properly.

#### 4.4. LCD Output

The LCD output displays three pieces of information, desired temperature, ambient temperature, and mode of operation (AC, heat, etc.). A voltage source will be used to simulate the output of the temperature circuit. The final PIC software will be loaded onto the PIC before any of the following tests are performed. The testing of the LCD will be done in three parts.

##### Ambient Temperature Test Procedure:

9. Power the remote circuit with an external voltage source. Connect a second voltage source to the ADC input of the PIC. The negative terminals of both supplies should be connected together. The voltage source supplying power to the remote circuit should be set to 5V before being turned on. The voltage source connected to the ADC should be set to 0V before being turned on.
10. Slowly sweep the voltage source connected to the ADC from 0V to 5V. Record the displayed ambient temperature. The measured values should start at 55 and increment to 85 in steps of 1. The values that will be changing on the LCD are the first two left most values as shown in Figure 4.4.1, test A. The third character position will remain blank.

**Desired Temperature Test Procedure:**

11. Power the remote unit with its normal power supply.
12. Press the “Temperature Down” position of the joystick until the fourth and fifth positions of the LCD read 55. Figure 4.4.2 shows where the desired temperature will be output on the LCD.
13. Press the “Temperature Up” position one time and record the reading on the fourth and fifth positions on the LCD. Repeat this process until the fourth and fifth positions of the LCD read 85. Figure 4.4.1, Test B, shows where the desired temperature will be output on the LCD.

If the recorded values increment from 55 to 85 in increments of one, then the desired temperature portion of the display is working correctly.

**Mode of Operation Test Procedure:**

14. Power the remote circuit with its normal supply.
15. Toggle the “Mode of Operation” position of the joystick until OFF is shown. Record the three right most values displayed on the LCD. The expected output is shown below in Table 4, Test C.
16. Toggle the “Mode of Operation” position of the joystick until AC is shown. Record the three right most values displayed on the LCD. The expected output is shown below in Table 4, Test D.
17. Toggle the “Mode of Operation” position of the joystick until HEAT is shown. Record the three right most values displayed on the LCD. The expected output is shown below in Table 4, Test E.
18. Toggle the “Mode of Operation” position of the joystick until FAN is shown. Record the three right most values displayed on the LCD. The expected output is shown below in Table 4, Test F.
19. Toggle the “Mode of Operation” position of the joystick until AUTO is shown. Record the three right most values displayed on the LCD. The expected output is shown below in Table 4, Test G.
20. If all of the recorded an expected values are the same then the LCD correctly displays the modes of operation.

A	Ambient Temp. (10's Position)	Ambient Temp. (1's Position)	Blank	XX	XX	XX	XX	XX
B	XX	XX	Blank	Ambient Temp. (10's Position)	Ambient Temp. (1's Position)	XX	XX	XX
C	XX	XX	Blank	XX	XX	O	F	F

D	XX	XX	Blank	XX	XX	Blank	A	C
E	XX	XX	Blank	XX	XX	Blank	H	T
F	XX	XX	Blank	XX	XX	Blank	F	N
G	XX	XX	Blank	XX	XX	Blank	A	T

Table 4. Position of ambient temperature in relation to LCD output (XX indicates a “don’t care”)

#### 4.5. Anticipator

The anticipator will be tested with both the remote and the base. To test the anticipator function on the remote, adjust the ambient temperature to 70°F. Set the heater anticipator to 3 degrees Fahrenheit and the mode to heat. Over a 5-minute period, change the ambient temperature to 75 degrees Fahrenheit. Monitor the HVAC output and note at what temperature the HVAC system cutoff. Repeat this procedure for heater anticipator settings of 2 and 1. The recorded values should be 75 minus the heater anticipator setting. Repeat this procedure for the base. To test the anticipator on the AC system, set the AC anticipator to 3, the operating mode to AC, and the ambient temperature to 80 degrees Fahrenheit. Follow the same procedure as above except reduce the temperature to 75°F. The recorded values should be 75 plus the AC anticipator setting.

#### 4.6. Over-Current

To test the over-current protection, an attempt will be made to turn the HVAC system on and off quickly.

##### Test Procedure:

1. Turn on the remote and place it in AC mode.
2. Adjust the ambient temperature to 70°F. Set the desired temperature to 67°F, or until the AC system is activated.
3. Wait for the system to reach the desired temperature and turn off.
4. Once the system is turned off, adjust the desired temperature to 63°F (or 4 degrees below the desired temperature used in step 2). Every 5 seconds adjust the desired temperature from 63°F to 62°F and then back to 63°F after another 5 seconds. Note how long it has been system the system AC turned off each time the desired temperature is adjusted.
5. Continue with step 4 until the AC system turns back on. The time it takes should be at least as long as the time set by the over-current protection system, and at most the time set by the over-current protection system plus one minute.

Repeat this process with the base unit having control of the AC system. To test the over-current with the heating system, follow the test procedure below.

##### Test Procedure

6. Turn on the remote and place it in heat mode.
7. Adjust the ambient temperature to 70°F. Set the desired temperature to 73°F, or until the heating system is activated.

8. Wait for the system to reach the desired temperature and turn off.
9. Once the system is turned off, adjust the desired temperature to 77°F (or 4 degrees above the desired temperature used in step 2). Every 5 seconds adjust the desired temperature from 77°F to 78°F and then back to 77°F after another 5 seconds. Note how long it has been system the system heating turned off each time the desired temperature is adjusted.
10. Continue with step 4 until the heating system turns back on. The time it takes should be at least as long as the time set by the over-current protection system, and at most the time set by the over-current protection system plus one minute.

#### 4.7. HVAC Connectivity

Our system will connect to HVAC units with a standard 5-wire interface. Some 5-wire systems contain a sixth wire that is a common. HVAC units that have the extra common wire will operate with this thermostat. No standard has been found that specifies this. However, this thermostat is not compatible with heat pump systems. The 5 wires are as follows:

- RC: Cooling Transformer
- RH: Heating Transformer
- G: Fan Relay
- Y: Cooling Relay
- W: Heat Relay

The HVAC interface will be tested at the rated voltage and current conditions of 24VAC and 1.2A to verify that they meet the specifications. This will be done by applying the rated voltages and currents to the load side of the circuitry for AC, heat and fan modes. It will be verified that the HVAC interface operates correctly in the on and off conditions as well as when switching.

#### 4.8 Supported Operating Systems

Windows 95, Windows 98, Windows ME, and Windows XP will be supported by the PC software. The software will be tested for correct operation and stability. Correct operation will be tested by sending and receiving data to and from the base unit. The purpose of the communications test is to determine that the drivers and operating system work correctly in transmitting and receiving data.

To test communication, a command will be sent to the base unit asking it to turn on the heating unit. The heat relay will then be checked to see that the contact is closed. The same will be followed again, except that the AC will be turned on and the AC contact checked. This will verify that the PC is correctly sending the data.

To test data received by the PC, the base unit will send temperature readings for three different temperatures. The received temperatures will be compared to the temperature that was expected. If the temperatures match, then the PC is correctly receiving the data.

The program will operate for four hours on each operating system to determine stability. If the program does not stall the computer, then the program will be considered stable.

### 5. TEST CERTIFICATION

This section contains the test results of the test certification. Tests were performed according to the procedures found in the Test Specification section. All of the tests results passed the design constraints.

#### 5.1. Temperature Sensing

The temperature was measured over a range of 65°F to 75°F. The ambient temperature of a glass thermometer was recorded along with the output voltage of the opamp. A table containing temperature values and calculated opamp output voltages was compared to the recorded values. Table 5 contains the calculated and recorded data. Figure 16 compares the calculated and recorded data.

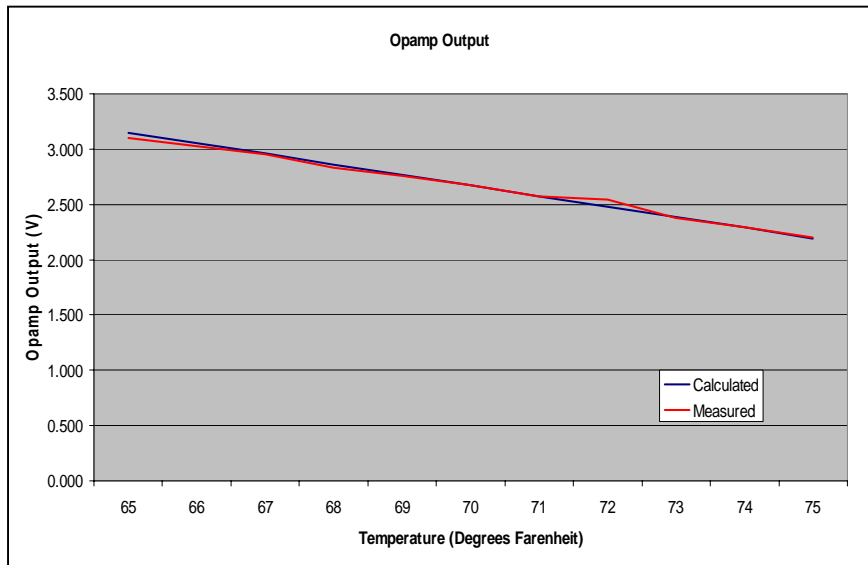


Figure 19. Opamp output over temperature range of 65 to 75 degrees Fahrenheit.

Temperature (Degrees Fahrenheit)	Calculated Output (V)	Opamp Output (V)
75	2.194	2.20
74	2.289	2.29
73	2.384	2.38
72	2.479	2.54
71	2.575	2.57
70	2.671	2.67
69	2.766	2.76
68	2.862	2.83
67	2.958	2.95
66	3.054	3.03
65	3.150	3.10

Table 5. Comparison of recorded and calculated Temperature sensor data.

This same test was also performed using the ADC in the PIC. The ambient temperature was output to the LCD and recorded. This temperature was compared with a thermometer and the temperatures matched.

Thermometer Temperature	LCD Temperature
75	75
76	76
77	77
78	78
77	77
78	78
79	79
80	80
81	81
82	82
83	83

## 5.2. Size and Weight

The weight of the remote was measured at 8.2oz. This is within the design constraint of 10oz. The maximum depth of the remote is 1.5". The length and width is 5.6"L x 3.2"W. Honeywell's remote thermostat is 4.75" L x 3.5"W 1.5" D which is very close to ours.

## 5.3. RS-232E

The RS-232E interface works correctly. Data was sent from the PC to the base hardware unit. The base hardware sent the correct reply to the PC.

## 5.4. LCD Output

The LCD works correctly. The ambient temperature, desired temperature, and mode are all displayed on the LCD. Furthermore, as the desired temperature and modes are adjusted to all possible values, the LCD displays the correct data. All ambient temperature values are also displayed correctly.

## 5.5. Anticipator

The input temperature value was driven within the range specified by the anticipator. At this point, the PC switched the HVAC relays off. The temperature was reset, and the test was repeatedly conducted and passed every time.

## 5.6. Over-Current

The AC over-current protection was tested using a modified program along with some loops in Visual Basic. A while loop was setup to continuously send data out of the serial port. A loop-back connector was placed on the serial port to send the data directly back to the PC. After the OnComm procedure receives the first set of data the Rthreshold property was set to zero to disable the serial port. The timer function was called to wait exactly one minute before the Rthreshold was set back to one. While the serial port is disabled the while loop is continuously sending information through the loop-back connector and back to the PC. We verified that the PC would not show any received data until after the six-second-time period, therefore proving that the over-current protection works properly.

### 5.7. HVAC Connectivity

For the prototyping stage, we decided to use 120VAC 5A relays. This was more than ample to handle the 24VAC 1.2A incoming from the HVAC. We connected our relays with their control circuit to a power supply and switched relays on and off, simulating an actual use-case. The relays performed as expected, and this test passed.

### 5.8. Cost

The parts total for the remote and the base came to a grand total of \$56.38. The cost breakup is as follows:

<b>Remote</b>	
Components	\$ 23.56
PCB	\$ 1.26
Packaging	\$ 3.10
<b>Remote Total</b>	<b>\$ 27.92</b>
<b>Base</b>	
Components	\$ 22.73
PCB	\$ 0.98
Packaging	\$ 4.75
<b>Base Total</b>	<b>\$ 28.46</b>
<b>Total Cost</b>	<b>\$ 56.38</b>

Table 5. Cost data for the Portable Thermostat

## 6. SUMMARY AND FUTURE WORK

At the conclusion of this project, the portable thermostat successfully works. The A/D converter on the PIC16F873 chips, found on both the remote and base unit, accurately gives the ambient temperature from the thermistor reading. Communications via the RS-232 and RF module work without the loss of any data. Also, the modes of HVAC operations, AC and heat anticipators, and over-current protection all work as desired. At this point in time, the project is complete and has met all of design constraints.

For future work, there are several directions in which the project could be taken. First, the PC could be totally eliminated from the project. The base would have an LCD and buttons to change the desired temperature and modes. These two units would have to communicate with each other. One unit would be designated as the base unit and one unit would be designated as the remote unit. The remote unit would have master control over the base unit similar to the current system. This base unit would be mounted to the wall similar to a traditional thermostat.

Another more complicated way to expand the system would be to create a system that works with multiple base units and multiple remotes. Many households have several different thermostats that control different HVAC systems within the house. This would require some code of identification to pair up the correct remote and base units without having any interference.

Minor improvements include refining the power supply to prolong battery life. The packaging could also be designed to withstand shock such as being dropped. An RF unit could be created so that the Maxstream wireless module would not be needed.

## 7. ACKNOWLEDGMENTS

We would like to thank Dr. Marion Hagler for his leadership and advice, Dr. Picone for guidance in our project, Jan Axelson for serial port programming help, and the staff from Maxstream, Inc., for suggestions on how to use their wireless systems. We also want to acknowledge Jamie Foster of Martin's Heating and Cooling, Mendenhall, MS, for his expert advice in HVAC systems and Jim Gafford for his help with Orcad and Layout Plus.

## 8. REFERENCES

- [1] California Energy Commission, "Get comfortable you're your setback thermostat", [http://www.energy.ca.gov/efficiency/home\\_energy\\_guide/SETBACK\\_THERMOSTATS.PDF](http://www.energy.ca.gov/efficiency/home_energy_guide/SETBACK_THERMOSTATS.PDF)
- [2] Verlag fur Architektur Artemis Zurich, *Ergonomics of the Home*. London. Taylor and Francis Ltd. 1973
- [3] J. Axelson, *Serial Port Complete*. Madison, WI: Lakeview Research, 2000.
- [4] Jamie Foster, Martin's Heating and Cooling, Mendenhall, MS. Personal Interview, March 25, 2003.
- [5] E.F. Mahoney, *Reading and Interpreting Diagrams in Air Conditioning and Refrigeration*. Reston, VA. Reston Publishing Company, Inc. 1983.
- [6] Quin Jones, Maxstream Wireless Inc., Personal Interview, April 21, 2003.

- [7] S. Brown, *Fundamentals of Digital Logic*. New York City, NY: McGraw-Hill Higher Education, 2000.
- [8] D. I. Schneider, *Visual Basic 6.0*. 4<sup>th</sup> ed. Upper Saddle River, NJ: Prentice, 1999.
- [9] R. J. Tocci, *Digital Systems*. Englewood Cliffs, NJ: Prentice-Hall, INC., 1977.
- [10] R. Grier, *Serial Communications 3*. Stanwood, WA: Mabry Publishing, 2002.
- [11] G. Perry, *Teach Yourself Visual Basic 6.0 in 24 Hours*. Indianapolis, IN: Sams Publishing, 1998.
- [12] P. H. Anderson, *PIC16F87X Tutorial by Example*. Baltimore, MD: June 2001.
- [13] J. Uffenbeck, *The 80x86 Family: Design, Programming, and Interfacing*. 2nd ed. Upper Saddle River, NJ: Prentice, 1998.
- [14] J. P. Cohoon and J.W. Davidson, *C++ Program Design: An Introduction to Programming and Object-Oriented Design*. 2nd ed. Boston, MA: WCB/McGraw-Hill, 1999.
- [15] K. R. Irvine, *Assembly Language for Intel-Based Computers*. 3rd ed. Upper Saddle River, NJ: Prentice, 1999.
- [17] R. Dorf. *Introduction To Electric Circuits*. New York, NY: Von Hoffman Press, 2001.
- [18] D. Neamen. *Electronic Circuit Analysis and Design*. New York, NY: McGraw-Hill Higher Education, 2001.
- [19] S. Ball. *Analog Interfacing to Embedded Microprocessors*. Woburn, MA: Newnes, 2001.
- [20] N. Mohan. *Power Electronics*. Hoboken, NJ: Wiley, 2003.
- [21] Myke Predko, *Programming Robot Controllers*. New York, NY. McGraw-Hill. 2003.