School of Engineering Science
Simon Fraser University
Burnaby, BC V5A 1S6
holla@hollahome.com

March 10, 2005

Lakshman One
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

RE: ENSC 440 – Design Specification for a Remotely Accessible Temperature Control System

Dear Mr. One,

Please find attached the document; *Design Specification for a Remotely Accessible Temperature Control System*, Holla Home Solutions' design specification for ENSC 440: Capstone Engineering Project.

This document details the design specifications for the Remotely Accessible Temperature Control System. This includes the system hardware and software specifications for the Engine, Wireless Microcontroller, Furnace, Thermometer, LCD, Button, and User Interface modules. Test plans for these specifications are also provided in this document.

Holla Home Solutions is comprised of four determined, talented, and attractive senior engineering students: Jennifer Fong, Steve Judd, Wojtek Piaseczny, and Chris Richardson. The Holla Home team is dedicated to bringing innovative solutions in our never ending search to efficiently and ecologically manage our homes and workplaces.

If you have any questions or concerns regarding our design specifications, contact me by e-mail at holla@hollahome.com or by phone at (604) 298-1885.

Sincerely,

Jennifer Fong

Jennifer Fong
President & CEO
Holla Home Solutions

Enclosure: Functional Specification for a Remotely Accessible Temperature Control System

holla
home solutions

Design Specification for a
Remotely Accessible Temperature Control System

Project Team: Jennifer Fong
Stephen Judd
Wojtek Piaseczny
Chris Richardson

Contact Person: Chris Richardson
holla@hollahome.com

Submitted to: Lucky One - ENSC 440
Mike Sjoerdsma - ENSC 305
Scott Logie - ENSC 305/440 TA
Tony Ottaviani - ENSC 305 TA
Amir Masoud Niroumand - ENSC 440 TA
Steve Whitmore
School of Engineering Science
Simon Fraser University

Issued Date: March 10, 2005

Revision: 1.0

Executive Summary

Since its inception, the Internet has revolutionized the communications world and has provided a powerful means of transferring information. Access to the Internet is available now more than ever, which is why the team at Holla Home Solutions envisions using the Internet to offer peace of mind and comfort, as well as decreased energy consumption in the home and office space.

Holla Home Solutions will develop a *Remotely Accessible Temperature Control System* as a first step in achieving their vision. The system allows the user to control their home's current heat settings, as well as to set a pre-programmed temperature schedule, from any Internet connection. Unexpected changes in schedule or simple bouts of absent-mindedness can be conveniently managed while on the road or even out of the country. Still remaining is the ability to make changes to settings manually in the home.

As part of the first phase of development, the Holla Home Solutions team will construct the basis of a remotely accessible smart appliance control system. There will be a potential to integrate the control of multiple appliances remotely from a single user interface. Additional controllers for these devices will follow in later development phases. The system will work in conjunction with existing home heating systems and wireless Internet connections.

As proof of concept, a prototype of the remotely accessible temperature control module will be completed by the beginning of April, 2005.

Table of Contents

List of Figures

List of Tables

Glossary

ARP	Address Resolution Protocol
HTTP	Hypertext Transfer Protocol
IEEE 802.11b	Often called Wi-Fi - is backward compatible with 802.11. The modulation used in 802.11 has historically been phase-shift keying (PSK). The modulation method selected for 802.11b is known as complementary code keying (CCK), which allows higher data speeds and is less susceptible to multipath-propagation interference.
IP	Internet Protocol.
PCMCIA	Personal Computer Memory Card International Association
RATCS	Remotely Accessible Temperature Control System
SOAP	Simple Object Access Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

1 Introduction

The Remotely Accessible Temperature Control System allows access and control to a user's home or office heating system from any Internet connection. This is the first phase in the development of a remotely accessible smart appliance control system, which would allow remote control to a network of integrated appliances. Working in conjunction with a home's existing heating system and wireless Internet connection, the temperature control module is scheduled for completion at the beginning of April, 2005.

1.1 Scope

This document describes the design specifications for an operational Remotely Accessible Temperature Control System. These design specifications begin with the system overview, detailing the system's functional, interface, and communication layers. The system overview is followed by the software design necessary to integrate each of the system's modules. Documented next is the required hardware specifications for the main modules that compose the system. Lastly, the interface design is outlined. Test plans for these specifications are included at the end of this document.

1.2 Intended Audience

The design specification is to be used by the design engineers as a reference during the design and implementation of the prototype. The Holla CEO and management will use this document to measure project performance and implementation objectives.

2 System Overview

The Holla Home Solutions Remotely Accessible Temperature Control System (RATCS) allows the user to control the temperature of his/her home from anywhere. RATCS connects to a home gas-furnace and to an existing home wireless infrastructure, and exposes an interface visible and accessible to the world. Figure 1 illustrates the relationship between the RATCS module and the required home infrastructure.

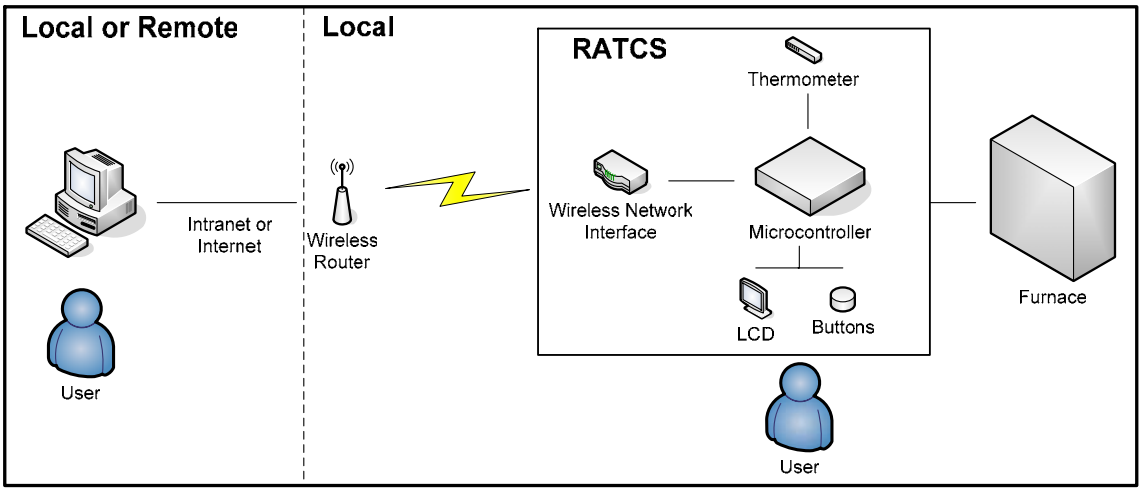

Figure 1: System Overview

The system can be split into three mutually exclusive subsystems: thermostat functionality, interface, and communication. Together, these subsystems will form the single coherent system that will allow access to a home's furnace through generic third party software. The purpose of each subsystem is introduced below, and thoroughly designed later in this document.

2.1 Functional Layer

The functional layer provides the core functionality of a thermostat. This includes the ability to control the temperature of a home in real-time, as well as to program the temperature in-advance.

2.2 Interface Layer

The interface layer provides two distinct portals into RATCS: local and remote. The local access will be provided through a set of buttons and a display module. The remote interface is provided by having an on-board web server. This service complies with industry standards to integrate easily with commonly used web browsers such as Mozilla

FireFox and Microsoft Internet Explorer. Both interfaces will provide the same functionality.

2.3 Communication Layer

The communication layer is used exclusively for remote access. Using standard wireless communication protocols allows RATCS to quickly connect to an existing wireless network. Once it is accessible to the network, exposing RATCS to the world is simply a matter of opening a communication port on the home router.

3 Software Design

3.1 Buttons

The software component for the operation of the buttons module consists of sending a button interrupt to the main system. The button interrupt returns the parameter of which button has been pressed, and the main system determines and carries out the appropriate functions accordingly.

Note, however, that prior to the transmission of a button interrupt to the main system, and before functions attributed to a change in button state can occur, the signal from the buttons must be debounced. A software debouncing subroutine has therefore been developed and is called to decipher the user's input to the digital system. The flowchart for a button input to the system, including the debouncing subroutine, is shown in Figure 2. Each button will be sampled and handled individually according to the button control loop.

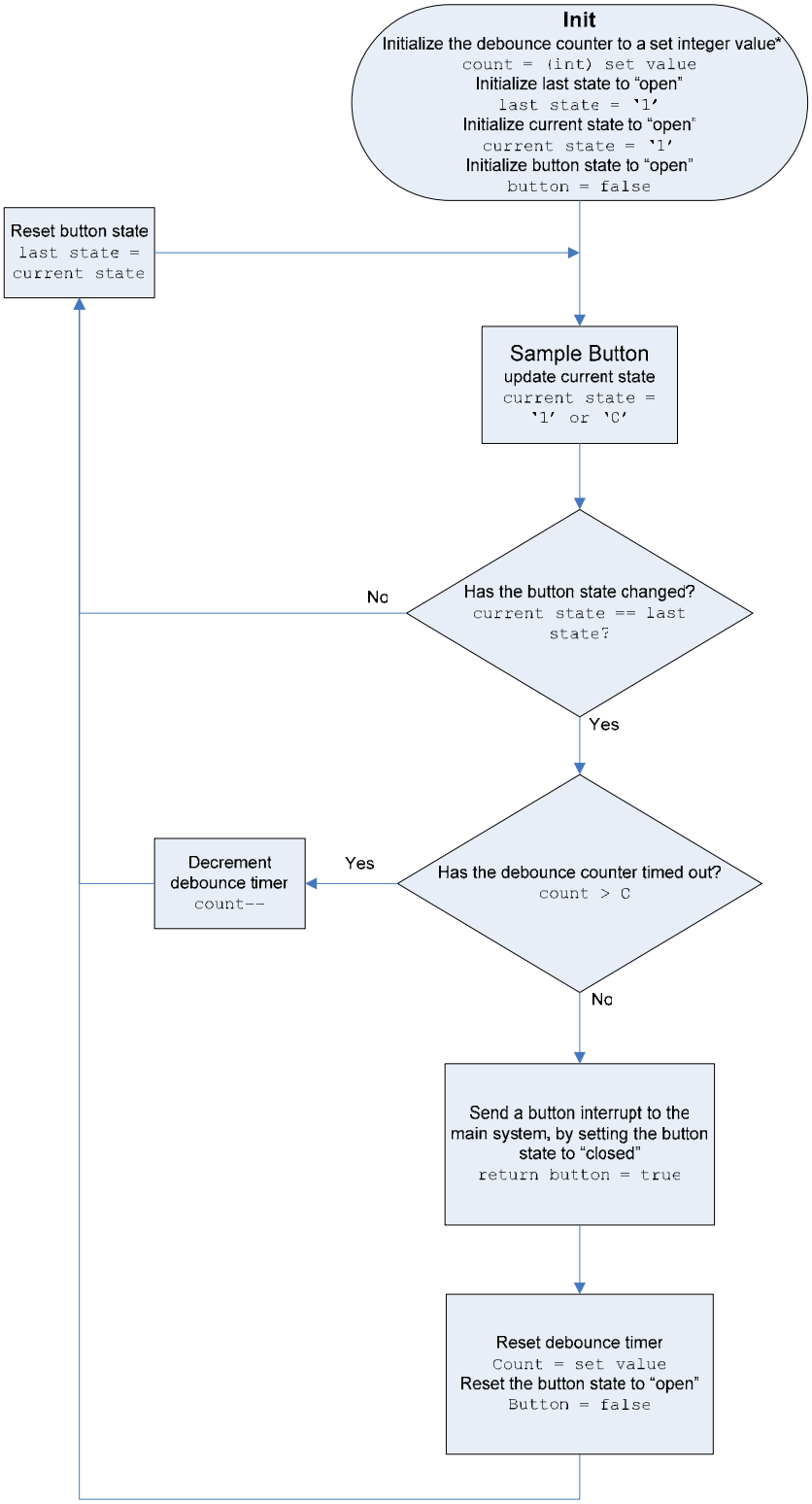

Figure 2: Button Control Loop

3.2 LCD

The data transfer to the LCD interface will be sent from the RB0 pin on the PIC18F452 microcontroller, at 9600 baud, 8 data bits long, no parity and 1 stop bit. Prior to a character being written to the LCD, a position request is sent. Operating in this manner will ensure that the new character will be in the correct position. Using absolute referencing around the 2x16 character screen will enable the software to directly pick out the next desirable location for display. The flow chart for writing a character to the LCD screen is exhibited in Figure 3, below.

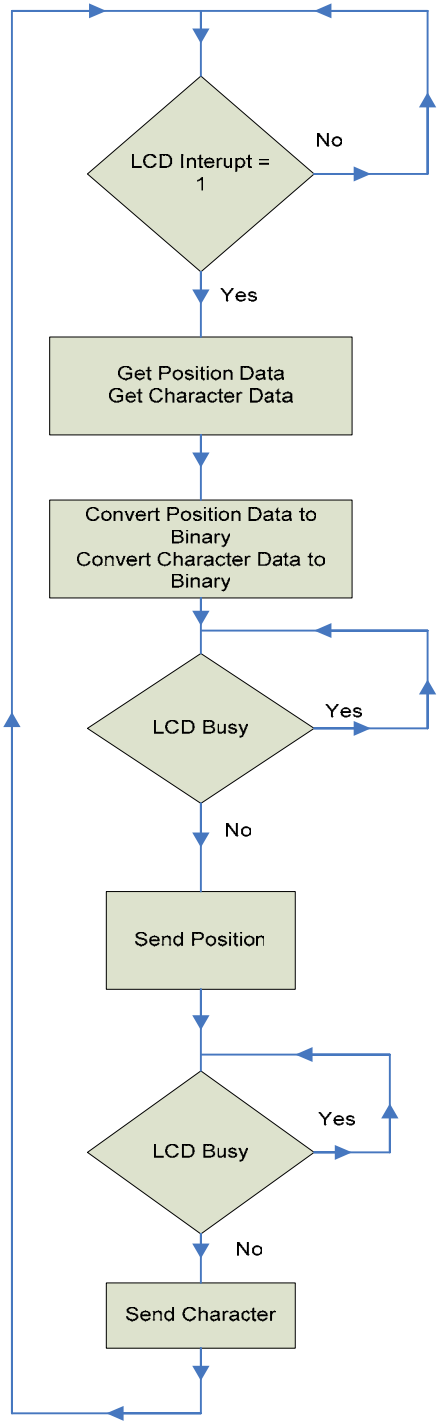

Figure 3: LCD Control Loop

3.3 Thermometer

Figure 4 below shows the functional diagram for the Digital Thermometer.

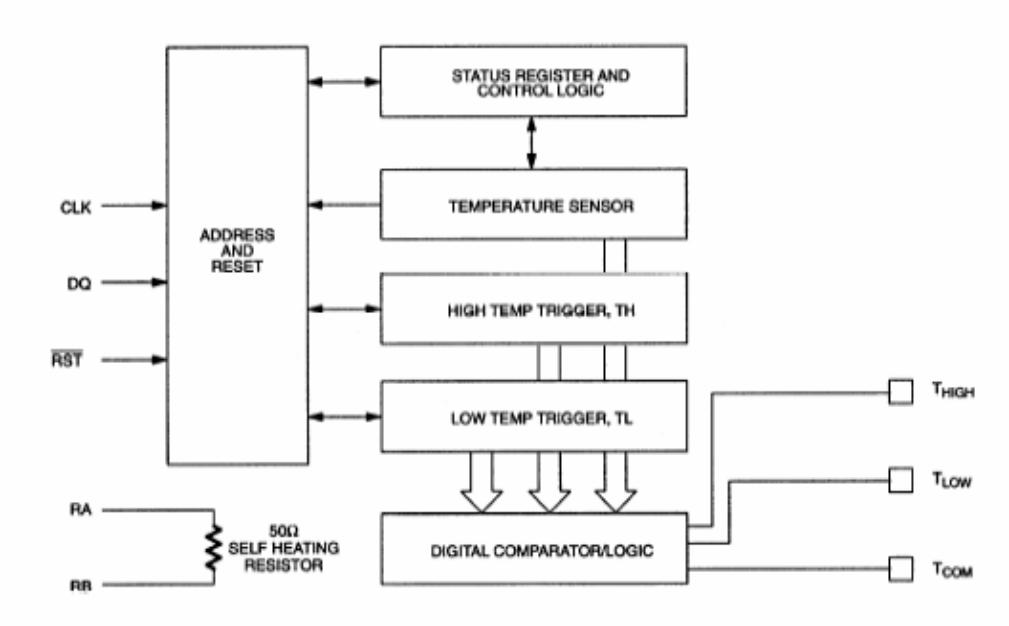

Figure 4: Digital Thermometer Functional Diagram

When a Read Temperature command is sent, the DS1620 will provide a 9-bit, two's complement value representing the temperature with a precision of $0.5°C$. Comparing this number with the temperature lookup table shown below in Table 1, will provide the temperature reading that will be displayed on the LCD.

Table 1: Thermometer Temperature Lookup Table

TEMP	Digital Output (Binary)	Digital Output (Hex)
$+125°C$	0 11111010	00FA
$+25°C$	0 00110010	0032
$+0.5°C$	0 00000001	0001
$0°C$	0 00000000	0000
$-0.5°C$	1 11111111	01FF
$-25°C$	1 11001110	01CE
$-55°C$	1 10010010	0192

The functional diagram depicting the blocks onboard the DS1620, which is used to make temperature readings, is shown below in Figure 5.

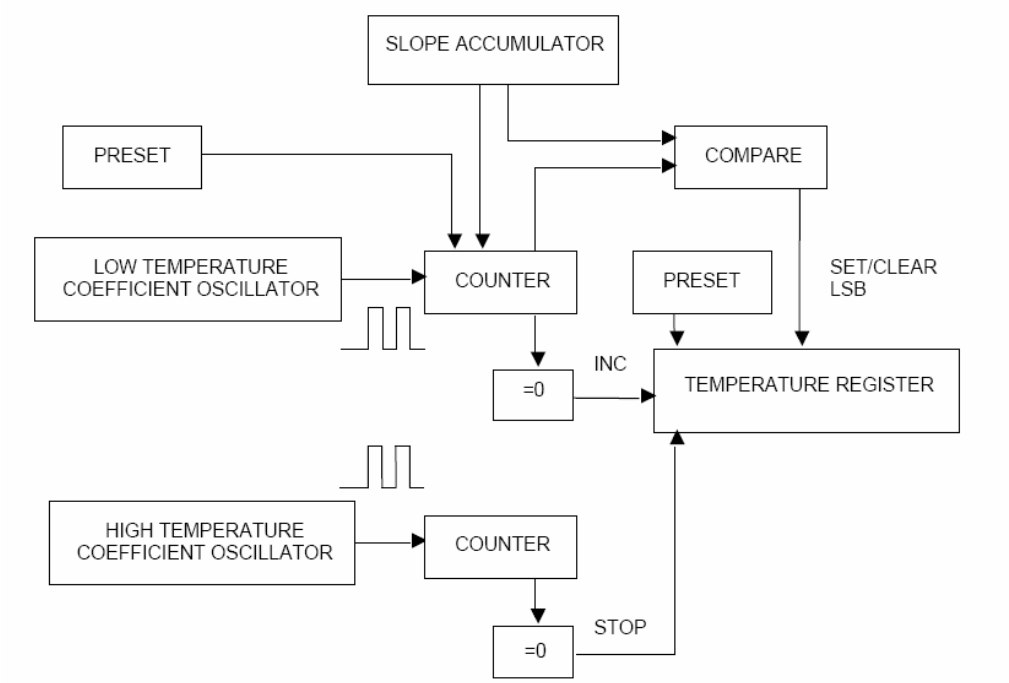

Figure 5: Temperature Reading Functional Diagram

The communication between the microcontroller and the DS1620 is comprised of three signals. To initiate a data transfer the *RST'* pin is driven high, and to terminate the data transfer, the *RST'* pin is driven low. For data to be read by the thermometer, the data must be valid during the rising edge of the clock cycle. The output from the DS1620 is done on the falling edge of the clock and these values must remain valid until the rising edge is complete.

There is a set of command protocols that must be supplied to the DS1620 before any operation is completed. Since the DS1620 will only be used to read temperatures, the Read command is the only one that will be utilized. Upon driving the *RST'* high, the opcode sent to the thermometer is the hex value of $AA. The wave diagram in Figure 6 shows the timing and pin levels of a read temperature operation.

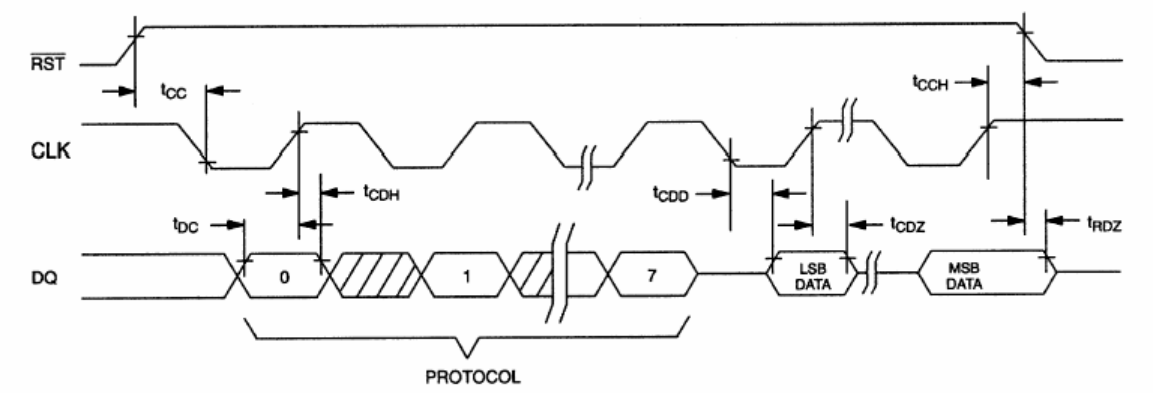

Figure 6: Thermometer Wave Diagram

In order to read the DQ at the right time to pick up the next data bit, a clock with a frequency of 1.75MHz will be supplied to the DS1620 via the CLK/*CONV'* pin. The flow chart for reading the temperature, seen below in Figure 7, describes the operation in more detail. Note, the clock signal is constantly sent out of the RC4 pin on the PIC18F452 microcontroller and is left out of the flowchart.

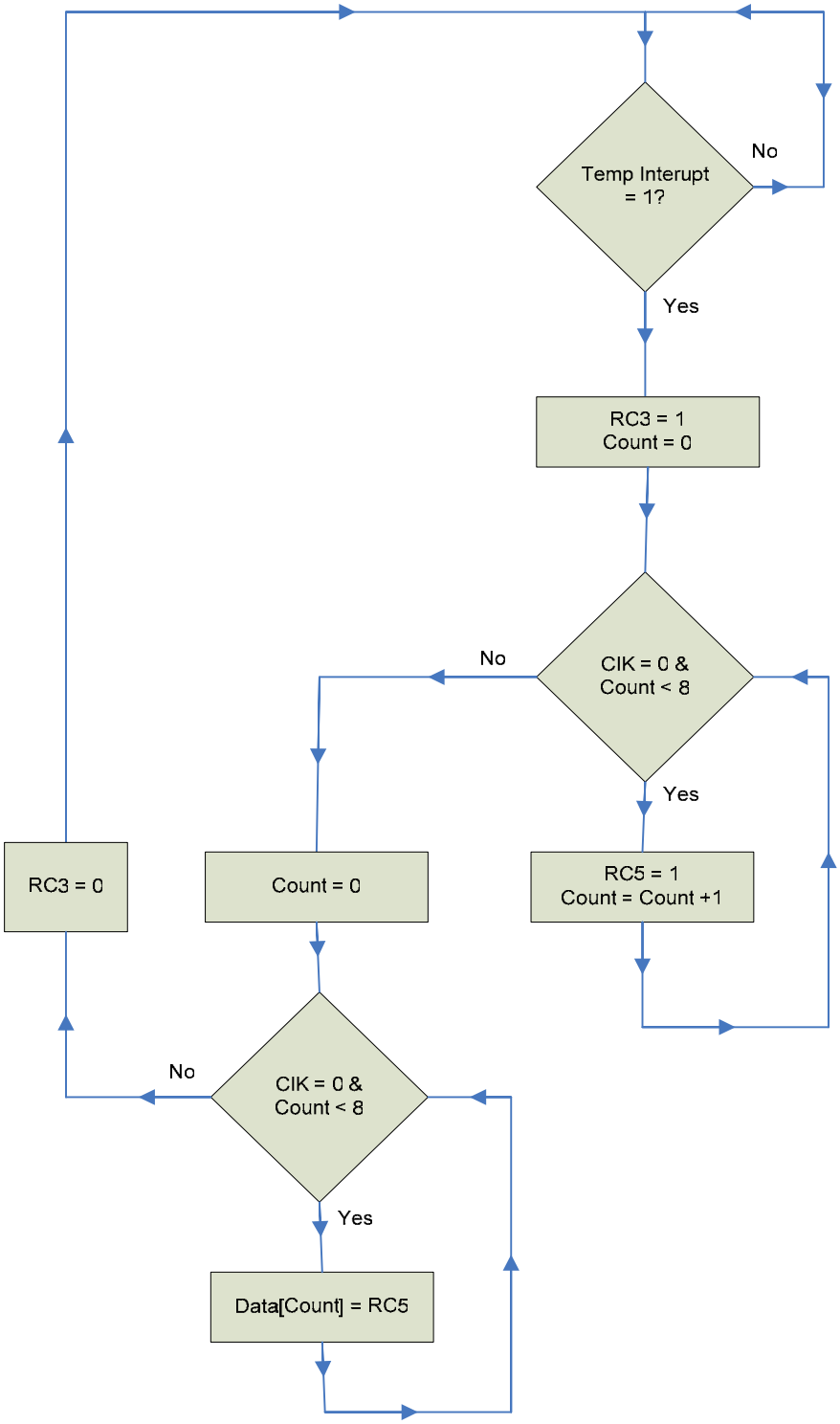

Figure 7: Thermometer Control Loop

3.4 Furnace

The furnace software module will consist of one function, which will take one parameter. The furnace function will take the parameter, high or low, and set the output line that the furnace is connected to high or low.

3.5 Communication

To make the communication subsystem comply with the desired networking protocols, several layers must be developed. Figure 8 shows each of the layers, starting with the physical connection layer (802.11b) and ending with the highest level, the application layer (SOAP).

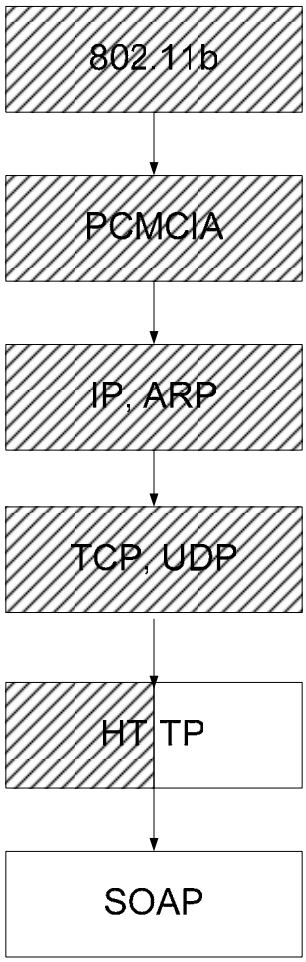

Figure 8: Communication Layers

The shaded layers in the figure above indicate functionality that was implemented as a part of the development kit being used. Each of these layers will be discussed briefly, followed by a detailed explanation of the layers that need to be created.

802.11b: The medium that signals travel on. This wireless standard is commonly used in implementing home networks, so it was chosen to enhance easy adoption of the RATCS. A third party 802.11b wireless card is used in the development kit. [5]

PCMCIA: Personal Computer Memory Card International Association. This interface standard defines the physical layout of pins. Communication between the microcontroller and the wireless card is done using this standard. [7]

IP, ARP: Internet Protocol and Address Resolution Protocol. This layer serves as the network interface. IP is responsible for routing and ensuring proper formation of information packets. ARP maps physical hardware addresses to dynamically assigned IP addresses. [1],[3]

TCP, UDP: Transmission Control Protocol and User Datagram Protocol. Both protocols establish connections between two hosts. However, TCP includes more error resilience checks that make it more reliable in critical data transmission. UDP excludes this overhead, so runs faster.[9],[10]

HTTP: Hypertext Transfer Protocol. HTTP is the application layer that optimizes the exchange of visual information. [2]

SOAP: Simple Object Access Protocol. This is a standard for transmitting XML over HTTP, and allows for functionality to be programmatically accessed from remote locations. [8]

The scope of the RATCS project includes writing only to the HTTP and SOAP layers.

3.5.1 HTTP Requests

A request coming into the microcontroller will be identified as an HTTP request, and at this point must be handled. Figure 9 illustrates the control loop that handles all HTTP requests. Several HTTP pages will be developed to interface to the functionality of the device. Further explanation is provided in Section 5.2.

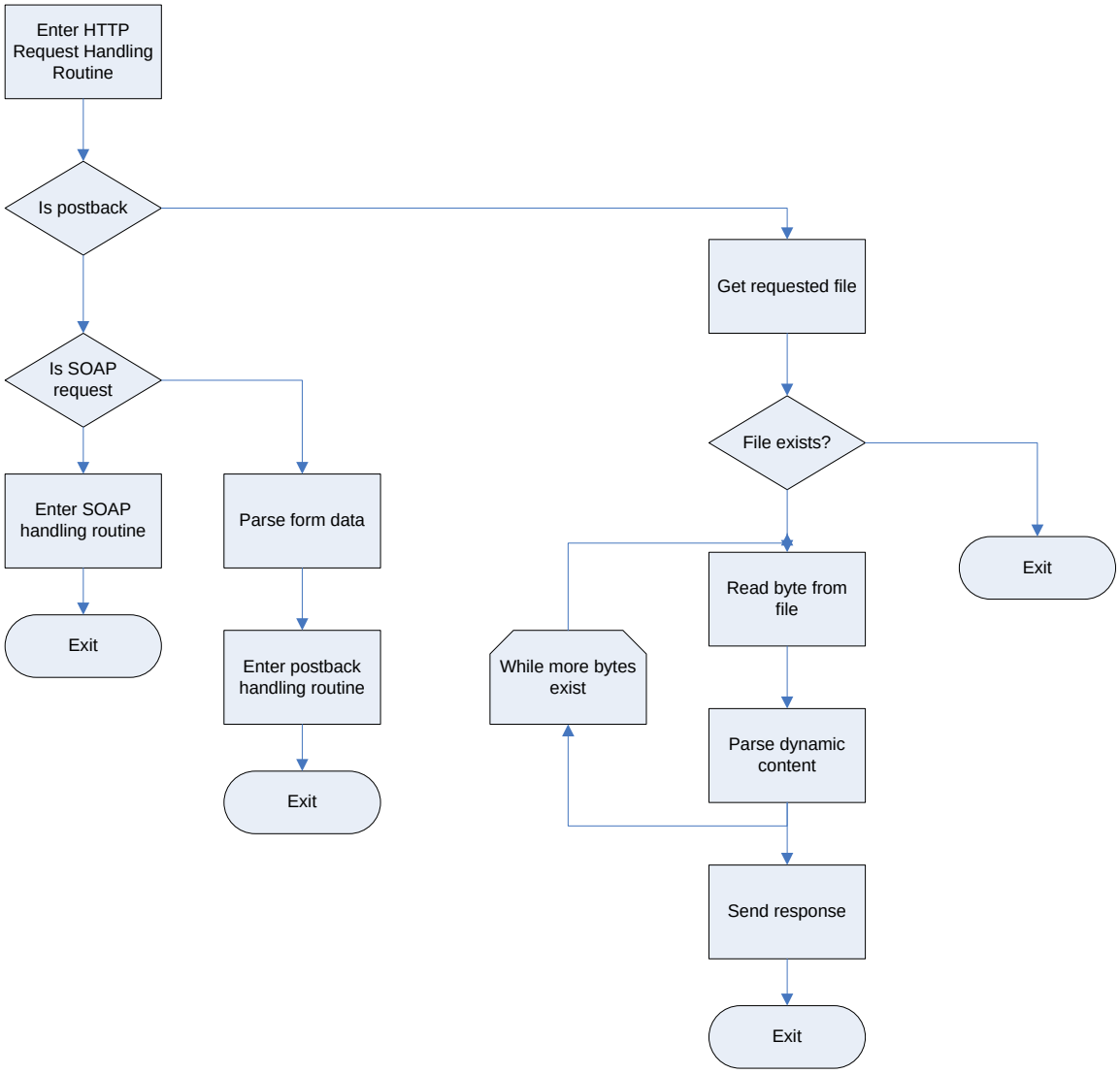

Figure 9: HTTP Request Handling

3.5.2 SOAP Requests

The HTTP request handler may recognize the request as a SOAP event, and will pass control to the SOAP handler immediately. The SOAP handler will be required to parse out the XML encoded in the request, and act appropriately. This flow is show in Figure 10.

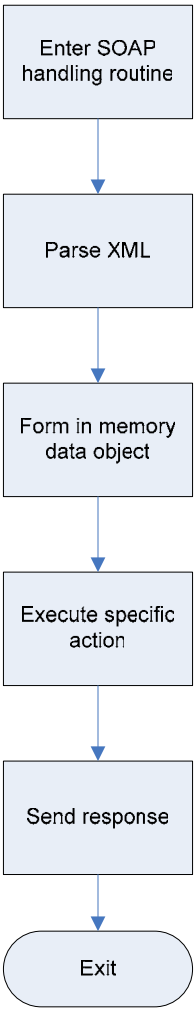

Figure 10: SOAP Request Handling

A separate SOAP event handler will be required for each of the functions that will be exposed through this method. Table 2 lists the functions that will be available through the SOAP interface.

Table 2: Supported SOAP Methods

Method Signature	Method Description
Temperature GetTemperature()	Queries the RATCS for the current temperature.
SetTemperature(Temperature)	Sets the current temperature on the RATCS.
Schedule GetSchedule()	Gets the current schedule.
SetSchedule(Schedule)	Sets the current schedule.

The data types referred to by the table above are defined below in Figure 11, Figure 12, and Figure 13.

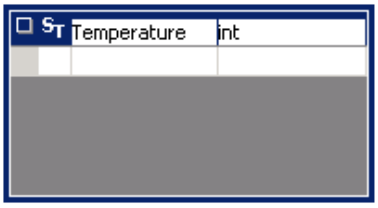

Figure 11: Temperature Data Type

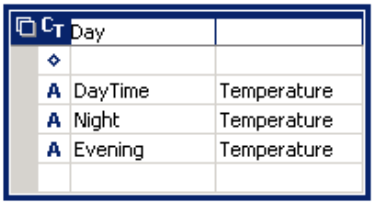

Figure 12: Day Data Type

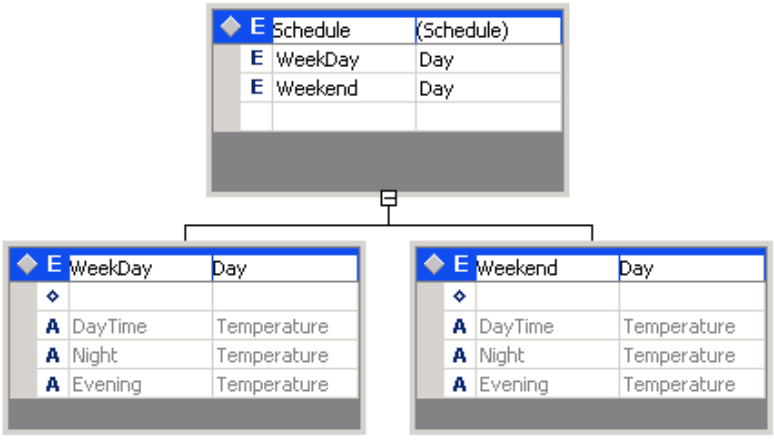

Figure 13: Schedule Data Type

3.6 Engine

The engine software acts as the control loop for the entire system; it is the piece of software that ties all of the component modules together on the microcontroller. The engine software will be written in the form of an interrupt driven control loop. Upon startup of the system, the engine will run through its initialization routine, which involves initializing each component module, fetching the current temperature, displaying the current temperature on the LCD, and initializing a timer that reads the current temperature every ten seconds. The engine will then enter a loop in which it waits for interrupts. Once the engine receives an interrupt it responds accordingly via a handler and then waits for the next interrupt. The flowchart for the engine software is shown below in Figure 14. There are three types of interrupt in the system: timer interrupts, button interrupts, and communications interrupts.

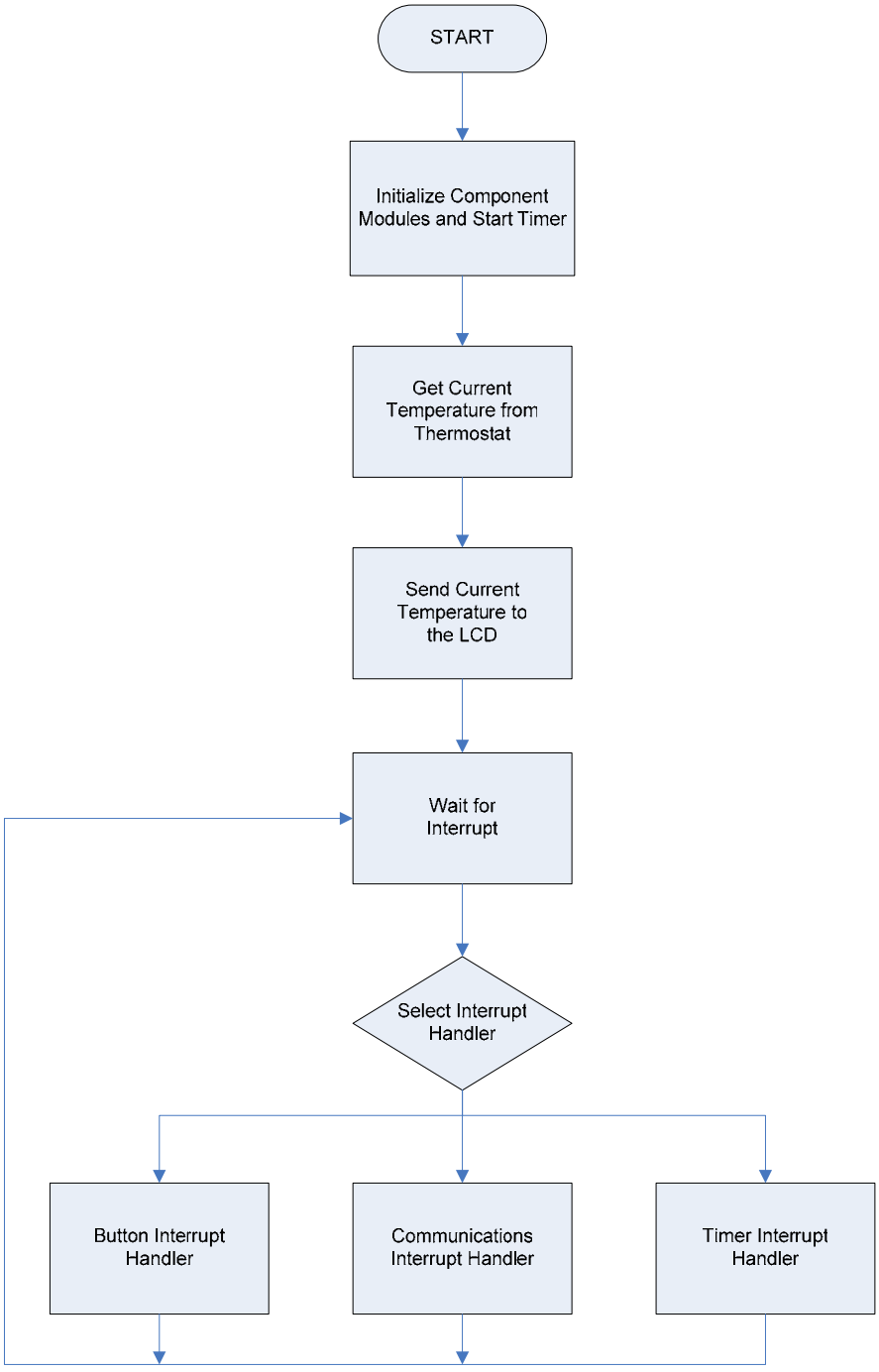

Figure 14: Engine Control Loop

Timer interrupts occur every ten seconds and involve the reading of one temperature value. The timer interrupt handler will look at previous temperature readings and make a decision, based on the current desired temperature, on whether to turn the furnace off or

on. The handler will then reset the timer and return to the engine control loop, which waits for the next interrupt.

Button interrupts occur when a button is pressed, and depending on which button is pressed the button interrupt handler responds accordingly. The button pressed may have been to set the current temperature up or down or to move to a time period in programmable mode. The button interrupt handler makes its decision on what to do and returns to the main loop.

Communications interrupts occur when a request or command has arrived from the remote interface. The communications interrupt handler must determine what request or command has arrived and then respond accordingly. The communication may have been a request, which simply involves sending out the desired information. However, if the communication was a command, the handler will then perform some action similar to that of the button interrupt handler and then return a response back to the remote interface.

4 Hardware Design

4.1 Hardware Block Diagram

The hardware block diagram, shown below in Figure 15, details the connectivity between each hardware component.

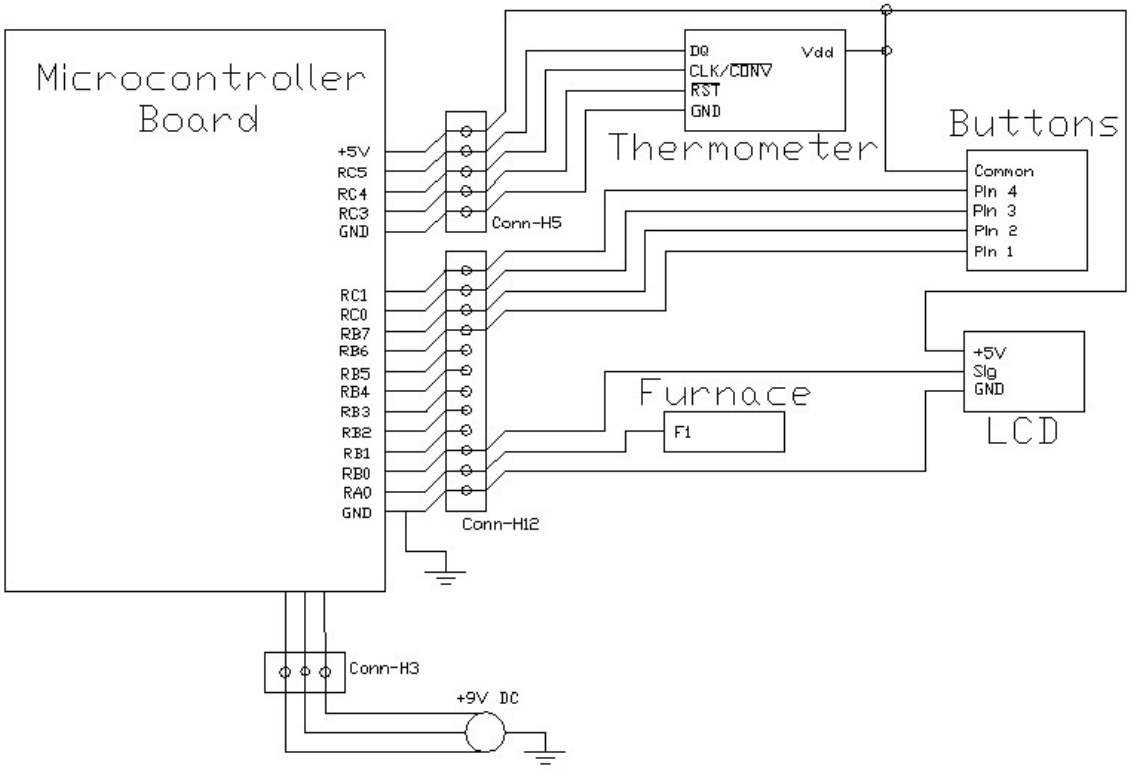

Figure 15: Hardware Block Diagram

4.2 Power

The Iosoft development board, which supplies power to external components, requires a 9V DC power supply to operate. A 9 Volt DC power supply, outputting at 900mA, that plugs into a simple wall socket will be used. A picture of the power supply is shown below in Figure 16.

Figure 16: Power Supply

4.3 Buttons

The manual input for the temperature control system is done via the buttons on the wall mounted panel, which also includes an LCD screen. The Storm 3000 Series 4-key keypad from Storm Interface provides the required number of input buttons, as specified previously, for the designed functionality. The pin detail is shown in Figure 17.

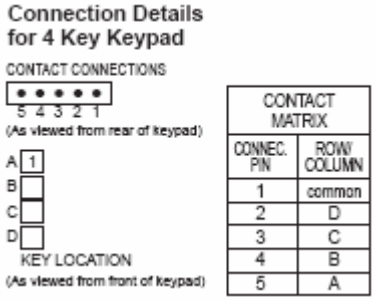

Figure 17: Button Pin Details

As shown in Figure 18, the buttons rely on a conductive element being pressed onto a gold and nickel plated, copper tracked, Printed Circuit Board. This particular keypad technology was chosen because an advanced manufacturing process is used to ensure that the conductive tracks on the PCB does not present sharp edges to the conductive element, effectively extending the operational life and reliability of the switches. Also, its keys are a molded silicone rubber material that provides a responsive 'over-centre' feel, as the key is pressed. The design of the switch actuator ensures that a reliable contact is made even if the top of the key is struck inaccurately or 'off centre'.

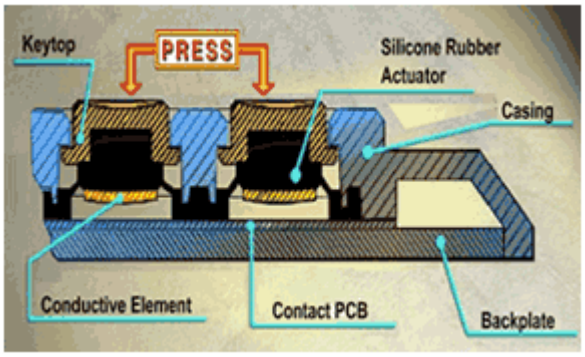

Figure 18: Button Design

During market research, a strong, sturdy feel and reliable operation were forefront when discussed with potential users.

The button module requires 5 Volts of power taken from the board, as well as a pull-up resistor for each button. The circuit diagram for an individual button is shown in Figure 19.

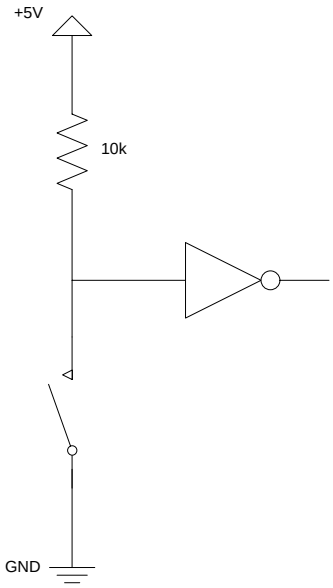

Figure 19: Individual Button Pull-up Circuit

4.4 LCD

The Econo-LCD Serial kit interfaces a generic 2x16 character display LCD to the PIC18F452 microcontroller with 16-Pin SIP Header. Its simple 2-wire TTL Serial interface and +5V supply voltage allow for convenient interfacing to the board. Its onboard character generator and RAM make writing to the LCD simple and quick. Figure 20 below, shows the circuitry for the LCD serial kit:

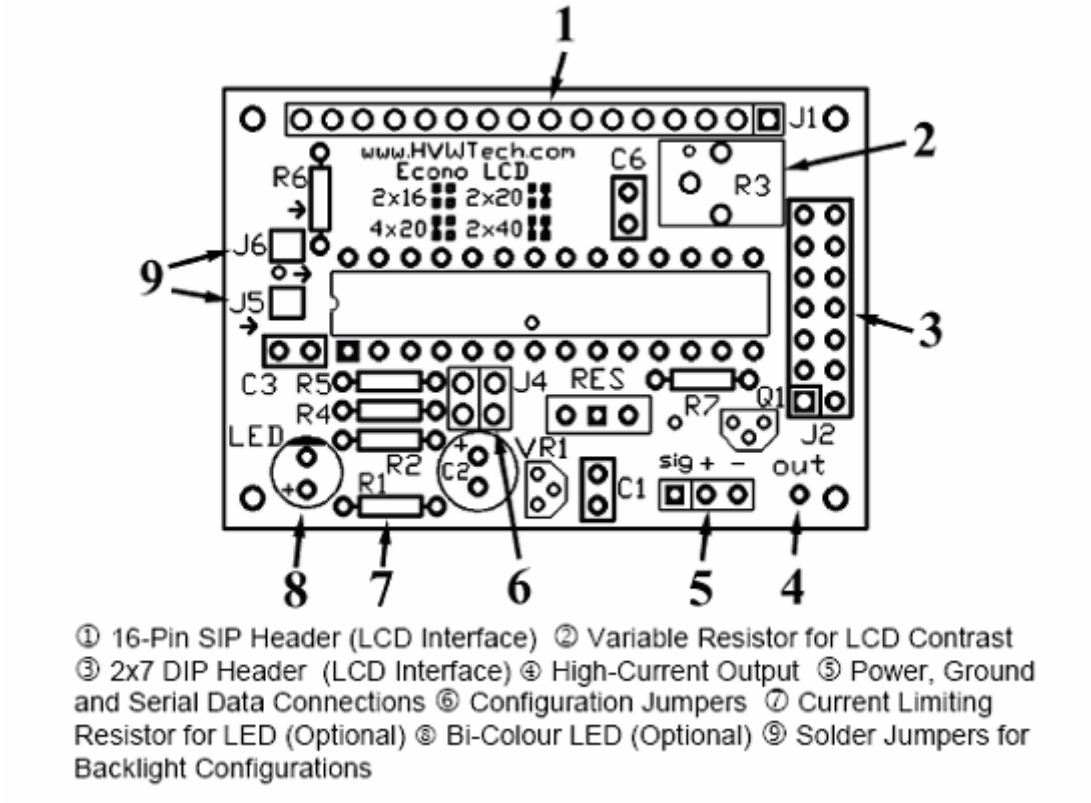

Figure 20: LCD Circuit Diagram

With reference to the optional LED, (#8 in the figure), a green/red bi-colour LED was implemented for debugging purposes and provides a visual indicator of proper operation.

The pins connections between the PIC18F452 microcontroller and the LCD interface can be seen in the overall circuit diagram but are also provided below, in Table 3.

Table 3: Pin Connections between the Microcontroller and the LCD

PIC18F452 microcontroller	LCD Serial Interface Pin
+5V	+5V
RB0	Sig
Gnd	Gnd

4.5 Thermometer

The DS1620 Digital Thermometer by Dallas Semiconductor has the ability to detect the temperature in its environment with a precision of $0.5°C$ between the $-55°C$ to $+125°C$ temperature range, every second. The 3-wire serial interface gives the PIC18F452

microcontroller quick and easy communication with the thermometer. Figure 21 below, is the figure of the pin arrangement on the device.

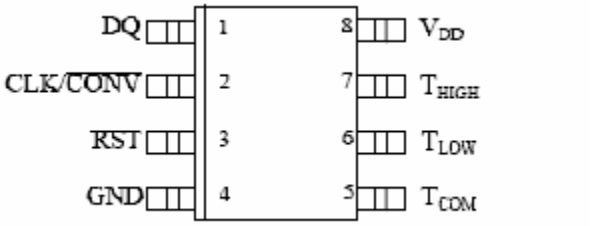

Figure 21: Thermometer Pin Arrangement

The pin descriptions of the digital thermometer are listed in Table 4.

Table 4: Thermometer Pin Descriptions

Pin	Description
DQ	3-Wire Input/Output
CLK/*CONV'*	3-Wire Input/Output & Stand-alone Convert Input
RST'	3-Wire Reset Input
GND	Ground
T_{HIGH}	High Temperature Trigger
T_{LOW}	Low Temperature Trigger
T_{COM}	High/Low Combination Trigger
V_{DD}	Power Supply Voltage (3V-5V)

The T_{HIGH}, T_{LOW} and T_{COM} pins would be utilized if a furnace was desired to be controlled via this thermometer. However, the software on the PIC18F452 microcontroller will decide this and control the furnace operation based on the output supplied by the thermometer via the DQ pin.

The pin connections between the PIC18F452 microcontroller and the DS1620 can be seen in the overall circuit diagram but are also listed below, in Table 5, for quick referencing.

Table 5: Pin Connections between the Microcontroller and the Thermometer

PIC18F452 microcontroller Pin	DS1620 Pin
+5V	V_{DD}
RC5	DQ
RC4	CLK/*CONV'*
RC3	*RST'*
Gnd	GND

4.6 Furnace

The furnace that will be used for prototyping the RATCS system is a common home gas-powered furnace. Figure 22 shows the circuitry contained in the furnace. It is controlled using the input line. An output pin will from the microcontroller will be used to control the signal line.

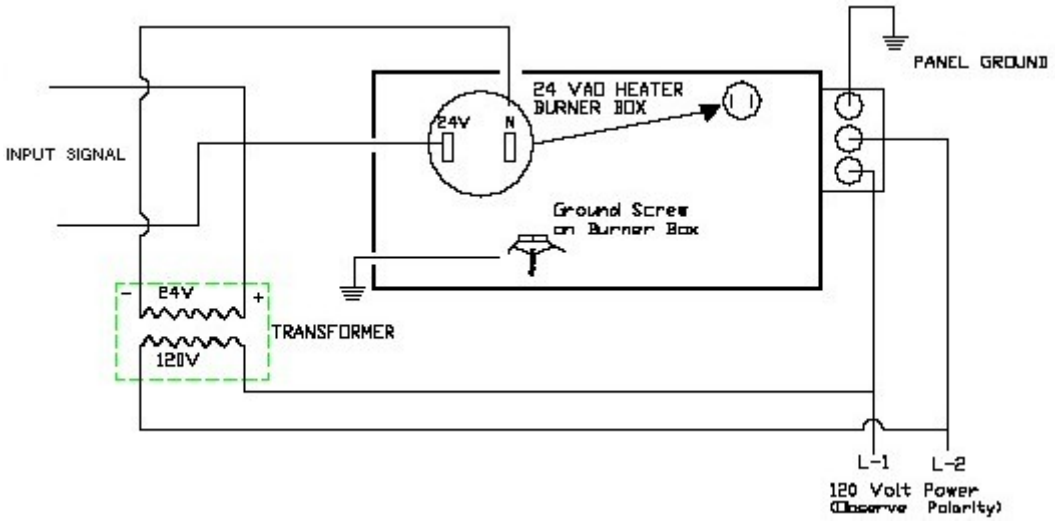

Figure 22: Furnace Circuit Diagram

4.7 Development Kit

The RATCS requires a development board with IEEE 802.11b wireless Ethernet capability that also has substantial computing power. One such board is offered by Iosoft and is called the ER22 [4]. The ER22 comes with a PCMCIA interface with an IEEE 802.11b wireless card included. The ER22 also comes with 32Kbyte storage space for web pages, which is required because the RATCS will act as a web server. The ER22 also has eleven I/O lines, which is enough to interface to each of the LCD, buttons, thermometer and furnace required by the RATCS. A picture of the ER22 is show below in Figure 23.

Figure 23: Iosoft ER22 Development Board

Computing power on the ER22 is provided by a PIC18F452 microcontroller with 32Kbyte of programmable memory and running at 40 MHz [6]. The PIC18F452 has 75 instructions in its instruction set, is programmable in C, and will be able to handle the programming requirements of the RATCS. The pin details of the PIC18F452 are shown below in Figure 24.

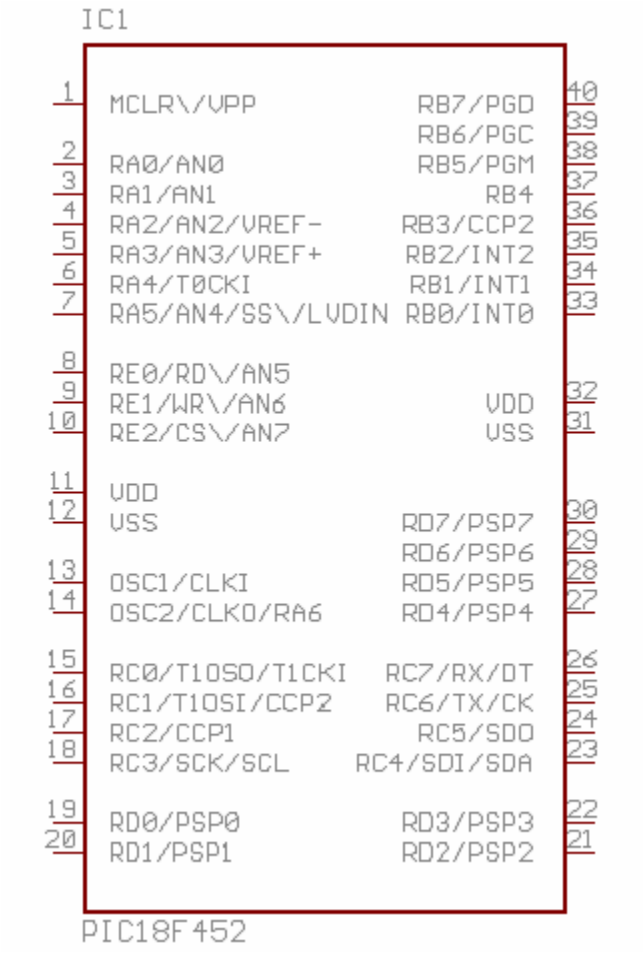

Figure 24: PIC Microcontroller Pin Details

5 Interface Design

5.1 Local

The physical user interface of the Remotely Accessible Temperature Control System will be embodied by an LCD and by four buttons on a panel. An LCD was chosen as the method of delivering visual feedback at the site the device is situated in the home because the hardware portion of the RATCS is intended to be a replacement for a common wall mounted thermostat. Wall mounted thermostats require visual feedback to the user to show what the current temperature is and what the desired temperature is set to. Since the functionality of the RATCS requires that the user be able to program temperature settings for different days of the week and different times of the day, a simple temperature gauge and slider would not suffice. Therefore, the LCD will show the current temperature on the right portion of the screen and the set temperature on the left portion of the screen in normal operating mode and will show days, times, and the desired temperature for that period in programmable mode. This information can be displayed on a screen that is sixteen characters wide and two lines tall, as per the functional specification.

Setting of temperatures in normal and programmable modes and changing modes will be done through the pressing of buttons. The minimum number of buttons required to implement the intended functionality is four; therefore, there will be four buttons. The buttons will be attached to a panel and each button will have an arrow on it pointing in one of the directions up, down, right, or left. The buttons pointing up and down will set the desired temperature value up and down by one degree centigrade per press in both normal and programmable modes. The range of temperatures allowed will be between ten and thirty degrees centigrade, which is standard for most in home thermostats. The buttons pointing right and left will be used for cycling through the set programmable time periods. The time periods have not been exactly determined, but will be in the realm of one time period each for daytime hours, evening hours, and night hours for weekdays and for weekends. In programmable mode, the LCD will show the current desired temperature to be set for this time period and the time period itself. Scrolling through every time period, either right or left, will return the user to the normal mode of operation in which the current temperature is set. The temperature set in normal mode will override the programmed settings for that time period. However, when the next time period arrives, the temperature will be set to the programmed value for that time period. The components of the physical interface are shown below in Figure 25 and Figure 26.

Figure 25: LCD Display

Figure 26: Buttons

5.2 Remote

The software interface of the RATCS will mirror the physical interface, but will be available over the worldwide web via HTTP. The user will open up their Internet browser, for example Internet Explorer, and will enter the physical address of their RATCS. The user will then be prompted with some form of authentication, which is to be determined, to ensure the security of the RATCS. After authentication, the user will then see a screen that will closely resemble the interface presented by the LCD and buttons. The current and desired temperatures will be shown, as well as buttons for changing the current temperature in normal and programmable modes of operation and

buttons for moving into the programmable mode of operation. The web interface will likely be improved over the physical interface presented by the physical interface. The improvements to the web interface will maintain the same functionality while making the interface more use friendly. The improvements in usability to the web interface have yet to be determined.

6 Test Plan

This section detail the objectives and the procedures required for testing that the design of the RATCS will meet the specified functional requirements. RATCS testing will begin with testing of the individual components and their software modules. Once each component has been fully tested, with all encountered problems resolved, the engine software will undergo integration testing before the entire system is tested. The guideline throughout testing will be the previously laid out functional specifications for each component and for the system as a whole. The functional specifications will act as a checklist during system testing to determine whether the device is fully operational or if more refinement is required.

6.1 Test Overview

This section detail the objectives and the procedures required for testing that the design of the RATCS will meet the specified functional requirements. RATCS testing will begin with testing of the individual components and their software modules. Once each component has been fully tested, with all encountered problems resolved, the engine software will undergo integration testing before the entire system is tested. The guideline throughout testing will be the previously laid out functional specifications for each component and for the system as a whole. The functional specifications will act as a checklist during system testing to determine whether the device is fully operational or if more refinement is required.

6.2 Testing Strategy

The testing of the RATCS will be conducted in four stages, two of which reflect the remote and onsite functionality of the system. First, the individual components of the system must be tested, which includes the digital thermometer, the LCD, the button panel, and the Iosoft board. However, to test the individual components of the design, the Iosoft board must first be working. Therefore, the first stage of testing will involve component testing and system integration at the same time. Once code written for the PIC microcontroller can be compiled and run on the Iosoft board, other components can begin to be tested as well. The LCD will be tested first by sending commands from the board to write characters to the LCD. The thermometer can then be tested through the reading temperature values and writing them to the LCD. The buttons can be tested at this point as well by writing different characters to the LCD when different buttons are pressed. In this way it can be assumes that each component interfaces with the board correctly and integration involving the correct functionality can continue.
The second stage of testing, following system integration, will involve testing the system to prove that it meets the functional specifications to act as a normal thermostat onsite. This will be done by using the complete system minus only the remote interface and running through the basic operations of the thermostat. Testing will be conducted by setting the current temperature up and down and by setting some of the programmable time periods. Functionality will be observed by monitoring when the board sends signals

to turn the furnace on via a simple LED. When testing proves that the RATCS meets the functional requirements of an in home thermostat, stage three will begin.

The third stage of testing will involve the integration of the remote interface and remote testing of the device. Proving functionality in this stage will be done in the same way as done in stage two. However, in this stage the system will be responding to input from the remote interface and not through physical button input. Functionality will be proven in this stage again by monitoring when the system would turn on the furnace via a simple LED. After the remote interface has been tested, complete system integration is considered complete.

The fourth stage on testing will be conducted to prove the robustness of the RATCS. Tests will be conducted to see how the system holds up over time with extensive user input and frequent changes in temperature. Statistics such as power consumption and performance in harsh environments will be looked as well. Once each functional specification has been tested and achieved, testing will be complete.

7 Conclusion

Design choices for the Remotely Accessible Temperature Control System have been documented and detailed within this design specification. Application of the outlined software, hardware, and interface designs are currently being implemented for each of the system's modules with test procedures occurring concurrently. Following these detailed specifications and standards, Holla Home Solutions is confident that a fully functional prototype of the Remotely Accessible Temperature Control System will be complete by the prescribed completion date, at beginning of April, 2005.

References

[1] Address Resolution Protocol
http://www.faqs.org/rfcs/rfc826.html

[2] Hypertext Transfer Protocol
http://www.w3.org/Protocols/rfc2616/rfc2616.html

[3] Internet Protocol
http://www.faqs.org/rfcs/rfc791.html

[4] Iosoft Ltd.
http://www.iosoft.co.uk

[5] Institute of Electrical and Electronics Engineers
http://www.ieee.com

[6] Microchip Technology Inc.
http://www.microchip.com

[7] Personal Computer Memory Card International Association
http://www.pcmcia.org/pccard.htm

[8] Simple Object Access Protocol
http://www.w3.org/TR/soap/

[9] Transmission Control Protocol
http://www.faqs.org/rfcs/rfc793.html

[10] User Datagram Protocol
http://www.faqs.org/rfcs/rfc768.html

[11] World Wide Web Consortium
http://www.w3c.org