2006-1802: ETHERNET ENABLED WI-FI MAIL-BOX

Jai Agrawal, Purdue University-Calumet

JAI AGRAWAL is a Professor in Electrical and Computer Engineering Technology. He received his Ph.D. in Electrical Engineering from University of Illinois, Chicago, in 1991, M.S. and B.S. also in Electrical Engineering from I.I.T. Kanpur, India in 1970 and 1968 respectively. Professor Agrawal has worked recently for two years in optical networking industry in the Silicon Valley in California. Professor Agrawal is the Founder Advisor to Agni Networks Inc., San Jose, California. His expertise includes optical networking at Physical and Data link layers, optical and WDM interface, SONET and Gigabit Ethernet and analog electronic systems. He is the author of a Textbook in Power Electronics, published by Prentice-Hall. His professional career is equally divided in academia and industry. He has authored several research papers in IEEE journals and conferences. Currently Professor Agrawal is doing research in nanophotonics.

Omer Farook, Purdue University-Calumet

OMER FAROOK is a member of the faculty of the Electrical and Computer Engineering Technology Department at Purdue University Calumet. Professor Farook received the Diploma of Licentiate in Mechanical Engineering and BSME in 1970and 1972 respectively. He further received BSEE and MSEE in 1978 and 1983 respectively from Illinois Institute of Technology. Professor Farook's current interests are in the areas of Embedded System Design, Hardware–Software Interfacing, Digital Communication, Networking, Distributed Systems, C++ and Java Languages. He is a member of ASEE and senior member of IEEE.

Chandra Sekhar, Purdue University-Calumet

CHANDRA R. SEKHAR is a member of the faculty of the Electrical and Computer Engineering Technology at Purdue University Calumet. Professor Sekhar earned a Bachelor's Degree in Chemistry from the University of Madras (India), a Diploma in Instrumentation from Madras Institute of Technology and Master's Degree in Electrical Engineering from University of Pennsylvania. Professor Sekhar's primary teaching and research focus is in the areas of Biomedical and Process Control Instrumentation and Clinical Engineering.

Michael Segura, Purdue University-Calumet

MICHAEL I. SEGURA is an alumnus of the Electrical and Computer Engineering Technology Department at Purdue University Calumet. Michael earned his Bachelor's Degree in ECET in 2005 with distinction. Michael's Ethernet enabled Wi-Fi mailbox project was selected by the ECET faculty as one of the best senior design projects for the Spring 2005 semester. He is currently employed with ETS, Inc., in Indianapolis, Indiana, as a Product Engineer in the engineering department. His duties include designing new and improved, listed or recognized tanning beds which compete in the global market.

ETHERNET ENABLED Wi-Fi MAIL-BOX

Abstract

Come winter and people in homes spread over acres in the countryside, dread the idea of walking on snow-covered driveways in icy conditions up to the mailbox from their cozy confinements and find that no mail arrived. This paper presents a project that will enable the home-resident to monitor the mail arrival while sitting in his or her cozy home. The laser beam interruption due to the mail droppings in the mailbox is relayed as a serial data stream comprised of Ethernet packets using the Wi-Fi communication link of the Wi-port controlled by the microcontroller from outside the home. The mailbox also monitors outside temperature and relays on the Wi-Fi link. On the controller side, the received data is fed into a GUI applet that the client interfaces and interacts with. This information is made available on the Internet using an embedded Web server and can be viewed from anywhere in the world and at any time. The mailbox is a rugged and environment –savvy design and is ac-dc powered with battery back-up. The Wi-Fi mailbox is just one application. The design can be utilized in many innovative applications.

I. Introduction

One of the authors got interested in the problem of mailboxes in the countryside being placed far away from the residence, either on the far end of a big lot or on the curbside, and the residents having to walk a long distance in cold weather only to find that no mail arrived, a burdensome chore. Commercially available was a wireless system that consisted of a transmitter and receiver. The receiver, in the homeowner's home, would beep to let them know that they had mail in their mailbox. This system did not have the capability to connect to the internet or any LAN so one could check for mail at work, or school. There was clearly the need for such a smart mailbox that could be monitored and controlled via the internet from anywhere in the world at any point in time. The idea of checking snail mail over the internet was exciting to many people.

II. System Description

The Wi-Fi mailbox is fabricated as a 6-inch extruded aluminum box for faraday shielding, with GFI outlets, an Ethernet RJ45 coupler, UTP CAT 5e Ethernet cable, Lightning arrestors and 2.4GHz -7dBi omni directional antenna with 2ft of LMR195 cable. The Wi-Fi mail box system consists of six subsystems: 1) Mailbox, 2) Sensors, 3) Development board, 4) Communication, 5) Interface and 6) Power supply.

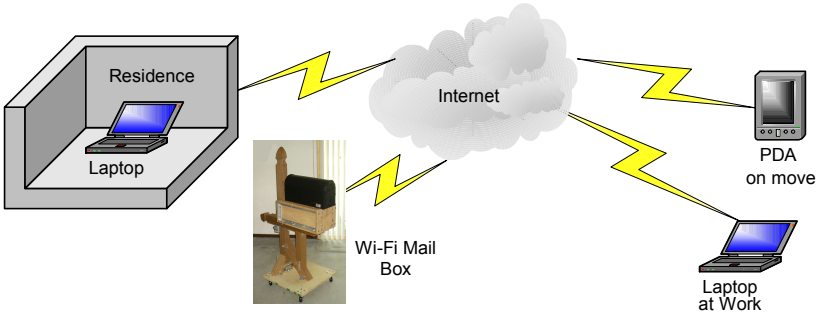


Fig. 1 Wi-Fi Mail Box

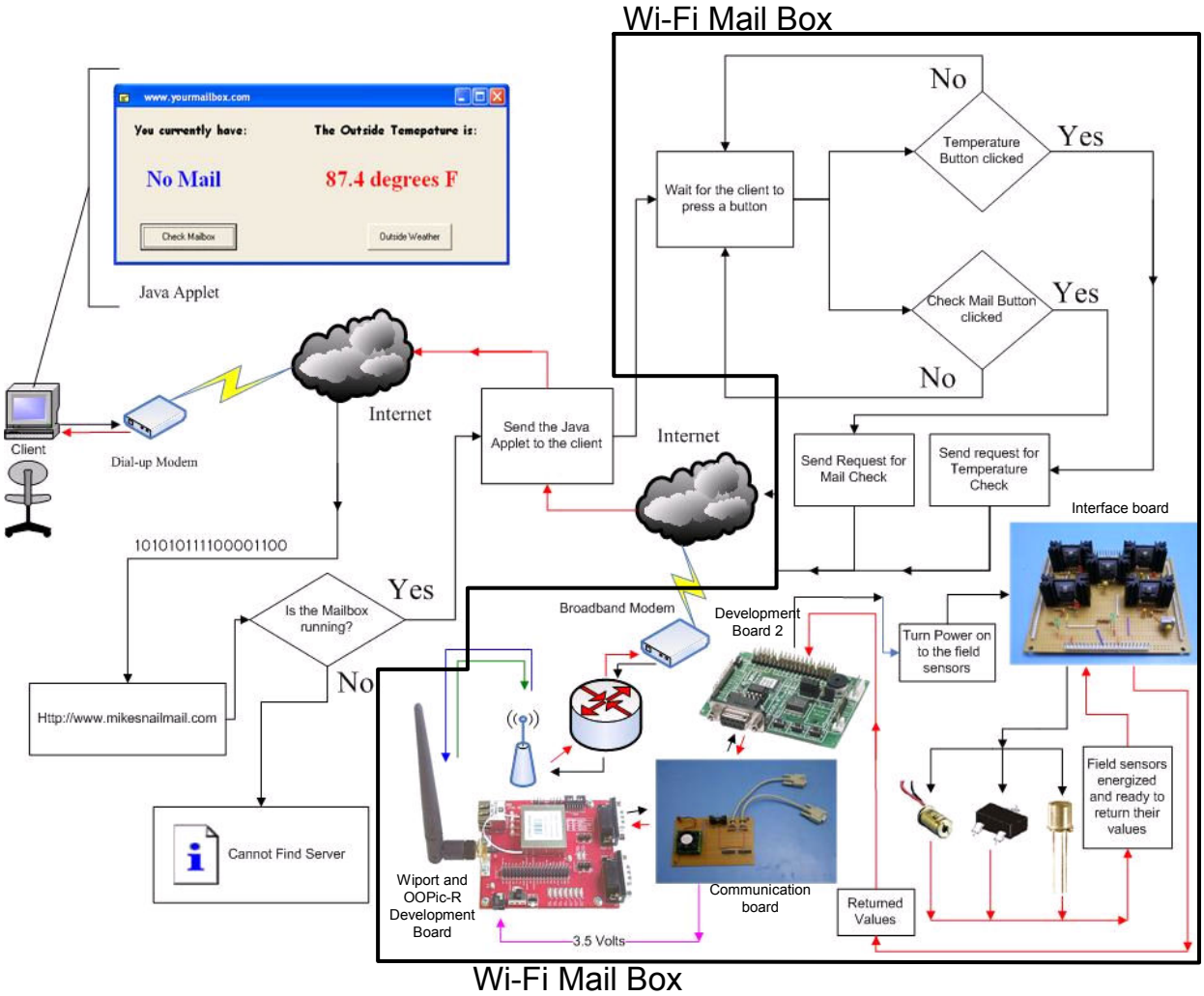
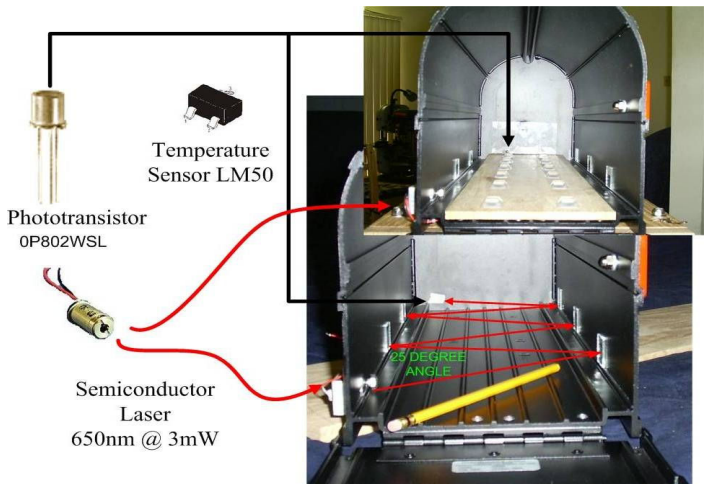


Fig. 2 Detailed block diagram of the Wi-Fi mail box

The development board in Fig. 2 consists of two printed circuit boards a) Wiport development board and b) OOPic-R development board. The Wiport Developmental board (Lantronix) has the ability of 802.11b wireless networking to any edge device with a serial interface, supports WLAN or Ethernet connectivity, has embedded web server, real time operating system, with 2MB of flash memory, a complete TCP/IP protocol stack, with WEP security, two high speed

com ports and capable of Java applets and HTML. The Wiport serves up the Java applet to the client, while at the same time converting, serial data to Ethernet packets and vice versa from the resulting interaction, of the applet. It also transmits data wirelessly or wired to the wireless, router or access point to and from the client and embedded web server, thus achieving remote access.

The OOPic-R Development board (Fig 2) contains a PICmicro operating system to use an Object-Oriented approach to hardware control, one serial port, 16 Digital I/O lines, I2C network, four 10-bit Analog to Digital converter channels, programmed in Basic, C, or Java syntax styles, 8KB of program space and 130 data objects. To convert the field sensor values to serial data and send it off to the Wiport for further data transport. The use of three I/O lines will be used for A/D conversion of the temperature sensor's analog data and for turning on power to it. The third I/O line will be used as a reference voltage for the A/D converter. Another set of I/O lines will be used for turning on the power circuit to the laser circuit and reading the logic levels present at the phototransistor.



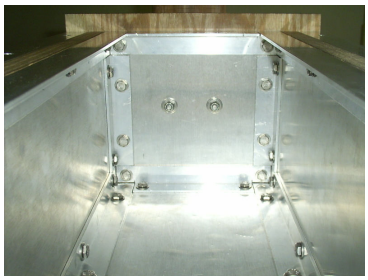


Fig. 3 Inside the mail box
Fig. 4 Lightning Arrestor and Surge Suppressors
Fig. 5 The Faraday Enclosure

The field sensors, shown inside the mail box as shown in Fig. 3, consists of a phototransistor, a class IIIa 650nm $\geq$ 3mW semiconductor laser module, and a LM 50 National Semiconductor temperature sensor. The LM 50 sensor, 0P802WSL 5mW/m^2 phototransistor, Laser module with automatic power control, plastic lens with anti-reflective coating, beam has an adjustable focus, runs off 3V, draws about 45mA, and has a divergence of < 2.0 mrd. LM50 semiconductor temperature sensor operates off of 4-10VDC, measures temperature between the ranges of - 40°C to +150 °C at 10mV/°C. 0P802WSL 5mW/m^2 phototransistor: has an E_e = 5mW/m^2, a sensitivity spectral bandwidth starting at 310nm to 1030nm in wavelength; 850nm being the highest sensitivity peek.

Communication board in Fig. 2 houses the connectors and chips needed for communication between the OOPic-R board and Wiport Dev Board. It also serves as a power source for the

Wiport Dev. Board. It sources up to 2.2 amps max at 3.5V with a massive heat sink. We added a fan to protect the board from thermal runaway. This board adds industry standard RS-232C voltage levels for effective serial protocol. It would allow for data communication to take place between the two developmental boards and will specifically supply power to the Wiport Dev. Board.

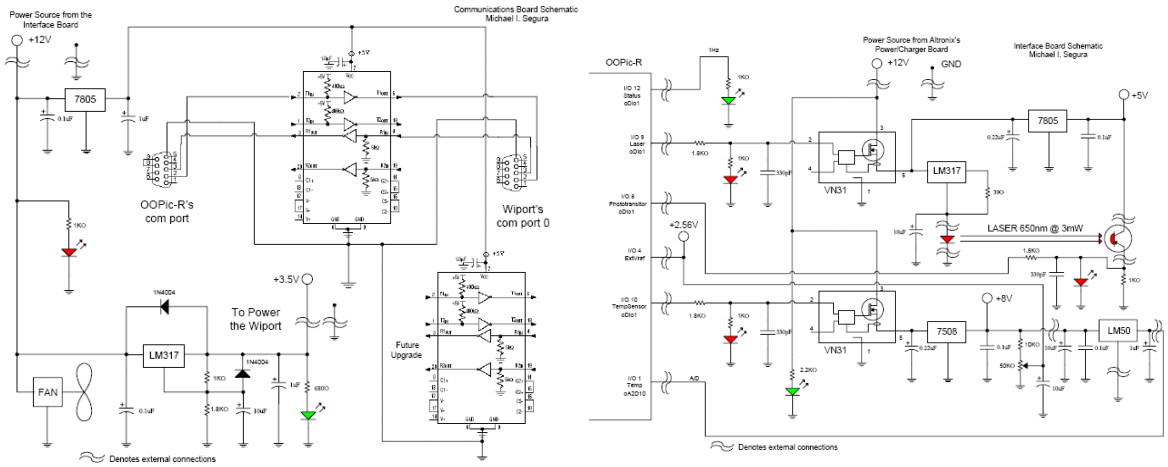


Fig. 6 Circuit for the laser source and the interface boards

The Interface board in Fig. 2 houses the connectors and chips needed to buffer to the OOPic-R board to the sensors. It also isolates the OOPic-R board from the outside world of which is where the sensors reside. The key component in achieving this isolation is the VN31 solid state relay. It takes TTL logic levels from the OOPic-R board and switches heavy loads at a maximum of 11.5 amps at 60V very fast. This board also uses separate voltage regulators for each required voltage level for each sensor, thus maintaining rock solid performance. It also includes the reference voltage for the OOPic-R board's A/D conversion circuit. It would allow for an easier flow of data control that is housed on a central board. It also allows for an easier installation as opposed to multiple connection points within the confined enclosure.

The power supply/Charger (Altronix)/ converts 28VAC / 100VA into a power limited 12VDC or 24VDC all from a 120VAC source. 2.5 amps of continuous supply current Filtered and electronically regulated outputs with a built in charger for sealed lead acid of gel type batteries Automatic switch over to stand-by battery when AC fails thermal overload protection short circuit protection.

When lightning strikes during a storm, it can cause serious damage to electrical equipment. There are three ways in which the Wi-Fi mailbox is subject to the negative effects of a lightning strike, or power surges. They are its 120VAC power, Ethernet port, and the attached antenna. These three liabilities are protected by devices called lightning arrestors, surge suppressors, and a GFI outlet. The GFI outlet is modified for sourcing of power from the front of the outlet, as opposed to the terminals on the back of the GFI outlet. So basically, the load is now only in the back terminals and the line is now in the front. Figures 4 and Fig. 5 show the lightning arrestor and the faraday enclosures respectively, used for protecting the sensitive electronic devices

within the enclosure from electromagnetic radiation produced by other devices and lightning. Figure 6 shows the circuits for the laser source and the interface boards.

III. Hardware Operation

As the Fig. 2 illustrates, the client can be either at home, school, work, or wherever there is a connection to the internet. Client will proceed to open any Java enabled internet browser to start the communication process to the Wi-Fi mailbox. Upon running the browser, the client will type in the URL address http://mikesnailmail.gotdns.com/test2.html for the address to the Wi-fi mailbox. At the moment the client hits the return key, the client will establish a TCP socket with the Wiport (Fig. 2) via his/her modem over the internet. A DNS service from www.dyndns.org will relate the dynamic IP address to a name like the one in Fig. 2. This process is known as DNS. In this case, its http://mikesnailmail.gotdns.com/test2.html and the IP address is the WAN IP address given to the wireless router, shown in the middle of Fig. 2, from the internet service provider (ISP). In the event the mailbox is not online, the client will receive the "Cannot Find Server" error 404 page while trying to establish the TCP socket. If the Wiport is online, the Wiport's embedded web server will send the client the HTML page containing the Java applet via the wireless router connected to the broadband modem over the internet designated by the red arrows. The green and blue arrows signify a wireless communication transmission. The actual Java applet is shown in Fig. 2. After the Java applet has loaded on the client's computer, it will wait for the client to click a button. He or she can either push the Check Mail button or the Check Temp button. As the client's mouse hovers over either of the two buttons they will change graphics to illustrate a potential option for the client. If the client chooses to see what the temperature is like outside where the mailbox is, he/she will click the Check Temp button. If that button is clicked on, the applet will send back to the Wiport, via the clients modem, over the internet to the broadband modem connected to the wireless router or access point to where wireless communication takes place with the Wiport, a request to check the outside temperature. The Wiport will convert the Ethernet packets from the wireless router to serial data and finally to RS-232C level data ($\pm$ 10V) that will be sent to the OOPic-R board through the communications board that converts these RS-232C voltage levels to TTL levels (0V and 5V respectively). The OOPic-R board will see this data and turn on the appropriate data I/O lines that propagate to the interface board of which will start the A/D conversion and power on the LM50 temperature sensor via the interface board. After the conversion has been made and received back from the interface board, the OOPic-R board will send this information out back to the communications board who in turn converts these received TTL voltage levels to RS-232C voltage levels suitable for the Wiport, designated with red arrows. The Wiport will convert the serial data back to packets and send it to the wireless router or access point. The attached modem will send it out over the internet to the client's modem and out to the computer that is running the Java applet. The final result will be calculated for degrees Celsius and Fahrenheit and displayed to the individual for interpretation. The same process happens for the checking of snail mail, only the different OOPic-R I/O lines will be turned on.

Figure 4 illustrates how the field sensors are positioned to provide effective results with minimum error. This is achieved by positioning a 650nm $\leq$3mW laser at an angle of 25 ° , embedded into a ½ inch thick aluminum slab mounted at the left lower corner side of the

mailbox wall. When activated by a mouse click, the laser module energizes as a result from OOPic-R's I/O pin 9, producing laser light that reflects and propagates along five mirrors within the cavity of the mailbox all the way to the back wall where the phototransistor is positioned. If there are no obstructions, in this case snail mail, hindering the light propagation incident on the phototransistor's base, this phototransistor will convert the laser photons into photocurrent at which will saturate the base and cause a current flow from the collector to the emitter, producing a voltage relative to the collector source voltage across a resister of 1kΩ, (0V for no light or 5V for laser light respectively). This voltage will drive the I/O pin 8 of the OOPic-R board to logic 1. This logic 1 is sent out to the Java Applet running on the client's machine. The Java applet will do an "if else" test condition on the result to select the right output to display to the client. It will either be "You have snail mail" for a logic 0 value, or "You have no snail mail" for a logic 1 value. In the event of mail in the mailbox, the laser light will be blocked from propagating to the phototransistor and causing logic 0 on the OOPic-R I/O pin 8 resulting in a "You have snail mail" condition.

The temperature sensor on the other hand resides on the small PC board of which is positioned in the support post of the structure. When activated by a mouse click, the temperature sensor will be energized by the OOPic-R's I/O pin 10 and produce a voltage out with respect to the temperature. It uses the following formula for calculating voltage out: Vout = (10mV/ °C * Temp °C) + 500mV to calculate temperature in degrees C. This voltage is sent to the OOPic-R's I/O pin 1 for A/D conversion. The OOPic-R also uses I/O pin 4 for a reference voltage (Vref) set around 2.56 volts and has to be set within the range of $1.75V \leq Vref \leq 5V$. 1.75V is the maximum voltage out of the LM 50 Temperature when the temperature is 125°C. It produces 100mV out when the temperature is -40°C, and 0°C at 500mV. Vref has to be equal to or in between the range of 1.75V and 5V. This is because the maximum voltage that can be applied on any I/O pin is 5V and at the same time, you can't have a value higher than 1 for the ratio of (Vout / Vref) in the formula. The OOPic-R takes these two voltage levels, Vout and Vref present on I/O pins 1 and 4, and produces a decimal value by using the following formula: Decimal Value = $[(Vout / Vref) * (2^n - 1)]$. This decimal value is sent to the Java Applet for processing. The Java applet takes the decimal value produced by the OOPic-R board and applies the following two math formulas for output temperatures in both degree Celsius and Fahrenheit: Temp °C = { ([(Decimal Value / 1023) * Vref] – 500mV) / 10mV } Temp °F = [{ ([(Decimal Value / 1023) * Vref] – 500mV) / 10mV } * (9 / 5)] + 32

The source code for the OOPic-R board resides in its EPROM in Fig. 2, and was created in the OOPic-R IDE environment coded in a high level language called Visual Basic. The Java Applet's source code, also shown in the Fig. 2, was coded and compiled with Sun Microsystems Java 2 platform (J2SE 5.0) and Xinox's JCreater LE. The code will reside in the Wiport's flash memory until retrieval is requested via a HTML page received from the client that was served up by the Wiport. The Java applet is a client side program that executes on the clients computer and not in the Wiport. The Wiport simply passes the information to the OOPic-R board for interpretation.

IV. Software Operation

In order for the two developmental boards inside the mail box to understand each other, excluding hardware requirements, software requirements have to be laid out and understood between the two. The OOPic-R board uses serial control protocol (SCP) as a way for outside users to control its hardware objects that were created and stored in its EPROM once the source code was compiled and uploaded at design time. A series of SCP string commands will allow access to the default properties of the various hardware objects in memory at any point in time via its com port. All the Wiport has to do is send to the OOPic-R board a series of SCP strings at a set time and the Wiport will be able to control the OOPic-R's hardware objects. There are five hardware objects, shown in Fig. 7**,** which are defined in the OOPic-R. These hardware objects take on the names: Laser, Phototransistor, TempSensor, Status, and Temp. The first four of the five are defined as Digital I/O Hardware Objects (oDio1). The last hardware object, Temp, is defined as an Analog to Digital Hardware Object (oA2D10). During operation or run time, the OOPic-R board just sits and runs the program in Fig. 8 continuously in the back round. At any point in time when the OOPic-R board sees a SCP command, namely /0, it will allow control of its defined hardware objects to whoever is accessing them, in this case the Wiport. The only hardware object that we let the OOPic-R control is the Status hardware object, oDio1. Its default property, value, will change from 1 to 0 and back to 1 again, every second, due to the Do Loop statement. This is not to say the Wiport can't control that hardware object because the OOPic-R is controlling it. The Wiport can take control over it as well; it's just that we had set up the Wiport to control the four hardware objects and allowed the OOPic-R to control the Status hardware object for a visual aid to me in seeing if the OOPic-R was running the program from start to finish. Basically, it is used to turn on and off a green LED every second.

These are the Serial Control Protocol (SCP) strings that the Wiport will have to send to the OOPic-R for the desired function to happen. The following are SCP strings needed to retrieve the temperature reading and check what the temperature is at the location of the mailbox. **\0** puts the OOPic-R in SCP mode, and **\A** ends the SCP session. The decimal numbers followed by the letter **J** are stored in the 16-bit memory address control register. The formatted two-character-per-digit hexadecimal followed by the letter **N** is what is stored in the memory location specified by the control registers. The decimal numbers followed by the letter **H** are stored in an 8-bit memory type control register. The decimal numbers followed by the letter **L** are stored in the 8-bit sub-address control register. The letter M is a SCP command that reads the memory specified by the control registers. When this command is given it sends its decimal values out the serial port followed by the lower case letter "m".

The following required SCP strings needed to check the temperature at the location of the mailbox are as follows:

> *TempSensor.value = ON\043J01N\A the hardware address for the oDio1 object is 43*
> *Temp.operate = True \044J144H57L01N\A the hardware address for the oA2D10 object is 44*

We have used a ~3 second delay to allow for the capacitors to charge up.

Temp.value = give me the result \044JM\A the hardware address for the oA2D10 object is 44

The result received will be "xxxxm", where "xxxx" can range from 0-1023. "m" has to be striped off before any mathematical operation can be performed.

TempSensor.value = OFF \043J00N\A the hardware address for the oDio1 object is 43
Temp.operate = FALSE \044J144H57L00N\A the hardware address for the oA2D10 object is 44

Here is what it would look like:

\043J01N\A\044J144H57L01N\A ***(3sec delay)*** *\044JM\A*
xxxxm what is returned \044J00N\A\044J144H57L00N\A

The math that has to be performed on the returned string "xxxxm" is as follows:

*Temperature in °C = ((((Decimal Value / 1023) * 2.5606) – 0.5) / 0.01)*
*Temperature in °F= ((((Decimal Value / 1023) * 2.5606) – 0.5) / 0.01) * (9 / 5)) + 32)*

Where Decimal Value = [(Vout / Vref) * $(2^n – 1)$], and is presented as the string "xxxxm". The lower case letter"m" has to be stripped off and the string has to be converted to a decimal value. For example, if I had a value of "235m" returned, I have to strip off the letter "m" and convert the string value "235" into the decimal value of 235, not the ASCII value 505153109.

The SCP commands to see if mail is present in the mailbox are as follows:
Laser.value = ON \041J01N\A the hardware address for the oDio1 object is 41
Again we have used a ~3 second delay to allow for the capacitors to charge up.

Phototransistor.value = give me the result \042JM\A the hardware address for the oDio1 object is 42
Laser.value = OFF \041J00N\A the hardware address for the oDio1 object is 41

The returned value from the OOPic-R will either be "0m" or "1m" where "0m" means that I have snail mail and "01" will mean that I have no snail mail.

Since there will be no math performed on the returned results, we do not have to strip off the lower case letter "m". We use it as a delimiter in telling it's the end of the returned string. Here is what it would look like:

\041J01N\A ***(3sec delay)*** *\042JM\A*
xm What is returned \041J00N\A

By sending the above SCP strings, we can check and see if the client has snail mail or what the temperature is at the location of the mailbox, at any point in time.

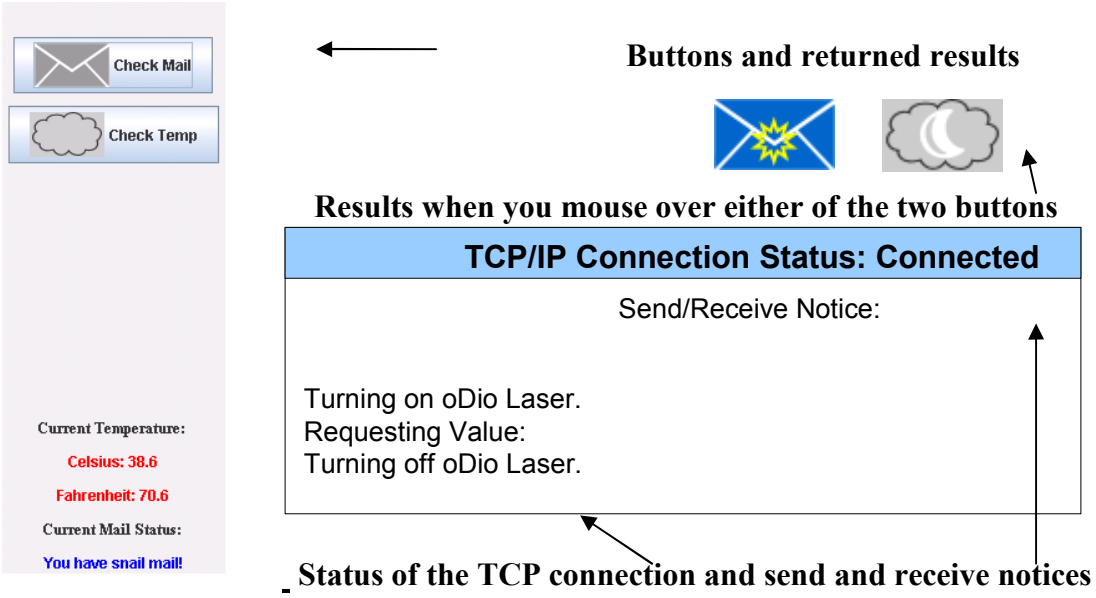
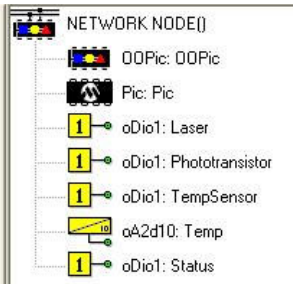


Fig. 7 The Java Applet with Zoomed in Sections

The Wiport is the key component in allowing remote communication to devices over the internet or local LAN. In doing so the Wiport has to transform the received packets into serial data and send it out its com port 0. The data that it receives and sends is useless to anybody until you design a program that knows what to do with the data. The programs are written as the Java applet and the Visual Basic programs. The Wiport can serve up a HTML documents that, in this case, has a Java applet embedded into the body of it, to a client. The Wiport is the medium for transfer of data based on what the applet tells it to pass along its com port. The Java applet runs on the packet side of the device and the resulting serial data runs on the serial side of the device where the Visual Basic program runs. The Java applet is running from the client's machine over the internet. The Visual Basic program is running on the device connected to the Wiport's com port 0, in this case, the OOPic-R. The Java source code for the Wiport, in the appendix, allows us to check either the presence of snail mail or the outside temperature at the location of the mailbox over the internet or LAN. We accomplished this by writing three classes named: DisplayError.java, OOPICRConstants.java, and TCPIP.java, which are used by the Java applet named Test.java to carry out the task of telling the OOPic-R board to basically read the value of the field sensors and relate the status of each invoked sensor back to the client. Together, these classes will create a graphical user interface (GUI) for the client to interact with, thus producing the desired SCP strings for the retrieval of the resulting value for temperature or presence of snail mail, based solely upon which button was clicked by the clients mouse. These Java files,

```
'This is the program that will run in the OOPic-R
'SCP will take control of the objects


Dim Laser As New oDio1
Dim Phototransistor As New oDio1
Dim TempSensor As New oDio1
Dim Temp as New oA2D10
Dim Status as New oDio1

Sub Main()

  OOPic.ExtVRef = cvOn 'Uses external voltage as a reference

  Laser.IOLine = 9
  TempSensor.IOLine = 10
  Phototransistor.IOLine = 8
  Status.IOLine = 12
  Temp.IOLine = 1

  Status.Direction = cvOutput
  Laser.Direction = cvOutput
  Phototransistor.Direction = cvInput
  TempSensor.Direction = cvOutput

  Do
    Status.Value = OOPic.Hz1  'Turns on and off at a rate of 1s
  Loop

End Sub
```



Five hardware objects

Figure 8 The IDE environment showing the Source Code and Hardware Objects

once compiled create class files with the same name, only now, with a class extension. A Test2.html file contains the Test.class file that extends the java applet class file. Together, they will form the heart of the communication process. The OOPICRConstants.class holds all the necessary SCP string commands and be waiting to pass along the desired string. When evoked, the DisplayError.class allows the errors to be displayed once caught, in a dialog box, in the event of errors with the connection and or platform incompatibility. The TCPIP.class opens a TCP connection, and allows reading and writing of byte arrays of the transmitted and received data from the WAN or LAN side on over to the serial side and vice versa. The serial com port 0 of the Wiport is defined as a number 10001, and is passed to this class from the Test.class. The IP address is acquired from the Wiport that was given to it by the wireless router through DHCP. This IP address is also passed to the TCPIP.class from the Test.class for a TCP communication socket to be established. Once a socket is created, communication to remote devices connected to the OOPic-R board can be controlled via the Wiport over the internet or LAN. The Interface and Communication board, also aid in this process, but only at the physical layer. The program flow is shown in Fig. 9.

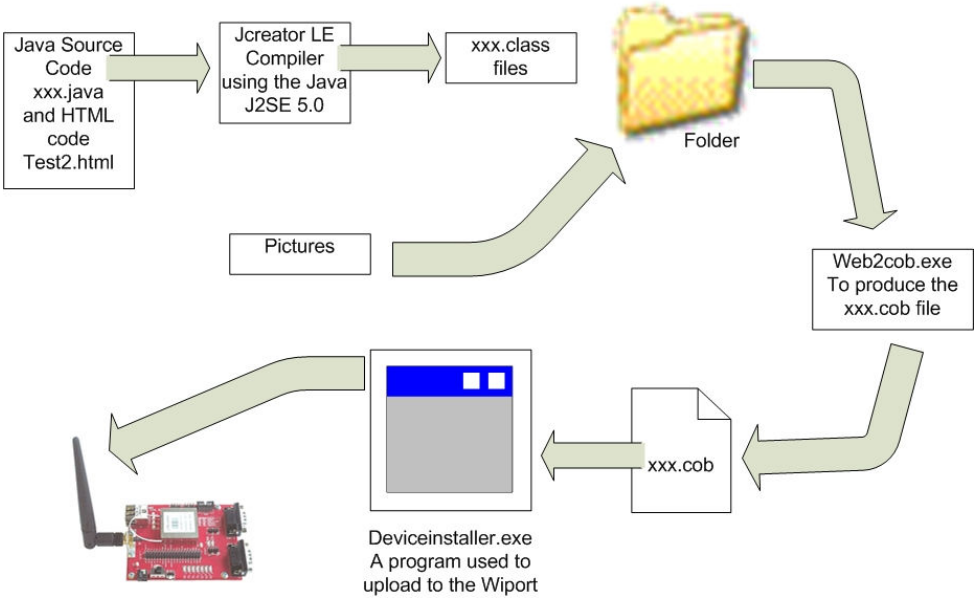


Figure 9 The program flow

V. Precautions, Limitations and Lessons Learned

We found that the laser module shell must be isolated from the power source. We did this by isolating the mailbox from the rest of the grounding system by cutting bigger holes in the aluminum plate that the mailbox bolts to between the ¾" thick oak. The same thing is true for the phototransistor mounting.

In the present state of the project, only one person can access the mailbox at a time. Once the browser that is running the applet is closed, or when move to another webpage, the mailbox is released from the client's session and thus allows for a new socket to be made for another client. This limitation can be removed in future work.

This senior design project involved the application of knowledge gained primarily in the classes such as: ECET 209 – Introduction to Microcontrollers, ECET 303 – Communication I, ECET 367– Internetworking and TCP/IP, and ECET 455 –C++ Object Oriented Programming. However, most important part was the aptitude the students acquire in the ECET program for identifying problems around us, researching on the topic and the components and subsystems. The lessons learned to do feasibility study and preparing time-line plan and task division and execution were part of the excellent course in ECET 397 –Directed Project Engineering. Many instances came when the task seemed impossible; however, the persistence and application opened the path to new solutions. Two of the examples were to solve the problem of isolation of laser source power supply and protection of the mail box from the surge in inclement weather.

VI Conclusion

Currently, there are no "Wi-Fi" networked mailboxes, let alone wired networked mailboxes on the market. In order to host the Wi-Fi mailbox, one would need a satellite or broadband connection to the internet and an 802.11b/g wireless access point or router. When utilizing the mailbox on the WAN side; one would need any type of internet connection along with a Java enabled web browser, i.e. Internet Explorer, or Mozilla, running on a Mac or a PC compatible system. The user can also utilize the Wi-Fi mailbox on the LAN side and not need an internet connection; only the switch integrated into the wireless router, of which most consumer level routers sport. In conclusion, this project will do well in the market in allowing the user to check if he/she has snail mail within the confines of their mailbox without leaving the home, work or school. Future improvements could include a camera in the mailbox to see actually what kind of mail you have received. Also, using more plastics in the design will make it cheaper to build and it would weigh a lot less.

Reference: (superscript in body)

[1] "Wi-Fi" 802.11b Network Mailbox with Temperature Awareness", ECET 491 – Senior Design Project, Phase II, Purdue University Calumet, Hammond, IN., May 2005

[2] "Electronic Communication systems", 6th Ed. –Wayne Tomasi, Prentice-Hall, 2001

[3] "Digital communications", 2nd Ed. – Andy Bateman (Addison-Wesley)

[4] Sheriff, Paul D. Paul Sheriff Teaches Visual Basic® 6. Ed. John L. Hawkins. Indianapolis: Que, 1999.

[5] Cadenhead, Rogers, and Laura Lemay. Sams Teach Yourself Java™ 2 in 21Days. Indianapolis: Sams, 2004

[6] Clark, Dennis. Programming and Customizing the OOPic Microcontroller: The Official, OOPic Handbook. New York: McGraw-Hill, 2003

[7] Flanagan, David. Java™ Examples in a Nutshell Java™ in a Nutshell. Sebastopol: O' Reilly, 2000

[8] Deitel, Paul J., and Harvey M. Deitel. Java How to Program. 6th ed. New York: Hall, 2004

[9] www.gridconnect.com

[10] www.superDroidRobots.com

[11] www.apinex.com

[12] www.newwark.com

[13] www.futurlec.com