122nd ASEE Annual Conference & Exposition
June 14 - 17, 2015
Seattle, WA
Paper ID #13971

Work-In-Progress: Remote Laboratory with a Microcontroller System as the Server

Dr. Abul K. M. Azad, Northern Illinois University

Abul K. M. Azad is a Professor with the Technology Department of Northern Illinois University. He has a Ph.D. in Control and Systems Engineering and M.Sc. and B.Sc. in Electronics Engineering. He has been in academics for 15+ years, and his research interests include remote laboratories, mechatronic systems, mobile robotics, and educational research. In these areas, Dr. Azad has over 100 refereed journal and conference papers, edited books, and book chapters. So far, he has attracted around $1.7M of research and development grants from various national and international funding agencies. He is a member of the editorial board for a number of professional journals as well as an Editor-in-Chief of the International Journal of Online Engineering. He is active with various professional organizations (IEEE, IET, ASEE, and ISA) as well as a member of board of Trustees of CLAWAR Association. He has served as Chair and Co-Chairs of numerous conferences and workshops, in addition to serving on the program committees of around 30 international conferences. Dr. Azad is a project proposal reviewer with various national and international funding agencies in US, Europe, and Australia

Mr. Syed Abdul Hadi Razvi, Northern Illinois University

Work-In-Progress: Remote Laboratory with a Microcontroller System as the Server

Abstract

Sustainability of Internet based remote laboratories is a major issue. Two of the reasons are development costs and the complicated design of such systems. This paper will report the design and development details of a remote laboratory system in which a commercially available microcontroller system will be used to replace the costly workstation and server. The proposed system uses an Arduino board to reduce the implementation costs and design complicacy as well as the physical size of the system. The development process has allow the group to understand the capabilities and challenges of microcontrollers for remote laboratory applications. Considering this is a work-in-progress, this paper provides a development description of the system; full details will be reported at a later stage.

1. Introduction

With the advent of the Internet of Things (IoT) we are not far from a time in which objects, animals, and people will be provided with a unique identifier, through which they can communicate from anywhere-anytime via a network connection.[1,2,3] This philosophy is already being used in education where performing experiments involving physical systems remotely over the Internet involving real hardware is now a reality.[4] Considering the progress in computer and internet technology, there is potential for growth in e-learning initiatives. This is also reflected in the remote laboratory area, where academics and researchers have taken a lead in developing remote laboratories.[5] Three kinds of remote laboratories are available over the web: complete simulation, full hardware, and hybrid. The first one is purely computer simulated experiments, while the second one is completely hardware based experiments. The third category is a mixture of real hardware and computer simulation, like Hardware in loop experiments.

However, one of the difficulties with the development of remote laboratories is the use of a personal computer (PC) that needs to be connected with an experiment. This increases the implementation costs of the remote laboratory facility when only a limited part of the processing power of a PC is needed for this development. In addition there needs to be a server to facilitate client access to the remote experiments. All these make the sustainability issue of remote laboratories more pressing.[6] To address this issue, this paper will report the design and development of a remote laboratory system in which a commercially available Arduino system will replace a PC. This will reduce the implementation costs as well as the physical size of the system. An Arduino system comes with a web server feature, analog to digital converters, and other input/output (I/O) requirements.

2. Traditional Remote Lab Design

There are a number of approaches to developing remote laboratories.[7,8,9] However, the main concept behind all these system designs comes as a three stage structure: the server for remote

access, the experiment workstation, and the experiment itself. A general block diagram showing a traditional remote laboratory structure is shown in Figure 1.

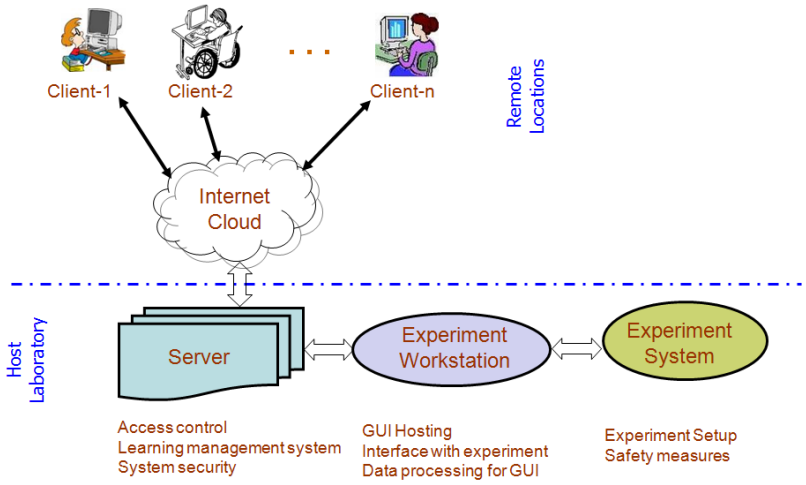


Figure 1: Block diagram of traditional remote laboratory structure.

With this structure, the experiment workstation connects with one or more experiments, depending on the capacity of the workstation and available I/O. An experiment system provides the experimental setup along with local safety measures. The experiment workstation implements the control design using the received data to produce commands for the experiment. It also processes the input/output data to make it suitable for a graphical user interface (GUI), usually hosted within the experiment workstation. In a way, one can operate the experiment locally from the experiment workstation via the GUI. To provide remote access, the GUI is made available to the remote user via a server. The server is also used for access control, the learning management system, and system security.

However, one of the major drawbacks of a traditional remote laboratory is the use of a computer as the experiment workstation. Although using a computer provides flexibility, it introduces considerable cost and requires additional space. Considering the advancement of microcontroller technology, it is now possible to think about using a microcontroller system instead of a computer as an experiment workstation. This paper will report the development process of a remote laboratory system where a commercially available microcontroller based system will function as the experiment workstation as well as a server.

3. Microcontroller Based Remote Lab

A number of powerful microcontroller/microcomputer systems for a remote laboratory design are available commercially; two major ones are Raspberry Pi and Arduino.[10,11] Arduino is a microcontroller board, whereas Raspberry Pi is a fully functional computer. The Raspberry Pi is many times faster than an Arduino when it comes to clock speed and also has higher RAM capability. Raspberry Pi is an independent computer that can run an actual operating system in Linux. It can multitask, support two USB ports, and connect wirelessly to the Internet. These make Raspberry Pi superior to Arduino when it comes to software applications.

On the other hand, Arduino has a real-time and analog handling capability that the Raspberry Pi does not. This flexibility allows the Arduino to work with just about any kind of sensor or chips. In most cases Raspberry Pi needs extra hardware for data sensing, whereas Arduino is more flexible in this matter. The Arduino integrated development environment (IDE) is significantly easier to use than Linux, since Arduino is not designed to run an operating system and has the flexibility of just plug and play. Arduino has more learning resources and is simple and robust; it also works with any computer and can run off of a battery. Additionally, it can be safely turned on and off at any time, whereas Pi needs an understanding of Linux and some programming, such as python, and can be damaged by unplugging it without a proper shutdown. Considering all these factors, it was decided to use an Arduino for this development.

3.1 System design

A system level block diagram of the proposed system is shown in Figure 2, where the experiment workstation and the server (shown in Figure 1) have been replaced with an Arduino microcontroller system. The experiments will be available to the remote clients using a PC as well as through mobile applications. The idea is implemented through the design and development of a small scale house.

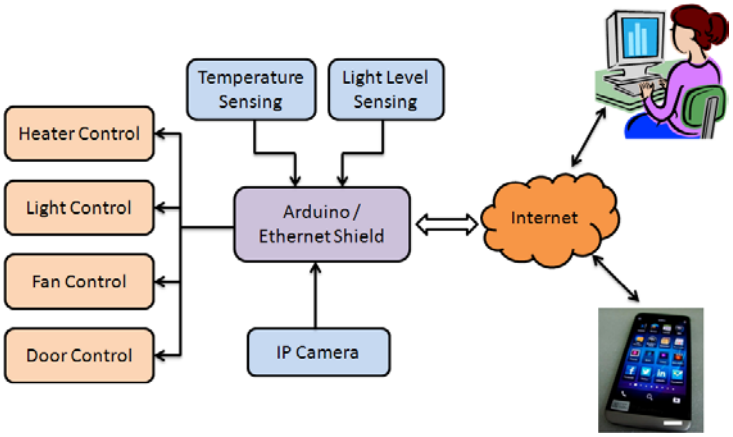


Figure 2: System block diagram of the developed remote lab.

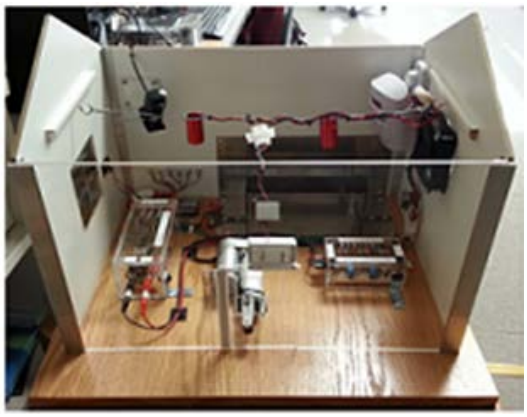


Figure 3: An image of the designed house.

This is a one room house (see Figure 3) with a number of features: temperature control, light control and web camera. Each of these features is associated with relevant sensors and actuators. All of the sensors are connected to the Arduino inputs and are then passed through relevant controller designs. The controller output is then passed to the relevant actuators via Arduino's output ports. As well as controller designs, the Arduino system is also used to implement the server designs. In addition to the Arduino board, an Ethernet shield (from the Arduino family) is used for server implementation. The server is then made available for Internet access by the remote clients using a personal computer or a smart phone (shown in Figure 2).

3.1.1 System Controller and Web Server

An Arduino board is used to implement the system controller and can be considered as the heart of the system along with an Ethernet Shield utilized for web server implementation. The Arduino program interacts with the sensors and actuators when the Ethernet Shield hosts the server along with GUI designs. The Ethernet Shield is also fitted with a memory card for hosting the GUI programs. Please see Appendix A for details of Arduino mega and the Ethernet Shield.

A connection diagram between the Arduino board and the Ethernet Shield is provided in Figure 4. There are four connections for SPI and two other connections for enabling the Ethernet control and Memory card. In this project a micro memory card is plugged in on the Ethernet Shield, which is holding an index.htm file along with the GUI code written in html and Java script.

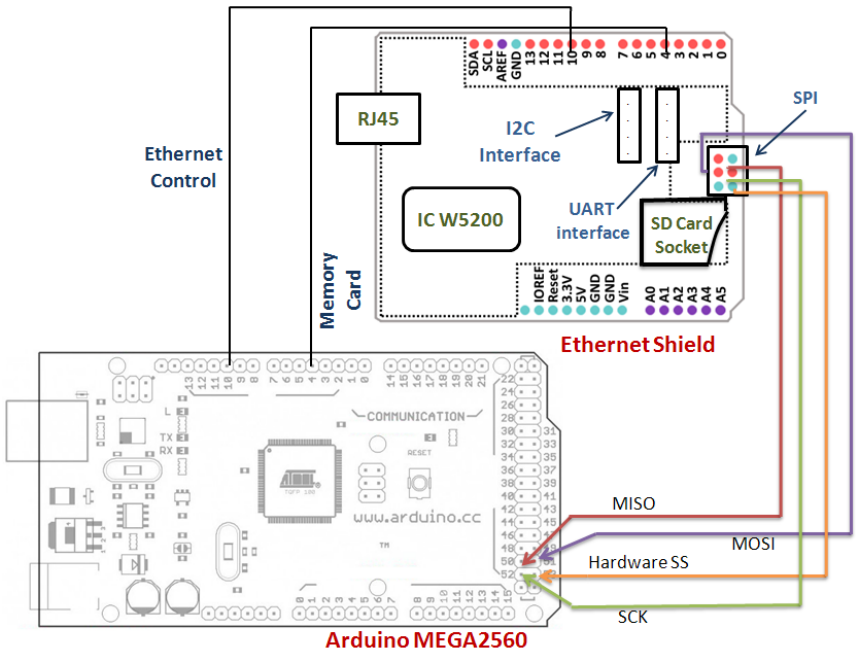


Figure 4: Diagram showing the connection between the Arduino and Ethernet Shield.

3.1.2 Sensors and Actuators

The Ardunio board is connected with sensors and actuators via its I/O ports. A hardware connection diagram of the Arduino board with the sensors and actuators is shown in Figure 5.

The board collects the data using its A/D converters, processes the data, and produces command outputs for the actuators.

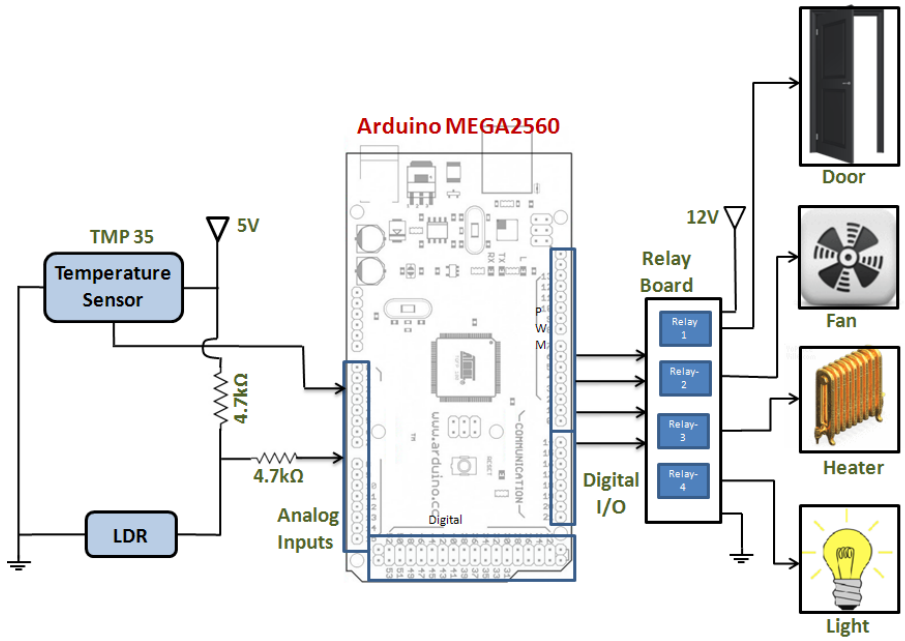


Figure 5: Connection diagram between the Arduino and sensors/actuators.

The temperature control system is implemented using an analog temperature sensor, an electric heater for heating and a fan and door for cooling. The temperature sensor is a semiconductor device composed of a temperature sensor as well as an inbuilt amplifier. [12] This is a low voltage, precision temperature sensor. It provides a voltage output that is linearly proportional to the temperature. The analog signal is converted to digital using the inbuilt analog to digital convertor within the Arduino. The data are then used as an input to the designed controller within the Arduino, and the output is used to activate the actuators (heater, fan, and door).

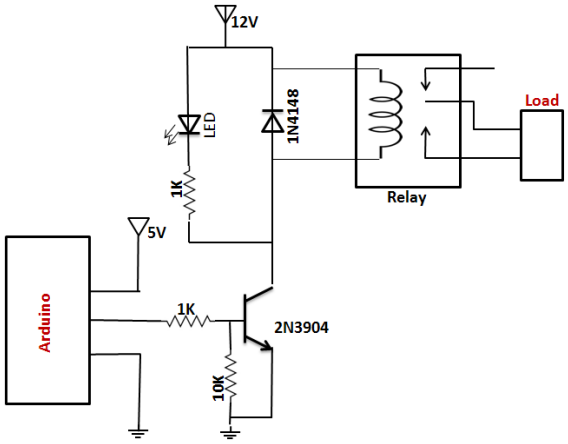


Figure 6: Schematic of single relay controller board.

To control the light, a light dependent resistor (LDR) is fitted in the house and the output is connected to the Arduino board via its analog input. Based on the light intensity, the Arduino board sends commands to turn the light on/off. All the actuators are driven by relays while

taking commands from the Arduino board. A schematic diagram of the relay used for this project implementation is provided in Figure 6.

The web cam feature is implemented using an IP camera, which streams live data and can be viewed remotely from any place via the Internet. The web cam is directly connected to the Ethernet Shield. The image from the web cam is embedded within the GUI. The camera used in this project is a Panasonic BL-C210 IP network camera, which is usually used for CCTV Video Camera. [13]

3.2 Communications between Arduino, Web Server and Client

Arduino has an inbuilt library for sever implementation, while the Ethernet Shield uses that library to host the web server application.[14] The header file for this library is called *Ethernet.h*. The Ethernet Shield acts as the client for the Arduino and a web server for remote users. Apart from GUI, all other programs are hosted within the Arduino board.

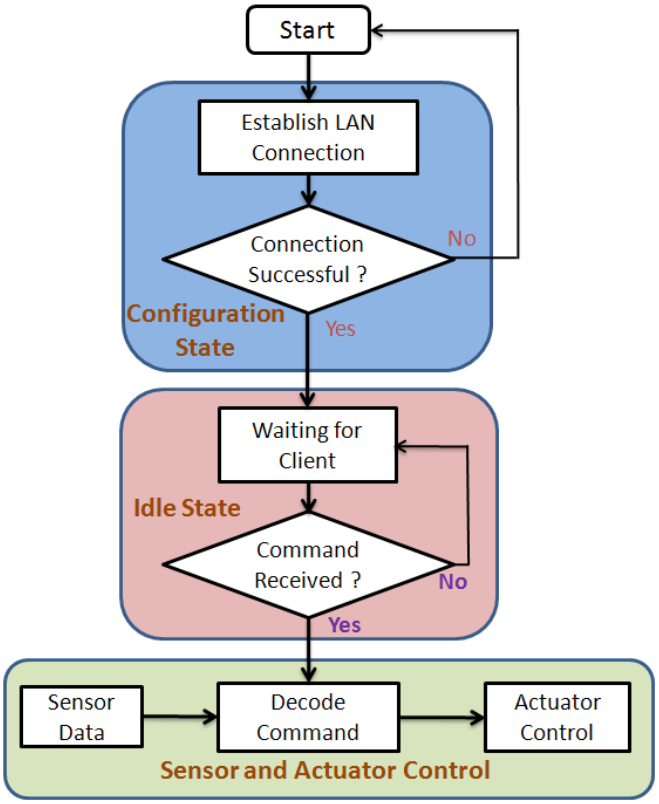


Figure 7: Flow chart for connection establishment with internet.

A diagram of the flowchart of three major tasks of this program: configuration state, idle state, and sensor/actuator control is shown in Figure 7. When the client makes a request or any other action happens within the client's browser, it sends a request using http. When the server identifies and accepts the request protocol, it then generates and sends an http response to the client's browser. Once a connection between a client and server has established, it will then enter into the configuration stage. Here the port number and url of the server are checked to establish the connection to the server. If the connection is successful, then the server waits for a

command. The server enters the idle mode until it receives any command. When the server receives a command, it will then allow users to take control of the actuators via sensor and actuator control.

A flowchart for the Arduino program used to communicate with the sensors and actuators is provided in Figure 8. Arduino collects data from the LDR and temperature sensor. The light switches on if the LDR data show the light level is lower than the threshold set for a given case. In terms of temperature management, if the measured temperature is lower than the pre-adjusted threshold, then the heater goes on and closes the door and switches off the fan. On the other hand, if the measured temperature is higher than the threshold, then the heater goes off and opens the door and turns on the fan.

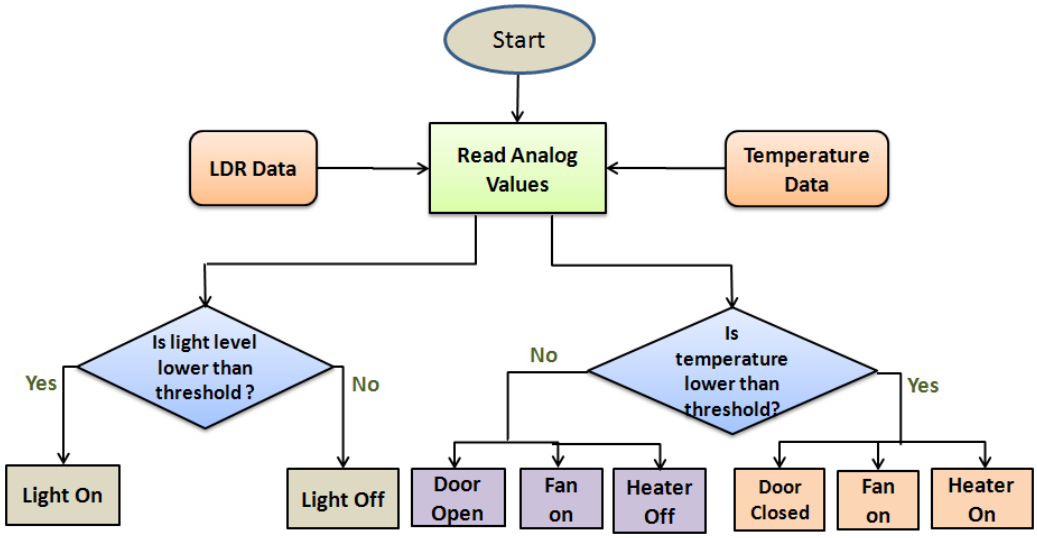


Figure 8: Flowchart for Arduino program for temperature and light control.

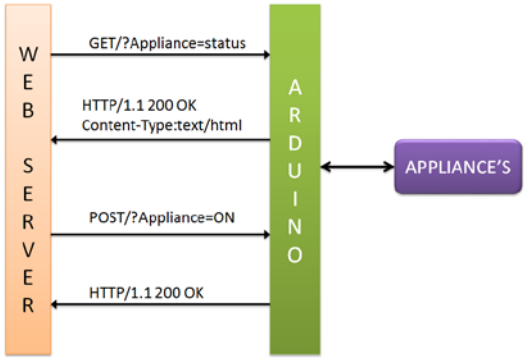


Figure 9: Communication protocol between Arduino and web server.

The complete protocol for establishing communication between the client and web server as well as controlling the appliances is shown in Figure 9.

The GUI consists of HTML, CSS and Ajax code. The client gains access to the system via a GUI. The GUI is initiated by the *Index.htm* file, which is hosted on the memory card plugged into the Ethernet Shield. The GUI interacts with Arduino for getting data from the appliances.

The software interfacing between the GUI and Arduino is done via a java script object nation (JSON). A diagram showing the process in shown in Figure 10.

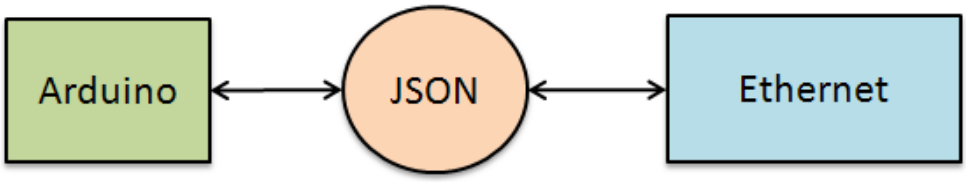


Figure 10: Communication protocol between Arduino and Ethernet Shield

3.3 HTTP Request and Response

When a button is clicked within the client's browser, the browser generates an http GET request that sends the name and value from the button to the web server. The following is the http request sent from the client browser to the web server:

GET /?APPLIANCE1=2 HTTP/1.1
Host: 10.0.0.20
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,/*;q=0.8*
Accept-Language: en-ZA,en-GB;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://10.158.161.25/
Connection: keep-alive

When a button is clicked again, the following http request is sent from the browser to the web server:

GET / HTTP/1.1
Host: 10.0.0.20
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-ZA,en-GB;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http:// *10.158.161.25*/?APPLIANCE1=2
Connection: keep-alive

The web server reads the http request header and checks for the text APPLIANCE1=2, and if found, the Arduino will toggle the APPLIANCE from off to on or on to off. Both of the above requests contain the APPLIANCE1=2 text, although in different places. When checking the button, the text is part of the GET request line. When unchecking the box, the text is part of the Referrer: header.

3.4 Graphical User Interface

A user gains access to the system (house in this case) via a GUI. An image of the GUI is shown in Figure 11. The GUI has three areas. One is the temperature information within the house, the second is the appliance status, and the third is the camera view. The temperature status is shown in Fahrenheit. The appliances are a lamp, the door, a fan, and a heater. The appliances will be controlled automatically based on the sensor information. At the same time a client can trigger an appliance 'on' or 'off' by clicking on the button under each item.

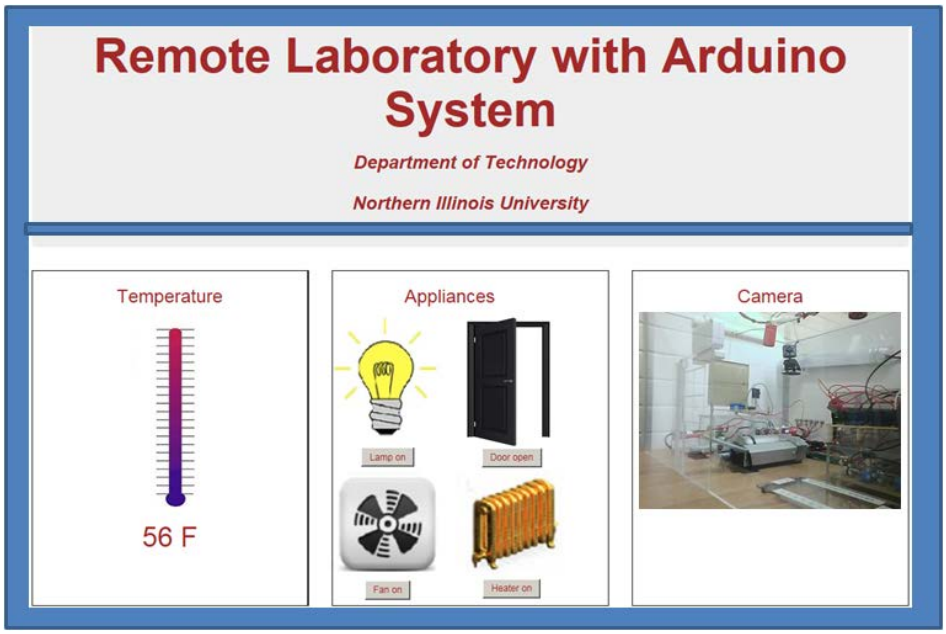
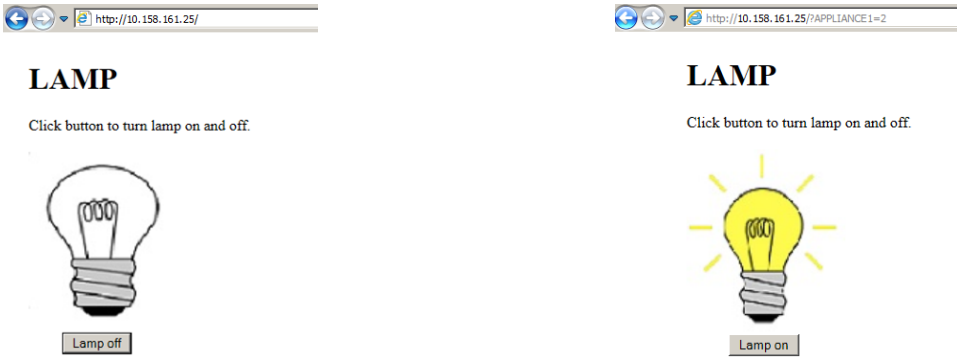


Figure 11: Image of the developed GUI.



(a) Default status. (b) Light has been turned on.

Figure 12: Details of lamp control.

A case study is provided to demonstrate the process of communication between the Arduino and the web. This is implemented by using an example for lamp control. An image of the GUI part for lamp control is shown in Figure 12. Figure 12(a) shows the default url address and current status of lamp when a client accesses the GUI. When a client clicks on the button placed under the bulb, the client's browser will send a specific url address (along with some additional text) to

the web server. The url address in this case is http://10.158.161.25/?APPLIANCE1=2. This is shown in Figure 12(b).

Here the "get" method is being called to turn on APPLIANCE1, which submits a form to the XML generated by Arduino, making changes in the button state and turning the lamp on. A control is added to the HTML form using the <button> tag. The button tag is used to create a button, and the following fields are included in the button tag:

- **type="** button **"** – displays this input control as a button
- **name="APPLIANCE1"** – user defined name of the control
- **id="APPLIANCE1"** – user defined id of the control
- **onclick="lamp();"** – submit the form when the button control is clicked

4. Conclusions

This paper illustrates the development process of a remote laboratory facility using an Arduino system as the experiment workstation and as the server. This process will reduce the development cost and also reduce design complicacy. An Arduino system has an advantage over the other commercially available microcontroller/microcomputer systems. The system is implemented with the design of a temperature and light controlled small scale house. The paper demonstrates the hardware design and implementation as well as the software details. For remote access to the experimental system, a design of the GUI is also presented. All the hardware and software modules are developed and tested before integrating them as a system. The whole system is also working as predicted. The proposed remote laboratory design is a work-in-progress, and the authors are still working to gather long term test results as well as determine the efficiency and reliability of the system. Those findings will be reported through future publication. The whole project is a tremendous challenge for the group while trying to come up with a cost effective way of remote laboratory implementation. This exercise provided the group with an understanding of the capabilities and the challenges for remote laboratory developments using microcontrollers.

Acknowledgement

The authors would like to thank the National Science foundation for its support for the reported work. This paper is based on a NSF TUES (Transforming Undergraduate Education in Science) project, award number DUE-1140502. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

5. References

[1] T. Danova, (2013). Morgan Stanley: 75 Billion Devices Will Be Connected To The Internet Of Things By 2020, Business Insider, [online]. http://www.businessinsider.com/75-billion-devices-will-be-connected-to-the-internet-by-2020-2013-10#ixzz3QWI7CyZh, (viewed on February 1, 2015)

[2] R. Piyare, Internet of Things: Ubiquitous Home Control and Monitoring System using Android based Smart Phone, *International Journal of Internet of Things*, Vol. 2 No. 1, 2013, pp. 5-11. doi: 10.5923/j.ijit.20130201.02.

[3] G. Kortuem, F. Kawsar, D. Fitton, and V. Sundramoorthy, "Smart objects as building blocks for the internet of things," *Internet Computing, IEEE*, vol. 14, pp. 44-51, 2010.

[4] D. Lowe, S. Murray, E. Lindsay, and D. Liu, Evolving remote laboratory architectures to leverage emerging Internet technologies, *IEEE Transactions on Learning Technologies*, vol, 2, no. 4, pp.289-294.

[5] E. S. Ruiz, A. P. Martín, P. Orduña, E. R. Larrocha, R. Gil, S.Martin, G. Díaz, M. J. Albert, A. C. Santos, R. Meier, M. Castro, Virtual and Remote Industrial, *IEEE Industrial Electronics Magazine*, vol. 8, no. 4, pp.45-58, 2014.

[6] A.K.M. Azad, Internet Accessible Remote Experimentation: Setting the Right Course of Action, *International Journal of Online Engineering*, vol. 6, no. 3, pp.4-12, 2010.

[7] J.M. Andujar, A. Mejias, M.A.Marquez, Augmented Reality for the Improvement of Remote Laboratories: An Augmented Remote Laboratory, IEEE Transactions on Education vol, 54, issue. 3, pp. 492-500, 2011.

[8] S. E. Esche and C. Chassapis, On Infrastructure for Educational Online Laboratories. Book chapter- Azad, A. K., Auer, M. E., & Harward, V. (2012). *Internet Accessible Remote Laboratories: Scalable E-Learning Tools for Engineering and Science Disciplines* (pp. 1-645). Hershey, PA: IGI Global. doi:10.4018/978-1-61350-186-3

[9] Internet Accessible Remote Laboratories, [online]. http://www.niu.edu/remotelab/ (viewed on Feb 1, 2015)

[10] Arduino Home, [online], http://arduino.cc/en/Main/ArduinoBoardMega2560, (viewed on Feb 1, 2015).

[11] Raspberry Pi Home, (online). http://www.raspberrypi.org/, (viewed on Feb 1, 2015).

[12] Analog Devices, TMP36:Voltage Output Temperature Sensors, http://www.analog. com/en /mems-sensors /igital-temperature- sensors/tmp36/products/product.html (viewed on Jan 31, 2015)

[13] Panasonic Operating Instructions (Network Camera), BB-HCM311A, http://service.us. panasonic.com/OPERMANPDF/BBHCM311A.pdf

[14] Arduino Web Server Tutorial, [online], Jan 14, 2013, http://blog.startingelectronics.com/?p=546, (viewed on Feb 1, 2015).

Appendix-A: Details for Arduino mega and Ethernet Shield

An Arduino mega board is used as the system controller, which is a microcontroller board with ATmega 2560 as the processor. It is an 8-bit microcontroller with 32kBytes of flash memory, 1KB EEPROM, 2KB of SRAM, and 23 general purpose I/O ports, 32 general purpose registers, 3 flexible timers/counters with compare modes, a byte-oriented 2-wire serial interface, one SPI parallel port, a 6-channel 10-bit ADC, programmable watchdog timers with internal oscillator, and 5 software selectable power saving modes, making it a very powerful tool for real time processing. It also has the capability of reading while writing. An image of the Arduino Mega is shown in Figure A-1.

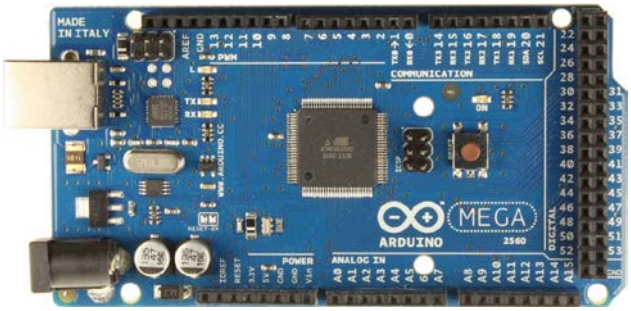


Figure A-1: An image of Arduino Mega 2560.

The web server is implemented by using the Ethernet Shield and is physically fitted with the Arduino board. An image of the Ethernet Shield is provided in Figure A-2. The Ethernet Shield uses a serial peripheral interface (SPI) protocol to communicate with the Ardunio board. It is based on the Wiznet W5100 Ethernet chip. The Wiznet provides a network Internet provider stack capable of both transmission control protocol (TCP) and user datagram protocol (UDP). It supports up to four simultaneous socket connections. There is an onboard micro-SD memory card slot, which can store files for serving over the network. Ethernet Shield is compatible with Arduino Uno and Mega. Using an Ethernet library (hosted in Arduino), the web server connects to the Internet using this particular hardware.

Figure A-2: An image of an Ethernet Shield.