Proc. of Int. Conf. on Advances in Signal Processing and Communication, SPC

Internet Equipped Notice Board an Application of Internet of Things

[1]Dr. Shaik Qadeer and [2]Mohammed Faizuddin
[1]Muffakham Jah College of Engineering and Technology, Electrical Engineering Department, Hyderabad, India
Email: haqbei@gmail.com
[2]Muffakham Jah College of Engineering and Technology, Electrical Engineering Department, Hyderabad, India
Email: mohammedfaizuddin@live.com

***Abstract*— Electronic or Moving Message Boards are being used in a wide variety of applications for communicating information to people quicker and in a cost effective manner when compared to traditional posters or paper notice boards. . While e-mail is a way to converse privately with one or more people over the Internet, electronic notice boards are totally public. Any message posted on one can be read (and responded to) by everybody else in the organization who has viewed it. This paper focuses on designing an e-display which can accept data wirelessly from any authorized person who has the access of the web terminal ; it means integrating the traditional moving message displays with an ARM Microprocessor (Arduino) so that they can be accessed wirelessly as an application of IoT (Internet of Things).**

***Index Terms*— Notice Board, Internet of Things, e display, Arduino, Project and Wi-Fi.**

I. INTRODUCTION

A Notice board is a surface intended for the posting of public messages, viz. to convey important news over an institution or places similar, to advertise items needed or required to sell, announce events, or provide information. Traditional notice boards are often made of a material such as cork to facilitate addition and removal of messages with the help of a printed material like paper which also had an impact on the wastage of paper as a number of notices would be printed wouldn't even be used.

Notice boards are particularly prevalent at universities. They are used by many sports groups and extracurricular groups and anything from local shops to official notices. Dormitory corridors, well-trafficked hallways, lobbies, and freestanding kiosks often have cork boards attached to facilitate the posting of notices. At some universities, lampposts, bollards, trees, and walls often become impromptu posting sites in areas where official boards are sparse in number.

But traditional notice boards have become a thing of the past. Nowadays Moving message displays are becoming a global replacement for traditional bulletin boards. Electronic notice boards are sometimes referred to as message boards. The terms notice board and electronic moving message displays are interchangeable, although often one Electronic Notice Board cannot replace the whole era of the traditional paper notice boards. It has the potential to grow and serve as a replacement in the coming future. Smart Notice boards are available in the market, the method of accessing the displays is either physical (through USB/COM port), or GSM/DTMF **[1-2]** based. Looking into the future this paper focuses on integrating the very own moving message displays into wireless e-displays; using the concept of Internet of Things.

The organization of the paper is as follows: Section II covers the IoT, whereas Section III gives the detail of the proposed work followed by Section IV which covers Implementation and related issues. Finally Section V concludes the paper.

II. Internet Of Things

The **Internet of Things** (**IoT**) is the network of physical objects or "things" embedded with electronics, software, sensors and connectivity to enable it to achieve greater value and service by exchanging data with the manufacturer, operator and/or other connected devices. Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing Internet infrastructure **[3]**.
With every breath technological entrepreneurs take, a new idea is pitched. Keeping up with these everyday breakthroughs can be tough. In less than 10 years we have witnessed the launch of yet a new wave, Web 3.0. Components of this breakthrough are still new and not many people are familiar with these concepts. In order to seize the opportunity to inform individuals, we created an Internet Equipped Notice Board that was focused on the area of Internet of Things. Internet of Things is a vision that was introduced in 2009. This vision encompasses the idea of connecting all devices and gadgets to the internet. The Internet of Things is truly changing our world. It is enhancing our lives, businesses, health and society as a whole by developing products which would ease our life. It is estimated that by 2020, 50 billion devices will be connected to the internet **[3].**
Things, in the IoT, can be a wide variety of devices such as heart monitoring implants, biochip transponders on farm animals, automobiles with built-in sensors, or field operation devices that assist fire-fighters in search and rescue. These devices collect useful data with the help of various existing technologies and then autonomously flow the data between other devices. Current market examples include smart thermostat systems and washer/dryers that utilize **Wi-Fi** for remote monitoring of themselves **[4]**.
The vision of the Internet of Things has evolved due to a convergence of multiple technologies, ranging from wireless communication to the Internet and from embedded systems to micro-electromechanical systems (MEMS). This means that the traditional fields of embedded systems, wireless sensor networks, control systems, automation (including home and building automation), and others all contribute to enabling the Internet of Things (IoT) **[5]**.
The concept of the Internet of Things first became popular in 1999, through the Auto-ID Center at MIT and related market-analysis publications. Radio-frequency identification (RFID) was seen as a prerequisite for the Internet of Things in the early days. If all objects and people in daily life were equipped with identifiers, computers could manage and inventory them. Besides using RFID, the tagging of things may be achieved through such technologies as near field communication, barcodes, QR codes and digital watermarking **[6]**.

III. Hardware and Software Description

A. Hardware Description

To visualize the concept of the paper was just the first step of the paper. Choosing the Hardware components was the next step. The Hardware used here to implement the proposed application is :

1. Arduino Uno
2. Ethernet Shield W5100
3. TTL to UART Convertor
4. A Wireless Router
5. A Moving Message Board

Following is the brief introduction to the hardware:

Arduino Uno
The Arduino Uno is a microcontroller board based on the ATmega328 .It has 14 digital input/output pins, 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started **[7].**

Ethernet Shield W5100
The Arduino that we were going to use did not have the in-built capability to connect to a network through a RJ 45 cable ; to make this possible we used an Ethernet Shield here .The Arduino Ethernet Shield connects

your Arduino to the internet in mere minutes. Just plug this module onto your Arduino board, connect it to your network with an RJ45 cable and follow a few simple instructions to start controlling your world with internet. It is based on the Wiznet W5100 Ethernet chip **[8]** . The Wiznet W5100 provides a network (IP) stack capable of both TCP and UDP. We use the Ethernet library to write sketches which connect to the internet using the shield.

TTL to UART Convertor
The outputs through the Arduino were TTL based which was a very low level signal; the board which we used here readily accepted the RS 232 signals. In this regard we used a TTL – RS 232 Convertor as an interface between the board and the Arduino. The TTL-RS232 Adapter was used to convert TTL level signals to an RS-232 interface. The TTL side is a 9- pin female connector and the RS-232 side is a 9-pin male connector. The unit was powered through connector pins **[9].**

A Wireless Router
The basic use of a router is usually routing one or more devices in one place. We used a TP Link Wireless Router over here in order to create a Wireless Local Area Network **(WLAN)** which would basically be the range of the board from where it could accept messages. This Router was connected to the Ethernet Shield through an RJ 45 Cable **[10]**.

A Moving Message Board
A scrolling moving message display was used here for the paper. This board had vast programmable scope and it could be manually programmed using a Hyper Terminal window while it was connected to a PC or Laptop through an RS 232 cable **[11]**.

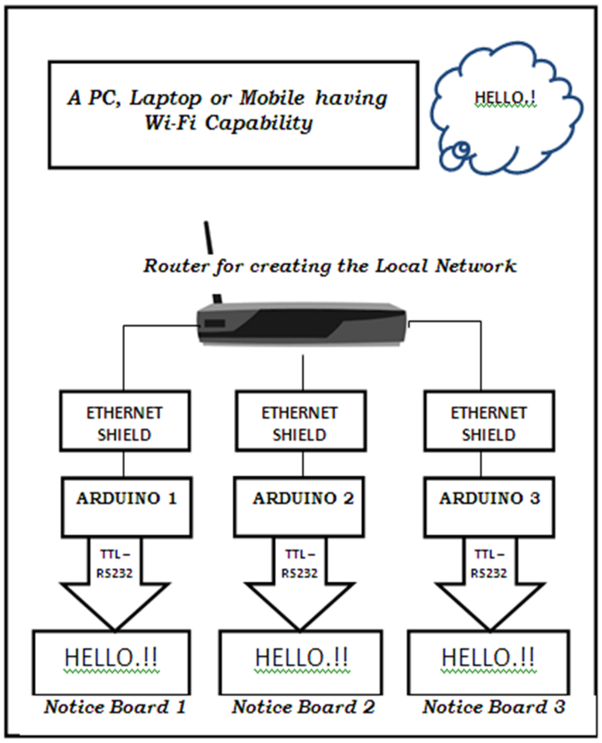


Figure 1 Block Diagram of Proposed work

B. Software Description

The only software that was used in implementing the paper was the one which helped us writing and uploading the sketch (program) into the Arduino which was the Arduino IDE.The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and derives from the IDE for the Processing programming language and the Wiring papers. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click [**12].**

IV. Implementation

A. Assembling the Hardware

The block diagram shown in Figure. 1 gives us the view of how we can assemble the various components for the overall implementation. As seen the notice board is interfaced to the
Arduino using a TTL- RS 232 jack, an Ethernet shield is then stacked over the Arduino to enable the Arduino to connect to
the network. This setup is then put into a network by connecting it to a wireless router that interfaces with the Ethernet Shield through an RJ 45 cable.
The PC/Client which needs to have access of the board simply needs to connect to the WLAN network created by the wireless router using a known pass key.

B. Configuring the router

After assembling the hardware; we encountered a problem which was related to the dynamically changing IP of the Ethernet Shield. This made it difficult for us to access the board as there was a different IP assigned every time the router used to reset. Routers are usually configured to give the devices trying to connect a different IP every time the router resets. This was a major setback to the paper because we needed to login to the routers configuration window and look through the DHCP client list to get the IP address of the board and then we could access it, which was not at all feasible.
In order to solve this problem, a technique is used ie. “IP-MAC binding”, which is a part of router configuration. This means configuring the router such that whenever our Ethernet shield tries to ping the router, the router is supposed to give it a fixed IP over which it could be accessed **[13].**
We did this with the help of IP MAC binding where we logged into the administration terminal of the router and assigned a fixed IP of “XXX.XX.XX.XXX” to the MAC address of the Ethernet Shield. Now whenever the Arduino stacked with Ethernet Shield used to ping the router for an IP the assigned IP is always given to it, so we needn’t worry about the dynamic IP problem.

C.Code Format and Description

The codes you write for your Arduino are known as sketches. They are written in C++.
Every sketch needs two *void type functions*, setup () and loop (). A void type function doesn’t return any value. The setup () method is ran once at the just after the Arduino is powered up and the loop () method is ran continuously afterwards. The setup () is where we want to do any initialisation steps, and in loop () we want to run the code we want to run over and over again.

The code was then written in the Arduino IDE in the following syntax:

```
void setup()
{

}

void loop()
{
```

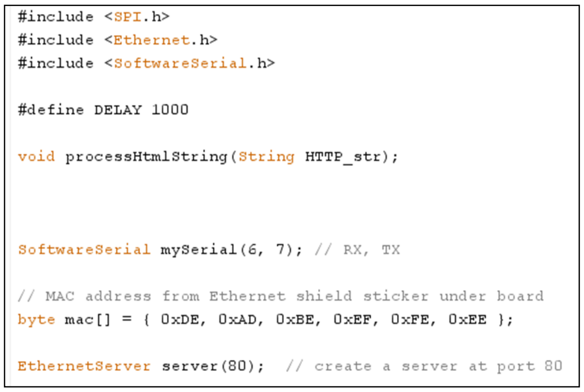
In order to go forward with the programming we have to choose the library files to be used. In this code we used three library files:

D. SPI.h

This library allows you to communicate with SPI devices, with the Arduino as the master device. Serial Peripheral Interface (SPI) is a synchronous serial data protocol used by microcontrollers for communicating with one or more peripheral devices quickly over short distances.

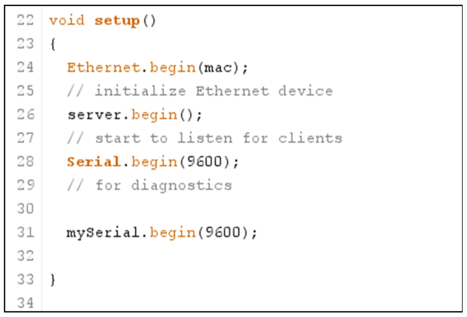
E. Ethernet.h

Along with the Arduino Ethernet Shield, this library allows an Arduino board to connect to the internet. It can serve as either a server accepting incoming connections or a client making outgoing ones.



```
#include <SPI.h>
#include <Ethernet.h>
#include <SoftwareSerial.h>

#define DELAY 1000

void processHtmlString(String HTTP_str);



SoftwareSerial mySerial(6, 7); // RX, TX

// MAC address from Ethernet shield sticker under board
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xEE };

EthernetServer server(80);  // create a server at port 80
```

Figure 2 Screenshot IV-1



```
void setup()
{
  Ethernet.begin(mac);
  // initialize Ethernet device
  server.begin();
  // start to listen for clients
  Serial.begin(9600);
  // for diagnostics

  mySerial.begin(9600);

}
```

Figure 3 Screenshot IV-2

F. Software Serial.h

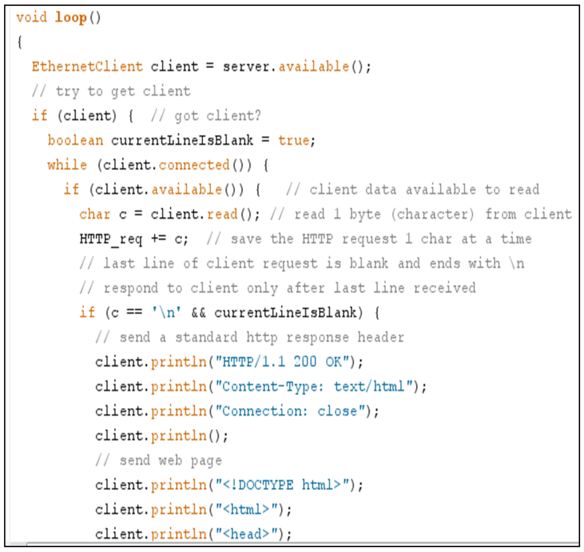
The Arduino hardware has built-in support for serial communication on pins 0 and 1 (which also goes to the computer via the USB connection). The native serial support happens via a piece of hardware (built into the chip) called a UART. This hardware allows the Atmega chip to receive serial communication even while working on other tasks, as long as there room in the 64 byte serial buffer. Here this library is used to communicate with the moving message display efficiently.

The SoftwareSerial library has been developed to allow serial communication on other digital pins of the Arduino, using software to replicate the functionality (hence the name "SoftwareSerial"). It is possible to have multiple software serial ports with speeds up to 115200 bps **[14].**

Using these library files the code is written using the predefined as well as user defined functions. The sketch is then verified and uploaded using the Arduino IDE which gives us any compilation error to be debugged beforehand.

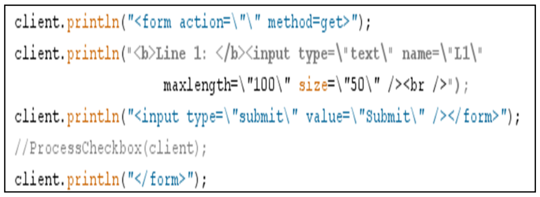
G. Understanding the Sketch

Before starting the setup function we include the header files (Figure 2) explained in the previous sub chapter namely SPI, Ethernet and SoftwareSerial.



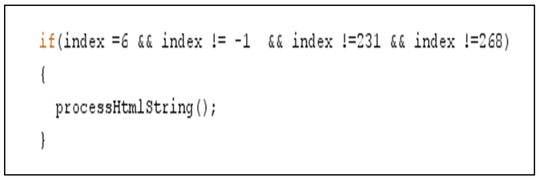
```
void loop()
{
  EthernetClient client = server.available();
  // try to get client
  if (client) {  // got client?
    boolean currentLineIsBlank = true;
    while (client.connected()) {
      if (client.available()) {   // client data available to read
        char c = client.read(); // read 1 byte (character) from client
        HTTP_req += c;  // save the HTTP request 1 char at a time
        // last line of client request is blank and ends with \n
        // respond to client only after last line received
        if (c == '\n' && currentLineIsBlank) {
          // send a standard http response header
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: close");
          client.println();
          // send web page
          client.println("<!DOCTYPE html>");
          client.println("<html>");
          client.println("<head>");
```

Figure 4 Screenshot IV-3



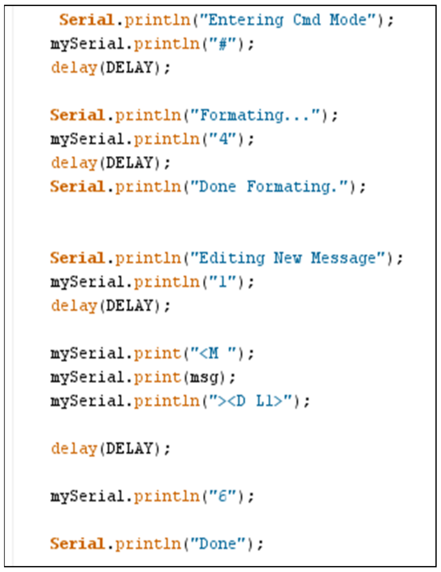
```
client.println("<form action=\"\" method=get>");
client.println("<b>Line 1: </b><input type=\"text\" name=\"L1\"
                    maxlength=\"100\" size=\"50\" /><br />");
client.println("<input type=\"submit\" value=\"Submit\" /></form>");
//ProcessCheckbox(client);
client.println("</form>");
```

Figure 6 Screenshot IV-4



```
  if(index =6 && index != -1  && index !=231 && index !=268)
  {
    processHtmlString();
  }
```

Figure 7 Screenshot IV-5



```
 Serial.println("Entering Cmd Mode");
mySerial.println("#");
delay(DELAY);

Serial.println("Formating...");
mySerial.println("4");
delay(DELAY);
Serial.println("Done Formating.");


Serial.println("Editing New Message");
mySerial.println("1");
delay(DELAY);

mySerial.print("<M ");
mySerial.print(msg);
mySerial.println("><D L1>");

delay(DELAY);

mySerial.println("6");

Serial.println("Done");
```

Figure 8 Screenshot IV-6

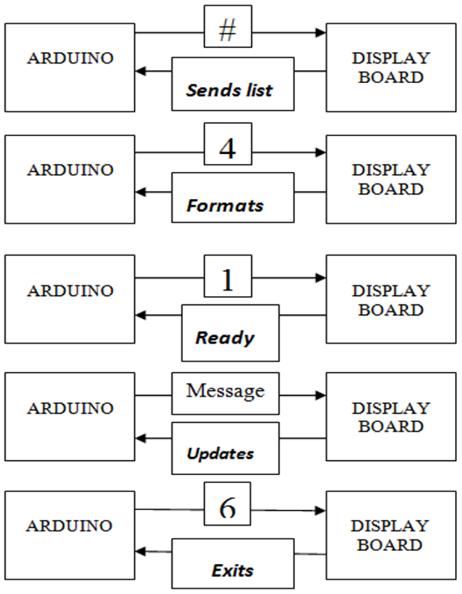


Figure 9 Flow Diagram of the two way communication between Arduino and Display

The code as stated earlier is divided into two parts: In the first part i.e. Setup () (Figure 3) a server is created using predefined functions from the libraries and give a serial command for diagnosis.
Now that everything is setup we move on to the loop () part
In the loop we first check for the availability of the client and send the terminal webpage to it which is written in html language. (Figure 4)
The terminal sent to the client receives the message from the user with the help of a html form and stores it in the variable “Line 1” (Figure 5)
After receiving the message we call (Figure 6) a user defined function namely “processHtmlString” in which we write the code where the data received is transferred to the Display board.
In the user defined function two ways communication between the Arduino Serial pins and the Display Board takes place. The code written here talks to the display board opens up its command window first by giving the command “#”, then the display sends back a list , the Arduino proceeds to formatting the memory of the display using the option “4” and later on updates the new message using “1” and exits the command window of the Display Board.(Figure 7)
Figure 8 shows the two way communication that is taking place between the Arduino and the display board.

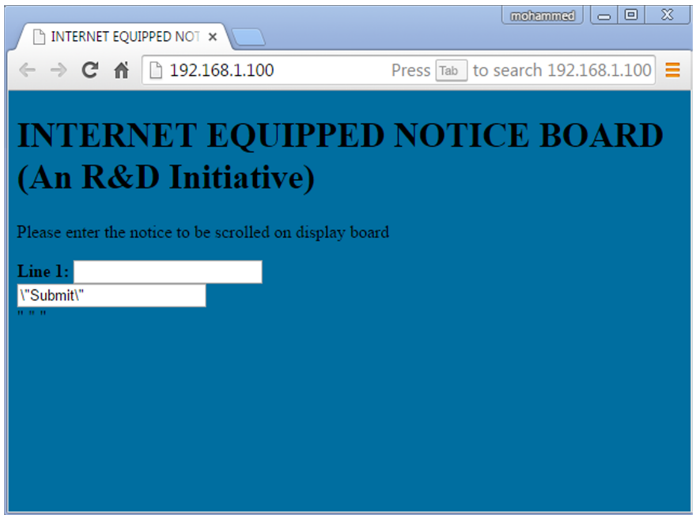
H. Accesing the Terminal

After uploading the sketch (program) into the Arduino the e-display is ready to use. The following steps need t be followed to access the board:

a) Switch on the display board .
b) Switch on the Wi-Fi of your Client device.

Connect to the Network created by the router using the known passkey.

a) Now open your Device’s browser.
b) Type in the assigned web address to access web terminal as shown in Figure. 9.
c) After the web terminal (as seen in Figure 9) loads.
d) Type in the message to be displayed and click Submit.
e) Check the message.



INTERNET EQUIPPED NOTICE BOARD (An R&D Initiative)

Please enter the notice to be scrolled on display board

Line 1:

\"Submit\"

" " "

Figure 10 The web terminal window

V. CONCLUSION

The model of the IoT based e display is efficiently developed. This model has the capability of receiving messages wirelessly from any authorized device on the local area network.It can be further enhanced to create a centralized notice board network for feasible communication

ACKNOWLEDGMENT

The authors would like to acknowledge the R & D MJCET for giving financial assistance for the implementation of the proposed work.

REFERENCES

[1] M.Baby, P.Harini, Y.Eleena Slesser, Y.Tejaswi, K.Ramajyothi, M.Sailaja and K.Annie Sumantha, “SMS Based Wireless E-Notice Board,” International Journal of Emerging Technology and Advanced Engineering, Vol 3 Issue3, pp181-185 , March 2013.
[2] Shruthi, K.; Chawla, H.; Bhaduri, A., "Smart notice board," Research & Technology in the Coming Decades (CRT 2013), National Conference on Challenges in , vol., no., pp.1,4, 27-28 Sept. 2013 doi: 10.1049/cp.2013.2512.
[3] http://en.wikipedia.org/wiki/Internet_of_Things
[4] M.Suresh , P.Saravana Kumar, Dr.T.V.P.Sundararajan, Data Security in IoT Environment, International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3297: 2007 Certified Organization) Vol. 3, Issue 4, April 2015.
[5] Divya Bahl, Gagangeet Singh Aujla, Enhanced Model Based Approach for Environmental Monitoring and Management Using Internet of Things, International Journal of Computing and Corporate Research ISSN (Online) : 2249-054X, Volume 5 Issue 2 March 2015.
[6] syslog.co.in/download.php?filename=ecembeddedsystem/...pdf
[7] http://portal.eng.asu.edu.eg/asusmartrobot/file/sbsoid57724dio/sbsfid11406dif.docx.
[8] http://www.arduino.cc/en/Main/ArduinoEthernetShield
[9] https://www.sparkfun.com/tutorials/215
[10] https://en.wikipedia.org/wiki/Wireless_router
[11] http://www.ngesc.com
[12] http://en.wikipedia.org/wiki/Arduino
[13] http://www.tp-link.us/faq-170.html
[14] http://www.arduino.cc/en/Reference