Study of Internet of Things Prototyping

[1]Vandana.C.P, [2]Priyanka.R, [3]Rakshitha.M, [4]Roshni, [5]Sarah Smruthi
[1]Senior Assistant Professor, [2]Final year student, [3]Final year student, [4]Final year student, [5]Final year student
Department of Information Science and Engineering,
New Horizon College of Engineering, Bangalore, India

Abstract: In this paper, integration of DHT11 and ECG sensors with Arduino board for IoT prototyping is presented. The pins and components of Arduino UNO are explained with the fundamentals on working with Arduino. The values from the integrated sensors are read and communicated through the Bluetooth module. The results and respective graphs are generated and displayed on the Serial Monitor of the IDE and on the application module based on the values read.

Keywords - Internet of Things; Sensors; Arduino

I. Introduction

The physical things can be embedded with sensors and software which enables them to communicate with each other and exchange data. This forms a network of physical devices known as Internet of Things (IoT)[1]. “Things” in IoT refers to variety of devices which collects useful data using existing technology and helps in flow of data between devices. The basic IoT architecture is as shown in Fig 1. Examples of physical devices are vehicles and home appliances.

The IoT creates opportunities for direct integration of the physical devices into the computer-based system. This allows the objects to be sensed or controlled remotely across the networks. The direct integration results in improved accuracy, efficiency and economic benefits along with reduction in human involvement. Nowadays, the IoT applications are including analytics and prediction models as an integral part of the systems. Due to the large number of links and interactions between different objects and modules IoT systems are considered as complex.

Arduino board is one of the platforms on which IoT can be implemented. Processing of measured data is done with the Arduino board [2]. Its designs use a variety of microprocessor and microcontroller. An embedded microprocessor [3] is a processor which functions like a computer’s CPU on small integrated circuit. A microcontroller is a single IC which functions like a small computer. ATmega328P is a type of microcontroller [4] present in the Arduino. It contains one or more CPUs, memory and input/output peripherals. These are used in devices such as implantable medical devices, remote controls and other embedded systems which can be automatically controlled. Microcontrollers are economical to digitally control more devices and processes. Hence, Arduino provide a low -cost solution. By 2020, experts have estimated that IoT will consist of around 13 billion objects. The value of the global market of IoT is also estimated to reach $7.1 million.

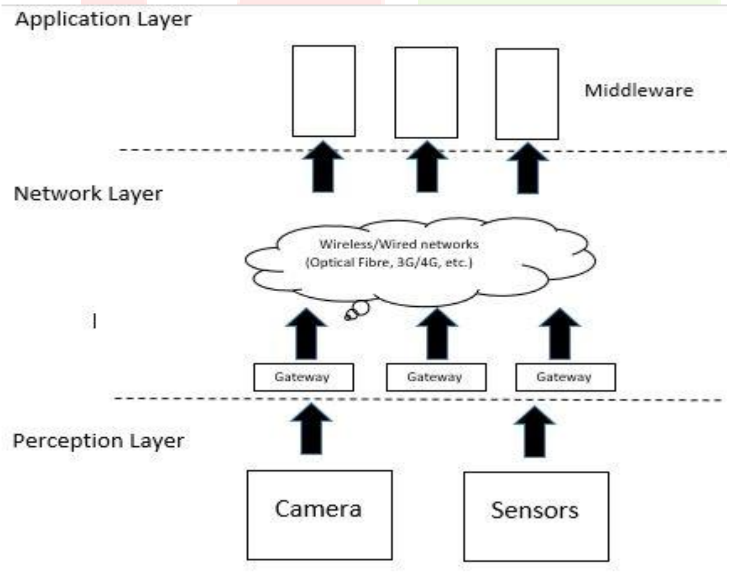


Fig 1: IoT Architecture

I. RELATED WORK

A. Arduino and IDE

Arduino is an open-source computer hardware/software platform for building an interactive network. The Arduino boards can be interfaced with other expansion boards (shields) and circuits through sets of digital and analog input/output pins.

The boards also have USB (Universal Serial Bus) which is a serial communication interface. This is used to load programs from the PC.

Few types of Arduino are :

1. Arduino Pro Mini
2. Arduino Uno
3. Arduino Mega

Arduino Pro Mini is the basic version. Arduino Mega is the advanced version which is used for industrial projects.

Arduino Uno is the intermediate version used for medium-scale projects. ATmega328 is used and it has 14 digital and 6 analog pins with 1 set of UART pins.

Description of pins:

Arduino UNO consists of 6 analog pins (A_0 to A_5). These pins are used if the module gives an analog output. The pulse range is 0-1023. It consists of 14 digital pins (0-13) [5]. Among the 14 pins, 6 pins are used as digital and PWM pins. It also has UART pins (RX0, TX1). It is used when the module gives a digital output. The range is 0-5(low to high). It consists of PWM pins where PWM stands for Pulse Width Modulation pins. On the Arduino board it is recognized by a tilde (~). These pins are used to vary the output. Example: varying the intensity of LED light and fan speed. Universal Asynchronous Receive Transmit (UART) pins also called as serial data are present. Frames of data are sent without any time consideration i.e., no request and acknowledgement frames sent. But the data loss cannot be managed. RX0 is the receiver pin and TX1 is transmit pin. Reset pin is a red button on the Arduino board which is used when code gets abruptly stopped. To reset the system, the button can be used or an instruction can be specified in the code. The Arduino UNO board also consists of few other pins. A 5V and 3.3 V pins are used to take the power from the board along with 3 GND pins.

Other Arduino Components include IC ATmega328 and a crystal oscillator. IC ATmega328 is a single chip microcontroller manufactured by Atmel. It is a 28 pin IC. The crystal oscillator generates the frequency of 16MHz. It provides a clock for the circuitry. Arduino needs a clock for its operation and crystal oscillator provides it. The speed (machine cycle) depends on crystal frequency. More is the operating frequency more is the power consumption. In Circuit Serial Programming it is also called as programmer pins. These pins are used to burn the program onto the board

7805 Voltage Regulator it is used to drop the power or voltage given to the IC. It mainly has 3 pins which includes input voltage, GND pin and output voltage pin. Heat Sink removes the extra voltage as heat.

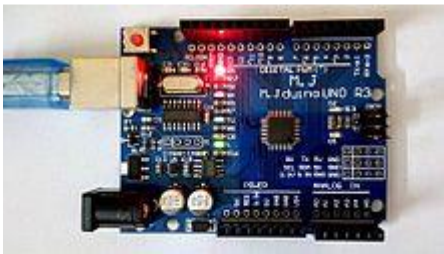
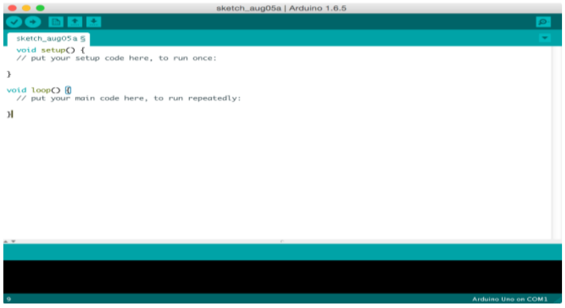


Fig 2 : Hardware

The Arduino Integrated Development Environment – or Arduino Software (IDE) is a software application which connects to the Arduino hardware. It facilitates programmers to upload programs and communicate with the hardware. The basic components are source code editor, build automation tools and debugger. This cross-platform application also includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic intending, brace matching, syntax highlighting and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus.



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Fig 3: Arduino IDE

B. Programming

A program for Arduino may be written in any programming language. The compilers used must produce binary machine code for the target processor. A sketch is defined as a program written in the IDE. Using special rules of code structuring, the C and C++ languages are supported by the IDE.

The Power LED (red) and User LED (green) is attached to Pin 13 on an Arduino compatible board as shown in Fig 2. Arduino IDE is an open source software which is available free of cost to download and test [6].

A minimal Arduino C/C++ program consists of only two functions as shown in Fig 3:

- setup(): This function is called only once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
- loop(): After setup() has been called, function loop() is executed iteratively.
- A typical program for a beginner to blink a LED repeatedly.

Example Program for a beginner to blink a LED repeatedly:

```
#define LED_PIN 13 //pin number attached to LED

void  setup()

{

pinMode(LED_PIN, OUTPUT);

// Configure pin 13 to be a digital output
}

void loop(){
digitalWrite(LED_PIN, HIGH);//turn on the LED

delay(1000);//wait 1 second

digitalWrite(LED_PIN, LOW);//turn off the LED
delay(1000);//wait 1 second
}
```

Fig 4

As shown in Fig 4, the digital pin 13 is defined as LED_PIN. It is set as the output using pinMode() function. The digitalWrite(LED_PIN,HIGH) function is used to turn ON the led. Similarly, the digitalWrite(LED_PIN,LOW) function is used to turn OFF the led. The delay(1000) function is included to introduce a delay of one second.

C. Simulators

IoT simulators are efficient tools to build and test large scale network of devices virtually. It is also cost effective.

There are many types of simulators. MATLAB includes an IoT module that allows the user to develop and test the devices. It also analyzes the collected data in the cloud. Patterns and algorithms are then extracted to create the prototype. Simulink is a graphical programming environment to develop algorithms, which are then deployed on the embedded hardware. MATLAB runs with a distributed server and toolbox program interface [7].

Iotify allows the user to develop large scale IoT solutions in the cloud. The user can also simulate heavy network traffic to check the overall system performance.

IBM's BlueMix is an innovative cloud platform that has built-in web console dashboards that in turn lets the user to monitor and analyze the data. This can be used to build and optimize own apps. Monitored parameters will be uploaded to a cloud server utilizing IBM BlueMix applications [8].

D. Web Services

A web service is a software system to support interaction between machines in a network. It is also a communication method that allows two software systems to exchange data over the internet.

The software system that requests data is called a service requestor. The software system that would process the request and provide the data is called a service provider.

Node.js is an open-source, cross-platform JavaScript runtime environment for executing JavaScript code server-side. NodeJS runs the program with single-threaded scheduling with event-driven mode [9].

Node.js handles a file request in the following way :

1. Sends the task to the computer's file system.
2. Ready to handle the next request.
3. When the file system has opened and read the file, the server returns the content to the client.

Node.js, waiting is eliminated as it continues with the next request. It is used to generate dynamic page content. It allows creating, opening, reading, writing, deleting and closing files on the server. It can also collect form data and add, delete, modify data in the database.

Node.js files have extension ".js". The working of NodeJS is as shown in Fig 5.

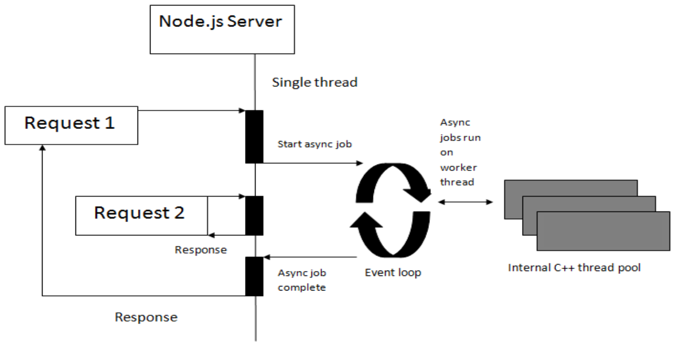


Fig 5: Node.js working

III. PROTOTYPE

In this prototype, two sensors and a Bluetooth module are integrated with Arduino UNO board.

The two sensors are DHT11 (temperature and humidity) and ECG module. The Bluetooth module considered here is HC-05 [10].

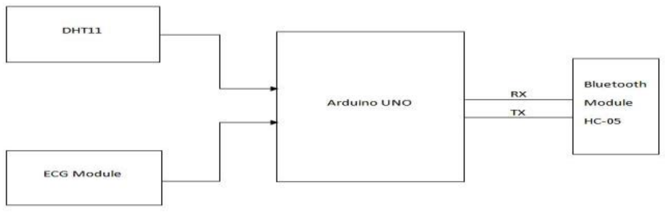


Fig 6: Basic Block Diagram of the Integration

The sensors and the Bluetooth module are connected as shown in Fig 6. The pin connections for DHT11, ECG module and Bluetooth module are given in Table 1, Table 2 and Table 3 respectively. The actual implementation of the prototype is as shown in Fig 7.

Pin connections:

Arduino UNO pins	DHT11 pins
+5V	Positive pin
GND	Negative pin
Digital pin 8	Output pin

Table 1

Arduino UNO pins	ECG module
+5V	+5V
GND	GND
A_0	Output pin
Digital pin 6	LO-
Digital pin 7	LO+

Table 2

Arduino UNO pins	Bluetooth module HC-05
+5V	+5V
GND	GND
Digital pin 11	TX
Digital pin 10	RX

Table 3

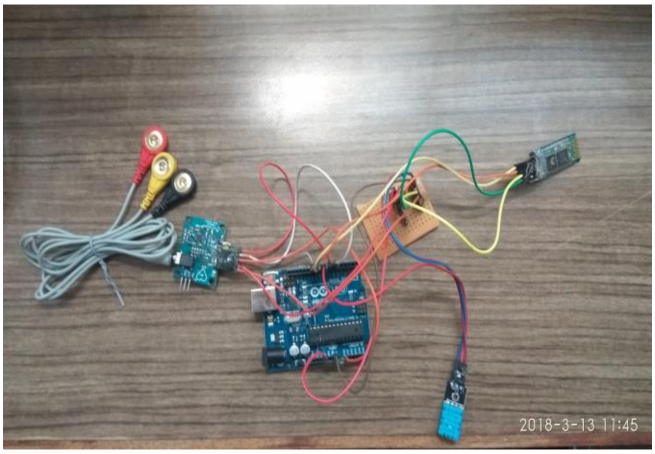


Fig 7: Integrated Prototype

The DHT11 sensor is based on I2C (Inter Integrated Circuit)[11] communication. The communication is between a master and multiple slave devices. It uses only 2 wires for communication and almost 128 devices can be connected. It is a complex network and hence slow.

The ECG sensor uses an insulated electrode which can measure ECG through an insulator [12]. The ECG module has 3 sensor pads which have to be kept in contact with the user’s body for accurate readings. The black cable is kept on the right arm. The blue cable on the left arm. The red cable on the right leg. This sensor enables ECG readings to be taken remotely[13].

The code shown in the Fig 8 should be uploaded to the Arduino IDE. The temperature, humidity and ECG values from the sensors are read. The dht.readHumidity() and dht.readTemperature() functions compute the humidity in percentage and temperature in degrees Celsius respectively. The analogRead(A0) function reads the ECG signal.

The baud rate is defined as the rate at which information is transferred in a communication channel. In Fig 7, the baud rate is 9600. In the serial port, 9600 baud rate means a maximum of 9600 bits per second can be transferred.

Fig 8 : Code

```
#include <SoftwareSerial.h>
#include <DHT.h>
#define DHTPIN 8
// digital pin 8 is connected to signal (out) pin

#define BT_SERIAL_RX 11
#define BT_SERIAL_TX 10

SoftwareSerial BluetoothSerial(BT_SERIAL_RX,
BT_SERIAL_TX);

int ecgSignal=0;

DHT dht(DHTPIN, DHT11,15);
float h;
float t;

void setup()
 {
  // initialize the serial communication:
  Serial.begin(9600);
  dht.begin();
  BluetoothSerial.begin(9600);

  pinMode(7, INPUT); // Setup for leads off detection LO +
  pinMode(6, INPUT); // Setup for leads off detection LO -
}
void loop()
{
  h = dht.readHumidity();
  t = dht.readTemperature();
  if (isnan(h) || isnan(t))
    {
       Serial.println("Failed to read from DHT sensor!");
    }
  if((digitalRead(6) == 1)||(digitalRead(7) == 1))
   {
      BluetoothSerial.println('!');
    }
  else
{
    // send the value of analog input 0:
// Serial.println(analogRead(A0));
      ecgSignal = analogRead(A0);
      sendThroughUart();
 }
  //Wait for a bit to keep serial data from saturating
     delay(10);
}

void sendThroughUart()
{
  BluetoothSerial.print("E");
  BluetoothSerial.print(h);
  BluetoothSerial.print(',');
  BluetoothSerial.print(t);
  BluetoothSerial.print(",");
  BluetoothSerial.print(ecgSignal);
  BluetoothSerial.print("\n");
  Serial.println(h);
  Serial.println(t);
}
```

The values read from the sensors are sent through the Bluetooth module to the mobile application. HC-05 is the Bluetooth module used here. It communicates with the microcontrollers using serial communication (UART). The default settings of HC-05 can be changed using AT (Attention) commands[14]. Few AT commands are:

1. AT – Testing Connection Command
2. AT+NAME? – Set Name
3. AT+UART? – Set Communication

In the mobile application, the digital values and the graphs are displayed based on user's choice.

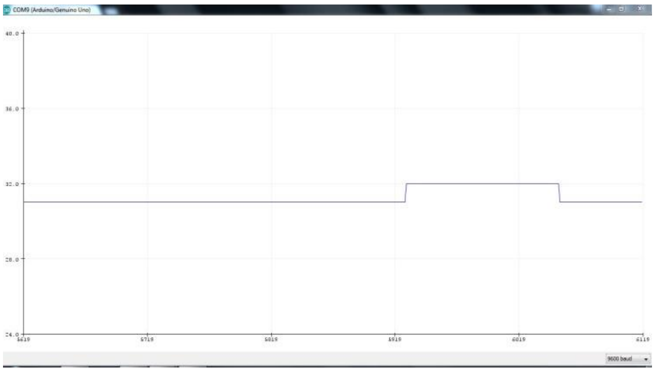


Fig 9: Temperature Graph

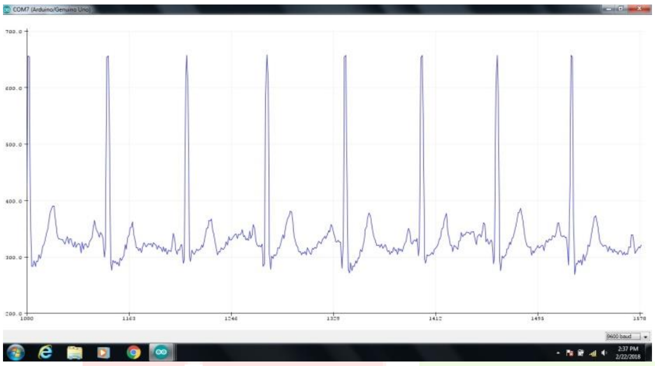


Fig 10: ECG Graph

The values are taken every one millisecond. The respective output graphs are shown in Fig 9 and Fig 10. In the experiment performed, the temperatures is constant until there is a variation in the environment. The appropriate variation is observed for that particular period of time as shown in Fig 9. In Fig 10, the highest peak is called the QRS complex. The heart beat is measured based on how many QRS complex pass in one minute. It is expressed in beats per minute (bpm)[15].

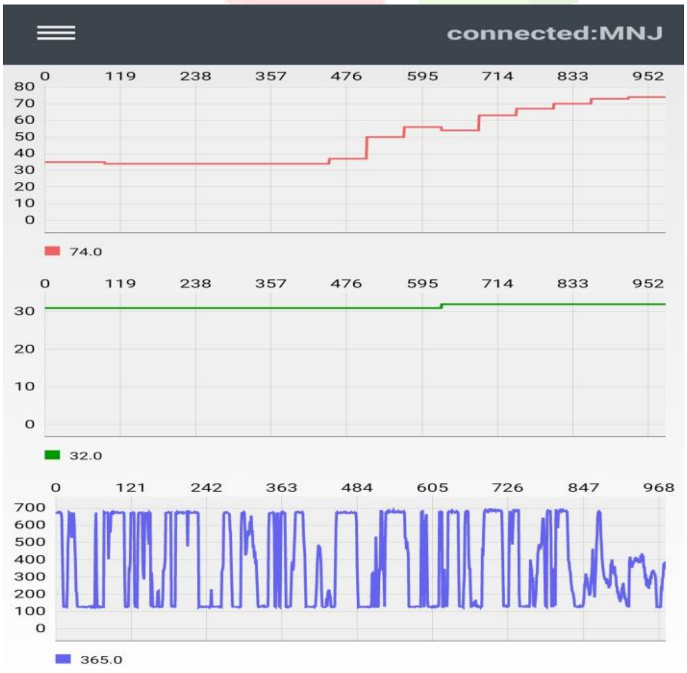


Fig 11: Snapshot of demo application

The Bluetooth module is named as MNJ using the AT+NAME=MNJ command. When the multiple graphs option is chosen in the demo application, all three graphs (humidity, temperature, ECG) will be displayed as shown in Fig 11.

IV. CONCLUSION

We presented a prototype consisting of only 2 sensors and a Bluetooth module. Here, the values obtained from the sensors are sent through the Bluetooth module to the display. The corresponding graphs are also generated and displayed. These graphs are used to infer information about the results.

V. FUTURE WORK

The future work can include integration of more sensors and modules like pulse sensor, body temperature sensor and Wi-Fi module for continuous monitoring of the user's health. Since health is also affected by indoor air pollution, sensors for CO and CO2 can be included to monitor the pollution levels. All the data collected can be further tabulated and stored on the cloud. This data can be analyzed to give predictions about future health conditions. More sophisticated data analysis could be performed which can be used in healthcare domain.

VI. REFERENCES

[1] IoT device management architecture based on proxy, 2017 6th International conference on computer science and Network technology (ICCSNT)

[2] A Non-Invasive Electricity Measurement within the Smart Grid Landscape: Arduino based Visualization platform for IoT, 2017 9th International Congress on Ultra-Modern Telecommunications and Control systems and Workshops (ICUMT)

[3] Internet of Things: Connecting the Physical and Digital Worlds

[4] Implementation of Microcontroller Based Driver Assistance and Vehicle Safety Monitoring System, 2015 International Conference on Information Processing (ICIP)

[5] An IoT Based Human Healthcare System using Arduino UNO board, 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)

[6] A Review of Arduino Board's, Lilypad's & Arduino shields

[7] A Distributed Artificial Immune Network for Optimizing Tracer Kinetic Models with MATLAB Distributed Computing Engine, 2011 10th International Symposium on Distributed Computing and Applications to Business, Engineering and Science

[8] Applied Internet of Things (IoT): Car Monitoring System using IBM BlueMix, 2016 International Seminar on Intelligent Technology and its Application

[9] The Communication System between Web Application Host Computers and Embedded Systems Based on Node.JS, 2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI 2017)

[10] Human Health Monitoring System Based on Piezoelectric Smart Sensor

[11] Design and Development of Daughter Board for Raspberry Pi to support Bluetooth Communication using UART, International Conference on Computing, Communication and Automation (ICCCA2015)

[12] Capacitively Coupled ECG Sensor System with Digitally Assisted Noise Cancellation for Wearable Application

[13] ECG based Authentication for Remote Patient Monitoring in IoT by Wavelets and Template Matching

[14] An IoT Application-Layer Protocol Modem: A Case Study on Interfacing IEEE 1888 with AT Commands, 2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery

[15] Real Time ECG monitoring system based on Internet of Things (IoT), International Journal of Scientific and Research Publications, Volume 7, Issue 8, August 2017