2019

Design of an Arduino Based Smart Home System

Qi Lu
University of Northern Iowa

Design of an Arduino Based Smart Home System

Abstract

Smart home is a home setup where smart appliances can be controlled remotely using networked devices. It has gained widely attentions due to its flexible integrations. However, not everyone is willing to upgrade their old home appliances to smart ones. One alternative solution is to design and implement a wireless controlled smart system that could utilize existing home appliances. This project aims to design an Arduino based smart home system. The system may upgrade existing appliances into smart home devices by integrating the wireless capability through XBee modules and low-cost Arduino boards.

Design of an Arduino Based Smart Home System

A Graduate Research Paper

Presented to the Graduate Faculty

of the

Department of Technology

University of Northern Iowa

In Partial Fulfillment of the Requirements

For the

Non-Thesis Master of Science in Technology Degree

by

Qi Lu

04/26/2019

Approved by:

Jin Zhu

4/29/2019

Signature of Advisor **Date**

5/7/2019

Signature of Second Reviewer **Date**

Abstract

Smart home is a home setup where smart appliances can be controlled remotely using networked devices. It has gained widely attentions due to its flexible integrations. However, not everyone is willing to upgrade their old home appliances to smart ones. One alternative solution is to design and implement a wireless controlled smart system that could utilize existing home appliances. This project aims to design an Arduino based smart home system. The system may upgrade existing appliances into smart home devices by integrating the wireless capability through XBee modules and low-cost Arduino boards.

Keywords: Arduino, XBee, smart home, wireless sensors

Introduction

The Internet of thing (IoT) could be described as a network of devices and sensors, where they are intelligently connected to Internet, enabling communication between electronic devices. IoT has been developing rapidly in recent years in various areas, including smart grid, smart home, smart metering, smart city, wearables devices, etc. Among them, smart home stands out as the most prominent application of IoT (Mohan, 2016). The smart home system may monitor home environment variables, such as temperature, humidity, light level, and remote control the appliances.

However, smart home has not been widely deployed yet. There are some reasons such as high device price, customers reluctance to adopt the newest technology, and technology to establish the infrastructure of smart home still under development. Table 1 shows the cost of some humidifiers available online with four stars or better average customer reviews. It can be seen that the price of smart devices is much higher than non-smart, traditional devices. That means customers usually need to spend much more money if they want to replace existing appliances with smart ones.

Table 1

Price of Smart Home Humidifier

Product	Brand	Platform	Price	Smart appliance or not
5L Cool Mist Ultrasonic Humidifier	Pallas	Amazon.com	$49.99	N
Top Fill Humidifier	Mooka	Amazon.com	$72.49	N
UC5501 Ultrasonic Humidifier 6L	Elexhomes	Amazon.com	$109.99	Y
Smart Ultrasonic Humidifier	Elite	bedbathandbeyong.com	$135.99	Y
AM10 Humidifier	Dyson	Amazon.com	$490	Y

One possible alternative solution is to design and implement an easily setup, wireless controlled smart system that is able to utilize the existing home appliances. This project demonstrated an implemented system that uses Arduino UNO microcontroller and XBee wireless communication boards to remote control a pre-owned home appliance such as a humidifier. The proposed system is easy to set up and can be adapted to different types of appliances.

Rational

As mentioned in the introduction, the cost of replacing traditional home appliances with smart devices could be expensive. To address this issue, a smart home system utilizing the existing home appliances has been proposed and implemented in this project. The objective of this project is to determine the suitable components and design, and to demonstrate the functionality of the system.

Literature Review

The Internet of Things (IoT), also called the Industrial Internet, is acknowledged as one of the most important technologies in the future. It is considered as a worldwide network where devices and machines could interact with each other (Lee & Lee, 2015). IoT has been in development and growth rapidly over the last ten years. Ratasuk et al. (2016) mentioned that there were about 0.4 billion devices connected to Internet using cellular network in 2015. This number is predicted to increase to 1.5 billion by the year 2021 with 27% yearly growth rate. As a forefront of innovation, smart home has become a major player in IoT appliances. According to Biljana (2016), smart home refers to the use of information and communication technologies (ICT) in home control, ranging from controlling appliances to automation of home features (temperature, lighting, etc.). The word "smart" here means intelligence and has been used in different areas (Yang et al., 2018). As one of the most popular IoT applications, smart home is changing our daily life. It can lead to energy efficiency, energy savings, and overall better quality of life (Sabry et.al., 2017). Take robotic vacuum cleaner as an example, people can control the robot to clean the house wherever they are and whenever they want by using mobile application

on their cellphones. When the home environment and facilities are connected through wireless networks, environment variables can be monitored conveniently and the equipment such as humidifiers, fans, and lights, can be remotely controlled based on the environment change. Smart home also helps users to enhance security (Charlie et.al, 2017). People can lock room's door and check house's security by using home security system while on vacation. Smart home offers installation and cost benefits for homeowners and installers and adds new markets and opportunity for dealers (Klotz-Young, 2013).

There are various wireless technologies available for smart home, including Wi-Fi, ZigBee, and Bluetooth. ZigBee, a low-cost and mesh-route capable wireless networking framework, can improve reliability while reducing the cost of wireless communication (Jiang et.al, 2017). Compared to Wi-Fi and Bluetooth, ZigBee network is cheaper and easier to use. ZigBee is based on the IEEE 802.15.4 standard. As shown in Table 2, ZigBee is a better choice for smart home due to many advantages over other wireless technologies, including lower cost, lower power consumption, and larger network size (Varga, 2014).

Table 2

Wireless Network Comparison (Varga, 2014)

Aspects	ZigBee	WiFi	Bluetooth
standard	802.15.4	802.11a,b,g	802.15.1
application	automation, control	Web, e-mail, video	replacement of cableing
data rate	50 - 60 kbyte	> 1 Mbyte	> 250 kbyte
battery lifetime	> 1000	1 -5	1 -7
network size	65535	32	7
bandwith (kb/s)	20 - 250	11000	720
transmission distance	100+ m	100 m	10 m
advantage	reliability, performance, cost	data rate, flexibility	cost, comfortable

Design Methodology

System Architecture

The proposed smart home system includes two modules: transmitter module and receiver module. The basic framework of the system is shown in Figure 1. The wireless module XBee S2C was installed in each of them, served as a ZigBee coordinator or a ZigBee router for communications. Data are exchanged between the coordinator and the router. There are three basic functions in this system, remote monitoring, control, and providing alert to users. If a sensor detects an abnormal situation, the transmitter module will trigger the alarm system and send the corresponding control command to the receiver module. Once the receiver module receives the data, it turns on or off the corresponding devices.

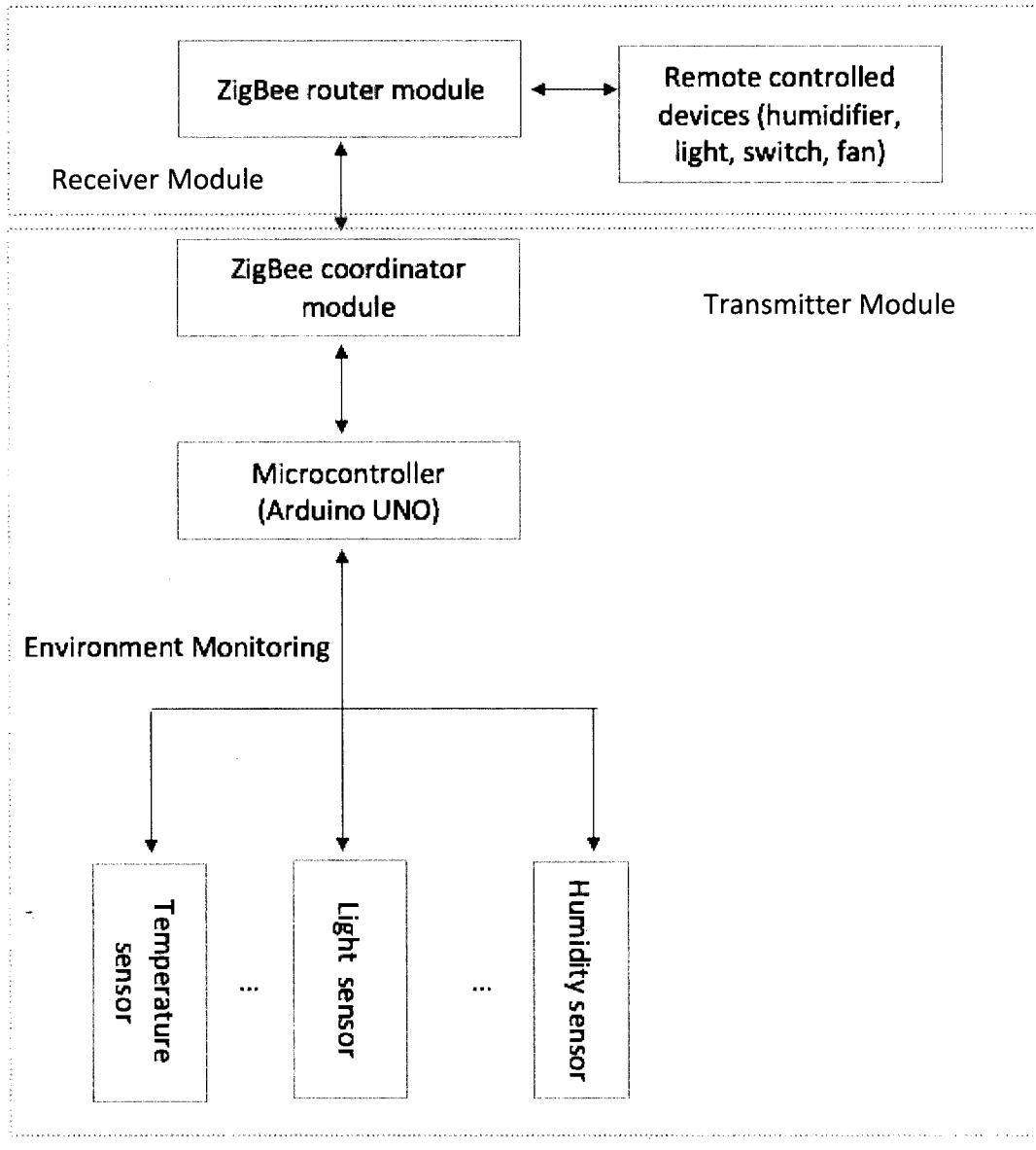

Figure 1. Framework of the Proposed System

Components Selection

Microcontroller

The Arduino UNO is an open-source microcontroller board developed by Arduino.cc ("Arduino UNO Rev3," n.d.). It is based on the Microchip ATmega328P microcontroller, a single-chip microcontroller in the megaAVR family. It has fourteen digital input/output pins (six of them can be used as PWM outputs) and six analog inputs.

XBee Module

XBee is a product of ZigBee compliant radios made by Digi International. They often ship with some enhancements to the software stack that makes them easier to use out of the box for small projects ("Digi XBee ZigBee," n.d.). It has 16 direct sequence channels, 4 analog input pins, and 20 digital input/output pins. It uses serial communication to interface with Arduino.

Temperature & Humidity Sensor (DHT22)

The DHT22 is a basic digital temperature and humidity sensor. A capacitive humidity sensor and a thermistor are used in DHT22. Ambient temperature and humidity are measured and converted to digital signal. Data are sent to MCU via single-bus in a 40-bit packet in the following format: 8 bit integral data, 8 bit decimal RH data, 8 bit integral T data, 8 bit decimal T data, and 8 bit check-sum.

Relay

Grove-Relay v1.2 is used in this project. Table 3 shows the basic information of the relay. This is a digital normally-open switch. By using relay, high voltage circuit can be controlled with low voltage (5V in this project). There is an indicator LED on the relay that will lights up when the controlled terminals get closed.

Table 3

Product Information of Grove-Relay v1.2 Relay

Product Release Date	9th June 2014
Operating Voltage	3.3V~5V
Operating Current	100mA
Relay Life	100,000 Cycle
Max Switching Voltage	250VAC/30VDC
Max Switching Current	5A

Hardware Design

Transmitter Module

Figure 2 shows the connection diagram of the transmitter module. In transmitter module, a DHT22 sensor is set up to monitor the temperature and humidity in the room. DHT22 sensor sends the updated temperature and humidity information to Arduino board

every second. Once the detected temperature is abnormal, for example, higher than a preconfigured value, the alarm on the board rings and the LCD screen displays "temperature abnormal". This module can monitor humidity as well. Once the humidity is lower than the threshold set up by a user, LCD screen shows "Humd on". Meanwhile, the transmitter module sends out the trigger signal to the receiver module through XBee board.

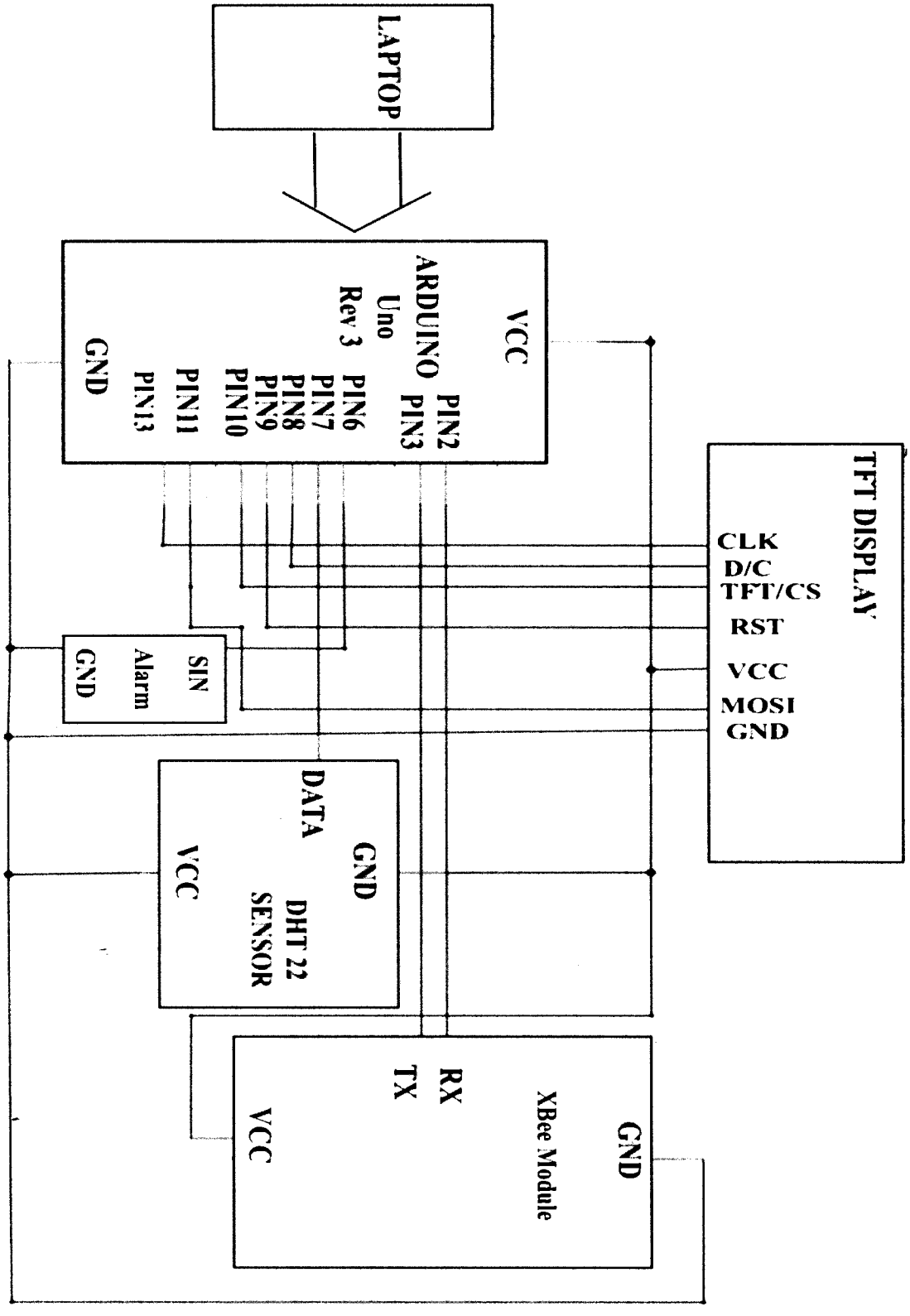

Figure 2. Connection Diagram of the Transmitter Module

Receiver Module

Figure 3 shows the connection diagram of the receiver module. It is powered by a 9V battery. Once the receiver module receives the turn-on signal, it turns on the relay connected to the humidifier, and the humidifier starts. Once the room humidity is higher than setting value, Arduino sends out turn off signal. XBee receiver module then turns off the relay when the signal is received.

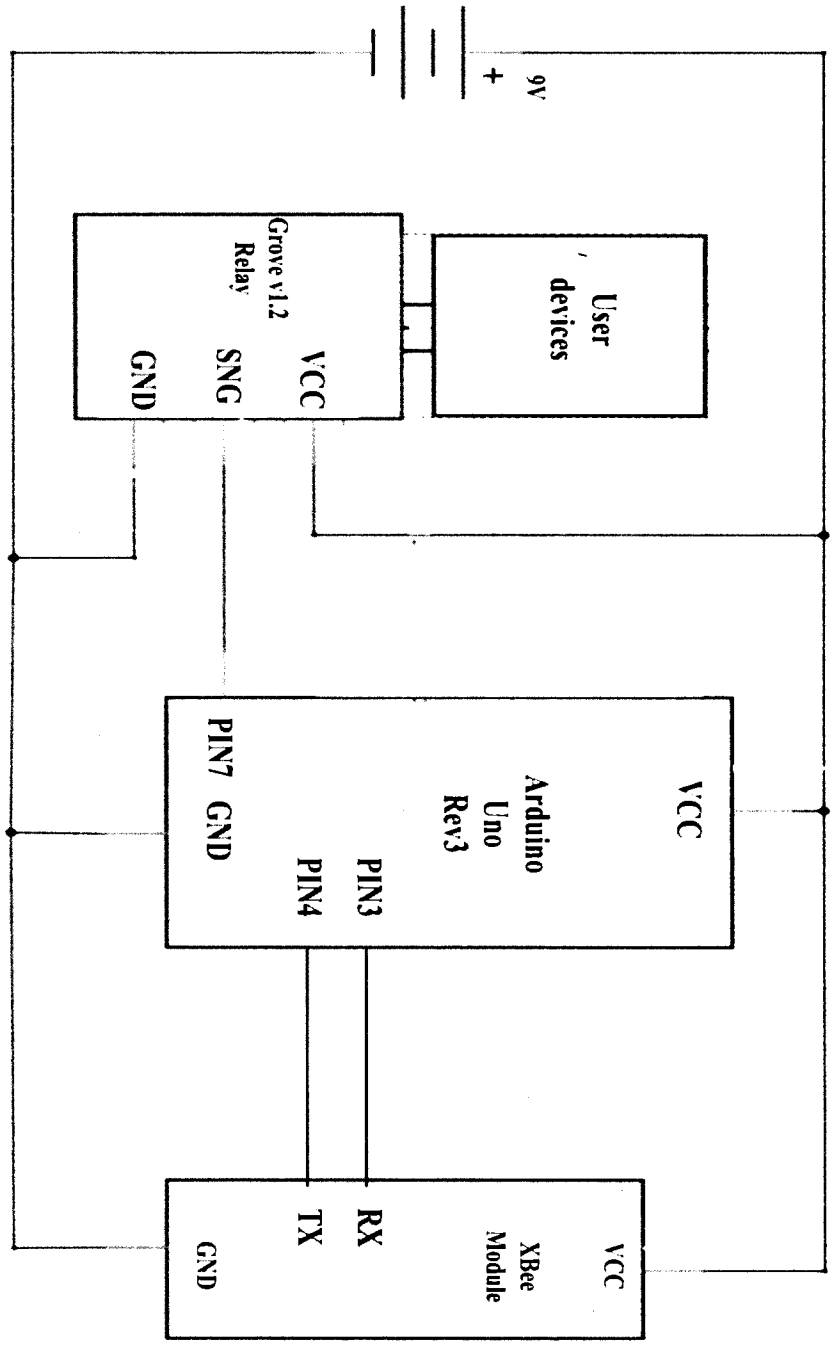

Figure 3. Connection Diagram of the Receiver Module

Software Design

Integrated Development Environment

This project is implemented using Arduino IDE and Microsoft Visual Studio.

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux). Arduino IDE is used to write and upload programs to Arduino boards and other third-party cores, and boards ("Arduino Software (IDE)," n.d.).

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, websites, web applications, and mobile apps as well. Visual Studio supports 36 different programming languages, and C# is used in this program ("Visual Studio," n.d.).

Software Implementation

One example application was implemented in this project to monitor and control temperature and humidity. Flowcharts are shown in Figure 4 and Figure 5. The source code for the implements is included in Appendix B.

Figure 4 shows the transmitter module flow chart. The transmitter module keeps obtaining temperature and humidity data from DHT22 sensor. When the obtained temperature value is above a threshold, the transmitter module sends an alarm Email to the user. In addition, once the detected humidity value is lower than the threshold, the transmitter module sends turn-on signal to the receiver module and an alarm email to the user at the same time.

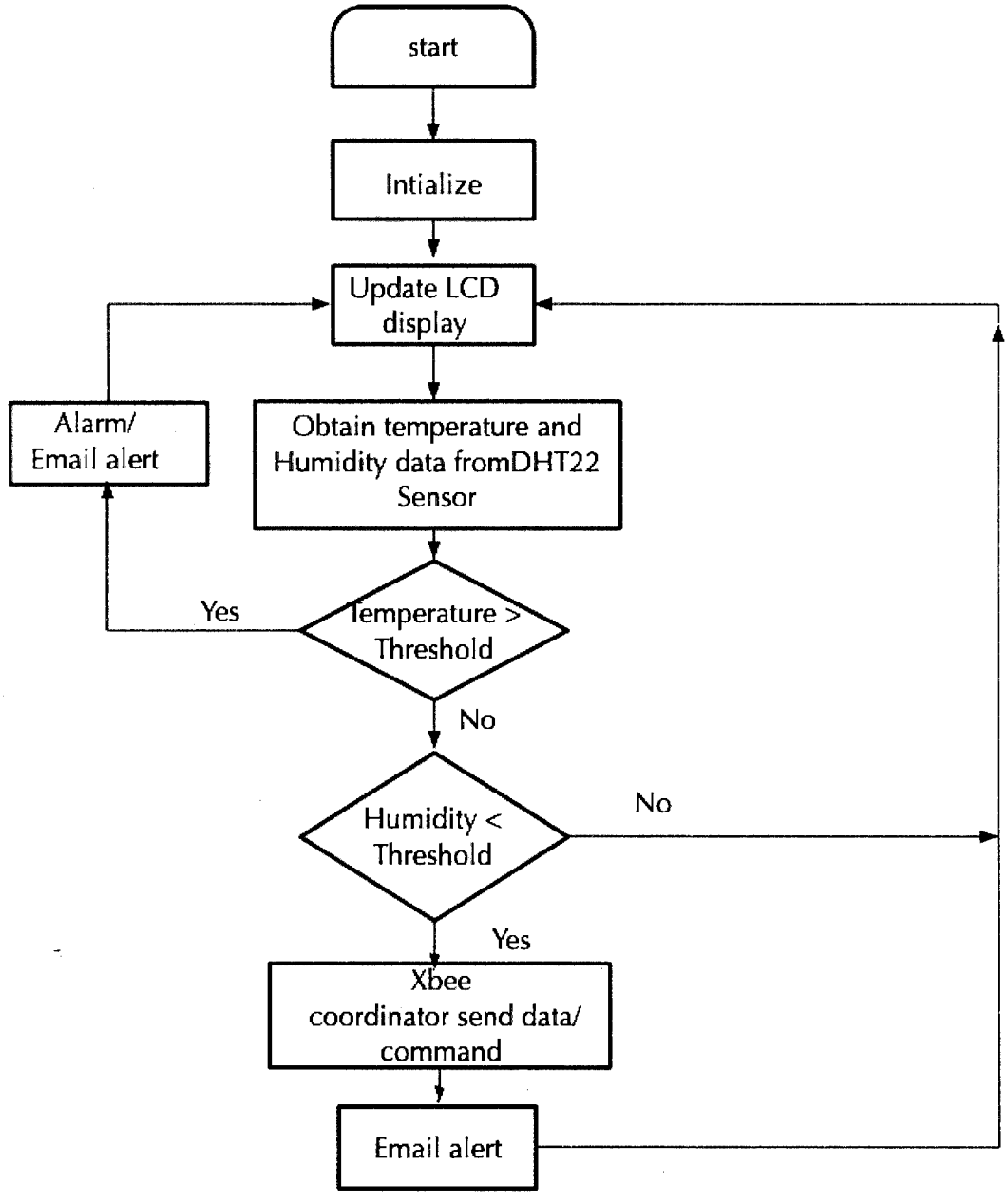

Figure 4. Transmitter Module Flow Chart

Figure 5 shows the receiver module flow chart. Receiver module keeps checking the record data from the coordinator. When the receiver module receives control signal from the transmitter module, it turns on or off the relay accordingly.

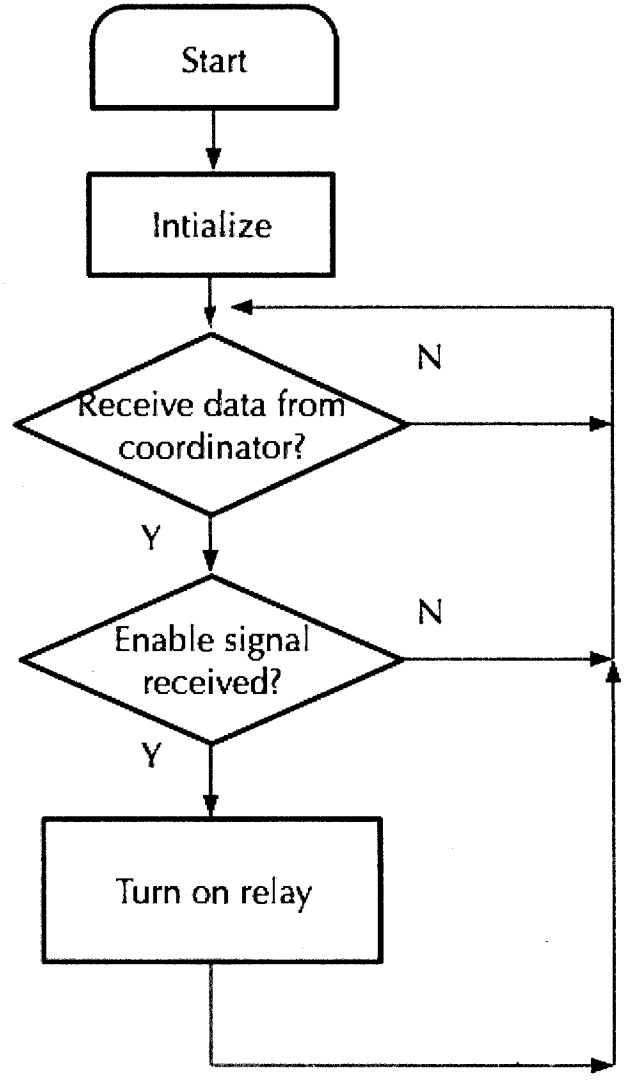

Figure 5. Receiver Module Flow Chart

Results and Discussion

Prototype

A prototype of the proposed smart home system was developed and implemented. Figure 6 and Figure 7 show the prototype and the setup of the system.

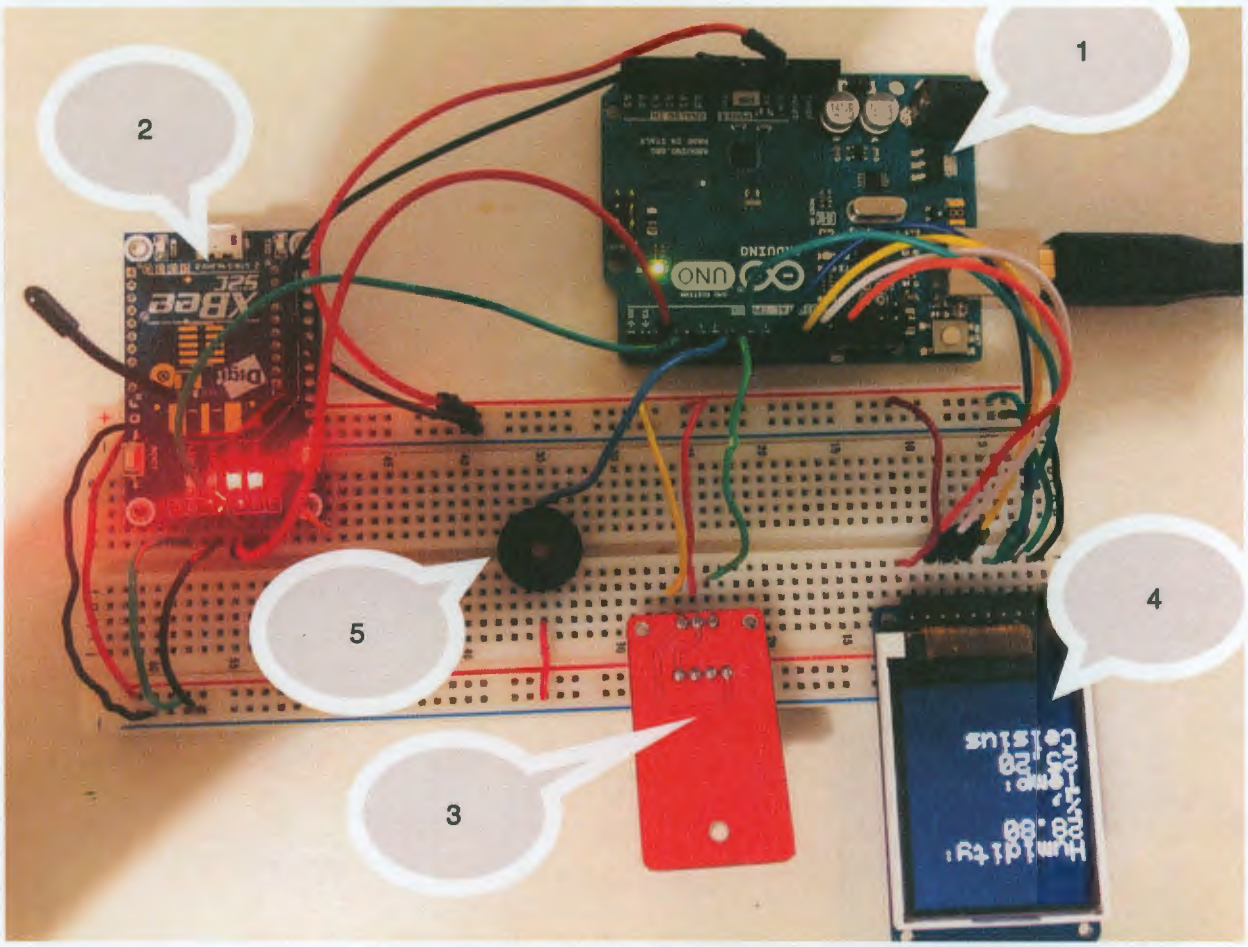

Figure 6. Prototype of the Transmitter Module:

(1) Arduino UNO Board, (2) XBee Coordinator,

(3) DHT 22 Sensor, (4)1.8'' TFT Display, (5) Buzzer

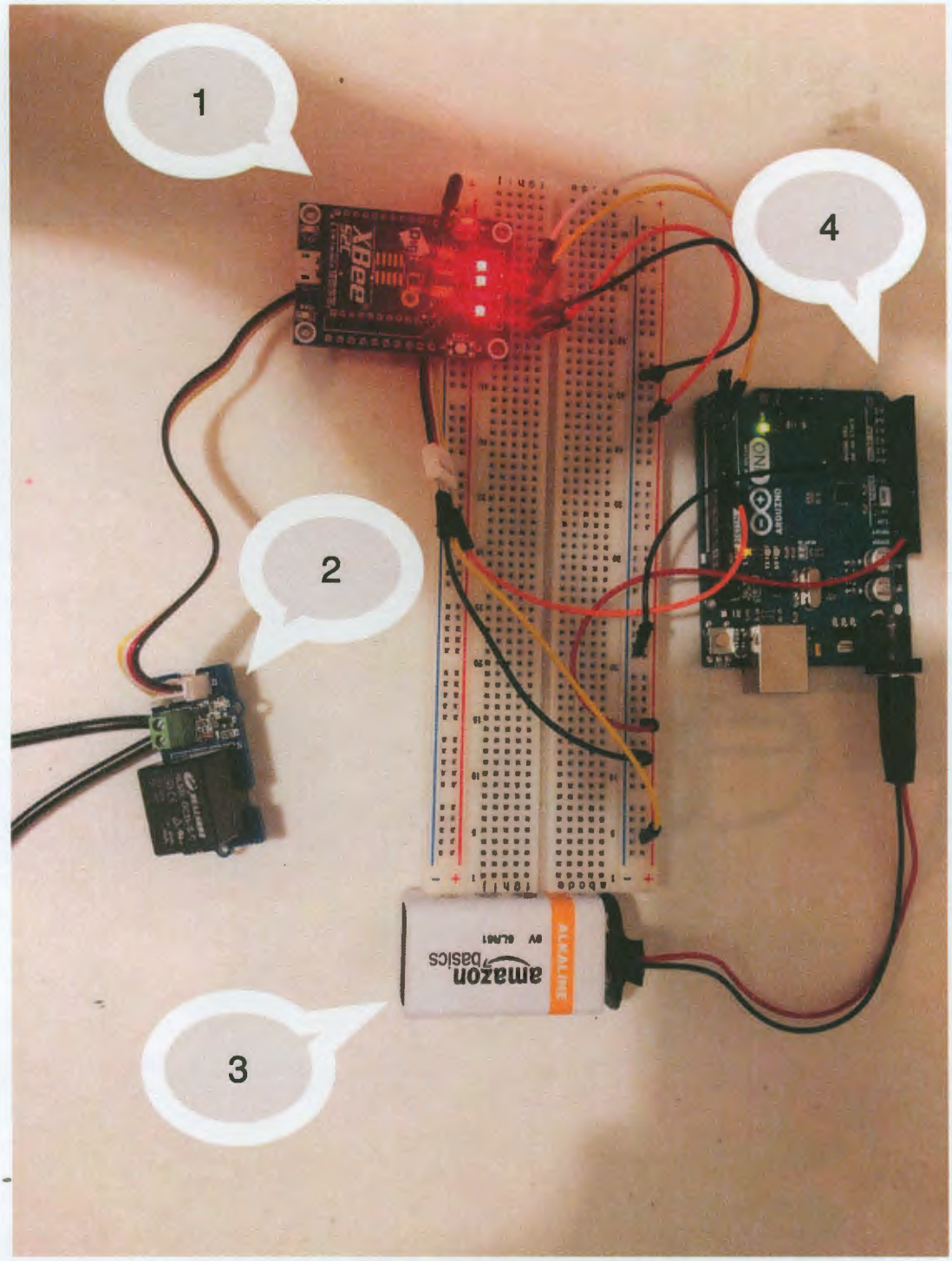

Figure 7. Prototype of the Receiver Module:

(1) XBee Router, (2) Grove-Relay v1.2 Relay,

(3) 9V Power Supply, (4) Arduino UNO Board

Results

Figure 8 shows one example result of the temperature and humidity monitoring and corresponding LED display. In this example, LCD screen displayed the real-time humidity and temperature. Humidity threshold was set as 35% and temperature threshold was set as 30°C. Once the detected humidity was lower than 35%, transmitter module sent turn-on signal to the receiver module through XBee. LCD displayed "Humd on" and the system sent an alarm email to the user. When the detected temperature is higher than 30°C, LCD displays "temperature abnormal" and the buzzer will ring.

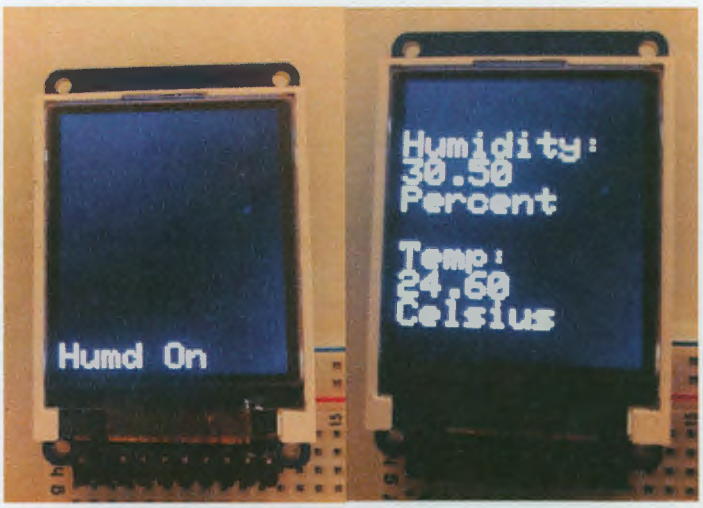

Figure 8. Temperature and Humidity Monitoring

The humidity alert email is shown in Figure 9. After the receiver module received the turn-on signal, it sent "HIGH" to the relay and the humidifier started running. When the humidity is higher than 50%, the transmitter module sends a signal to the receiver module to turn off the humidifier.

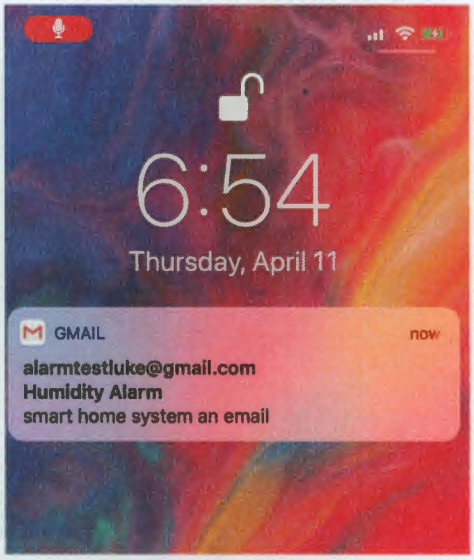

Figure 9. Humidity Alarm Email

Limitations and Future Improvement

This proposed smart home system can be adapted to work with other appliances and sensors. Current system has some limitations. Although Arduino board is open source and easy to interface with other circuits, the cost is not low enough. Appendix A shows the cost of the entire system. For the future development, the cost of the system could be reduced by using a single Arduino board to control multiple devices. More home appliances can be added into the system and the performance could be tested when multiple devices are remotely controlled by the users at the same time.

Network security is also an issue to be addressed. In current design, there is no security measure. To ensure network security, a network key could be added in this project in the future.

Conclusion

As one of the most popular IoT applications, smart home is changing people's daily life. It enables home appliances to be more intelligent, and remote controllable. However, home devices are still relatively expensive. This project demonstrated an implemented system that transfers regular house appliance to a wireless, automatic controlled appliance. There are two parts in this system: transmitter module and receiver module. XBee boards are used in each module to exchange trigger signals. DHT22 sensor is used to monitor the environment feature. The system sends an alarm to the user and automatically control the home appliance based on the temperature and humidity condition. It provides a convenient way to implement a low-cost smart home.

References

Arduino UNO Rev3. (n.d.). Retrieved April 26, 2019, from https://www.arduino.cc/

Arduino Software (IDE). (n.d.). Retrieved April 26, 2019, from https://www.arduino.cc/

Biljana, L., Risteska, S., & Kire, V. (2017). A review of Internet of Things for smart home: Challenges and solutions, *Journal of Cleaner Production, 140*, 1454-1464. https://doi.org/10.1016/j.jclepro.2016.10.006

Charlie, W., Tom, H., & Richard, H. B. (2017). Benefits and risks of smart home technologies, *Energy Policy, 103,* 72-83. https://doi.org/10.1016/j.enpol.2016.12.047

Jiang, D., Yu, L., Wang, F., Xie, X., & Yu, Y. (2017) Design of the smart home system based on the optimal routing algorithm and ZigBee network. *Public Library of Science ONE, 12*, 11. http://dx.doi.org.proxy.lib.uni.edu/10.1371/journal.pone.0188026

Klotz-Young, H. (2013). Modern home access. *Security Distributing & Marketing, 43(10),* 79-82.

Lee, I., Lee, K. (2015). The Internet of Things (IoT): Applications, investments, and challenges for enterprises. *Science Direct, 58(4),* 431-440. https://doi.org/10.1016/j.bushor.2015.03.008

Mohan, P., Thangavel, M., & Saravanan, V. (2016). IOT based home appliances monitoring using Arduino kit. *Advances in Natural and Applied Sciences, 10(3)*, 63-72.

Ratasuk, R., Mangalvedhe, N., Zhang, Y., Robert, M., & Koskinen, J.-P. (2016). Overview of narrowband IoT in LTE Rel-13. *Standards for Communications and Networking, 2016*, 1-7. 10.1109/CSCN.2016.7785170

Sabry, A. H., Hasan, W. Z. W., Ab. Kadir, M., Radzi, M. A. M., & Shafie, S. (2017). DC-based smart PV-powered home energy management system based on voltage matching and RF module. *Public Library of Science ONE, 12(9),* 22. http://dx.doi.org.proxy.lib.uni.edu/10.1371/journal.pone.0185012

Varga, A, K. (2014). ZigBee based wireless sensor networks. *Proceeding of International Scientific Conference on Advances in Mechanical Engineering 09-10,* 173-178.

Visual Studio. (n.d.). Retrieved April 26, 2019, from https://visualstudio.microsoft.com/

Yang, H., Lee, W., & Lee, H. (2018). IoT smart home adoption: The importance of proper level automation, *Journal of Sensors, 2018*, 11. https://doi.org/10.1155/2018/6464036

Appendix A

Bill of Materials

Bill of Materials			
No.	Name	QTY.	Cost
1	Arduino UNO	2	$33.8
2	XBee S2C	2	$44.12
3	Wire	1 (pack)	$5.00
4	Grove-Relay v1.2	1	$4.99
5	DHT22	1	$7.39
6	9V Battery	1	$1.00
7	1.8" TFT display	1	$12.79
8	Buzzer	1	$0.5
Total Cost			$109.59

Appendix B

Code for Transmitter Module

```
#include <gfxfont.h>
#include <SoftwareSerial.h>
#include <Adafruit_GFX.h>
#include <Adafruit_GFX.h>    // Core graphics library
#include <Adafruit_ST7735.h> // Hardware-specific library
#include <SPI.h>
#include <XBee.h>
#include <Printers.h>
// These pins will also work for the 1.8" TFT shield
#define TFT_CS     10  //TFT CS pin 10
#define TFT_RST    9  //TFT reset pin 9
```

```
#define TFT_DC    8  //TFT dc pin 8
#include <DHT.h> // DHT library
#define DHTPIN 7    // DHT sensor pin 7
#define DHTTYPE DHT22   // DHT 22
#define piezo 6      // buzzer pin 6
DHT dht(DHTPIN, DHTTYPE); //// Initialize DHT sensor for normal 16mhz Arduino
//Variables
const int xb_rx = 2; // Rx = pin2
const int xb_tx = 3; //Tx=pin3
bool detected = false; // avoid contact bonce (chatter)
int chk;
float hum; //Stores humidity value as floating-point constants
float temp; //Stores temperature value as floating-point constants

float sinVal;
int toneVal;
SoftwareSerial xbee(xb_rx, xb_tx);
int data = 1;
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_RST);
#define TFT_SCLK 13   // TFT clock to pin 13
#define TFT_MOSI 11   // TFT MOSI to pin 11
float p = 3.1415926;
void setup(void)
 {
  Serial.begin(9600);
  xbee.begin(9600);
  dht.begin();
  Serial.print("Hello! This is Luke's Proj Test"); // TFT display welcome screen
  tft.initR(INITR_BLACKTAB);   // Initialize LCD, black tab
  Serial.println("Initialized"); // Serial print “Initialized”
  uint16_t time = millis();
  tft.fillScreen(ST7735_BLACK);
  time = millis() - time;
  Serial.println(time, DEC);
  delay(500);
  // large block of text
  tft.fillScreen(ST7735_BLACK); // Screen fill color: black
  tft.setTextSize(2); //Test size: 2
  tft.println("Master"); // TFT display project title
  tft.println("Project");
  tft.println(" ");
  tft.println("Made by");
  tft.println("Qi Lu ");
  tft.println(" ");
```

```
 tft.println(" ");
 tft.setTextSize(1); // Set text size as 1
 // testdrawtext("                                                    THIS SYSTEM
IS DEVELOPED BY AND BELONG TO QI LU, WHICH IS A MASTER STUDENT IN THE
UNIVERSITY OF NOTHERN IOWA. IF YOU HAVE ANY QUESTION, PLEASE CONTECT
LUQAA@UNI.EDU", ST7735_WHITE);
 delay(1000);
 // tft print function!
 tftPrintTest();
 delay(400);
 tft.fillScreen(ST7735_BLACK);
 Serial.println("done");
 delay(1000); // delay 1 sec
}

void loop() {
 delay(2000);
 int val;
 int x;
 int y;
 val = temp;
 y = hum;
 // DHT22 reads data and store it to variables hum and temp
 hum = dht.readHumidity();
 temp = dht.readTemperature();
 if (val > 30) //if the tempure is higher than 30
 {
  for ( x = 100; x < 3000; x++) // Buzzer tune up
    tone (06, x, 3000); // Turn on buzzer
  delay(1); // play the tone
  tft.fillScreen(ST7735_BLACK);
  tft.setTextSize(2);
  tft.println("Tempt");  // TFT print
  tft.println("Anomaly"); //" tempt anomaly"
  delay (100); //delay 100ms
 } else // Else if not function
 {
  noTone(10);
 }
/* Once detected humidity is abnormal, for example is lower than 35%.
 LCD screen print "Humd on", and send alert email to users*/

 if (y < 35) // If humidity < 35%
 {
```

```
  tft.fillScreen(ST7735_BLACK);
  tft.setTextSize(2);
  tft.println("Humd On"); // LCD screen displays "Humd on"

  xbee.print(data);     //XBee transmitter send trigger signal
  Serial.println ("1"); //Serial print trigger signal
  detected = true;
  delay(1000); //delay 1000ms
 } else
 {
  if (y > 50)   // if humidity < 50%
    noTone(10); // Stop buzzer
  xbee.print(0);
  Serial.println (0);
  detected = false; // Disconnect the conection
 }

 //Print temp and humidity values to serial monitor
 Serial.print("Humidity: "); //Serial print Humidity
 Serial.print(hum);
 Serial.print(" %, Temp: ");
 Serial.print(temp);   //Serial print temperature
 Serial.println(" Celsius"); /
 delay(1000); //Delay 1 sec.
 tft.invertDisplay(true); //Switch back ground color (white to black)
 delay(500);
 tft.invertDisplay(false);
 delay(500);

 tft.fillScreen(ST7735_BLACK);
 tft.setCursor(0, 30);
 tft.setTextColor(ST7735_WHITE);
 tft.setTextSize(2);
 tft.println("Humidity: "); //LCD displays Humidity
 tft.println(hum);
 tft.println("Percent ");
 tft.println("  ");
 tft.println("Temp:"); //LCD displays temperature
 tft.println(temp);
 tft.println("Celsius ");
}
void testdrawtext(char *text, uint16_t color) //TFT test drawing
{
 tft.setCursor(0, 0);
```

```
  tft.setTextColor(color);
  tft.setTextWrap(true);
  tft.print(text);
}
void tftPrintTest()  //TFT test
{
  tft.setTextWrap(false);
  tft.fillScreen(ST7735_BLACK);
  tft.setCursor(0, 30);
  tft.setTextColor(ST7735_RED);
  tft.setTextSize(1);
  tft.println("Luke!"); //TFT test
  tft.setTextColor(ST7735_YELLOW);
  tft.setTextSize(2);
  tft.println("Luke!"); //TFT test
  tft.setTextColor(ST7735_GREEN);
  tft.setTextSize(3);
  tft.println("Luke!"); //TFT test
  tft.setTextColor(ST7735_BLUE);
  tft.setTextSize(4);
  tft.print(66666);
  delay(10000);
  tft.setCursor(0, 0);
  tft.fillScreen(ST7735_BLACK);
  tft.setTextColor(ST7735_WHITE);
  tft.setTextSize(1);
  tft.println("Made By Luke"); // TFT display author information (name, project title,
version, Tel number etc.)
  tft.setTextSize(1);
  tft.setTextColor(ST7735_GREEN);
  tft.print("Home Monitoring");
  tft.println(" ");
  tft.setTextColor(ST7735_WHITE);
  tft.println("Version 1.0 Beta");
  tft.println(" ");
  tft.print("Tel No.:");
  tft.println("319-8839124");
  tft.println(" ");
  tft.setTextColor(ST7735_WHITE);
  tft.println("Sketch has been");
  tft.println("running for: ");
  tft.setTextColor(ST7735_MAGENTA);
  tft.print(millis() / 1000);
  tft.setTextColor(ST7735_WHITE);
```

```
 tft.print(" seconds.");
 delay(10000);
}
```

Code for Receiver Module

```
#include <SoftwareSerial.h>
#include <XBee.h>
#include <Printers.h>
const int xb_rx = 2; // Rx= pin2
const int xb_tx = 3; //Tx = pin3
SoftwareSerial xbee(xb_rx, xb_tx);
int data = 0;
void setup() {
 Serial.begin(9600);
 xbee.begin(9600);
 pinMode(7, OUTPUT);  //Relay pin mode=7
}
void loop() {
 while (xbee.available() > 0) // while XBee receiver module receives trigger signal
 {
  data = xbee.read(); // Check the received signal
  Serial.println(data); // Serial print current receive signal
 }
 if (data == 49) // if received signal =49
 {
  digitalWrite(7, HIGH); // Switch pin mode to "High"
 }
 else
 {
  digitalWrite(7, LOW); // Switch pin mode to "Low"
 }
}
```

Code for Email Alert Function

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.IO;
using System.IO.Ports;
using System.Text;
using System.Net.Mail;
namespace gmail2
{
    class Program
    {
        static void Main(string[] args)
        {
            /* allow a serial communication with arduion'/
            SerialPort myport = new SerialPort();
            myport.BaudRate = 9600;
            myport.PortName = "/dev/cu.usbmodem14301"; //co
nnect to port
            myport.Open();
            /* this part is for mail sender*/
            // (from, to, header, body)
            MailMessage mail = new MailMessage("alarmtestlu
ke@gmail.com",
                "alarmtestluke@gmail.com",
                "Humidity Alarm",
                "smart home system an email");
            SmtpClient client = new SmtpClient("smtp.gmail.
com");
            client.Port = 587;
            /* gmail login in */
            client.Credentials = new System.Net.NetworkCred
ential("alarmtestluke@gmail.com",

        "mypassword1234");
            client.EnableSsl = true;
            while (true) // once receive signal form port
            {
                string data_rx = myport.ReadLine();
                if (Convert.ToChar(data_rx[0]) == '1') //if
 receive signal =1
                {
                    client.Send(mail); // send email
                    myport.WriteLine("1");
                }
            }
        }
    }
}
```