Noor Blum

CPE301  UCLV

Weather Shield

CPE 301 EMBEDDED SYSTEM DESIGN S 2015

Contents

[Goal: 2](#_Toc450292316)

[Deliverables 2](#_Toc450292317)

[Literature Survey: 2](#_Toc450292318)

[Block Diagram 3](#_Toc450292319)

[Flowchart 4](#_Toc450292320)

[4](#_Toc450292321)

[Working 4](#_Toc450292322)

[ Hardware 5](#_Toc450292323)

[ Software 5](#_Toc450292324)

[ Cloud 5](#_Toc450292325)

[Components 5](#_Toc450292326)

[Schematic 6](#_Toc450292327)

[PCB Design 6](#_Toc450292328)

[Implementation: 7](#_Toc450292329)

[Code 7](#_Toc450292330)

[References 7](#_Toc450292331)

**TITLE: Weather Shield using ESP8266 , DHT11 & ATmega328**

# Goal:

* Measure Temperature and humidity
* Send data on Cloud
* Monitor from Android Phone and PC

Deliverables:

* Source Code
* Hardware Schematics
* Android APK ( Generated )
* PCB Design Files
* Report Files

# Literature Survey:

We are always curious to measure our local weather irrespective of weather.com and accuweather.com, it always amused me to measure current room temperature and garden temperature, it is great fun to see and create a log of temperature on my cellphone with the help of cloud.

So this weather shield takes humidity and temperature readings from sensor DH11 and transfer the reference voltages to ADC of ATmega328p, now Uart of ATmega328 will send the data to ESP8266, which will send data to free cloud service, and our android app and web browser will show the current temperature and humidity values with Pass Graph.

# Block Diagram

**Cloud**

**ATMega328P**

**ESP8266**

DH11

Sensor

**OPTIONAL**

**(Only for Debugging)**

16x2 Matrix LCD

**Cloud**



# Flowchart

Initialize ADC

Initialize Port

Start

Send Data to Cloud

Read ADC Values

Initialize Wi-Fi

# 

# Working

Working of Weather Shield is simple enough to understand majorly there are 3 Parts Hardware, Software and Cloud, Basically Work of Hardware is to generate the value and DH11 will generate the value and Send to ADC of ATmega328P , then firmware inside ATmega328P will collect the data from ADC Port , Convert it into Actual readable format then Send the Data to TX of wifi module ,

Now wifi module will interact with internet and free cloud service, it will send data to cloud service and now device where you want to see, I have created one basic app which will see the cloud URL and display the picture.

## Hardware

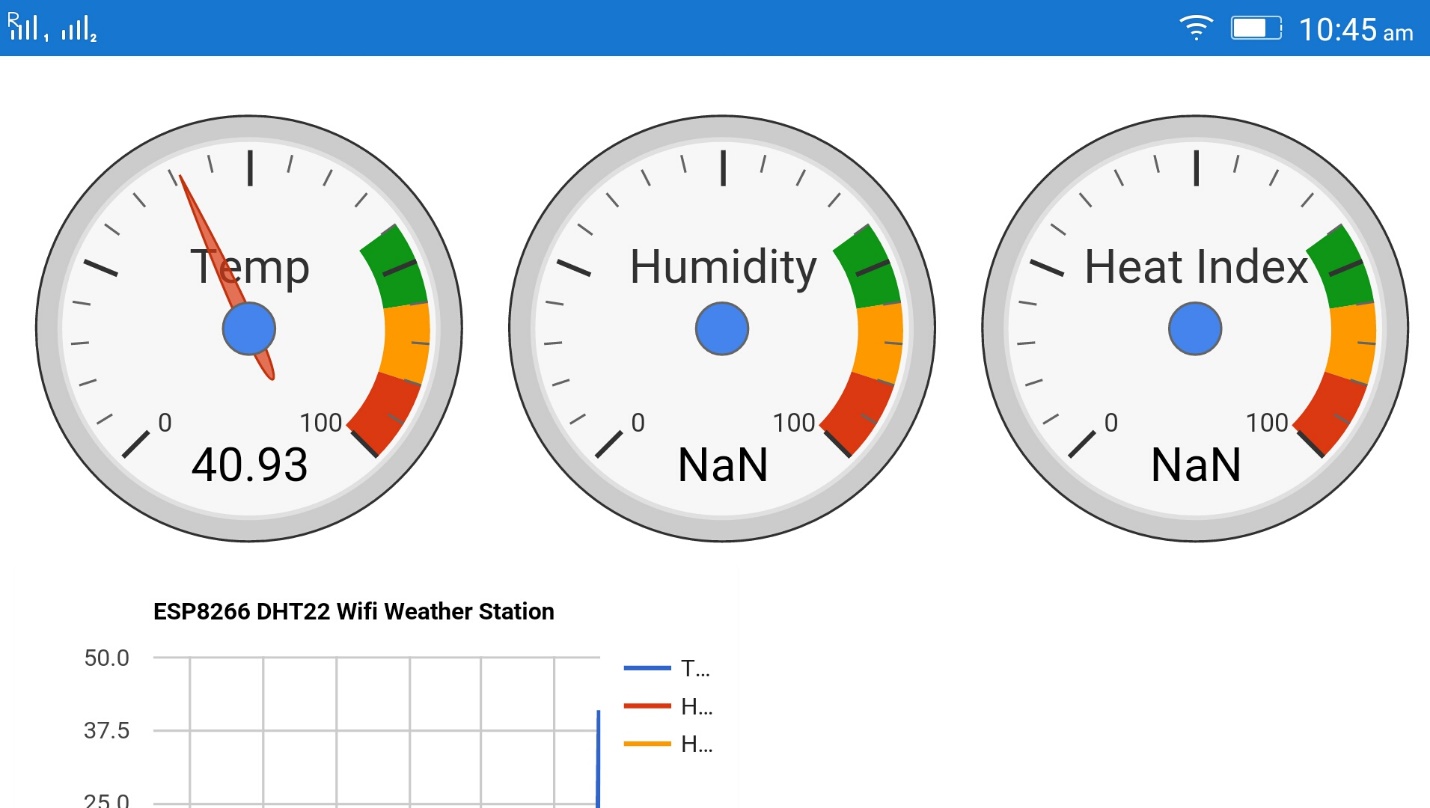
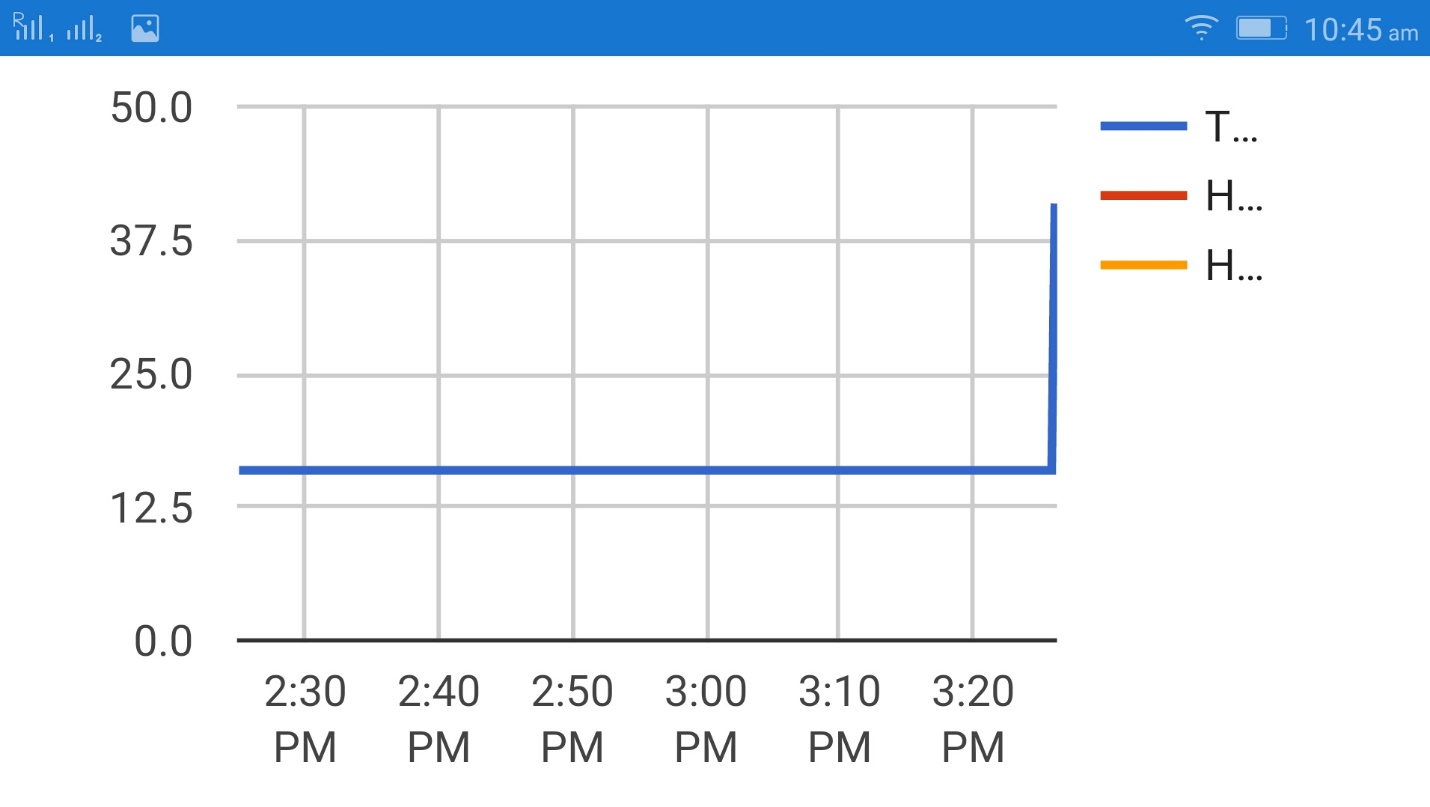
Atmega328P is main hardware of this project , basically it configure ports and convert data into actual ratings , it has various IO ports , in portd we have connected 16x2 LCD just to debug the data taken by sensor ,it can be removed in production version of Weather shield.

## Software

Software has major role in this project, firmware and other is app, the esp8266 will be configured through firmware and we are sending the data through the same firmware. So firmware plays an important role.

## Cloud

We have used free cloud service <http://data.sparkfun.com> , it has provided basic interface which will generate the cloud data in the graphical format



# Components

**Explain the main characteristics, interface, and limitation of the components used**

**Main Characteristic:**

In this project the main components are as below:

Microcontroller (ATMEGA328P)

* High Performance, Low Power Atmel®AVR® 8-Bit Microcontroller
* 32KBytes of In-System Self-Programmable Flash program memory
* 1KBytes EEPROM
* 2KBytes Internal SRAM
* 8-channel 10-bit ADC
* Programmable Serial USART

Temperature Sensor (DTH11)

* Measurement Range - 20-90%RH 0-50 ℃
* Humidity Accuracy - ±5％RH
* Temperature Accuracy - ±2℃
* Resolution -1

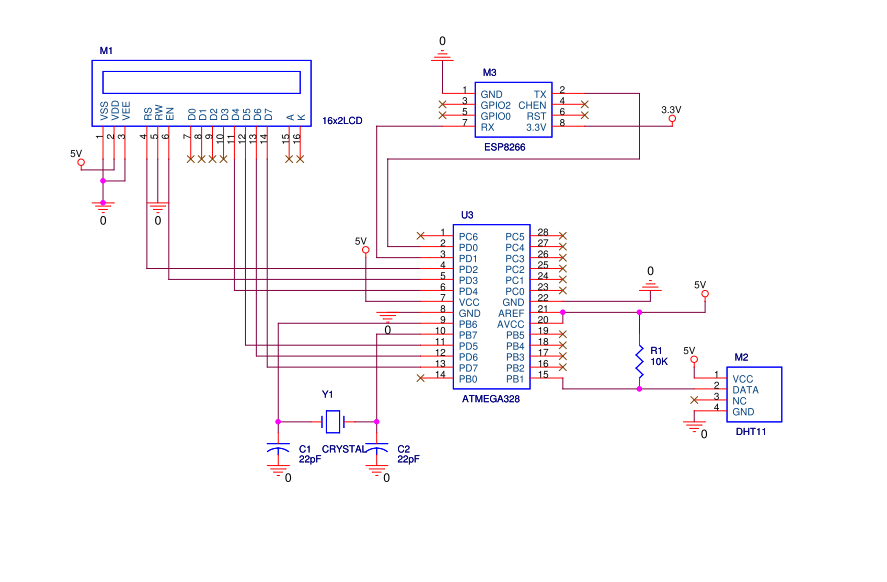
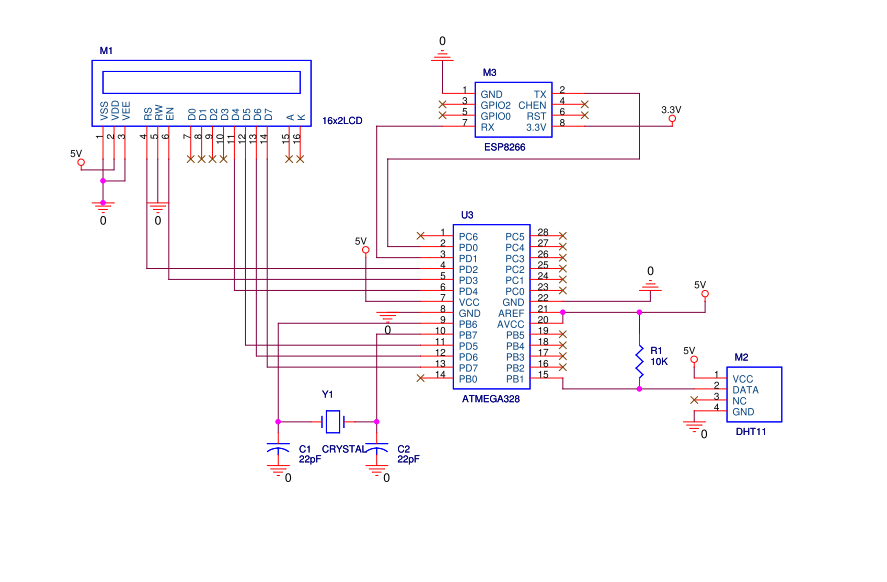
WiFi Module ( ESP8266 )

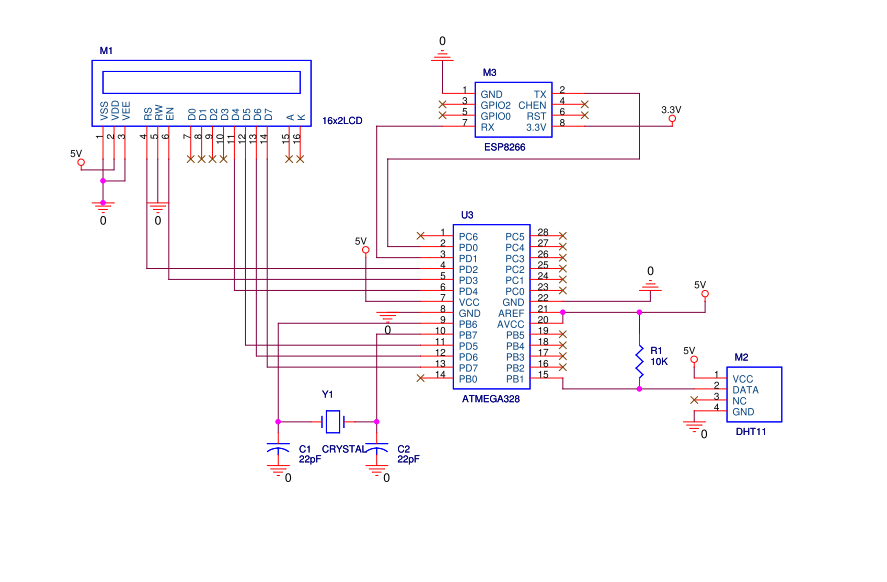
* 802.11 b/g/n
* Integrated low power 32-bit MCU
* Integrated 10-bit ADC
* Integrated TCP/IP protocol stack
* WiFi 2.4 GHz, support WPA/WPA2

LCD Display (16x2 LM016L)

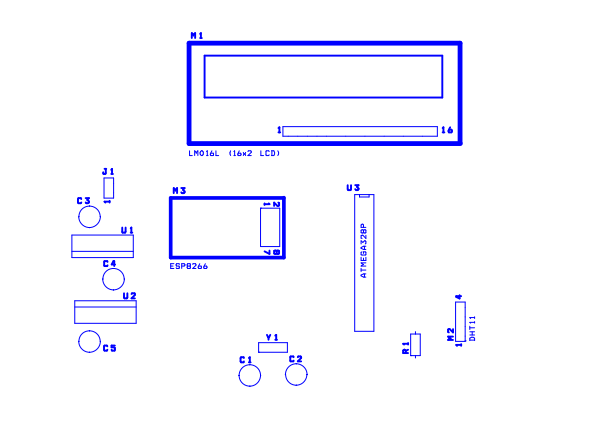
* Mechanical dimension – 80mm x 36mm x 13.5mm
* Power Voltage – 7V max
* Operating Temperature – 0 - 50 C

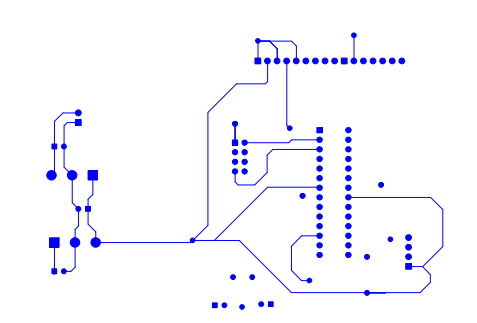
# Schematic

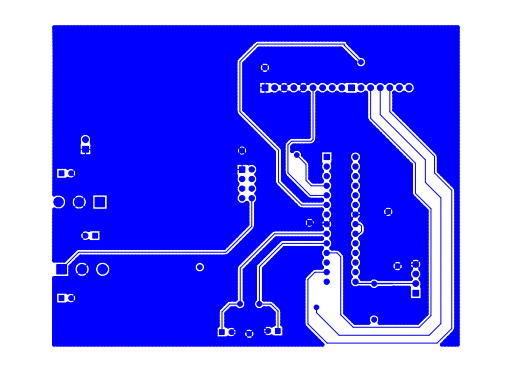


****

# PCB Design







# Implementation

* Create a C Code which communicate with ESP8266 , take temperature from DTH11 and Display the commands on LCD Screen
* Connect the hardware as Schematic
* Now C code will take data from Sensor DTH11 and upload to cloud.
* Change the wifi SSID and password into code as per your requirement.
* Log the data and see the results on HTML Webpage and Andoird App

# Snapshots/Screenshot/Videos

[https://github.com/blumn/weathershield/1.jpg](https://github.com/blumn/weathershield)

<https://github.com/blumn/weathershield/2.jpg>

<https://github.com/blumn/weathershield/3.jpg>

<https://github.com/blumn/weathershield/4.jpg>

<https://github.com/blumn/weathershield/5.jpg>

<https://github.com/blumn/weathershield/6.jpg>

<https://github.com/blumn/weathershield/7.jpg>

<https://github.com/blumn/weathershield/8.jpg>

<https://github.com/blumn/weathershield/videos.txt>

# 

# Code

## lcd\_header.h

#define F\_CPU 16000000ul //define CPU Clock

#include <util/delay.h> //Included delay function , which used later in code

#define rs PD2 //define RS (register select) pin

#define en PD3 //define EN pin

void cmnd() //function cmnd() which take care timing of en and rs pins

{

PORTD&=(~(1<<rs)); // rs pin Low

PORTD|=(1<<en); // en pin high

*\_delay\_ms*(5); // delay of 5ms

PORTD&=(~(1<<en)); // en pin low

}

void lcdcmd(char ch) // lcd command writing function

{

PORTD=ch & 0xF0; // assign the data on pins

cmnd(); // called cmnd function

PORTD=(ch<<4) & 0xf0; // Data serially to pins

cmnd(); // called cmnd function

}

void lcd\_init() // initialize sequence of LCD

{

lcdcmd(0x02);

lcdcmd(0x28);

lcdcmd(0x0e);

lcdcmd(0x01);

}

void data() // lcd data timing function same as cmnd

{

PORTD|=(1<<rs); // rs pin low

PORTD|=(1<<en); // en pin high

*\_delay\_ms*(5); // delay 5 ms

PORTD&=(~(1<<en)); // en pin low

}

void lcddata(char ch) // lcd data witing funcition

{

PORTD=ch & 0xF0; // configuring the ports

data(); // calling timing function

PORTD=(ch<<4) & 0xf0; // shifting pin serially

data(); // calling data function

}

void lcdprint(char \*str) // data printing function on lcd

{

while(\*str)

{

lcddata(\*str); // transferring data until it is printed

str++; // incrementing

}

}

## serial\_header.h

void serialbegin(unsigned int BAUD, unsigned int FOSC) //serial transfer init function

{

unsigned int MYUBRR=((((FOSC\*1000000)/16))/BAUD)-1; //setting up the baud rate

UBRR0H = (MYUBRR >> 8);

UBRR0L = MYUBRR;

UCSR0B |= (1 << RXEN0) | (1 << TXEN0); // Enable receiver and transmitter

UCSR0B |= (1 << RXCIE0); // Enable reciever interrupt

UCSR0C |= (1 << UCSZ01) | (1 << UCSZ00); // Set frame: 8data, 1 stp

}

char serialread() // Serial read function

{

while(!(UCSR0A & (1<<RXC0))); // run until the RXC0 bit is set

return UDR0;

}

void serialwrite(unsigned char ch) // Serial write function

{

UDR0=ch;

while(!(UCSR0A & (1<<UDRE0))); // run until the URE0 bit set

}

void serialprintln(char \*str) // Serial Printing function

{

serialprint(str);

serialprint("\r\n");

}

void serialprint(char \*str) // Serial Printing String

{

while(\*str)

{

serialwrite(\*str);

str++;

}

}

## main.c

#include <avr/io.h> //include AVR IO lib header

#include <stdint.h> //include stdint headers

#include <avr/interrupt.h> //include interrupt headers

#include "serial\_header.h" //include serial\_header.h for sending data ESP5266

#include "lcd\_header.h" //include lcd\_header.h for printing status messages

#define sensor PB1 //Sensor DTH11 connected with this port

char \*pvt\_key="gzBw6x6Nx6Io2qd9Mlq5"; //pvt key used to log the data into cloud server

char \*public\_key="Jxr64n4Kn4Ir4KWpQXKL"; //public to get the log data and display

static char postUrl[200]; //url decleration

char \*text,mytext[4];

unsigned char a = 0, b = 0,d = 0,t1 = 0,t2 = 0,

rh1 = 0,rh2 = 0,sum = 0;

char temperature[3];

char humidity[3];

int i=0;

char rec[100];

char ReceivedChar=0;

void StartSignal() //Function for Starting Sensor DTH11 value

{

DDRB|= 1<<sensor; //Configure RD2 as output

PORTB&= ~(1<<sensor); //RD2 sends 0 to the sensor

*\_delay\_ms*(18); //Delay Function

PORTB|=1<<sensor; //RD2 sends 1 to the sensor

DDRB&=~(1<<sensor);

PORTB&=~(1<<sensor);

}

void CheckResponse() //Checking the Sensor Response

{

a = 0;

*\_delay\_us*(40);

while((PINB&(1<<sensor))); // check Sensor pin is Set

if(!(PINB&(1<<sensor))) // if sensor pin is low

{

*\_delay\_us*(80); // delay

if (PINB&(1<<sensor))// if sensor in high

{

a = 1; // set variable a to 1

*\_delay\_us*(80);// delay

}

}

}

void ReadData() // Read Sensor reading

{

for(b=0;b<8;b++)

{

while(!(PINB&(1<<sensor))); //Wait until PORTD.F2 goes HIGH

*\_delay\_us*(40); // Delay

if(!(PINB&(1<<sensor))) // if Sensor pin low

d&=~(1<<(7-b)); // Clear bit (7-b)

else

{

d|= (1<<(7-b)); //Set bit (7-b)

while(PINB&(1<<sensor));//Until Sensor Pin is high

}

}

}

void show() // Calculate and Show Reading to LCD

{

if(a == 1)

{

ReadData(); // Read data from Sensor

rh1 =d; // Store data in variable

ReadData();

rh2 =d;

ReadData();

t1 =d;

ReadData();

t2 =d;

ReadData();

sum = d; // Read the sum of data

if(sum == rh1+rh2+t1+t2) // if Sum is Equal

{

lcdcmd(1); // LCD command

text = "Temp: .0C"; // Text Writing on LCD

lcdprint(text); // LCD Showing the above Text

lcdcmd(192); // LCD Command for writing at 0x192

text = "Humidity: .0%"; // Text should be writing on Sencond Line

lcdprint(text); // Display the Text

*sprintf*(temperature,"%d",t1); // Update the value with Temperature

lcdcmd(0x85); // Move the cursor

lcdprint(temperature); // Display the Temperature

*sprintf*(humidity,"%d",rh1); // Update the value with humidity

lcdcmd(0xc9); // Move the cursor

lcdprint(humidity); // Display the Humidity

}

else

{

lcdcmd(1); // Writing to first digit

lcdprint("Check sum error"); // Display “Check sum Error”

}

}

else

{

lcdcmd(1); // if Sensor is not connected

lcdprint("No response"); // No Response will display

lcdcmd(192); // Second line

lcdprint("from the sensor"); // From the Sensor will display

}

}

ISR(USART\_RX\_vect) // Interrupt Service Routine

{

ReceivedChar = UDR0; // Read data from the RX buffer

rec[i++]=ReceivedChar; // Write the data to the TX buffer

}

void get\_ip() // get network IP Connection

{

char IP[16];

char ch=0,j=0;

char flag=0;

while(flag==0)

{

i=0;

serialprintln("AT+CIFSR"); //Send the AT Command CIFSR

*\_delay\_ms*(2000); // Delay

if(i>0)

{

for(j=0;j<i;j++)

{

lcdcmd(1); // Display 1

lcdprint("Wait...."); // Display wait

if(rec[j]=='S' && rec[j+1]=='T' && rec[j+2]=='A' && rec[j+3]=='I' && rec[j+4]=='P')

{

j=j+6;

int n=0;

while(i!=j)

{

while(rec[j]!='+')

{

IP[n++]=rec[j++];

}

flag=1;

break;

}

}

}

}

}

lcdcmd(1);

lcdprint("IP:"); // Display the IP

lcdprint(IP); // Print the IP

lcdcmd(192); // Print the Port on Second line

lcdprint("Port:"); // Print “Port”

lcdprint(80); // Print number 80

i=0; // Delay

*\_delay\_ms*(5000);

}

void send(char \*str, unsigned int time) // Send Character on Serial to ESP8266

{

while(1) // this is forever loop ,

{ // through this function we send AT Commands

int j=0,temp=0;

i=0;

serialprintln(str); // send serial command

lcdcmd(1); // Move the Cusor 1

lcdprint(str); // lcd print the Serial Command

for(int t=0;t<time+2000;t++) // Delay

*\_delay\_ms*(1);

if(i>0) // print OK

{

for(j=0;j<i;j++)

{

if(rec[j]=='O' && rec[j+1]=='K')

{

lcdcmd(192);

lcdprint("OK");

temp=1;

*\_delay\_ms*(1000);

i=0;

break;

}

else if(rec[j]=='E' && rec[j+1]=='R' && rec[j+2]=='R' && rec[j+3]=='O' && rec[j+4]=='R') // Print Error

{

lcdcmd(192);

lcdprint("Error");

*\_delay\_ms*(1000);

i=0;

}

}

}

if(temp==1)

break;

}

}

void connect\_wifi() // Connect Wifi function

{

send("AT",1000); // Send few AT Commands

send("AT+CWMODE=1",1000); // AT command CWMOD

lcdprint("Connecting WIFI"); //LCD will display

send("AT+CWJAP=\"wifissid\",\"wifipassword\"",10000); // Send command

}

void httpGet(char \* ip, char \*path, int port) // Get the Cloud Url

{

int resp;

port=80;

serialprint("AT+CIPSTART=\"TCP\",\""); // Sending URL

serialprint(ip); // Sending to Cloud

serialprintln("\",80");

*\_delay\_ms*(1000);

int len = *strlen*(path); // Path

char buff[3];

char \*atSendCmd = "AT+CIPSEND=";

*sprintf*(buff,"%d",len); // Send the Length

serialprint(atSendCmd); // Send Command

serialprintln(buff); // Send buffer

//GET /invoice/ HTTP/1.0\r\n\r\n

serialprint("GET /");

serialprint(path);

serialprintln(" HTTP/1.0\r\n\r\n");

}

int main( void )

{

int z=0;

DDRD=0xfE; //initialize Port D

DDRB=0xff; //initialize Port B

DDRB|=1<<PB5;

lcd\_init(); //initialize LCD

serialbegin(115200,16); // baad rate and frequency in MHz

sei(); // Enable Global Interrupts

lcdprint("System Ready"); //LCD will display after Power ON

*\_delay\_ms*(1000); //Delay

while(1) //Infinite Loop

{

lcdcmd(1); //Move the curson to 1st pos

StartSignal(); //Getting Sensor Data

CheckResponse(); //Checking Sensor Response

show(); //Display humidity and temperature reading on Sensor

if(z==5)

{

connect\_wifi(); //Connect wifi

float temp=t1;

float humid=rh1;

long pressure=10;

char tempStr[8];

char humidStr[8];

char presStr[8];

*dtostrf*(temp, 5, 3, tempStr); //Convert to String

*dtostrf*(humid, 5, 3, humidStr); //Convert to String

*dtostrf*(pressure, 5, 3, presStr); //Convert to String

*sprintf*(postUrl, "input/%s?private\_key=%s&humidity=%s&temp=%s",public\_key, pvt\_key, humidStr,tempStr); // post to Cloud URL

httpGet("data.sparkfun.com", postUrl, 80); //Get the URL Data

*\_delay\_ms*(100);

serialprintln("AT+CIPCLOSE=0"); // Send AT Command to Close the Connection

*\_delay\_ms*(2000); // Delay

z=0;

}

*\_delay\_ms*(2000); // Delay

z++;

}

}

# References

* <http://www.atmel.com/Images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Summary.pdf>
* <http://www.micropik.com/PDF/dht11.pdf>
* <https://cdn-shop.adafruit.com/product-files/2471/0A-ESP8266__Datasheet__EN_v4.3.pdf>
* <https://www.sparkfun.com/datasheets/LCD/ADM1602K-NSW-FBS-3.3v.pdf>
* data.sparkfun.com (for cloud service)
* [www.sparkfun.com](http://www.sparkfun.com)
* [http://hobbyist.co.nz](http://hobbyist.co.nz/sites/default/files/datalog/graph/graph.html)

Noor Blum Page 17/17