IoT Based Weather Reporting System

Mohana C, Nandini G N

Sakthi ragavan, sumith, Syed ashwaq ahmed, Irfan,Varadaraju

Department of Electronics and communication, Dr. TTIT

[mohana@drttit.edu.in](mailto:mohana@drttit.edu.in), [nandini@drttit.edu.in](mailto:nandini@drttit.edu.in)

[sakthiragavan359@gmail.com](mailto:sakthiragavan359@gmail.com), [sumithrblezzer@gmail.com](mailto:sumithrblezzer@gmail.com), [syedashwaq153@gmail.com](mailto:syedashwaq153@gmail.com), [mohammedirfanm785@gmail.com](mailto:mohammedirfanm785@gmail.com), varadarajud51@gmail.com

# Abstract The IoT-based Weather Monitoring and Reporting System project is used to get Live reporting of weather parameters. It will Monitor temperature, humidity, altitude, and pressure. when Scientists/nature analysts want to monitor changes in a particular environment like a volcano or rainforest and agriculture to see the weather condition to get the best yield or to monitor the physical parameters in a laboratory. It then displays all data in the Blynk application. The project has been developed using ESP 32 Microcontroller, DHT 11 temperature and humidity sensor, LDR (light dependent resistor), BMP280 pressure and altitude sensor, and OLED display to show the parameters measured. This system will monitor the changes of weather condition happening over the environment and then provides the users fastest way to access the information from anywhere

***Keywords*** *IOT, weather station, android interface for Wi-Fi communication, microcontroller, embedded c/c++ and weather monitoring.*

# I INTRODUCTION

The weather reports these days is unpredictable and need to be accurate because the climate changes drastically over the weather. because of that, the Weather Reporting System is needed used to monitor the continuously changing climatic and weather conditions over various areas like house, industry, agriculture, etc. And monitoring a weather parameter can help in Labotories as well for maintaining certain parameters. in real-time monitoring. Internet of Things (IoT) platform use is Blynk it’s should be able to displaying the weather parameters and the information will also be visible from any place of the world and it’s also displaying on the OLED display with microcontroller communicates via Wi-Fi hotspots to get the internet access. The value of some place that is sent by satellite weather report system does not give the accurate condition. However, the problem comes when we needed the accurate weather report at the current time. With this weather reporting system, all-weather parameters measured using sensors will be controlled by the ESP32 microcontroller as the main node that will send all the data collected by sensors to the database off Blynk and will visible from anywhere in the world and also on the OLED display that uses Wemos D1 mini as its microcontroller and a I2C connection. Hear the display and the pressure altitude and temperature sensor use the I2C connection for communication which stands for the inter interconnected circuit where the one master device can control 64 slave devices (*because it has an 8-bit address line*) so the display is connect to the address 0x3C and the BMP280 is connected to 0x76 address the added advantage of this is using only 2 lines called SDK and SDL (laving GND and VCC) therefore only using 2 lines we can control many devices(64) simultaneously , the DHT11 using the serial commutation and the LDR using the analog to digital converter to senses the changes, This MCU will monitor the changes of weather condition happening over the environment and then provides the users fastest way to access the information from anywhere using and android application Blynk.

# METHODOLOGY

This project will concentrate on the design and development of the Blynk (an *IoT platform)* that shows the data of the sensor on an android or iso platforms. The process is divided into two parts which are hardware circuit part and software development part. The hardware development involves the circuit construction and develops the structure and circuitry. On the other hand, the software part involves microcontroller programing, and data analyzing.

By using three (3) types of sensors to monitoring the weather parameter that are temperature, humidity, luminance, pressure, and altitude the system will be able to display the weather parameters with the sensor value data. All the data will be control by a microcontroller ESP32 and Wemos as the driver IC that will receive the sensor data from ESP32 and display it on OLED display. Furthermore, this system will also be seen on an android and iOS application that is Blynk to display the weather parameters value. The data collected will be sent to the Blynk web servers and those data will be reflected in all the applications and microcontroller. The Internet of Things (IoT) will connect the system with the user wireless and online without the need of checking manually.

**SYSTEM DESIGN**

Our main microcontroller which we are using is called as the node MCU because we can connect multiple of those controllers together and form a network just like computers do because of this special feature multiple microcontrollers like this can be connected to gather and form a network where each microcontroller will act like a node hence it is called the **node** MCU and the MCU stands for the microcontroller unit

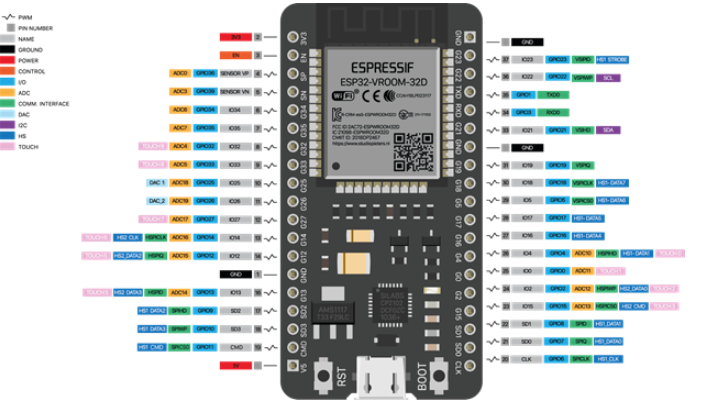
Since this is an IOT based project, the microcontroller needs to be connected to the interne at least for the first time boot up because it needs to fetch the data like current date, time and altitude of our current location

The microcontroller can be programed using 2 ways one using the Arduino ide which is well known for the Arduino development board and the other way is using the BareMetal programming using the standalone language specifically designed for this microcontroller in our project we are using Arduino ide and our project is written in c and C++

**ESP32 DEVELOPMENT BOARD:** ESP32 is a low-cost, low-power consumption systems on a chip of microcontrollers with integrated Wi-Fi and dual-mode Bluetooth (low power and performance mode where the low power is for the short range and the high power can be used for long range and while the initial setup). The ESP32 microcontroller has a Tensilica Xtensa LX6 microprocessor it is a both dual-core and single-core CPU chick can be designed by the programmer and includes built-in antenna switches for the Bluetooth and Wi-Fi connectivity, low-noise receive amplifier, RF balun, power amplifier, filters, and power management modules. ESP32 is developed and manufactured by Espress Systems, a Shanghai-based Chinese company, and is manufactured by TSMC using its 40 nm process.

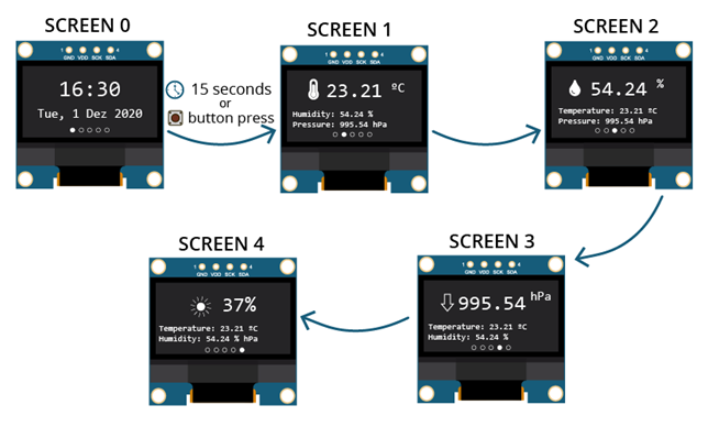
**Features**

* processor CPU: Xtensa dual-core (or single-core) 32-bit LX6 microprocessor, operating at 160 or 240 -MHz and performing at up to 600 DMIPS, Ultra-low-power (ULP) co-processor.
* Memory: 520 KiB SRAM, 448 KiB ROM.
* Wireless connectivity: Wi-Fi: 802.11 b/g/n, Bluetooth: v4.2 BR/EDR and BLE (shares the radio with Wi-Fi).
* Peripheral interfaces:
  1. 34 × programmable GPIOs.
  2. 12-bit SAR ADC up to 18 channels.
  3. 2 × 8-bit DACs. 10 × touch sensors (capacitive sensing GPIOs).
  4. 4 × SPI. 2 × I²S interfaces.
  5. 2 × I²C interfaces. × UART.
  6. SD/SDIO/CE-ATA/MMC/eMMC host controller.
  7. SDIO/SPI slave controller.
  8. CAN bus 2.0. Infrared remote controller (TX/RX, up to 8 channels).
  9. Motor PWM. LED PWM (up to 16 channels).
  10. Hall effect sensor and temperature sensor



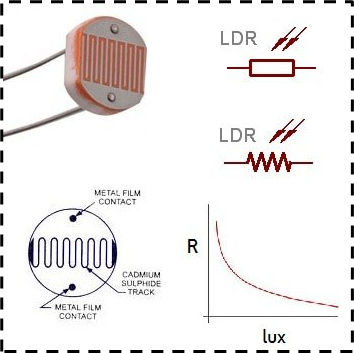
**ESP32 PIN DIAGRAM (Figure 1)**

**OLED DISPLAY:** This display is made of 128x64 individual white OLED pixels; each one is turned on or off by the SH1106 controller IC as its microcontroller chip. the display made up of individual pixels hence no backlight is required and the display will also be compact and light weight to carry. This reduces the power required to run the OLED display which has the diagonal size of 0.96 inches. The 128x64 pixels OLED Display has an outline dimension of 26.70 x 19.26 mm and AA size 21.74x10.86 mm, it is built-in with SH1106 controller IC and it communicates via 6800/8080 8-bit parallel, I2C, and 4-wire serial interface. In our project we have the display programed to perform the 5 different operation which are more likely called as screens, the 1st display shows the current date and time of the current location which will be fetched by the microcontroller once it is connected to the internet , 2nd display shows the temperature as a main parameter and humidity and pleasure as the secondary parameters and corresponding the red led will be turned on, 3rd display we have the humidity as the main parameter and temperature and altitude as the secondary parameters and the corresponding the blue led will be turned on, 4th display shows the pleasure as a main parameter and humidity and temperature as the secondary parameters and corresponding the green led will be turned on and finally 5th display we have the luminance as the main parameter and temperature and humidity as the secondary parameters and the corresponding the blue led will be turned on Apart from these 5 led we have other 2 Leds one of them is the WIFI status led and display switch led. The Wi-Fi status led is a multicolor led shows us whether the microcontroller is connected to internet or not once the microcontroller is connected to the internet the Wi-Fi Staus led turns into green color and when the Wi-Fi is not connected the Wi-Fi status led will be in blue color, since this is an IOT based project the microcontroller need to be connected to the internet at least for the first time boot up because it need to get some data like current date and time our current location the location will be automatically identified once the microcontroller is connected to the internet. Well, the other led is the display switch indication led which will turn on once the display switch commend is given from a Blynk application which will be mentioned in the upcoming pages



**OLED DISPLAY AND ITS INTERFACE (Figure 2)**

**LDR (LIGHT DEPEENDENT RESISTOR):** A photoresistor (also known as a light-dependent resistor, LDR, or photo-conductive cell) is a passive component that decreases resistance concerning receiving luminosity (light) on the component's sensitive surface. The resistance of a photoresistor decreases with an increase in incident light intensity; in other words, it exhibits photoconductivity



**LIGHT DEPEENDENT RESISTOR (Figure 3)**

The LDR stands for the light dependent resistor in our project led is used to senses the amount of light present in the environment as a result instead of displaying the luminance we will be displaying percentage of light falling on the sensor (the minimum and maximum values can in modified in the program which will be loaded into the microcontroller), the LDR 1st terminal is connected to the ground and the other pin is connected to the 10k resistor to limit the current and the resistor leads to the +VCC. The junction of +VCC and the resistor is used as the data lines which will be connected to the analog to digital converter of the microcontroller and the code is done in such a way that the output will in the percentage

# **BMP280 SENSOR:** BMP280 Digital Pressure Sensor is an absolute biometric pressure sensor specially designed for mobile applications. The sensor module is made in an extremely compact package. Bosch Sensor tec is the company who manufactured BMP280 Digital Pressure Sensor small dimensions and its low power consumption allow for the implementation in battery-driven devices.

# The BMP280 is based on piezo-resistive pressure sensor technology, featuring high accuracy and linearity as well as long-term stability and high EMC robustness. Numerous device operation options offer the highest flexibility to optimize the device regarding power consumption, resolution, and filter performance.

# 

# **BMP280 (Figure 4)**

# The above diagram shows the connection for the for I2C connection apart from the ground and VCC this type of connection has the SDA and SDK where SDA stand for the serial data line and SDK stands for serial clock line, both of these lines are connected to the microcontroller SDA and SDK respectively.

# **FEATURES:** this sensor is mainly used to measure the temperature pressure and altitude

# 300hPa to 1100hPa pressure range, 8-pin LGA metal-lid package

# ±0.12hPa, equiv. to ±1m (950 … 1050hPa @25°C)

# relative accuracy Typ. ±1hPa (950 to 1050hPa, 0 to +40°C)

# absolute accuracy 1.5Pa/K, Equiv. to 12.6cm/K temperature coefficient offset

# Digital Interfaces

# I2C (up to 3.4MHz)

# SPI (3 and 4 wire, up to 10MHz)

# 2.7μA @ 1Hz sampling rate current consumption

**Circuit:** The block diagram consists of the components that are used in this project. Firstly, the microcontroller will involve ESP32mcu. This microcontroller board will communicate with other sensors as specified to get sensor data from the respective sensors and send data collected to servers via wireless communication and hotspot Wi-Fi. the data is then synced with the Blynk server and displayed to OLED display. The microcontroller will display the sensor data on OLED and also on the Blynk app. The data collected will be analyzed and displayed in the application

# 

# **Block diagram (Figure 5)**

|  |  |
| --- | --- |
| **components** | **GPIO pin numbers** |
| OLDE Display - SDI | GPIO-21 |
| OLDE Display - SDI | GPIO-22 |
| BMP280 Display - SDI | GPIO-21 |
| BMP280 Display - SDI | GPIO-22 |
| DHT11 - data | GPIO-00 |
| LDR | GPIO-33 |

|  |  |
| --- | --- |
| WI-FI LED | GPIO-16 |
| RED LED | GPIO-05 |
| BLUE LED | GPIO-32 |
| GREEN LED | GPIO-27 |
| YELLOW LED | GPIO-12 |

# **Pin configuration and circuit**

# 

# **Hardware circuit connection (Figure 6)**

# The microcontroller needs to be powered by constant 5v 1A power supply via the vin and ground pin or using the micro-USB port, we will be using the USB type A to micro-USB cable and a charger adaptor for powering as it is maintained under specified range, the microcontroller can maximum output of 5v,3.3v with the maximum current flow of 80mA so to limit the current flow 270 ohm resistors are used in series with the led for a stable output and all the connections are made in the breadboard, our project consist of a touch sensor which can be connected to any Metalica surfaces to make it touch sensitive, the touch sensor measures the amount of charge present in the surface, this touch sensor is used to switch between the displays(screens) and the touch sensor is ground isolated from the supply , more over the microcontroller board has a built in voltage regulator so we can power it up with 3.3V to 12V supply

# **ESP32 power consumption**

|  |  |
| --- | --- |
| **Mode** | **Power consumption** |
| Wi-Fi Tx packet 13dBm~21dBm | 160~260mA |
| Wi-Fi/BT Tx packet 0dBm | 120mA |
| Wi-Fi/BT Rx and listening | 80~90mA |

**Android interface:**

# **Blynk:** The system can display sensor data and those data will be sent to Blynk servers and will be visible from anywhere around the world. The sensor data completely shown on the Blynk app and we can also control the microcontroller from the application. Thisandroid app is free to build and can be download it on Google Play store and register with an email which can be used to login and view the data. Blynk application was designed and developed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data and visualize it. every time you press a Button in the Blynk app, the message will be transmitted to the Blynk Cloud, where it magically finds its way to your hardware. It works the same in the opposite direction and everything happens in a Blynk of an eye.

# **Bynk Server:** This is the procedure need to be followed to access the data from a smartphone and the initial setup is also include which need to be done by developer. microcontroller need to be connected to the Wi-Fi or hotspot of the smart phone. name of the hotspot and password need to be included in the program via the Arduino ide, then the microcontroller uses the token code from Blynk lets the microcontroller to connect to the server of Blynk. the server of Blynk application will process the data and sends the data to the smartphone connection. Blynk server will check for internet connection. The information included to the code must be match with the hotspot information to allow microcontroller to connect with the WIFI to be as a channel to exchange commands between smart phone and microcontroller. Remaining processes are just commands sent from Blynk application to microcontroller.

# After the installation we will be in the login page there we can either use the email id and password to login or we can use the QR code to get access to the application to control and manage the microcontroller

# 

# **Installation (Figure 7)**

# We can start searching for Blynk on the play store in the android smart phone (use apple Appstore for iOS) once that’s done, we just need to download and install

# 

# **Setup page (Figure 8)**

# 

# **QR code (Figure 9)**

# Scanning this QR cord from the Blynk setup page will lead you to the project home page

# 

# **Project home page (Figure 10)**

# Well, this is the home page as we can see there are 2 gauges in the 1st row the first one is the temperature Guage and the 2nd one is the luminance's Guage, below that we have the graph named data which shows the variation of parameters in a graphical way, we can inaugurally tune on and turn off parameters in the graph, as well we can view the past 3 mounts data on the graph by toggling the buttons given below

# The last row has one button called screen and the showing the value of the altitude parameter, using the screen button we can switch between displays wirelessly from the android application

# 

# **Graph's (Figure 11)**

# As shown in figure 11 all the 4 parameters data can be accessed at a time to get a clear view of the weather condition. The buttons in the bottom row let us view the live data or the data which had been recorded in the cloud using the application we can only store 3 Monts of data so after every 3 months all the data can be exported to the google spread sheets which let us record a lot of data which can be used in many applications

# **CODE:** https://github.com/asterisk-ragavan/wether-station/blob/main/project.ino

# 

# **Figure 12(pressure)**

# 

# **Figure 13 (Humidity)**

# 

# Figure 14 (temperature)

# 

# **Figure 15 (luminous)**

**Conclusion:** as a conclusion, this project has cleared the objective that to build a system that can measure the weather parameters by the wireless system and IoT. The Weather station will be communicated by hotspot Wi-Fi and its data sent to the cloud and can be viewed from anywhere in the world. This weather reporting system will display the sensor data to the Blynk app that can be installed in the google play store or Appstore. The main importance of collecting the weather data is that we can predict the future climate conditions and the data which we collected can be used to find the climate changes as well which is a very important thing to be noticed and maintain it properly to keep our mother earth clean

# **REFERENCES**

* 1. randomnerdtutorials.com, “esp32-weather-station” [Online]. https://randomnerdtutorials.com/esp32-weather-station-pcb/
  2. Anita M. Bhagat, Ashwini G. Thakare, Kajal A. Molke, Neha S. Muneshwar, “IOT Based Weather Monitoring and Reporting System,” Volume: 3 | Issue: 3 | Mar-Apr 2019 Available Online: www.ijtsrd.com e-ISSN
  3. https://github.com/adafruit/Adafruit\_BME280\_Library.
  4. https://create.arduino.cc/projecthub/najad/interfacing-and-displaying-images-on-oled-59344a
  5. http://arduino.esp8266.com/stable/package\_esp8266com\_index.json
  6. https://github.com/espressif/arduino-esp32