Objective :

The experiment aims to implement a weather forecast system using the ADC modules of an ~~Arduiki~~ Arduino. The BMP180 or MPL115A sensor is used to measure environmental parameters, such as temperature, pressure and humidity. The sensor is stationary and located in a protected area to avoid interference from strong airflows. The pressure changes due to the weather are slow, requiring a few hours to determine the sloping of the pressure change. The BMP180 or MPL115A is well-suited for weather pattern prediction, with it's pressure range and resolution. Low pressure is typically seen as the precursor to worsening weather, while high pressure increases can indicate improving or clearing weather. The experiment provides

a simple approach for weather prediction by analyzing the trend of increasing or decreasing pressure signaling a worsening "cloudy" or "rainy" day. The results can be interpreted by looking at the 12-hour time frame. Local weather stations should also be consulted to check the results of the weather forecast system.

The apparatus and software name:

i) Arduino IDE
ii) Arduino Uno Board
iii) BMP180 / MPL115A
iv) Inches96 inch OLED 128X64
v) Breadboard
vi) Connecting Wires.

Theory :

The theory of this experiment involves using the ADC modules of an Arduino microcontroller to measure environmental parameters such as temperature, pressure, and humidity and using these measurements to create a weather forecast system. The BMP180 or MPL115A pressure sensor is used to measure barometric pressure, which can be used to predict weather patterns. Changes in biometric pressure can directly correlate to changes in weather, with low pressure often indicating worsening weather and high pressure indicating improving or clearing weather. To create a standalone weather station, it is important to consider the location of the sensor

and potential environmental factors. that may affect the accuracy of the readings, such as strong air flows. It is also important to normalize pressure readings for altitude, as different altitudes can affect barometric pressure measurments. Algorithms for weather prediction can involve looking at the trend of increasing or decreasing pressure over time to predict "sunny" or "clean" days versus "cloudy" or "rainy" days. The approach can be interpreted as an increase or decrease gradient over time, and the user can look at the results over a 12-hour time frame to make weather predictions.

Analysis	Output
$dP > +0.25kPa$	Sun Symbol
$-0.25kPa < dP < 0.25kPa$	Sun/cloud symbol
$dP < -0.25kPa$	Rain Symbol

Procedure :

i) Collect the required materials such as Arduino board, BMP180 or MPL115A sensor, wires, breadboard, and a computer with Arduino IDE software installed.

ii) Connect the BMP180 or MPL115A ses sensor to the Arduino board using wires and a breadboard as per the circuit diagram.

iii) Open the Arduino IDE software on the computer and write the code to interface the sensor with the Arduino board and obtain the temperature,

pressure and humidity data.

iv) Upload the code to the Arduino board using a USB cable and check the serial monitor for the obtained data.

v) Calibrate the sensor if required by comparing the obtained data with a local weather forecast or data obtained from another reliable source.

vi) Apply algorithms to the obtained data to predict weather patterns and display the results on an LCD screen or any other display unit.

vii) Test the system over a period of time and compare the results with a local weather forecast to verify its accuracy.

Source Code:

sketch_mar30a | Arduino IDE 2.0.4

File Edit Sketch Tools Help

Select Board

sketch_mar30a.ino

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_BMP085.h>
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT);
Adafruit_BMP085 bmp;
#define SEALEVELPRESSURE_HPA (101500)
float simpleweatherdifference, currentpressure, predictedweather, currentaltitude;
void setup() {
  // put your setup code here, to run once:
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
if (!bmp.begin()) {
  Serial.println("Could not find a valid BMP085 sensor, check wiring!");
  while (1) {}
  }
}

void loop() {
  // put your main code here, to run repeatedly:
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);

  display.setCursor(0,5);
  display.print("BMP180");
  display.setCursor(0,19);
```

sketch_mar30a | Arduino IDE 2.0.4

File Edit Sketch Tools Help

Select Board

sketch_mar30a.ino

```

  /*prints BME180 pressure in Hectopascal Pressure Unit*/
  display.setCursor(0,30);
  display.print("P=");
  display.print(bmp.readPressure()/100.0F,1);
  display.println("hPa");

   /*prints BME180 altitude in meters*/
  display.setCursor(0,40);
  display.print("A=");
  display.print(bmp.readAltitude(SEALEVELPRESSURE_HPA),1);
  display.println("m");
  delay(6000);
  display.display();

 currentpressure=bmp.readPressure()/100.0;
 predictedweather=(101.3*exp(((float)(currentaltitude))/(-7900)));
 simpleweatherdifference=currentpressure-predictedweather;
 //display.clearDisplay();
 display.setCursor(0,50);
 if (simpleweatherdifference>0.25)
   display.print("SUNNY");

  if (simpleweatherdifference<=0.25)
  display.print("SUNNY/CLOUDY");

 if (simpleweatherdifference<-0.25)
  display.print("RAINY");
  display.display();
delay(2000);
}
```

Simulation:

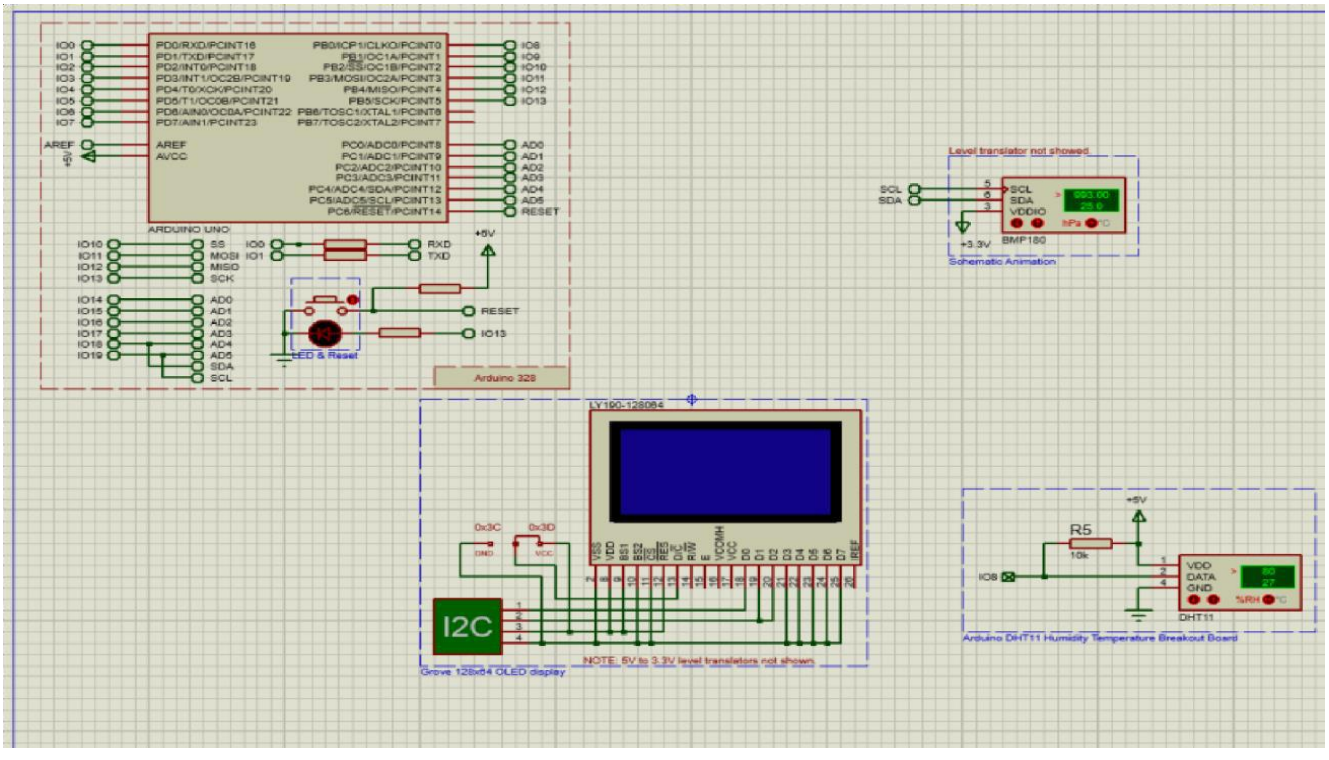


Figure: Schematic Capture before Run

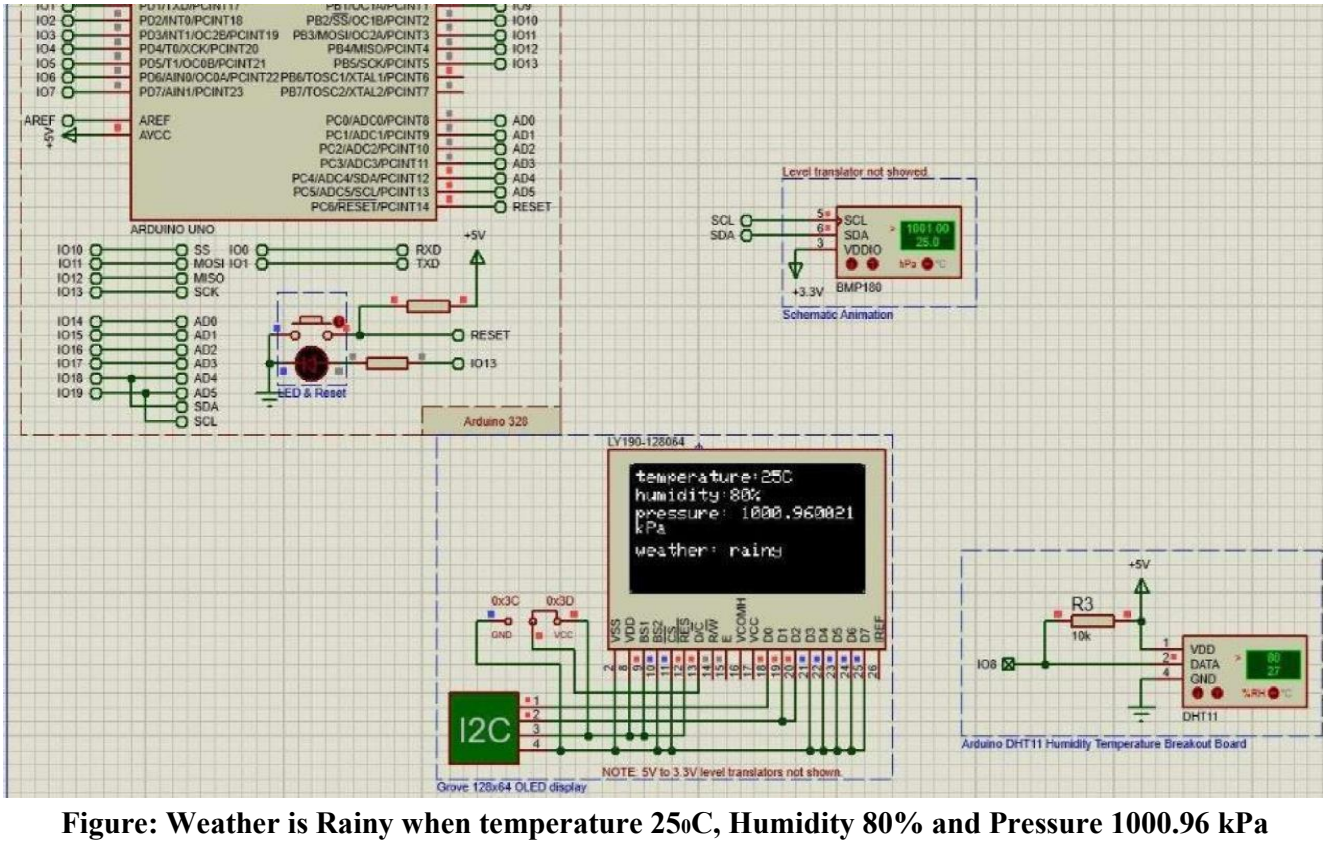


Figure: Weather is Rainy when temperature 25$_0$C, Humidity 80% and Pressure 1000.96 kPa

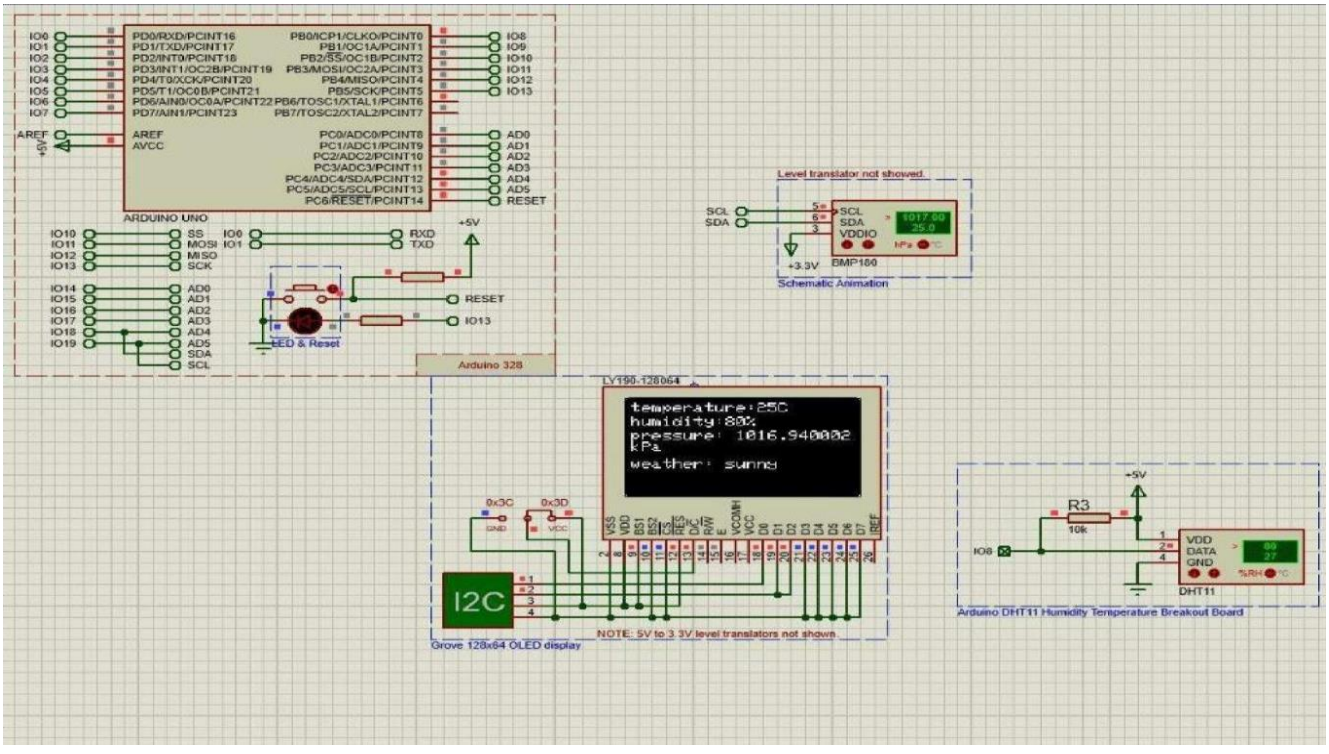
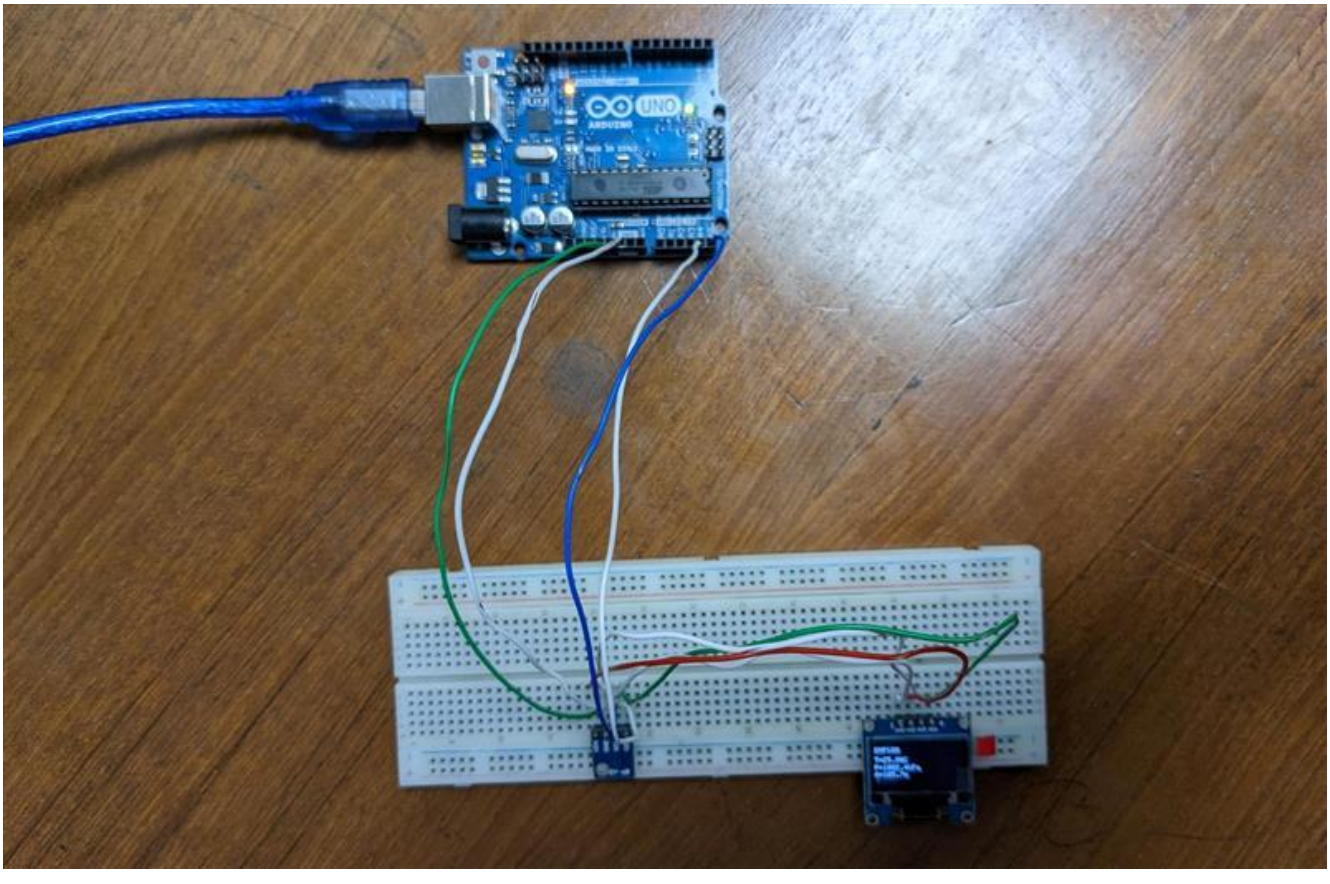
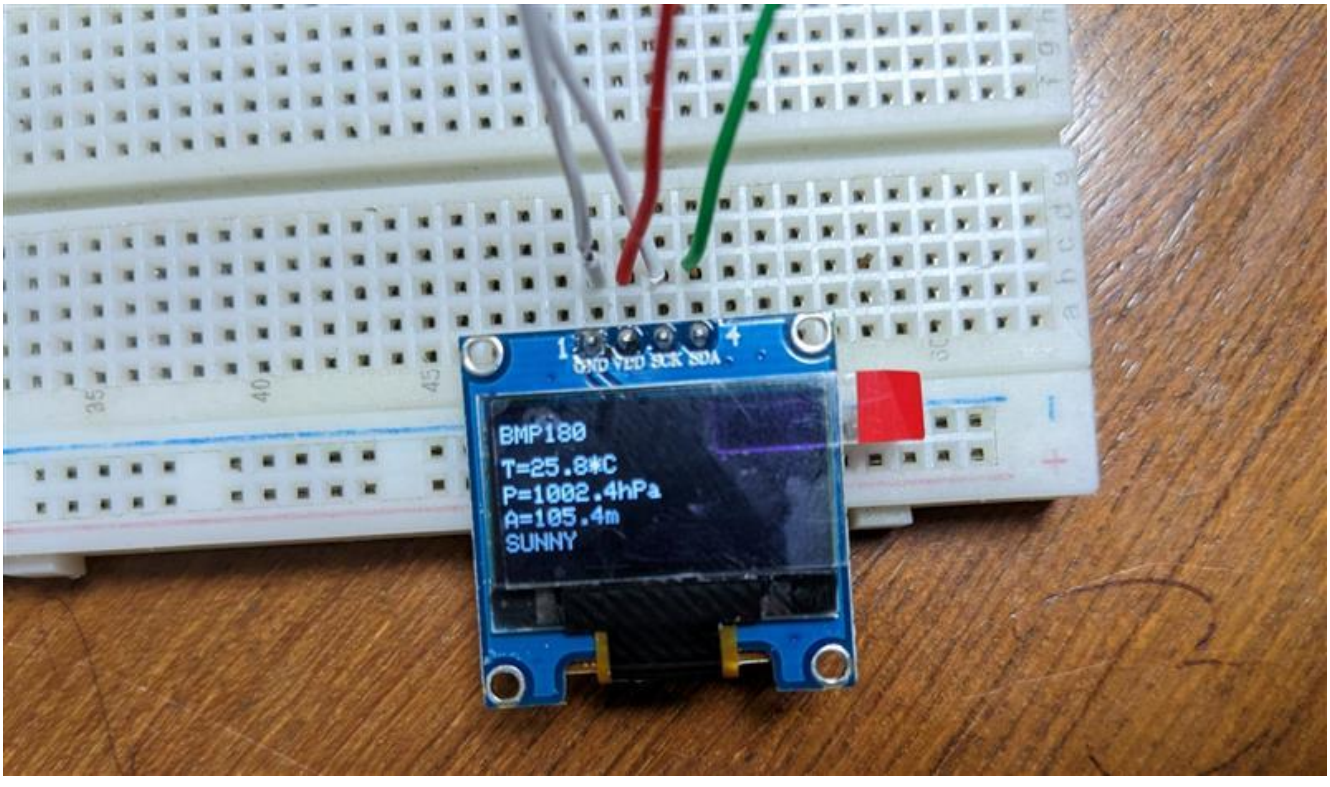


Figure: Weather is Sunny when temperature 25$_0$C, Humidity 80% and Pressure 1016.94 kPa

Lab Simulation:





Report:

All codes, scripts and proteus simulation of the blink program and traffic light system is attached above.

Discussions :

The main purpose of this lab experiment was to create weather forecast design with the help of software. Before starting the experiment, our respected teacher gave us an important discussion about weather forecast. Through this we get an idea of how it works, what its benefits are, or what kind of effect it can have if there is a problem. Then we start designing according to teacher's instructions. As a result, after finishing all the designs, when we went to run the simulation, there were some problems. After the problems, finally it got succeed after trying several times.

Conclusions:

The implementation of a weather forecast system using the ADC modules of an Arduino provides an excellent opportunity for us to familiarize it and with a microcontroller-based weather system. This system measures environmental parameters, such as temperature, pressure, and humidity, using the BMP180 or MPL115A sensor. The pressure changes due to weather are slow, requiring a few hours to determine the sloping of the pressure over time, with low pressure signaling worsening weather and high pressure indicating improving or clearing weather. Normalization is necessary to

compare the barometric pressure at different locations, and a simple algorithm can be used to predict the weather. Overall, the project provides a hands-on learning experience that combines electronics, data analysis, and weather forecasting.