AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Faculty of Engineering

Lab Report

Experiment # 08

Experiment Title: Implementation of a weather forecast system using the ADC modules of an Arduino.

Date of Perform:	7th May 2025	**Date of Submission:**	14th May 2025
Course Title:	Microprocessor and Embedded Systems Lab		
Course Code:	EEE4103	**Section:**	**R**
Semester:	Spring 2024-25	**Degree Program:**	BSc in CSE
Course Teacher:	**Prof. Dr. Engr. Muhibul Haque Bhuyan**		

Declaration and Statement of Authorship:

1. I/we hold a copy of this Assignment/Case Study, which can be produced if the original is lost/damaged.
2. This Assignment/Case Study is my/our original work; no part has been copied from any other student's work or any other source except where due acknowledgment is made.
3. No part of this Assignment/Case Study has been written for me/us by any other person except where such collaboration has been authorized.
   by the concerned teacher and is acknowledged in the assignment.
4. I/we have not previously submitted or am submitting this work for any other course/unit.
5. This work may be reproduced, communicated, compared, and archived to detect plagiarism.
6. I/we permit a copy of my/our marked work to be retained by the Faculty Member for review by any internal/external examiners.
7. I/we understand that Plagiarism is the presentation of the work, idea, or creation of another person as though it is your own. It is a form of cheating and is a very serious academic offense that may lead to expulsion from the University. Plagiarized material can be drawn from, and presented in, written, graphic, and visual forms, including electronic data, and oral presentations. Plagiarism occurs when the origin of the source is not appropriately cited.
8. I/we also understand that enabling plagiarism is the act of assisting or allowing another person to plagiarize or copy my/our work.

** Student(s) must complete all details except the faculty use part.*

** Please submit all assignments to your course teacher or the office of the concerned teacher.

Group # 05

Sl No	Name	ID	PROGRAM	SIGNATURE
1	Shahriar Hossain	22-48990-3	BSc in CSE	
2	Al Mubtasim	22-49002-3	BSc in CSE	
3	Adiba Tanzila	22-49012-3	BSc in CSE	
4	MD.Rakib Hasan	22-49029-3	BSc in CSE	
5	Md.Imdadul Hasan(Ayon)	22-49959-3	BSc in CSE	

Faculty use only

FACULTY COMMENTS	**Marks Obtained**	
	Total Marks	

Table of Contents

Marking Rubrics (to be filled by Faculty):

Level Category	Excellent [5]	Proficient [4]	Good [3]	Acceptable [2]	Unacceptable [1]	No Response [0]
Title and Objectives	Able to clarify the understanding of the lab, no issues are missing and formatting is good.	Able to clarify the understanding of the lab experiment, no issues are missing but its formatting is not good.	Able to clarify the understanding of the lab experiment, but a few issues are wrong, and its formatting is bad.	Able to clarify the understanding of the lab experiment, but it lacks a few important issues of the experiment without maintaining the format.	Unable to clarify the understanding of the lab experiment.	No Response/ copied from others/ identical submissions with gross errors/image file printed
Codes and Methods	Able to explain the experimental codes and simulation methods using Proteus very well.	Able to explain the experimental codes and simulation methods using Proteus but is not formatted well.	Able to explain the experimental codes but simulation method using Proteus is not explained well.	Presents the experimental codes but didn't explain simulation methods using Proteus clearly.	Presents the experimental codes but didn't explain simulation methods using Proteus.	
Results	Key results and images are there. Figures/Tables have all identifications and refer to them properly in the texts.	Key results and images are there. Figures/Tables have all identifications, such as the axis labels, numbers, and captions with a few minor errors; the texts refer them.	Key results and images are there. Figures/Tables lack a few identifications, such as the axis labels, numbers, and captions; the texts refer them.	Misses several key results and images. Figures/Tables lack identification, such as the axis labels, numbers, and captions; the texts don't refer them.	Major results, such as experimental and simulation results' images are not included. Figures and tables are poorly constructed or not presented.	
Discussion and Conclusion	Proper interpretation of results and summarizes the results to draw a conclusion, discusses its applications in real-life situations to connect with the report's conclusion.	Proper interpretation of results and summarizes the results to draw a conclusion but didn't discuss its applications in real-life situations to connect with the conclusion of the report.	Interpretation of results is presented. However, there is a disconnect between the results and discussion.	Misses the interpretation of key results. There is little connection between the results and discussion.	Very poor interpretation of the results. No connection between results and discussions.	
Question and Answer	Able to produce all questions' answers correctly maintaining the lab report format.	Able to produce all questions' answers but didn't maintain the lab report format.	Able to produce all questions' answers but wrong answers to a few questions.	Able to produce all questions' answers but wrong/missing answers to multiple questions.	Unable to produce all questions' answers and completely wrong answers.	
Comments						**Total Marks (25)**

Objectives:

The objectives of this experiment are to-

- Familiarize the students with the Micro-controller-based weather forecast system.
- Measure environmental parameters, such as temperature, pressure, and humidity.

Equipment List:

1) Arduino Uno Board
2) BMP180/MPL115A
3) Inches96 inch OLED 128X64
4) Breadboard
5) Jumper Wires

Circuit Diagram:

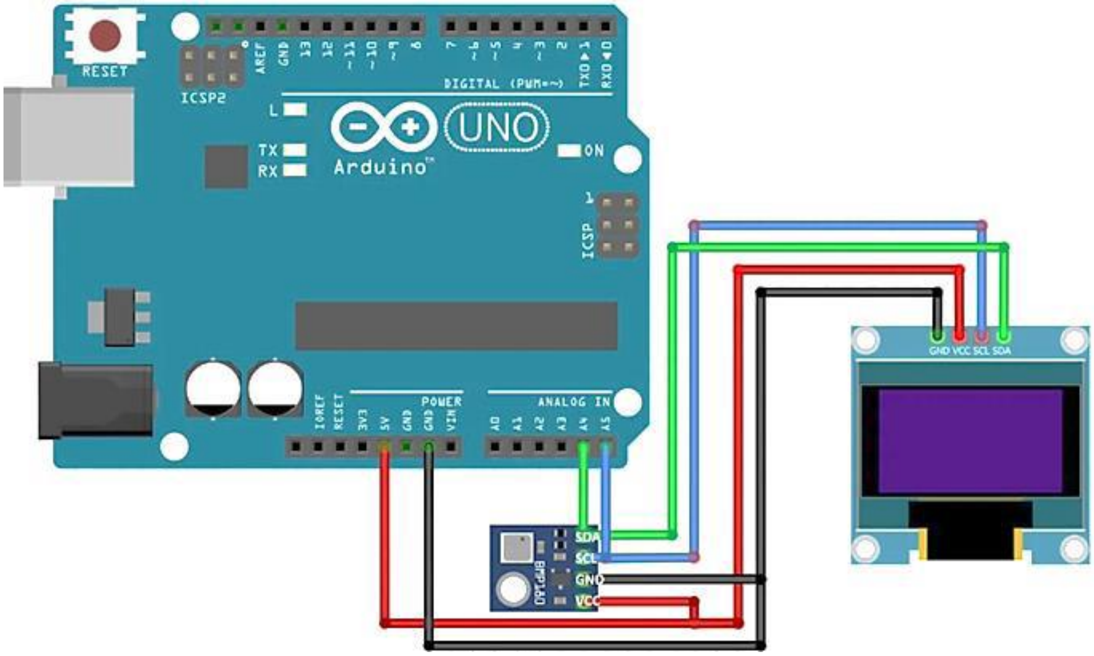


Fig : Arduino board’s pin connections with a weather sensor and an OLED (schematic diagram)

Experimental Output Results:

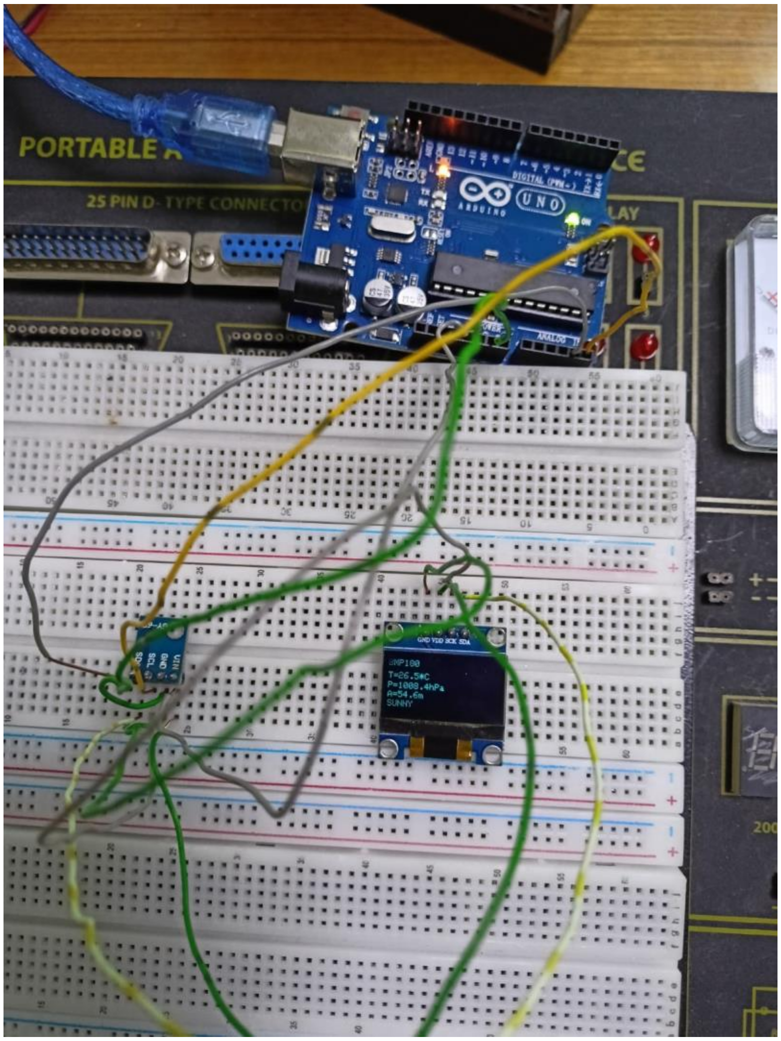
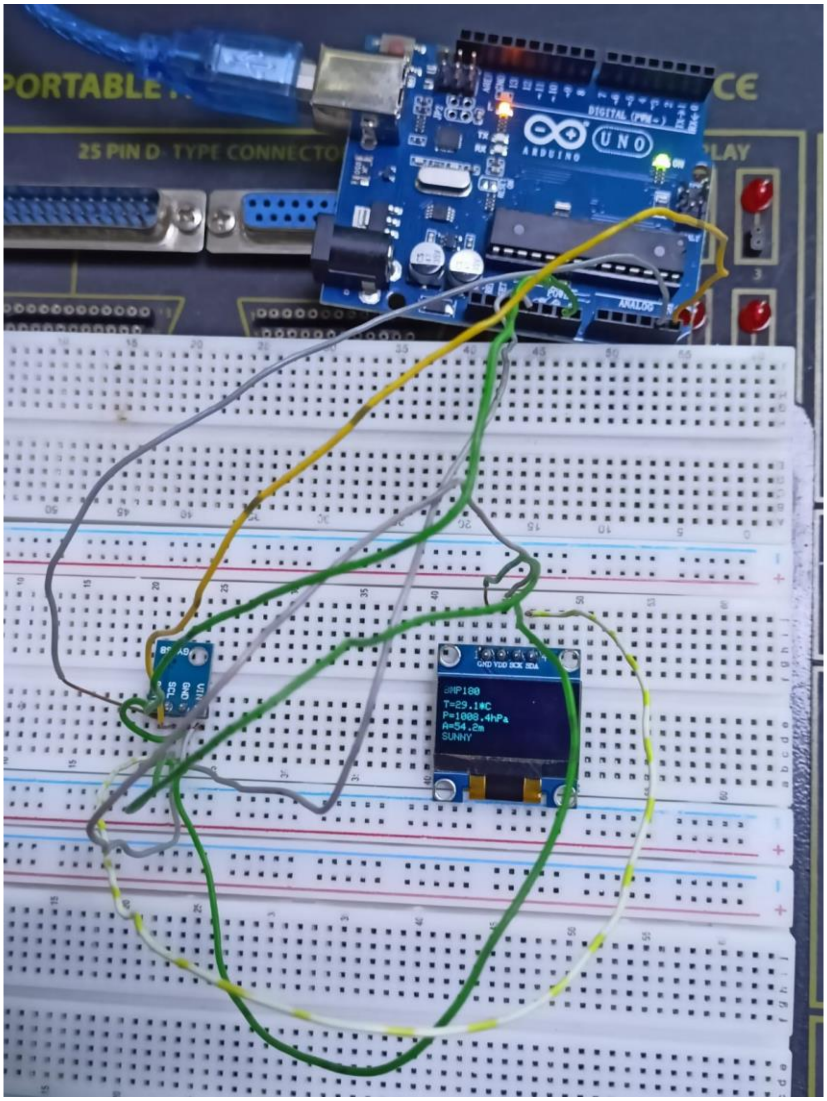


Fig. Screen Showing Low temperature



```
BMP180
T=29.1°C
P=1008.4hPa
A=54.2m
SUNNY
```

Fig. Screen Showing Medium temperature

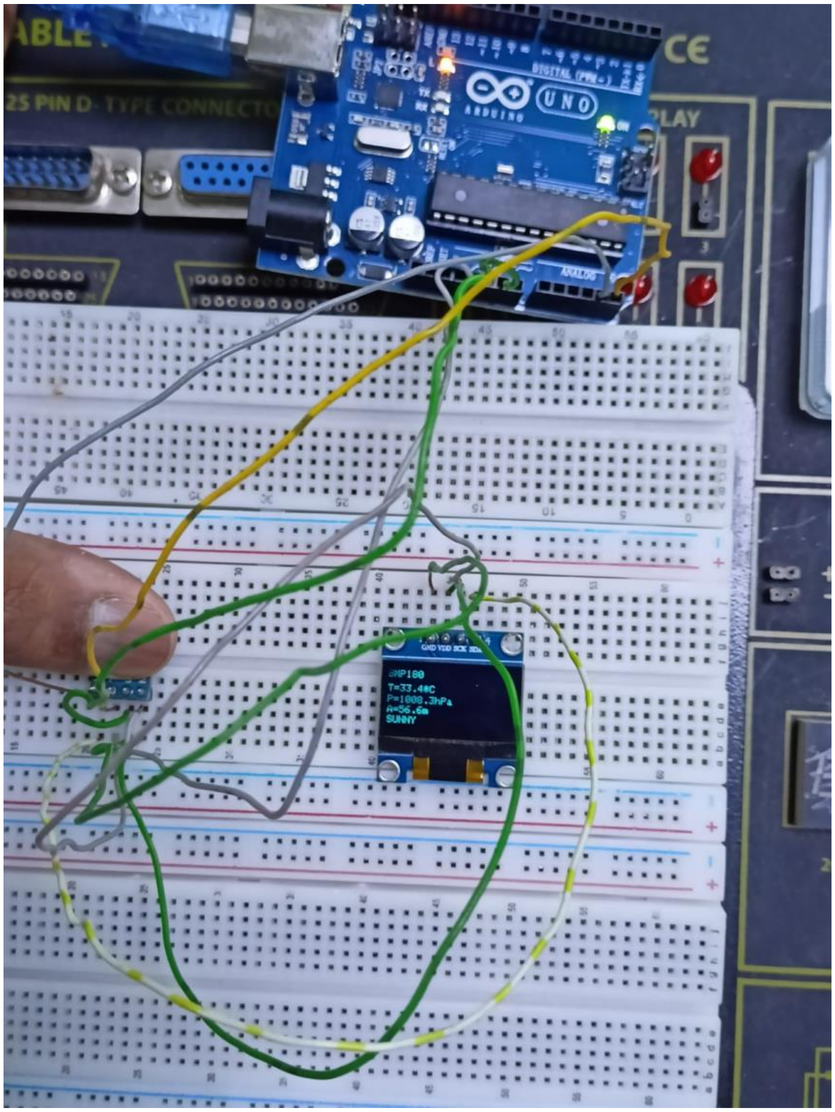
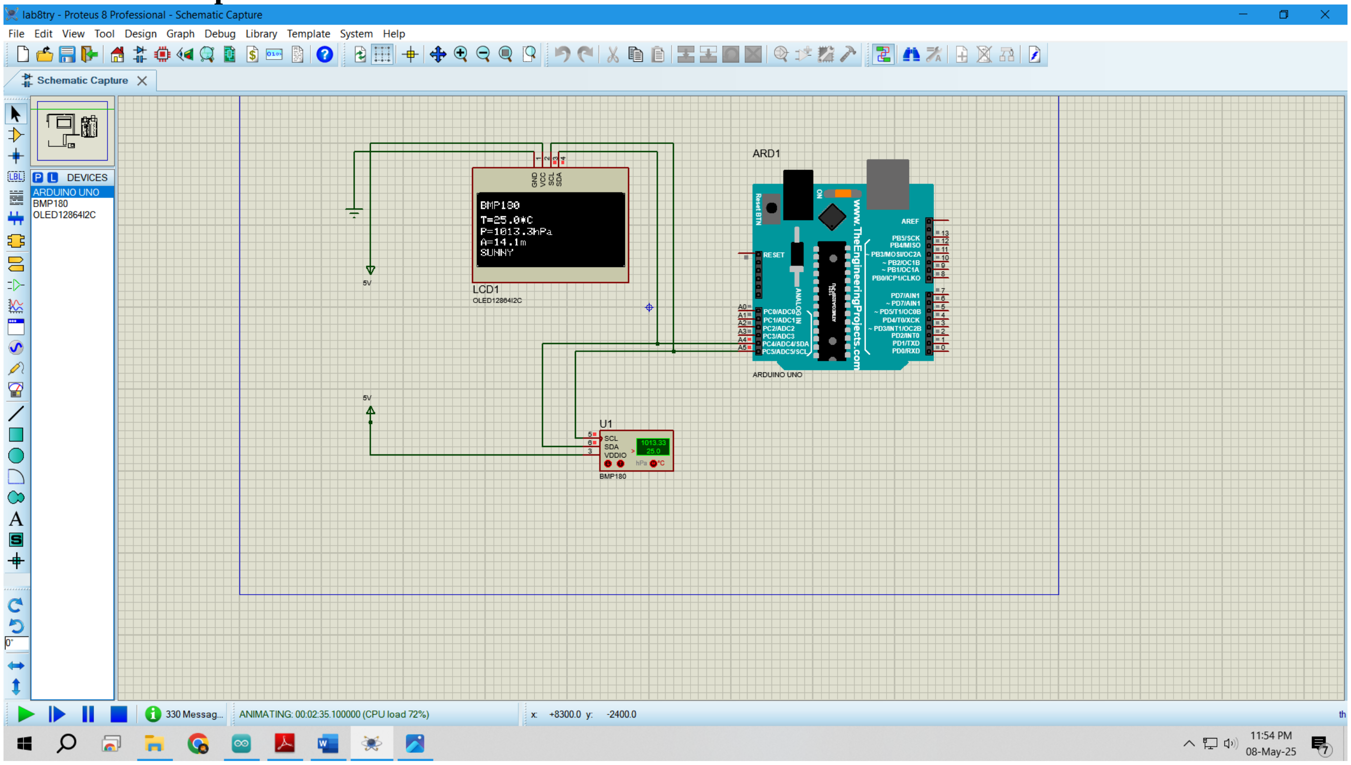


Fig. Screen Showing High temperature after increasing

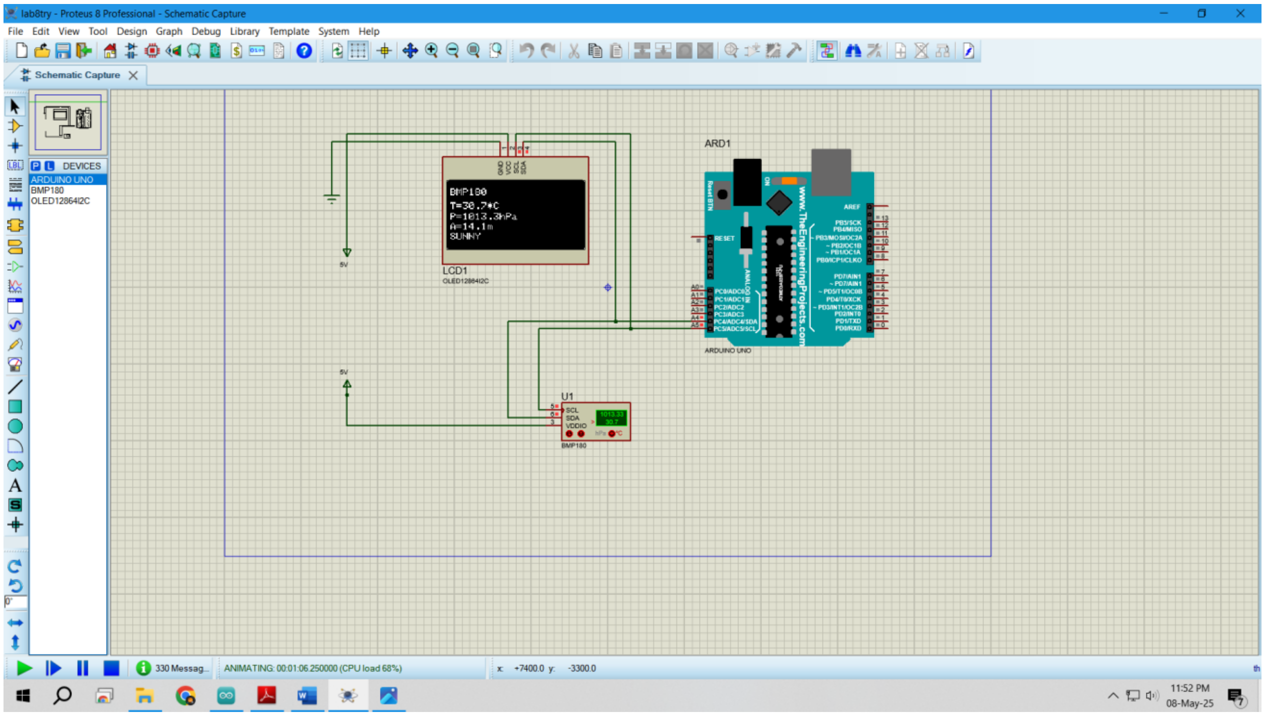
Simulation Output Results:



```
BMP180
T=25.0*C
P=1013.3hPa
A=14.1m
SUNNY
```

ANIMATING: 00:02:35.100000 (CPU load 72%)

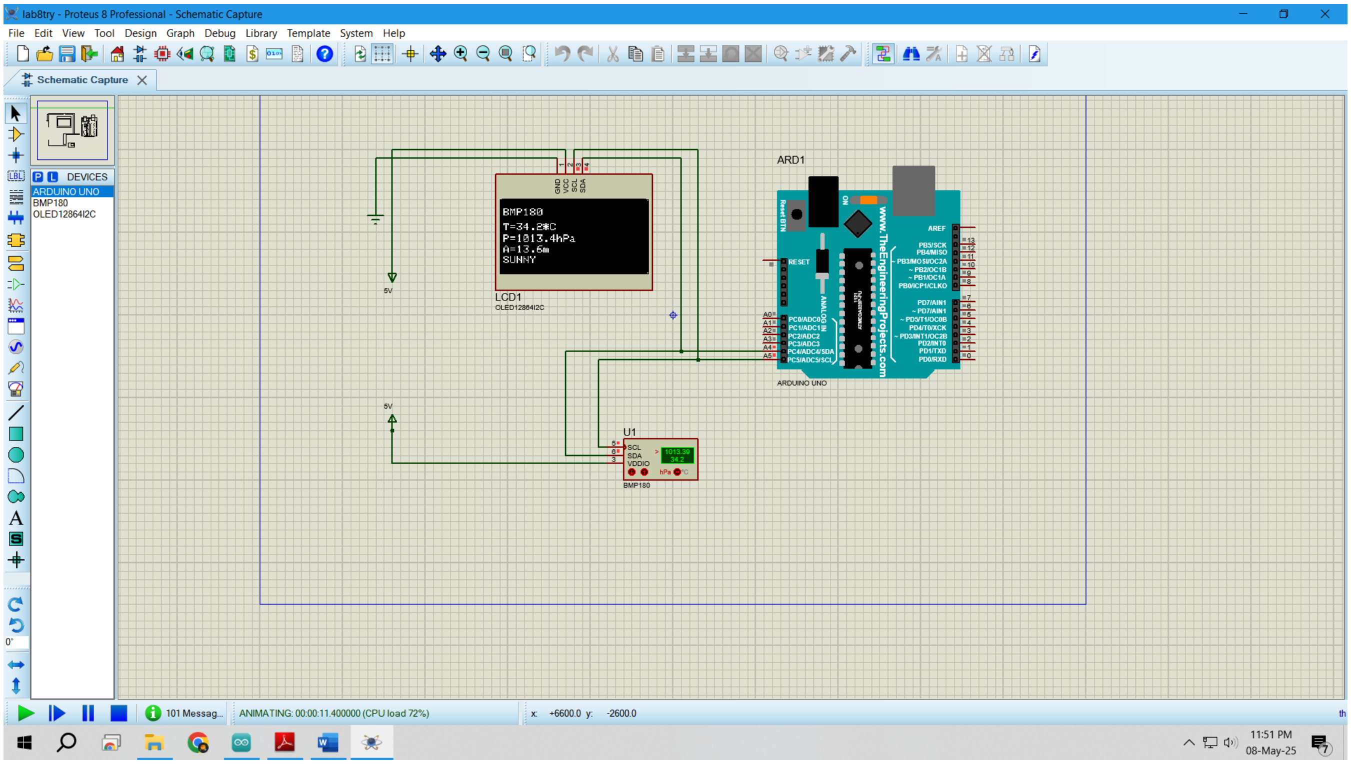
Fig: Simulation at Low Temperature



```
BMP180
T=30.7*C
P=1013.3hPa
A=14.1m
SUNNY
```

ANIMATING: 00:01:06.250000 (CPU load 68%)

Fig: Simulation at Medium Temperature



```
BMP180
T=34.2*C
P=1013.4hPa
A=13.6m
SUNNY
```

Fig: Simulation at High Temperature

Explaination:

- Proteus simulation software was used for circuit simulation.
- Arduino IDE was used to write and compile the code.
- The circuit was first designed in Proteus.
- Arduino UNO, OLED, and BMP180 were added from the component library.
- A HEX file was generated from the Arduino IDE after compiling the code.
- The HEX file was then imported into the Arduino UNO in Proteus.
- The simulation was run to observe the temperature, pressure, and altitude outputs.
- These parameters were varied to observe corresponding changes in the display.

Answers to the Questions in the Lab Manual:

```
#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);
display.print("T=");
display.print(bmp.readTemperature(),1);
display.println("*C");
/*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);
}
```

Explaination:

• #include <SPI.h>, <Wire.h>, <Adafruit_GFX.h>, <Adafruit_SSD1306.h>, <Adafruit_BMP085.h>

o These libraries enable SPI and I2C communication, OLED display control, and BMP180 sensor functionality.

- #define SCREEN_WIDTH 128 and #define SCREEN_HEIGHT 64
  o Define the width and height of the OLED screen in pixels.
- Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT);
  o Creates an OLED display object with 128×64 resolution.
- Adafruit_BMP085 bmp;
  o Creates an object to communicate with the BMP180/BMP085 sensor.
- #define SEALEVELPRESSURE_HPA (101500)
  o Sets the reference sea-level pressure in Pascals (1015.00 hPa) for accurate altitude calculations.
- float simpleweatherdifference, currentpressure, predictedweather, currentaltitude;
  o Declares variables to store pressure, altitude, and weather prediction calculations.
- void setup()
  o display.begin(SSD1306_SWITCHCAPVCC, 0x3C); initializes the OLED display with the I2C address 0x3C.
  o bmp.begin(); initializes the BMP180 sensor for temperature, pressure, and altitude readings.
  o If the BMP180 sensor is not detected, an error is printed to the Serial Monitor, and the program halts.
- void loop()
  o display.clearDisplay(); clears the OLED screen before displaying new values.
  o display.setTextSize(1); and display.setTextColor(SSD1306_WHITE); set display text properties.
  o display.setCursor(x, y); positions the cursor on the OLED for printing text.
  o bmp.readTemperature(); reads and displays temperature in Celsius.
  o bmp.readPressure()/100.0F; reads and displays atmospheric pressure in hPa (hectopascals).
  o bmp.readAltitude(SEALEVELPRESSURE_HPA); calculates and displays current altitude in meters.
  o delay(6000); waits for 6 seconds before continuing (helps with readability on OLED).
  o display.display(); sends the data to the OLED for visual output.
- Weather Prediction Logic
  o currentpressure = bmp.readPressure()/100.0; reads current atmospheric pressure.
  o predictedweather = 101.3 * exp(currentaltitude / -7900); estimates expected pressure at altitude.
  o simpleweatherdifference = currentpressure - predictedweather; calculates pressure deviation.
  o if (simpleweatherdifference > 0.25) → prints "SUNNY".
  o if (simpleweatherdifference <= 0.25) → prints "SUNNY/CLOUDY".
  o if (simpleweatherdifference < -0.25) → prints "RAINY".
  o display.display(); updates the screen with the weather prediction.
  o delay(2000); adds a 2-second pause before the next update.

Discussions: In this experiment, a BMP180 sensor was used to measure temperature, pressure, and altitude, with the data displayed on an OLED screen via I2C communication. The sensor's functionality and pin configuration were thoroughly studied before interfacing it with an Arduino Uno. The OLED was also configured appropriately to show real-time values. The system was tested both physically and in simulation (Proteus), and the outputs were compared. Minor discrepancies were observed between simulated and real readings, likely due to sensor precision limits and human error. Overall, the hardware and software performed as expected, and the experiment successfully met its objectives.

Reference(s):

[1] Arduino IDE, https://www.arduino.cc/en/Main/Software accessed on May 3, 2019.
[2] Arduino and Proteus Library, https://etechnophiles.com/add-simulate-ultrasonic-sensorproteus-2018-edition/ accessed on May 3, 2019.
[3] Ultrasonic Distance Sensor in Arduino with the TinkerCad https://www.instructables.com/id/Ultrasonic-Distance-Sensor-Arduino-Tinkercad/ accessed on May 3, 2019.