- Parts used :

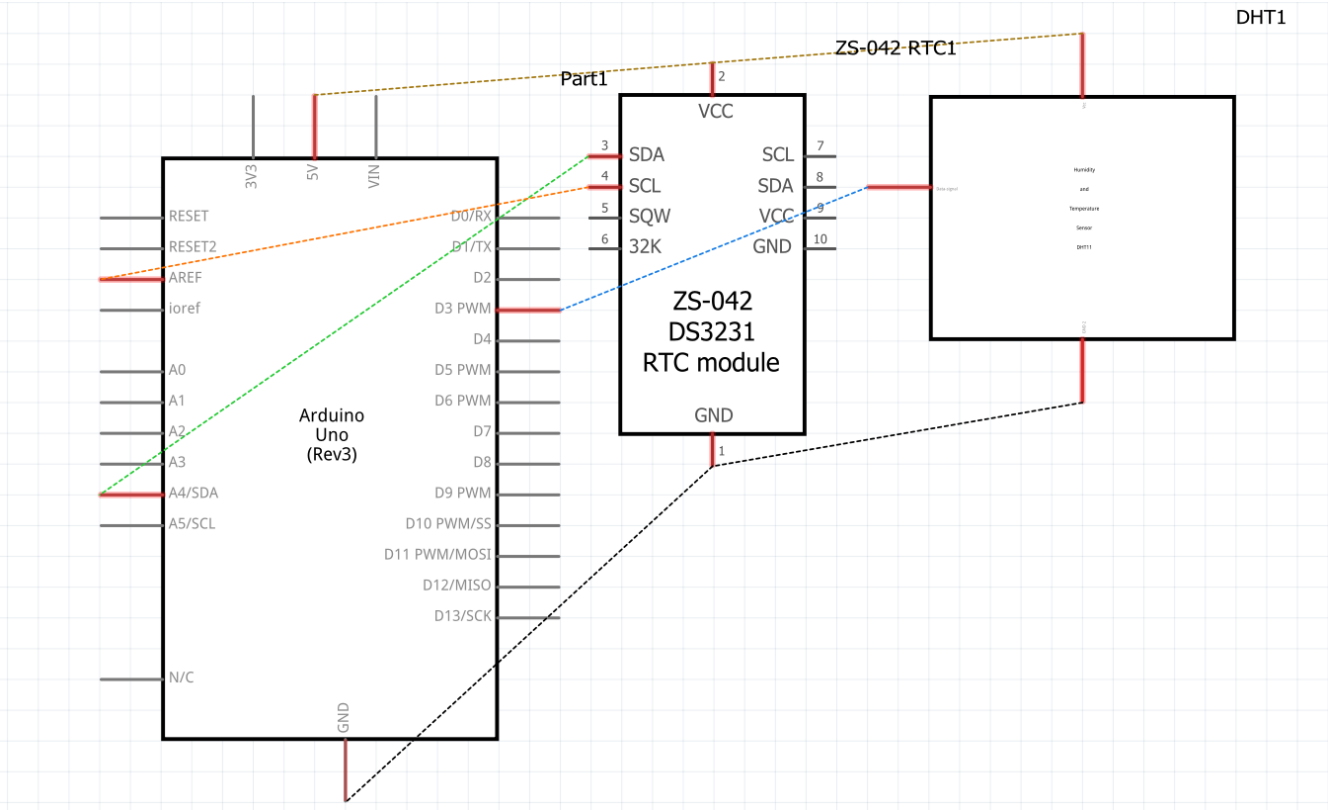
- 1x Breadboard
- Jumper Wires
- Arduino Uno R3
- DHT11 Temperature and Humidity Sensor
- DS3231 Real Time Clock Module

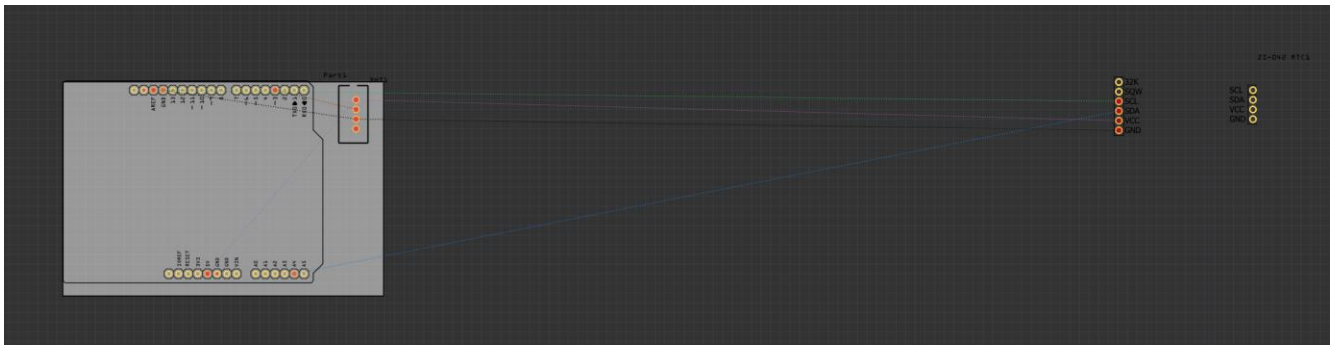
- Software used :

- Arduino – IDE
- Visual Studio - Python
- CoolTerm

Project Description :

In this comprehensive project, we create a sophisticated real-time temperature and humidity monitoring system using an Arduino board, A DS3231 real-time clock (RTC) module, And a DHT11 temperature and humidity sensor. The primary objective of this project is to collect precise environmental data, timestamp it with the DS3231 RTC and DHT11, and log it into a text file using CoolTerm every 15 minutes.





Algorithm and Program source code:

A) Algorithm :

1. **Include necessary libraries:**
    - Include the Wire.h library for I2C communication.
    - Include the DS3231.h library for the DS3231 RTC module.
    - Include the SimpleDHT.h library for the DHT11 temperature and humidity sensor.

2. **Define variables:**
    - Declare an integer variable 'pinDHT11' and set it to 2. This variable represents the digital pin to which the DHT11 sensor is connected.
    - Create an instance of the 'SimpleDHT11' class named 'dht11' to interface with the DHT11 sensor.
    - Create an instance of the 'RTClib' class named 'myRTC' to interface with the DS3231 RTC module.
    - Declare a boolean variable' dataPrinted' and initialize it to false. This variable is used to track whether data has been printed.

3. **Setup function:**
    - Initialize serial communication with a baud rate of 57600.
    - Initialize the I2C communication using '**Wire.begin()**'.
    - Add a delay of 1 second (1000 milliseconds) to allow for the initialization of serial communication and I2C.

4. **Loop function:**
    - Get the current date and time from the DS3231 RTC module and store it in the '**now'** variable.
    - Retrieve the current minute and second values from the 'now' variable.

5. **Check if it's time to print data:**
    - If the current minute is 0, 15, 30, or 45 and the current second is 0, enter this conditional block.
    - Check if '**dataPrinted'** is false (indicating that data hasn't been printed for this minute).

6. **Print date and time:**
    - Print the current date and time in the format "YYYY/MM/DD HH:MM:SS".
    - Read temperature and humidity data from the DHT11 sensor and store it in variables '**temperature'** and '**humidity'**.
    - If the DHT11 read operation fails, print an error message and return.

7. **Print temperature and humidity:**
    - Print the temperature in Celsius and humidity as a percentage.

      Set '**dataPrinted'** to true to indicate that data has been printed for this minute.

8. **If it's not time to print data:**
    - f the current minute is not one of the specified values or the current second is not 0, set **dataPrinted** to false to allow data to be printed in the next valid minute.

9. **Add a delay of 1 second (1000 milliseconds) before the next iteration of the loop.**

The code continuously checks the time and prints date and temperature and humidity data at 0, 15, 30, and 45 minutes past each hour, if the second is 0.

B) Program Source Code :

```
#include <Wire.h>
#include <DS3231.h>
#include <SimpleDHT.h>

int pinDHT11 = 2;
SimpleDHT11 dht11;
RTClib myRTC;
bool dataPrinted = false;

void setup()
{
  Serial.begin(57600);
  Wire.begin();
  delay(1000);
}

void loop()
{
  DateTime now = myRTC.now();
  int currentMinute = now.minute();
  int currentSecond = now.second();

  if ((currentMinute == 0 || currentMinute == 15 || currentMinute == 30 || currentMinute == 45) && currentSecond == 0)
  {
    if (!dataPrinted)
    {
      Serial.print("The Date And Time = ");
      Serial.print(now.year(), DEC);
      Serial.print('/');
      Serial.print(now.month(), DEC);
      Serial.print('/');
      Serial.print(now.day(), DEC);
      Serial.print(' ');
      Serial.print(now.hour(), DEC);
      Serial.print(':');
      Serial.print(now.minute(), DEC);
      Serial.print(':');
      Serial.print(now.second(), DEC);
      Serial.println();
      byte temperature = 0;
      byte humidity = 0;
      byte data[40] = {0};
      if (dht11.read(pinDHT11, &temperature, &humidity, data))
      {
        Serial.println("Read DHT11 failed.");
        return;
      }
      Serial.println();
      Serial.print("Temperature : ");
      Serial.print((int)temperature);
      Serial.print(" *C, ");
      Serial.print((int)humidity);
      Serial.println(" %");
      Serial.println();
      dataPrinted = true;
    }
  }
  else
  {
    dataPrinted = false;
  }
  delay(1000);
}
```

Program description :

1. Libraries Used:

- **Wire.h: Used for I2C communication.**
- **DS3231.h: Provides functions to interface with the DS3231 RTC module.**
- **SimpleDHT.h: Allows communication with the DHT11 temperature and humidity sensor.**

2. Hardware Setup:

- **The program assumes the DS3231 RTC module is connected and properly functioning.**
- **The DHT11 sensor is connected to digital pin 2 of the Arduino board.**

3. Program Logic:

- **The program initializes serial communication with a baud rate of 57600 and I2C communication.**
- **It enters the main loop where it continuously checks the current minute and second of the RTC.**

4. Data Printing:

- **If the current time matches specific conditions (when the minute is 0, 15, 30, or 45, and the second is 0) and dataPrinted is false:**
    - **It prints the current date and time in the format "YYYY/MM/DD HH:MM:SS".**
    - **It reads temperature and humidity data from the DHT11 sensor.**
    - **If the DHT11 reading fails, it prints an error message.**
    - **The temperature is printed in Celsius, and humidity is printed as a percentage.**
    - **It sets dataPrinted to true, indicating that data has been printed for this minute.**
- **If the current time doesn't match the conditions, dataPrinted is set to false, allowing data to be printed in the next valid minute.**

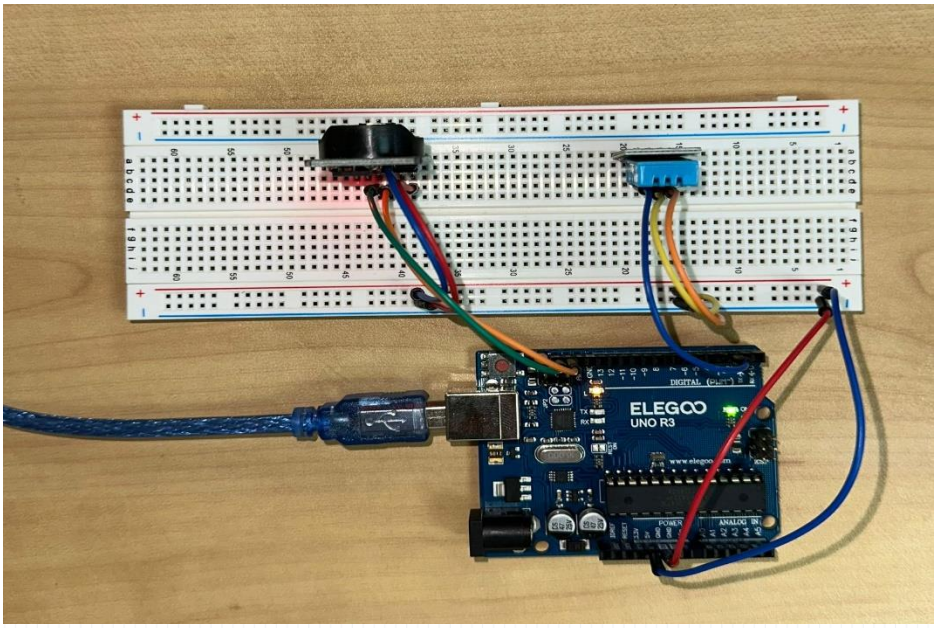
5. Delay:

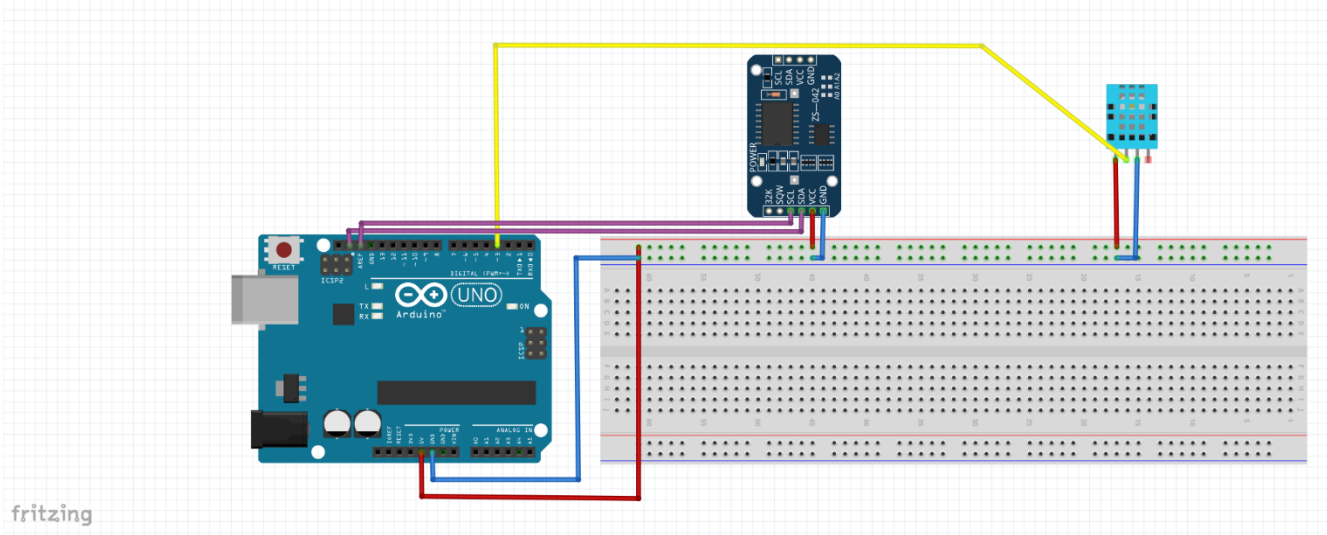
- **The program includes a delay of 1 second between iterations of the loop, ensuring that the RTC and DHT11 sensor are not read too frequently.**

6. Output:

- **The program continuously prints the current date, time, temperature, and humidity to the serial communication interface.**
- **We used a CoolTerm terminal program, to connect to the Arduino board and view the output. The set of CoolTerm will be same baud rate as specified in the program, which is 57600 in this case.**
- **Data is printed once per 15 minute when the specified conditions are met, and it will be visible in your terminal program's interface.**

Implementation In the Laboratory





Text Note

Project3_CoolTerm_Test1.txt

ملف تحرير عرض

```
The Date And Time = 2023/10/30 10:45:0
% Temperature : 20 *C, 65
The Date And Time = 2023/10/30 11:0:0
% Temperature : 20 *C, 65
```

Python source code:

Python Script:

- **Importing Modules:** The Python script imports the necessary modules, including '**serial**', '**time**', and '**datetime**'.
- **Serial Connection:** The script establishes a serial connection to the Arduino using the '**serial**' module. It specifies the serial port and baud rate for communication.
- **Data Reception:** The script continuously listens for data from the Arduino using' **ser.readline()**'. It decodes the received data, strips any leading/trailing whitespace, and stores it in the '**arduino_data**' variable.
- **Timestamp:** The current timestamp is generated using the '**datetime**' module and formatted as 'YYYY-MM-DD HH:MM:SS'. This timestamp represents when the data was received.
- **Data Logging:** The script appends the timestamp and received data to a text file named 'data.txt' using the 'a' mode of file handling. This allows data to be logged over time.
- **Real-Time Monitoring:** The script also continuously prints the received data and timestamp to the console, providing real-time monitoring of the data as it arrives.

The Python script should be executed on the computer, specifying the correct serial port ('COM5' in this case) and baud rate (57600).