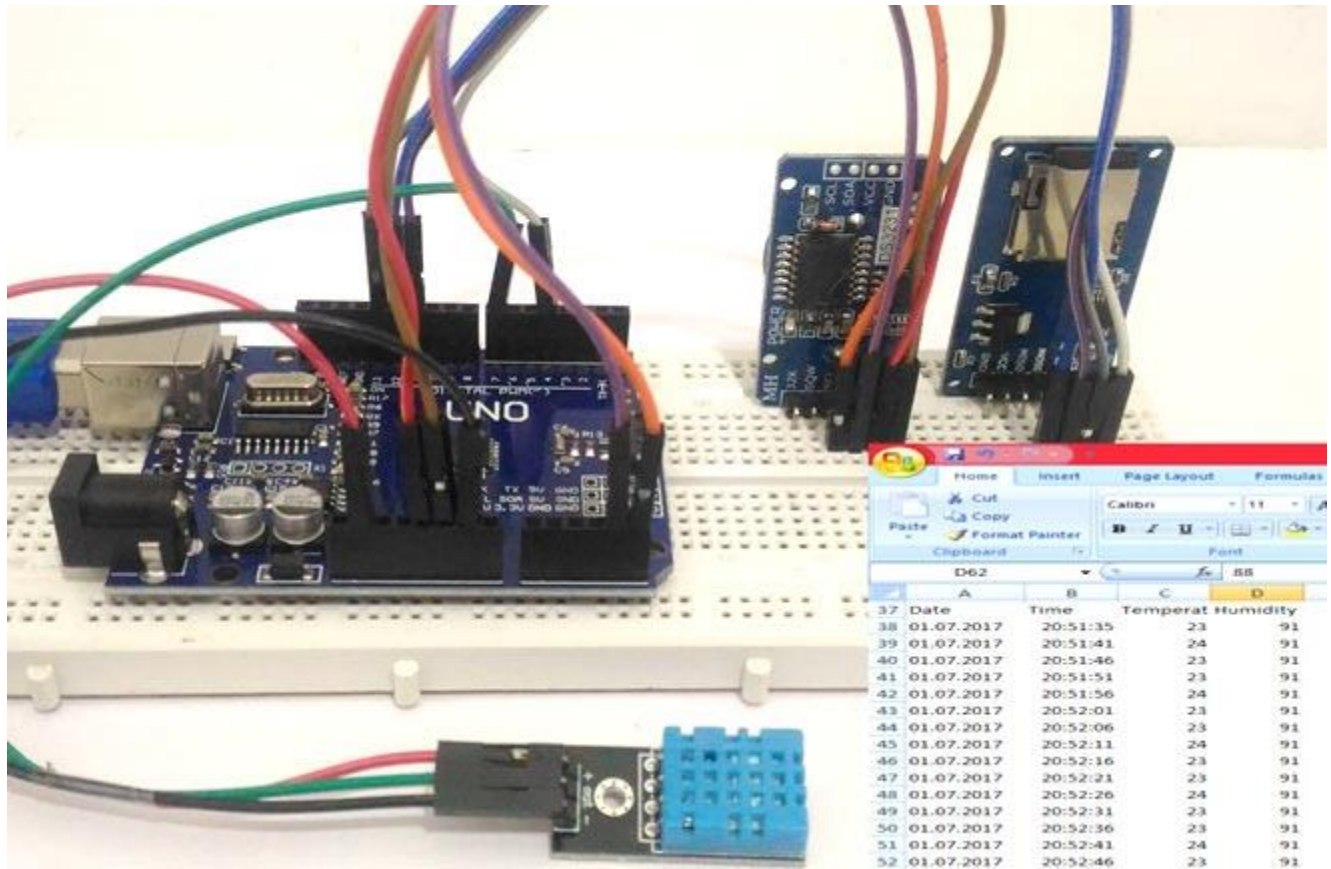


Arduino Data Logger (Log Temperature, Humidity, Time on SD Card and Computer)



Arduino Temperature Data Logger using DHT11

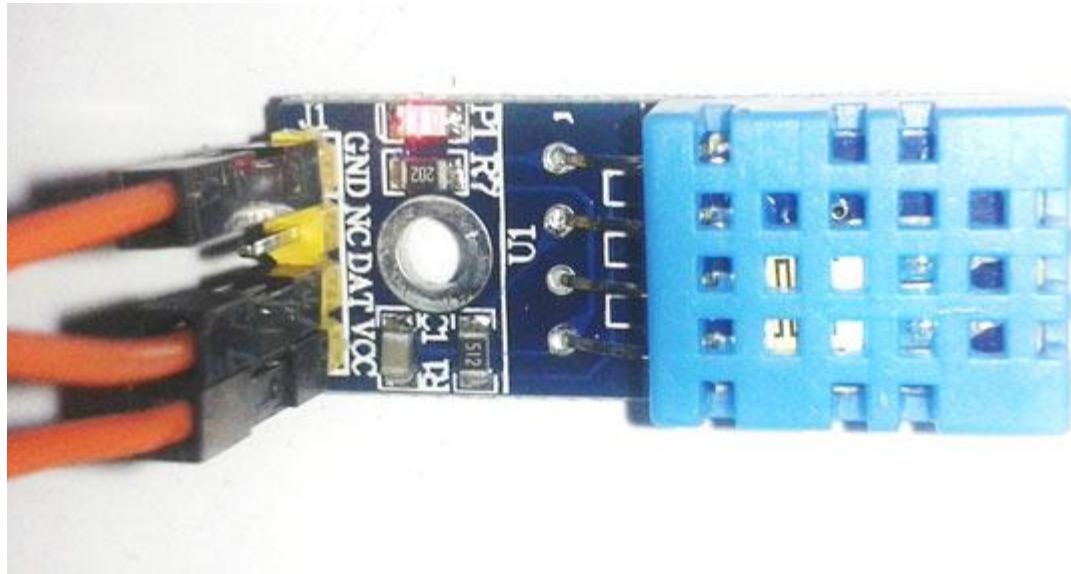
As Engineers/Developers we always rely upon the data collected to design or improve a system. Recording data and analyzing them is a common practice in most of the industries, here we are building **Arduino Data Logger Project** where we will learn how we can log data at a specific interval of time. We will use an Arduino board to read some data (here temperature, humidity, date and time) and save them on a SD card and the computer simultaneously.

The data saved can be easily opened in an Excel Sheet for further analyses. To maintain the date and time we will use the famous **RTC module DS3231** and to get the Temperature and Humidity we will use the **DHT11 Sensor**. At the end of the project you will learn

1. How to log data into SD card with Date, Time and sensor values.
2. How to write data directly to Excel Sheet on PC via serial communication.

Materials Required:

1. Breadboard
2. Arduino UNO (any Arduino board)
3. DHT11 Temperature sensor
4. DS3231 RTC module
5. SD card module
6. SD card
7. Connecting wires
8. Computer/Laptop

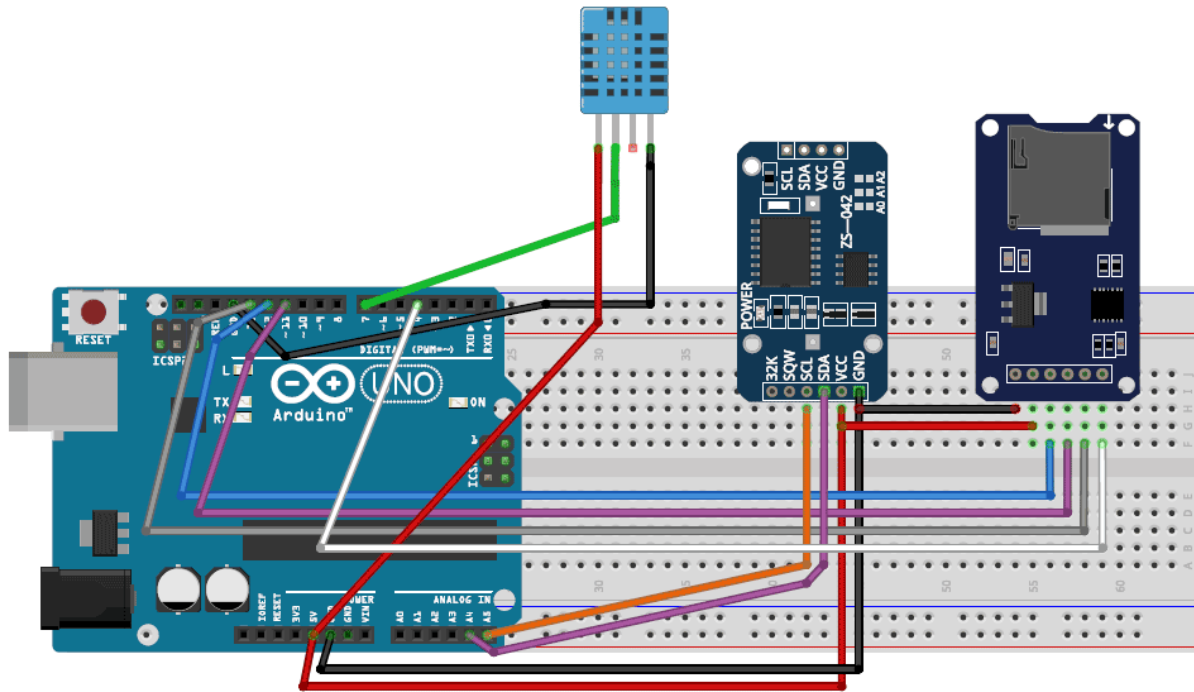


Temperature and Humidity Sensor

DHT11

Circuit Diagram:

The circuit Diagram for this **Arduino Temperature Logger Project** is shown below.



fritzing

As shown in the circuit diagram the connections are very simple since we have used them as modules we can directly build them on a breadboard. The connections are further classified in the table below

Arduino Pin	Module Pin
Temperature Sensor – DHT11	
Vcc	5V
Gnd	Gnd
Nc	Nc
Out	Pin 7
RTC module DS3231	
Vcc	5V
Gnd	Gnd

SCL	Pin A5
SDA	Pin A4
SD card Module	
Vcc	5V
Gnd	Gnd
MISO	Pin 12
MOSI	Pin 11
SCK	Pin 13
CS	Pin 4

You can replace the DHT11 temperature sensor with any of your sensor from which you need to log the values. You can check [LM35 with Arduino to read temperature](#).

The RTC module DS3231 is interfaced with Arduino using the I2C communication (SCL, SDA) and the SD card module is interfaced using the SPI Communication (MISO, MOSI, SCK, CS). The pins 4 and 7 are defined as the CS pin and output pin by Arduino program, you can change them to any other pin if required. We previously [interfaced SD card with Arduino](#) in Music player project.

Arduino Program Explanation:

We have to write the Arduino program which can do the following.

1. Read data from DTH11 Sensor (or any other data that you wish to log).
2. Initialize the I2C bus to read data from RTC module.
3. Initialize the SPI bus to interface the SD card module with Arduino.
4. Store the Date, Time, Temperature and Humidity into the SD card.
5. Store the Date, Time, Temperature and Humidity on a Excel Sheet running on a computer/Laptop.

The above steps might sound complicated but they are very easy since we have the libraries to do the hard job for us. You have to download the following two libraries

1. [DHT11 Sensor Library](#) from GitHub
2. [DS3231 RTC module library](#) from Rinky-Dink Electronics

Once you have downloaded the library add them to your Arduino IDE by following

Sketch->Include Library -> Add .ZIP Library

To feed the data from Arduino lively into an Excel sheet on computer we will also need to install software called [PLX-DAQ](#) provided by Parallax Inc. Follow the link to download the file and install them based on your operating system. This should have created a folder named PLS-DAQ on your desktop. We will take care of it later in our working section.

Now after adding both libraries and after installing the software, you can use the **Complete Code**(given at bottom of tutorial) and upload them to your Arduino. I have tried my best to keep the code as simple as possible and the explanations are also given through comment sections. Further, I will explain the important segments below.

1. Reading Data from DS3231:

DS3231 is a RTC (Real Time Clock) module. It is used to maintain the date and time for most of the Electronics projects. This module has its own coin cell power supply using which it maintains the date and time even when the main power is removed or the MCU has gone through a hard reset. So once we set the date and time in this module it will keep track of it always.

Using this module is very easy because of the library provided by Arduino.

```
// Init the DS3231 using the hardware interface
DS3231 rtc(SDA, SCL);
void Initialize_RTC()
{
    // Initialize the rtc object
    rtc.begin();

    //#### the following lines can be uncommented to set the date and time for the first
    time###
    /*
    rtc.setDOW(FRIDAY);    // Set Day-of-Week to SUNDAY
    rtc.setTime(18, 46, 45);    // Set the time to 12:00:00 (24hr format)
    rtc.setDate(6, 30, 2017);    // Set the date to January 1st, 2014
    */
}
```

Note: When using this module for the first time you have to set the date and time. It can be done by simply removing the comments as mentioned above and writing the date and time. Make sure you comment them back and upload it, else each time you run the board the date and time will be set again. You can also use [RTC IC DS1307 for reading the time with Arduino](#).

2. Reading Data from DHT11:

DHT11 is a Temperature and Humidity sensor. It sends the values of temperature and humidity as an 8-bit data serially through the output pin of the module. The library reads this data by using the software serial function of the Arduino.

```
#define DHT11_PIN 7 //Sensor output pin is connected to pin 7

dht DHT; //Sensor object named as DHT

void Read_DHT11()
{
  int chk = DHT.read11(DHT11_PIN);
}
```

Here I have connected the output pin to pin 7 as example you can choose any pin that supports Software Serial. Calling *DHT.read(pin number)*; will read the value of temperature and humidity and store it in the parameter *DHT.temperature* and *DHT.Humidity* respectively. Also check this [DHT11 based Arduino Temperature Measurement](#).

3. Initializing the SD card module:

```
void Initialize_SDcard()
{
  // see if the card is present and can be initialized:
  if (!SD.begin(chipSelect)) {
    Serial.println("Card failed, or not present");
    // don't do anything more:
    return;
  }

  // open the file. note that only one file can be open at a time,
  // so you have to close this one before opening another.
  File dataFile = SD.open("LoggerCD.txt", FILE_WRITE);
  // if the file is available, write to it:
```

```

    if (dataFile) {
        dataFile.println("Date,Time,Temperature,Humidity"); //Write the first row of the
        excel file
        dataFile.close();
    }
}

```

Using an SD card with Arduino is easy because of the SD card library which will be added to the Arduino IDE by default. In the SD card initialize function we will create a text file named "LoggerCD.txt" and write the first row of our content. Here we separate the values by using a "," as a delimiter. Meaning when a comma is placed it means we have to move to the next cell in the Excel sheet.

4. Writing Data to SD card

```

void Write_SDcard()
{
    // open the file. note that only one file can be open at a time,
    // so you have to close this one before opening another.
    File dataFile = SD.open("LoggerCD.txt", FILE_WRITE);

    // if the file is available, write to it:
    if (dataFile) {
        dataFile.print(rtc.getDateStr()); //Store date on SD card
        dataFile.print(","); //Move to next column using a ","

        dataFile.print(rtc.getTimeStr()); //Store date on SD card
        dataFile.print(","); //Move to next column using a ","

        dataFile.print(DHT.temperature); //Store date on SD card
        dataFile.print(","); //Move to next column using a ","

        dataFile.print(DHT.humidity); //Store date on SD card
        dataFile.print(","); //Move to next column using a ","

        dataFile.println(); //End of Row move to next row
    }
}

```

```

    dataFile.close(); //Close the file
}
else
    Serial.println("OOPS!! SD card writing failed");
}

```

As said earlier our intention is to **save the Date, Time, Temperature and Humidity into our SD card**. With the help of the DS3231 library and the DHT11 library our Arduino will be capable of reading all these four parameters and storing them into the following parameters as shown in table below

Date	rtc.getDateStr();
Time	rtc.getTimeStr();
Temperature	DHT.temperature
Humidity	DHT.humidity

Now we can directly use these parameters to store them on the SD card using the print line

```
dataFile.print(parameter);
```

You can notice that each parameter is separated by a comma to make it look legible and a *dataFile.println();* is used to indicate the end of the line.

5. Writing Data to PLX-DAQ

PLX-DAQ is Microsoft Excel Plug-in software that helps us to write values from Arduino to directly into an Excel file on our Laptop or PC. This is my personal favourite because of two reasons:

1. You can write and monitor the data at the same time and provides us way to plot them as graphs.
2. You do not need a RTC Module like DS3231 to keep track of date and time. You can simply use the date and time running on your Laptop/computer and save them directly on Excel.

To use this software with Arduino we have to send the data serially in a specific pattern just like displaying value on serial monitor. The key lines are explained below:

```

void Initialize_PlxDaq()
{

```



```

Serial.println("CLEARDATA"); //clears up any data left from previous projects
Serial.println("LABEL,Date,Time,Temperature,Humidity"); //always write LABEL, to indicate it as first line
}
void Write_PlxDaq()
{
    Serial.print("DATA"); //always write "DATA" to indicate the following as Data
    Serial.print(","); //Move to next column using a ","

    Serial.print("DATE"); //Store date on Excel
    Serial.print(","); //Move to next column using a ","

    Serial.print("TIME"); //Store date on Excel
    Serial.print(","); //Move to next column using a ","

    Serial.print(DHT.temperature); //Store date on Excel
    Serial.print(","); //Move to next column using a ","

    Serial.print(DHT.humidity); //Store date on Excel
    Serial.print(","); //Move to next column using a ","

    Serial.println(); //End of Row move to next row
}

```

The software can recognize keywords like LABEL, DATA, TIME, DATE etc. As shown in the Initialize function the keyword "LABEL" is used to write the first ROW of the Excel sheet. Later in the Write function we use the keyword "DATA" to indicate that the following information should be considered as DATA. To indicate that we have to move to next row we have to use comma (","). To indicate the end of row we have to send a *Serial.println()*.

As said earlier we can write the system date and time by sending the keywords "DATE" and "TIME" respectively as shown above.

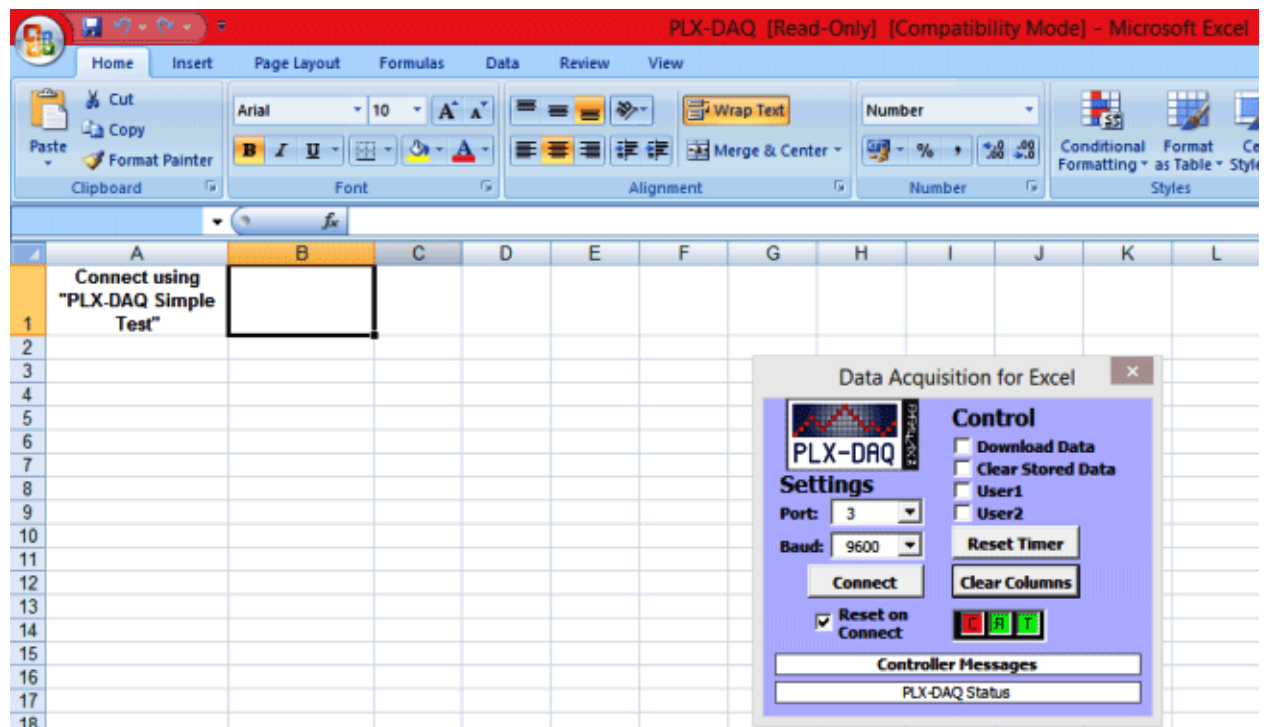
Note: Do not use serial monitor when using this PLX_DAQ software.

Working Explanation:

Working of the **Arduino Data Logger** is simple. Once the hardware and the software are ready it is time to burn the program into your Arduino Board. As soon your program gets uploaded, your temperature and humidity values will start to get stored in your SD card. You have to follow the steps below to enable PLX-DAQ to log the into Excel sheet in the computer.

Step 1: Open the “Plx-Daq Spreadsheet” file that was created on your desktop during installation.

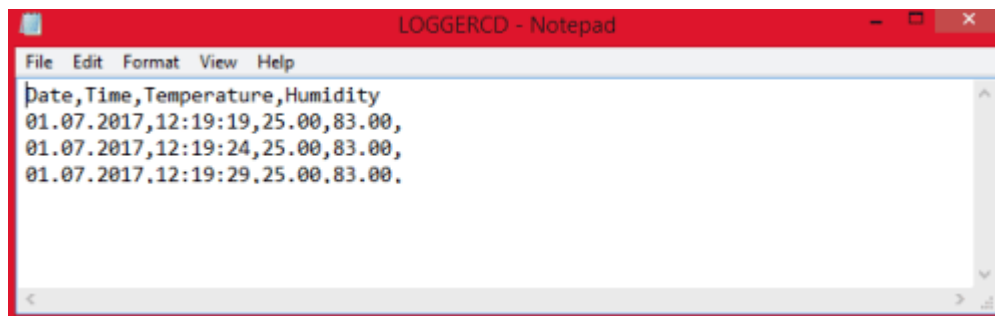
Step 2: If there is a Security block, click on *Options->Enable the content -> Finish -> OK* to get the following screen.



Step 3: Now select the baud rate as “9600” and the port to which your Arduino is connected and click on Connect. Your values should start to get logged like shown in the picture below.

	A	B	C	D	E	F	G	H	I
1	Date	Time	Temperature	Humidity					
2	07-01-2017	06:19:50 PM	24	91					
3	07-01-2017	06:19:55 PM	24	91					
4	07-01-2017	06:20:00 PM	24	91					
5	07-01-2017	06:20:05 PM	24	91					
6	07-01-2017	06:20:10 PM	24	91					
7	07-01-2017	06:20:15 PM	24	91					
8									
9									
10									
11									
12									
13									
14									
15									
16									

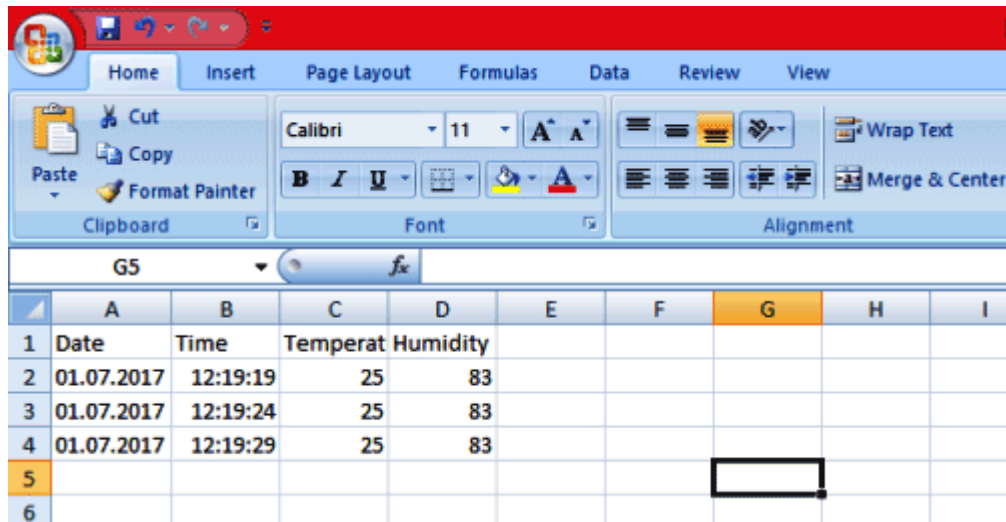
You can leave this excel sheet open and monitor the values as they get logged. As this is happening our SD card would also have saved the same values. To check is that is working simply remove the SD card and open it on your Computer. You should find a text file named “LoggerCD.txt” in it. When opened it would look something like this.



This file has data, but it would be hard to analyse them on a notepad. Hence we can open it on Excel as a CSV (Comma separated values) file, thus making it more effective. To open in excel

- 1.Open Excel. Click on File->Open and select “All file” at bottom right corner and select the “LoggerCD” file from the SD card. This will open a text import wizard.
- 2.Click on “Next” and select comma as a delimiter. Click on “Next” again. Then Finish.

3.Now your values will be opened in a Excel file as shown below

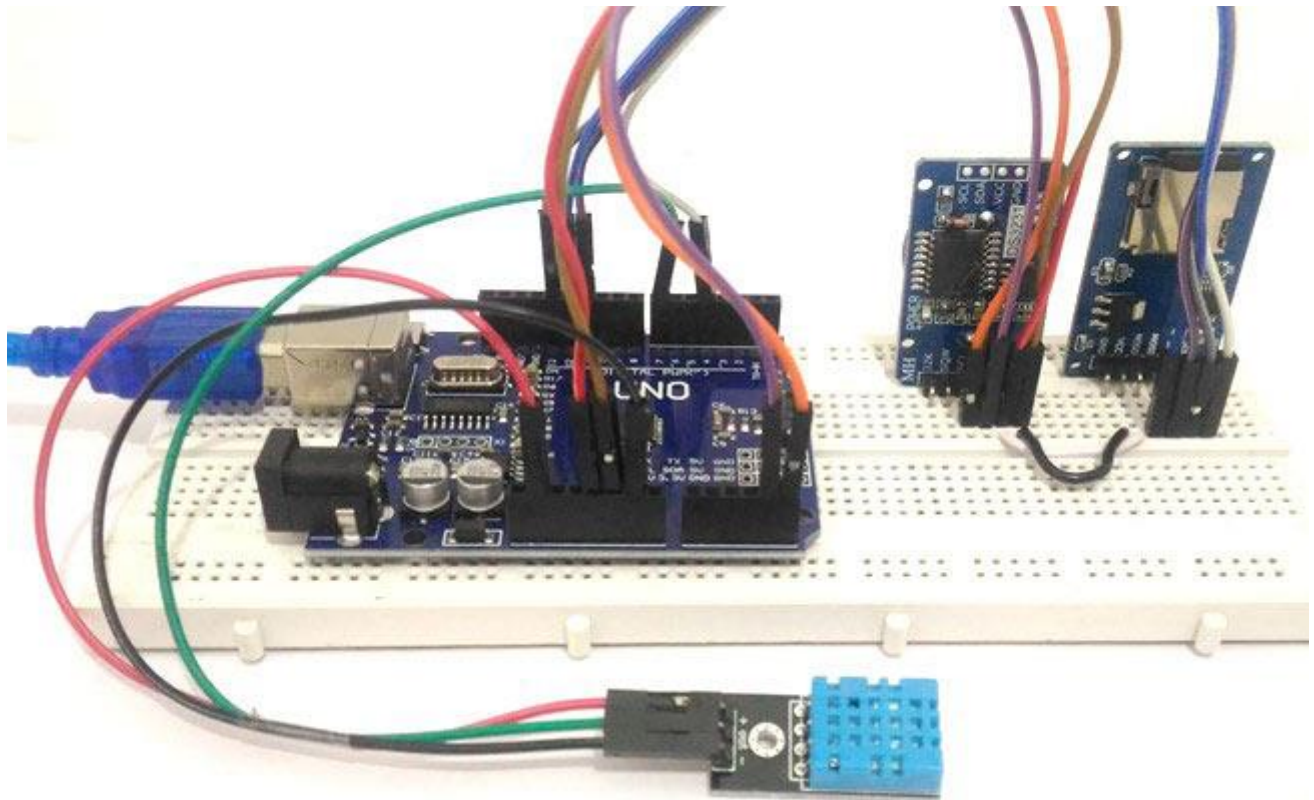


The screenshot shows the Microsoft Excel interface with the 'Home' tab selected. The ribbon includes options for Clipboard, Font, and Alignment. The active cell is G5. The data table is as follows:

	A	B	C	D	E	F	G	H	I
1	Date	Time	Temperat	Humidity					
2	01.07.2017	12:19:19	25	83					
3	01.07.2017	12:19:24	25	83					
4	01.07.2017	12:19:29	25	83					
5									
6									

I have logged the values for every 5 seconds; you can log them for any desired time by changing the delay function in the program. For detailed understanding of the working please **watch the video** below.

Hope you liked the project, if you have any doubt write them in the below comment section and I will help you out.



Bonus Improvement- Wireless Data Logging Using Arduino:

Once you have succeeded up to this point, then with few advancements and just adding a few lines of code you can log data wirelessly.

Simply connect a Bluetooth Device like HC-05 and write the data to PLX-DAQ via Bluetooth instead of Serial. That is replace *Serial.print(parameter);* with *BluetoothName.print(parameter);* and connect your Laptop to your Bluetooth Module and select the COM port to which your Laptops Bluetooth is connected and Taadaaa..... You have a working a Wireless Data Logging System in no time.

Code:

```
/*
 * Program to demonstrate Data Logging/Visualisation using Arduino
 *
 * ###Connection with SD card module###
 * Vcc->5V
 * Gnd->Gnd
 * MISO->pin 12
 * MOSI->pin 11
 * SCK-> pin 13
 * CS-> pin 4
 *
 * ###Connection with DS3231###
 * Vcc->5V
 * Gns->Gnd
 * SCL->pin A5
 * SDA-> pin A4
 *
 * ###Connection with DT11###
 * Vcc->5V
 * Gnd->Gnd
 * Out-> pin 7
 *
 */

#include <DS3231.h> //Library for RTC module (Download from Link in article)
#include <SPI.h> //Library for SPI communication (Pre-Loaded into Arduino)
#include <SD.h> //Library for SD card (Pre-Loaded into Arduino)
#include <dht.h> //Library for dht11 Temperature and Humidity sensor (Download from Link in article)

#define DHT11_PIN 7 //Sensor output pin is connected to pin 7
dht DHT; //Sensor object named as DHT

const int chipSelect = 4; //SD card CS pin connected to pin 4 of Arduino

// Init the DS3231 using the hardware interface
DS3231 rtc(SDA, SCL);

void setup()
{
    // Setup Serial connection
    Serial.begin(9600);
    Initialize_SDcard();
    Initialize_RTC();
    Initialize_PlxDaq();
}
```

```

void loop()
{
  Read_DHT11();
  Write_SDcard();
  Write_PlxDaq();
  delay(5000); //Wait for 5 seconds before writing the next data
}

void Write_PlxDaq()
{
  Serial.print("DATA"); //always write "DATA" to Indicate the following as Data
  Serial.print(","); //Move to next column using a ","
  Serial.print("DATE"); //Store date on Excel
  Serial.print(","); //Move to next column using a ","
  Serial.print("TIME"); //Store date on Excel
  Serial.print(","); //Move to next column using a ","
  Serial.print(DHT.temperature); //Store date on Excel
  Serial.print(","); //Move to next column using a ","
  Serial.print(DHT.humidity); //Store date on Excel
  Serial.print(","); //Move to next column using a ","
  Serial.println(); //End of Row move to next row
}

void Initialize_PlxDaq()
{
  Serial.println("CLEARDATA"); //clears up any data left from previous projects
  Serial.println("LABEL,Date,Time,Temperature,Humidity"); //always write LABEL, to indicate it as first line
}

void Write_SDcard()
{
  // open the file. note that only one file can be open at a time,
  // so you have to close this one before opening another.
  File dataFile = SD.open("LoggerCD.txt", FILE_WRITE);

  // if the file is available, write to it:
  if (dataFile) {
    dataFile.print(rtc.getDateStr()); //Store date on SD card
    dataFile.print(","); //Move to next column using a ","
    dataFile.print(rtc.getTimeStr()); //Store date on SD card
    dataFile.print(","); //Move to next column using a ","
    dataFile.print(DHT.temperature); //Store date on SD card
    dataFile.print(","); //Move to next column using a ","
    dataFile.print(DHT.humidity); //Store date on SD card
    dataFile.print(","); //Move to next column using a ","
    dataFile.println(); //End of Row move to next row
    dataFile.close(); //Close the file
  }
  else
  Serial.println("OOPS!! SD card writing failed");
}

void Initialize_SDcard()
{
  // see if the card is present and can be initialized:

```

```

if (!SD.begin(chipSelect)) {
    Serial.println("Card failed, or not present");
    // don't do anything more:
    return;
}
// open the file. note that only one file can be open at a time,
// so you have to close this one before opening another.
File dataFile = SD.open("LoggerCD.txt", FILE_WRITE);
// if the file is available, write to it:
if (dataFile) {
    dataFile.println("Date,Time,Temperature,Humidity"); //Write the first row of the excel file
    dataFile.close();
}
}

void Initialize_RTC()
{
    // Initialize the rtc object
    rtc.begin();

    ##### The following lines can be uncommented to set the date and time for the first time###
    /*
    rtc.setDOW(FRIDAY);    // Set Day-of-Week to SUNDAY
    rtc.setTime(18, 46, 45); // Set the time to 12:00:00 (24hr format)
    rtc.setDate(6, 30, 2017); // Set the date to January 1st, 2014
    */
}

void Read_DHT11()
{
    int chk = DHT.read11(DHT11_PIN);
}

/*void Read_DateTime()
{
    // Send date
    Serial.print(rtc.getDateStr());
    Serial.print(" -- ");

    // Send time
    Serial.println(rtc.getTimeStr());
}*/

/*void Read_TempHum()
{
    Serial.print("Temperature = ");
    Serial.println(DHT.temperature);
    Serial.print("Humidity = ");
    Serial.println(DHT.humidity);
    // delay(1000);
}*/

```