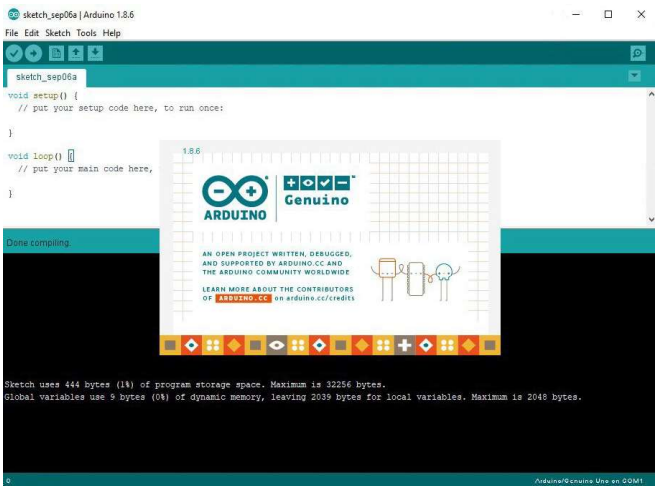
ARDUINO DATA LOGGING SYSTEM

Arduino is an open source electronic prototyping platform which is known for its flexibility, easy to use hardware and software. It is an open-source physical computing platform supported on a microcontroller board, and a development environment for writing software for the board. It simplifies the method of making an impact system by providing the standard board which will be programmed and connected to the system without PCB design and implementation.

It is a microcontroller board with an Universal Serial Bus (USB) plug which we can to the computer and has number of ports that can be wired to external electronic devices such as motors, sensors, relays, diodes, speakers, microphones and many more. The microcontroller board can be powered through the USB connection to the computer or use an external source of 9V power supply. Every component connected to the microcontroller board can be controlled from the computer or programmed by the computer and then disconnected and allowed to work independently. Arduino IDE can be downloaded from the Arduino webpage for free and all the necessary programming for the board can be done there.

In this experimental setup Arduino MEGA 2560 is used to measure the temperature values from the K-Type thermocouple and the temperature values are logged automatically into the computer for further analysis. This system helps us to save time and provides us highly accurate results. There is no need for manual recording of temperature using a temperature indicator.



Arduino IDE

The components used to build this system are:

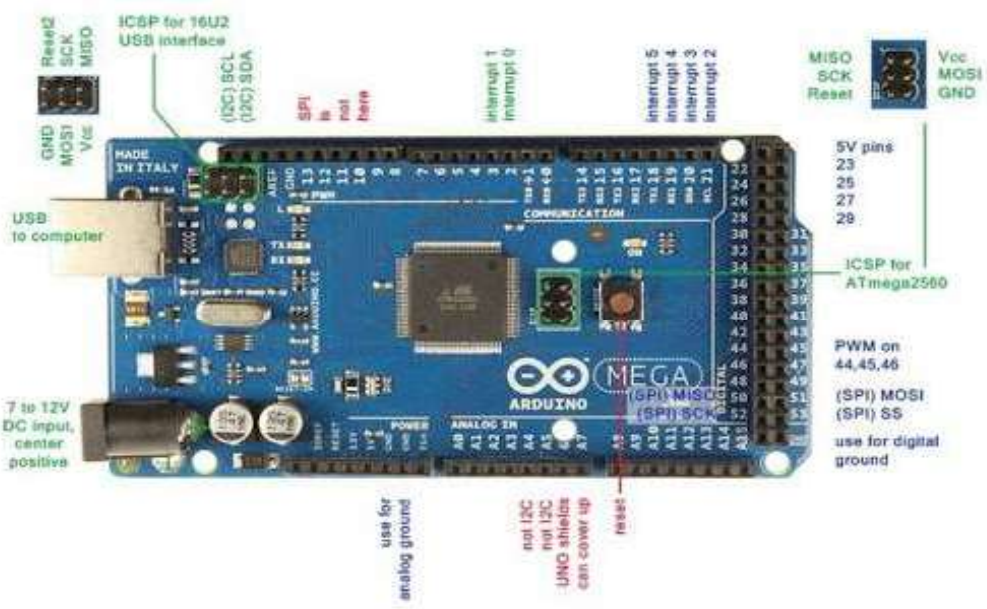
1. Arduino MEGA 2560
2. max6675 module
3. K-Type thermocouple sensor
4. 5V power supply
5. Bread board
6. PCB board
7. Connecting wires

1. Arduino MEGA 2560

Arduino MEGA is regarded as the muscle car of Arduino boards. The processor has an increased number of input and output pins and the ATmega1280 is fixed permanently to the board. If supplied with less than 7V, the board becomes unstable because the 5V pin supplies less than five volts. If supplied with more than 12V, the board get damaged due to the high heat developed in voltage regulator. The recommended range is 7 to 12 volts.

Features of Arduino Mega2560 board are given below:

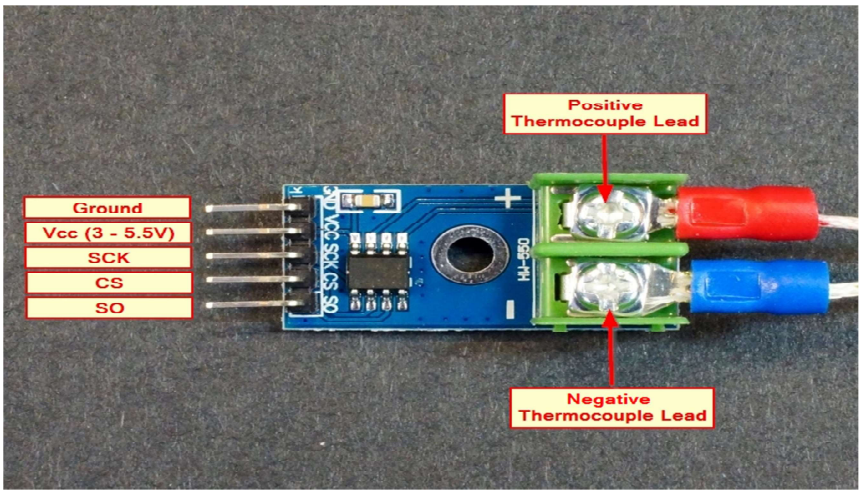
- 54 input/output pins
- 128KB of flash memory to store fixed data and sketches
- 8KB of RAM and 4KB of EEPROM
- 16 analog input,
- 4 UARTs (hardware serial ports)
- a 16 MHz crystal oscillator
- a USB connection
- a power jack
- a reset button.



Arduino MEGA 2560

2. Max6675 module

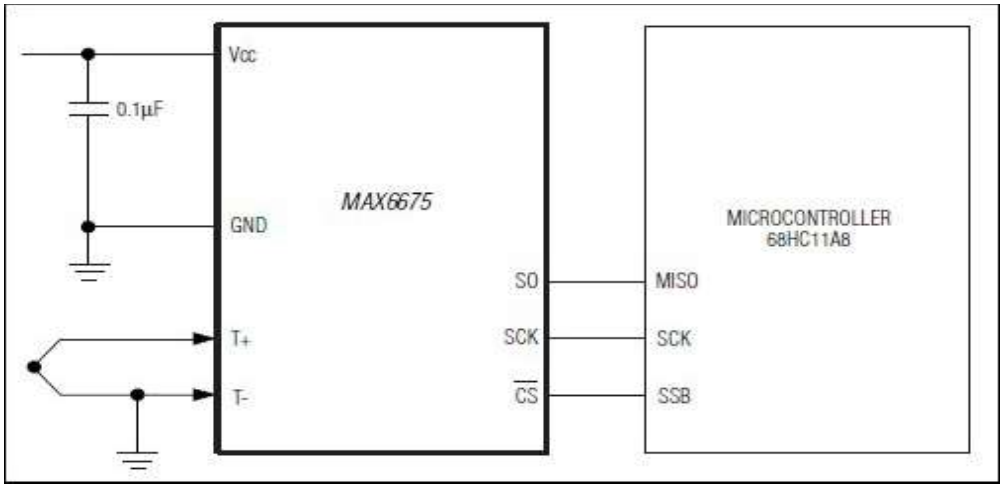
The K-Type thermocouple sensors cannot be connected directly to the Arduino MEGA board because the sensors do not give a digital output; instead it only gives a EMF value by the cold junction. Therefore, we need a digital convertor between the thermocouple and the Arduino input. This can be done with the help of max6675 module.



Max6675 module

The cold-junction compensation is done in max6675 module and digitizes the signal from a thermocouple sensor. This converter allows readings as high as +1024°C and temperature is resolved to 0.25°C, The max6675 is a thermocouple-to-digital converter which has a inbuilt 12-bit Analog-to-Digital Converter (ADC). The max6675 has a digital controller, cold-junction compensation sensing and correction, and SPI-compatible interface, and control logic.

Each of the ten thermocouple has its own max6675 module and connecting wires or jumper cables are used for the connection. A common power supply of 5V is given by the adapter to every module using a PCB board by a series connection.



Application circuit of max6675

PIN NAME	DESCRIPTION
GND	Ground
T-	Alumel Lead of K- Type Thermocouple. Should be connected to ground externally.
T+	Chromel lead of the K-Type thermocouple
Vcc	Power supply (5V)
SCK	Serial clock input
CS	Chip select
SO	Serial output

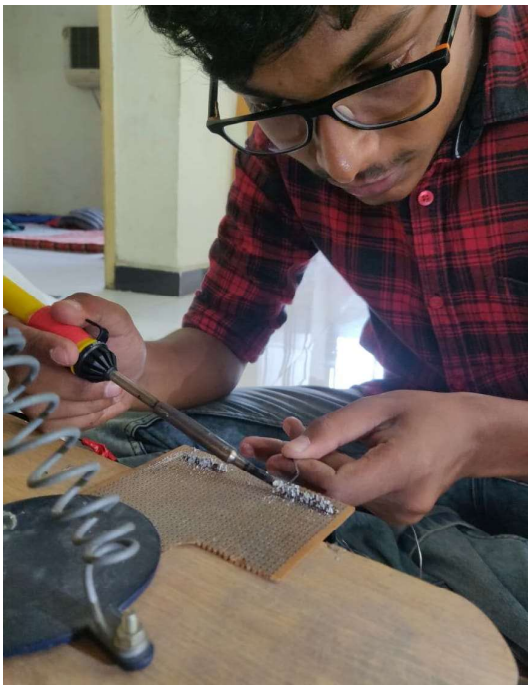
Max6675 pin description

HARDWARE IMPLEMENTATION

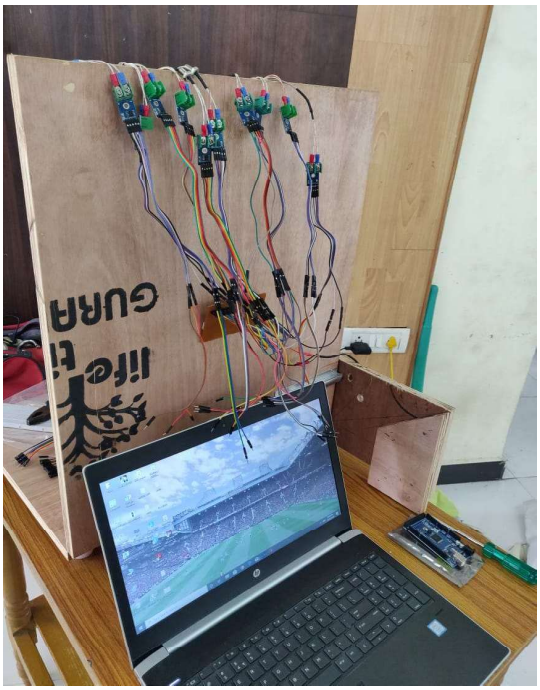
This microcontroller is easy to wire and program. It has sufficient digital input pins to connect all thermocouple sensors and the max6675 modules. The Arduino is used for sensor readings collection, changing sensor parameters, and transferring sensor output the other hardware components whenever needed.

Procedures involved in the hardware implementation are as follows,

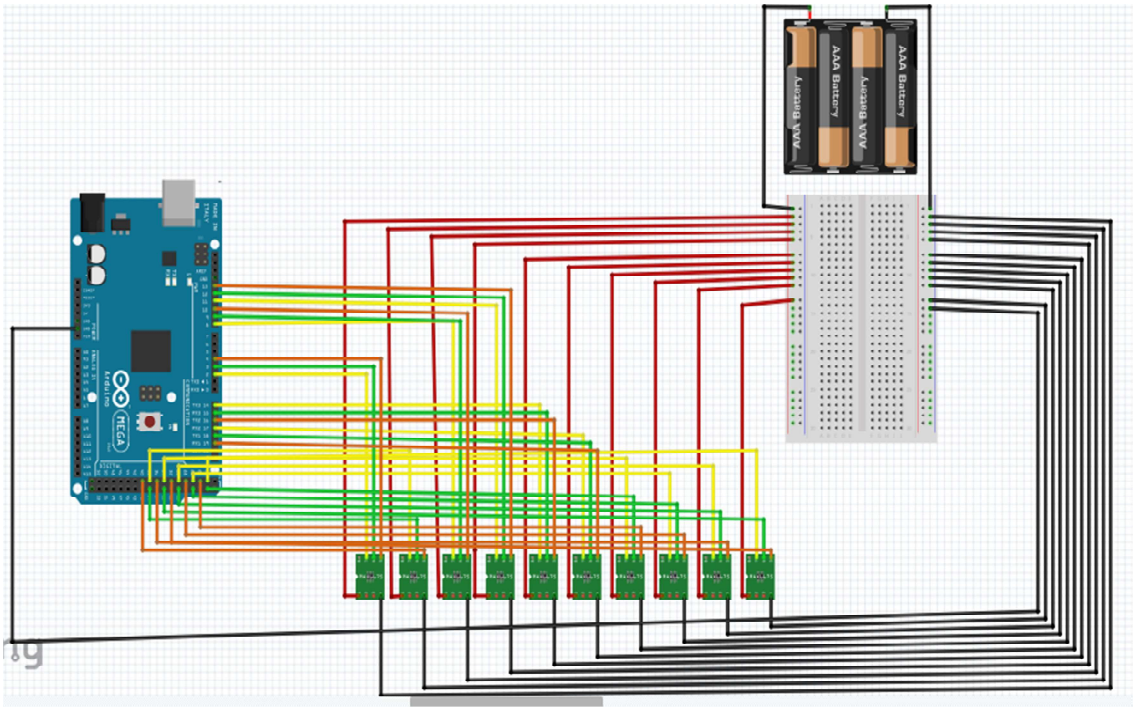
- Each K-Type thermocouple is connected to individual max6675 module. The fork terminals of the sensors are screwed into the module depending on the positive and negative terminals.
- A PCB board is taken; the positive wire of the 5V power supply adapter is soldered in series with the entire Vcc terminal of the max6675.
- Similarly, the negative wire of the 5V power supply adapter is soldered in series with the entire ground terminal (GND) of the max6675 and another wire is connected to the ground of the Arduino Mega2560 microcontroller board.
- Then we take max6675 module individually and connect the three pins (SCK, CS, SO) to the Arduino board's digital input pins. We must be well aware of the pin configuration number, which makes the programming task simple.
- The connecting wires of SCK, CS, SO of each module are taped for easy process.
- All the thermocouple sensors and module are numbered.
- The entire connections are checked for proper connections and insulated tightly for proper working of the system.
- The USB from pc is connected to the microcontroller for further programming and working.
- The serial monitor is ran to visualize the output of the program and the sensors.



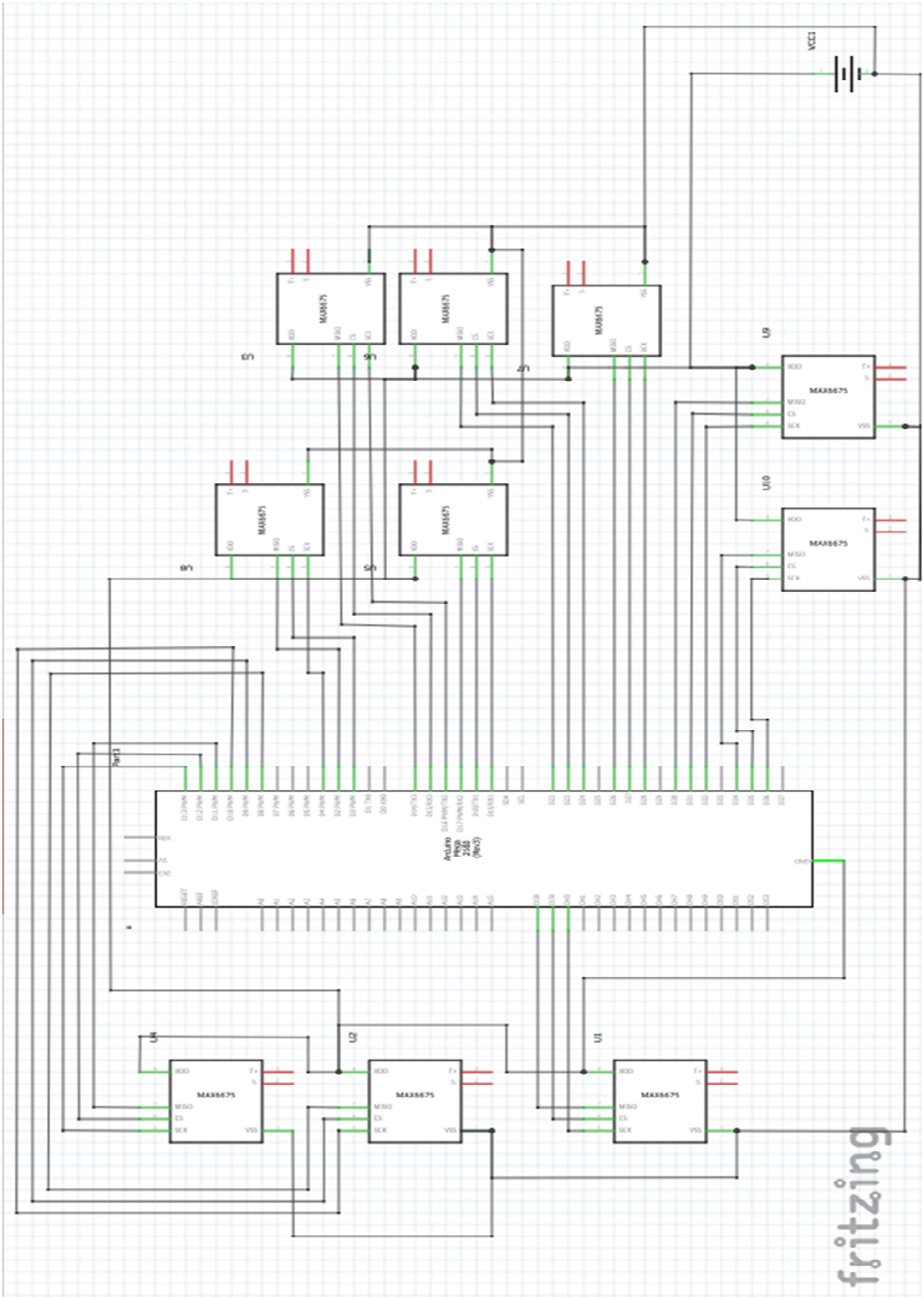
Hardware implementation - 1



Hardware implementation - 2



Breadboard diagram



schematic diagram

SOFTWARE IMPLEMENTATION

The necessary programming for the above hardware connections are carried out in Arduino IDE (Integrated Development Environment). The software implementation of this system is carried out in two components: the code describing for microcontroller program and the data logging software.

An Arduino code generally consists of a setup block and a loop block. In the setup block, pin modes are initialized and start serial communication. The program execute first here, therefore called as preparation block. This function is executed only once and has to be included even if there are no statements to execute. The loop block is known as the execution block where reading inputs, returning outputs and condition checking takes place. Loop function executes the set of statements repeatedly.

Procedure involved in coding the microcontroller:

- The USB from pc is connected to the microcontroller and the Arduino IDE is installed and opened.
- In order to connect the module, **max6675.h** library file is downloaded and installed which is provided by the supplier. Then we have to include that in the program.
- Then each of the module wire and the respective Arduino input pins are initialized using int. **eg.** int ktcS01 = 2; Which implies that the SO pin of first sensor is connected to the pin 2 of Arduino.
- Then the code is started in setup function and delay is provided in milliseconds.
- Serial.print function is used to display the output in serial monitor and ktc1.readCelsius() gives us the temperature of sensor one in Celsius.
- Finally the delay is provided in the loop for the time period.
- Then the code verified and complied in the IDE.
- After that the code is uploaded into the Arduino Mega microcontroller.
- Then the output of the program is viewed in the serial monitor of the Arduino IDE and further changes can be made and uploaded to the microcontroller.

PROGRAM CODE

NOTE:

The lines starting with // is a comment and this doesn't have any effect in the program. These are used to inform the user on how the program works

```
#include "max6675.h"
    // max6675 library is included in the program
int ktcSO1 = 2;
int ktcCS1= 3;
int ktcCLK1 = 4;
    // pin numbers of SO, CS AND CLK are initialized with the integer data type
    // similarly, it is done for remaining modules
int ktcSO2 = 38;
int ktcCS2= 39;
int ktcCLK2 = 40;
int ktcSO3 = 8;
int ktcCS3= 9;
int ktcCLK3 = 10;
int ktcSO4 = 11;
int ktcCS4= 12;
int ktcCLK4 = 13;
int ktcSO5 = 14;
```

```
int ktcCS5= 15;

int ktcCLK5 = 16;

int ktcSO6 = 17;

int ktcCS6= 18;

int ktcCLK6 = 19;

int ktcSO7 = 22;

int ktcCS7= 23;

int ktcCLK7 = 24;

int ktcSO8 = 26;

int ktcCS8= 27;

int ktcCLK8 = 28;

int ktcSO9 = 30;

int ktcCS9= 31;

int ktcCLK9 = 32;

int ktcSO10 = 34;

int ktcCS10= 35;

int ktcCLK10 = 36;


MAX6675 ktc1(ktcCLK1, ktcCS1, ktcSO1);

    // initialized pin numbers are grouped with respect to the module numbers (ie: SO1,
    CS1 and CLK1 are grouped with ktc1 to get the output in serial monitor)

MAX6675 ktc2(ktcCLK2, ktcCS2, ktcSO2);

MAX6675 ktc3(ktcCLK3, ktcCS3, ktcSO3);
```

```
MAX6675 ktc4(ktcCLK4, ktcCS4, ktcSO4);

MAX6675 ktc5(ktcCLK5, ktcCS5, ktcSO5);

MAX6675 ktc6(ktcCLK6, ktcCS6, ktcSO6);

MAX6675 ktc7(ktcCLK7, ktcCS7, ktcSO7);

MAX6675 ktc8(ktcCLK8, ktcCS8, ktcSO8);

MAX6675 ktc9(ktcCLK9, ktcCS9, ktcSO9);

MAX6675 ktc10(ktcCLK10, ktcCS10, ktcSO10);


void setup()

        // program starts from the setup function

{

 Serial.begin(9600);

        // tells the microcontroller to exchange information with the serial monitor at a rate of
        9600 bits per second

  delay(500);

        // delay is used to pause the program for specified amount of time, which is given in
        milliseconds

}


void loop()

{

   Serial.print("Deg C1 = ");

  Serial.print(ktc1.readCelsius());
```

// **serial.print()** is used as the output statement for serial monitor. The information in double quotes are printed directly. While **readCelsius()** is used to read or get the temperature values from each max6675 module

```
Serial.print("Deg C2 = ");

Serial.print(ktc2.readCelsius());

Serial.print("Deg C3 = ");

Serial.print(ktc3.readCelsius());

Serial.print("Deg C4 = ");

Serial.print(ktc4.readCelsius());

Serial.print("Deg C5 = ");

Serial.print(ktc5.readCelsius());

Serial.print("Deg C6 = ");

Serial.print(ktc6.readCelsius());

Serial.print("Deg C7 = ");

Serial.print(ktc7.readCelsius());

Serial.print("Deg C8 = ");

Serial.print(ktc8.readCelsius());

Serial.print("Deg C9 = ");

Serial.print(ktc9.readCelsius());

Serial.print("Deg C10 = ");

Serial.print(ktc10.readCelsius());

Serial.println("\n");
```

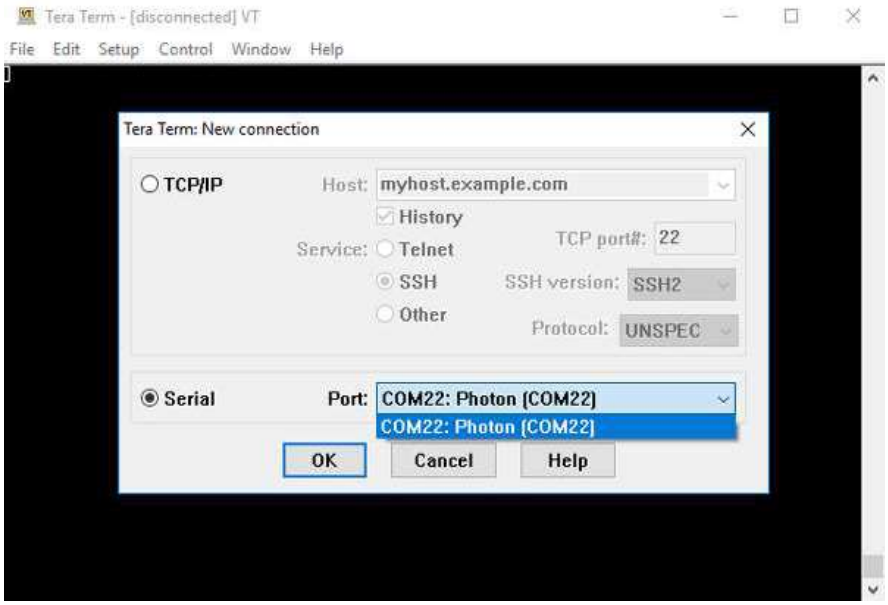
```
  delay(5000);
      // pauses the program for 5 seconds and start the next set of reading. This specifies the
      time period
}
```

DATA LOGGING

The result which we obtain in the serial monitor of the Arduino IDE is logged with the help of an open source software known as **Tera Term.** This Software allows us to log the data directly from the IDE platform into a excel sheet. The serial monitor should be operating while this operation. It emulates different types of computer terminals. It supports serial port connections , telnet and Secure Shell (1 and 2).

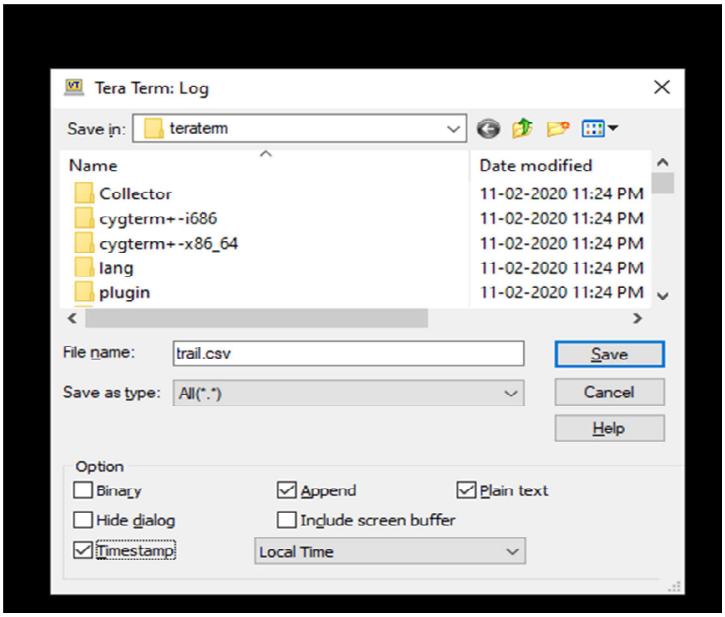
Procedure involved in data logging is as follows:

- Tera Term is downloaded and installed in the computer.
- The serial port in which the Arduino is connected, is selected.
- Then file -> log is selected.
- The file name is typed with an extension of .csv ,in order to save in excel sheet.
- The time stamp check box is click to record the time for each set of readings



Data logging step-1

.



Data logging 2

- After the experiment, file -> stop logging is selected and the excel sheet is opened for alteration in orientation.