3 (a) .Create a program that blinks the LED on the development board using MBED software

Aim:

To write a program for blink led using tinker Cad.

Components Used:

1.Arduino Uno

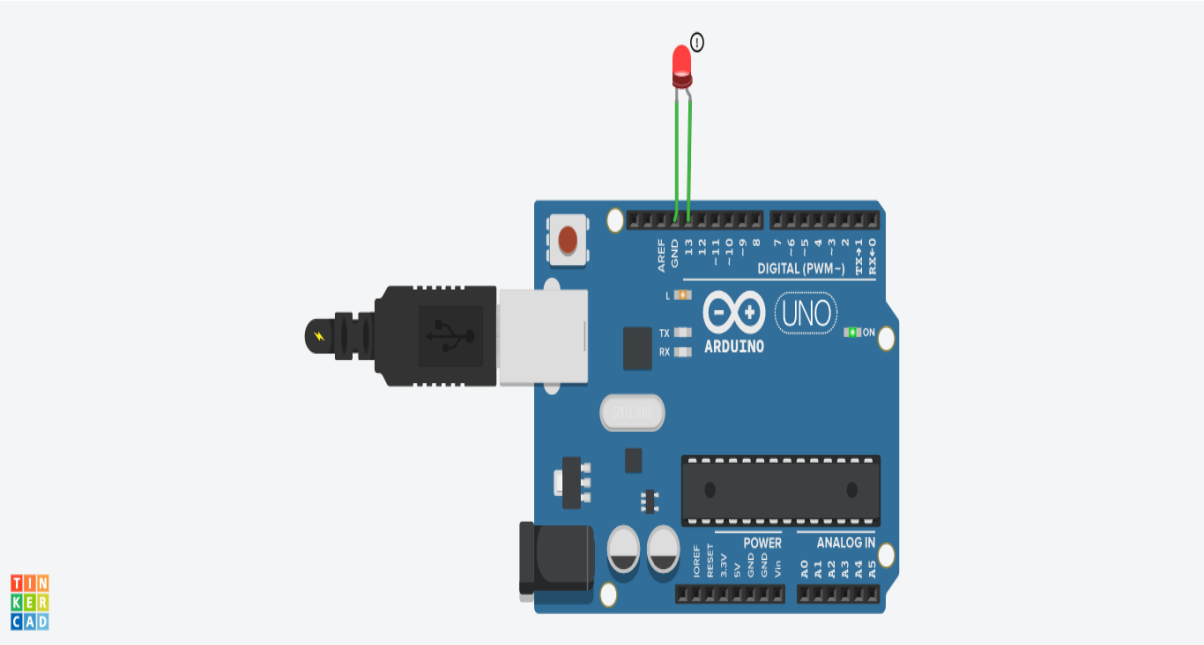
2. LED(Light Emitting Diode)

Procedure:

1.Get the Arduino uno board from the components

2.Get the LED from the components

3.LED has two side which is positive (anode) and Negative(Cathode).Negative side is connected to the Ground(GND).Positive side is connected to Digital pin 13 of Arduino .

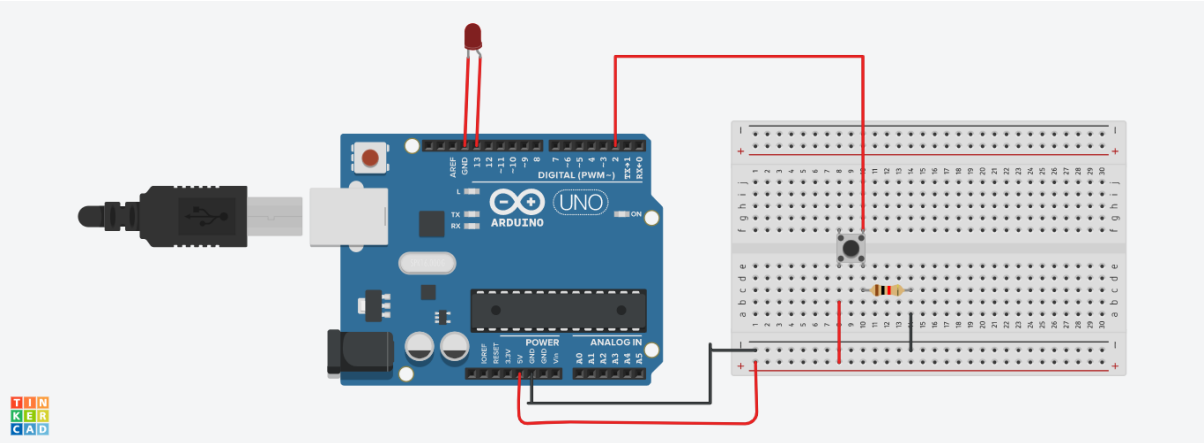


Code:

```
void setup()
{
  pinMode(13, OUTPUT);
}
```

```
void loop()
{
  digitalWrite(13, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(13, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

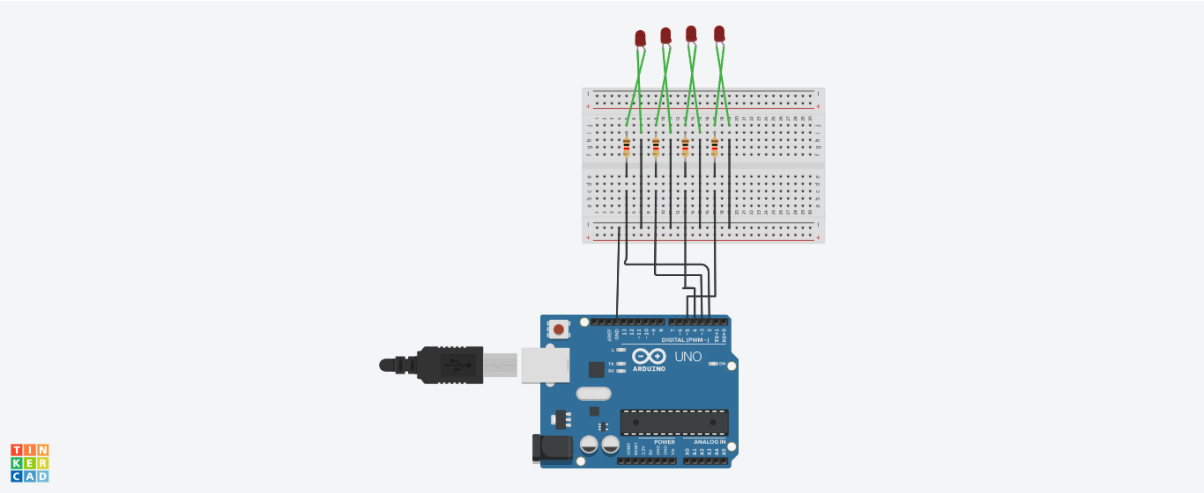
3 (b) Through Button blink LED




```
void setup()
{
  pinMode(2, INPUT);
  pinMode(13,OUTPUT);
}

void loop()
{
  if(digitalRead(2)==1)
  {
    digitalWrite(13,HIGH);
  }
  else
  {
```

```
    digitalWrite(13,LOW);
  }
}
```

3 (c). Led with digital counter



CODE:

```
int pin2=2;
int pin3=3;
int pin4=4;
int pin5=5;
int stime=500;

void setup()
{
  pinMode(pin2,OUTPUT);
  pinMode(pin3,OUTPUT);
  pinMode(pin4,OUTPUT);
  pinMode(pin5,OUTPUT);
```

```
}

void loop()
{


 digitalWrite(pin2,LOW);
 digitalWrite(pin3,LOW);
 digitalWrite(pin4,LOW);
 digitalWrite(pin5,LOW);
 delay(stime);

 digitalWrite(pin2,LOW);
 digitalWrite(pin3,LOW);
 digitalWrite(pin4,LOW);
 digitalWrite(pin5,HIGH);
 delay(stime);

 digitalWrite(pin2,LOW);
 digitalWrite(pin3,LOW);
 digitalWrite(pin4,HIGH);
 digitalWrite(pin5,LOW);
 delay(stime);

 digitalWrite(pin2,LOW);
 digitalWrite(pin3,LOW);
 digitalWrite(pin4,HIGH);
 digitalWrite(pin5,HIGH);
 delay(stime);

 digitalWrite(pin2,LOW);
```

```
digitalWrite(pin3,HIGH);
digitalWrite(pin4,LOW);
digitalWrite(pin5,LOW);
delay(stime);

digitalWrite(pin2,LOW);
digitalWrite(pin3,HIGH);
digitalWrite(pin4,LOW);
digitalWrite(pin5,HIGH);
delay(stime);


digitalWrite(pin2,LOW);
digitalWrite(pin3,HIGH);
digitalWrite(pin4,HIGH);
digitalWrite(pin5,LOW);
delay(stime);

digitalWrite(pin2,LOW);
digitalWrite(pin3,HIGH);
digitalWrite(pin4,HIGH);
digitalWrite(pin5,HIGH);
delay(stime);

digitalWrite(pin2,HIGH);
digitalWrite(pin3,LOW);
digitalWrite(pin4,LOW);
digitalWrite(pin5,LOW);
delay(stime);
digitalWrite(pin2,HIGH);
digitalWrite(pin3,LOW);
```

```
digitalWrite(pin4,LOW);
digitalWrite(pin5,HIGH);
delay(stime);

digitalWrite(pin2,HIGH);
digitalWrite(pin3,LOW);
digitalWrite(pin4,HIGH);
digitalWrite(pin5,LOW);
delay(stime);

digitalWrite(pin2,HIGH);
digitalWrite(pin3,LOW);
digitalWrite(pin4,HIGH);
digitalWrite(pin5,HIGH);
delay(stime);

digitalWrite(pin2,HIGH);
digitalWrite(pin3,HIGH);
digitalWrite(pin4,LOW);
digitalWrite(pin5,LOW);
delay(stime);

digitalWrite(pin2,HIGH);
digitalWrite(pin3,HIGH);
digitalWrite(pin4,LOW);
digitalWrite(pin5,HIGH);
delay(stime);

digitalWrite(pin2,HIGH);
digitalWrite(pin3,HIGH);
digitalWrite(pin4,HIGH);
```

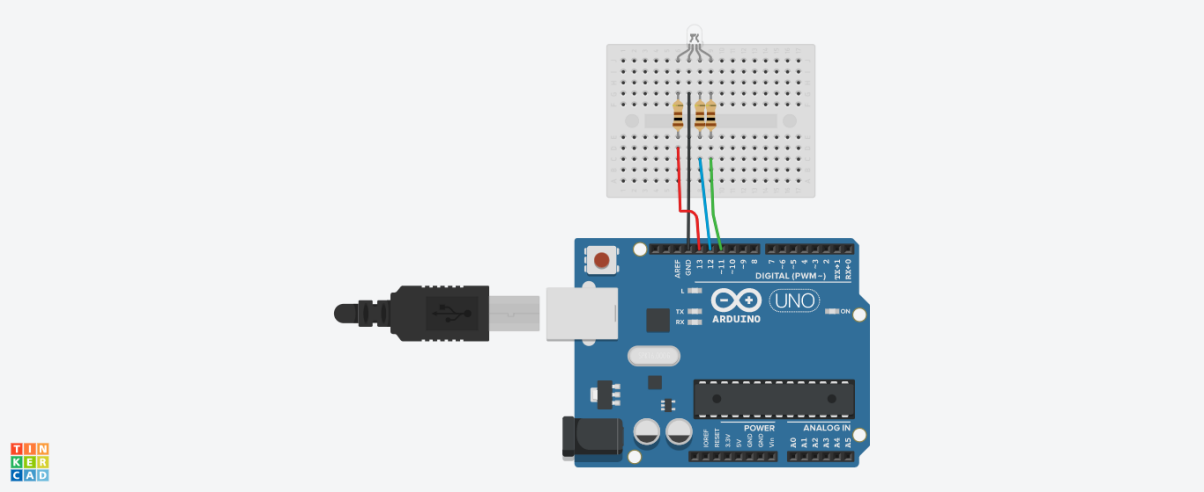
```
 digitalWrite(pin5,LOW);
 delay(stime);

 digitalWrite(pin2,HIGH);
 digitalWrite(pin3,HIGH);
 digitalWrite(pin4,HIGH);
 digitalWrite(pin5,HIGH);
 delay(stime);

}
```

3 (d) Pick one-one from the available sensors and actuators and find or create code that will display the sensed data on the PC.

Actutors:

Convert the electrical signal into light energy.

Capacitors-100 ohm



Code:

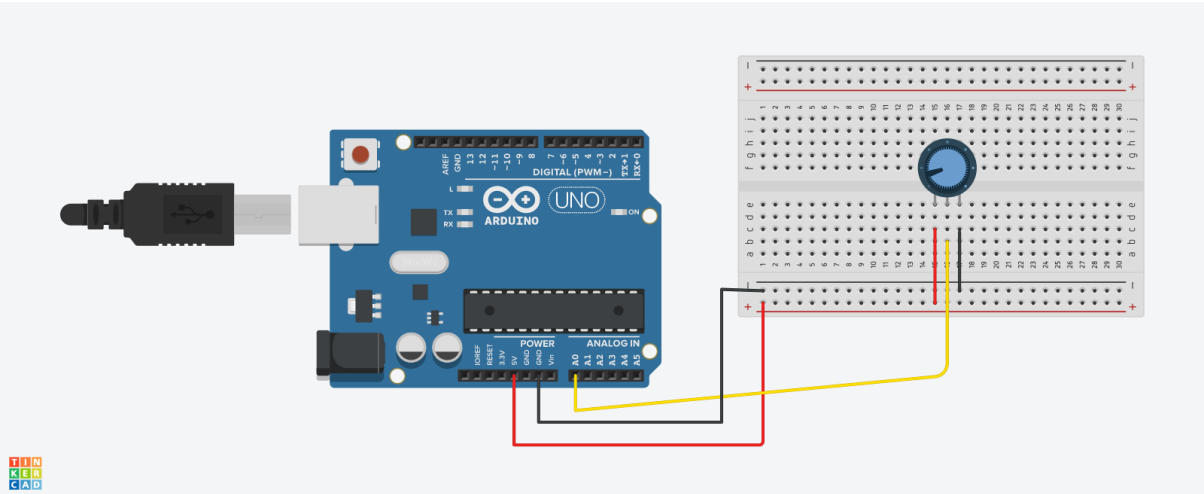
```
int redled=13;
int blueled=12;
int greenled=11;

void setup()
{
```

```
  pinMode(redled, OUTPUT);
  pinMode(blueled,OUTPUT);
  pinMode(greenled,OUTPUT);
  Serial.begin(9600);
  Serial.println("rgb");
}

void loop()
{
  digitalWrite(redled, HIGH);
  digitalWrite(blueled,HIGH);
  digitalWrite(greenled,HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
   // Wait for 1000 millisecond(s)
}
```

Reading numbers from te serial monitor-toto18

EX 4 (a).

Pick one-one from the available sensors and actuators and find or create code that will display the sensed data on the PC.



Code:

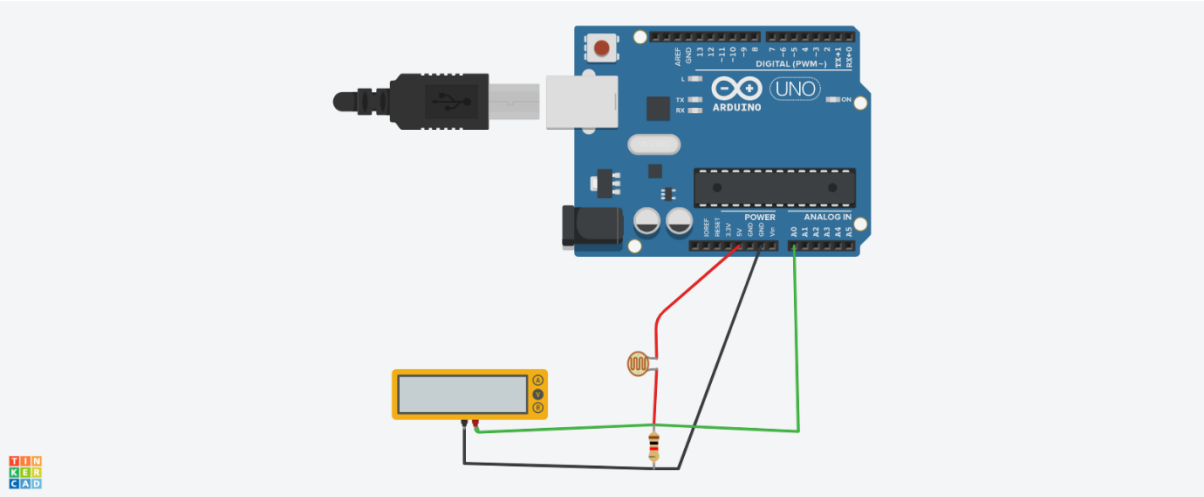
```
int pot=A0;
void setup()
{
```

```
 Serial.begin(9600);
}

void loop()
{
  int potvalue=analogRead(pot);
  Serial.print("potvalue");
  Serial.println(potvalue);
  delay(10);
}
```

Output

```
Serial Monitor
6
6
6
439
452
464
559
```

4 (b).



Code:

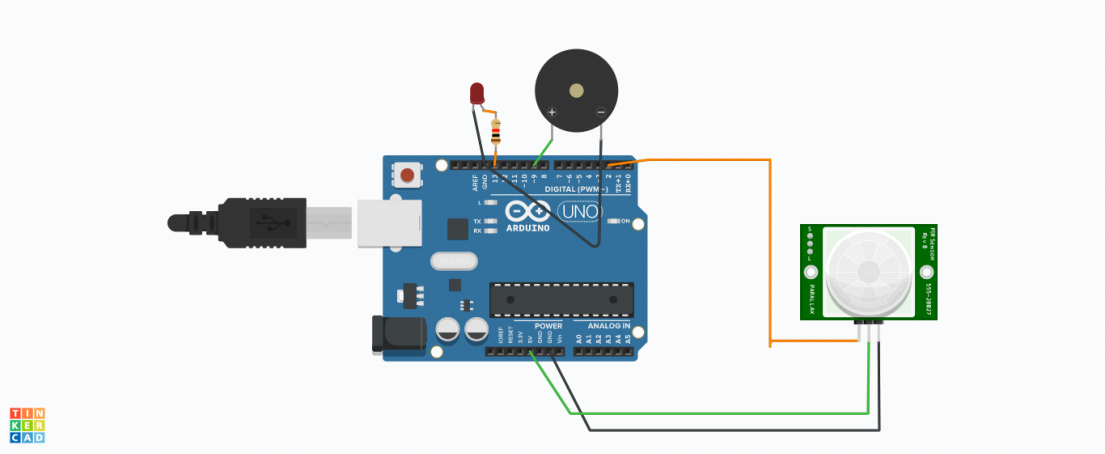
```
void setup()
{
```

```
  pinMode(A0, INPUT);
  Serial.begin(9600);
}

void loop()
{
  int lightvalue=analogRead(A0);
  Serial.println(lightvalue);
  delay(1000);
}
```

Output:

```
6
6
6
6
6
6
6
6
118
517
517
607
607
607
607
```

5.Create a program that displays data from the sensor in regular intervals in a compact format.

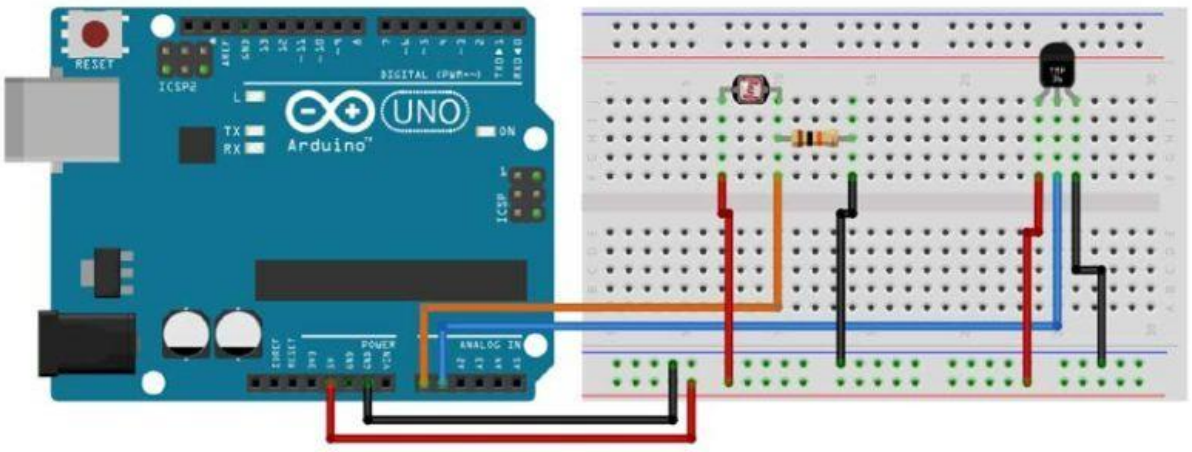


```
void setup()
{
  pinMode(2, INPUT);
  pinMode(13, OUTPUT);
  pinMode(9, OUTPUT);
}

void loop()
{
  if (digitalRead(2) >= HIGH) {
    digitalWrite(13, HIGH);
    tone(9, 523, 1000); // play tone 60 (C5 = 523 Hz)
  } else {
    digitalWrite(13, LOW);
    noTone(9);
  }
  delay(1); // Wait for 1 millisecond(s)
}
```

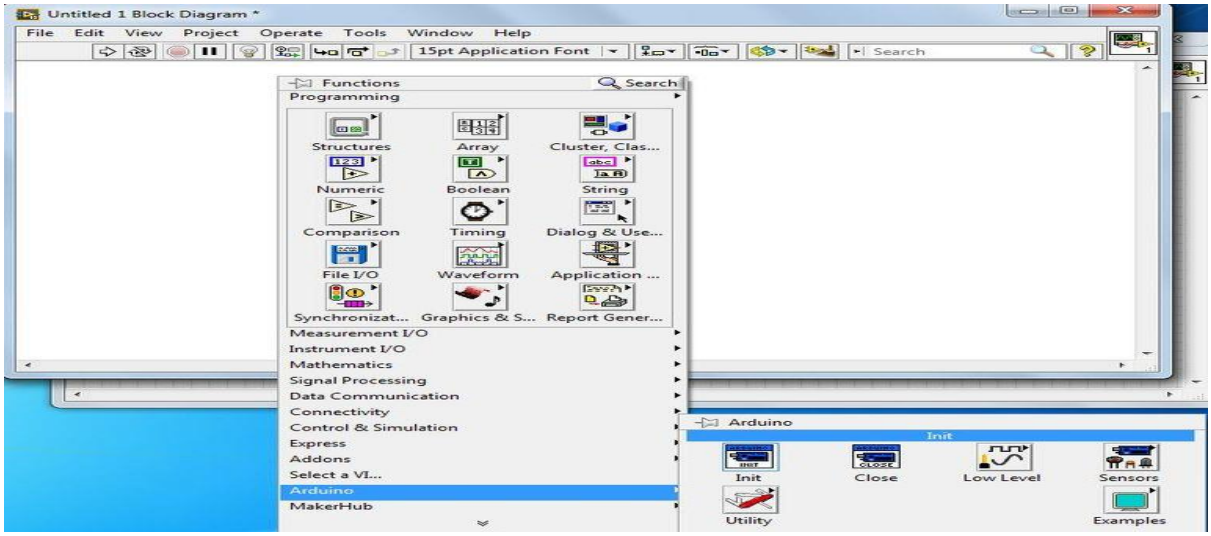
6.

7. WEATHER STATION USING LAB VIEW

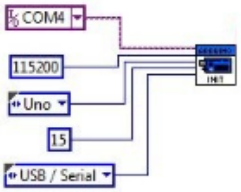
Circuit Diagram



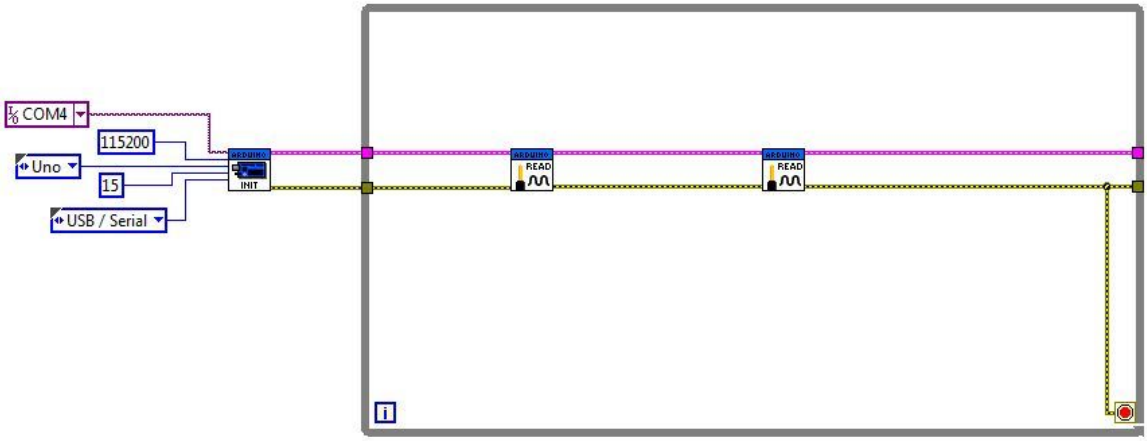
1. Start the LabVIEW.
2. Creat Blank VI for weather station as in Tutorial 1.
3. Go to LabVIEW “Block Diagram” Panel
4. Right Click on white space. Go to “Arduino” and select “init”. This will add Initializations of Arduino board.



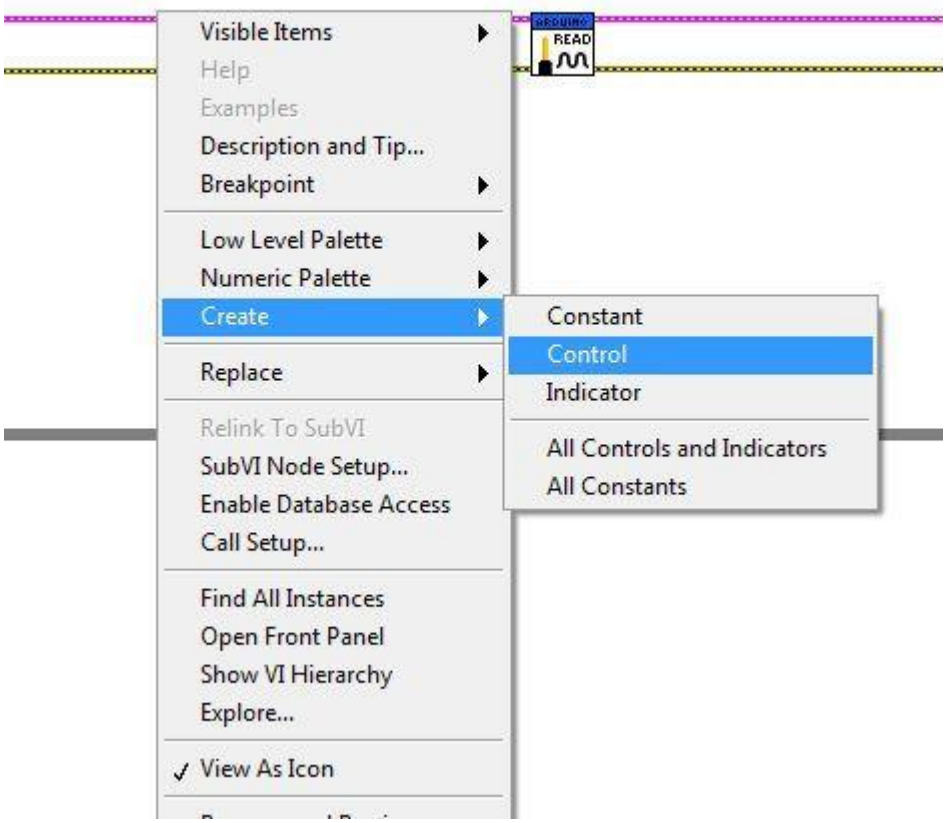
5. Bring Cursor to anywhere in LabVIEW “Block Diagram” panel and place the “Init”.
6. First input is “VISA resource”. It is the serial port you are using for interfacing of Adruino. You can find it in “device manager” of your computer under “ports (COM & LPT)….” Make sure Arduino board is connected with computer otherwise it won’t be shown. In my case it is COM4.
7. Bring cursor on first input of “Init” until it shows “VISA resource”. Right click on it. Go to “create” and select “constant”. As it will be a constant value of Port which will be always used for serial communication.
8. Click on arrow it will show available option. In my case its “COM4”. Select appropriate one after checking from device manger as mentioned above otherwise it won’t work.
9. Second input is “Baud Rate”. Create it as constant as done for “VISA resource”. Right click on “Baud Rate” then “create” and then “constant”.
10. Third input is “Board Type”, fourth is “Bytes per packet” and fifth is “Connection type” make them also constant.
11. Click on white space on LabVIEW “Block Diagram” and follow “Structure → select While loop”.
12. Place two “analogue read pin” as follows one by one.
13. Result is as follows.



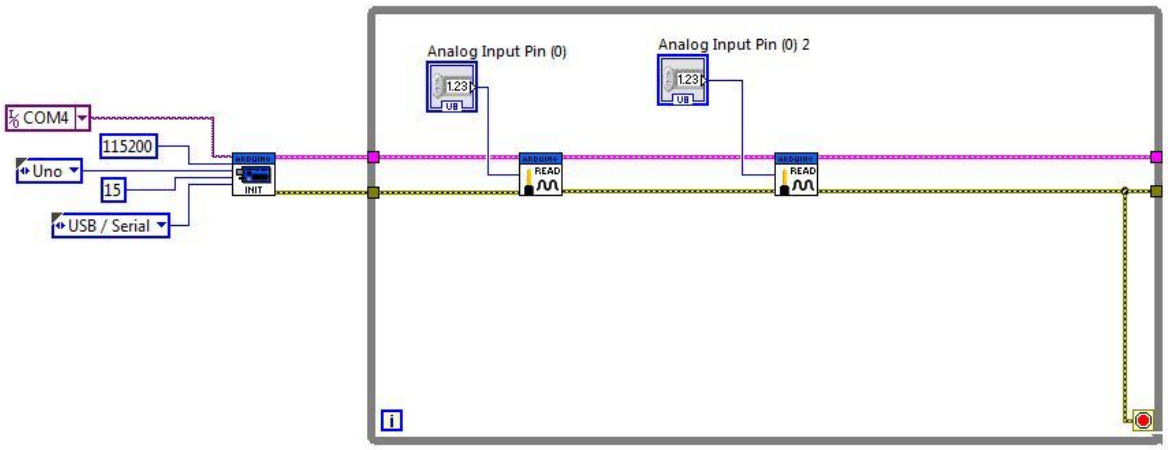
14. Join diagram as follows



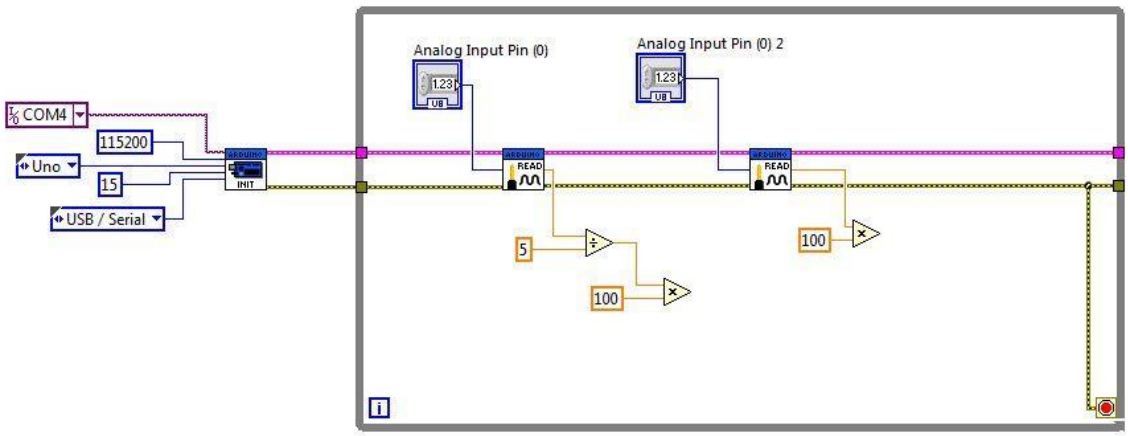
15. Create “Control” for input parameter “Analogue input pin” of both “Analogue Read Pin” blocks as follows.



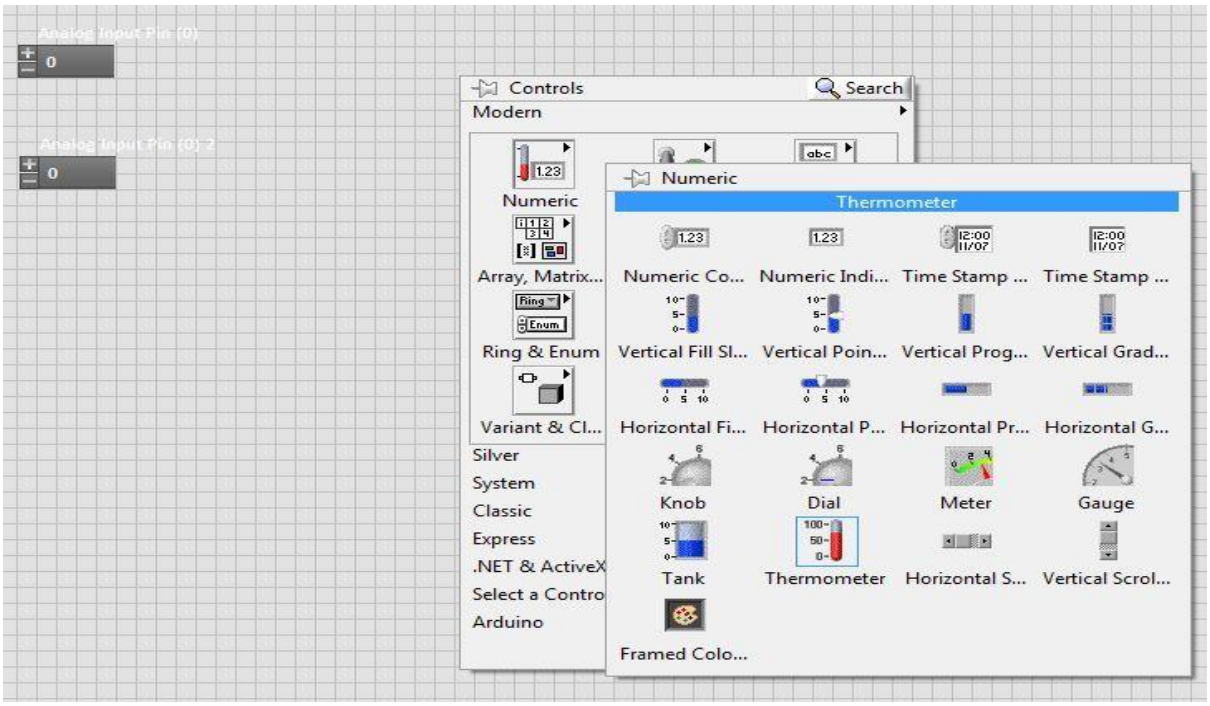
16. This step results as



17. Right click on white area. Go to “numeric” and find “divide”.
18. Similarly find “multiply” and place it and join them as follows. Create indicated “constants”.
19. Similarly create “multiply” and “constant” as indicated.

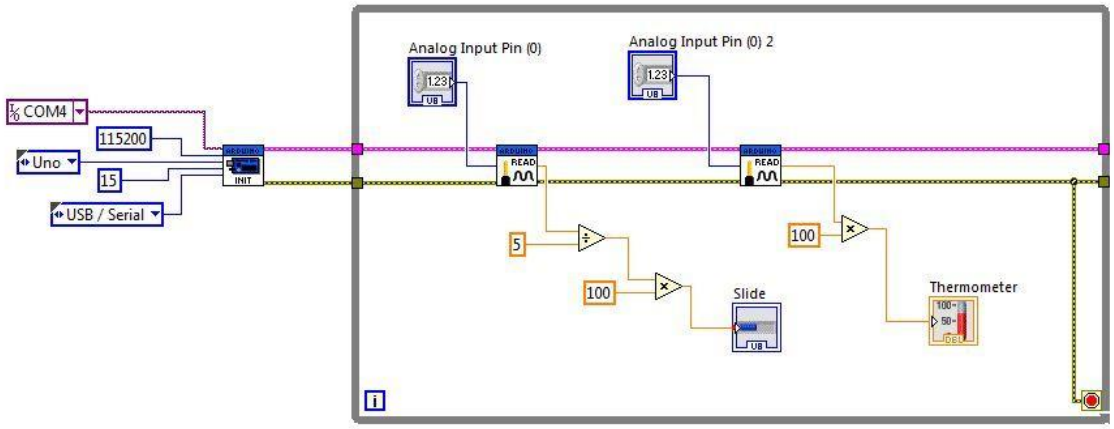


20. Go to Front panel and find “thermometer” as shown on block diagram of LabVIEW.

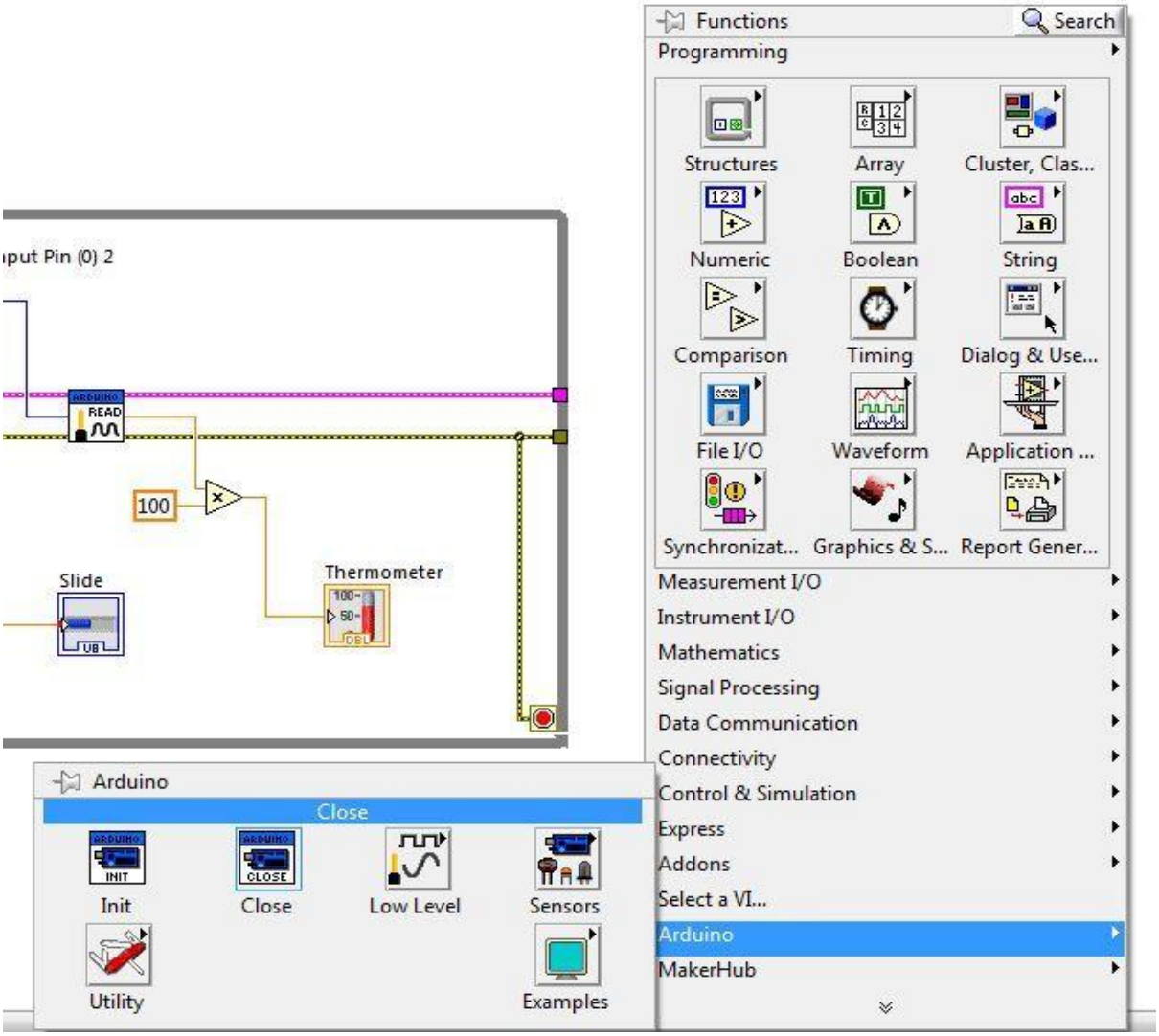


21. Also find “slide” as we found thermometer. Place both of them on front panel one by one.

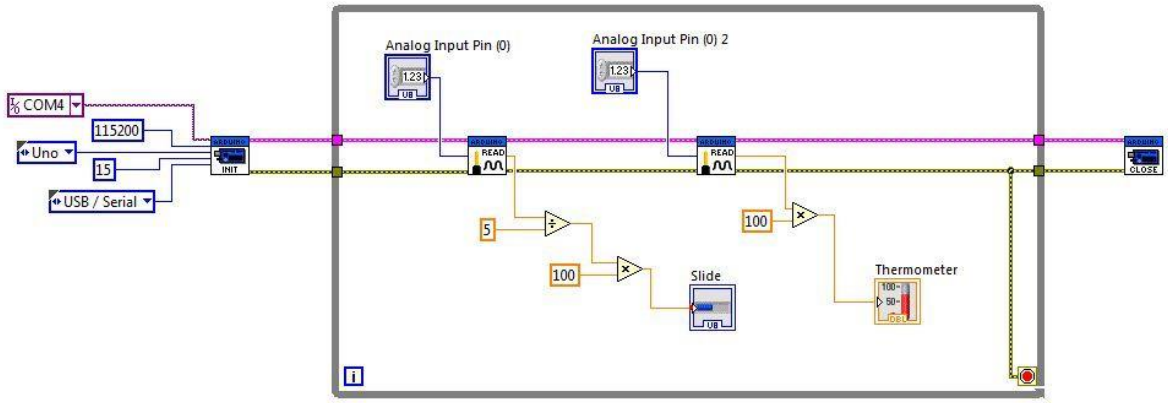
22. Join “Slide” and “temperature” as shown on block diagram of LabVIEW.



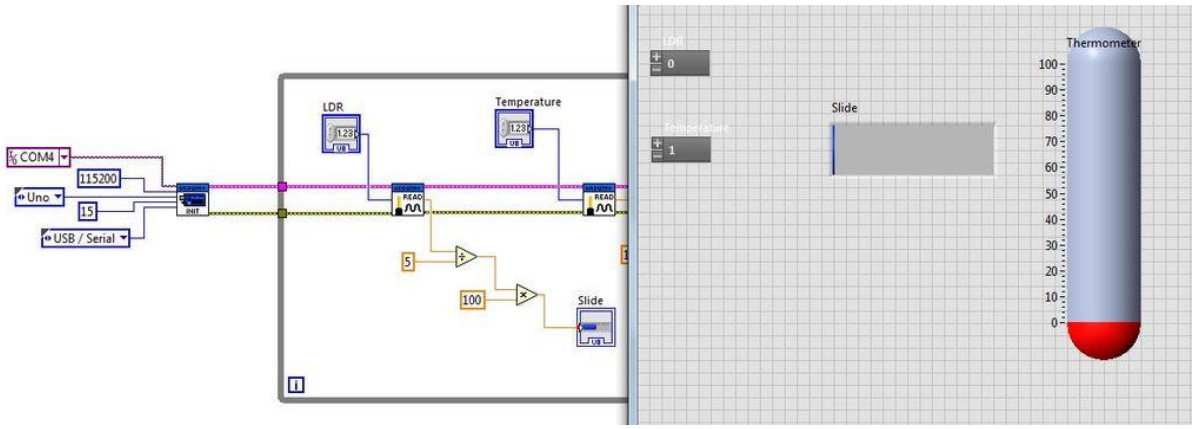
23. Create “Close” block as follows on block diagram of LabVIEW.



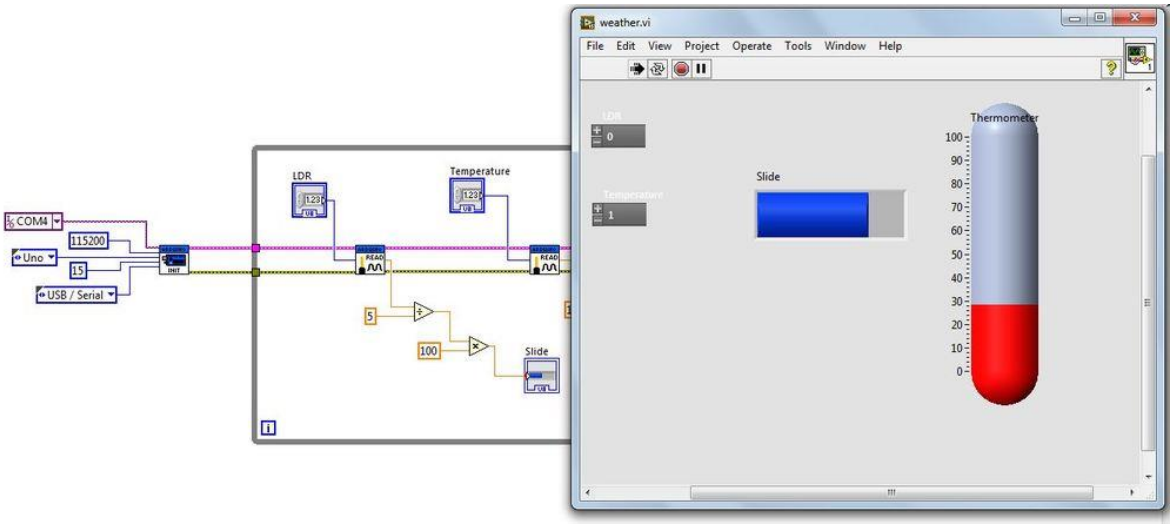
24. Join as shown on block diagram of LabVIEW.

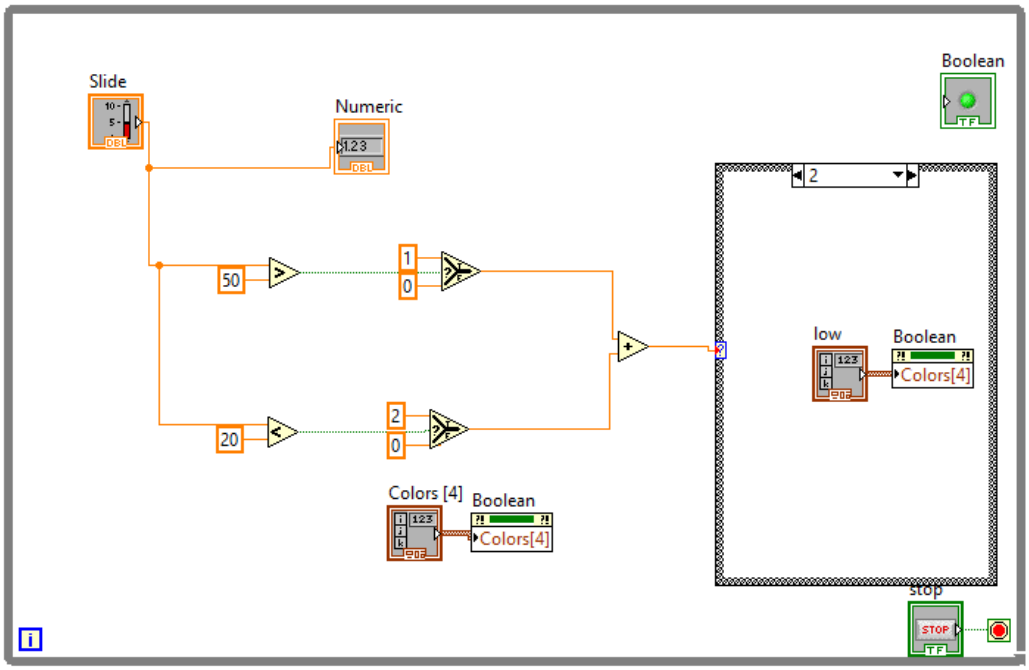


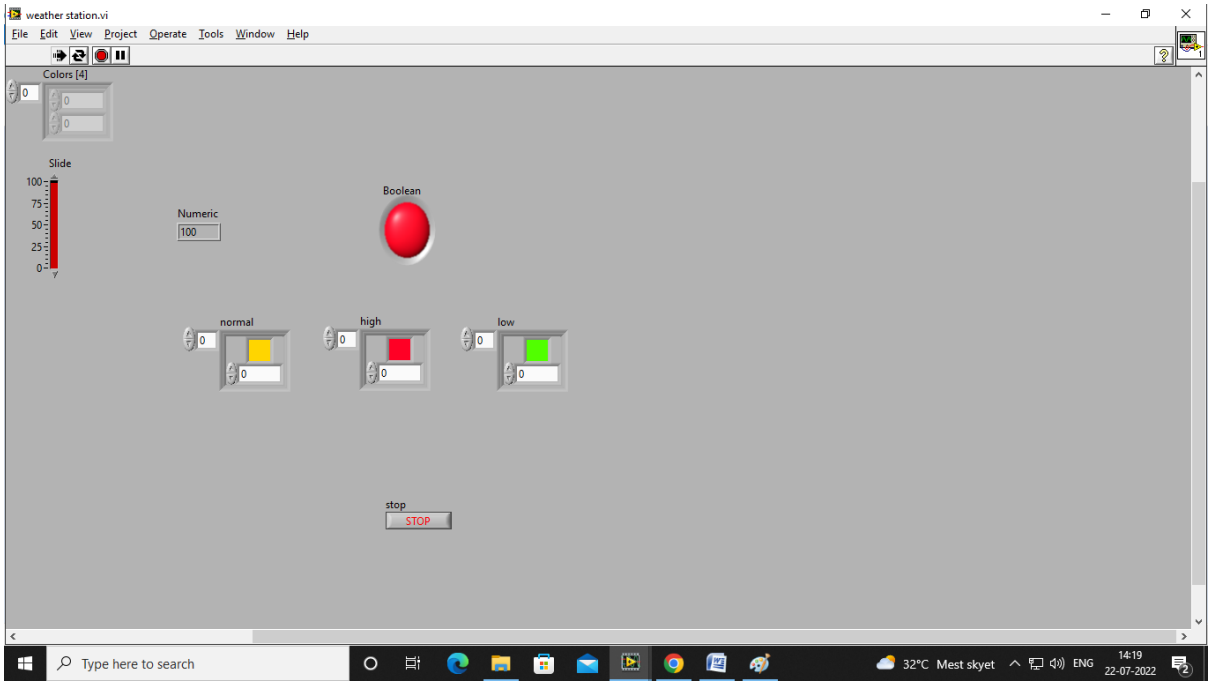
25. Now Start Ardunio IDE
26. Click “File” then “Open” and Follow as shown. Go through all these folders from “Computer” onward and open LIFA_BASE Arduino file
27. Upload the program opened in Arduino IDE using Arrow button on top of Arduino IDE
28. Once uploading done close the Arduino IDE. It’s very important to close Arduino IDE because both LabVIEW and Arduino are using COM4. If not closed LabVIEW will not be able to communicate and LabVIEW will crash
29. Go to front panel. Write “0” in LDR box and “1” in Temperature box. They indicate the arduino pins to which LDR and LM35 are connected in real time

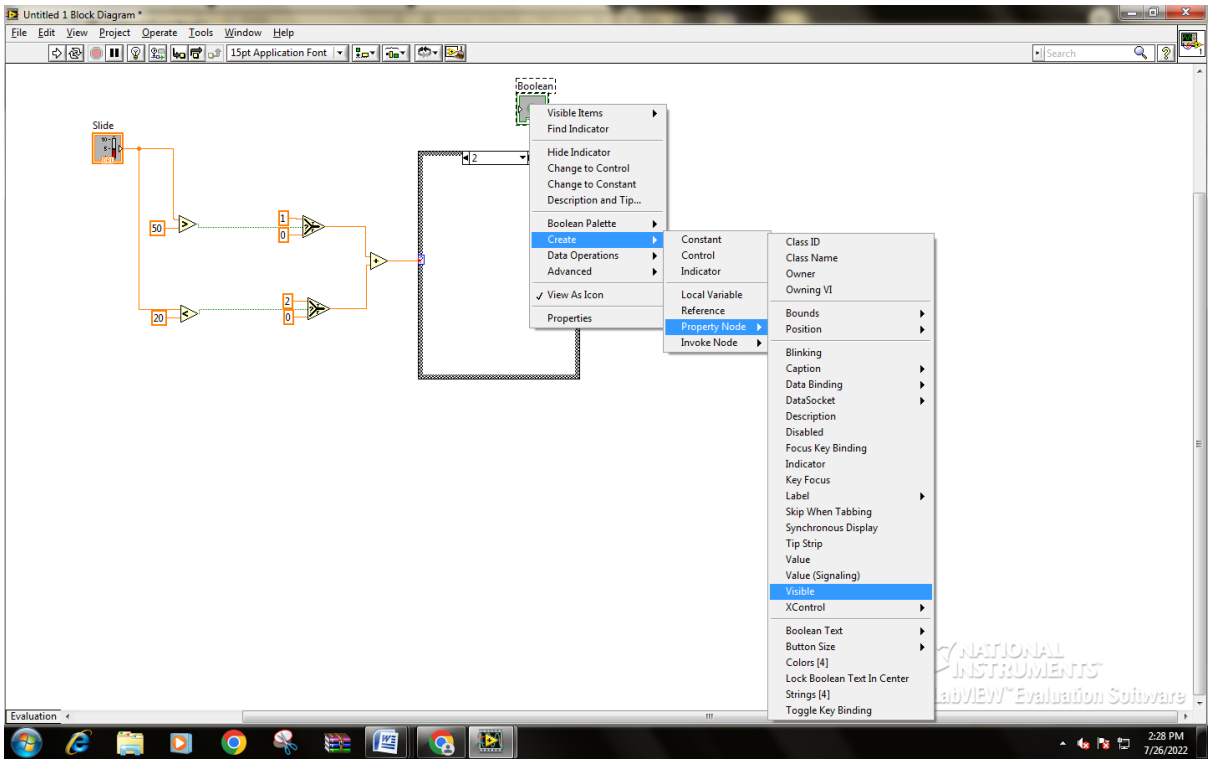


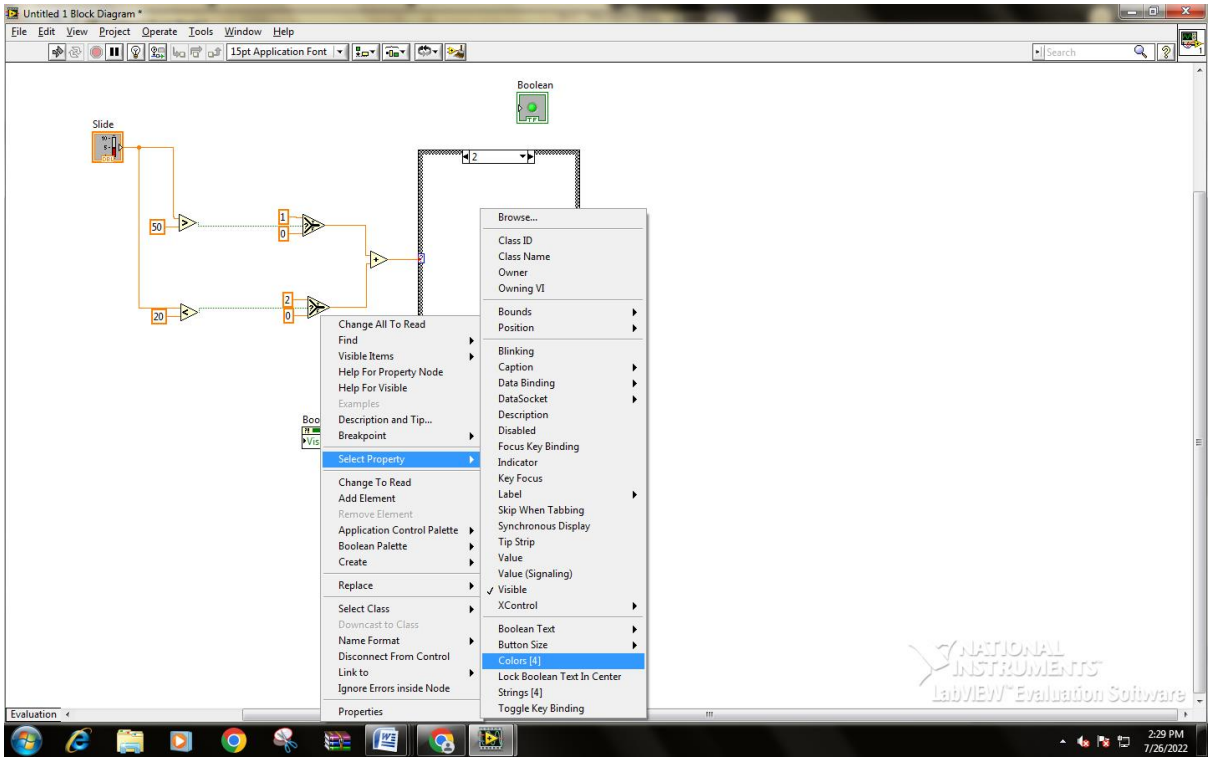
30. Run the program for weather station in toolbar menu of LabVIEW. Complete weather station status is as below when running the program

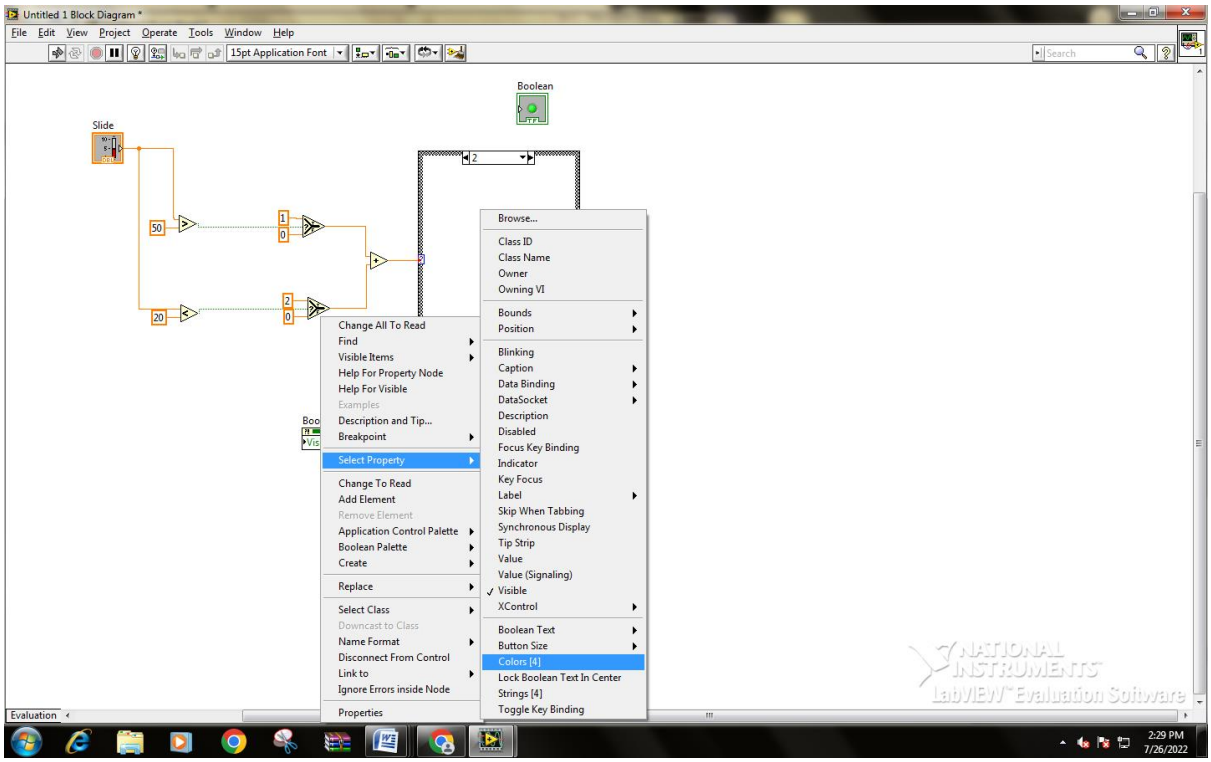

weather.vi
File Edit View Project Operate Tools Window Help
COM4
115200
Uno
15
USB / Serial
LDR
Temperature
5
100
Slide
Thermometer
100
90
80
70
60
50
40
30
20
10
0

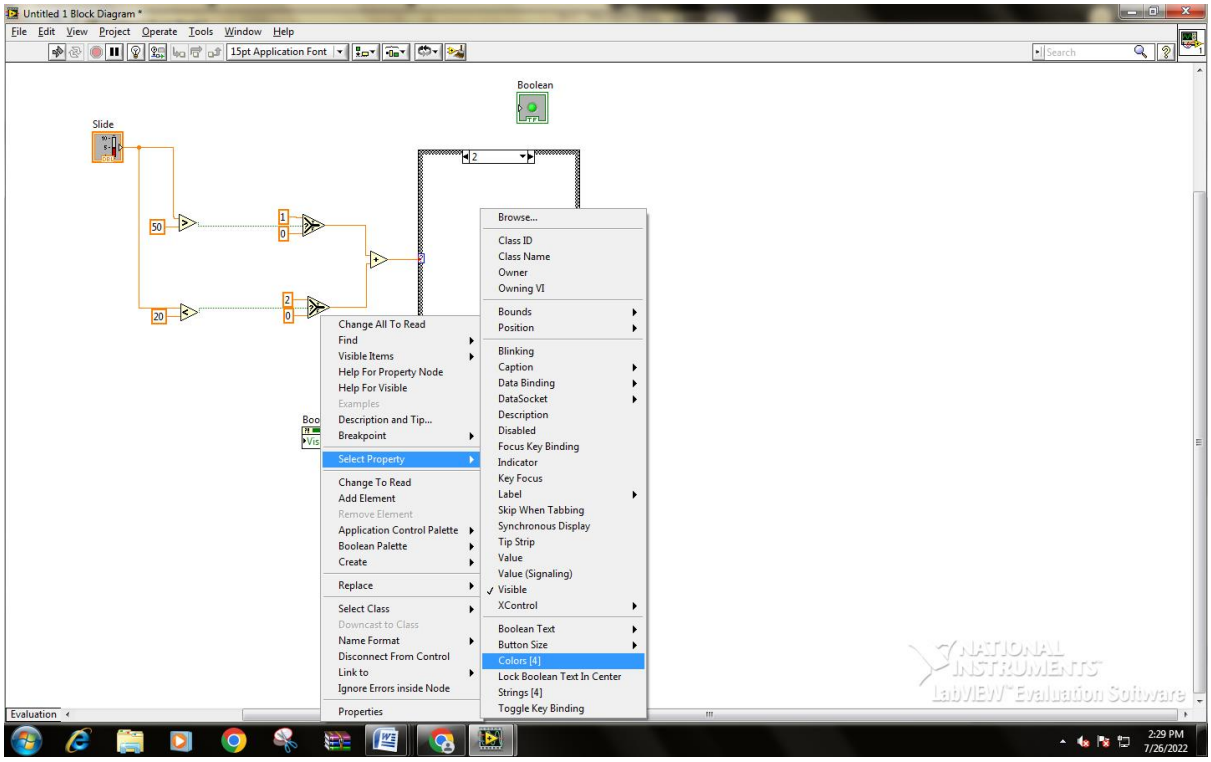

Slide
Numeric
Boolean
50
20
1
0
2
0
2
low
Boolean
Colors[4]
Colors [4]
Boolean
Colors[4]
stop
STOP


weather station.vi
File Edit View Project Operate Tools Window Help
Colors [4]
Slide
Numeric
100
Boolean
normal
high
low
stop
STOP
Type here to search
32°C Mest skyet
ENG
14:19
22-07-2022


Untitled 1 Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Search
Boolean
Slide
Visible Items
Find Indicator
Hide Indicator
Change to Control
Change to Constant
Description and Tip...
Boolean Palette
Create
Data Operations
Advanced
View As Icon
Properties
Constant
Control
Indicator
Local Variable
Reference
Property Node
Invoke Node
Class ID
Class Name
Owner
Owning VI
Bounds
Position
Blinking
Caption
Data Binding
DataSocket
Description
Disabled
Focus Key Binding
Indicator
Key Focus
Label
Skip When Tabbing
Synchronous Display
Tip Strip
Value
Value (Signaling)
Visible
XControl
Boolean Text
Button Size
Colors [4]
Lock Boolean Text In Center
Strings [4]
Toggle Key Binding
NATIONAL INSTRUMENTS
LabVIEW Evaluation Software
Evaluation
2:28 PM
7/26/2022


Untitled 1 Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Search
Boolean
Slide
50
20
2
Change All To Read
Find
Visible Items
Help For Property Node
Help For Visible
Examples
Description and Tip...
Breakpoint
Select Property
Change To Read
Add Element
Remove Element
Application Control Palette
Boolean Palette
Create
Replace
Select Class
Downcast to Class
Name Format
Disconnect From Control
Link to
Ignore Errors inside Node
Properties
Browse...
Class ID
Class Name
Owner
Owning VI
Bounds
Position
Blinking
Caption
Data Binding
DataSocket
Description
Disabled
Focus Key Binding
Indicator
Key Focus
Label
Skip When Tabbing
Synchronous Display
Tip Strip
Value
Value (Signaling)
Visible
XControl
Boolean Text
Button Size
Colors [4]
Lock Boolean Text In Center
Strings [4]
Toggle Key Binding
NATIONAL INSTRUMENTS
LabVIEW Evaluation Software
Evaluation
2:29 PM
7/26/2022


Untitled 1 Block Diagram *
File Edit View Project Operate Tools Window Help
15pt Application Font
Search
Boolean
Slide
50
20
2
Change All To Read
Find
Visible Items
Help For Property Node
Help For Visible
Examples
Description and Tip...
Breakpoint
Select Property
Change To Read
Add Element
Remove Element
Application Control Palette
Boolean Palette
Create
Replace
Select Class
Downcast to Class
Name Format
Disconnect From Control
Link to
Ignore Errors inside Node
Properties
Browse...
Class ID
Class Name
Owner
Owning VI
Bounds
Position
Blinking
Caption
Data Binding
DataSocket
Description
Disabled
Focus Key Binding
Indicator
Key Focus
Label
Skip When Tabbing
Synchronous Display
Tip Strip
Value
Value (Signaling)
Visible
XControl
Boolean Text
Button Size
Colors [4]
Lock Boolean Text In Center
Strings [4]
Toggle Key Binding
NATIONAL INSTRUMENTS
LabVIEW Evaluation Software
Evaluation
2:29 PM
7/26/2022



8.

Aim:

To study the MQTT Protocol and examine the components of MQTT protocol.

Introduction:

MQTT stands for Message Queuing Telemetry Transport. It is an extremely lightweight and publish-subscribe messaging transport protocol. This protocol is useful for the connection with the remote location where the bandwidth is a premium. It is a publish and subscribe system where we can publish and receive the messages as a client. It makes it easy for communication between multiple devices. It is a simple messaging protocol designed for the constrained devices and with low bandwidth, so it's a perfect solution for the internet of things applications.

Characteristics of MQTT

o It is a machine-to-machine protocol, i.e., it provides communication between the devices.

o It is designed as a simple and lightweight messaging protocol that uses a publish/subscribe system to exchange the information between the client and the server.

o It does not require that both the client and the server establish a connection at the same time.

o It provides faster data transmission, like how WhatsApp/messenger provides a faster delivery. It's a real-time messaging protocol.

o It allows the clients to subscribe to the narrow selection of topics so that they can receive the information they are looking for.

Components of MQTT

o Message

o Client

o Server or Broker

o TOPIC

Message

The message is the data that is carried out by the protocol across the network for the application. When the message is transmitted over the network, then the message contains the following parameters:

1. Payload data

2. Quality of Service (QoS)

3. Collection of Properties

4. Topic Name

Client

In MQTT, the subscriber and publisher are the two roles of a client. The clients subscribe to the topics to publish and receive messages.

Publish:

When the client sends the data to the server, then we call this operation as a publish.

Subscribe:

When the client receives the data from the server, then we call this operation a subscription.

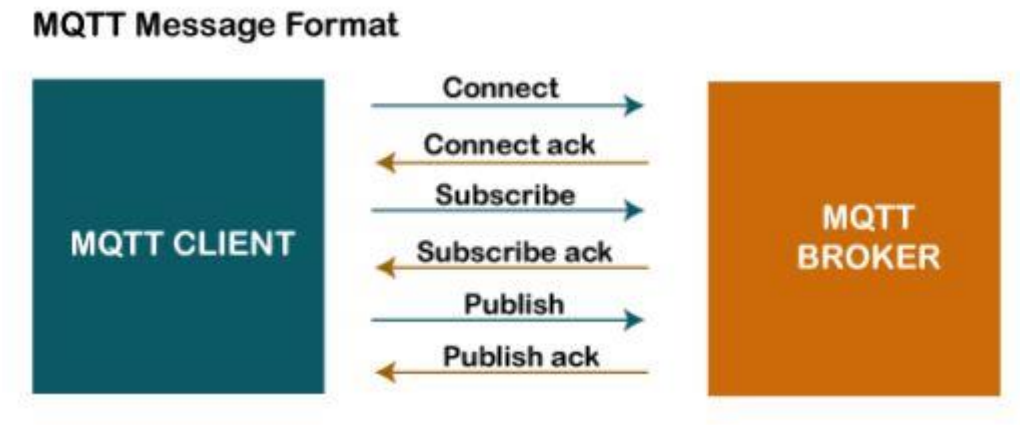
Server

The device or a program that allows the client to publish the messages and subscribe to the messages. A server accepts the network connection from the client, accepts the messages from the client, processes the subscribe and unsubscribe requests, forwards the application messages to the client, and closes the network connection from the client.

TOPIC

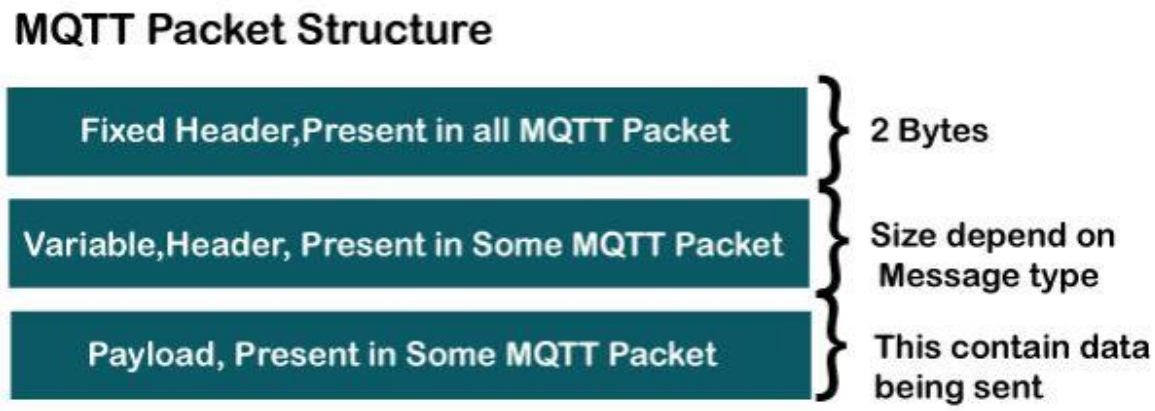
The label provided to the message is checked against the subscription known by the server is known as TOPIC.

MQTT Message Format



The MQTT uses the command and the command acknowledgment format, which means that each command has an associated acknowledgment.

MQTT Packet Structure



The MQTT message format consists of 2 bytes fixed header, which is present in all the MQTT packets. The second field is a variable header, which is not always present. The third

field is a payload, which is also not always present. The payload field basically contains the data which is being sent.

Conclusion: Thus, MQTT protocol and its components are studied and examined.