1. Study about embedded components such as sensors and actuators.

Sensor: A sensor is an object whose purpose is to detect events or changes in its environment, and then provide a corresponding output. A sensor is a type of transducer; sensors may provide various types of output, but typically use electrical or optical signals. For example, a thermocouple generates a known voltage (the output) in response to its temperature (the environment).

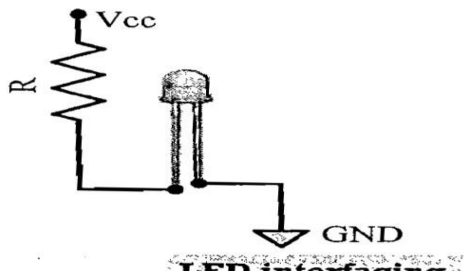
- A Sensor is used for taking Input
- It is a transducer that converts energy from one form to another for any measurement or control purpose

Ex. A Temperature sensor

Actuator: An actuator is a component of machines that is responsible for moving or controlling a mechanism or system. An actuator requires a control signal and source of energy. The control signal is relatively low energy and may be electric voltage or current, pneumatic or hydraulic pressure, or even human power
Actuator is used for output. It is a transducer that may be either mechanical or electrical which converts signals to corresponding physical actions.

I/O subsystem: The I/O sub system of the embedded system facilitates the interaction of embedded system with the external world

LED (Light Emitting Diode):
LED is an important output device for visual indication in any embedded system.LED used as an indicator for the status of various signals or situations. LED is a p-n junction diode and contains a **CATHODE** and **ANODE.** For functioning the anode is connected to +ve end of power supply and cathode is connected to –ve end of power supply. The maximum current flowing through the LED is limited by connecting a RESISTOR in series between the power supply and LED as shown in the figure below



LED interfacing

There are two ways to interface an LED to a microprocessor/microcontroller:

The Anode of LED is connected to the port pin and cathode to Ground : In this approach the port pin sources the current to the LED when it is at logic high(i.e. 1).

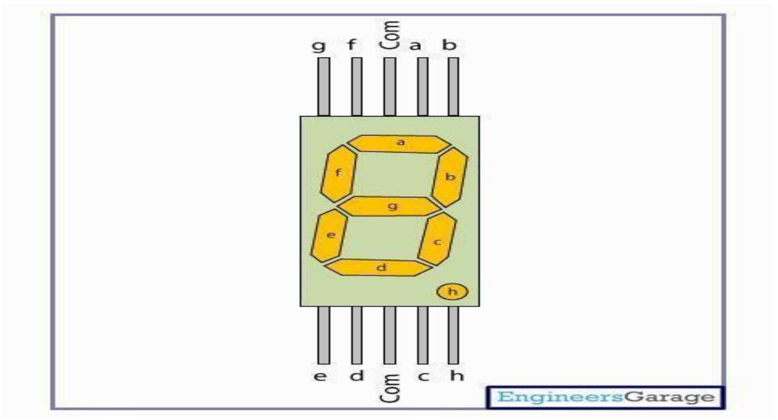
The Cathode of LED is connected to the port pin and Anode to Vcc : Here the port pin sinks the current and the LED is turned ON when the port pin is at Logic low (i.e. 0)

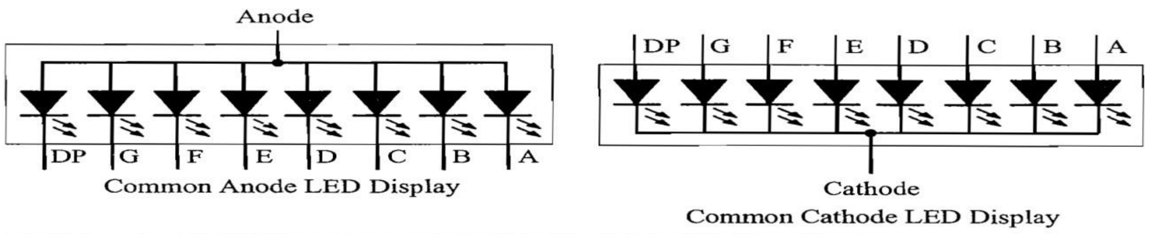
Segment LED display:

- **It** is the most basic electronic display device that can display digits from 0-9.
- They find wide application in devices that display numeric information like digital clocks,radio, microwave ovens, electronic meters etc.
- The most common configuration has an array of eight LEDs arranged in a special pattern to display these digits.
- They are laid out as a squared-off figure '8'. Every LED is assigned a name from 'a' to 'h'and is identified by its name.
- Seven LEDs 'a' to 'g' are used to display the numerals while eighth LED 'h' is used to display the dot/decimal.

A seven segment is generally available in ten pin package. While eight pins correspond to the eight LEDs, the remaining two pins (at middle) are common and internally shorted. These segments come in two configurations, namely, Common cathode (CC) and Common anode (CA). In CC configuration, the negative terminals of all LEDs are connected to the common pins. The common is connected to ground and a particular LED glows when its corresponding pin is given high. In CA arrangement, the common pin is given a high logic and the LED pins are given low to display a number.

Pin diagram:





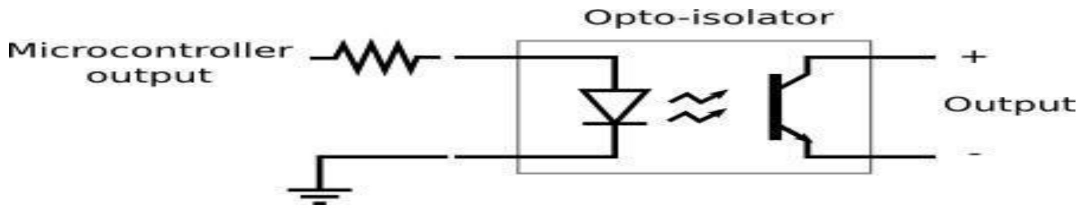
Common anode and cathode configurations of a 7-segment LED Display

For common cathode configurations, the anode of each LED segment is connected to the port pins of
the port to which the display is interfaced. The anode of the common anode LED display is connected to the SV supply voltage through a current limiting resistor and the cathodé of each LED segment is connected to the respective port pin lines. For an LED segment to lit in the Common anode LED configuration, the port pin to which the cathode of the LED segment is connected should be at logic 0.

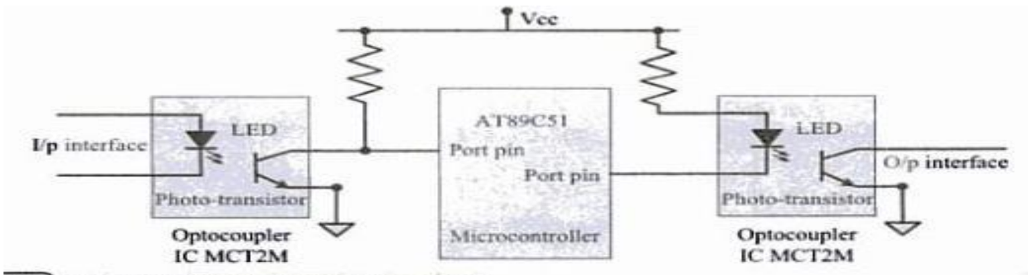
7 – segment LED display is popular choice for low cost embedded applications like, public telephone call monitoring devices, point of scale terminals, etc..

Optocoupler: Optocoupler is , also called photocoupler, or optical isolator, is a component that transfers electrical signals between two isolated circuits by using light. Optocoupler prevent high voltages from affecting the system receiving the signal. Commercially available opto- isolators withstand input-to-output voltages up to 10 kV and voltage transients with speeds up to 10 kV/μs.

An optocoupler consists of an LED and a phototransistor in the same opaque package.



when an electrical current is applied to the LED, infrared light is produced and passes through the material inside the optoisolator. The beam travels across a transparent gap and is picked up by the receiver, which converts the modulated light or IR back into an electrical signal. In the absence of light, the input and output circuits are electrically isolated from each other.



2.17 Optocoupler in Input and Output circuit

Optocoupler is available as ICs from different semiconductor manufacturers. The MCT2M IC from Fairchild semiconductor is an example for optocoupler IC.

STEPPER MOTOR

A stepper motor 1s an electro-mechanical device which generates discrete displacement (motion) in response, to de electrical signals. It differs from the normal de motor in its operation. The de motor produces continuous rotation on applying de voltage whereas a stepper motor produces discrete rotation in response to the de voltage applied to it. Stepper motors are widely used in industrial embedded applications, consumer electronic products and robotics control systems. The feed mechanism of a printer/fax makes use of stepper motors for its functioning.

Based on the coil winding arrangements, a two-phase stepper motor is classified into two. They are:

1. Unipolar
2. Bipolar

1. **Unipolar**

   A unipolar stepper motor contains two windings per phase. The direction of rotation (clockwise or anticlockwise) of a stepper motor is controlled by changing the direction of current flow. Current in one direction flows through one coil and in the opposite direction flows through the other coil. It is easy to shift the direction of rotation by just switching the terminals to which the coils are connected.

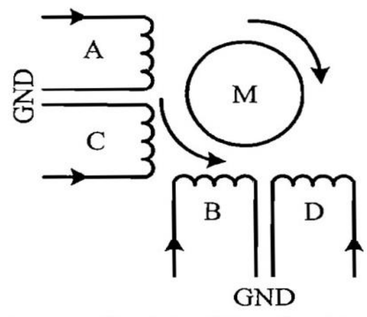


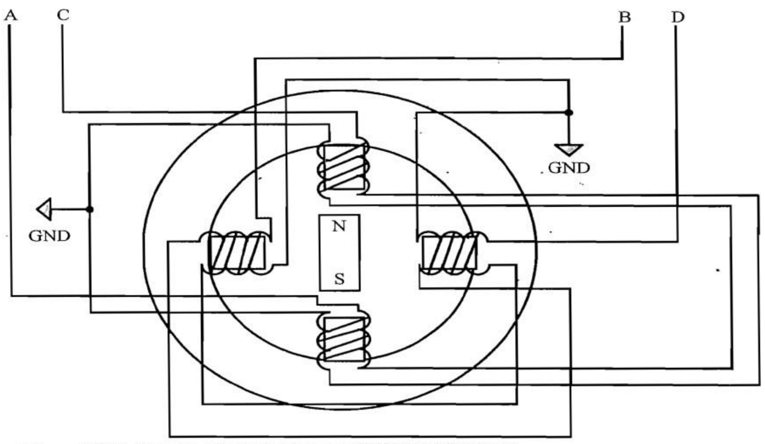
Fig. 2.18 2-Phase unipolar stepper motor

Figure 2.18 illustrates the working of a two-phase unipolar stepper motor.

The coils are represented as A, B, C and D. Coils A and C carry current in opposite directions for phase 1 (only one of them will be carrying current one at a time)

2.Bipolar:

A bipolar stepper motor contains single winding per phase.For reversing the motor rotation the current flow through the winding is reversed dynamically. It requires complex circuitry for current flow reversal. The stator winding details for a two phase unipolar stepper motor is shown in Fig 2.19



Stator Winding details for a 2 Phase unipolar stepper motor

The stepping of stepper motor can be implemented in different ways by changing the sequence of activation of the stator windings. The different stepping modes supported by stepper motor are explained below.

Full Step.

In the full step mode both the phases are energised simultaneously. The coils A, B, C and D are energised in the following order:

STEP	COIL A	COIL B	COIL C	COIL D
1	H	H	L	L
2	L	H	H	L
3	L	L	H	H

4	H	L	L	H

It should be noted that out of the two windings, only one winding of a phase is energized at a time

Wave Step

In the wave step mode only one phase is energized at a time and each coils of the phase is energized alternatively .The coils A,B,C and D are energized in the following order

STEP	COIL A	COIL B	COIL C	COIL D
1	H	H	L	L
2	L	H	L	L
3	L	L	H	L
4	L	L	L	H

Half Step

It uses the combination of wave & full step.It has the highest torque and stability. The coil energizing sequence for half step is given below

STEP	COIL A	COIL B	COIL C	COIL D
1	H	L	L	L
2	L	H	L	L
3	L	H	L	L
4	L	H	H	L
5	L	L	H	L
6	L	L	H	H
7	L	L	L	H
8	H	L	L	H

The rotation of the stepper motor can be reversed by reversing the order in which the coil is energized ,Two-phase unipolar stepper motors are the popular choice for embedded applications. The current requirement for stepper motor is little high and hence the port pins of a microcontroller/processor may not be able to drive them directly. Also the supply voltage required to operate stepper motor varies normally in the range SV o 24 V. Depending on the current and voltage requirements, special driving circuits are required to interface the stepper motor with microcontroller/processors. Commercial off-the-shelf stepper motor driver ICs are available in the market and they can be directly interfaced to the microcontroller port. ULN2803 is an octal peripheral driver array available from ON semiconductors and ST microelectronics for driving a SV stepper motor. Simple driving circuit can also be built using transistors.

The following circuit diagram (Fig 2.20) illustrates the interfacing of a stepper motor through driver circuit connected to the port pins of a microcontroller/processor.

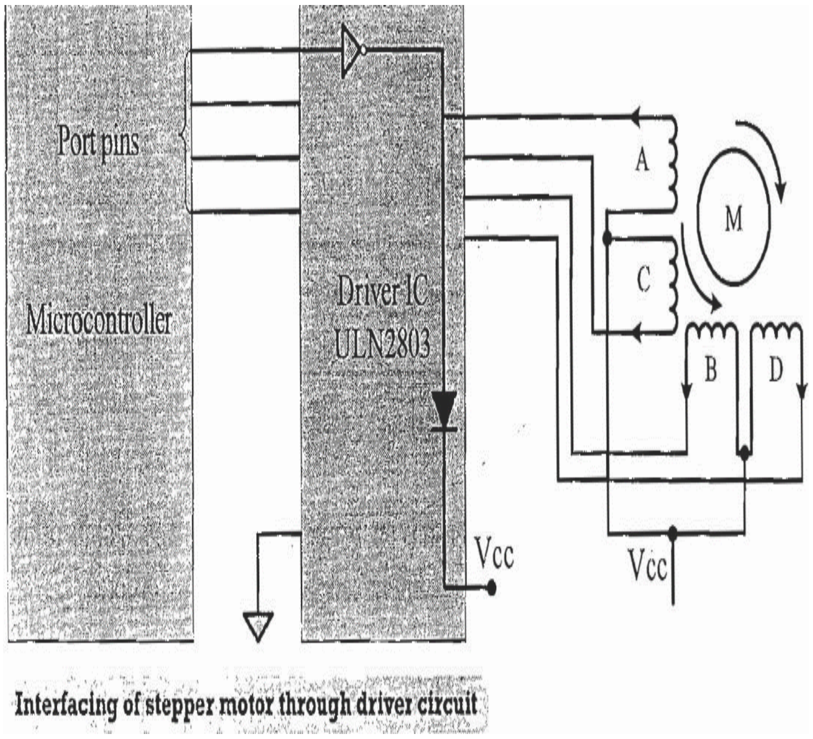


Fig 2.20 Interfacing of a Stepper Motor

Push Button

Push Button Switch It is an input device. Push button switch comes in two configurations, namely "Push to Make' and 'Push to Break'. In the 'Push to Make' configuration, the switch is normally in the open state and it makes a circuit contact when it is pushed or pressed. In the 'Push to Break' con-figuration, the switch is normally in the closed state and it breaks the circuit contact when it is pushed or pressed. The push button stays in the 'closed' (For Push to Make type) or 'open' (For Push to Breaktype) state as long as it is kept in the pushed state and it breaks/makes the circuit connection when it is released. Push button is used for generating a mo-mentary pulse. In embedded application push button is generally used as reset and start switch and pulse |generator. The Push button is normally connected to the port pin of the host processor/controller. Depend- port pin Port pin,on the way in which the push button interfaced to the controller, it can generate either a 'HIGH' pulse or 'LOW' pulse. Figure 2.23 illustrates how the pushbutton can be used for generating "LOW" and 'HIGH'pulses**.**

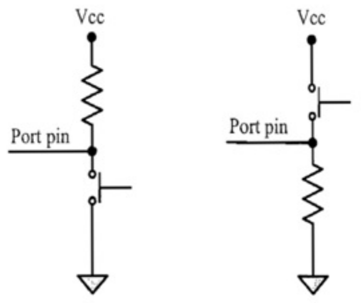


Fig 2.23 PushButton LOW and HIGH pulse

EXERCISE 2

Study about Industrial Revolution and Cyber physical system opportunities and challenges.

A **cyber-physical system** (**CPS**) or intelligent system is a computer system in which a mechanism is controlled or monitored by computer-based algorithms. In cyber-physical systems, physical and software components are deeply intertwined, able to operate on different spatial and temporal scales, exhibit multiple and distinct behavioral modalities, and interact with each other in ways that change with context.CPS involves transdisciplinary approaches, merging theory of cybernetics, mechatronics, design and process science. The process control is often referred to as embedded systems. In embedded systems, the emphasis tends to be more on the computational elements, and less on an intense link between the computational and physical elements. CPS is also similar to the Internet of Things (IoT), sharing the same basic architecture; nevertheless, CPS presents a higher combination and coordination between physical and computational elements.

Examples of CPS include smart grid, autonomous automobile systems, medical monitoring, industrial control systems, robotics systems, and automatic pilot avionics.Precursors of cyber-physical systems can be found in areas as diverse as aerospace, automotive, chemical processes, civil infrastructure, energy, healthcare, manufacturing, transportation, entertainment, and consumer appliances.

INDUSTRY 4.0

Industry 4.0 refers to the fourth industrial revolution, although it is concerned with areas that are not usually classified as industry applications in their own right, such as smart cities

Fourth Industrial Revolution

The first industrial revolution came with the advent of mechanisation, steam power and water power.This was followed by the second industrial revolution, which revolved around mass production and assembly lines using electricity.The third industrial revolution came with electronics, I.T. systems and automation, which led to the fourth industrial revolution that is associated with cyber physical systems.

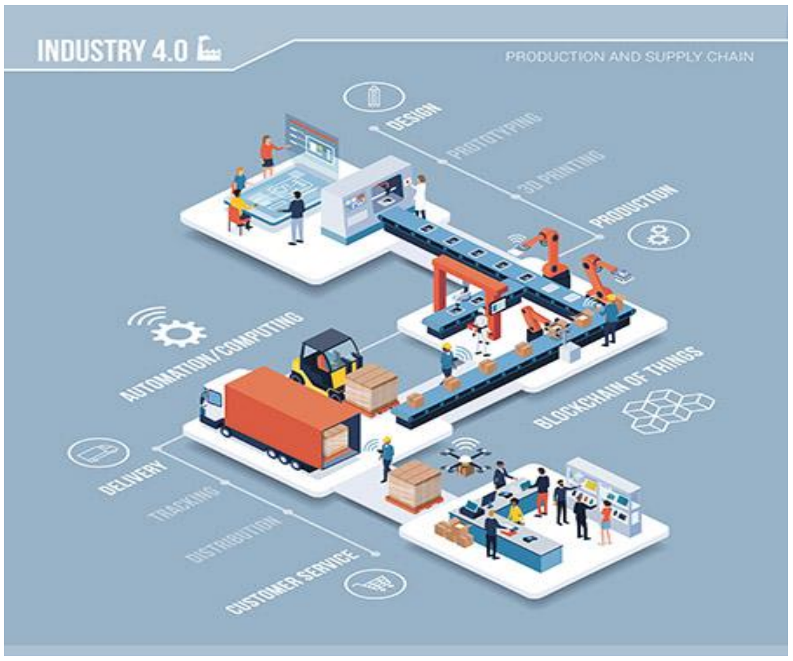


Figure 1: Industry 4.0

Industry 4.0 Technologies

Generally-speaking, Industry 4.0 describes the growing trend towards automation and data exchange in technology and processes within the manufacturing industry, including:

- The internet of things (IoT)
- The industrial internet of things (IIoT)
- Cyber-physical systems (CPS)
- Smart manufacture
- Smart factories
- Cloud computing
- Cognitive computing
- Artificial intelligence

This automation creates a manufacturing system whereby machines in factories are augmented with wireless connectivity and sensors to monitor and visualise an entire production process and make autonomous decisions.

Wireless connectivity and the augmentation of machines will be greatly advanced with the full roll out of 5G. This will provide faster response times, allowing for near real time communication between systems.

The fourth industrial revolution also relates to digital twin technologies. These digital technologies can create virtual versions of real-world installations, processes and applications. These can then be robustly tested to make cost-effective decentralised decisions.

These virtual copies can then be created in the real world and linked, via the internet of things, allowing for cyber-physical systems to communicate and cooperate with each other and human staff to create a joined up real time data exchange and automation process for Industry 4.0 manufacturing.

This automation includes interconnectivity between processes, information transparency and technical assistance for decentralised decisions.

In short, this should allow for digital transformation. This will allow for automated and autonomous manufacturing with joined-up systems that can cooperate with each other.

The technology will help solve problems and track processes, while also increasing productivity.

Example of the Industry 4.0 Revolution

Industry 4.0 has already been demonstrated through business models such as offline programming and adaptive control for arc welding, taking the process from product design through simulation and onto the shop floor for production.

There are also examples of businesses implementing Industry 4.0 in automotive manufacture and a variety of smart factories across the world.

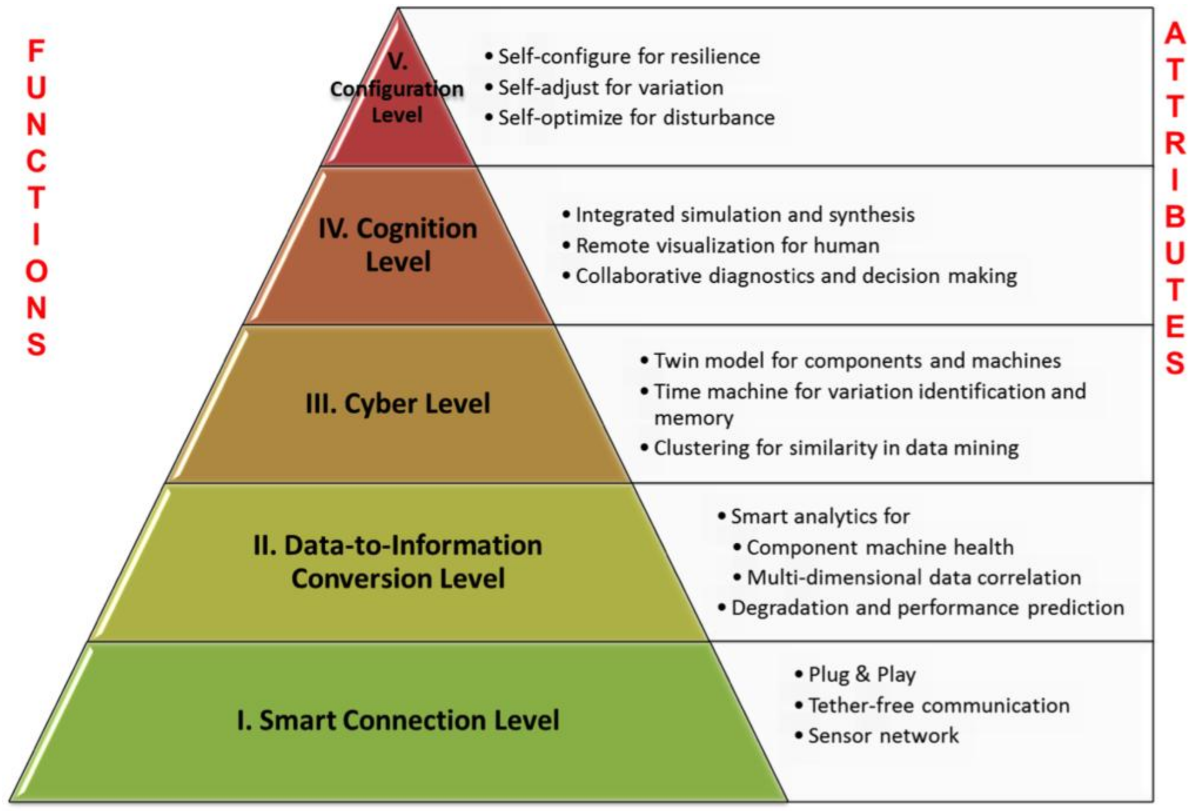

FUNCTIONS
ATTRIBUTES
V. Configuration Level
• Self-configure for resilience
• Self-adjust for variation
• Self-optimize for disturbance
IV. Cognition Level
• Integrated simulation and synthesis
• Remote visualization for human
• Collaborative diagnostics and decision making
III. Cyber Level
• Twin model for components and machines
• Time machine for variation identification and memory
• Clustering for similarity in data mining
II. Data-to-Information Conversion Level
• Smart analytics for
• Component machine health
• Multi-dimensional data correlation
• Degradation and performance prediction
I. Smart Connection Level
• Plug & Play
• Tether-free communication
• Sensor network

Figure 2: Industry 4.0 Revolution

A challenge in the development of embedded and cyber-physical systems is the large differences in the design practice between the various engineering disciplines involved, such as software and mechanical engineering. Additionally, as of today there is no "language" in terms of design practice that is common to all the involved disciplines in CPS. Today, in a marketplace where rapid innovation is assumed to be essential, engineers from all disciplines need to be able to explore system designs collaboratively, allocating responsibilities to software and physical elements, and analyzing trade-offs between them. Recent advances show that coupling disciplines by using co-simulation will allow disciplines to cooperate without enforcing new tools or design methods. Results from the MODELISAR project show that this approach is viable by proposing a new standard for co-simulation in the form of the Functional Mock-up Interface.

CYBER PHYSICAL SYSTEM OPPORTUNITIES

The science of **CPS has the capability to impact technology in a wide variety of industries and organizations**, and CPS allows us to imagine, create, develop, refine and perpetuate smart systems in fields that result in the betterment of industries, communities and individuals.

Regarding the benefits of cyber-physical systems technology, according to the Information Technology Laboratory: "The ability to design and build successful cyber-physical systems will address many national priorities in ways that traditional computer science cannot in areas as diverse as **aerospace, automotive, energy, disaster response, health care, manufacturing and city management**. Standards, protocols and test methods that support the discovery, interoperability and composition of components used to build these cyber physical systems will promote innovation, improve economic viability at the same time allowing systems to become more efficient and reduce resource-use."

In other words, the immeasurable benefits of cyber-physical systems is the acceleration of technological progress — progress that positively impacts countless fields, organizations and ultimately, the lives of others.

Here are just a few ways in which the benefits of CPS technology have impacted different industries:

Smart city management

TechTarget defines a smart city as “a municipality that uses information and communication technologies to increase operational efficiency, share information with the public and improve both the quality of government services and citizen welfare.” A smart city ecosystem is complex, including systems related to intelligent traffic management, emergency response technologies and public safety solutions — the use of cyber-physical systems technology is paramount in planning, implementing and improving the optimization of smart cities.

Infrastructure

In 2021, the United States’ infrastructure received a C- rating from the American Society of Civil Engineers’ Report Card for America’s Infrastructure.

What is the solution to our unstable infrastructure? Improving infrastructure starts with technology. Using advanced digital technologies like IoT sensors and video cameras, smart infrastructure empower smart cities to enhance the experience of citizens, businesses and city operators.

CPS engineers must master cutting-edge technologies in order to upgrade existing city infrastructure systems — many of which have not been updated in years. Bringing novel technologies to physical frameworks allows for more upgrades to critical infrastructure.

Automotive

IoT and CPS technology have made specific advancements in smart car technologies that can make vehicle transportation safer; “blind-spot monitoring, lane-departure warning and forward collision warning” are just three features that, if implemented in all cars in the United States, could reduce the number of crashes and in turn, save millions of dollars a year.

Agriculture

Sometimes referred to as smart agriculture or digital farming, CPS-related technology has resulted in advancements that help drive efficiencies on farms: from drones and satellites that relay images related to plant health to smart sensors on tractors or harvesters that provide information on soil type and condition.

Sustainability

Society continues to seek more solutions to today’s overwhelming need for sustainable practices in business, health care and countless other industries, and CPS technology often makes the advancement of these solutions possible — solutions such as public electric transport, affordable energy storage, accessible solar power and clean recycling initiatives are a direct result of IoT and CPS technology.

Security

The emergence of smart technology has increased security measures in a variety of ways. “From mobile app development for real-time remote monitoring to fully fledged intelligent surveillance systems,” CPS technology has empowered smart security to advance and improve.

Health care

CPS technology has made a multitude of medical advancements possible. From a smart monitoring system that tracks cancer patients’ response to treatment to a smart continuous tool that sends data on glucose levels to the wearer’s smartphone, the field of health care has greatly improved as a result of CPS science and technology.

Challenges and risks in Cyber-Physical Systems related to Industry 4.0 are:

- Data protection and data security.
- Lack of benefit quantification.
- Lack of prioritization by top management.
- Industrial broadband structure.
- Industrial espionage/sabotage.
- Production outages due to non availability of data.

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

- It is a machine-to-machine protocol, i.e., it provides communication between the devices.
- 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.
- It does not require that both the client and the server establish a connection at the same time.
- It provides faster data transmission, like how WhatsApp/messenger provides a faster delivery. It's a real-time messaging protocol.
- 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

- **Message**
- **Client**
- **Server or Broker**
- **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.

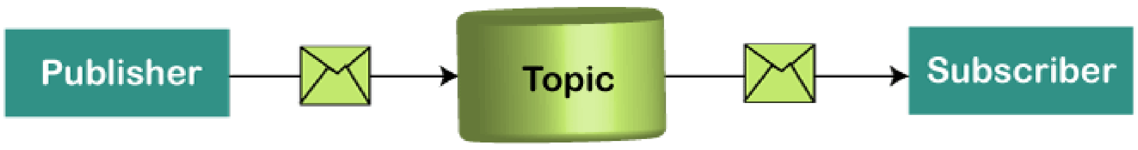
Publish: When the client sends the data to the server, then we call this operation as a publish.

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

Server

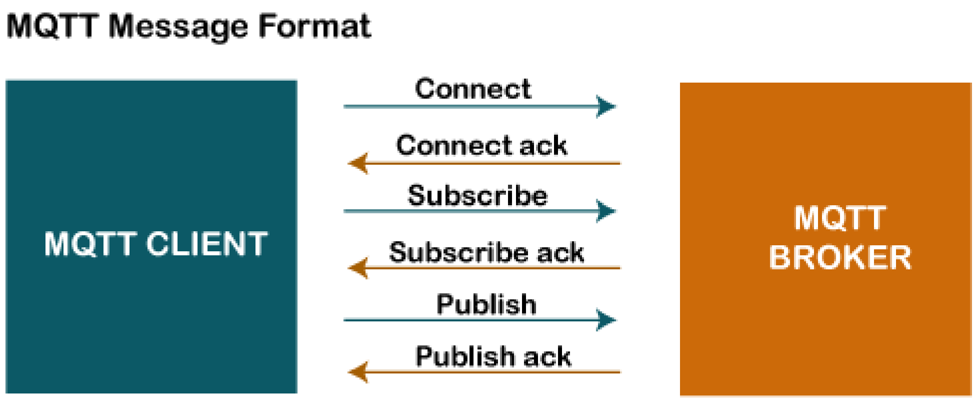
The device or a program that allows the client to publish the messages and subscribe to the messages. A server accepts the network connection from the client, accepts the messages from the client, processes the subscribe and unsubscribe requests, forwards the application messages to the client, and closes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

TOPIC



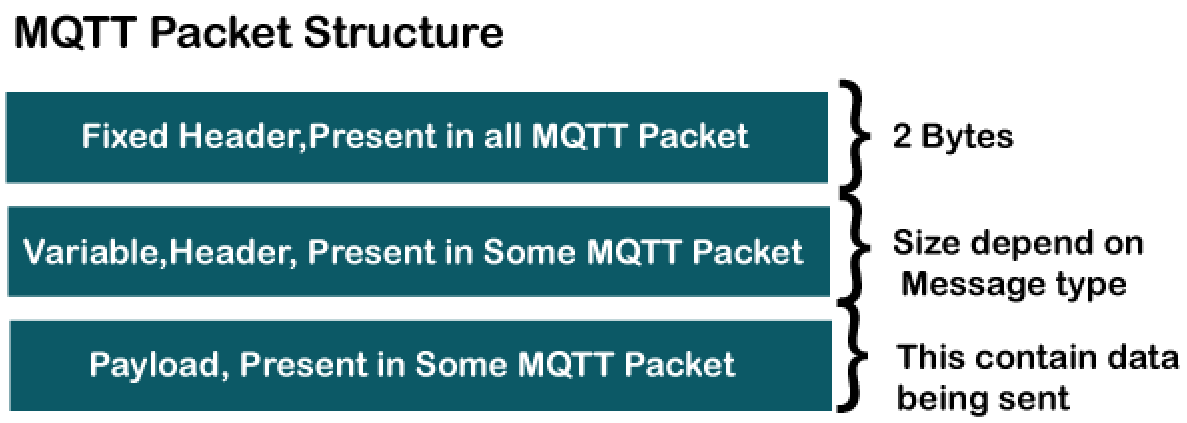
The label provided to the message is checked against the subscription known by the server is known as TOPIC.

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


MQTT Packet Structure
Fixed Header,Present in all MQTT Packet
Variable,Header, Present in Some MQTT Packet
Payload, Present in Some MQTT Packet
2 Bytes
Size depend on Message type
This contain data being sent

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.

<u>3.Create a program that blinks the LED on the development board using MBED software</u>

<u>Aim:</u>To write a program for blink led using tinker Cad.

<u>Components Used:</u>

1.Arduino Uno

2. LED(Light Emitting Diode)

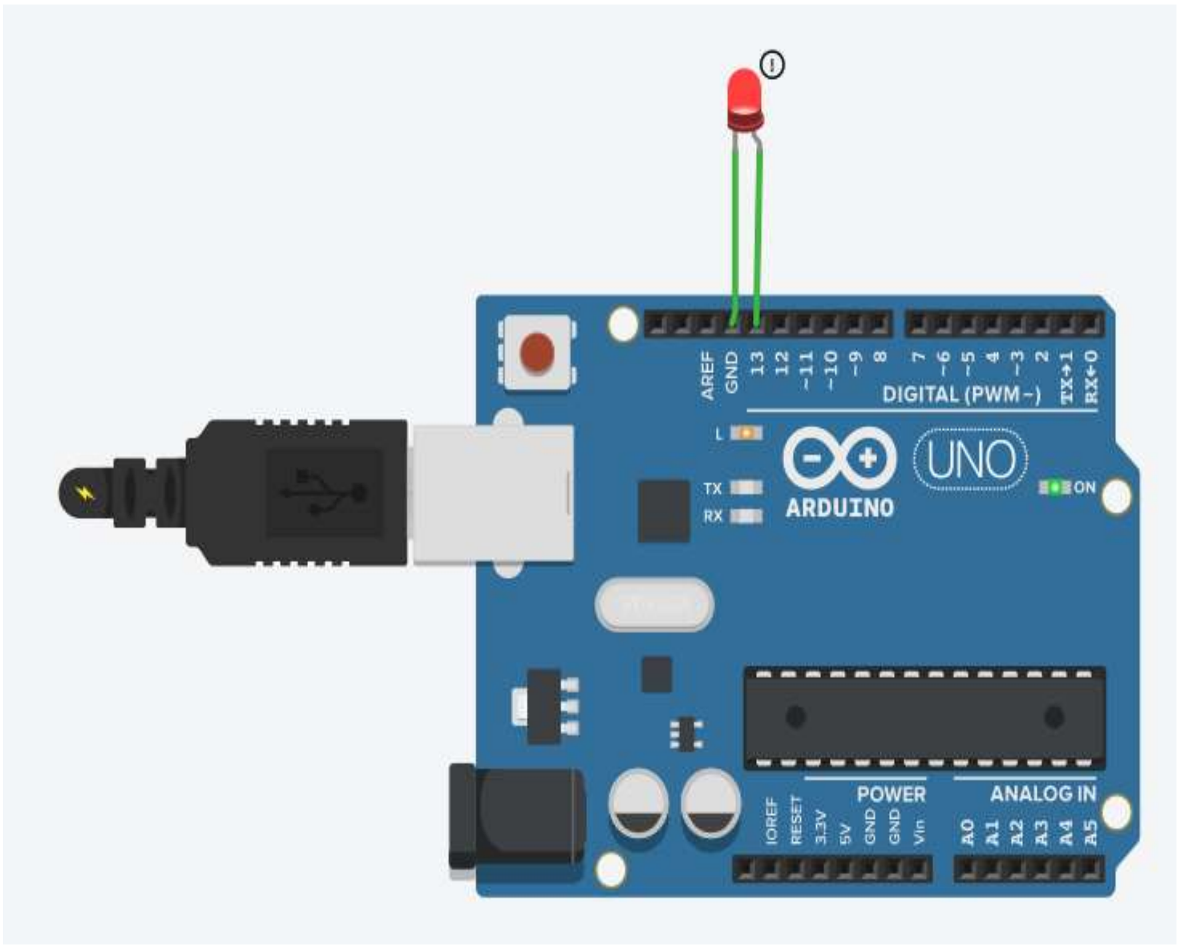
<u>Procedure:</u>

1.Get the Arduino uno board from the components

2.Get the LED from the components

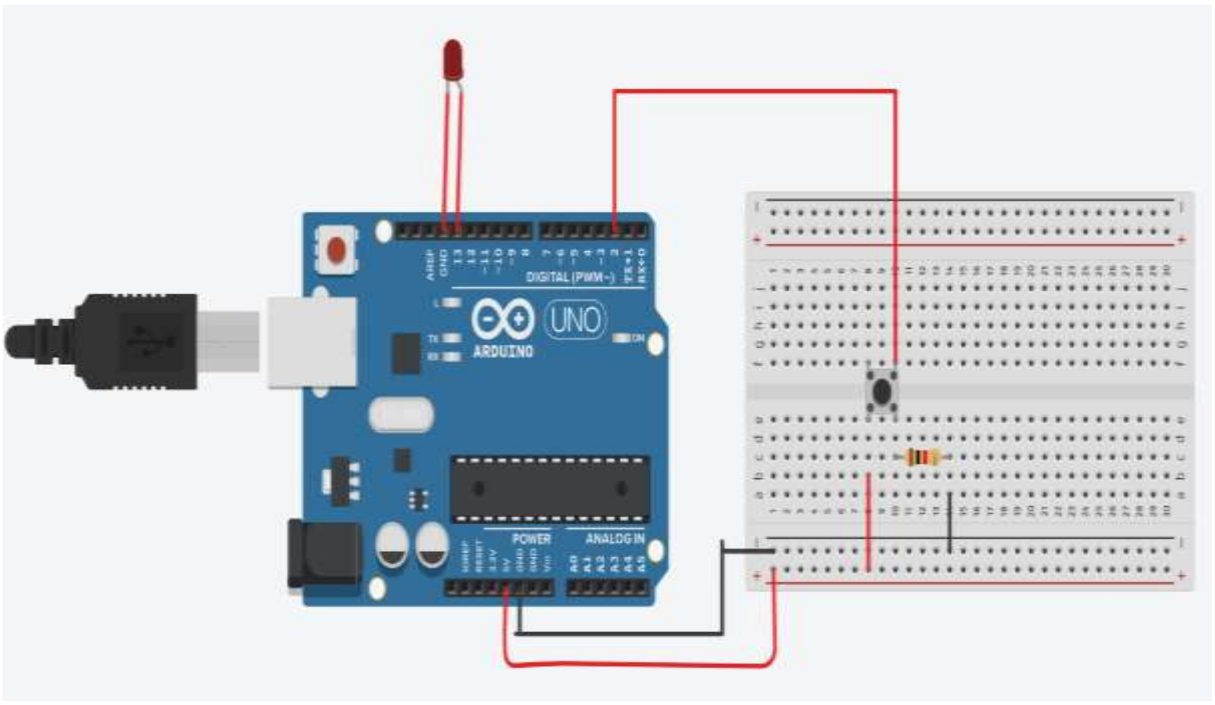
3.LED has two side which is positive (anode) and Negative(Cathode).Negative side is connected to the Ground(GND).Positive side is connected to Digital pin 13 of Arduino .

<u>3.1 Code:</u>



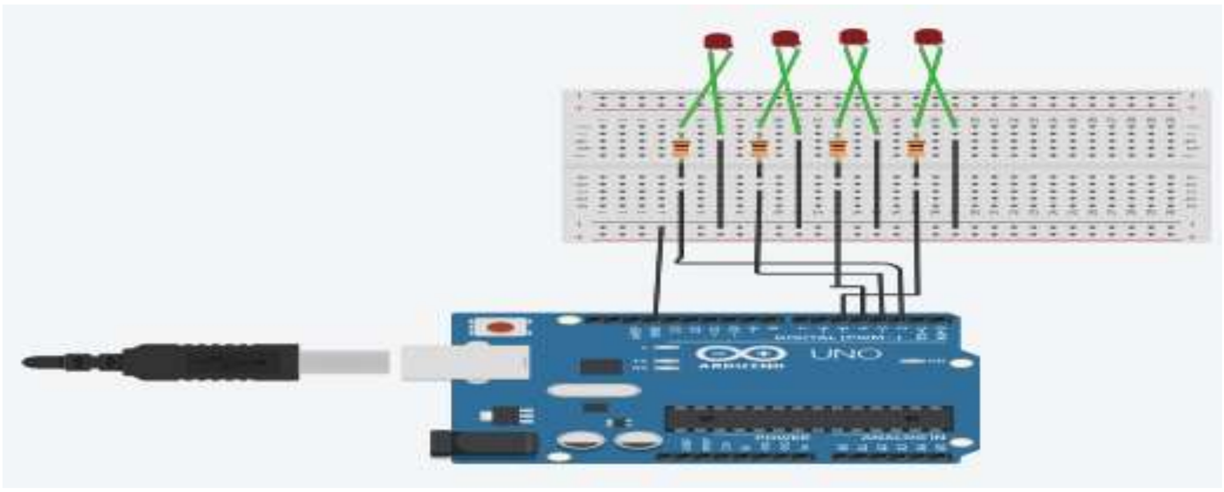
```
void setup()
{
pinMode(13, OUTPUT);

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

<u>3.2.Through Button blink LED</u>



```
void setup()
{
 pinMode(2, INPUT);
 pinMode(13,OUTPUT);
}
void loop()
{
 if(digitalRead(2)==1)
 {
  digitalWrite(13,HIGH);
 }
 else
 {
  digitalWrite(13,LOW);
 }
}
```

3.3. 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);
}
```

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

A) Analog potentiometer

Aim :To write a program for analog potentiometer using tinker Cad.

Procedure:

1.Get the Arduino Uno board from the components

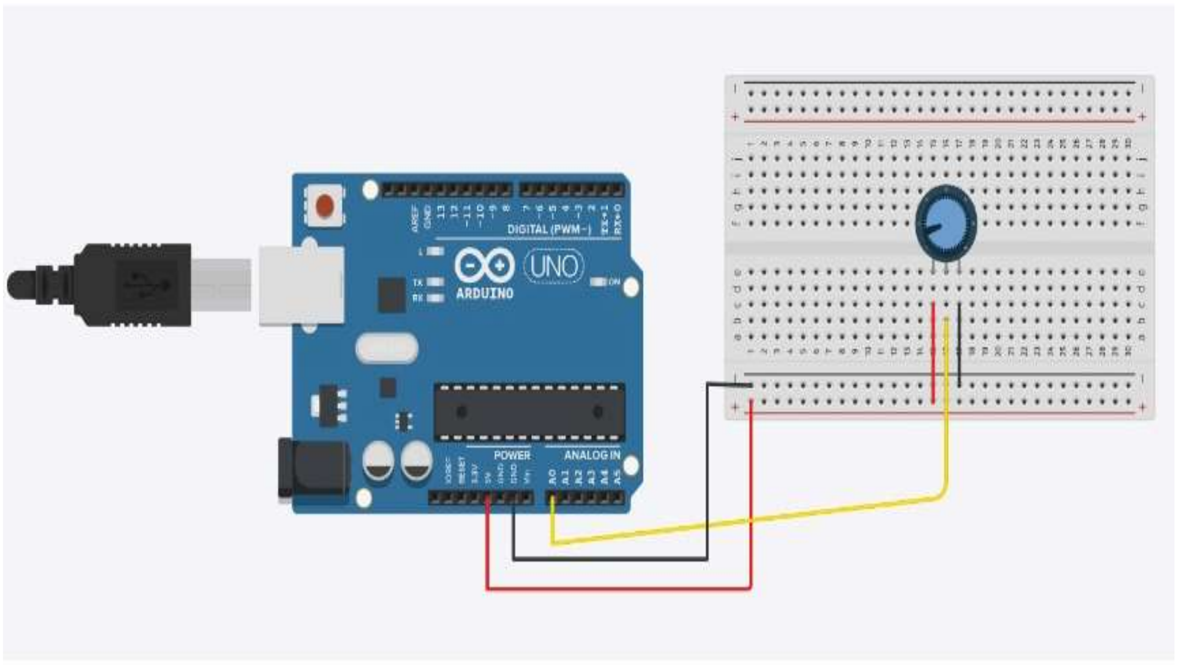
2.Get the bread board from the components

3.Get the potentiometer from component .The potentiometer has 3 pins. First is connected to 5 v .Second pin is connected to A0 in analog pin.Third pin is connected into a gnd

Code :

```
Int pot=A0;
Void setup()
{
Serial.begin(9600);
}
Void loop()
{
  Int potvalue=analogRead(pot);
  Serial.print("pot value");
  Serial.println(potvalue);
  Delay(1);
}
```

Prototype :



Output :

```
Pot value818
Pot value777
Pot value716
Pot value696
Pot value675
Pot value675
```

Result : The above experiment is executed successfully

B) Reading sensor

Aim : To write a program for reading sensors using tinker Cad.

Procedure:

1.Get the Arduino Uno board from the components

2.Get the photoresistor from the components

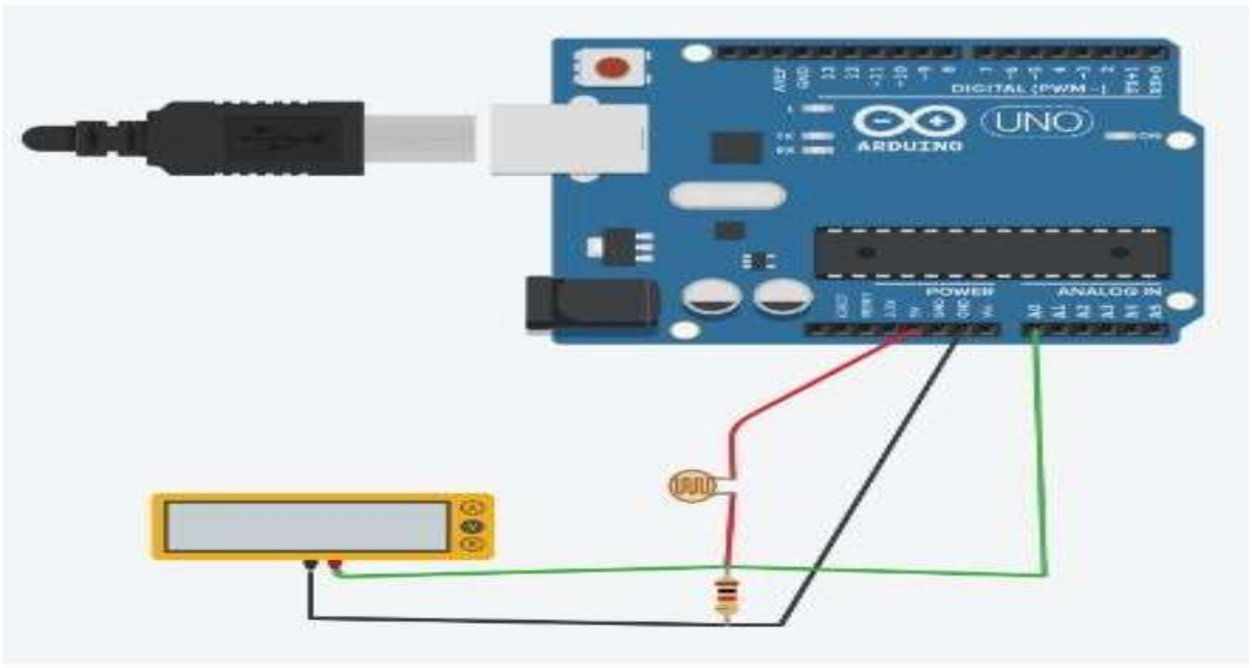
3.Get the resistor from the components

4. Get the multimeter from the components

Code :

```
Void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);
}
Void loop()
{
   Int lightvalue=analogRead(A0);
   Serial.println(lightvalue);
    Delay(1000);
}
```

Prototype:



Output :

```
6
379
526
640
658
663
654
476
```

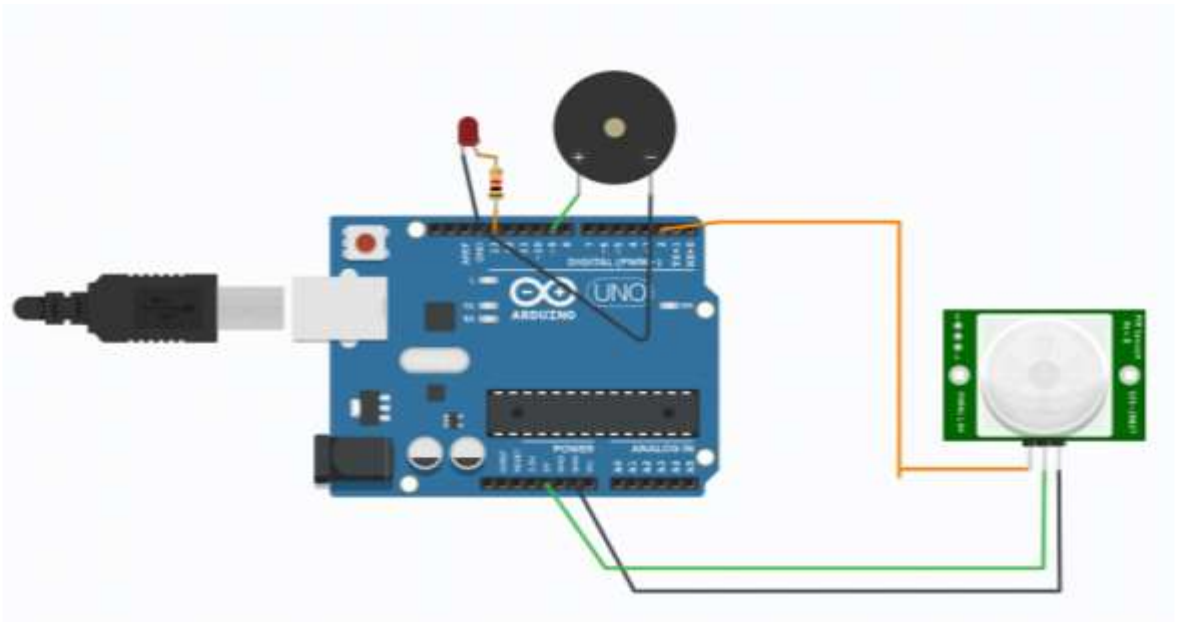
Result : The above experiment is executed successfully

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

Aim: To create a program that displays data from the sensor in regular intervals in a compact format.

Algorithm:

1. Set up pin modes for input (pin 2) and outputs (pins 13 and 9).
2. Enter an infinite loop.
3. Read the state of pin 2.
4. If the state is high, turn on the LED (pin 13) and play a tone (523 Hz) on pin 9.
5. If the state is not high, turn off the LED and stop the tone.
6. Delay for 1 millisecond.
7. Repeat from step 3.



```
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)
}
```

Result: To create a program that displays data from the sensor in regular intervals in a compact format is executed.

6. To perform simple addition using embedded components in Lab view software.

AIM: To build a VI that performs addition of two numbers and displays the result using LABVIEW.

ALGORITHM

1.Open labview software

2.Open blank VI.

3.Right click on the front panel window to open the control palette.

4.Insert the numeric controls as a input from the control palette.

5.Right click on the block diagram to open the function palette.

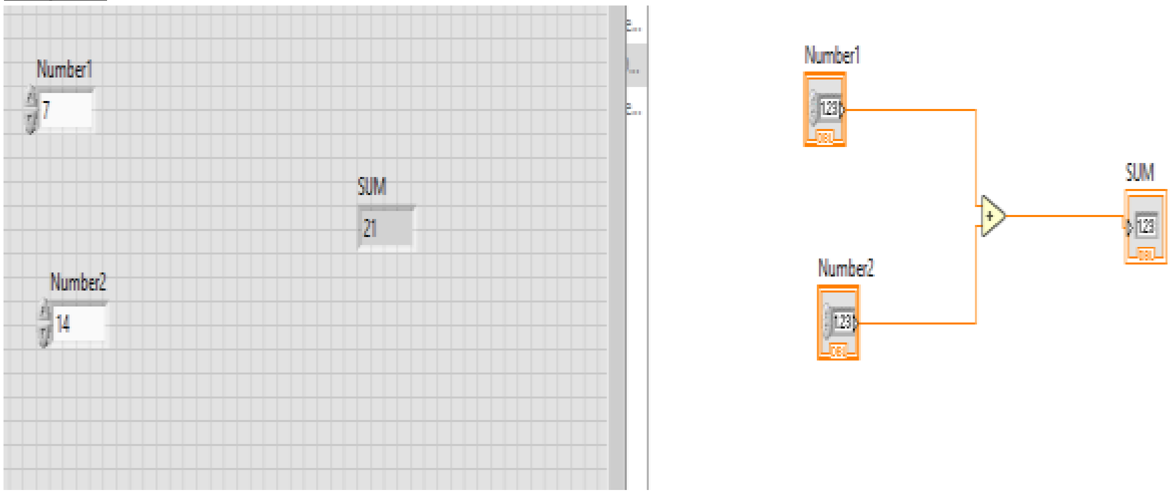
6.Add the Adder tool from the numeric sub palette from the function palette.

7.Insert the numeric indicator as an output from the control palette.

8.Connect all the terminals in the block diagram window.

9.Enter inputs in the front panel and click run to display the result.

Diagram



RESULT: Thus, a VI performs addition operation and displays the result using LABVIEW.

7. To perform string operations using embedded components in Lab view software.

a) STRING LENGTH

Aim: To build a VI that performs string length operation using embedded components and displays the result using LABVIEW.

Algorithm:

1.Create a new VI.

2.Drag and drop a string control and a numeric indicator onto the front panel.

3.Connect the string control to the input of the numeric indicator.

4.Right-click on the numeric indicator and select "Properties."

5.Set the display format of the numeric indicator to "Decimal."

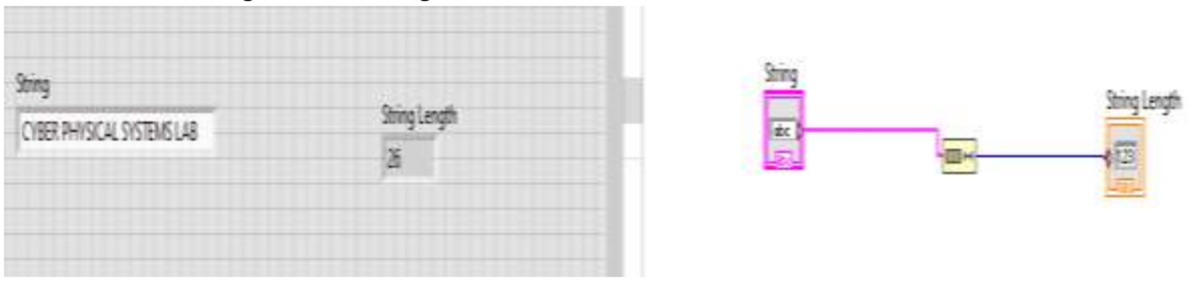
6.Add a String Length property node to the block diagram.

7.Connect the string control to the input of the String Length property node.

8.Connect the output of the String Length property node to the numeric indicator.

9.Run the VI and enter the input string in the string control.

10.Observe the length of the string on the numeric indicator.



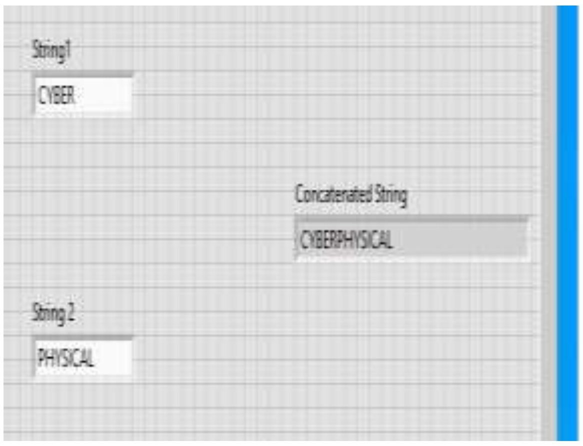
Result: Thus, a VI performs string length operation and displays the result using LABVIEW

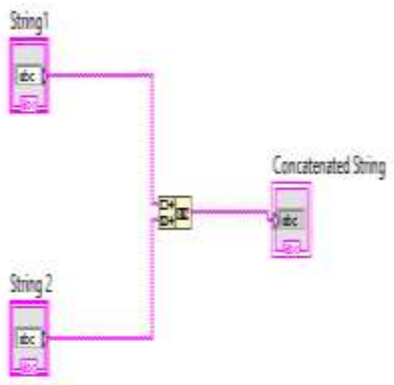
b) STRING CONCATENATION

Aim: To build a VI that performs string concatenation operation using embedded components and displays the result using LABVIEW.

Algorithm:

1.Create a new VI.
2.Drag and drop two string controls and a string indicator onto the front panel.
3.Connect the two string controls to the inputs of the string indicator.
4.Add a Concatenate Strings function or use the concatenation operator.
5.Connect the two string controls to the inputs of the Concatenate Strings function or use the concatenation operator.
6.Connect the output of the Concatenate Strings function or the concatenation operator to the string indicator.
7.Run the VI and enter the input strings in the string controls.
8.Observe the concatenated string on the string indicator.





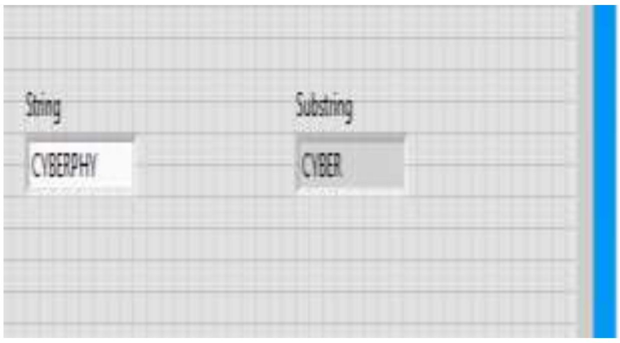
Result: Thus, a VI performs string concatenation operation and displays the result using LABVIEW

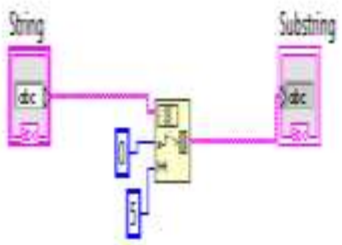
c) SUBSTRING

Aim: To build a VI that performs substring operation using embedded components and displays the result using LABVIEW.

Algorithm:

1.Create a new VI.
2.Drag and drop a string control, two numeric controls, and a string indicator onto the front panel.
3.Connect the string control, starting index control, and length control to the inputs of the string indicator.
4.Add a Substring function to the block diagram.
5.Connect the string control to the input of the Substring function.
6.Connect the starting index control to the "Start" input of the Substring function.
7.Connect the length control to the "Length" input of the Substring function.
8.Connect the output of the Substring function to the string indicator.
9.Run the VI and enter the input string, starting index, and length.
10.Observe the extracted substring on the string indicator.





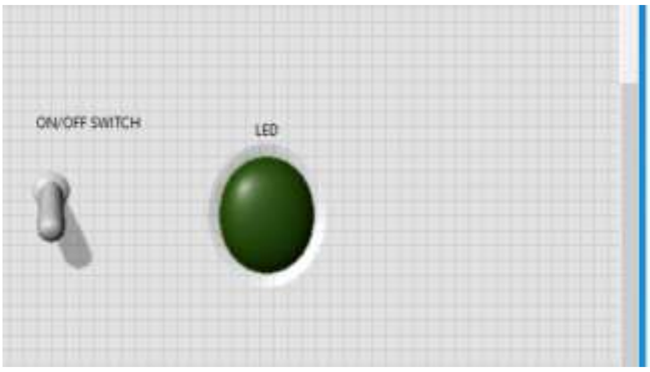
Result: Thus, a VI performs substring operation and displays the result using LABVIEW

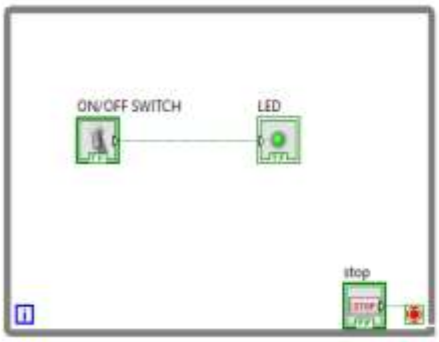
8. To perform LED ON/OFF switch using Lab view software.

Aim: To build a VI that performs led on/off switch operation using embedded components and displays the result using LABVIEW.

Algorithm:

1.Connect the LED to the digital output pin of the microcontroller.
2.Open LabVIEW software and create a new VI (Virtual Instrument).
3.Add a digital output module to the block diagram.
4.configure it to control the digital output pin connected to the LED.
5.Add a Boolean switch to the front panel of the VI to control the LED ON/OFF state.
6.Connect the Boolean switch to the digital output module on the block diagram.
7.Run the VI and toggle the Boolean switch to turn the LED ON/OFF.
8.Observe the LED ON/OFF switch operation on the indicator.





Result: Thus, a VI performs led on/off switch operation and displays the result using LABVIEW

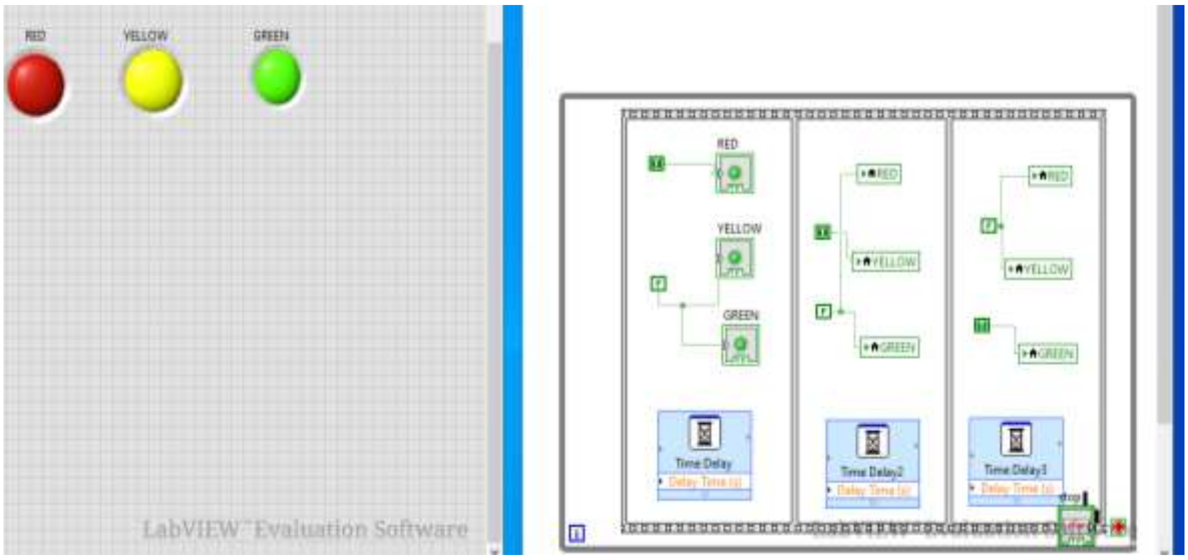
9. To Design Traffic Signal Light using embedded components using Lab view software.

Aim: To build a VI that performs Traffic signal light operation using embedded components and displays the result using LABVIEW.

Algorithm:-

1. Initialize the System:
   *set up the LabVIEW environment and ensure all necessary components (microcontroller, LED, etc.) are connected and functioning correctly.
2. Define Signal Timing:
   *Decide on the timing intervals for each phase of the traffic signal (eg: green, yellow red times for each direction).
3. Initialize Signal states:
   * Set the initial state of the traffic signal to a predefined starting configuration (eg: all red or a specific direction green).
4. Loop for Traffic signal control:
   *Enter a continuous loop that will handle the traffic signal control.
5. Check Input:

*Monitor the sensors or inputs to detect vehicle presence or traffic demands for different directions.
6. Implement Traffic signal logic.
*Based on the Input from the previous step, implement the traffic signal Logic to determine the appropriate signal states for each direction.
*Typical logic involves transitioning from green to yellow before changing to red and vice versa.
7. Update signal outputs:
8. wait for signed change
9. Repeat the loop.
10. End the program.



Result: To Design Traffic Signal Light using embedded components using Lab view software is implemented successfully.