Project Report

for the course -

Internet of Things

by YASH ARORA

Roll No.: 03670211217

Batch : MET - 2017

(Delhi Institute of Tool Engineering)

under the guidance of

R. K. DHAMMI (HOD) (DITE)

Prof. Ian Harris

(Coursera – University of California, Irvine)

Acknowledgement

I would like to express my special thanks of gratitude to HoD Mr. Arun Gupta as well as our Director Dr. Sagar K. Maji who gave me the golden opportunity to do this wonderful project on the topic (Write the topic name), which also helped me in doing a lot of Research and I came to know about so many new things I am really thankful to them.

Secondly, I would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

Delhi Institute of Tool Engineering

DITE offers Graduate and Post Graduate courses with specialization in Tool Engineering and is perhaps the first institute in North India to offer these courses. It had started the BTech course in Tool Engineering in the year 2008 at DITE Campus-II with a total intake of 60. It is a four-year degree course approved by the AICTE and affiliated to Guru Gobind Singh Indraprastha University (GGSIPU), Delhi. Admission in this course is done through an entrance examination conducted by the GGSIPU Students. DITE has well trained experienced full time/ visiting faculties having M.tech and PhD qualifications from reputed institutions i.e. IIT, NSIT, DCE. The department possesses modern laboratories equipped with latest experimental set- ups and research facilities for instrumentation, the strength of materials, fluid mechanics and others supported by Software's like Pro E, Catia, CAD/CAM and other High-end software viz., Unigraphics, Abacus (CAE), Hyper form, Moldflow for use and training of BTech and MTech students. The Institute has basic & high-end software labs, CNC Classroom, Engineering Graphics Lab, well equipped Technological, PLC, and CNC M/c, Industrial El Electronics, Digital Electronics, Motor Drives & Control, Pneumatic & Hydraulics Labs and Workshops

Introduction to IoT

IoT comprises things that have unique identities and are connected to internet. By 2020 there will be a total of 50 billion devices /things connected to internet. IoT is not limited to just connecting things to the internet but also allow things to communicate and exchange data.

Definition

A dynamic global n/w infrastructure with self configuring capabilities based on standard and interoperable communication protocols where physical and virtual —thingsǁ have identities, physical attributes and virtual personalities and use intelligent interfaces, and are seamlessly integrated into information n/w, often communicate data associated with users and their environments.

Characteristics:

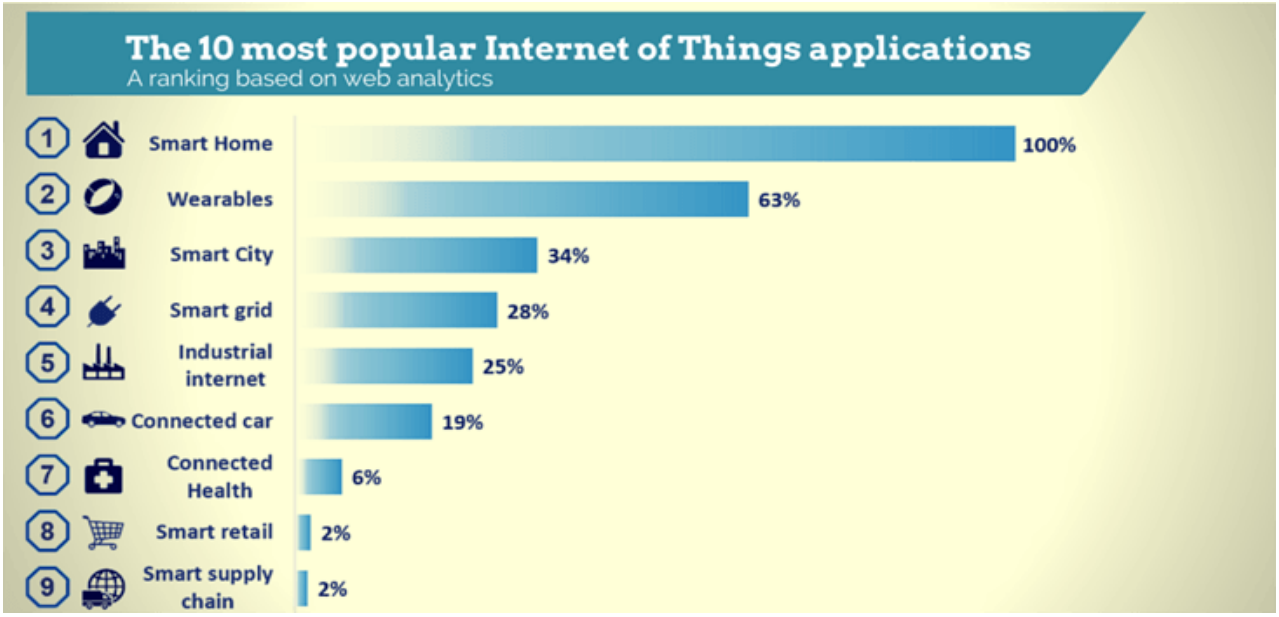
1) **Dynamic & Self Adapting:** IoT devices and systems may have the capability to dynamically adapt with the changing contexts and take actions based on their operating conditions, user‘s context or sensed environment.

   Eg: the surveillance system is adapting itself based on context and changing conditions.

2) Self Configuring: allowing a large number of devices to work together to provide certain functionality.
3) Inter Operable Communication Protocols: support a number of interoperable communication protocols ans can communicate with other devices and also with infrastructure.
4) Unique Identity: Each IoT device has a unique identity and a unique identifier(IP address).
5) Integrated into Information Network: that allow them to communicate and exchange data with other devices and systems.

Applications of IoT:

1) Home
2) Cities
3) Environment
4) Energy
5) Retail
6) Logistics
7) Agriculture
8) Industry
9) Health & LifeStyle Physical Design



Physical Design of IoT:

1) Things in IoT: The things in IoT refers to IoT devices which have unique identities and perform remote sensing, actuating and monitoring capabilities. IoT devices can exchange data with other connected devices applications. It collects data from other devices and process data either locally or remotely. An IoT device may consist of several interfaces for communication to other devices both wired and wireless. These includes (i) I/O interfaces for sensors, (ii) Interfaces for internet connectivity (iii) memory and storage interfaces and (iv) audio/videointerfaces.

2) IoT Protocols:

a) Link Layer: Protocols determine how data is physically sent over the network‘s physical layer or medium. Local network connect to which host is attached. Hosts on the same link exchange data packets over the link layer using link layer protocols. Link layer determines how packets are coded and signaled by the h/w device over the medium to which the host is attached.

Protocols:

- 802.3-Ethernet: IEEE802.3 is collection of wired Ethernet standards for the link layer.
  Eg: 802.3 uses co-axial cable; 802.3i uses copper twisted pair connection; 802.3j uses fiber optic connection; 802.3ae uses Ethernet over fiber.
- 802.11-WiFi: IEEE802.11 is a collection of wireless LAN (WLAN) communication standards including extensive description of link layer.
  Eg: 802.11a operates in 5GHz band, 802.11b and 802.11g operates in 2.4GHz band, 802.11n operates in 2.4/5GHz band, 802.11ac operates in 5GHz band, 802.11ad operates in 60Ghzband.
- 802.16 - WiMax: IEEE802.16 is a collection of wireless broadband standards including exclusive description of link layer. WiMax provide data rates from 1.5 Mb/s to 1Gb/s.
- 802.15.4-LR-WPAN: IEEE802.15.4 is a collection of standards for low rate wireless personal area network (LR-WPAN). Basis for high level communication protocols such as ZigBee. Provides data rate from 40kb/s to250kb/s.
- 2G/3G/4G-Mobile Communication: Data rates from 9.6kb/s (2G) to up to100Mb/s(4G).

b) Network/Internet Layer: Responsible for sending IP datagrams from source n/w to destination n/w. Performs the host addressing and packet routing. Datagrams contains source and destination address**.**

Protocols:

- IPv4: Internet Protocol version4 is used to identify the devices on a n/w using a hierarchical addressing scheme. 32 bit address. Allows total of 2**32addresses.
- IPv6: Internet Protocol version6 uses 128 bit address scheme and allows 2**128 addresses.
- 6LOWPAN:(IPv6overLowpowerWirelessPersonalAreaNetwork) operates in 2.4 GHz frequency range and data transfer 250 kb/s.

c) Transport Layer: Provides end-to-end message transfer capability independent of the underlying n/w. Set up on connection with ACK as in TCP and without ACK as in UDP. Provides functions such as error control, segmentation, flow control and congestion control.

Protocols:

- TCP: Transmission Control Protocol used by web browsers (along with HTTP and HTTPS), email(along with SMTP, FTP). Connection oriented and stateless protocol. IP Protocol deals with sending packets, TCP ensures reliable transmission of protocols in order. Avoids n/w congestion and congestion collapse.
- UDP: User Datagram Protocol is connectionless protocol. Useful in time sensitive applications, very small data units to exchange. Transaction oriented and stateless protocol. Does not provide guaranteed delivery.

d) Application Layer: Defines how the applications interface with lower layer protocols to send data over the n/w. Enables process-to-process communication using ports.

Protocols:

- **HTTP:** Hyper Text Transfer Protocol that forms foundation of WWW. Follow request response model Stateless protocol.
- **CoAP:** Constrained Application Protocol for machine-to-machine(M2M) applications with constrained devices, constrained environment and constrained n/w. Uses client server architecture.
- **WebSocket:** allows full duplex communication over a single socket connection.
- **MQTT:** Message Queue Telemetry Transport is light weight messaging protocol based on publish-subscribe model. Uses client server architecture. Well suited for constrained environment.
- **XMPP:** Extensible Message and Presence Protocol for real time communication and streaming XML data between network entities. Support client-server and server-server communication.
- **DDS:** Data Distribution Service is data centric middleware standards for device-to-device or machine-to-machine communication. Uses publish-subscribe model.
- **AMQP:** Advanced Message Queuing Protocol is open application layer protocol for business messaging. Supports both point-to-point and publish-subscribe model.

Project on Automated Belt Conveyer

Theory :

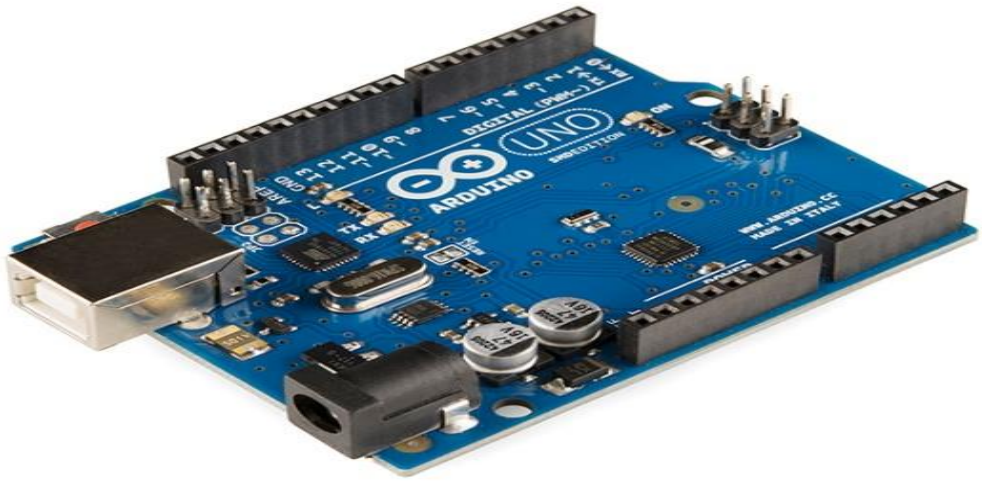
The project consists of a belt conveyor, for filling of bottles to a required level of liquid (height), a few sensors to take input, a controller to control actuations and some actuators to fulfill the overall task. The inputs to the system will be a pre-set (controllable) liquid level required. The sensors will detect the presence of a bottle under filling station and the instantaneous amount of liquid level reached while filling. The controller controls the conveyor movement and the actuator to start and stop filling. This project can be used as a simulator for automatic filling stations in beverages, pharmaceutical and dairy process industries. The objective of this project is to create an automated filling station for cans / bottles, using detection technique for bottles and controlled level filling.

Components Required (Fully Automated and AI based without Button to start stop water filling)

DC Motor
Solenoid Valve
Sonar Sensor
IR Transmitter + Receiver
LCD Display
Atmel ATmega 328
Relay
Acrelic Box
Other Misc components (Wire, Glue, Soldering etc.)

Arduino UNO :

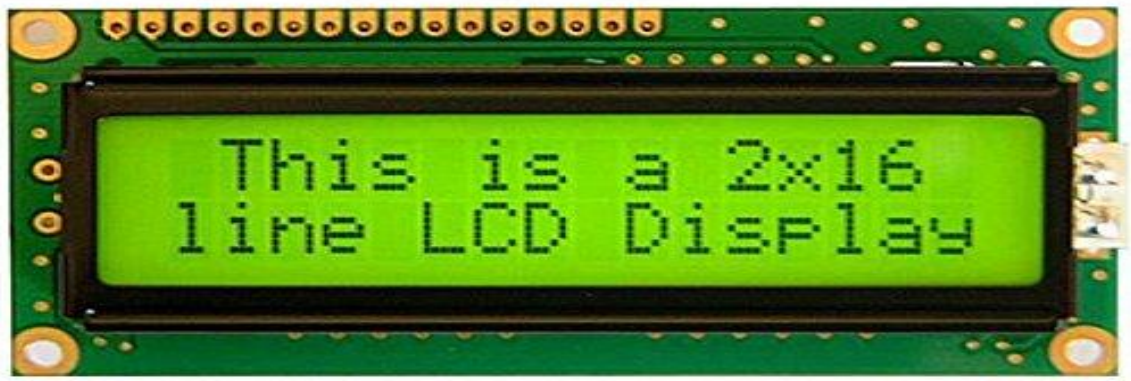
Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.



The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

LCD Display

A liquid-crystal display (LCD) is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers. Liquid crystals do not emit light directly, instead using a backlight or reflector to produce images in color or monochrome. LCDs are available to display arbitrary images or fixed images with low information content, which can be displayed or hidden, such as preset words, digits, and seven-segment displays, as in a digital clock.



An LCD (Liquid Crystal Display) screen is an electronic display module and has a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines.

Relay

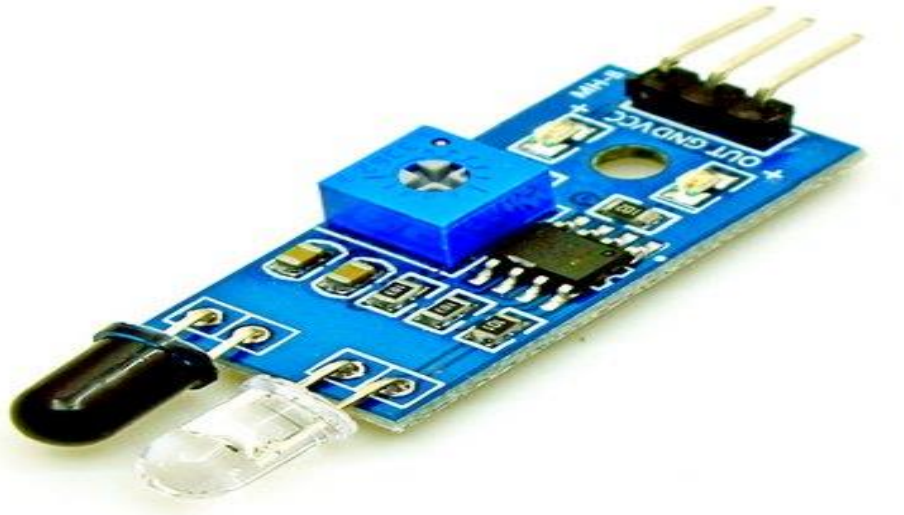
A relay is an electrically operated switch. It consists of a set of input terminals for a single or multiple control signals, and a set of operating contact terminals. The switch may have any number of contacts in multiple contact forms, such as make contacts, break contacts, or combinations thereof.

A relay is a programmable electrical switch, which can be controlled by Arduino or any micro-controller. It is used to programmatically control on/off the devices, which use the high voltage and/or high current.

Circuits that operate at high voltages or at high currents cannot be controlled directly by an Arduino. Instead, you use a low-voltage control signal from the Arduino to control a relay, which is capable of handling and switching high-voltage or high-power circuits. A relay consists of an electromagnet that, when energized, causes a switch to close or open. Relays provide complete electrical isolation between the control circuit and the circuit being controlled.

IR Sensor

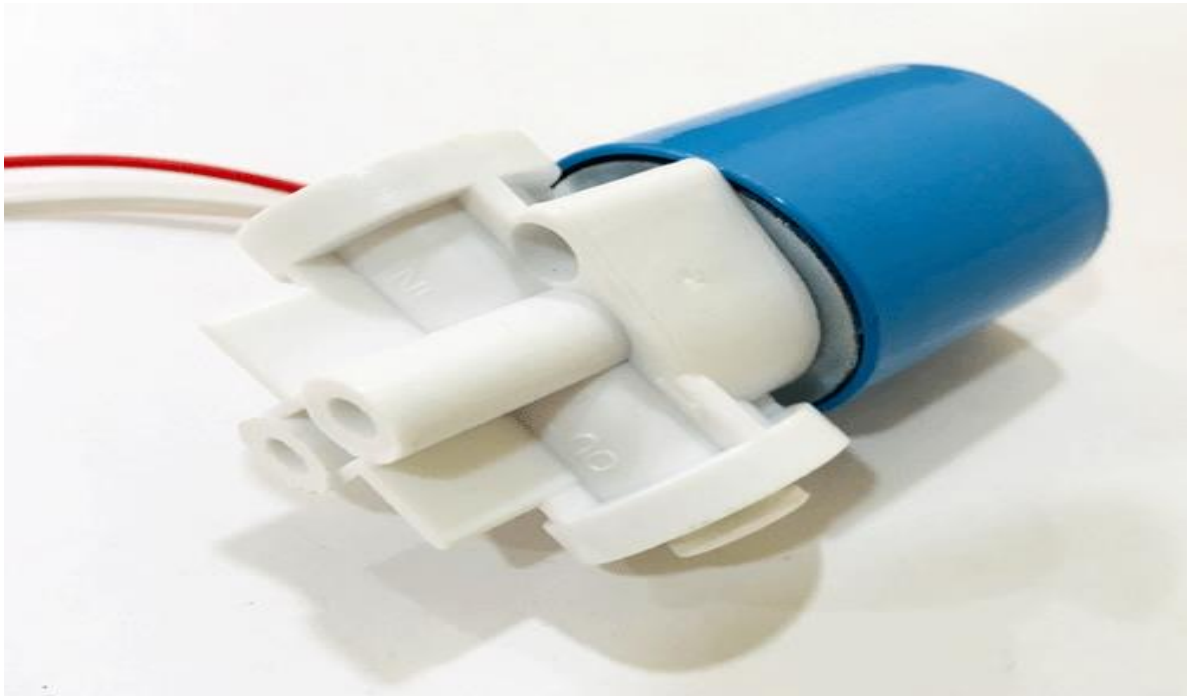
IR sensor is an electronic device, that emits the light in order to sense some object of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation. These types of radiations are invisible to our eyes, but infrared sensor can detect these radiations.



Setting up IR sensor connection to Arduino is very simple. Beside VCC and GND pin, the sensor has only one output pin that should be connected to one of digital pins of the Arduino.

Solenoid Valve

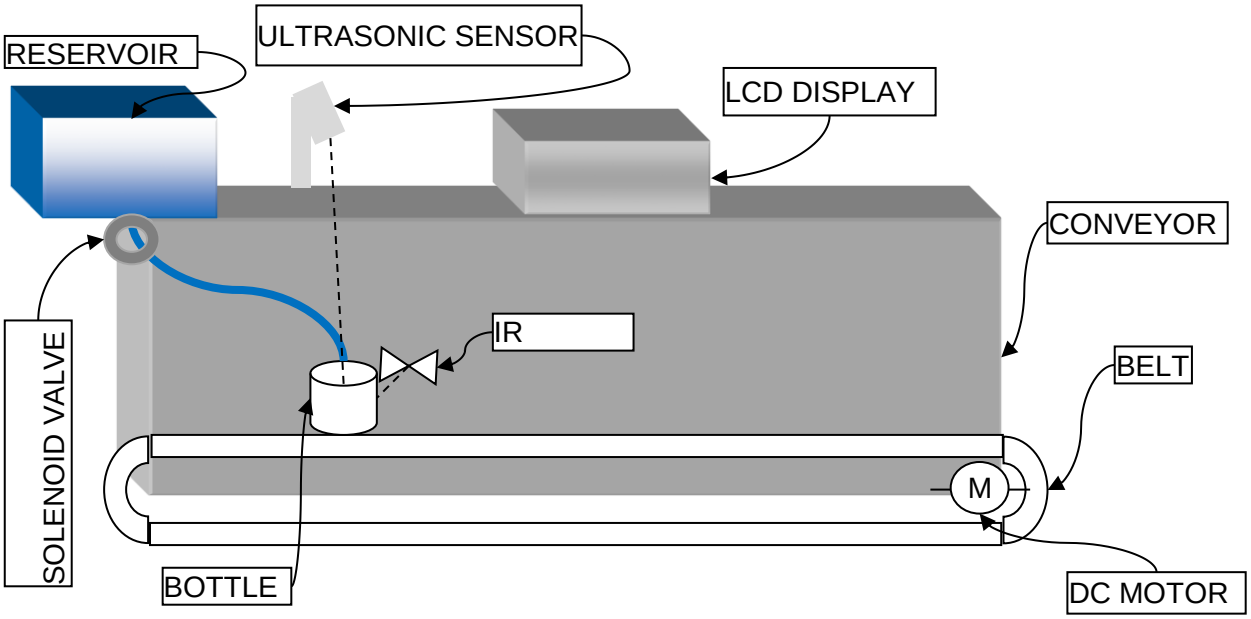
A Solenoid is a coil that when energised, produces a controlled magnetic field down through its centre. By placing a magnetic armature inside that field, the armature can move in or out of the coil. Team this with our Arduino and we open up a number of interesting applications.



Electric solenoids work on similar electromagnetic principles to those of DC motors, however, solenoids can use the magnetic energy to push or pull something rather than turn it. Solenoids are found in paintball guns, pinball machines, printers, valves and even automobiles.

Setting up Solenoid Valve connection to Arduino is very simple. Beside VCC and GND pin, the Valve has only one output pin that should be connected to one of digital pins of the Arduino.

Basic Design of the Project:



Source Code of the Project:

```
#include <LiquidCrystal.h>
#include <iostream>

/*Created by Abhishek Tyagi and Yash Arora, Department of Mechatronics on 30 Novembver 2020*/


int solenoid1 = 3;
int solenoid2 = 4;
int solenoid3 = 5;
int solenoid4 = 6;
int solenoid5 = 7;

// initialize the library with the numbers of the interface pins
// LCD Display usage and the example to write the characters in this GitHub Link : https://github.com/adafruit/STEMMA_LiquidCrystal/tree/master/examples to understand in detai
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

#define echoPin 7 // Echo Pin Ultrasonic // to understand Ultrasonic https://create.arduino.cc/projecthub/abdularbi17/ultrasonic-sensor-hc-sr04-with-arduino-tutorial-327ff6
#define trigPin 8 // Trigger Pin Ultrasonics
#define relay 10 // Onboard LED
#define Glass 9 // Bottle detection
// Conveyor DC Motor use and understanding the circuit diagram https://www.tutorialspoint.com/arduino/arduino_dc_motor.htm#:~:text=Following%20is%20the%20schematic%20diagram%2
byte armsUp[8] = {
  0b00100,
  0b01010,
  0b00100,
  0b10101,
  0b01110,
  0b00100,
  0b00100,
   0b00100,
   0b01010
}; // make some custom characters: on LCD Display these are ASCII and Binary value for characterisation link to understand this : https://github.com/adafruit/STEMMA_LiquidCrys

int maximumRange = 200; // Maximum range needed depends on the motor used it can be 200+ and more
int minimumRange = 0; // Minimum range needed and change as suitable for conveyor for industry usage
long duration, distance; // Duration used to calculate distance to understand this


void setup() {
 lcd.createChar(4, armsUp);
  lcd.begin(16, 2);
  lcd.write(4);
   lcd.print(" Production Line Automation ");
    for (int positionCounter = 15; positionCounter < 84; positionCounter++) {
    // scroll one position left:
    lcd.scrollDisplayLeft();
    // wait a bit: and change time if bottle is not hold at right position
    delay(400);
  }
  delay(1000);
  lcd.clear();
   delay(2000);
 pinMode(trigPin, OUTPUT);
 pinMode(solenoid1, OUTPUT);
 pinMode(solenoid2, OUTPUT);
 pinMode(solenoid3, OUTPUT);
 pinMode(solenoid4, OUTPUT);
 pinMode(solenoid5, OUTPUT);

 pinMode(Glass, INPUT);
```


Activate Windows
Go to Settings to activate Windows.

```
  pinMode(Glass, INPUT);
  pinMode(echoPin, INPUT);
  pinMode(relay, OUTPUT); // Use indicator like led or buzzer whatever suitable for you

  int IRSensor1 = 2;// connect ir sensor to arduino pin 2
  int LED = 13; // conect Led to arduino pin 13

  int solenoid_A;
  int solenoid_B;
  int solenoid_C;
  int solenoid_D;
  int solenoid_E;

  int sensorStatus;

  cout<<"Enter 1 for yes AND 0 for no.";

  cout<<"Do you want to run Solenoid A : ";
  cin>>solenoid_A;

  cout<<"Do you want to run Solenoid B : ";
  cin>>solenoid_B;

  cout<<"Do you want to run Solenoid C : ";
  cin>>solenoid_C;

  cout<<"Do you want to run Solenoid D : ";
  cin>>solenoid_D;

  cout<<"Do you want to run Solenoid E : ";
  cin>>solenoid_E;
```



```
  cin>>solenoid_E;

    pinMode (IRSensor1, INPUT); // sensor pin INPUT
    pinMode (LED, OUTPUT); // Led pin OUTPUT
 }


 void loop(){

  digitalWrite(trigPin, LOW); // Change delay while debugging
  delayMicroseconds(2);

  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);

  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);

  //Calculate the distance (in cm) based on the speed of sound. Ultrasonic sound to distance conversion formula HC-SR04 Ultrasonic to stop the conveyor
  distance = duration/58.2;

  int Detect_glass =digitalRead(IRSensor1);

 if(Solenoid5 > Solenoid4){

 digitalWrite(IRsensor5, HIGH);

 digitalWrite(IRsensor1, LOW);
 digitalWrite(IRsensor3, LOW);
 digitalWrite(IRsensor4, LOW);
 digitalWrite(IRsensor2, LOW);
```

```
digitalWrite(IRsensor2, LOW);
}


else if(Solenoid4 > Solenoid3){

digitalWrite(IRsensor4, HIGH);

digitalWrite(IRsensor1, LOW);
digitalWrite(IRsensor3, LOW);
digitalWrite(IRsensor2, LOW);
digitalWrite(IRsensor5, LOW);
}


else if(Solenoid3 > Solenoid2){

digitalWrite(IRsensor3, HIGH);

digitalWrite(IRsensor1, LOW);
digitalWrite(IRsensor2, LOW);
digitalWrite(IRsensor4, LOW);
digitalWrite(IRsensor5, LOW);
}


else if(Solenoid2 > Solenoid1){

digitalWrite(IRsensor2, HIGH);

digitalWrite(IRsensor1, LOW);
```



```
 digitalWrite(IRsensor1, LOW);
 digitalWrite(IRsensor3, LOW);
 digitalWrite(IRsensor4, LOW);
 digitalWrite(IRsensor5, LOW);
 }


 else {

 digitalWrite(IRsensor1, HIGH);

 digitalWrite(IRsensor2, LOW);
 digitalWrite(IRsensor3, LOW);
 digitalWrite(IRsensor4, LOW);
 digitalWrite(IRsensor5, LOW);
 }

 if(Detect_glass == HIGH){
   lcd.setCursor(0, 0); // Understand solenoid valve usage here: https://bc-robotics.com/tutorials/controlling-a-solenoid-valve-with-arduino/ and https://create.arduino.cc/proj
     lcd.print("Conveyor  Stop");
     digitalWrite(relay, LOW);   //conveyor off
     delay(1000);
     //sonar value for level detector change according to your suitability
     if(distance<=9)
     {
digitalWrite(solenoid, LOW);  //water off because solenoid put the valve down/close
 delay(1000);
 digitalWrite(relay, HIGH); //conveyor on
 delay(3000); // change delay as convinient to you also debug every single step while pouring the fluid
    }

```

```
179
180     if(solenoid_A == 1){
181     digitalWrite(solenoid1, HIGH);
182     digitalWrite(solenoid2, LOW);
183     digitalWrite(solenoid3, LOW);
184     digitalWrite(solenoid4, LOW);
185     digitalWrite(solenoid5, LOW);
186
187     }
188
189     if(solenoid_B == 1){
190     digitalWrite(solenoid1, LOW);
191     digitalWrite(solenoid2, HIGH);
192     digitalWrite(solenoid3, LOW);
193     digitalWrite(solenoid4, LOW);
194     digitalWrite(solenoid5, LOW);
195
196     }
197     if(solenoid_C == 1){
198     digitalWrite(solenoid1, LOW);
199     digitalWrite(solenoid2, LOW);
200     digitalWrite(solenoid3, HIGH);
201     digitalWrite(solenoid4, LOW);
202     digitalWrite(solenoid5, LOW);
203
204     }
205     if(solenoid_D == 1){
206     digitalWrite(solenoid1, LOW);
207     digitalWrite(solenoid2, LOW);
208     digitalWrite(solenoid3, LOW);
209     digitalWrite(solenoid4, HIGH);
```



```
208     digitalWrite(solenoid3, LOW);
209     digitalWrite(solenoid4, HIGH);
210     digitalWrite(solenoid5, LOW);
211
212     }
213     if(solenoid_E == 1){
214     digitalWrite(solenoid1, LOW);
215     digitalWrite(solenoid2, LOW);
216     digitalWrite(solenoid3, LOW);
217     digitalWrite(solenoid4, LOW);
218     digitalWrite(solenoid5, HIGH);
219
220     }
221   }
222
223    delay(2000);
224
225     digitalWrite(solenoid1, LOW);
226     digitalWrite(solenoid2, LOW);
227     digitalWrite(solenoid3, LOW);
228     digitalWrite(solenoid4, LOW);
229     digitalWrite(solenoid5, LOW);
230
231
232
233
234   }
```

Conclusion :

The project consists of a belt conveyor, for filling of bottles to a required level of liquid (height), a few sensors to take input, a controller to control actuations and some actuators to fulfill the overall task. The inputs to the system will be a pre-set (controllable) liquid level required. The sensors will detect the presence of a bottle under filling station and the instantaneous amount of liquid level reached while filling. The controller controls the conveyor movement and the actuator to start and stop filling. This project can be used as a simulator for automatic filling stations in beverages, pharmaceutical and dairy process industries. The objective of this project is to create an automated filling station for cans / bottles, using detection technique for bottles and controlled level filling.

Certificate :