IOT Based Automatic Bottle Filling and Capping System Using Arduino

Project Report

Submitted by

Bobin B Paul (1117211004)

Fayaz M (1117211005)

Natana Guru M (1116211901)

Under the guidance of

Dr. V. Amsaveni

in partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND INSTRUMENTATION ENGINEERING

NOORUL ISLAM CENTRE FOR HIGHER EDUCATION

(DECLARED AS DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT. 1956)
ACCREDIATED BY NAAC WITH 'A' GRADE
KUMARACOIL, THUCKALAY, KANYAKUMARI DISTRICT,
TAMILNADU, PIN: 629180.

JUNE -2021

NOORUL ISLAM CENTRE FOR HIGHER EDUCATION
(DECLARED AS DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT. 1956)
ACCREDIATED BY NAAC WITH 'A' GRADE
KUMARACOIL, THUCKALAY, KANYAKUMARI DISTRICT,
TAMILNADU, PIN: 629180.

Department of Electronics and Instrumentation Engineering

BONAFIDE CERTIFICATE

PROJECT WORK

This is to certify that the project report entitled "**IOT Based Automatic Bottle Filling and Capping System Using Arduino**" submitted by **Bobin B Paul (1117211004)** towards the partial fulfilment of the requirements for the award of the degree in Bachelor of Engineering in Electronics and Instrumentation Engineering of Noorul Islam Centre for Higher Education is the record of work carried out by him under my supervision and guidance. The work submitted has in our opinion reached a level required for being accepted for examination.

[illegible]

Head of the Department

Dr. S. S. Kumar
Associate Professor,
Department of Electronics and
Instrumentation Engineering,
Noorul Islam Centre for
Higher Education,
Kumaracoil – 629180.

[illegible]

Project Guide

Dr. V. Amsaveni
Associate Professor,
Department of Electronics and
Instrumentation Engineering,
Noorul Islam Centre for
Higher Education,
Kumaracoil – 629180.

Submitted for the project Viva-Voce Examination held on 09/06/2021

Internal Examiner

External Examiner

NOORUL ISLAM CENTRE FOR HIGHER EDUCATION
(DECLARED AS DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT. 1956)
ACCREDIATED BY NAAC WITH 'A' GRADE
KUMARACOIL, THUCKALAY, KANYAKUMARI DISTRICT,
TAMILNADU, PIN: 629180.

DECLARATION

We hereby declare that the project work entitled as "**IOT Based Automatic Bottle Filling and Capping System Using Arduino**" is an authentic recorded of the original work carried out by the Department of Electronics and Instrumentation Engineering at Noorul Islam Centre for Higher Education as required for the project for the award of degree of B.E Electronics and Instrumentation Engineering under the guidance of **Dr. V. Amsaveni.**

Candidate Name with Register No.	Signature of the Candidate
Mr. Bobin B Paul (1117211004)	[illegible]

I certify that the declaration made above by the candidates is true.

[illegible]

Dr. V. Amsaveni,
Associate Professor,
Department of Electronics and Instrumentation Engineering,
Noorul Islam Centre for Higher Education,
Kumaracoil – 629180.

ACKNOWLEDGEMENT

First and foremost, I acknowledge the abiding presence and the abounding grace of OUR ALMIGHTY GOD for his unseen hand yet tangible guidance all through the formation of this project.

I take this opportunity to place a record to my heartiest gratitude and sincere appreciation of the tremendous support given to me, by each and everyone associated with this project.

I extend my heartfelt, sincere thanks to the Chancellor **Dr. A. P. Majeed Khan**, Pro Chancellor **Mr.M. S. Faizal Khan**, Vice Chancellor **Dr. A.K. Kumaraguru**, Registrar **Dr. P. Thirumalvalavan**, and Pro-Vice chancellor **Dr. N. Chandrasekhar**, who allowed me to do my project in the campus.

I highly thanks to **Dr. S.S Kumar**, Head of the Department, Electronics and Instrumentation Engineering, for providing necessary facilities for the successful completion of this project work.

I sincerely thank to Dr**. V. Amsaveni**, Associate professor, Department of Electronics and Instrumentation Engineering for her guidance and for providing necessary facilities and encouragements for the successful completion of this project work.

We thank all Assistant professors, Non-teaching staffs of our department, and our friends who gave encouraged us to complete the project.

ABSTRACT

IOT Based Automatic Bottle Filling and Capping System Using Arduino Project is to provide easy access to the company for Filling Bottles Automatically. This type of project is mainly used in the soft drinks Manufacturing Company and Medicine Manufacturing Company in which drinks and Syrup are Automatically Filled in the Bottle. This project is designed with the help of micro controller, IR Transmitter and Receiver, Relay Driver, DC Motor and Mechanical arrangements. Arduino are flash type reprogrammable Microcontroller. Infrared Transmitter is the one type of LED generally called IR. IR Receiver is used to receive such type of Infrared rays transmitted by the IR Transmitter. Mechanical arrangement is Straight line conveyor in which Bottles are placed with particular distance, the Mechanical arrangement is designed in this manner. When power is ON the IR Transmitter passing ray to the receiver and giving the signal to the Microcontroller through signal conditioning unit. Initially the Micro controller activates the DC Motor through the Driver circuit which is connected to conveyor. So, the Bottles are moving when the Bottles comes in between the IR Sensor it blocks the rays. Now the Arduino stop the DC motor so movement of conveyor will be stopped. The bottle position is straight line to filling system. Arduino activates the pump for filling system. . After setting time limit for the quantity motor is switched OFF for filling system and motor is switched ON for moving tray system. For capping same as filling bottle, bottle is detect by IR and using Servo motor and DC motor to cap the bottle. Likewise, the Bottles are filled and capped with materials automatically. This project is used to reduce the Manpower as well as time consumption. Using IOT the process can be control and monitor wirelessly and remotely.

TABLE OF CONTENT

LIST OF FIGURES

Chapter 1
Introduction of Automatic Bottle Filling

1.1 Introduction

Automatic Bottle Filling System project is to provide easy access to the company for Filling Bottles Automatically. This type of project is mainly used in the soft drinks Manufacturing Company and Medicine Manufacturing Company in which drinks and Syrup are Automatically Filled in the Bottle. This project is designed with the help of micro controller, IR Transmitter and Receiver, Relay Driver, DC Motor and Mechanical arrangements. Arduino are flash type reprogrammable Microcontroller. Infrared Transmitter is the one type of LED generally called IR. When the supply is given to this LED it generates and Transmitting the infrared rays. IR Receiver is used to receive such type of Infrared rays transmitted by the IR Transmitter. Mechanical arrangement is Straight line conveyor in which Bottles are placed with particular distance, the Mechanical arrangement is designed in this manner. When power is ON the IR Transmitter passing ray to the receiver and giving the signal to the Microcontroller through signal conditioning unit. Initially the Micro controller activates the DC Motor through the Driver circuit which is connected to conveyor. So, the Bottles are moving when the Bottles comes in between the IR Sensor it blocks the rays. Now the Arduino stop the DC motor so movement of conveyor will be stopped. The bottle position is straight line to filling system. Arduino activates the pump for filling system. . After setting time limit for the quantity motor is switched OFF for filling system and motor is switched ON for moving tray system. Likewise, the Bottles are filled with materials automatically. This project is used to reduce the Manpower as well as time consumption.

1.2 Working mechanism of plant

Following figure shows the general mechanism of automatic bottle refilling plant which can be controlled by any type of Microcontroller. Which is able to control the whole process and with the help of software we can access the whole plant far from the venue.

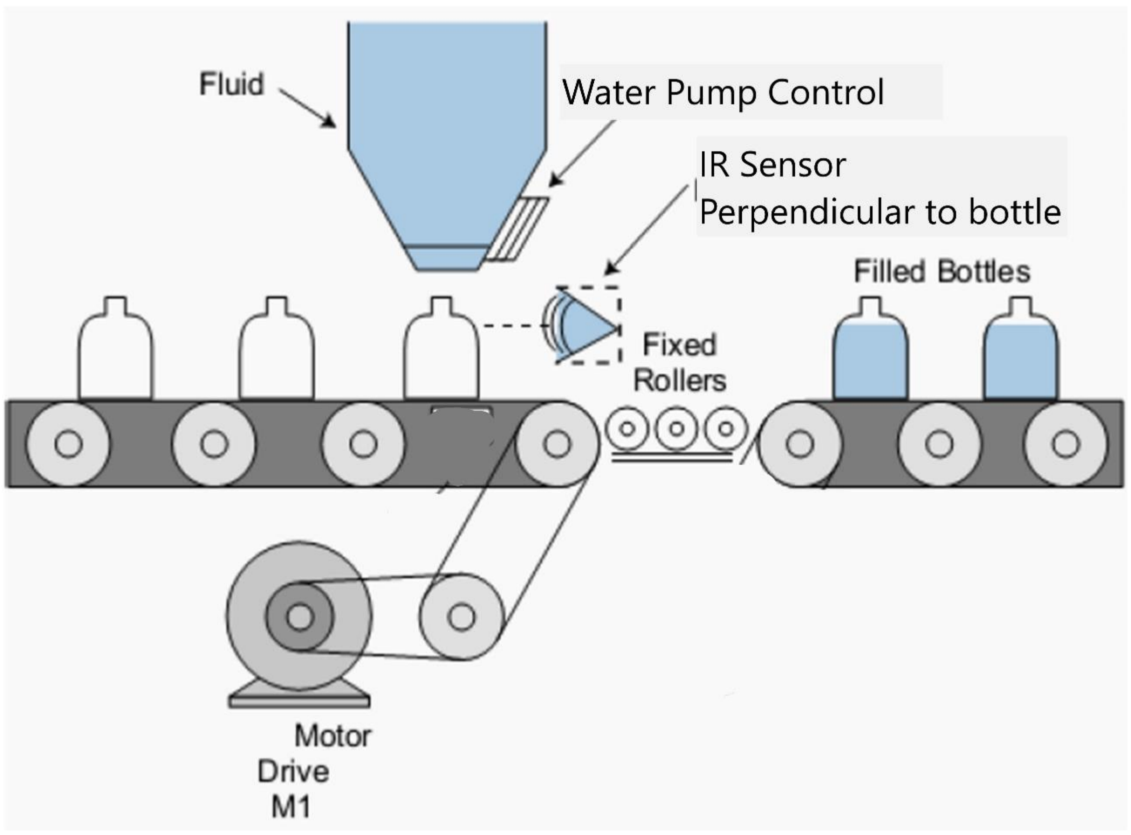


Fig 1.1 working mechanism of automatic bottle filling system

Types of controller which can be used in system

Generally in the industrial PLC (Programmable logic control) is in practise and some if the updated software like SCADA (Supervisory Control and Data Acquisition) is used to control the whole plant but we can also use some another type of microcontroller to do the same task at different condition which are following.

- PLC (Programmable Logic Controller)
- Microcontroller 8051
- Arduino UNO Board
- Other Microcontroller Board

1.3 PLC (Programmable Logic Controller)

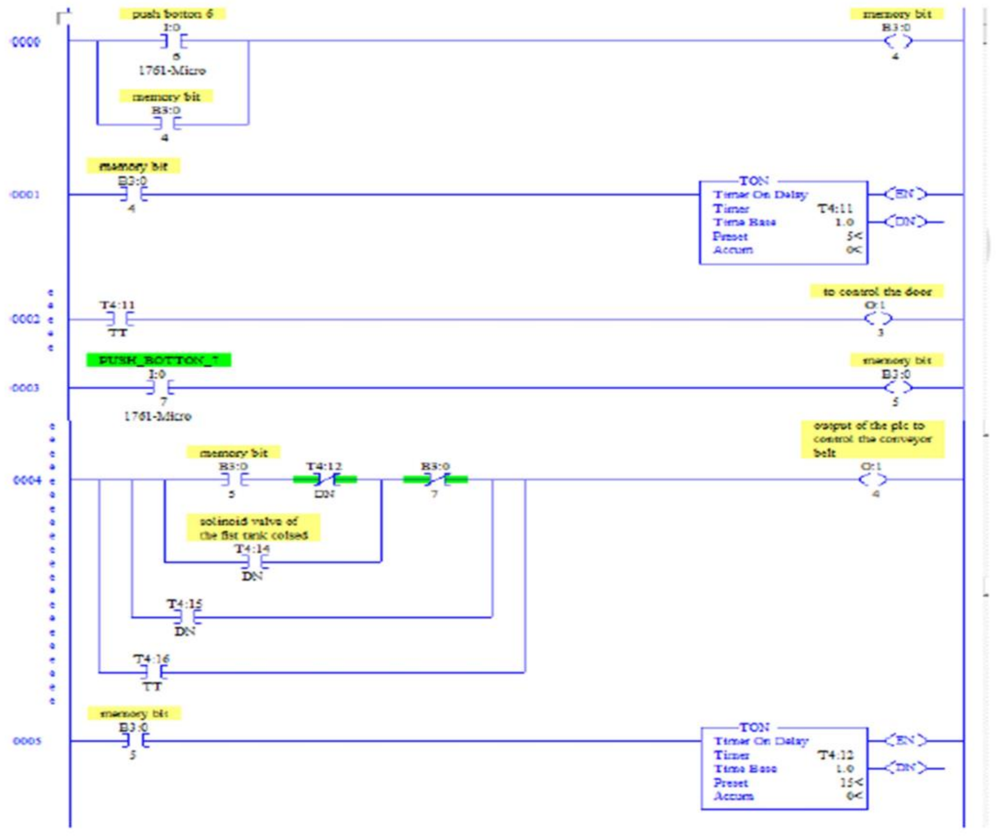
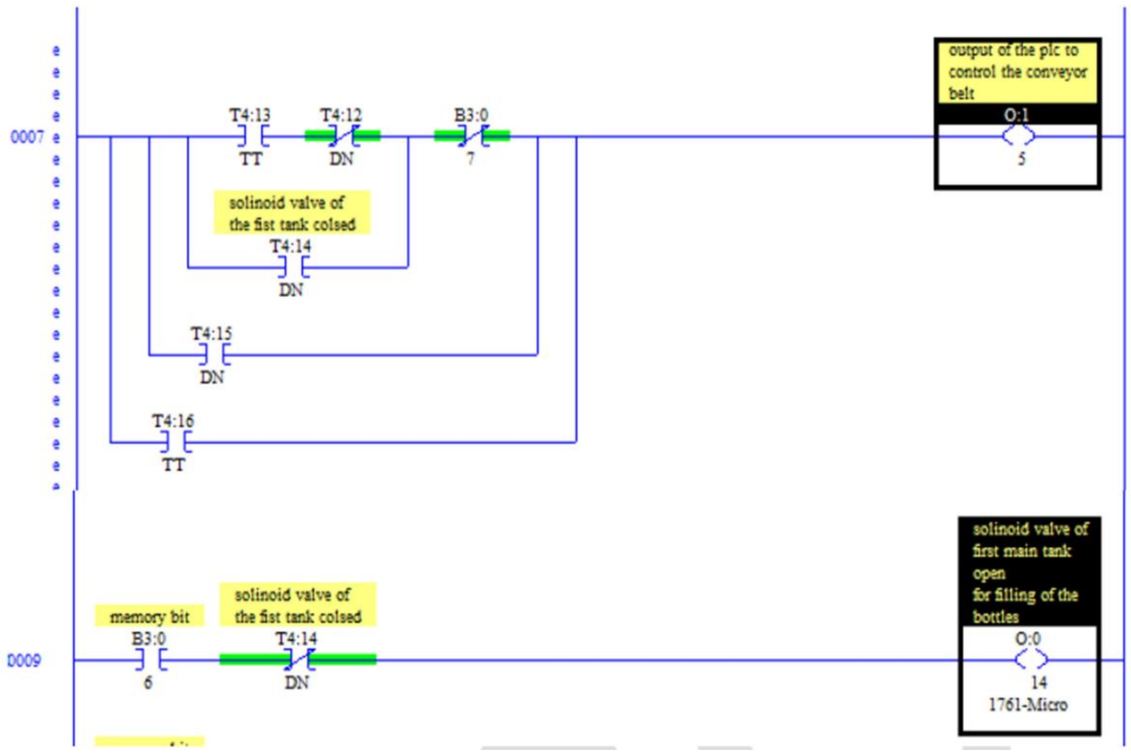


Fig 1.2 Ladder network used by the PLC types of devices



output of the plc to control the conveyor belt
O:1
T4:13
TT
T4:12
DN
B3:0
solinoid valve of the fist tank colsed
T4:14
T4:15
T4:16
memory bit
solinoid valve of the fist tank colsed
solinoid valve of first main tank open for filling of the bottles
O:0
1761-Micro

Figure 1.3 Ladder network used by the PLC types of devices

1.4 Arduino UNO

Arduino is the microcontroller which is widely used by the common people because of the following but now also it is not used by the any type of industries because if following reasons.

<u>Advantages:</u>

- It can be easily programmable using C or C++.
- Easily available in the market.
- Easy in operation.
- Most popular microcontroller board used by most of the people.

<u>Disadvantages:</u>

- Not used in industrial applications.
- It requires external DC supply, it can’t work in AC.

Chapter 2

Components used in this project

1. Arduino UNO
2. IR Sensor Module
3. R365 DC Diaphragm Water Pump
4. Relay Module
5. LCD Display
6. DC GEARED MOTOR with DRIVER
7. Arduino Sensor Expansion Shield
8. 12V 200 RPM DC Motor
9. Servo Motor
10. WIFI Module
11. Conveyor & Parts

2.1 Arduino UNO

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

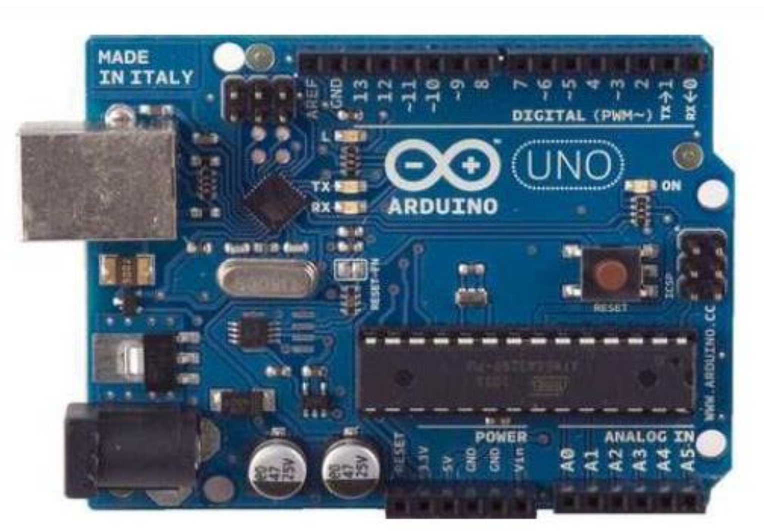


Fig2.1 Arduino UNO Board

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and VIN pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is s7 to 12 volts.

The Atmega328 has 32 KB of flash memory for storing code (of which 0.5 KB is used for the boot loader); It has also 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an *.inf file is required.

2.1.1 Arduino Architecture

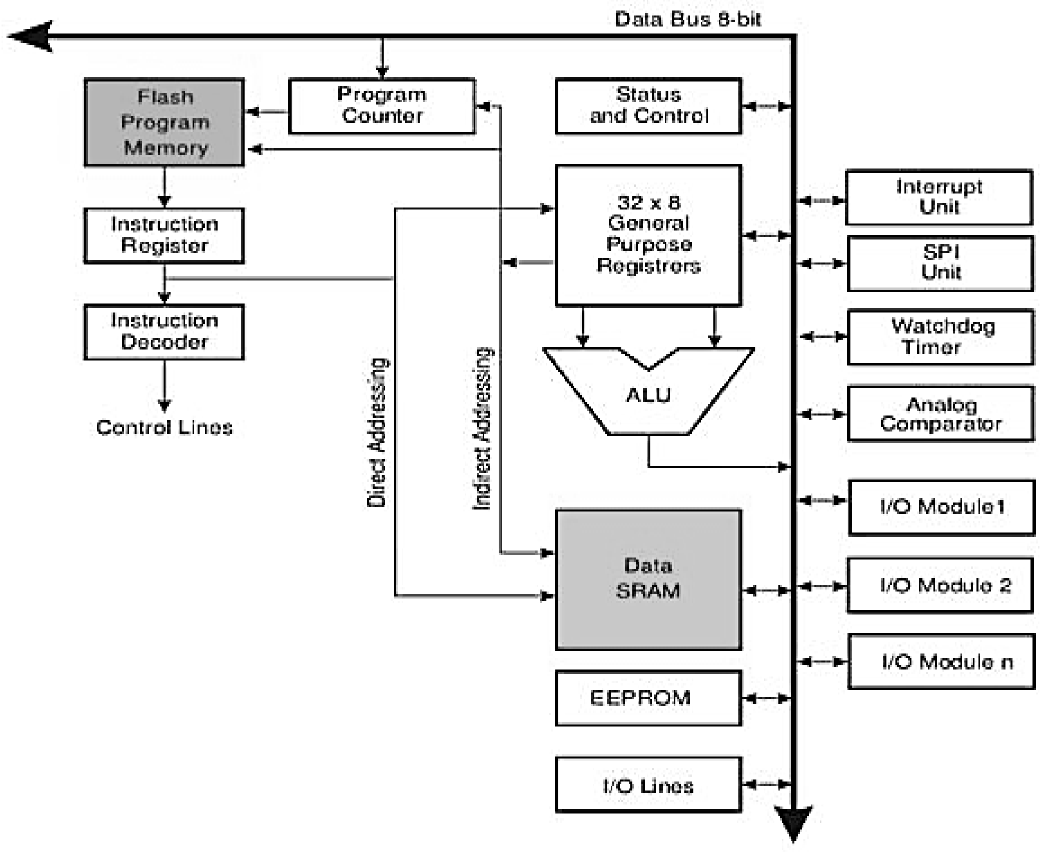


Fig 2.2 Arduino Architecture

- **Register:** In general purpose registers the registers are of 8-bit but there are 3 16-bit registers also.
    - 8-bit registers are used to store data for normal calculations and results.
    - 16-bit registers are used to store data of timer counter in 2 different register. E.g. X-low & X-high. They are fast, and are used to store specific hardware functions.
- **Instruction Register:** The data is uploaded in serial via the port (being uploaded from the computer's Arduino IDE). The data is decoded and then the instructions are sent to instruction register and it decodes the instructions on the same clock pulse. On the next clock pulse the next set of instructions are loaded in instruction register.
- **EEPROM:** EEPROM stores data permanently even if the power is cut out. Programming inside an EEPROM is slow.
- **ISR**: Interrupt Unit checks whether there is an interrupt for the execution of instruction to be executed in ISR (Interrupt Service Routine).
- **Serial Peripheral Interface (SPI):** Serial Peripheral Interface (SPI) is an interface bus commonly used to send data between microcontrollers and small peripherals such as Camera, Display, SD cards, etc. It uses separate clock and data lines, along with a select line to choose the device you wish to talk to.
- **Timer:** Watchdog timer is used to detect and recover from MCU malfunctioning.**Analog comparator:** Analog comparator compares the input values on the positive and negative pin, when the value of positive pin is higher the output is set.
- **Status and control:** Status and control is used to control the flow of execution of commands by checking other blocks inside the CPU at regular intervals.
- **ALU**: Arithmetic and Logical unit, the high performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations b/w general purpose registers are executed. The ALU operations are divided into 3 main categories – arithmetic, logical and bit-function.

- **Input and Output Pins**: I/O pins the digital inputs and outputs (digital I/O) on the Arduino are what allow you to connect the Arduino sensors, actuators, and other ICs.

Learning how to use them will allow you to use the Arduino to do some really useful things, such as reading switch inputs, lighting indicators, and controlling relay outputs.

2.1.2 Pin Configuration

The Pin Configuration of Arduino Microcontroller is given by

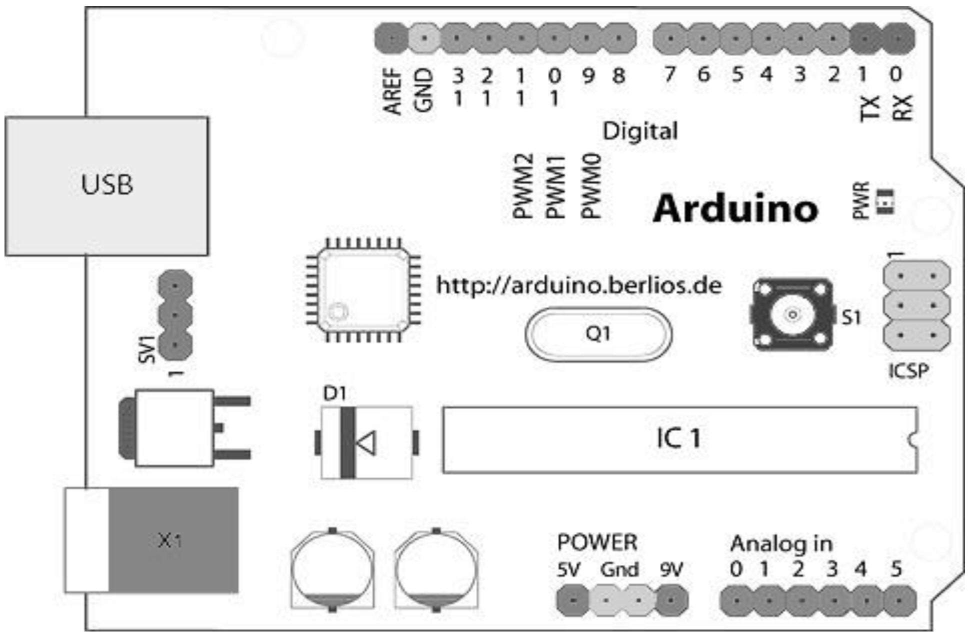


Fig 2.3 Pin Configuration

- Analog Reference pin (orange)
- Digital Ground (light green)
- Digital Pins 2-13 (green)
- Digital Pins 0-1/Serial In/Out - TX/RX (dark green) - These pins cannot be used for digital I/O (digital Read and digital Write) if you are also using serial communication (e.g. Serial. Begin).
- Reset Button - S1 (dark blue)
- In-circuit Serial Programmer (blue-green)
- Analog In Pins 0-5 (light blue)
- Power and Ground Pins (power: orange, grounds: light orange)
- External Power Supply In (9-12VDC) - X1 (pink)
- Toggles External Power and USB Power (place jumper on two pins closest to desired supply) - SV1 (purple)
- USB (used for uploading sketches to the board and for serial communication between the board and the computer; can be used to power the board) (yellow)

2.1.3 Digital Pins

In addition to the specific functions listed below, the digital pins on an Arduino board can be used for general purpose input and output via the pinMode(), digitalRead(), and digitalWrite() commands. Each pin has an internal pull-up resistor which can be turned on and off using digitalWrite() (w/ a value of HIGH or LOW, respectively) when the pin is configured as an input. The maximum current per pin is 40 mA.

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. On the Arduino Diecimila, these pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip. On the Arduino BT, they are connected to the corresponding pins of the WT11 Bluetooth module. On the Arduino Mini and LilyPad Arduino, they are intended for use with an external TTL serial module (e.g. the Mini-USB Adapter).
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function. On boards with an ATmega8, PWM output is available only on pins 9, 10, and 11.
- **BT Reset: 7.** (Arduino BT-only) Connected to the reset line of the bluetooth module.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- **LED: 13.** On the Diecimila and LilyPad, there is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

2.1.4 Analog Pins

In addition to the specific functions listed below, the analog input pins support 10-bit analog-to-digital conversion (ADC) using the analogRead() function. Most of the analog inputs can also be used as digital pins: analog input 0 as digital pin 14 through analog input 5 as digital pin 19. Analog inputs 6 and 7 (present on the Mini and BT) cannot be used as digital pins.

I^{2}C: 4 (SDA) and 5 (SCL). Support I^{2}C (TWI) communication using the Wire library (documentation on the Wiring website).

2.1.5 Power Pins

- **VIN** (sometimes labelled "9V"). The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. Note that different boards accept different input voltages ranges, please see the documentation for your board. Also note that the LilyPad has no VIN pin and accepts only a regulated input.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** (Diecimila-only) A 3.3 volt supply generated by the on-board FTDI chip.
- **GND.** Ground pins.

2.1.6 Other Pins

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** (Diecimila-only) Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

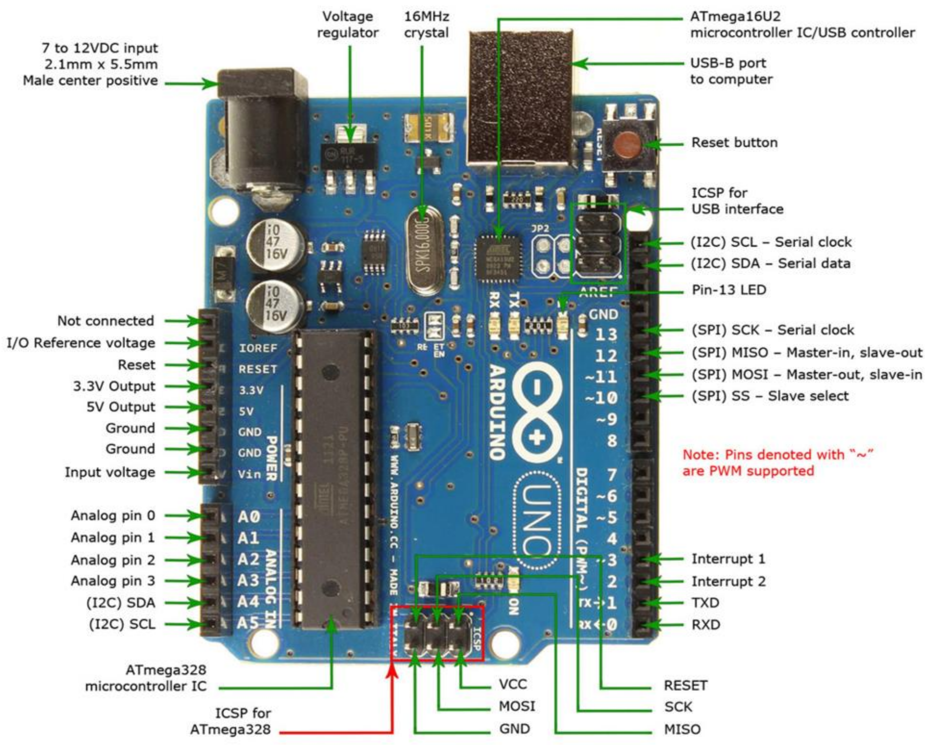


Figure 2.4 Pin out of Arduino

ATmega-328 is basically an Advanced Virtual RISC (AVR) microcontroller. It supports the data up to eight (8) bits. ATmega-328 has 32KB internal built-in memory. This micro-controller has a lot of other characteristics. ATmega 328 has 1KB Electrically Erasable Programmable
Read Only Memory (EEPROM).
This property shows if the electric supply supplied to the microcontroller is removed, even then it can store the data and can provide results
after providing it with the electric supply. Moreover, ATmega-328 has 2KB
Static Random Access Memory (SRAM). Other characteristics will be
explained later. ATmega 328 has several different features which make it the most popular device in today's market. These features consist of advanced RISC architecture, good performance, low power consumption, real timer counter having separate oscillator, 6 PWM pins, programmable Serial USART, programming lock for software security, throughput up to 20 MIPS etc. ATmega-328 is mostly used in Arduino.

2.1.1 Specification of Arduino UNO:

Components	**Range**
Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Digital I/O Pins	14
PWM Digital I/O Pins	6

2.2 IR Sensor

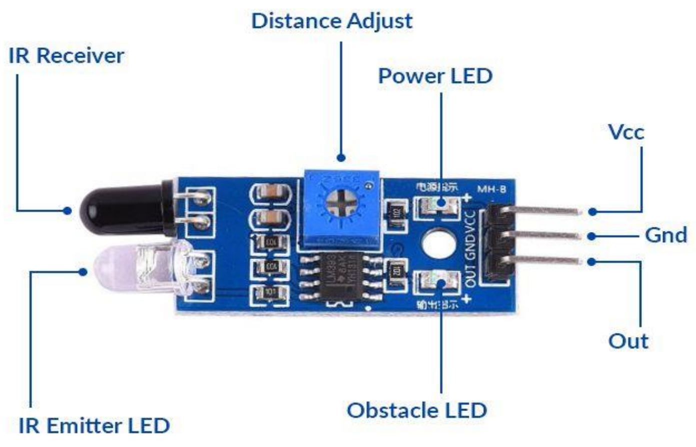


Fig. 2.2 IR Sensor

The emitter is simply an IR LED (Light Emitting Diode) and the detector is simply an IR photodiode that is sensitive to IR light of the same wavelength as that emitted by the IR LED. When IR light falls on the photodiode, the resistances and the output voltages will change in proportion to the magnitude of the IR light received.

Working Principle

The working principle of an infrared sensor is similar to the object detection sensor. This sensor includes an IR LED & an IR Photodiode, so by combining these two can be formed as a photo-coupler otherwise optocoupler. The physics laws used in this sensor are planks radiation, Stephan Boltzmann & weins displacement.

IR LED is one kind of transmitter that emits IR radiations. This LED looks similar to a standard LED and the radiation which is generated by this is not visible to the human eye. Infrared receivers mainly detect the radiation using an infrared transmitter. These infrared receivers are available in photodiodes form. IR Photodiodes are dissimilar as compared with usual photodiodes because they detect simply IR radiation. Different kinds of infrared receivers mainly exist depending on the voltage, wavelength, package, etc.

2.2.1 Specifications

Components	**Range**
Main Chip	LM393
Operating Voltage (VDC)	3.6 ~ 5
Average Current Consumption (mA)	0.06
Detection Angle	35 Â°
Distance Measuring Range (CM)	2 ~ 30
Dimensions (mm) LxWxH	48 x 14 x 8
Weight (gm)	5
Shipment Weight	0.135 kg
Shipment Dimensions	5 × 4 × 1 cm

2.3 DC DIAPHRAGM WATER PUMP

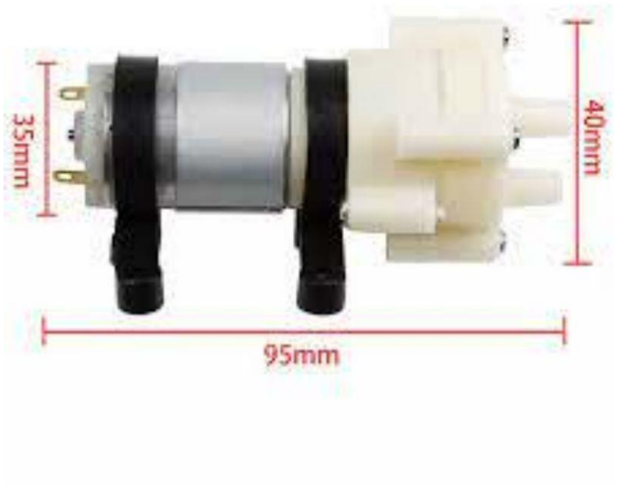


Fig 2.3 R365 DC Micro Diaphragm Pump

A double diaphragm is a positive displacement pump which utilises two flexible diaphragms that reciprocate back and forth, creating a temporary chamber, which both draws in and expels fluid through the pump. The diaphragms work as a separation wall between the air and the liquid. The very basic construction of a dc motor contains a current carrying armature which is connected to the supply end through commutator segments and brushes and placed within the north south poles of a permanent or an electro-magnet

A Diaphragms pump uses two bendable diaphragm that respond back and forward to make temporary hall where both receives and eject liquid through the pump .

The working principle of the pump is on air displacement principle which is like a separation partition among the air as well as the fluid.

2.3.1 Specifications

Name	**Range**
Model	R365 DC micro diaphragm pump
Working voltage	DC 12V
No load current	0.23A
Maximum flow	2-3 liters/minute
Outlet maximum pressure	1-2.5 kg
Maximum lift	-2.5 meters
Maximum suction	2 meters
Motor length	32mm
Motor diameter	28mm
Pump length	36mm
Total length:	69mm
Pump diameter	40mm x 35mm

2.4 RELAY MODULE

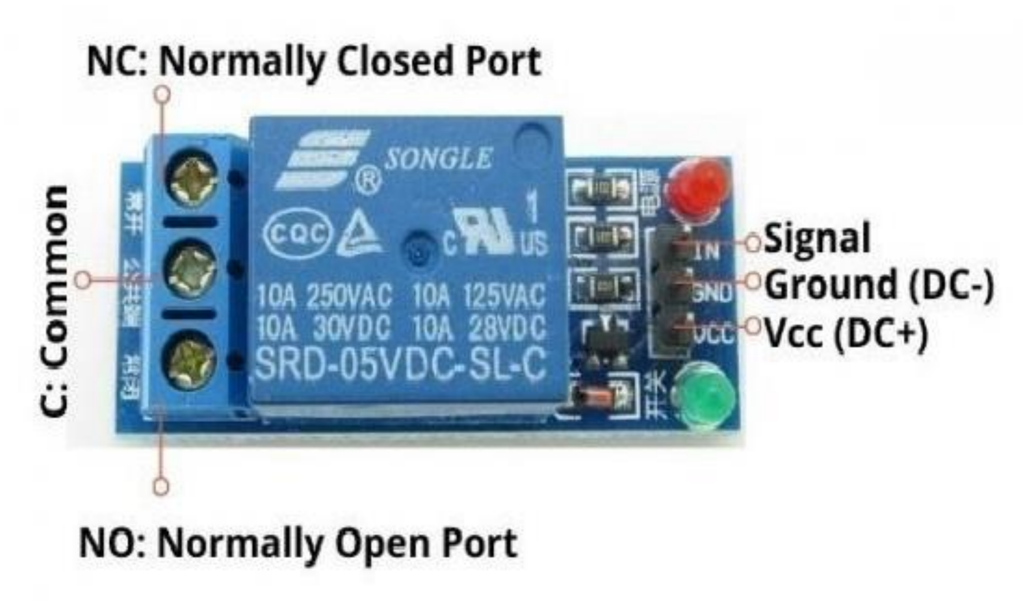


Fig 2.4 Single Channel Relay Module

Relay is an electromagnetic device. It is used as an automatic switch. It opens and closes the circuit from the supply to the device under control. Relays can be of different types like electromechanical, solid state. Electromechanical relays are frequently used. Although many different types of relay were present, their working is same. Every electromechanical relay consists of an consists of an

• Electromagnet

• Mechanically movable contact

• Switching points

• Spring

Relay Electromagnet is constructed by wounding a copper coil on a metal core. The two ends of the coil are connected to two pins of the relay as shown. These two are used as DC supply pins. Generally, two more contacts will be present, called as switching points to connect high ampere load. Another contact called common contact is present in order to connect the switching points.

These contacts are named as normally open (NO), normally closed(NC) and common(COM) contacts. Relay can be operated using either AC or DC.

In case of AC relays, for every current zero position, the relay coil gets demagnetized and hence there would be a chance of continues breaking of the circuit. So, AC relays are constructed with special mechanism such that continues magnetism is provided in order to avoid above problem. Such mechanisms include electronic circuit arrangement or shaded coil mechanism.

2.4.2 Working Principle

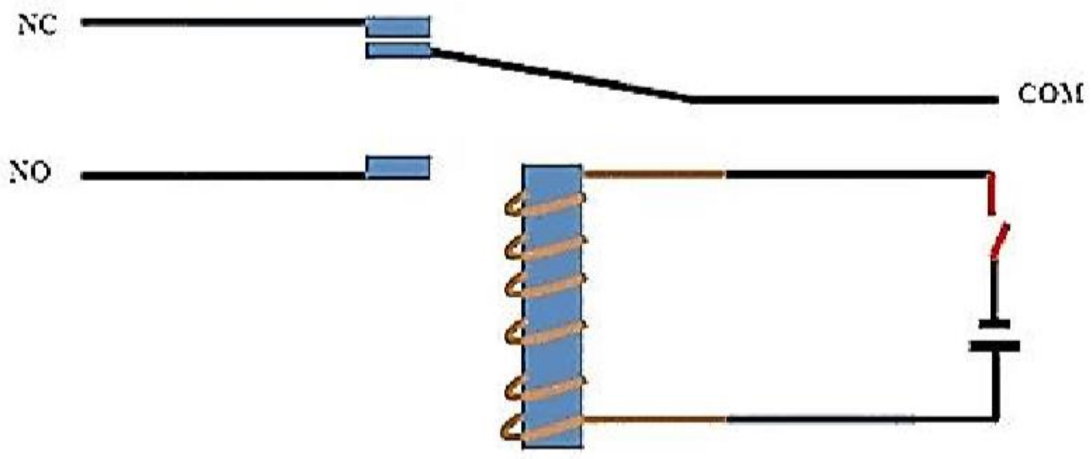


Fig 2.5 Working Principle

Relay works on the principle of electromagnetic induction. When the electromagnet is applied with some current it induces a magnetic field around it. Above image shows working of the relay. A switch is used to apply DC current to the load. In the relay Copper coil and the iron core acts as electromagnet. When the coil is applied with DC current it starts attracting the contact as shown. This is called energizing of relay. When the supply is removed it retrieves back to the original position. This is called De energizing of relay. There are also such relays, whose contacts are initially closed and opened when there is supply i.e. exactly to opposite to the above shown relay. Solid state relays will have sensing element to sense the input voltage and switches the output using opto-coupling.

Depending on the poles and throws, relays are classified into

• Single pole single throw

• Single pole double throw

• Double pole single throw

• Double pole double throw

• Single Pole Single Throw

Single Pole Single Throw

A single pole single throw relay can control one circuit and can be connected to one output. It is used for the applications which require only ON or OFF state.

Single Pole Double Throw

A single pole double throw relay connects one input circuit to one of the two outputs. This relay is also called as changeover relay. Though the SPDT has two output positions, it may consist of more than two throws depends on the configuration and requirement of the application.

Double pole single throw

A double pole single throw relay has two poles and single throw and it can be used to connect two terminals of a single circuit at a time. For example, this relay is used for connecting both phase and neutral terminals to the load at a time.

Double pole double throw

A DPDT (double pole double throw) relay has two poles and two throws for each pole. In motor direction control, these are used for phase or polarity reversal. The switching action between contacts for all these relays is performed when the coil get energized as shown in figure below.

Relays can be classified into different types depending on their functionality, structure, application etc.

2.4.1 SPECIFICATIONS:

Components	Range
Supply Voltage	3.75 to 6 V
Supply Current with Relay De-Energized	2 mA
Supply Current with Relay Energized	70 to 72 mA
Input Control Signal	Active Low
Input Control Signal Current	1.5 to 1.9 mA
Relay Max Contact Voltage	250 VAC or 30 VDC
Relay Max Contact Current	10 A

2.5 LCD Display 20x4 with I2C Module

Fig 2.5 LCD Display 20x4

This 20x4 Character LCD Display is built-in with RW1063 controller IC which are 6800, 4 line SPI or I2C interface options. The WH2004G 20x4 LCD Display have the same AA size and pin assignment as existing WH2004A and WH2004B character LCD modules but with smaller outline and VA size. Here it use I2C interface.

I2C Module LCD Serial Display Adapter

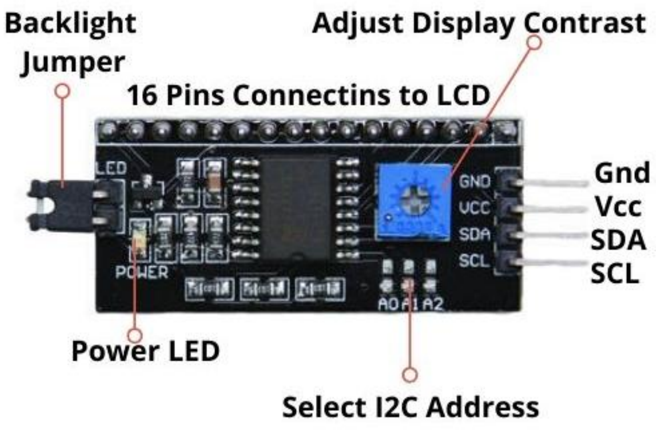


Fig.2.6 I2c Module

I2C Module is a parallel to serial converter compatible with LCD2004 and LCD1602. By using this module, LCD can be interfaced with using only 2 wires. LCD displays take 8 pins so sometimes user can get out of resources, this device helps to save the resources as it takes only 4 pins. Standard 5V supply voltage, Contrast is adjustable. The backlight can be turned on and off using sorting jumper.

2.5.1 Specification

Specification	**Range**
Model	LCD2004
Characters	20x4
Character Colour	Black
Backlight	Yellow
Input Voltage (V)	5

2.6 DC GEARED MOTOR with DRIVER

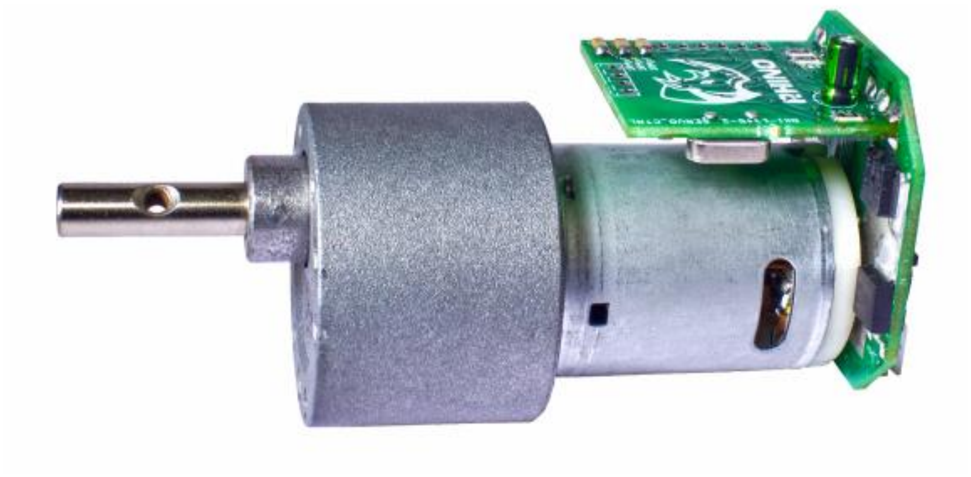


Fig.2.7 DC Geared Motor

12V Geared motors are generally a simple DC motor with a gearbox attached to it. This can be used in all-terrain robots and variety of robotic applications. These motors have a 3 mm threaded drill hole in the middle of the shaft thus making it simple to connect it to the wheels or any other mechanical assembly

2.7.1 Specifications

Specification	**Range**
Supply Voltage	11v to 13v
No-load current , Load current	500mA ,up to 7.5A(Max)
Base motor Speed	18000RPM
Diameter shaft with M3 thread hole	6mm
Motor Diameter	28.5mm
Length	63mm without shaft
Shaft length	30mm

Current controlled Drive Specifications

- High-Current DC Constant-Torque motor drive integrated with the motor
- Motor speed control interface via UART, I2C, PPM signal and analog input
- Speed control possible in both directions down to almost 1% of max. speed
- Small package and integration allows for easy installation and operation
- Speed can be controlled using a terminal or MCU via simple UART commands
- I2C master device can control multiple RMCS-210x via simple I2C command structures
- An RC receiver or any PPM source can directly control the speed of the motor
- An analog signal or fixed analog voltage from a potentiometer can directly control the speed of the motor

2.6.3 Power and Input Terminal Assignments Terminal

Terminal 1	GND	BLACK	Ground
Terminal 2	PPM	BROWN	PPM input signal
Terminal 3	SDA	RED	I2C Data / Analog Input Sense
Terminal 4	UART TXD	ORANGE	UART Data Transmit
Terminal 5	UART RXD	YELLOW	UART Data Receive
Terminal 6	V+	GREEN	V+

2.7 Arduino Sensor Expansion Shield

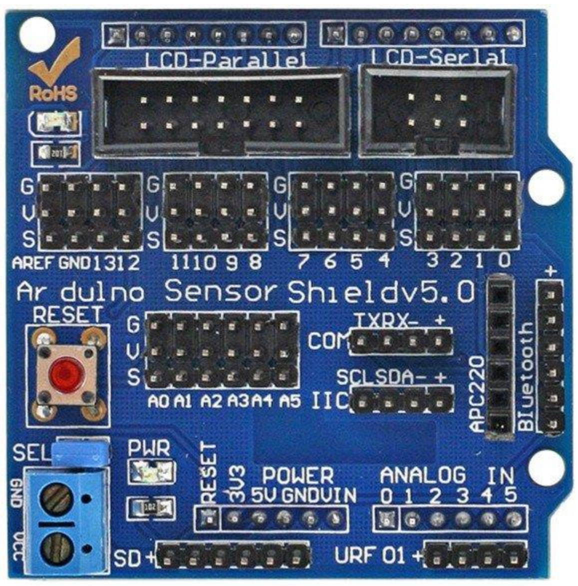


Fig. 2.8 Arduino Sensor Expansion Shield

Sensor Shield allows you to connect to various modules like sensors, servos, relays, buttons, potentiometers and many more directly to your Arduino through this Sensor Shield. Each functional module has buckled port with VCC, GND, and Output, which has the corresponding port on the Sensor Shield, connected with a plain 2.54mm dual-female cable. Each functional module has buckled a port with VCC, GND and Output, which has a corresponding port on the Sensor Shield. It have a Reset Button. Red LED indicates the power status.

I/O Ports:

- 14 Digital I/O Ports: D0-D13 (including 6 ports for PWM function)
- 8 Analog I/O Ports: A0-A5

Interfaces:

- SD Card Interface
- Ultrasonic Interface
- Bluetooth Interface
- APC220 Interface
- IIC Interface
- Servo Interface
- 12864 LCD Parallel/ Serial Interface

2.8 12V 200 RPM DC Motor

Fig. 2.9 DC Motor

These motors are simple DC Motors featuring gears for the shaft for obtaining the optimal performance characteristics. They are known as Centre Shaft DC Geared Motors because their shaft extends through the centre of their gearbox assembly.

2.8.1 Specifications

Specification	**Range**
Gear Material	Plastic
Rated Speed (RPM)	200
Operating Voltage (VDC)	12
Rated Torque(kg-cm)	1.5
Stall Torque(Kg-Cm)	5.4
Load Current Max	300
No-Load Current (mA)	60
Gearbox Diameter (mm)	37
Motor Diameter(mm)	32
Motor Length (mm)	75
Shaft Diameter (mm)	6
Shaft Length (mm)	22
Weight (gm)	80

2.9 Servo Motor

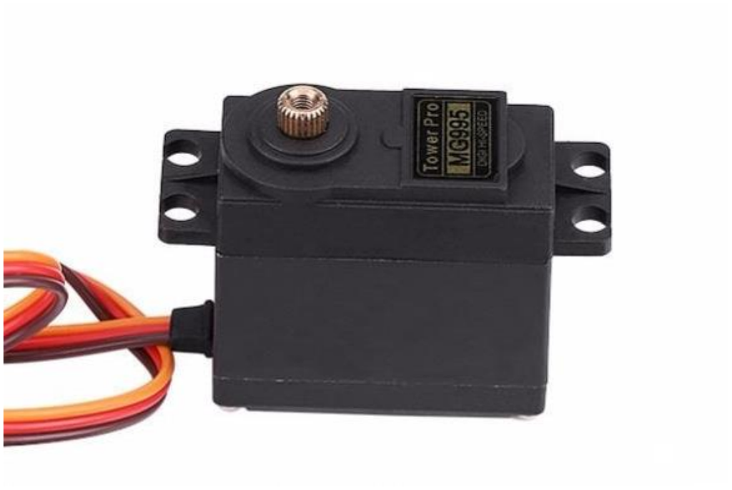


Fig. 2.10 Servo Motor

The simplicity of a servo is among the features that make them so reliable. The heart of a servo is a small direct current (DC) motor, similar to what you might find in an inexpensive toy. These motors run on electricity from a battery and spin at high RPM (rotations per minute) but put out very low torque. An arrangement of gears takes the high speed of the motor and slows it down while at the same time increasing the torque. The gear design inside the servo case converts the output to a much slower rotation speed but with more torque (big force, little distance). The amount of actual work is the same, just more useful. Gears in an inexpensive servo motor are generally made of plastic to keep it lighter and less costly. Servo Motor rotates 90° in each direction making it a 180° servo motor. It is a Digital Servo Motor that receives and processes PWM signal faster and better. It equips sophisticated internal circuitry that provides good torque, holding power, and faster updates in response to external forces.

Wire Description

RED – Positive

Brown – Negative

Orange – Signal

2.9.1 Specifications

Specification	**Range**
Model	MG995
Weight(gm)	55
Operating Voltage (VDC)	4.8 ~ 7.2
Operating Speed @4.8V	0.20sec/60°
Operating Speed @6.6V	0.16sec/60°
Stall Torque @ 4.8V (Kg-Cm)	10
Stall Torque @6.6V (Kg-Cm)	12
Operating Temperature (°C)	-30 to 60
Dead Band Width (µs)	1
Gear Type	Semi-Metal
Rotational Degree	180º
Servo Plug	JR
Cable Length (cm)	30
Length (mm)	40.5
Width (mm)	20
Height (mm)	44
Shipment Weight	0.059 kg
Shipment Dimensions	9 × 8 × 6 cm

2.10 WI-FI Module

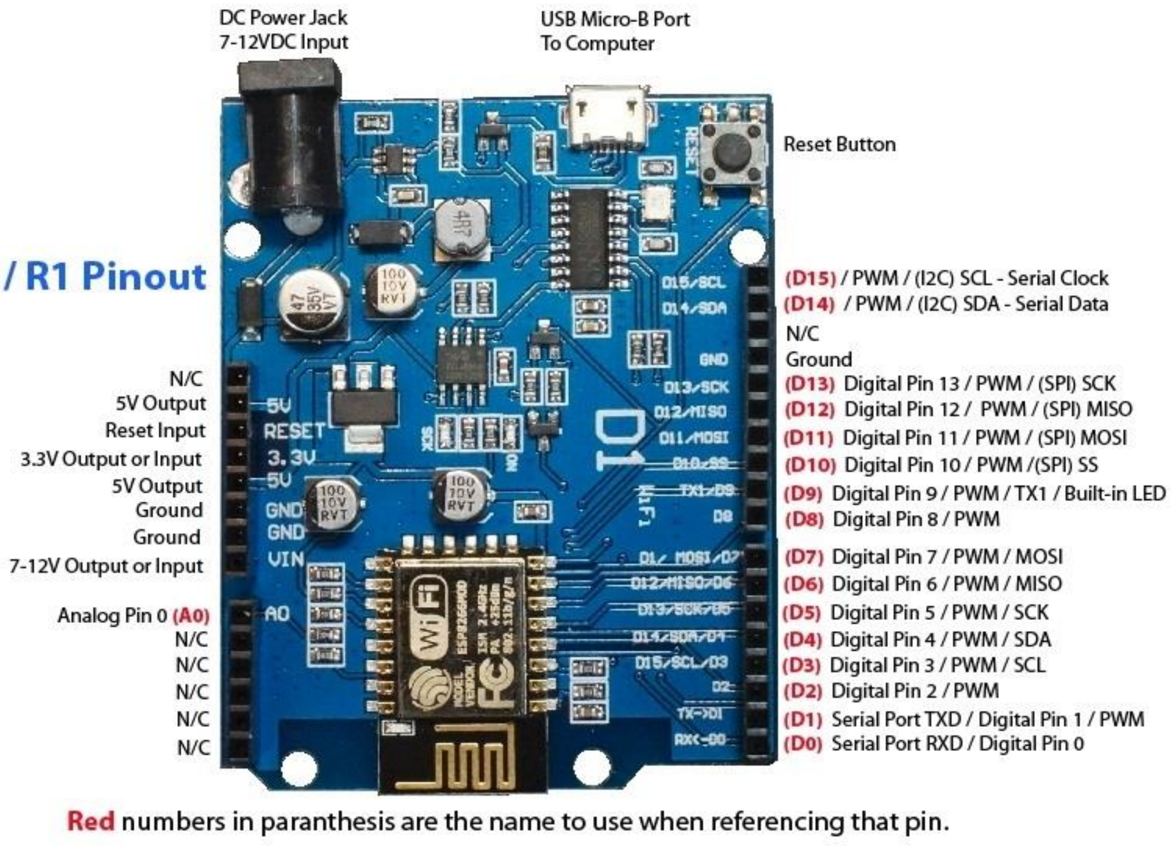

DC Power Jack
7-12VDC Input
USB Micro-B Port
To Computer
Reset Button
D1 / R1 Pinout
(D15) / PWM / (I2C) SCL - Serial Clock
(D14) / PWM / (I2C) SDA - Serial Data
N/C
Ground
(D13) Digital Pin 13 / PWM / (SPI) SCK
(D12) Digital Pin 12 / PWM / (SPI) MISO
(D11) Digital Pin 11 / PWM / (SPI) MOSI
(D10) Digital Pin 10 / PWM /(SPI) SS
(D9) Digital Pin 9 / PWM / TX1 / Built-in LED
(D8) Digital Pin 8 / PWM
(D7) Digital Pin 7 / PWM / MOSI
(D6) Digital Pin 6 / PWM / MISO
(D5) Digital Pin 5 / PWM / SCK
(D4) Digital Pin 4 / PWM / SDA
(D3) Digital Pin 3 / PWM / SCL
(D2) Digital Pin 2 / PWM
(D1) Serial Port TXD / Digital Pin 1 / PWM
(D0) Serial Port RXD / Digital Pin 0
N/C
5V Output
Reset Input
3.3V Output or Input
5V Output
Ground
Ground
7-12V Output or Input
Analog Pin 0 (A0)
N/C
N/C
N/C
N/C
N/C
Red numbers in paranthesis are the name to use when referencing that pin.

Fig. 2.11 WEMOS D1 R1 Pin Out

WEMOS D1 R1 is a WI-FI development board based on ESP8266 12E. The functioning is similar to that of NODEMCU, except that the hardware is built resembling Arduino UNO. The D1 board can be configured to work on Arduino environment using BOARDS MANAGER. So, it’s not necessary to use a separate Arduino board, the D1 itself can be programmed on Arduino IDE. This is handy in using with IOT projects. Further many Arduino SHIELDs are compatible with the D1 board. Different versions of D1 boards are available in the market R1, R2, MINI as the name implies D1 Mini is a smaller version R1, R2 boards look like Arduino UNO board, but version is not printed on board. On some boards WEMOS D1is printed, as the ESP12 chip is manufactured by WEMOS. In some other boards it is printed just D1, where the ESP12 chip is that of AI Thinker. As the ESP12 chip has only one ANALOG pin, on board there is only A0 & other Analog pins are dummy. WeMos-D1R2 is an ESP8266-12 based Wi-Fi enabled microprocessor unit on a Arduino-UNO footprint. That means the board looks and works (in most cases) like an UNO.

Apparently several shields, sensors and output devices that are manufactured for the Arduino platform will work on the WeMos-D1R2 with the added advantage of built-in Wi-Fi.

The D1 R1 is a mini Wi-Fi board based on ESP-8266EX. 11 digital input/output pins, all pins have interrupt/PWM/I2C/one-wire supported(except D0) 1 Analog input(3.3V max input) A Micro USB connection A power jack, 9-24V power input. Compatible with Arduino.

2.10.1 Specifications

Specification	**Range**
Microcontroller	ESP8266
Digital I/O Pins	11
Operating Voltage (VDC)	3.3
Analog I/O Pins	1
Flash Memory	4 MB
Dimensions in mm (LxWxH)	68x54x12
Weight (gm)	20

All IO have interrupt/PWM/I2C/one-wire supported (except D0) Programming: The D1 R1 has a micro USB for auto programming. Also you can programming it using OTA Warnings: All IO is work at 3.3V.

2.11 Conveyor

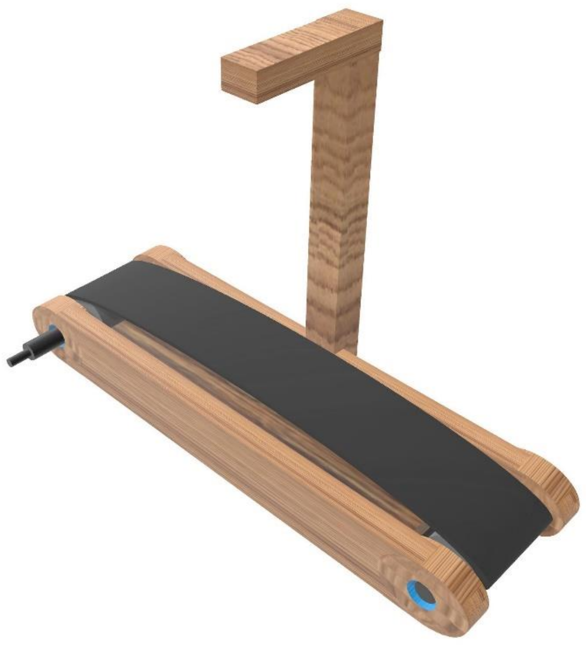


Fig. 2.12 Conveyor System

A conveyor system is a mechanical handling equipment that moves materials from one location to another.

There are two types of conveyor

- Rotational Conveyor
- Straight line Conveyor

This is a straight-line conveyor

It uses motor for the rotation of the roller of conveyor so that the conveyor belt moves, using a controller the speed and direction of the conveyor belt can be controlled, other side of the roller is freely rotating.

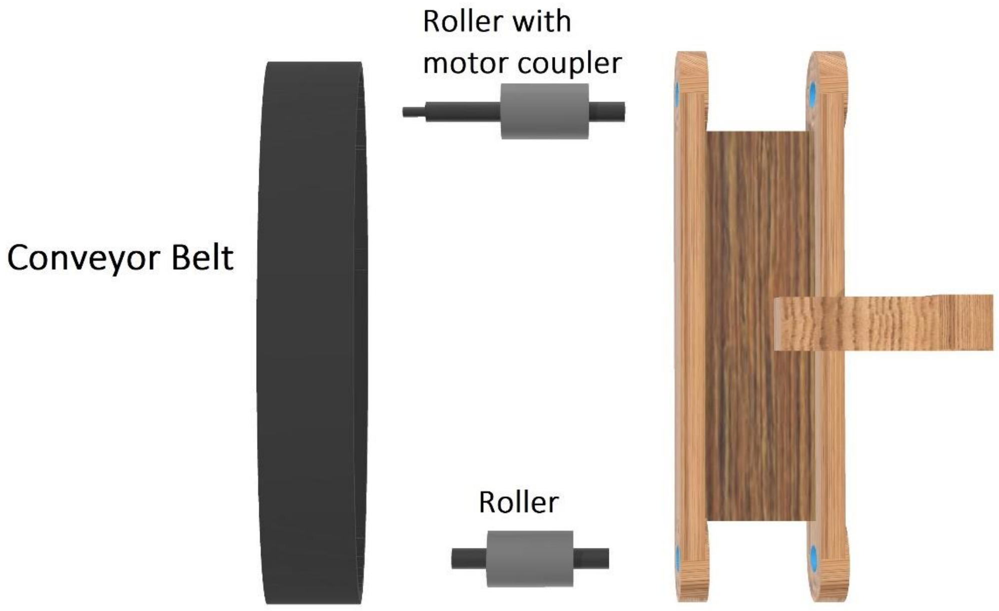


Fig.2.13 Conveyor Parts

It consists of a platform for support, conveyor belt, two rollers, one roller is coupled with motor and also support for water filling system and capping system.

Chapter 3

Implementation of components with Arduino

3.1 Interfacing of Arduino with IR Sensor

Here we are controlling filling based on the input given by IR sensor for our project we are sensing the Bottle which is to be used for filling the material.

IR Senor output is connect with Arduino's Digital PIN 5 which is used as Arduino Pin mode as Input mode, VCC and GND of the sensor is connected with Arduino's Power PIN. IR Sensor can transmit and receive infrared light to detect the object by sending IR light into the object and the IR light reflect back into the IR receiver. The output of the IR Sensor either can be HIGH or LOW where High is 5v and Low is 0v.

Connection diagram shown below

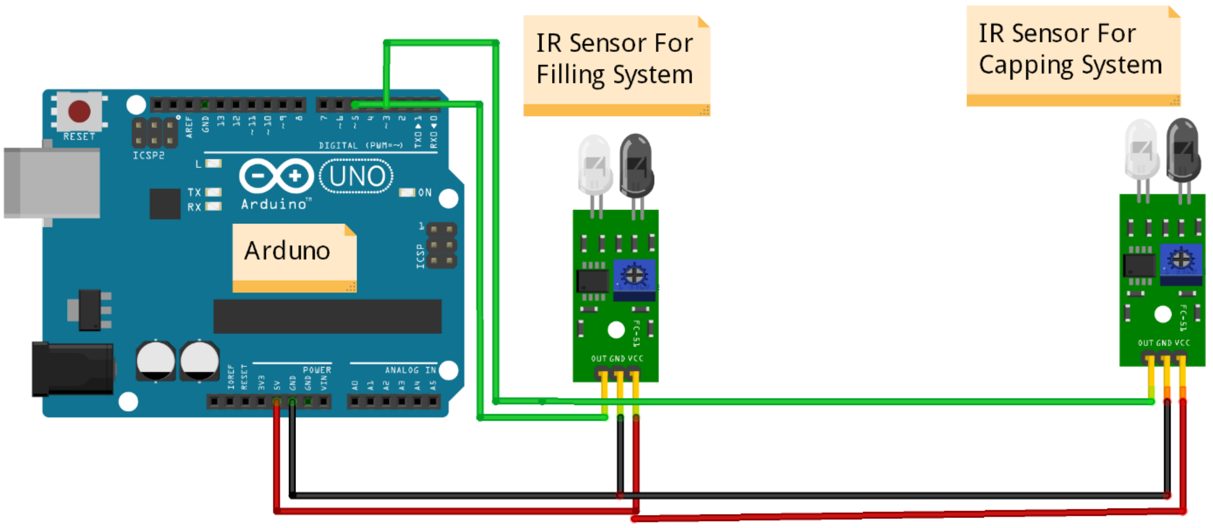


Fig. 3.1 IR Sensor Connection Diagram

3.2 Interfacing of Arduino with Relay of Water Pump

Here we are controlling Water Pump by Relay Module which is controlled by Arduino when the bottle is detect by IR sensor.

Relay Input is connected with Arduino's Digital PIN 6 which is used as Arduino Pin mode as Output mode, VCC and GND of the relay module is connected with Arduino's Power PIN. Normal Open (NO) and GND connection of the relay is connected with 12v power supply

DC Water pump is turn ON/OFF by using relay when the Arduino using 5v from the Arduino the electromagnet close the contract of the circuit make the pump connect with 12v Power Supply. Relay is controlled by Arduino when the output pin becomes 5v therefore the output is HIGH so relay turn ON and close the circuit and it open the circuit when output is LOW which is 0v.

Connection diagram shown below.

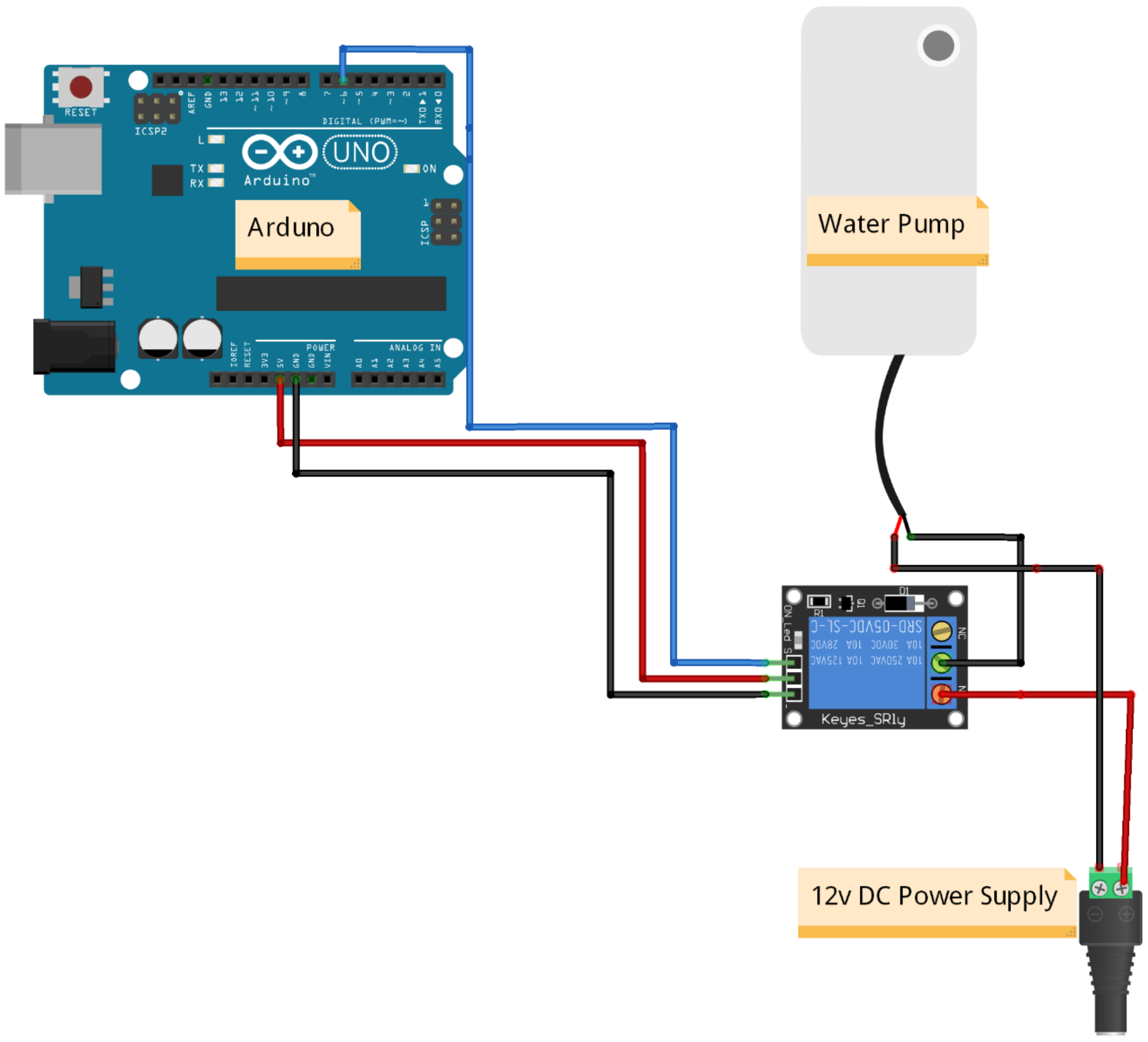


Fig. 3.2 Water Pump and Relay Connection Diagram

3.3 Interfacing of Arduino with LCD

For this project we are using LCD display to show at current status which process is running. Arduino Analog PINS A4 and A5 are used to communication with Arduino and LCD Module.

There are two ways of communication between Arduino and LCD module

- Serial Communication
- Parallel Communication

Here LCD Module use I2C communication to transfer data between Arduino and LCD module with 12C module. I2C LCD Adapter is a chip converts the I2C data from an Arduino into the parallel data required by the LCD display. The board also comes with a small potentiometer to make fine adjustments to the contrast of the display. In addition, there is a jumper on the board that supplies power to the backlight.
Connection diagram shown below.

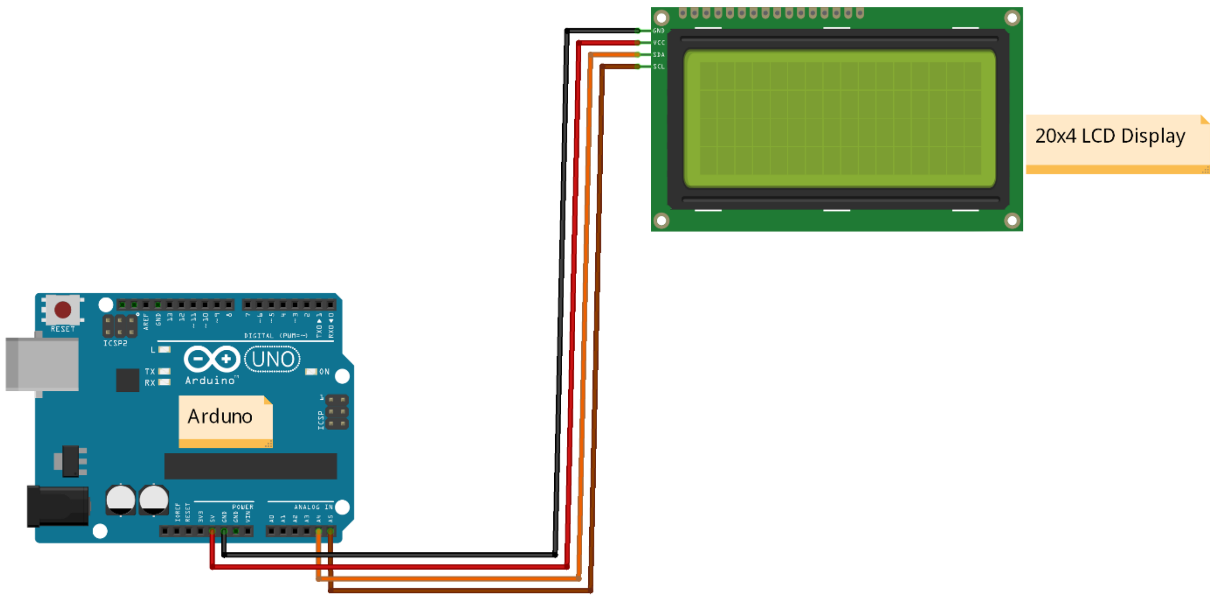


Fig. 3.3 LCD Display Connection Diagram

3.4 Interfacing of Arduino with DC Geared Motor

In this project we are using DC motor to control the conveyer belt which can be control by Arduino UNO microcontroller board with the help of following
Connecting as per as circuit diagram shown below.

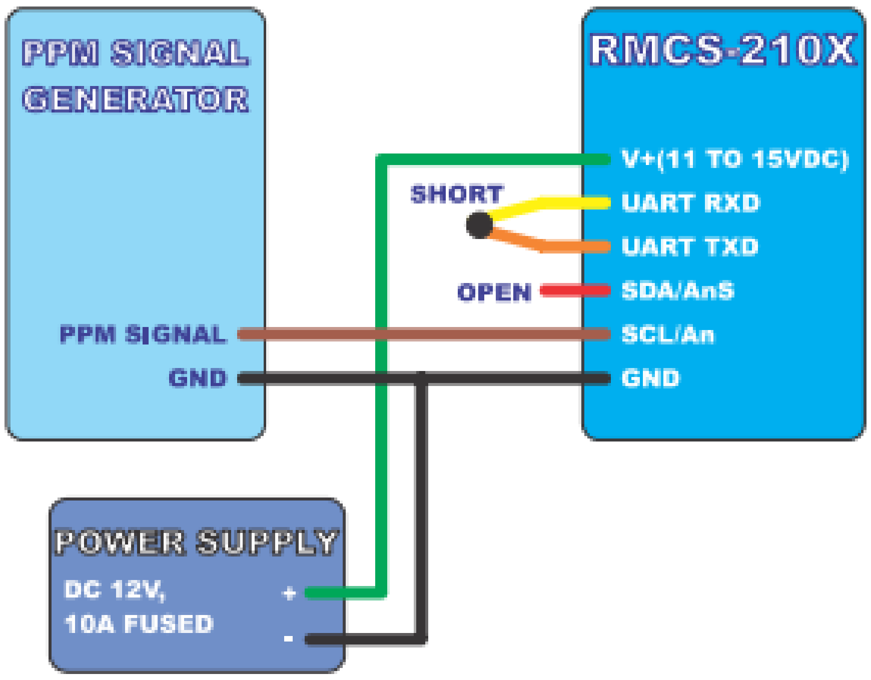


Fig. 3.4 Connection of Arduino with DC Motor of Conveyor

The motor use PPM (Pulse Position Modulation) SCL is connect to the Arduino Digital PIN 7, the UART RXD and UART TXD pins are short and SDA pin is open ,V+ and GND is connect to power and ground of Arduino.

3.5 Interfacing of Arduino with Relay of Capping DC Motor

Here we are controlling DC Motor by Relay Module which is controlled by Arduino when the bottle is detect by IR sensor.

Relay Input is connected with Arduino's Digital PIN 2 which is used as Arduino Pin mode as Output mode, VCC and GND of the relay module is connected with Arduino's Power PIN. Normal Open (NO) and GND connection of the relay is connected with 12v power supply. Relay is controlled by Arduino when the output pin becomes 5v therefore the output is HIGH so relay turn ON and close the circuit and it open the circuit when output is LOW which is 0v. Connection diagram shown below.

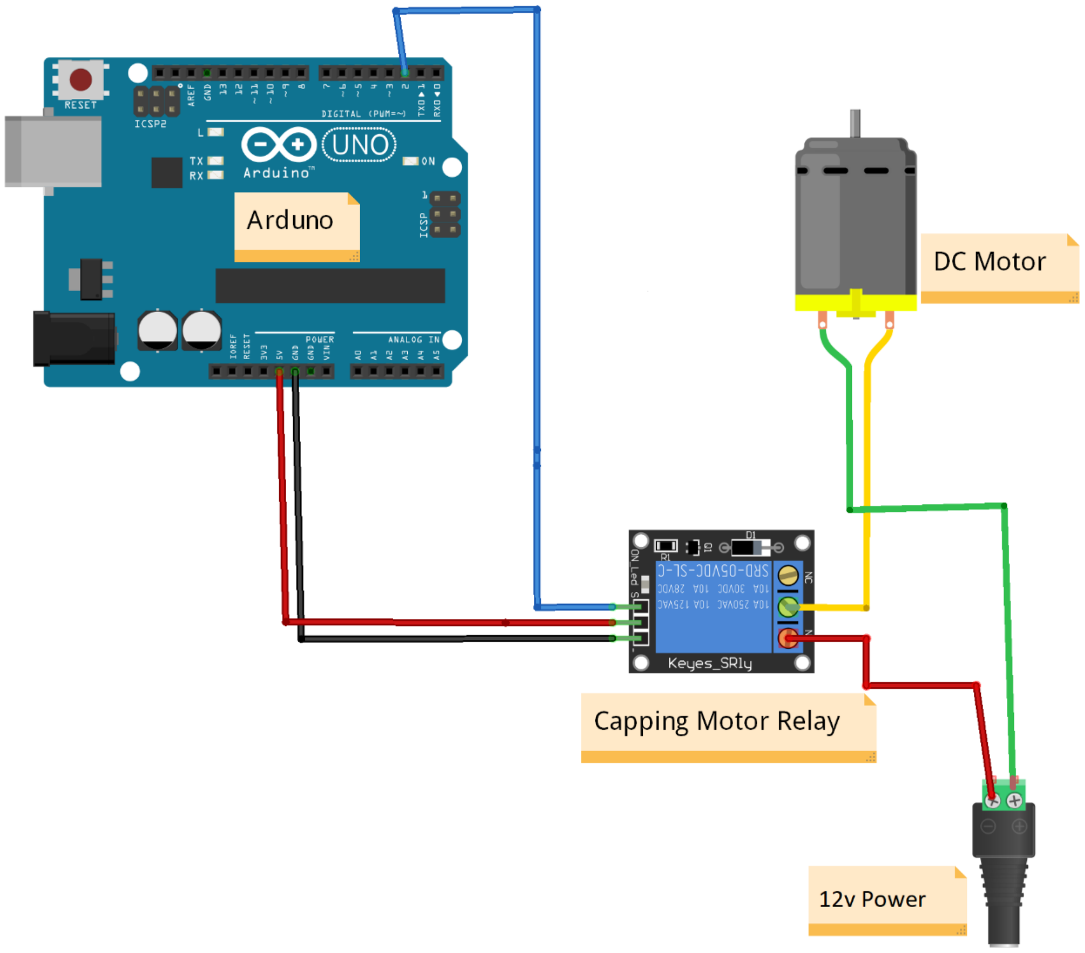


Fig. 3.5 Connection of Arduino with DC Motor and Relay

3.6 Interfacing of Arduino with Servo Motor

Servo Motor is connected to the Digital Pin 8 of the Arduino and VCC and GND is connected to the Arduino's 5v Power Pin and GND Pin.

Servo Motor is used to turn the capping motor up and down when the bottle is detected. Default position of Servo motor arm is programmed to 45°.

Servo is a general term for a closed loop control system. A closed loop system uses the feedback signal to adjust the speed and direction of the motor to achieve the desired result. It can be control the servo motor by sending a series of pulses to the signal line. A conventional Analog servo motor expects to receive a pulse roughly every 20 milliseconds (i.e. signal should be 50Hz). PWM (Pulse Width Modulation) signal is send by Arduino to servo motor, Pulses ranging between 1ms and 2ms will move the servo shaft through the full 180 degrees of its travel.

Connection diagram shown below.

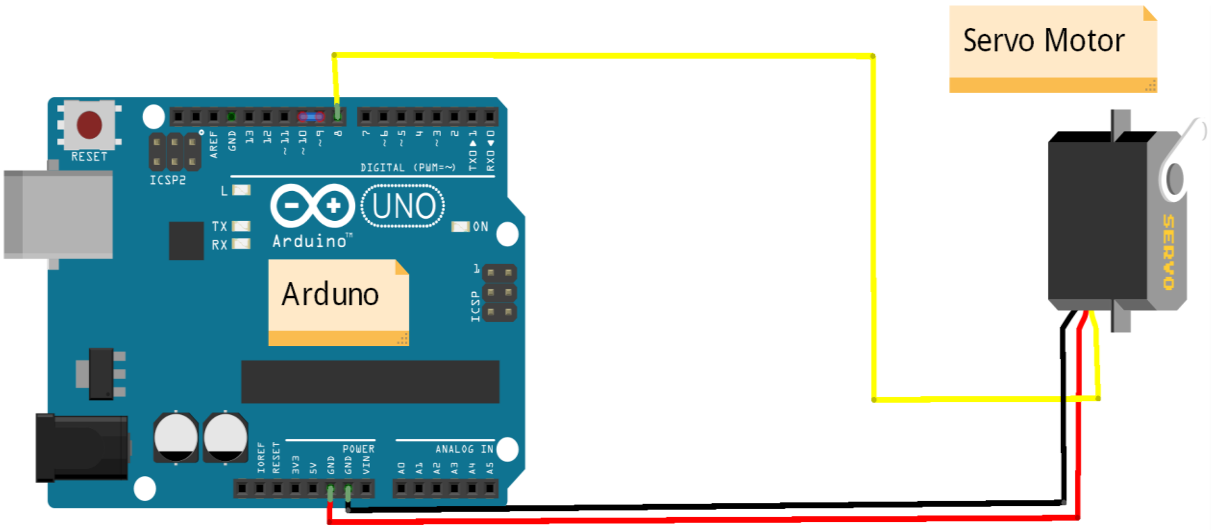


Fig. 3.6 Connection of Arduino with Servo Motor

3.7 Interfacing of Arduino with Wi-Fi Module

Wi-Fi Module here we use WEMOS D1 R1 Development Board for transfer data between Wi-Fi Based Web Server and the Arduino. It is based on ESP8266-12E which is similar to NODEMCU.

Here WEMOS Board Digital Pins are connected with the Arduino's Digital Pins from 1 to 8 and WEMOS can be powered from Micro-USB or build-in Power Pins. Wi-Fi module can either host or connect to a network and create a webserver. Webserver is programmed into the WEMOS board as HTML code. The changes in the input and output pins of Arduino is detected by this module and send and receive request to the webserver by AJAX (Asynchronous JavaScript and XML) which is set of web development techniques using many web technologies on the client-side to create asynchronous web applications. With Ajax, web applications can send and retrieve data from a server asynchronously without interfering with the display and behaviour of the existing page.
Connection diagram shown below.

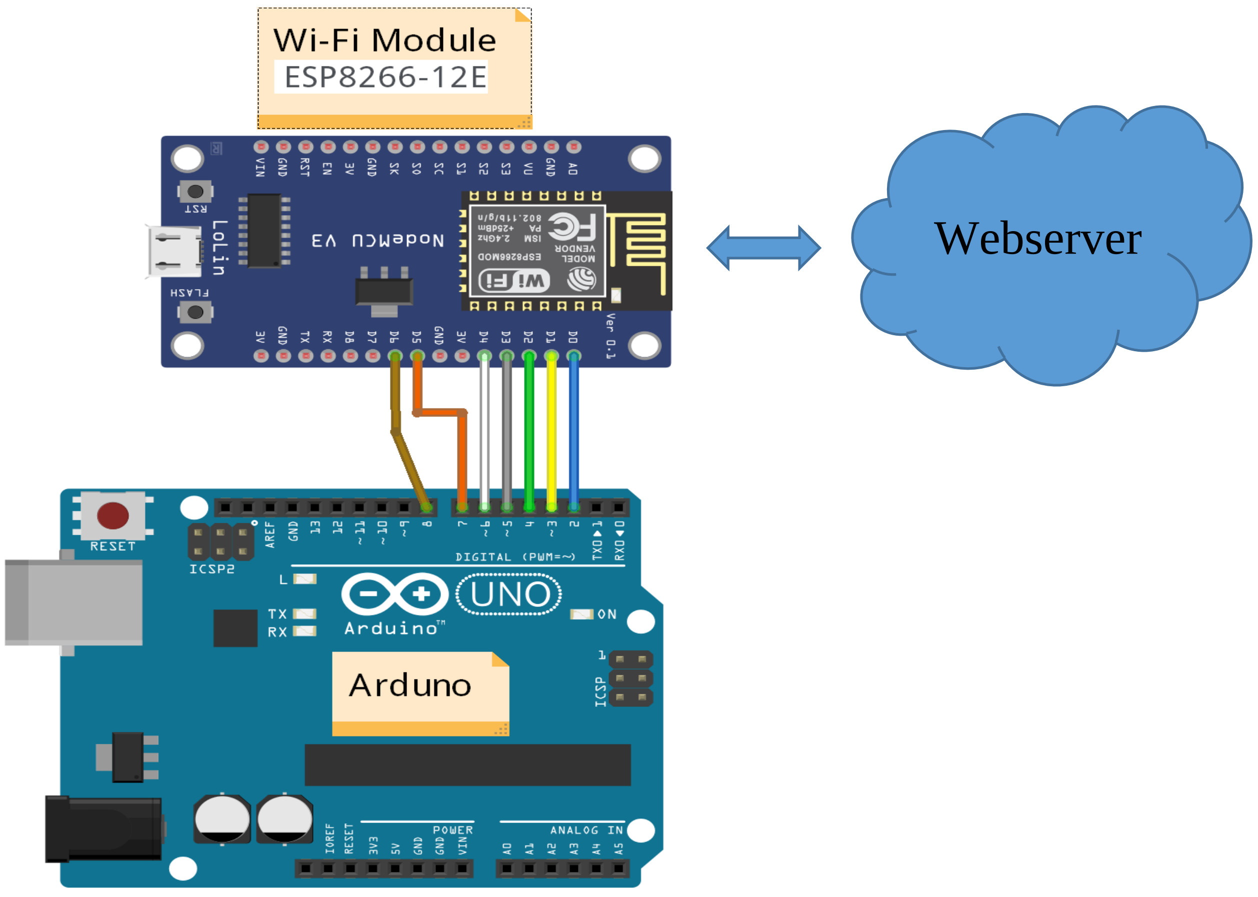


Fig. 3.7 Connection of Arduino Wi-Fi Module

3.8 Circuit Diagram

Here Circuit Diagram shows the all connect between Arduino and other modules. There are two power input which is 5v USB or 12v to the Arduino's barrel jack input then 12v power input for the two motor and water pump. All the connections from the modules are connected to the Arduino Expansion Shield which is connected on Arduino, It make the connection easy and 5v power supply can be drawn from it, therefore no need to use breadboard.

Connection diagram shown below.

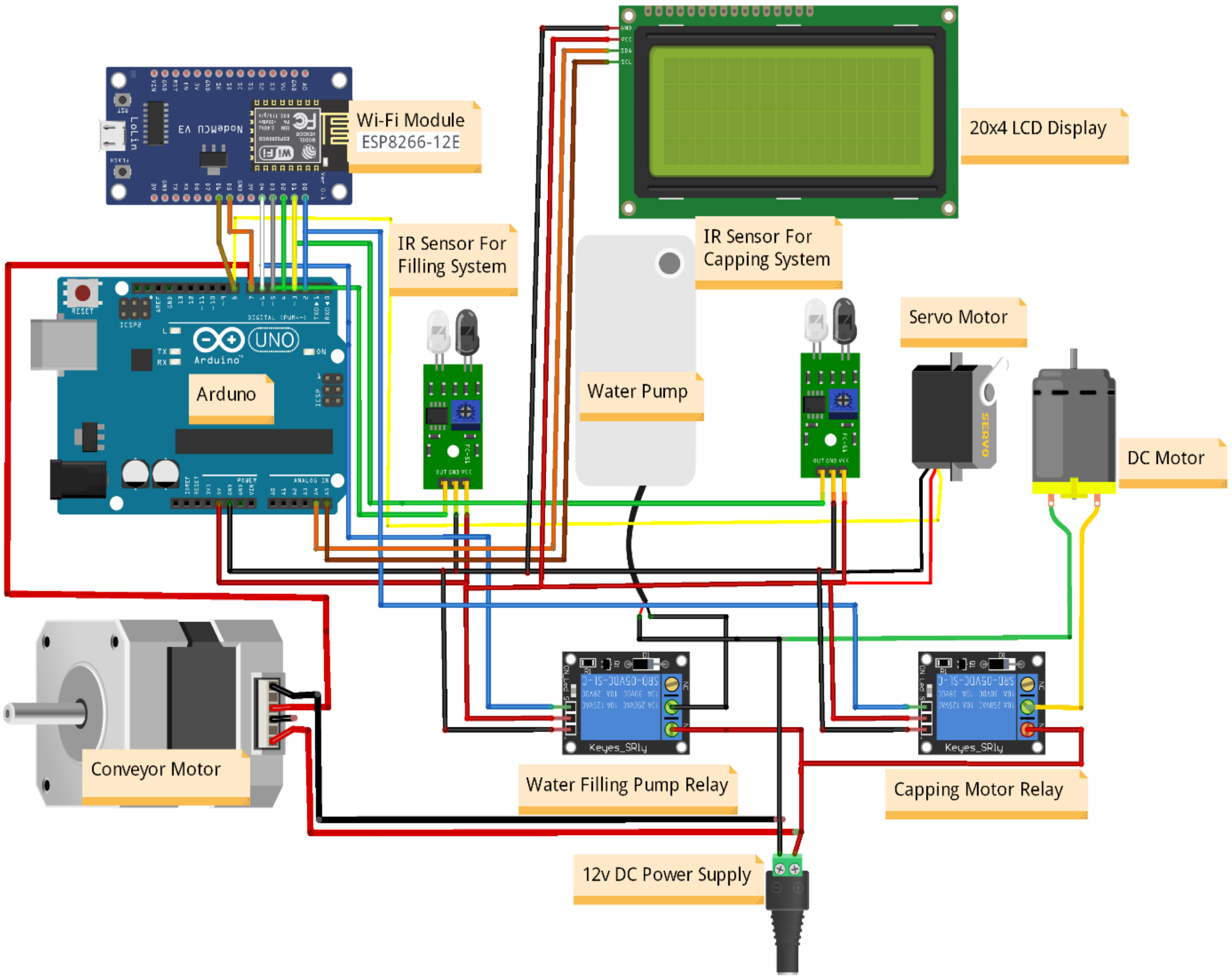


Fig. 3.8 Connection of Arduino with other modules

Chapter 4

Implementation of Arduino

4.1 Block Diagram

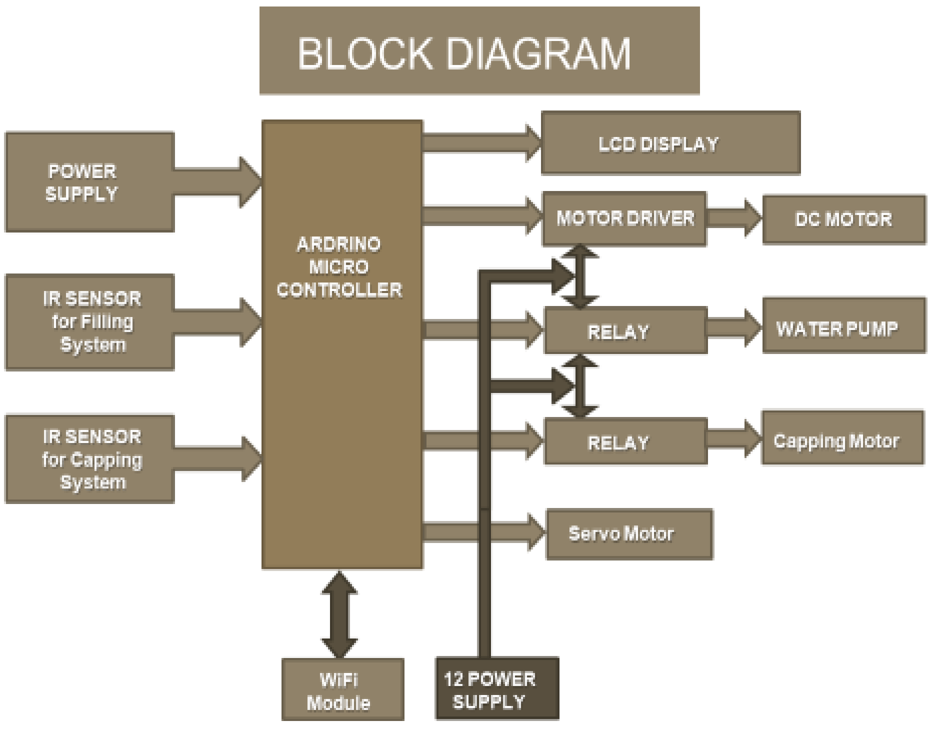


Fig. 4.1 Block Diagram

- **Power Supply:** There are two power supply in this system, First one is power supply of Arduino can be either 5v USB or 12v barrel jack connection and second one is used to power water pump and DC motor which is 12v
- **IR Sensor:** Both IR Sensor is to detect bottle in filling system and capping system, both are powered by Arduino itself. These are the only two inputs in this system.
- **LCD Display:** LCD Display is used to show the status of the process, it used I2C interface, It is a output device**.**
- **DC Motor:** There are two DC motor in this system, one is used to drive the conveyor which is a high torque motor and other one is used to rotate bottle cap. Both use 12v.
- **Relays:** There are two relays in this system, one is used to control water pump and another one is used to control DC motor of capping system.
- **Servo motor**: Servo motor is used to turn the capping motor up and down when it detect bottle.
- **Wi-Fi Module**: Wi-Fi Module is used to transmit and receive signal from Arduino which is used to display on webserver.

4.2 Working Flow Chart

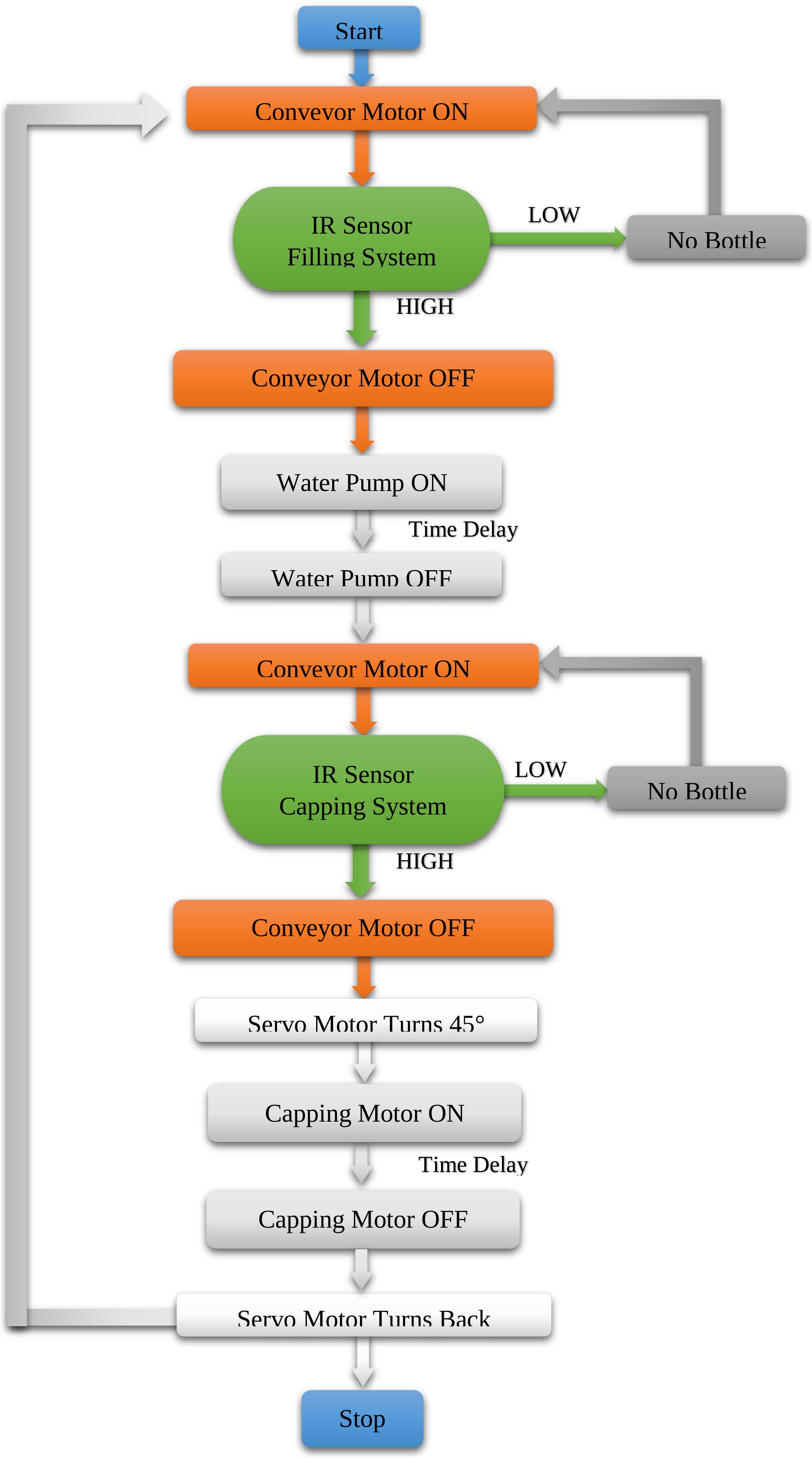
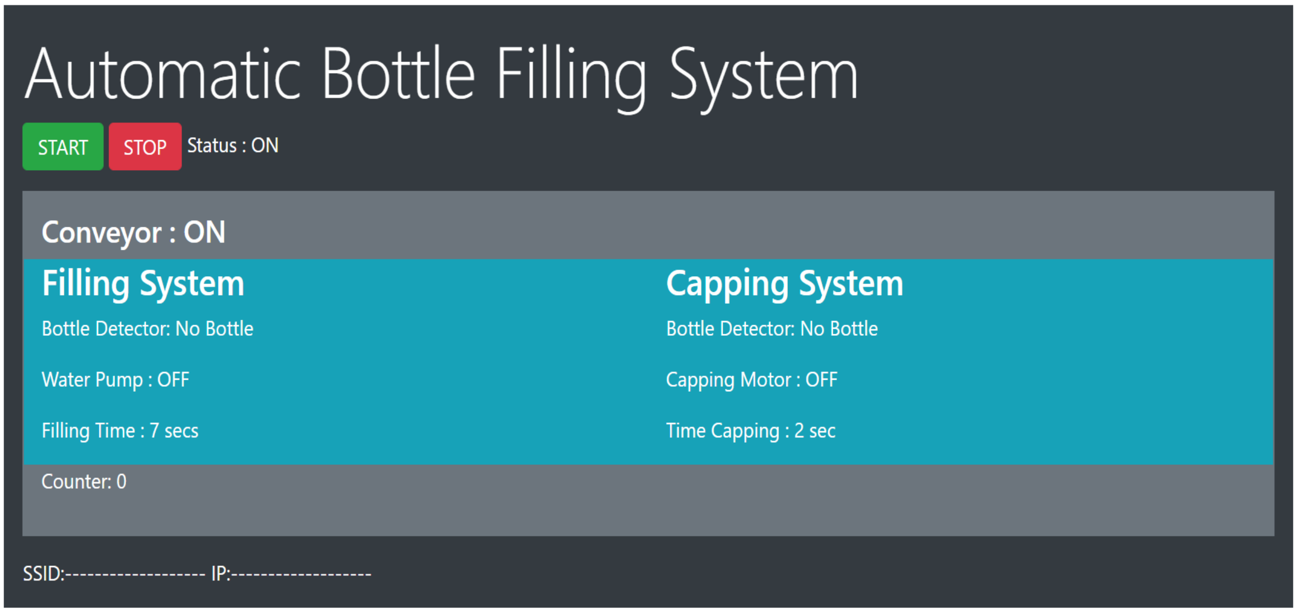


Fig. 4.2 Flow Chart

4.3 Human Machine Interface



Automatic Bottle Filling System

START STOP Status : ON

Conveyor : ON

Filling System

Bottle Detector: No Bottle

Water Pump : OFF

Filling Time : 7 secs

Capping System

Bottle Detector: No Bottle

Capping Motor : OFF

Time Capping : 2 sec

Counter: 0

SSID:------------------- IP:-------------------

Fig. 4.3 Human Machine Interface

Human-Machine Interface (HMI) is a user interface or dashboard that connects a person to a machine, system, or device. Here its displays the current status of modules and also it can control the process. It Wi-Fi based HMI where the signals are transmit from Arduino to Wi-Fi Module which create a webserver locally on a IP address, It can either host itself or connect to a network. It uses AJAX (Asynchronous JavaScript and XML) is a set of web development techniques using many web technologies on the client-side to create asynchronous web applications. With Ajax, web applications can send and retrieve data from a server asynchronously without interfering with the display and behaviour of the existing page i.e. the webpage can update itself than refreshing the page in which it reduce lag.

Chapter 5

Result

5.1 Bottle Filling Output

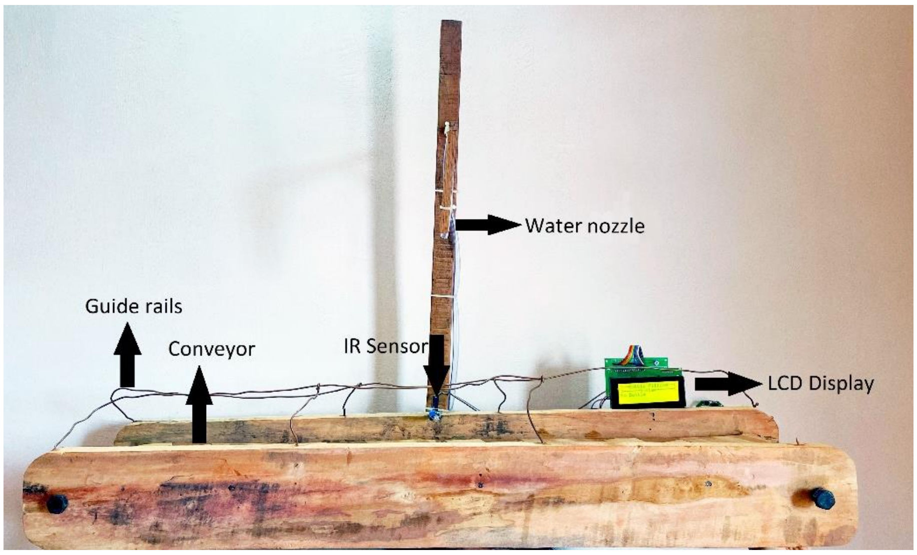


Fig. 5.1 Side view of Conveyor and filling system

Here the conveyor and its support is shown along with guide rails used to guide the bottle straight to the nozzle, IR sensor is placed in perpendicular to the water nozzle. LCD Display is placed at the edge of the conveyor platform.

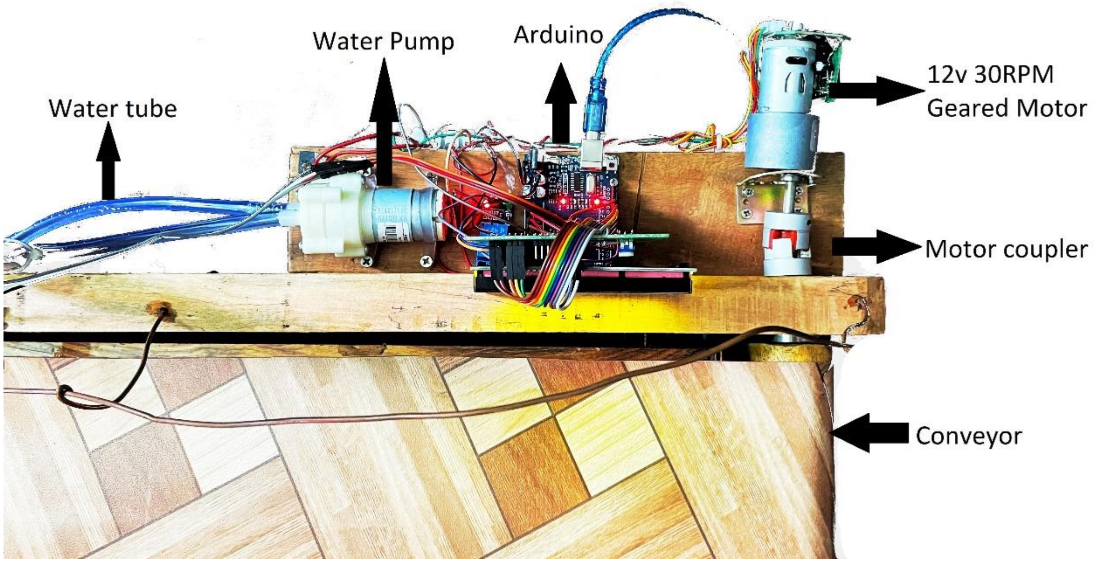


Fig. 5.2 Top view of Arduino and other components

Here top view of the system shows the water pump and tube connection, Geared DC motor is connected to conveyor with a coupler. Arduino is connected with all sensor modules wires and LCD.

5.2 Bottle Capping Output

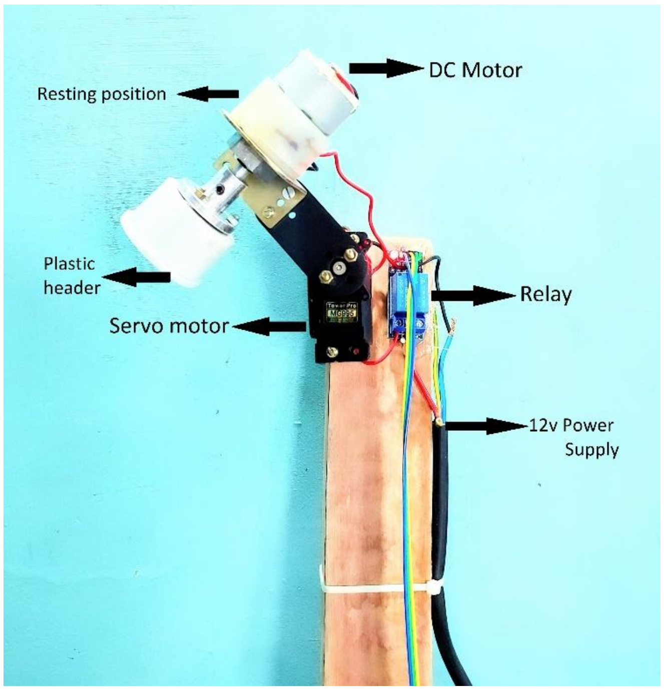


Fig. 5.3 Side view of capping system

Here the capping motor is in resting position i.e. 45°, holding the capping DC motor. Relay is in OFF state supply by 12v. Plastic header is a mould of bottle cap in which is used to rotate bottle cap.

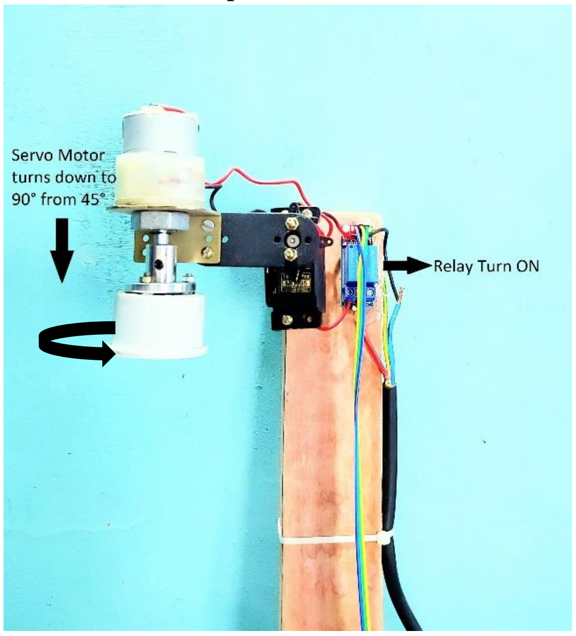
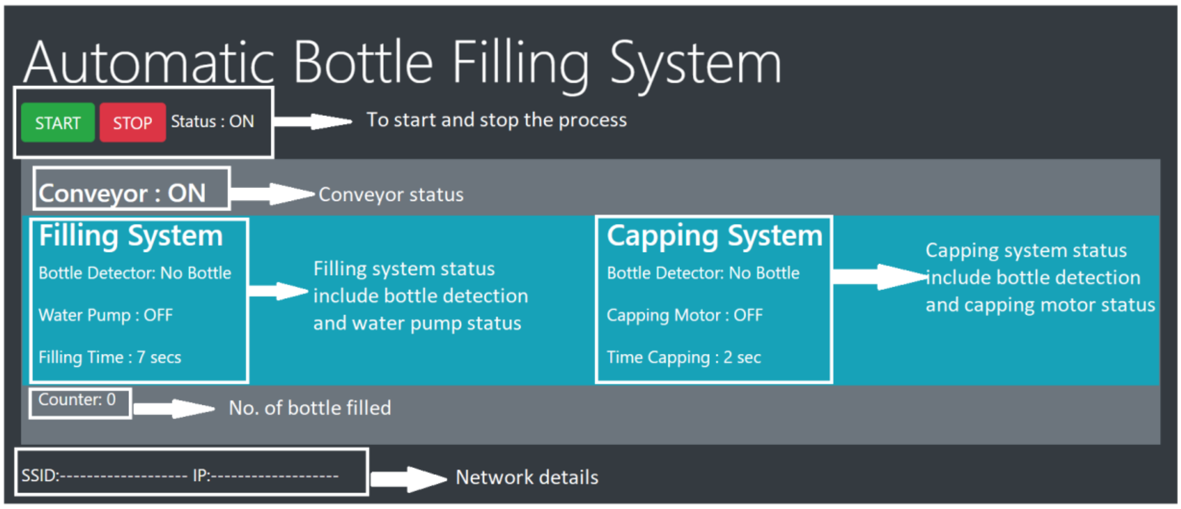


Fig. 5.4 Slide view of capping process

Here the servo motor turns down to 90°, relay is turn ON therefore the DC motor starts rotates.

5.3 HMI Output


Automatic Bottle Filling System

START STOP Status : ON

To start and stop the process

Conveyor : ON

Conveyor status

Filling System

Bottle Detector: No Bottle

Water Pump : OFF

Filling Time : 7 secs

Filling system status include bottle detection and water pump status

Capping System

Bottle Detector: No Bottle

Capping Motor : OFF

Time Capping : 2 sec

Capping system status include bottle detection and capping motor status

Counter: 0

No. of bottle filled

SSID:------------------ IP:------------------

Network details

Fig. 5.5 Web view of HMI

Thus the output of HMI of the system controlled and monitored by web after connected to the network. It consist of monitoring of system includes bottle detection, water pump and capping motor state ,time conveyor status and counter. Control of system is to start and stop the process with START/STOP buttons. Network details are also shown there.

Chapter 6

Conclusion and Future scope

The main objective of this project to develop an Automatic bottle filling system using User-friendly controlling system. By using Arduino, this can be successfully implemented. The present system will provide a great deal of applications in the field of automation, especially in mass production industries where there are larger number of components to be processed and handled in a short period of time and there is need for increased production.

More features can be added depending on the size, shape and weight of the bottles, filling operations can be implemented. Sorting and packing can also be added using Arduino.

References

[1] Mallaradhya H M and K R Prakash, Department of Industrial Automation Engineering, VTURO, Mysore, Karnataka, India , “AUTOMATIC LIQUID FILLING TO BOTTLES OF DIFFERENT”, issued on 14th july 2013.

[2] Jaymin Patel Department Of Physics and Electronics, Hemchandracharya North Gujarat, University, Patan, India, “PLC(Programmable Logic Controller) BASED AUTOMATIC BOTTLE FILLING”, may-June 2015

[3] T.Kalaiselvi and R.Praveena, Assistant professor, Easwari Engineering College, Chennai, “PLC Based Automatic Bottle Filling and Capping System With User Defined Volume Selection”, Issue 8, August 2012.

[4] Lokeshwar, Assistant Professor, R.N. College of Engineering & Management, Rohtak, India,“Implementation and Performance Analysis of Bottle Filling Plant Using Ladder Language”, Volume 3 Issue 7, July 2014

[5] Kala Meah Department of B.Sc. from Bangladesh University of Engineering and Technology.” AN AUTOMATED BOTTLE FILLING AND CAPPING PROJECT ”issued on may 2010.

[6] Ahmed Ullah Abu Saeed, Md. Al-Mamun and A.H.M. Zadidul Karim of International Journal of Scientific & Engineering Research,” Industrial Application of PLCs in Bangladesh” Volume 3, Issue 6, June-2012.

[7] RUHAIRI BIN ABDUL RAHIM Faculty of Electronic and Computer EngineeringUniversity Teknikal Malaysia Melaka “AUTOMATED MULTIPLE WATER FILLING(AMWF) MACHINE” On april 2009 .

[8] D.Baladhandabany, S.Gowtham, T.Kowsikkumar, P.Gomathi UG Student, Assistant Professor, Department of EEE, INFO Institute of Engineering,” PLC BASED AUTOMATIC LIQUID FILLING SYSTEM” IJCSMC, Vol. 4, Issue. 3,March 2015, pg.684 – 692.

[9] Circuit Digest https://circuitdigest.com/microcontroller-projects/automatic-bottle-filling-system-using-arduino

[10] Arduino UNO Guide https://www.arduino.cc/en/Guide/ArduinoUno

Appendix

HTML Code:

```
<!DOCTYPE html>
<html lang="en">
 <style>
.p1 {
font-family: 'Work Sans', sans-serif;
}
.p2 {
font-family: 'Roboto', sans-serif;
}
</style>
<body>
<div class="container p-3 my-3 bg-dark text-white">
<div class="container p-3 rounded my-3 bg-dark text-white">
   <h1 class="display-4">Automatic Bottle Filling System</h1>
 <p  class="p2" ><button type="button" class="btn btn-success" onclick="send(1)"
>START</button> <button type="button" class="btn btn-danger"
onclick="send(0)">STOP</button>   Status :<span id="state"></span> </p>

<div class="container p-3 my-3 bg-secondary text-white">
     <h4 class="p2">Conveyor : ON </h4>
  <div class="row">
    <div class="col-sm-6 bg-info rounded">
      <h3 class="p1" >Filling System</h3>
      <p class="p2">Bottle Detector: <span id="fillbd">No Bottle</span></p>
      <p class="p2">Water Pump    : <span id="fillwp">OFF</span></p>
      <p class="p2">Filling Time   : <span id="fillt">7 secs</span></p>
    </div>
    <div class="col-sm-6 bg-info rounded">
      <h3 class="p1">Capping System</h3>
      <p class="p2">Bottle Detector: <span id="capbd">No Bottle</span></p>
      <p class="p2">Capping Motor  : <span id="capm">OFF</span></p>
      <p class="p2">Time Capping   : <span id="captc">2 sec</span></p>
    </div>

  </div>
    <p  class="p2" >Counter: <span id="countr">0</span></p>
</div>
         <h  class="p2">IP:<span id="ipaddress">---------</span></h7>
</div>
     <script>
      setInterval(function() {getSensorData();}, 1000); // Call the update function every set
interval e.g. 1000mS or 1-sec
       function send(led_sts)
      {
```

```
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("state").innerHTML = this.responseText;
    }
  };
  xhttp.open("GET", "stater?state="+led_sts, true);
  xhttp.send();
}
 function getSensorData() {

  //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("capm").innerHTML = this.responseText;
    }
  };
  xhttp.open("GET", "capmr", true);
  xhttp.send();

  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("capbd").innerHTML = this.responseText;
    }
  };
  xhttp.open("GET", "capbdr", true);
  xhttp.send();
  //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("countr").innerHTML = this.responseText;}
  };
  xhttp.open("GET", "countrr", true);
  xhttp.send();
  //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("fillbd").innerHTML = this.responseText;}
  };
  xhttp.open("GET", "fillbdr", true);
  xhttp.send();
  //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
```

```
          if (this.readyState == 4 && this.status == 200) {
            document.getElementById("fillwp").innerHTML = this.responseText;}
        };
        xhttp.open("GET", "fillwpr", true);
        xhttp.send();
         //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        var xhttp = new XMLHttpRequest();
        xhttp.onreadystatechange = function() {
          if (this.readyState == 4 && this.status == 200) {
            document.getElementById("ipaddress").innerHTML = this.responseText;}
        };
        xhttp.open("GET", "ipaddressr", true);
        xhttp.send();
        //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        var xhttp = new XMLHttpRequest();
        xhttp.onreadystatechange = function() {
          if (this.readyState == 4 && this.status == 200) {
            document.getElementById("state").innerHTML = this.responseText;}
        };
        xhttp.open("GET", "stater", true);
        xhttp.send();
        //~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

      }
    </script>

<head>
<link rel="preconnect" href="https://fonts.gstatic.com">
<link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400&family=Work+San
s:wght@300&display=swap" rel="stylesheet">
  <title>Automatic Bottle Filling System</title>
  <meta charset="utf-8">
  <meta name="viewport"
 content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
  <script src='https://kit.fontawesome.com/a076d05399.js' crossorigin='anonymous'></script>
</head>
  </body>
</html>
```

Upload the code using Adurino IDE software
Code

```
#include <Servo.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd = LiquidCrystal_I2C(0x27, 20, 4);

int ir = 5;                              // Digtal PIN 2 for IR Sensor
int fillrelay = 6;                                // Digtal PIN 2 for fillrelay for Water
Pump
int filltime = 6000; //                        Filling Time (ms)
int irc = 3;     //IR sensor pin for capping
int caprelay = 2;    // Analog PIN 5 for relay for Capping Motor
int servopin = 8;
Servo myservo;  // create servo object to control a servo
// twelve servo objects can be created on most boards
int count=0;
int pos = 45;    // vari  able to store the servo position
int captime = 1800; //                            Capping Time (ms) amount of time to
rotate cap

void setup()
{

  lcd.begin(20, 4);
  pinMode(ir,INPUT);                            // Input of IR Sensor
  pinMode(fillrelay,OUTPUT);                             //Output of fillrelay for Water
Pump
  Serial.begin(9600);
  myservo.attach(servopin);  // attaches the servo on pin 4 to the servo object
  pinMode(irc,INPUT);                            // Input of IR Sensor For Capping
  pinMode(caprelay,OUTPUT);                              //Output of Relay for Capping
Motor
  Serial.begin(115200);
  digitalWrite(caprelay,HIGH);
  Serial.println("Bottle Filling System");
  lcd.setCursor(0, 0);
  lcd.print(">--Bottle Filling--<");
  lcd.setCursor(0, 1);
  lcd.print(">------System------<");
  digitalWrite(fillrelay,HIGH);
   //initiallize default ppm values
  for(int i=0; i<CHANNEL_NUMBER; i++){
     ppm[i]= CHANNEL_DEFAULT_VALUE;
  }

  pinMode(sigPin, OUTPUT);
  digitalWrite(sigPin, !onState);  //set the PPM signal pin to the default state (off)

  cli();
  TCCR1A = 0; // set entire TCCR1 register to 0
  TCCR1B = 0;
  OCR1A = 100;  // compare match register, change this
  TCCR1B |= (1 << WGM12);  // turn on CTC mode
  TCCR1B |= (1 << CS11);  // 8 prescaler: 0,5 microseconds at 16mhz
  TIMSK1 |= (1 << OCIE1A); // enable timer compare interrupt
  sei();

}

void loop() {

  if (digitalRead(ir) == LOW)
{
  filling();                              //For Filling Process
   delay(1500);
}

else{

   nofilling();                              //No Bottle
     }
 if (digitalRead(irc) == LOW)
  {
    capping();               // capping process
    counter();
   delay(1500);

  }
  else
  {
    nocapping();              // no bottle
  }

}

ISR(TIMER1_COMPA_vect){  //leave this alone
  static boolean state = true;

  TCNT1 = 0;

  if (state) {  //start pulse
    digitalWrite(sigPin, onState);
    OCR1A = PULSE_LENGTH * 2;
    state = false;
  } else{  //end pulse and calculate when to start the next pulse
    static byte cur_chan_numb;
    static unsigned int calc_rest;

    digitalWrite(sigPin, !onState);
    state = true;

    if(cur_chan_numb >= CHANNEL_NUMBER){
      cur_chan_numb = 0;
      calc_rest = calc_rest + PULSE_LENGTH;
      OCR1A = (FRAME_LENGTH - calc_rest) * 2;
      calc_rest = 0;
    }
    else{
      OCR1A = (ppm[cur_chan_numb] - PULSE_LENGTH) * 2;
      calc_rest = calc_rest + ppm[cur_chan_numb];
      cur_chan_numb++;
    }
  }
}

void nofilling()                              //No Bottle
{

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(">--Bottle Filling--<");
  lcd.setCursor(0, 1);
  lcd.print(">------System------<");
  lcd.setCursor(0, 2);
  lcd.print("Filling - No Bottle");
 Serial.println("Filling -No Bottle");
  delay(200);                                 //Refreshing
 PULSE_LENGTH = 600;

}
void filling()                             //For Filling Process
{
  PULSE_LENGTH = 1510;
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(">--Bottle Filling--<");
  lcd.setCursor(0, 1);
  lcd.print(">------System------<");
  lcd.setCursor(0, 2);
  Serial.println("Filling - Start");
  lcd.print("Filling Started");
  digitalWrite(fillrelay,LOW);
  delay(filltime);
  bmove();
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(">--Bottle Filling--<");
  lcd.setCursor(0, 1);
  lcd.print(">------System------<");
  lcd.setCursor(0, 2);
  Serial.println("Filling - Completed");
  lcd.print("Filling Completed");
  digitalWrite(fillrelay,HIGH);


 }
 void bmove()
 {
      PULSE_LENGTH = 600;
 }
 void capping()                              //For Capping Process
{

   myservo.write(95);              // tell servo to go to position in variable
       Serial.println("servo start");
   delay(750);
   digitalWrite(caprelay,LOW);
         Serial.println("relay start");
   delay(captime);
   digitalWrite(caprelay,HIGH);
         Serial.println("relay stop");
   delay(750);
   myservo.write(45);              // tell servo to go to position in variable
         Serial.println("servo back");
   delay(500);


}
void nocapping()                              //For No bottle for Capping
{
   myservo.write(45);              // tell servo to go to position in variable
   delay(500);
}
 void counter()
 {
     count++;
     String StrUno = "Counter: ";
     String StrDos = StrUno + count;
     Serial.println(StrDos);
 }
```