1.0 INTRODUCTION

The project is designed to develop a Robotic Rover vehicle with a soft holding gripper. It can be used for pick and place of an object. It can sense whether the object picked is metallic or non-metallic. It can also detect the rover is in gaseous surrounding or not (LPG, I-Butane, Methane, Hydrogen, Alcohol, smoke). It can also tell the temperature and humidity level. The robotic vehicle is controlled by an android application for remote operation.

At the transmitting end using an Android application device, commands are sent to the receiver to control the movement of the robot to move forward, backward, and left or right. The android application device transmitted acts as a remote control that has the advantage of a certain range, while the receiver end is fed to the micro controller to drive DC motors via motor driver IC for necessary work. Remote operation is achieved by any smart-phone/Tablet, etc. The main advantage of this robot is its holding arm, which is designed to avoid extra pressure on the suspected object for safety reasons. A Rover is the one which is used to pick up an object and place it in the desired location. It is an articulated Robotic Rover, which contains a fixed robot with 3 vertical axes and rotary arms.

Manipulator is the main body of the vehicle which consisting of several joints and links

2.0 PROECT BRIEF

AIM

To design and build a Robotic Rover.

OBJECTIVES

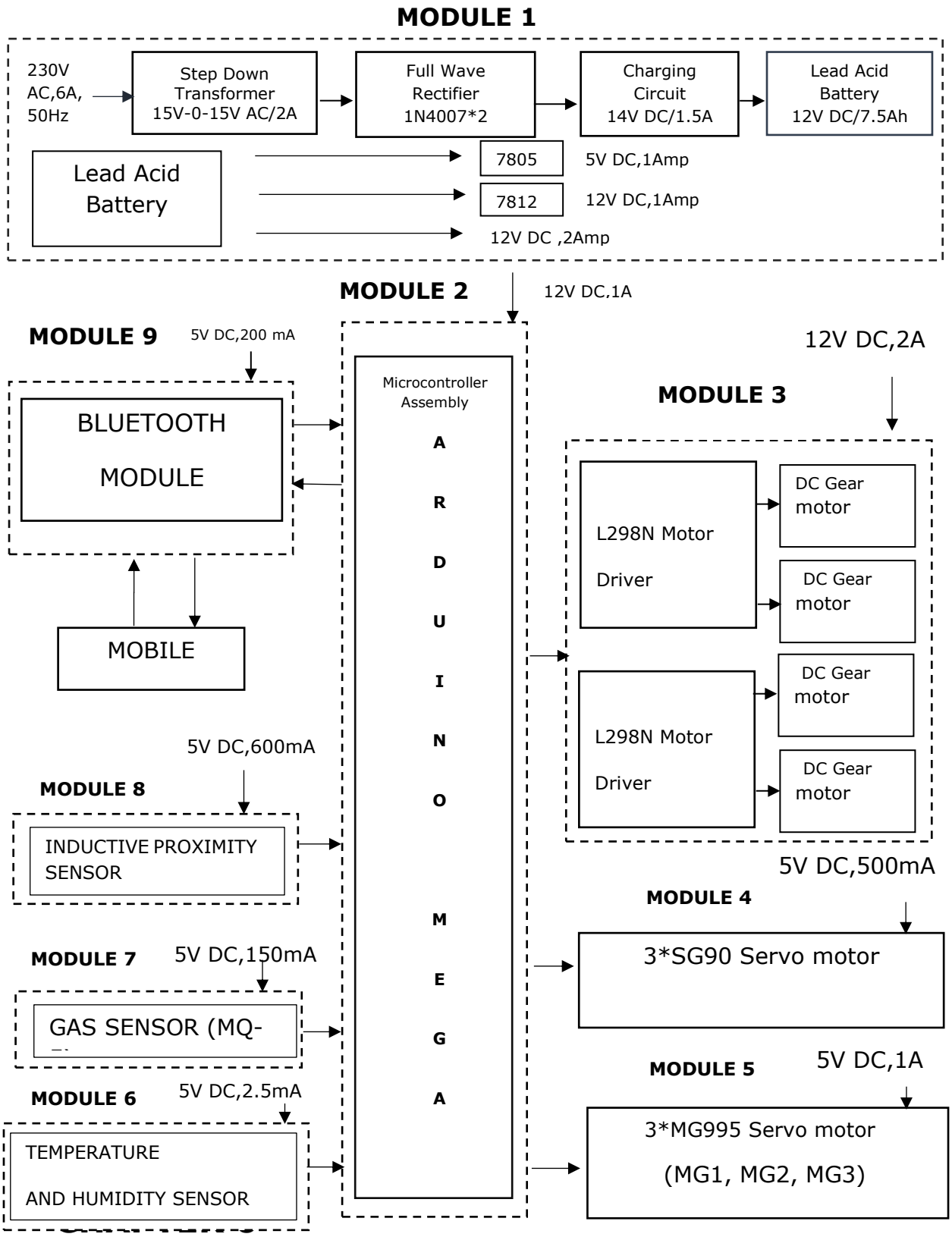
- To design a speed control system for the Robotic Rover
- Pick and place the objects using manipulator
- To implement wireless android application to control the Robotic Rover
- To detect gases in surrounding (LPG, I-Butane, Methane, Hydro gen, Alcohol)
- To display temperature and humidity level
- To sense whether the object is metallic or non-metallic

3.0 PROJECT TARGET

To create a wireless based Robotic Rover which can move an object from one place to another place and also to an unsustainable condition or environment.

- To Reduce the human effort
- Pick and Place the required component
- Easy to implement
- Low maintenance

4.0 MODULAR BLOCK DIAGRAM



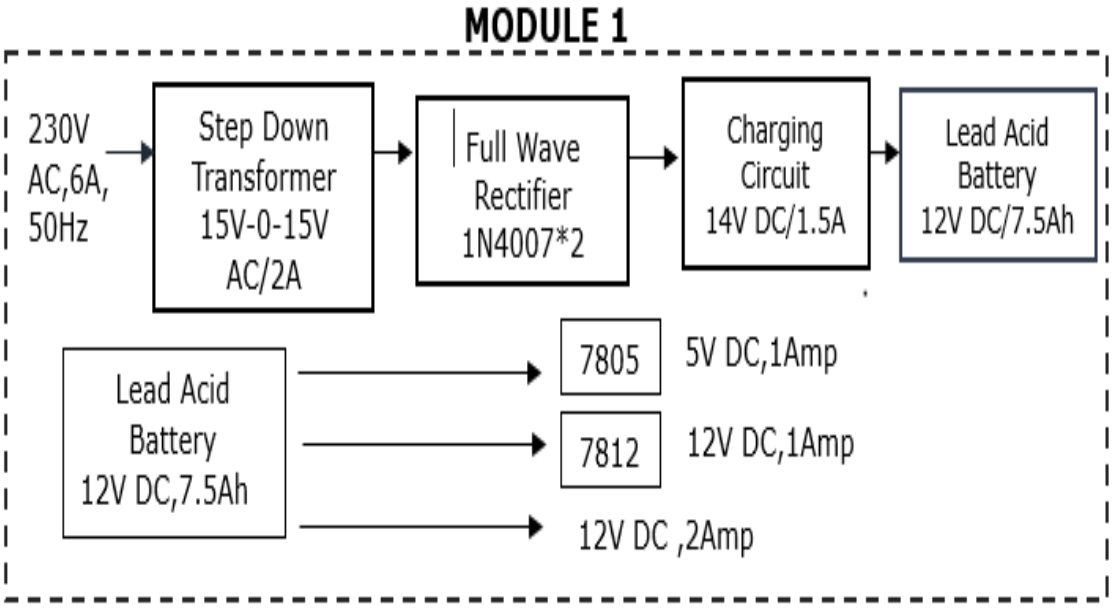
5.0 POWER SUPPLY & CHARGING CIRCUIT

5.1 DESCRIPTION

Step down transformer gives 15V-0-15V/2A AC to the center tap full wave rectifier formed by diodes D1, D2. Rectifier will convert 30V AC to 19V DC. Capacitor C1 is introduced to filter the ripples of converted DC, so it used as filter as well as backup capacitor. To remove the ripple voltage. The Rectifier is connected to the capacitors 1000uf,50V and they are connected to regulator LM317 for regulating the voltage to 13.5v for charging the battery

and a freewheeling is used to stop the back current from the battery to protect the circuit. In this circuit we are used auto cutoff system to overcome the problem of overcharging.

5.2 BLOCK DIAGRAM



BLOCK DIGRAM DESCRIPTION

INPUT

The input 230V/50Hz is given to the transformer.

STEPDOWN TRNSFORMER

A Step-Down Transformer is a device which converts high primary voltage to a low secondary voltage. In a Step-Down Transformer, the primary winding of a coil has more turns than the secondary winding.

CENTER-TAP FULL WAVE RECTIFIER

A Full Wave Rectifier is a circuit, which converts an ac voltage into a pulsating dc voltage using both half cycles of the applied ac voltage. It uses two diodes of which one conducts during one half cycle while the other conducts during the other half cycle of the applied ac voltage.

FILTER

The pulsating output of the rectifiers has an average DC value and an AC portion that is called ripple voltage. Filter capacitors reduce the amount of ripple voltage to a level that is acceptable.

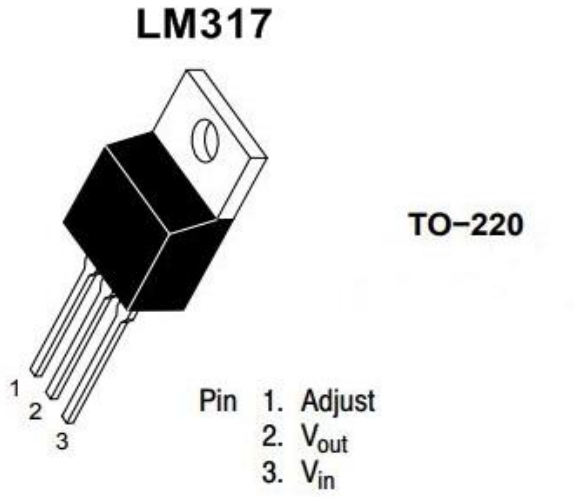
REGULATOR

We are using regulator ICs to maintain a constant output voltage in a power supply circuit. We are using regulator IC LM317 for getting 13V respectively through variable resistor to this we are connected to the battery.

For 12V,1Amp supply we are using 7812 for getting 12V respectively, and for 5V,1Amp supply we are using 7805 for getting 5V respectively.

LM317

The LM317 is an adjustable 3−terminal positive voltage regulator capable of supplying in excess of 1.5 A over an output voltage range of 1.2 V to 37 V. This voltage regulator is exceptionally easy to use and requires only two external resistors to set the output voltage.



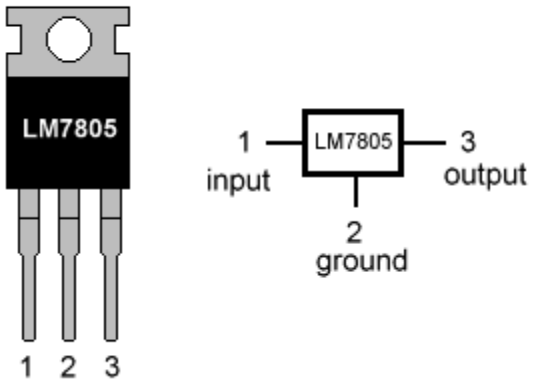
Features

- Output Voltage Range Adjustable From 1.25 V to 37 V
- Output Current Greater Than 1.5 A
- Internal Short-Circuit Current Limiting
- Thermal Overload Protection
- Output Safe-Area Compensation

LM7805

7805 is a 5V fixed three terminal positive voltage regulator IC. The IC has features such as safe operating area protection, thermal shut down, internal current limiting which makes the IC very rugged. Output currents up to 1A can be drawn from the IC provided that there is a proper heat sink.

LM7805 PINOUT DIAGRAM

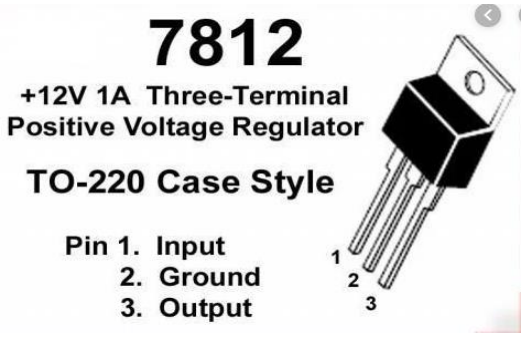


Features

- 5V Positive Voltage Regulator
- Minimum Input Voltage is 7V
- Maximum Input Voltage is 25V
- Operating current (I_Q) is 5mA
- Internal Thermal Overload and Short circuit current limiting protection is available.
- Junction Temperature maximum 125 degree Celsius
- Available in TO-220 and KTE package

LM7812

7812 is a 12V fixed three terminal positive voltage regulator IC. The IC has features such as safe operating area protection, thermal shut down, internal current limiting which makes the IC very rugged. Output currents up to 1A can be drawn from the IC provided that there is a proper heat sink.



Features

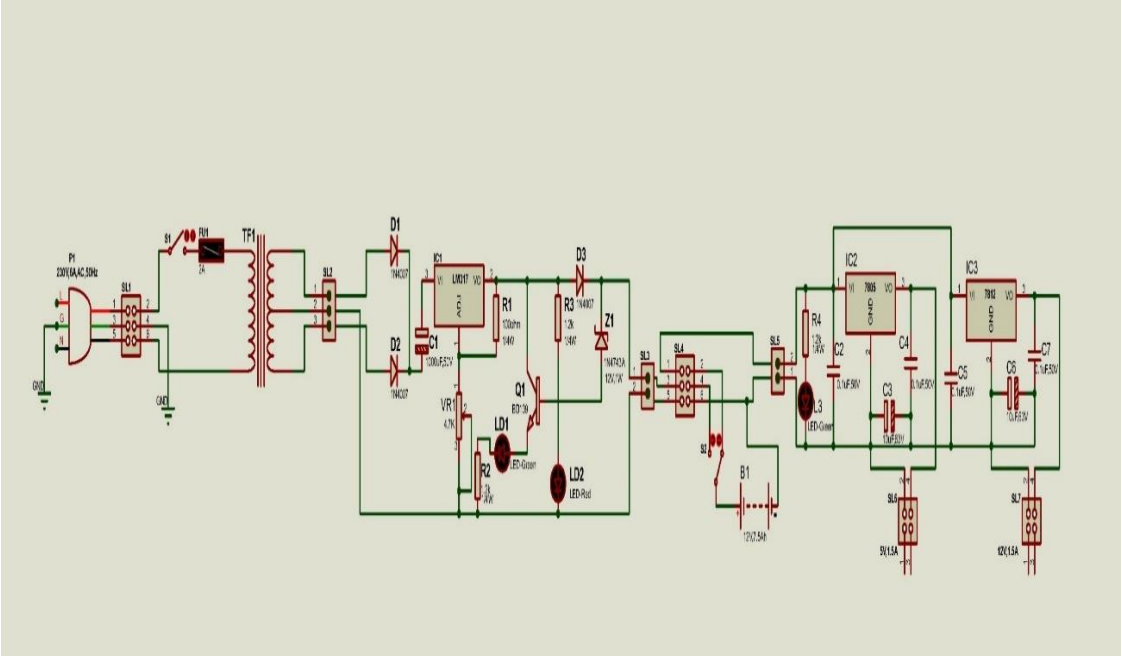
- Fixed voltage linear regulator
- 14-35V input voltage range
- 12V fixed output voltage
- 1A continuous current
- TO-220 package

Output:

The output of power supply is 12VDC/1A.

The output of power supply is 5 VDC/1A.

5.3 SCHEMATIC DIAGRAM



5.4 DESIGN

CHARGING CIRCUIT REQUIREMENTS:

Voltage = 15V DC

Current = 1A

TRANSFORMER SPECFICATION:

Voltage =15V-0-15V AC

Current =2A

For Center Tap Full Wave Rectifier:

VDC = Pulsating DC Voltage (Volts)

Vmax = V peak Voltage (Volts)

VDC = 2xVmax/π

VDC =0.636xVmax

VDC =0.636x30

VDC=19.108V

As per the data sheet the Diode is 1N4007

Iout = Output Current (Amp)

Iout = 1A

In center tap full wave rectifier the frequency gets double

F=2 x F

Value of the capacitor:

C = Iout/F x Vripple

Vripple =20% of VDC

C = 1Amp/100HzX0.95

C = 950 µf

C =~ 1000 µf

Working Voltage of Capacitor:

V1 = Operating /Working Voltage

V1 =DC Voltage +25% (extra 25% for reduce the ripple voltage)

= 19+4.75

= 23.75

= 24V

Capacitor standard Voltage we are consider 50V

V1=50V

VOLTAGE REGULATOR LM317:

Consider:

Vin = 19V

Vout = Output Voltage(Volts)

R1=100 Ω

R2=70 Ω

Vout = 1.25 (R1/R1+R2) Vin

= 1.25(100Ω/100 Ω +70 Ω) Vin

= 1.25(100Ω/100 Ω +70 Ω) X19V

= 1.25 X 11,1764

= 13.97V

Vout=~14V

For R2, R3, R4 (Ohm's Law)

R = V/I

R = 14/20mA

R = 0.7 X 1000

R = 700 Ω

Power Rating for Resistor:

I = 20mA

V = 14V

P = V x I

= 14(20mA)

= 280 x 10^-3

= 0.28 W

P =~0.25W

For Heatsink:

PD = (Vin-Vout) X I out

PD = power dissipation (Watt)

Vin = Input voltage (Volts)

Vout = output voltage (Volts)

Iout = output current (Amp)

PD = (19-14V) X 1Amp

= 5W

θJA = Total junction to ambient thermal resistance (°C/W)

TJ =Junction temperature (°C)

TA = Ambient temperature (°C)

θJA (total) = TJ-TA/PD

= 125°C -23 °C/5W

= 102/5W

θJA=20.4 °C/W

5.5 BILL OF MATERIALS

MODULES

Sl.no	Item description	Symbol	Qty	Price(₹)
Module 1 (Power Supply & Charging Circuit)	(i) 3 core patch chord 230V AC,6A,50Hz	P1	1	50
	(ii) Connector 6 pin Poly carbonate,10A	SL1, SL4	2	12
	(iii) Connector 3 pin Poly carbonate,10A	SL2	1	3
	(iv) Connector 2 pin Poly carbonate,5A	SL3, SL5, SL6, SL7	4	8
	(v) Switch (Rocker SPST) 230V AC,6A	S1	1	20
	(vi)Fuse & Fuse holder (Glass Catridge Fuse 2A, Fuse holder round screw,6A, AC)	FU1	1	50

	(vii) Transformer Center Tap (15-0-15)V AC,2A	TF1	1	250
	(viii) Diode: 1N4007 Axial Type	D1, D2, D3	3	9
	(ix)Electrolytic Capacitor (50V,1000uf) Radial Type	C1	1	8
	(x) Carbon film Resistor (10k,1/4W CFR, ±5%)	R1, R2, R3, R4	4	4
	(xi) Trimpot (10k,1/2W)	VR1	1	10
	(xii)LM317	IC1	1	15
	(xiii) Heat sink: E3A-T220-25E	-	3	45
	(xiv) NPN Transistor: BD139	Q1	1	10
	(xv) Zener Diode: 1N4742 Axial Type	Z1	1	2
	(xvi) LED 5mm radial type (3V,20mA)	LD1, LD2, LD3	3	10

	(xvii) Lead Acid Battery (12V,7.5Ah)	B1	1	700
	(xviii)Ceramic Capacitor (104) Radial Type (50V,0.1uf)	C2, C4, C5, C7	4	4
	(xix) Electrolytic Capacitor (63V,10uf) Radial Type	C3, C6	2	12
	(xx) DC +ve Regulator 7805,1A	IC2	1	10
	(xxi) DC +ve Regulator 7812,1A	IC3	1	10
	(xxii) Toggle switch: SPDT, SDT-106D-0-1(6A,125VAC)	S2	1	30
	TOTAL			1282

6.0 MICROCONTROLLER ASSEMBLY MODULE

6.0 ARDUINO MEGA 2560

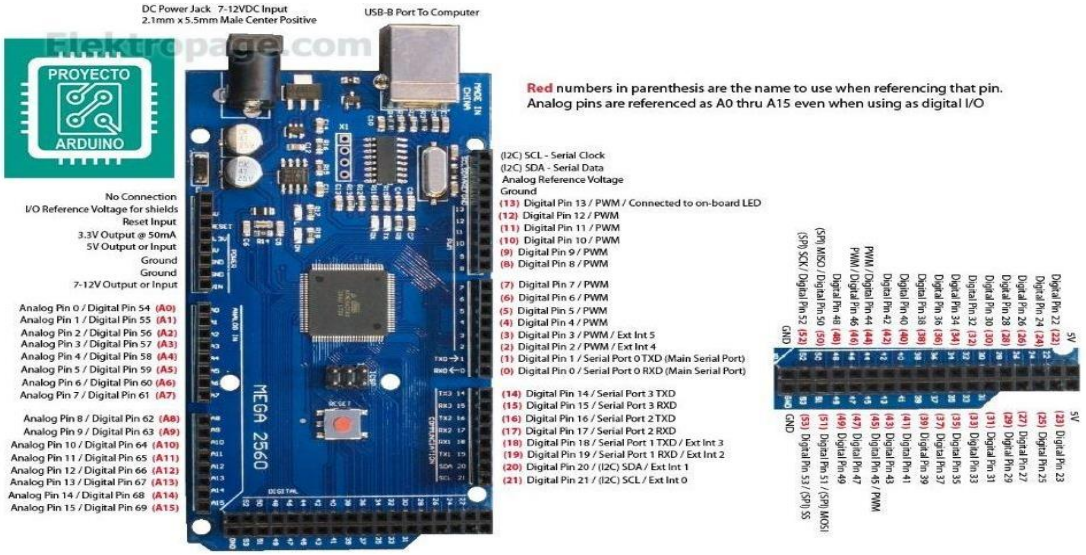
6.1 DESCRIPTION

The Arduino MEGA 2560 is designed for projects that require more I/O lines, more sketch memory and more RAM. With 54 digital I/O pins, 16 analog inputs and a larger space for your sketch it is the recommended board for 3D printers and robotics projects. The Arduino Mega 2560 is programmed using the software (IDE) our Integrated Development Environment common to all our boards and running both online and offline.

6.2 FEATURES OF ARDUINO MEGA 2560

- The ATmega2560 is a Microcontroller
- The operating voltage of this microcontroller is 5volts
- The recommended Input Voltage will range from 7volts to 12volts
- The input voltage will range from 6volts to 20volts
- The digital input/output pins are 54 where 15 of these pins will supply PWM o/p.
- Analog Input Pins are 16
- DC Current for each input/output pin is 40 mA
- DC Current used for 3.3V Pin is 50 mA
- Flash Memory like 256 KB where 8 KB of flash memory is used with the help of bootloader
- The static random access memory (SRAM) is 8 KB
- The electrically erasable programmable read-only memory (EEPROM) is 4 KB
- The clock (CLK) speed is 16 MHz
- The USB host chip used in this is MAX3421E

6.3 PIN CONFIGURATION:



PIN DIAGRAM EXPANATION:

POWER PINS:

VIN: The input voltage to the Arduino board when it's using an external power source can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
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: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
GND: Ground pins.

MEMORY:

The ATmega2560 has 256 KB of flash memory for storing code of which 8 KB is used for the bootloader, 8 KB of SRAM and 4 KB of EEPROM

INPUT AND OUTPUT

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor of 20-50 kOhms. In addition, some pins have specialized functions

• **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

• **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5),19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** 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: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.

• **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).**
These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.

• **LED: 13.** 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.

• **I2C: 20 (SDA) and 21 (SCL).** Support I2C (TWI) communication using the Wirelibrary.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution. By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

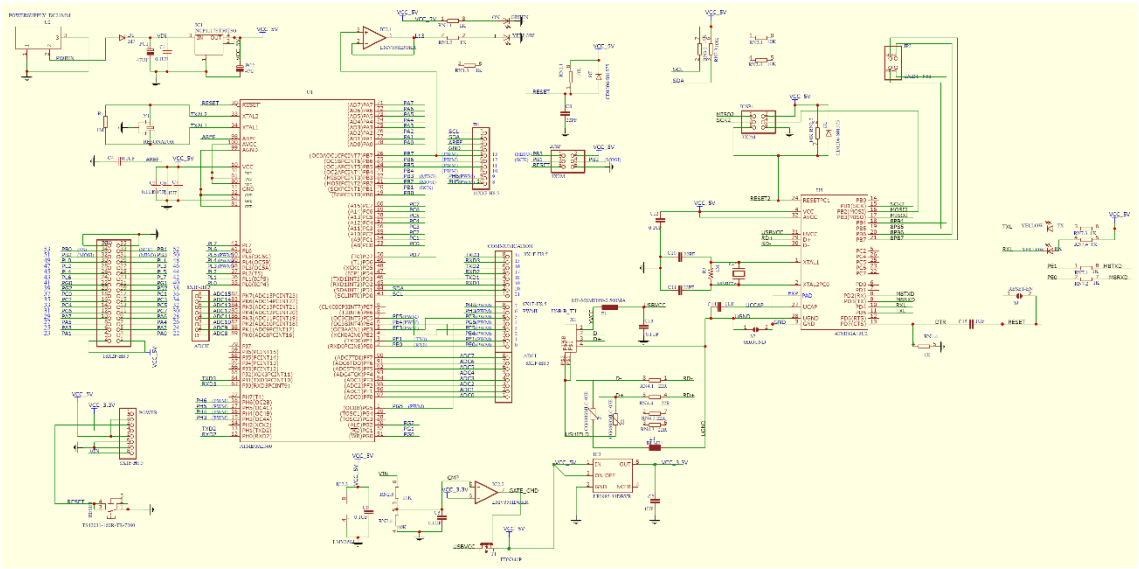
There are a couple of other pins on the board:

• **AREF.** Reference voltage for the analog inputs. Used with analogReference().

• **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset

button to shields which block the one on the board.

6.4 SCHEMATIC DIAGRAM:



6.5 BILL OF MATERIAL

SL NO	Item description	symbol	Qty	Price(₹)
1	ATMEGA2560	U1	1	27
2	POWERSUPPLY_DC21MM	U2	1	19
	47UF	PC1	1	3
4	0.1UF	C1, C4, C5, C6, C7, C3, C9, C12, C13, C15	10	7
5	RESONATOR	Y1	1	5
6	NCP1117ST50T3G	IC1	1	6
7	1M	R1, R3	2	3
8	M7	D1	1	6
9	18X2F-H8.5	XIO	1	6
10	8X1F-H8.5	POWER, ADCH, COMMUNICATION, PWML, ADCL	5	20
11	TS42031-160R-TR-7260	RESET	1	5
12	10X1F-H8.5	JP1	1	6

13	CD1206-S01575	D3, D2	2	8
14	22PF	C8, C10, C11	3	3
15	3X2M	ICSP, ICSP1	2	8
16	1UF	C2, C14	2	2
17	LP2985-33DBVR	IC3	1	9
18	FDN340P	T1	1	50
19	2X2M – NM	JP2	1	33
20	16MHZ	Y2	1	19
21	USB-B_TH	X1	1	80
22	MF-MSMF050-2 500MA	F1	1	
23	CG0603MLC-05E	Z1, Z2	2	49
24	BLM21	L1	1	12
25	SJ	GROUND, RESET-EN	2	30
26	ATMEGA16U2	U5	1	16
27	47U	PC2	1	17

28	GREEN	ON	1	2
29	YELLOW	L, TX, RX	3	2
30	1u	C16, C18	2	1
31	1u	C17, C19	2	2
32	1N4448	D4, D5	2	8
33	1k	R2, R4	2	2
34	Оптопара	U3, U4	2	26
35	BC847AWT1G	Q1, Q2	2	50
36	LMV358IDGKR	IC2	1	69
37	1K	RN1, RN7	2	12
38	10K	RN2, RN3	2	12
39	22R	RN4	1	12
TOTAL			700	

7.0 JOHNSON MOTOR WITH L298N MOTOR DRIVER

7.1 DESCRIPTION

L298N MOTOR DRIVER

This dual bidirectional motor driver, is based on the very popular L298 Dual H-Bridge Motor Driver Integrated Circuit. The circuit will allow you to easily and independently control two motors of up to 2A each in both directions. It is ideal for robotic applications and well suited for connection to a microcontroller requiring just a couple of control lines per motor. It can also be interfaced with simple manual switches, TTL logic gates, relays, etc. This board equipped with power LED indicators, on-board +5V regulator and protection diodes.

FEATURES

- Input Voltage: 3.2V~40Vdc. Brief Data:
- Driver: L298N Dual H Bridge DC Motor Driver
- Power Supply: DC 5 V - 35 V
- Peak current: 2 Amp
- Operating current range: 0 ~ 36mA
- Low: -0.3V ≤ Vin ≤ 1.5V.
- High: 2.3V ≤ Vin ≤ Vss.
- Maximum power consumption: 20W (when the temperature T = 75 ℃).

JOHNSON MOTOR:

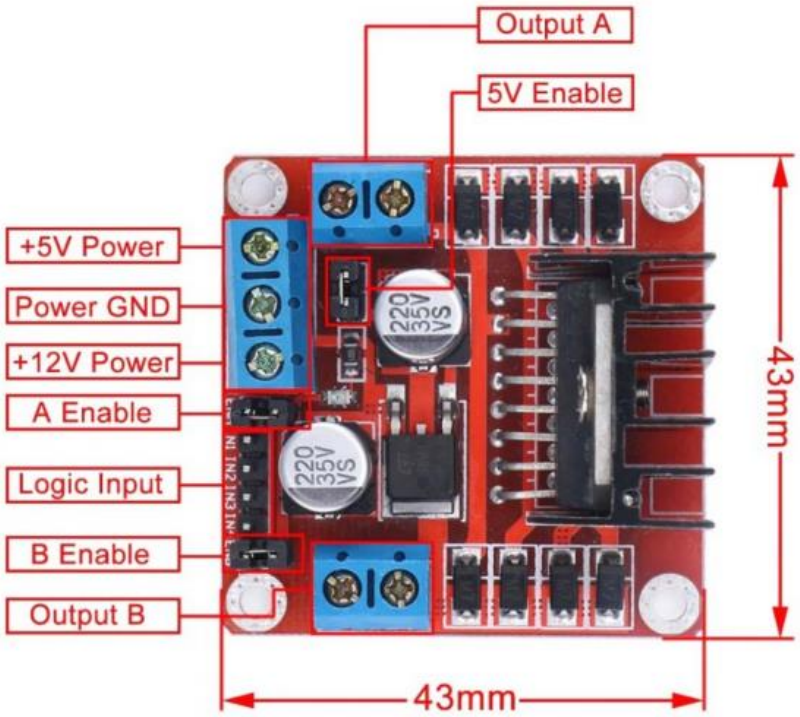
It is a simple DC motor featuring metal gearbox for driving the shaft of the motor, so it is a mechanically commutated electric motor which is powered from DC supply. The Johnson Geared Motors are known for their compact size and massive torque-speed characteristic.

The Johnson Motor comes with side shaft also known as an off-centered shaft and six M3 mounting holes. The shaft of the motor equips metal bushes which makes these DC gear motors Shaft wear resistant. The shaft of the motor has a hole for better coupling.

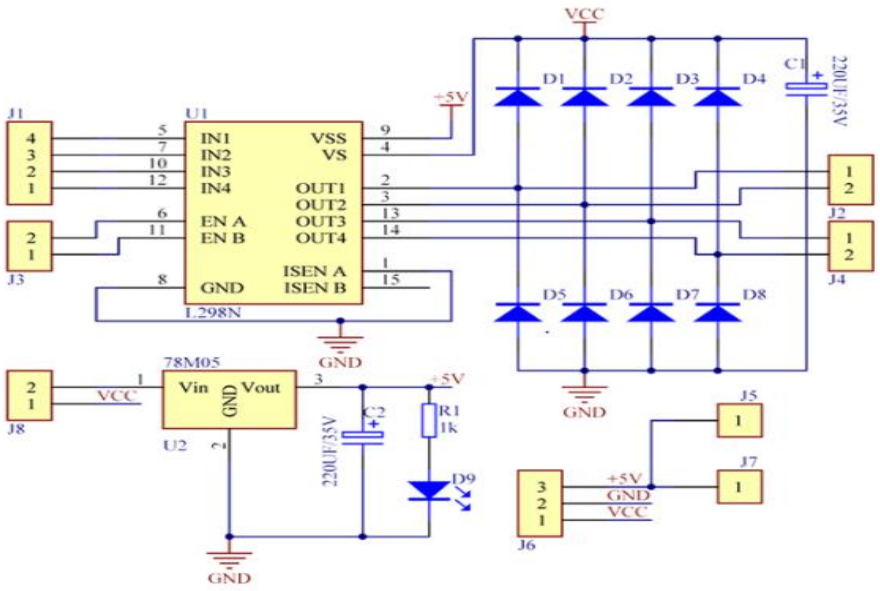
FEATURES:

- 10RPM 12V DC motors with Metal Gearbox and Metal Gears
- 18000 RPM base motor
- 6mm Dia shaft with M3 thread hole
- Gearbox diameter 37 mm.
- Motor Diameter 28.5 mm
- Length 63 mm without shaft
- Shaft length 30mm
- 180gm weight
- 120kgcm Holding Torque
- No-load current = 800 mA, Load current = upto 7.5 A(Max)
- Recommended to be used with DC Motor Driver 20A or Dual DC Motor Driver 20A

7.2 PIN DIAGRAM:



7.3 SCHEMATIC DIAGRAM:



7.4 BILL OF MATERIALS:

SL.NO	Item Description	Symbol	Qty	Price (₹)
1	L298	U2	1	20
2	L297DIL20-1	U1	1	5
3	KF301-5.0 2P	J3, J4, J5, J6, J7, J8	5	20
4	LM7805EE	U3	1	50
5	470uf	C1	1	2
6	220uf	C2	1	2
7	PIN 3	J1, J2	2	10
8	10k	R1, R2	2	2
9	CAP 0805	C4, C3	2	8
10	22uf	C5	1	2
11	1N4007	D1, D2, D3, D4, D5, D6, D7, D8	8	15
12	1 Ohm	R4, R10	2	3
13	22k	R5, R7	2	3
14	332 uf	C7	1	3
15	5kOhm	R6	1	2
TOTAL				147

TOTAL BILL OF MATERIALS:

SL.NO	Item Description	Symbol	Qty	Price (₹)
1	Motor Driver (L298N)	L1	1	147
2	Johnson Motor	M1	1	150
TOTAL				297

8.0 SG-90 SERVO MOTOR

8.1 DESCRIPTION

The TowerPro SG90 9g Mini Servo is 180° rotation servo. It is a Digital Servo Motor which 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.

8.2 PIN DIAGRAM:

8.3 FEATURES

- Operating Voltage is +5V typically
- Torque: 2.5kg/cm
- Operating speed is 0.1s/60°
- Gear Type: Plastic
- Rotation: 0°-180°
- Weight of motor: 9gm

8.4 BILL OF MATERIALS

SL.NO	Item Description	Symbol	Qty	Price (₹)
1	Servo Motor (SG-90)	SM1, SM2	2	150X2
TOTAL				300

*

9.0 MG995 SERVO MOTOR

9.1 DESCRIPTION

The high-speed standard servo can rotate approximately 120 degrees (60 in each direction). We can use any servo code, hardware or library to control these servos, so gear box, especially since it will fit in small places. It equips sophisticated internal circuitry that provides good torque, holding power, and faster updates in response to external forces.

9.2 PIN DIAGRAM

PIN CONFIGURATION:

MG995 has three terminals as mentioned in pin diagram and the function of each pin is given below.

Pin	Name	Function
1	Signal pin (Orange pin)	The PWM signal which states the axis position is given through this pin.
2	VCC (Red pin)	Positive power supply for servo motor is given to this pin.
3	Ground (Brown pin)	This pin is connected to ground of circuit or power supply.

9.3 FEATURES:

- Weight: 55g
- Dimension: 40.7 × 19.7 × 42.9 mm
- Operating Speed (4.8V no load): 20sec / 60 deg
- Operating Speed (6.0V no load): 16sec / 60 deg (no load)
- Stall Torque (4.8V): 10kg/cm
- Stall Torque (6.0V): 12kg/cm
- Operation Voltage: 4.8 - 7.2Volts
- Gear Type: All Metal Gears
- Stable and shock proof double ball bearing design
- Dead band width: 5 µs
- Temperature range: 0 ºC – 55 ºC.

9.4 BILL OF MATERIALS

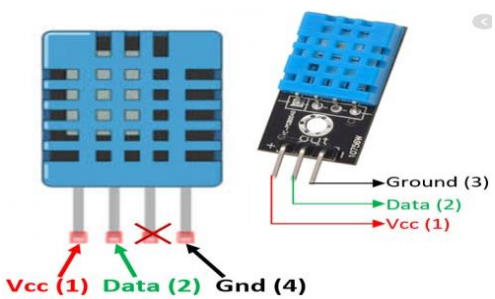
SL.NO	Item Description	Symbol	Qty	Price (₹)
1	Servo Motor (MG-995)	SM3, SM4, SM5	3	200X3
TOTAL				600

10.0 DHT11 (TEMPERATURE AND HUMIDITY SENSOR)

10.1 DESCRIPTION:

It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin. The only real downside of this sensor is you can only get new data from it once every 2 Sec. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high-performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness. Each DHT11 element is strictly calibrated in the laboratory that is extremely accurate on humidity calibration. The calibration coefficients are stored as programs in the OTP memory, which are used by the sensor's internal signal detecting process. The single-wire serial interface makes system integration quick and easy. Its small size, low power consumption and upto 20-meter signal transmission making it the best choice for various applications, including those most demanding ones.

10.2 PIN DIAGRAM:



PIN DISCRIPTION

Pin 1 is the VDD power supply 3.5～5.5V DC

Pin 2 is for DATA serial data, a single bus

Pin 2 NC, empty pin

Pin 4 GND ground, the negative power

10.3 FEATURES

- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%
- Resolution: Temperature and Humidity both are 16-bit
- Accuracy: ±1°C and ±1%

10.4 BILL OF MATERIALS

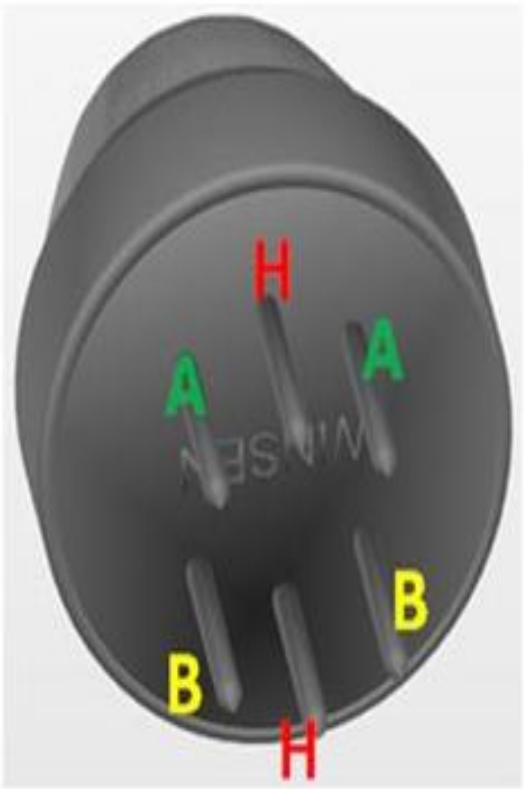
SL.NO	Item Description	Symbol	Qty	Price (₹)
1	DHT-11	TS1	1	85
TOTAL				85

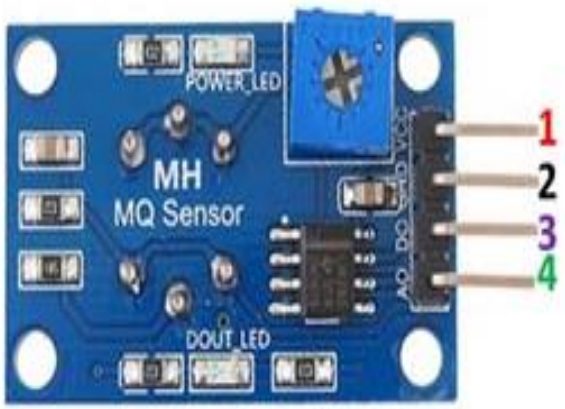
11.0 MQ-5 (GAS SENSOR)

11.1 DESCRIPTION

The MQ5 Gas Sensor module is useful for gas leakage detecting. It can detect LPG, i-butane, methane, alcohol, Hydrogen, CO2. The sensitivity can be adjusted using the on-board potentiometer, and you'd use this sensor by reading the analog pin to which it is connected.

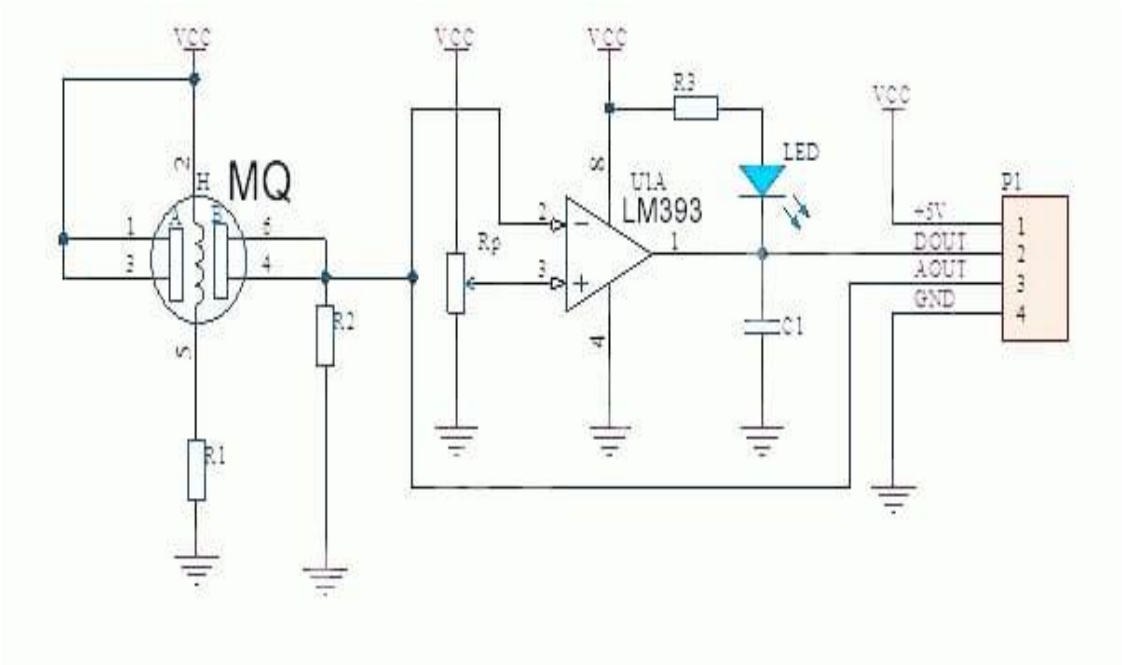
11.2 PIN DIAGRAM





Pin No.	Pin Name
1	Vcc(+5V)
2	Ground
3	Digital Out
4	Analog out

11.3 SCHEMATIC DIAGRAM:



11.4 BILL OF MATERIALS

SL.NO	Item Description	Symbol	Qty	Price (₹)
1	LM393, Boost Converter	U1	1	10
2	2.7M ohm	R1	1	2
3	910K ohm	R2	1	2
4	10uH	L1	1	13
5	100uf	C1	1	2
6	100uf	C2	1	2

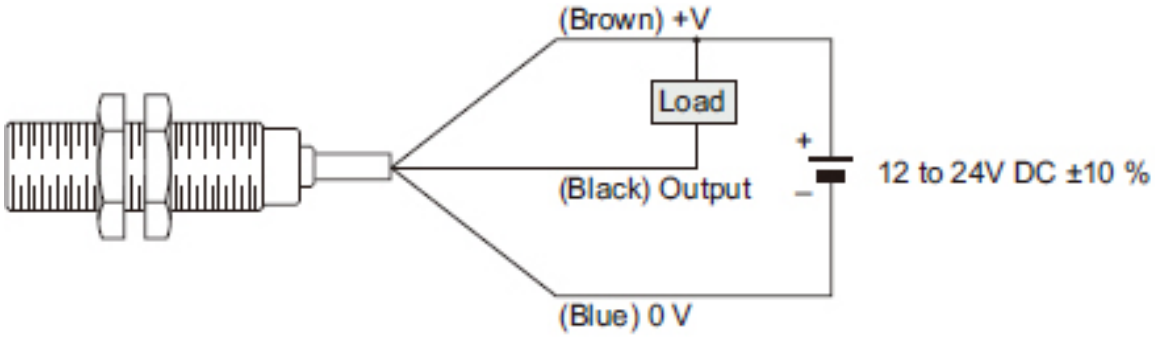
7	100nf	C3	1	2
8	1N5819, LED	D1	1	5
9	HDR2X3	P1	1	5
10	LM393	U2	1	40
11	LED	D2, D3	2	3
12	1uf	C4	1	3
13	1k ohm	R3, R4	2	3
14	SIP4	P2	1	2
15	4.7k ohm	R5	1	2
16	5R1 ohm	R6	1	2
17	100k ohm	R7	1	2
TOTAL				100

12.0 INDUCTIVE PROXIMITY SENSOR

12.1 DESCRIPTION

An inductive sensor is a device that uses the principle of electromagnetic induction to detect or measure objects. An inductor develops a magnetic field when a current flow through it; alternatively, a current will flow through a circuit containing an inductor when the magnetic field through it changes. This effect can be used to detect metallic objects that interact with a magnetic field. Non-metallic substances such as liquids or some kinds of dirt do not interact with the magnetic field.

12.2 PIN DIAGRAM



12.3 FEATURES

- Red LED checks the state of the proximity sensor
- High repeated positioning accuracy
- High switching frequency
- Wide voltage range
- Outer (Thread) Diameter: M12
- Antivibration, dust, water and oil prevention

12.4 BILL OF MATERIALS

SL.NO	Item Description	Symbol	Qty	Price (₹)
1	Inductive Proximity Sensor	IP1	1	200
TOTAL				200

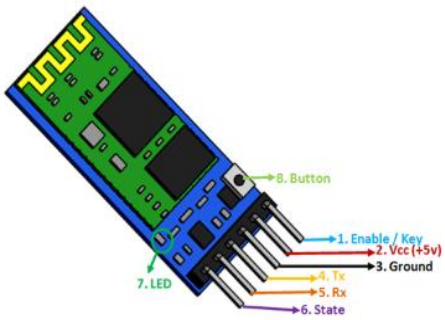
13.0 BLUETOOTH MODULE(HC-05)

13.1 DESCRIPTION

The HC-05 has two operating modes, one is the Data mode in which it can send and receive data from other Bluetooth devices and the other is the AT Command mode where the default device settings can be changed. We can operate the device in either of these two modes by using the key pin as explained in the pin description.

It is very easy to pair the HC-05 module with microcontrollers because it operates using the Serial Port Protocol (SPP). Simply power the module with +5V and connect the Rx pin of the module to the Tx of MCU and Tx pin of module to Rx of MCU as shown in the figure below

13.2 PIN DIAGRAM



13.3 FEATURES

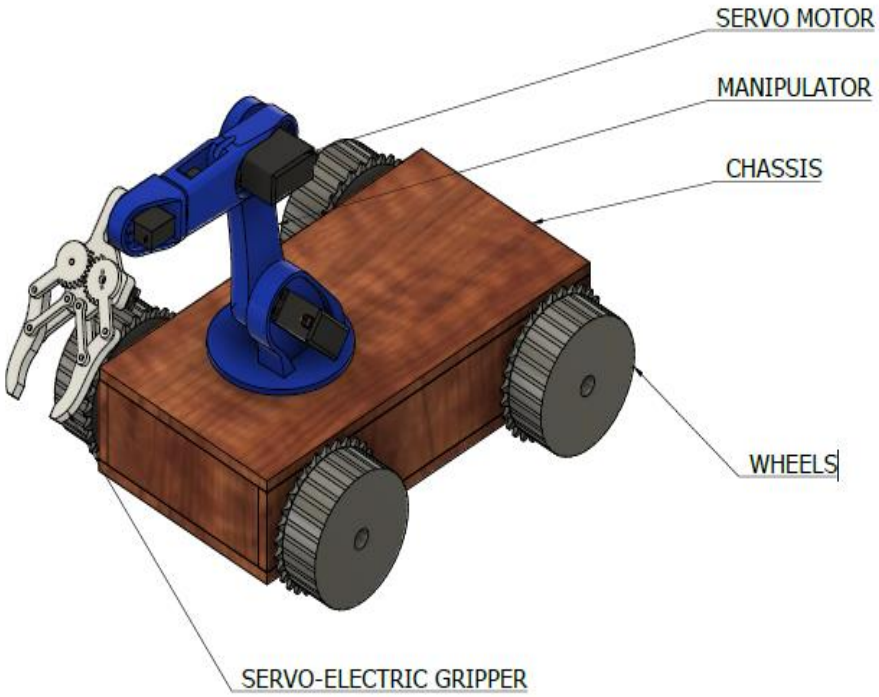
- Serial Bluetooth module for Arduino and other microcontrollers
- Operating Voltage: 4V to 6V (Typically +5V)
- Operating Current: 30mA
- Range: <100m
- Works with Serial communication (USART) and TTL compatible
- Follows IEEE 802.15.1 standardized protocol
- Uses Frequency-Hopping Spread spectrum (FHSS)

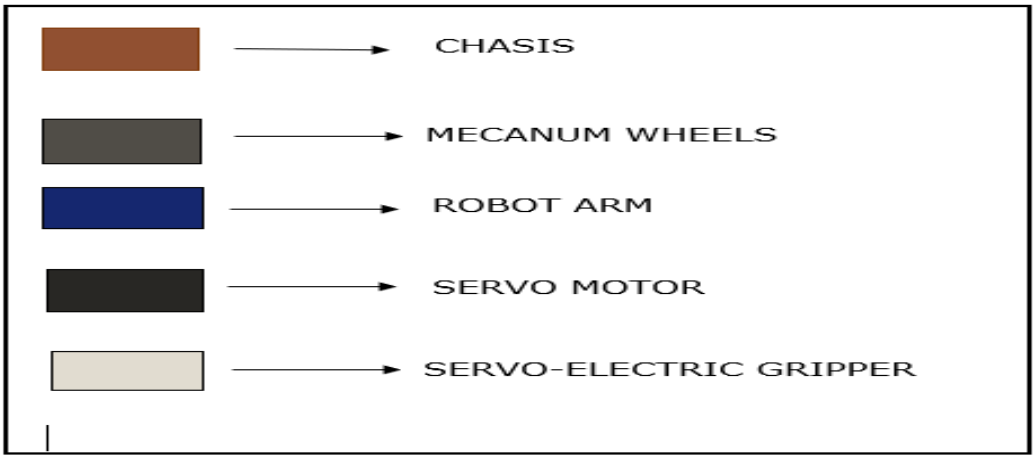
14.4 BILL OF MATERIALS

SL.NO	Item Description	Symbol	Qty	Price (₹)
1	BLUETOOTH MODULE	BL1	1	250
TOTAL				250

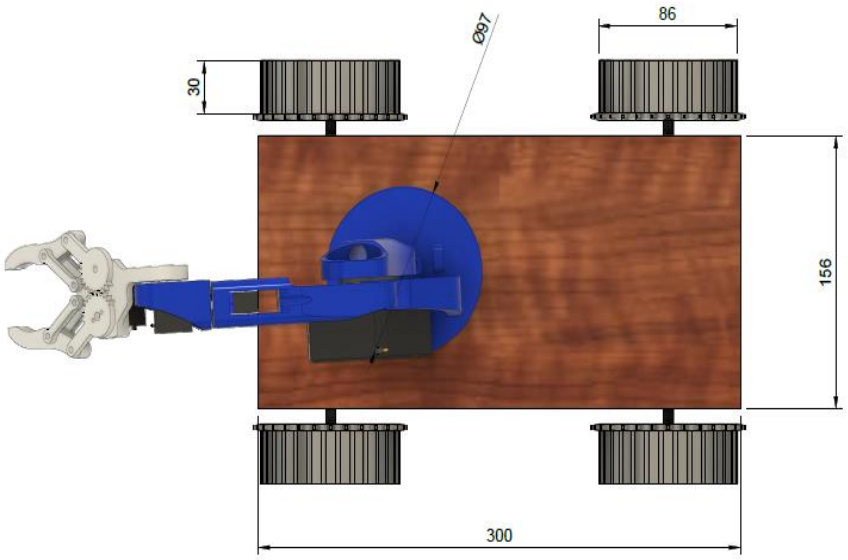
14.0 MECHANICAL DESIGN

14.1 OVERALL ISOMETRIC VIEW



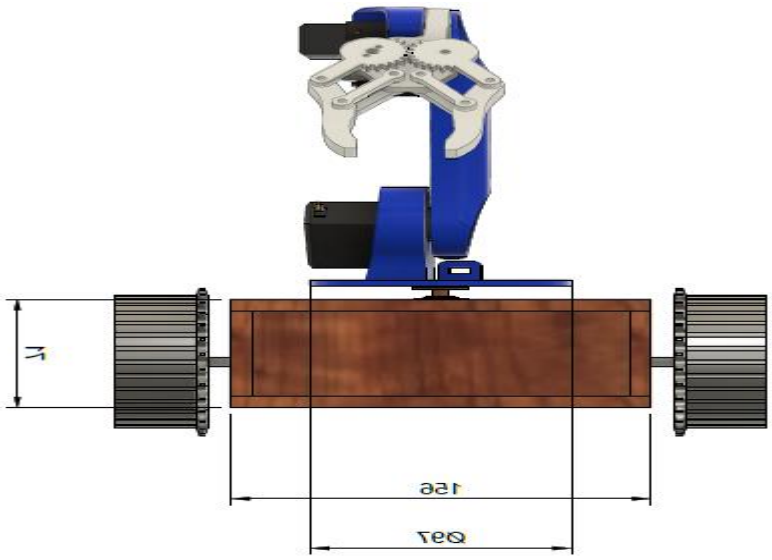


14.2 TOP VIEW



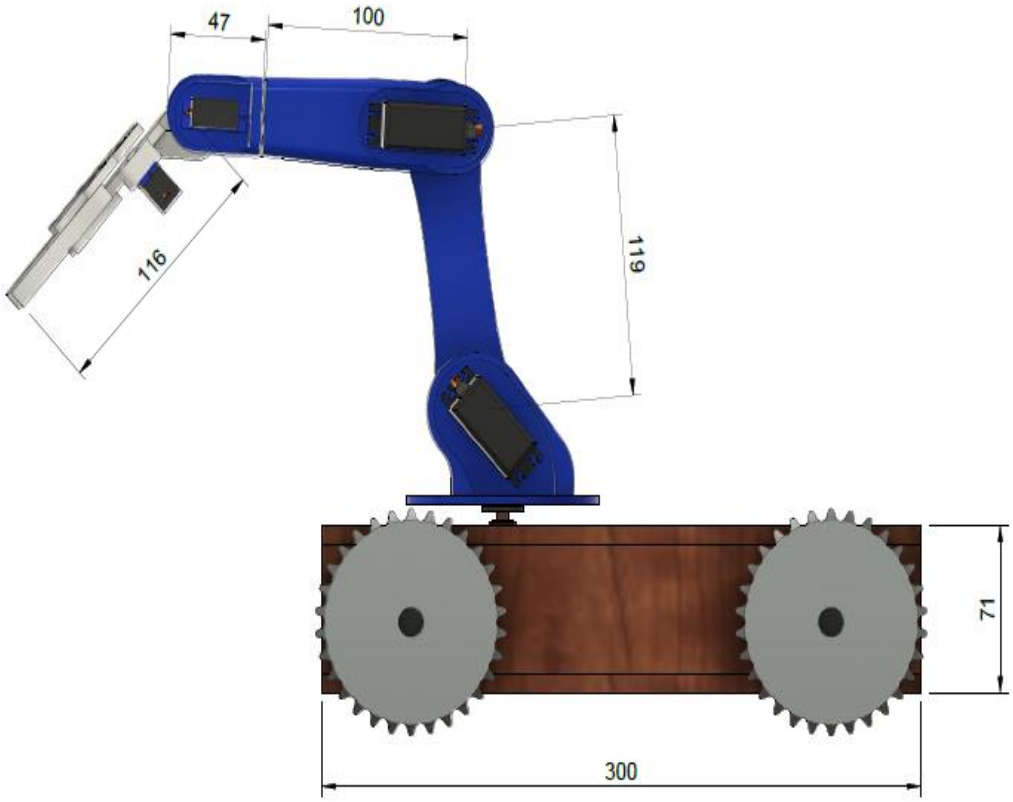
ALL DIMENSIONS ARE IN mm

14.3 SIDE VIEW



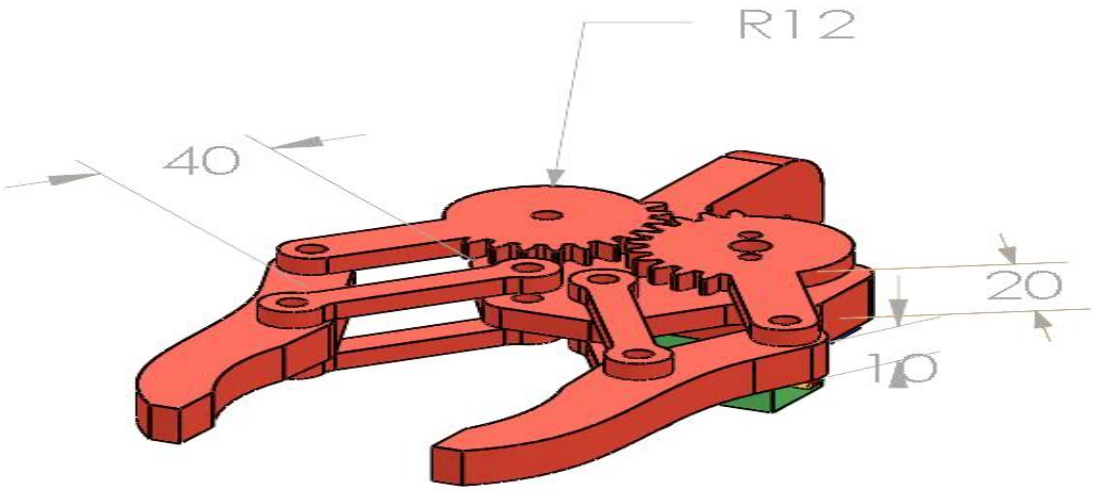
ALL DIMENSIONS ARE IN mm

14.4 FRONT VIEW



ALL DIMENSIONS ARE IN mm

14.5 SERVO ELECTRIC-GRIPPER



ALL DIMENSIONS ARE IN mm

15.0 BILL OF MATERIAL

15.1 MODULES

Sl.no	Modules	Individual Price (₹)	Qty	Total Price (₹)
1	Module1: Charging Circuit	1280	1	1280
2	Module2: Arduino Mega 2560	700	1	700
3	Module3: L298N with DC Gear motor	400	4	1300
4	Module4:SG-90	100	3	300
5	Module6:MG-995	250	3	750
6	Module7:DHT11	85	1	85
7	Module8:MQ-5	100	1	100
8	Module9: Inductive Proximity Sensor	200	1	200
9	Module10: Bluetooth module (HC-05)	250	1	250
TOTAL				4,965

15.2 MECHANICAL COMPONENTS

Sl.no	Mechanical components	Specification	Qty	Price (₹)
1	Screw Nut Spacers	M3*15mm M3.5*0.6mm M3*10mm, M3*35mm	200g 150g 500g	200
2	L-clamps	20 mm x 20 mm	20 pieces	50
3	plywood	IS 303 Grade (5mm x 500mm X 400mm)	1 sheet	200
4	wheels	100 mm (R50)	4	500
Total				950

15.3 ELECTRICAL COMPONENTS

Sl.no	Item Description	Specification	Quantity	Price (₹)
1	Insulation tape	10.5mm*10m	1	30
2	Sleeves	Ø1mm Ø3mm Ø6mm	2m 2m 2m	30 40 40
3	Connecting Wires	Multi Stranded (24 AWG, 10 AWG)	10m	150
Total				290

15.4 MISCELLANEOUS

Sl.no	Item description	Specifications	Quantity	Price (₹)
1	Anabond	202 (Cyanoacrylate Adhesive)	20ml	100
2	Nylon Bush	25mm	8	40

3	Plastic loop Lock Pin Tag	3”	75g	30
Total				170

15.5 OVERALL BILL OF MATERIAL

Items	**price (₹)**
Mechanical Components	950
Electrical Components	290
Modules	4,965
Miscellaneous	170
3D Parts for Robotic Arm	3,200
Total	9,575

16.0 SUB ASSEMBLY AND TESTING

16.1 POWER SUPPLY UNIT

In this project we requires one battery charger power supply (12V)

TESTING STEPS

- Checked all components.
- Connected on the PCB board according to the design.
- Gave input (230V AC).
- Verified the output by observing the reading on the multi-meter.
- Soldered the components on a general purpose circuit board.
- Checked for shorts on the circuits by using a multimeter.
- Gave supply and checked whether corrected voltage is coming on the correct pins.

16.2 CONTROL MODULE

we are using ATMEGA 2560 microcontroller as intelligent brain of our system.

The unit consist of

- Oscillator circuit
- Reset circuit
- 5V Power supply
- WDT
- POWER ON RESET
- RESET SWITCH
- PWM
- I/O PORTS

TESTING STEPS

- Checked all the components.
- Connected on the PCB according to the design.
- Gave input (0/1) to pins according to the condition for reset circuit.
- Connected the oscillator circuit as per design.
- Verified the output by pressing the reset switch in the reset circuit.
- Soldered the components on a general purpose circuit board.
- Checked for shorts on the circuits by using a multimeter.
- Gave supply and checked whether corrected voltage is coming on the correct pins.

16.3 MOTOR DRIVER MODULE

- Motor driving unit consist of ic L298N.
- It receives input from microcontroller.
- It drives the gear motor according to the receiving command and programs.

TESTING STEPS

- Checked all the components.
- Connected on the PCB according to the design.
- Gave input (0/1) to the pins according to the condition.
- Verified the output by abserving the rotation direction of motor.
- Check for shorts on the circuit by using the multimeter.
- Gave supply and checked whether corrected voltage is coming on the correct pin.
- Verify the connection is correct or not.
- Checked the mechanical structure for assembly.
- Gave input and verified the output

16.4 INPUT MODULE

- It consists of Bluetooth client.
- Remote controlled system using Bluetooth master with Arduino application (.apk).

TESTING STEPS

- Give input (0/1) according to the conditions.
- Gave supply and checked whether correct voltage is coming on the correct pins.

16.5 OUTPUT MODULE

- It consists of DC motors.
- DC motors are used for the motion of the robotic rover.
- It consists of servo motor used to move the robot arm.

TESTING STEPS

- Give input (0/1) according to the conditions.
- Verified the output by observing the motion of motors.
- Gave supply and checked whether correct voltage is coming on the correct pins.

17.0 OVERALL ASSEMBLY, TESTING AND OBSERVATION

17.1 MECHANICAL

- According to our design made all mechanical parts for our project. The required parts for our project are
- Moving part of vehicle
- Body of vehicle
- Robotic arm
- Assembled all mechanical parts

17.2 ELECTRICAL AND ELECTRONICS

According to our design of each sub circuits made all modules for our project. The required modules for our project are.

- Charging circuit
- Microcontroller Unit
- Motor Driver Circuit
- From the output of the +5V power supply, made each connection to Microcontroller unit.
- From the output of the +12V power supply, made each connection to Driver circuit.

- The output of the switch circuit and RF Module is connected to the input of the Microcontroller unit.
- From the output of the Microcontroller unit a connection is made to the input of Driver circuit.
- The shaft of the motor then coupled with rack and pinion.
- Checked the continuity of all connections.
- Corresponding switches is pressed and checked the output status.
- Measured the voltage at each input and output terminal

18.0 INTER CONNECTION DIAGRAM

19.0 OVERALL CIRCUIT DIAGRAM

20.0 MODULAR DIAGRAM

21.0 ADVANTAGES & DISADVANTAGES

ADVANTAGES

- Reducing Man Power
- Fast Production
- Time Management

DISADVANTAGES

- Less Payload Capacity
- Full Manual Controlled
- Limited Operating Range

APPLICATIONS

- Pick & Place
- Industrial Usage
- Component Handling

22.0 PROGRAM

22.1 PROGRAM

```
#include <SoftwareSerial.h>

#include <AccelStepper.h>

#include <Servo.h>


Servo servo01;

Servo servo02;

Servo servo03;

Servo servo04;

Servo servo05;

Servo servo06;


SoftwareSerial Bluetooth(A8,36); // Arduino(RX, TX) - HC-
05 Bluetooth (TX, RX)


// Define the stepper motors and the pins the will use

AccelStepper LeftBackWheel(1, 42, 43);   // (Type:driver,
STEP, DIR) - Stepper1

AccelStepper LeftFrontWheel(1, 40, 41);  // Stepper2

AccelStepper RightBackWheel(1, 44, 45);  // Stepper3

AccelStepper RightFrontWheel(1, 46, 47); // Stepper4
```

```
#define led 14


int wheelSpeed = 1500;


int lbw[50], lfw[50], rbw[50], rfw[50]; // arrays for storing positions/steps


int servo1Pos, servo2Pos, servo3Pos, servo4Pos, servo5Pos, servo6Pos; // current position

int servo1PPos, servo2PPos, servo3PPos, servo4PPos, servo5PPos, servo6PPos; // previous position

int servo01SP[50], servo02SP[50], servo03SP[50], servo04SP[50], servo05SP[50], servo06SP[50]; // for storing positions/steps

int speedDelay = 20;

int index = 0;

int dataIn;

int m = 0;


void setup() {

  // Set initial seed values for the steppers

  LeftFrontWheel.setMaxSpeed(3000);

  LeftBackWheel.setMaxSpeed(3000);
```

```
  RightFrontWheel.setMaxSpeed(3000);
  RightBackWheel.setMaxSpeed(3000);
  pinMode(led, OUTPUT);
  servo01.attach(5);
  servo02.attach(6);
  servo03.attach(7);
  servo04.attach(8);
  servo05.attach(9);
  servo06.attach(10);
  Bluetooth.begin(4800); // Default baud rate of the
Bluetooth module
  Bluetooth.setTimeout(5);
  delay(20);
  Serial.begin(9600);
  // Move robot arm to initial position
  servo1PPos = 90;
  servo01.write(servo1PPos);
  servo2PPos = 100;
  servo02.write(servo2PPos);
  servo3PPos = 120;
  servo03.write(servo3PPos);
  servo4PPos = 95;
```

```
  servo04.write(servo4PPos);
  servo5PPos = 60;
  servo05.write(servo5PPos);
  servo6PPos = 110;
  servo06.write(servo6PPos);
}

void loop() {
  // Check for incoming data
  if (Bluetooth.available() > 0) {
    dataIn = Bluetooth.read();  // Read the data
Serial.println(m ="");
    if (dataIn == 0) {
      m = 0;
    }
    if (dataIn == 1) {
      m = 1;
    }
    if (dataIn == 2) {
      m = 2;
    }
    if (dataIn == 3) {
```

```
  m = 3;
}
if (dataIn == 4) {
  m = 4;
}
if (dataIn == 5) {
  m = 5;
}
if (dataIn == 6) {
  m = 6;
}
if (dataIn == 7) {
  m = 7;
}
if (dataIn == 8) {
  m = 8;
}
if (dataIn == 9) {
  m = 9;
}
if (dataIn == 10) {
  m = 10;
```

```
}
if (dataIn == 11) {
  m = 11;
}
if (dataIn == 12) {
  m = 12;
}
if (dataIn == 14) {
  m = 14;
}
if (dataIn == 16) {
  m = 16;
}
if (dataIn == 17) {
  m = 17;
}
if (dataIn == 18) {
  m = 18;
}
if (dataIn == 19) {
  m = 19;
}
```

```
if (dataIn == 20) {
  m = 20;
}
if (dataIn == 21) {
  m = 21;
}
if (dataIn == 22) {
  m = 22;
}
if (dataIn == 23) {
  m = 23;
}
if (dataIn == 24) {
  m = 24;
}
if (dataIn == 25) {
  m = 25;
}
if (dataIn == 26) {
  m = 26;
}
if (dataIn == 27) {
```

```
  m = 27;
}

// Move the Mecanum wheels platform
if (m == 4) {
  moveSidewaysLeft();
}
if (m == 5) {
  moveSidewaysRight();
}
if (m == 2) {
  moveForward();
}
if (m == 7) {
  moveBackward();
}
if (m == 3) {
  moveRightForward();
}
if (m == 1) {
  moveLeftForward();
}
```

```
if (m == 8) {
  moveRightBackward();
}
if (m == 6) {
  moveLeftBackward();
}
if (m == 9) {
  rotateLeft();
}
if (m == 10) {
  rotateRight();
}

if (m == 0) {
  stopMoving();
}

// Mecanum wheels speed
if (dataIn > 30 & dataIn < 100) {
  wheelSpeed = dataIn * 20;
}
```

```
// Move robot arm
// Move servo 1 in positive direction
while (m == 16) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo01.write(servo1PPos);
  servo1PPos++;
  delay(speedDelay);
}
// Move servo 1 in negative direction
while (m == 17) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo01.write(servo1PPos);
  servo1PPos--;
  delay(speedDelay);
}
// Move servo 2
while (m == 19) {
  if (Bluetooth.available() > 0) {
```

```
    m = Bluetooth.read();
  }
  servo02.write(servo2PPos);
  servo2PPos++;
  delay(speedDelay);
}
while (m == 18) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo02.write(servo2PPos);
  servo2PPos--;
  delay(speedDelay);
}
// Move servo 3
while (m == 20) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo03.write(servo3PPos);
  servo3PPos++;
  delay(speedDelay);
```

```
}
while (m == 21) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo03.write(servo3PPos);
  servo3PPos--;
  delay(speedDelay);
}
// Move servo 4
while (m == 23) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo04.write(servo4PPos);
  servo4PPos++;
  delay(speedDelay);
}
while (m == 22) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
```

```
  servo04.write(servo4PPos);
  servo4PPos--;
  delay(speedDelay);
 }
 // Move servo 5
 while (m == 25) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo05.write(servo5PPos);
  servo5PPos++;
  delay(speedDelay);
 }
 while (m == 24) {
  if (Bluetooth.available() > 0) {
    m = Bluetooth.read();
  }
  servo05.write(servo5PPos);
  servo5PPos--;
  delay(speedDelay);
 }
 // Move servo 6
```

```
    while (m == 26) {
      if (Bluetooth.available() > 0) {
        m = Bluetooth.read();
      }
      servo06.write(servo6PPos);
      servo6PPos++;
      delay(speedDelay);
    }
    while (m == 27) {
      if (Bluetooth.available() > 0) {
        m = Bluetooth.read();
      }
      servo06.write(servo6PPos);
      servo6PPos--;
      delay(speedDelay);
    }


    // If arm speed slider is changed
    if (dataIn > 101 & dataIn < 250) {
      speedDelay = dataIn / 10; // Change servo speed
(delay time)
    }
```

```
    // If button "SAVE" is pressed
    if (m == 12) {
      //if it's initial save, set the steppers position to 0
      if (index == 0) {
        LeftBackWheel.setCurrentPosition(0);
        LeftFrontWheel.setCurrentPosition(0);
        RightBackWheel.setCurrentPosition(0);
        RightFrontWheel.setCurrentPosition(0);
      }
      lbw[index] = LeftBackWheel.currentPosition();  // save
position into the array
      lfw[index] = LeftFrontWheel.currentPosition();
      rbw[index] = RightBackWheel.currentPosition();
      rfw[index] = RightFrontWheel.currentPosition();


      servo01SP[index] = servo1PPos;  // save position into
the array
      servo02SP[index] = servo2PPos;
      servo03SP[index] = servo3PPos;
      servo04SP[index] = servo4PPos;
      servo05SP[index] = servo5PPos;
      servo06SP[index] = servo6PPos;
```

```
    index++;                    // Increase the array index
    m = 0;
  }

  // If button "RUN" is pressed
  if (m == 14) {
    runSteps();

    // If button "RESET" is pressed
    if (dataIn != 14) {
      stopMoving();
      memset(lbw, 0, sizeof(lbw)); // Clear the array data
to 0
      memset(lfw, 0, sizeof(lfw));
      memset(rbw, 0, sizeof(rbw));
      memset(rfw, 0, sizeof(rfw));
      memset(servo01SP, 0, sizeof(servo01SP)); // Clear
the array data to 0
      memset(servo02SP, 0, sizeof(servo02SP));
      memset(servo03SP, 0, sizeof(servo03SP));
      memset(servo04SP, 0, sizeof(servo04SP));
      memset(servo05SP, 0, sizeof(servo05SP));
      memset(servo06SP, 0, sizeof(servo06SP));
```

```
        index = 0;  // Index to 0
      }
    }
  }
  LeftFrontWheel.runSpeed();
  LeftBackWheel.runSpeed();
  RightFrontWheel.runSpeed();
  RightBackWheel.runSpeed();


  // Monitor the battery voltage
  int sensorValue = analogRead(A0);
  float voltage = sensorValue * (5.0 / 1023.00) * 3; //
Convert the reading values from 5v to suitable 12V i
  //Serial.println(voltage);
  // If voltage is below 11V turn on the LED
  if (voltage < 11) {
    digitalWrite(led, HIGH);
  }
  else {
    digitalWrite(led, LOW);
  }
}
```

```
void moveForward() {
  LeftFrontWheel.setSpeed(wheelSpeed);
  LeftBackWheel.setSpeed(wheelSpeed);
  RightFrontWheel.setSpeed(wheelSpeed);
  RightBackWheel.setSpeed(wheelSpeed);
}
void moveBackward() {
  LeftFrontWheel.setSpeed(-wheelSpeed);
  LeftBackWheel.setSpeed(-wheelSpeed);
  RightFrontWheel.setSpeed(-wheelSpeed);
  RightBackWheel.setSpeed(-wheelSpeed);
}
void moveSidewaysRight() {
  LeftFrontWheel.setSpeed(wheelSpeed);
  LeftBackWheel.setSpeed(-wheelSpeed);
  RightFrontWheel.setSpeed(-wheelSpeed);
  RightBackWheel.setSpeed(wheelSpeed);
}
void moveSidewaysLeft() {
  LeftFrontWheel.setSpeed(-wheelSpeed);
  LeftBackWheel.setSpeed(wheelSpeed);
  RightFrontWheel.setSpeed(wheelSpeed);
```

```
  RightBackWheel.setSpeed(-wheelSpeed);
}
void rotateLeft() {
  LeftFrontWheel.setSpeed(-wheelSpeed);
  LeftBackWheel.setSpeed(-wheelSpeed);
  RightFrontWheel.setSpeed(wheelSpeed);
  RightBackWheel.setSpeed(wheelSpeed);
}
void rotateRight() {
  LeftFrontWheel.setSpeed(wheelSpeed);
  LeftBackWheel.setSpeed(wheelSpeed);
  RightFrontWheel.setSpeed(-wheelSpeed);
  RightBackWheel.setSpeed(-wheelSpeed);
}
void moveRightForward() {
  LeftFrontWheel.setSpeed(wheelSpeed);
  LeftBackWheel.setSpeed(0);
  RightFrontWheel.setSpeed(0);
  RightBackWheel.setSpeed(wheelSpeed);
}
void moveRightBackward() {
  LeftFrontWheel.setSpeed(0);
```

```
  LeftBackWheel.setSpeed(-wheelSpeed);
  RightFrontWheel.setSpeed(-wheelSpeed);
  RightBackWheel.setSpeed(0);
}
void moveLeftForward() {
  LeftFrontWheel.setSpeed(0);
  LeftBackWheel.setSpeed(wheelSpeed);
  RightFrontWheel.setSpeed(wheelSpeed);
  RightBackWheel.setSpeed(0);
}
void moveLeftBackward() {
  LeftFrontWheel.setSpeed(-wheelSpeed);
  LeftBackWheel.setSpeed(0);
  RightFrontWheel.setSpeed(0);
  RightBackWheel.setSpeed(-wheelSpeed);
}
void stopMoving() {
  LeftFrontWheel.setSpeed(0);
  LeftBackWheel.setSpeed(0);
  RightFrontWheel.setSpeed(0);
  RightBackWheel.setSpeed(0);
}
```

```
// Automatic mode custom function - run the saved steps

void runSteps() {

  while (dataIn != 13) {   // Run the steps over and over
again until "RESET" button is pressed

    for (int i = 0; i <= index - 2; i++) {  // Run through all
steps(index)

      if (Bluetooth.available() > 0) {      // Check for
incomding data

        dataIn = Bluetooth.read();

        if ( dataIn == 15) {           // If button "PAUSE" is
pressed

          while (dataIn != 14) {         // Wait until "RUN" is
pressed again

            if (Bluetooth.available() > 0) {

              dataIn = Bluetooth.read();

              if ( dataIn == 13) {

                break;

              }

            }

          }

        }

        // If speed slider is changed

        if (dataIn > 100 & dataIn < 150) {
```

```
        speedDelay = dataIn / 10; // Change servo speed
(delay time)
      }
      // Mecanum wheels speed
      if (dataIn > 30 & dataIn < 100) {
        wheelSpeed = dataIn * 10;
        dataIn = 14;
      }
    }
    LeftFrontWheel.moveTo(lfw[i]);
    LeftFrontWheel.setSpeed(wheelSpeed);
    LeftBackWheel.moveTo(lbw[i]);
    LeftBackWheel.setSpeed(wheelSpeed);
    RightFrontWheel.moveTo(rfw[i]);
    RightFrontWheel.setSpeed(wheelSpeed);
    RightBackWheel.moveTo(rbw[i]);
    RightBackWheel.setSpeed(wheelSpeed);

    while (LeftBackWheel.currentPosition() != lbw[i] &
LeftFrontWheel.currentPosition() != lfw[i] &
RightFrontWheel.currentPosition() != rfw[i] &
RightBackWheel.currentPosition() != rbw[i]) {
      LeftFrontWheel.runSpeedToPosition();
```

```
      LeftBackWheel.runSpeedToPosition();
      RightFrontWheel.runSpeedToPosition();
      RightBackWheel.runSpeedToPosition();
    }
    // Servo 1
    if (servo01SP[i] == servo01SP[i + 1]) {
    }
    if (servo01SP[i] > servo01SP[i + 1]) {
      for ( int j = servo01SP[i]; j >= servo01SP[i + 1]; j--)
{
        servo01.write(j);
        delay(speedDelay);
      }
    }
    if (servo01SP[i] < servo01SP[i + 1]) {
      for ( int j = servo01SP[i]; j <= servo01SP[i + 1];
j++) {
        servo01.write(j);
        delay(speedDelay);
      }
    }

    // Servo 2
```

```
    if (servo02SP[i] == servo02SP[i + 1]) {
    }
    if (servo02SP[i] > servo02SP[i + 1]) {
      for ( int j = servo02SP[i]; j >= servo02SP[i + 1]; j--)
{
        servo02.write(j);
        delay(speedDelay);
      }
    }
    if (servo02SP[i] < servo02SP[i + 1]) {
      for ( int j = servo02SP[i]; j <= servo02SP[i + 1];
j++) {
        servo02.write(j);
        delay(speedDelay);
      }
    }

    // Servo 3
    if (servo03SP[i] == servo03SP[i + 1]) {
    }
    if (servo03SP[i] > servo03SP[i + 1]) {
      for ( int j = servo03SP[i]; j >= servo03SP[i + 1]; j--)
{
```

```
        servo03.write(j);
        delay(speedDelay);
      }
    }
    if (servo03SP[i] < servo03SP[i + 1]) {
      for ( int j = servo03SP[i]; j <= servo03SP[i + 1];
j++) {
        servo03.write(j);
        delay(speedDelay);
      }
    }

    // Servo 4
    if (servo04SP[i] == servo04SP[i + 1]) {
    }
    if (servo04SP[i] > servo04SP[i + 1]) {
      for ( int j = servo04SP[i]; j >= servo04SP[i + 1]; j--)
{
        servo04.write(j);
        delay(speedDelay);
      }
    }
    if (servo04SP[i] < servo04SP[i + 1]) {
```

```
      for ( int j = servo04SP[i]; j <= servo04SP[i + 1];
j++) {
        servo04.write(j);
        delay(speedDelay);
      }
    }



    // Servo 5
    if (servo05SP[i] == servo05SP[i + 1]) {
    }
    if (servo05SP[i] > servo05SP[i + 1]) {
      for ( int j = servo05SP[i]; j >= servo05SP[i + 1]; j--)
{
        servo05.write(j);
        delay(speedDelay);
      }
    }
    if (servo05SP[i] < servo05SP[i + 1]) {
      for ( int j = servo05SP[i]; j <= servo05SP[i + 1];
j++) {
        servo05.write(j);
```

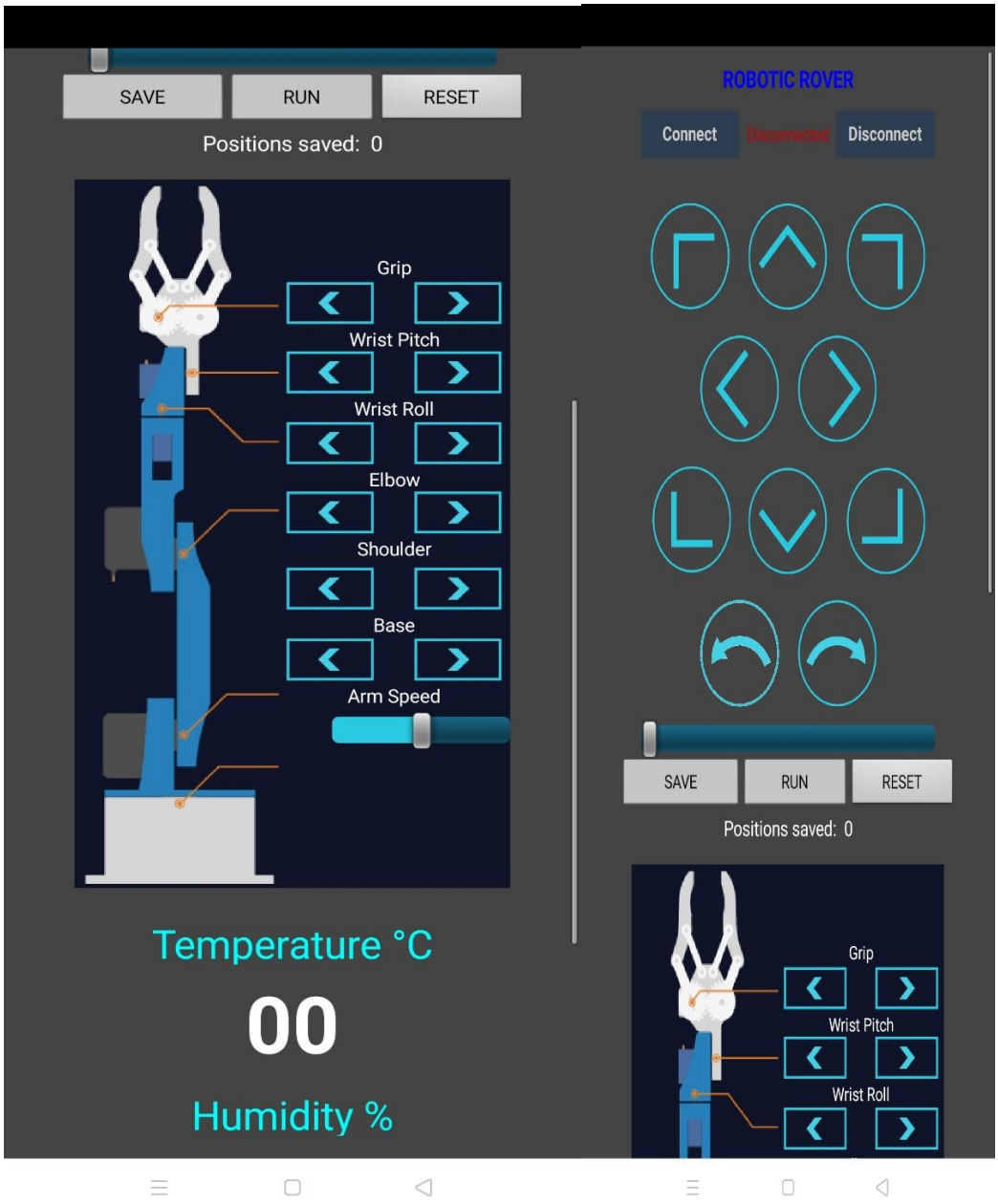
```
        delay(speedDelay);
      }
    }

    // Servo 6
    if (servo06SP[i] == servo06SP[i + 1]) {
    }
    if (servo06SP[i] > servo06SP[i + 1]) {
      for ( int j = servo06SP[i]; j >= servo06SP[i + 1]; j--)
{
        servo06.write(j);
        delay(speedDelay);
      }
    }
    if (servo06SP[i] < servo06SP[i + 1]) {
      for ( int j = servo06SP[i]; j <= servo06SP[i + 1];
j++) {
        servo06.write(j);
        delay(speedDelay);
      }
    }
   }
  }
```

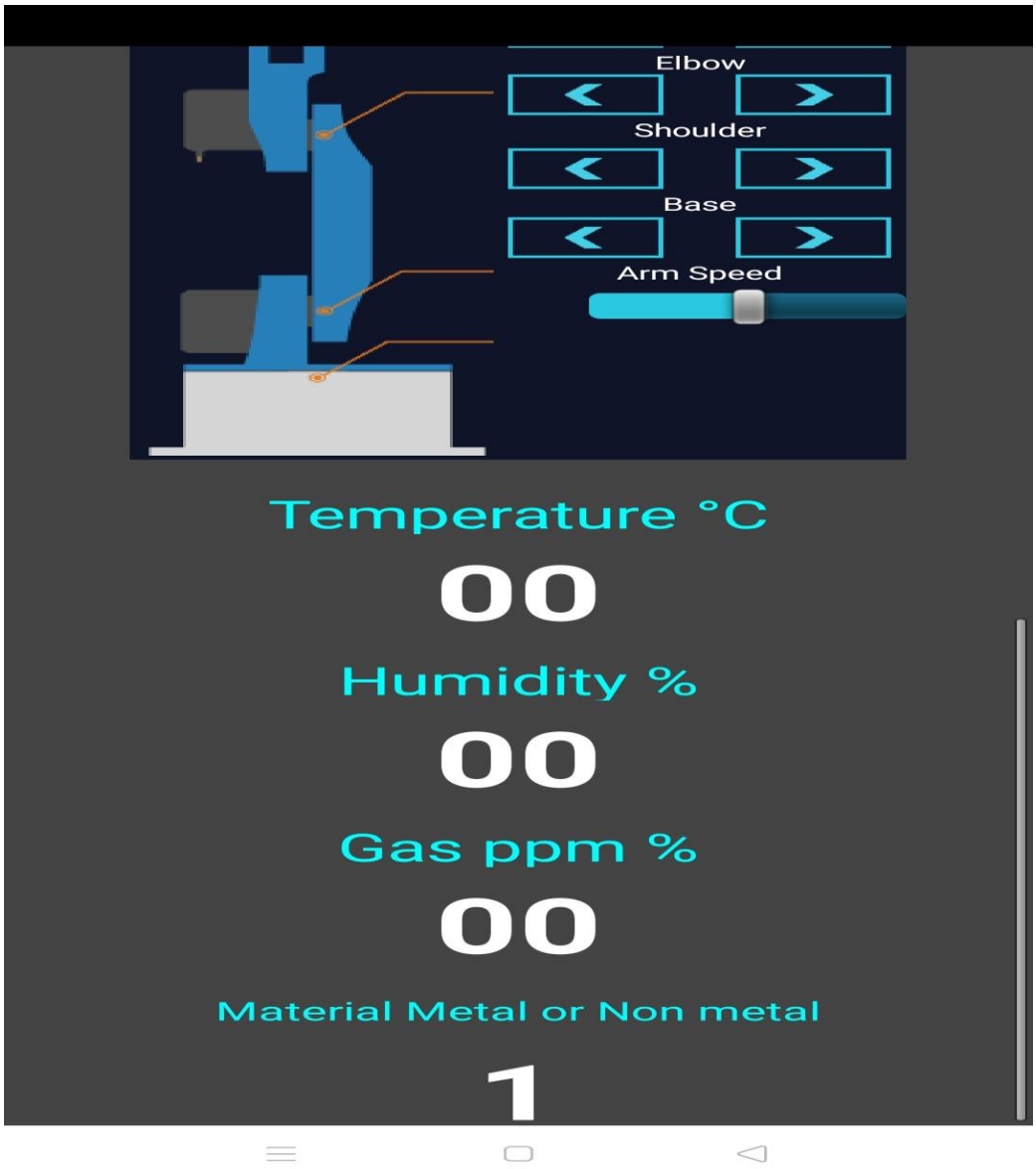
}

22.2 MIT APP INVENTOR

BLOCK DESIGN:

USER INTERFACE:




Elbow
Shoulder
Base
Arm Speed
Temperature °C
00
Humidity %
00
Gas ppm %
00
Material Metal or Non metal
1

23.0 CONCLUSION

This gives us extreme pleasure to conclude this project report prepared on the basis of our 6th semester academic project undergone in NEC-NTTF training center. We are well fed with all necessary information and knowledge about the technology of present days.

Thus, the robotic rover can help for gripping up harmful objects and place them safely and these robots are used in manufacturing industries to move objects.