**Meteor 2017**

**User Manual**

**Disclaimer:**

This is a user manual for all participants of Meteor 2017 and other enthusiasts who have gained access to it for your learning of robotics. There may be questions you face while you read through the manual and you may contact the members of the team, TRSKNCOE, at any time for the clarification of your doubts and also for any new things you wish to know about. We will update in the user manual as and if we find errors or missing data hindering with the process of your own development of robots or other projects in which you choose to use the hardware or software provided to you in Meteor 2017. If you wish to buy another kit, you can contact TRSKNCOE at any time and purchase the same.

“I just want the future to happen faster. I can’t imagine the future without robots”

Before we start we should have a clear cut idea of what we require and what basic concepts should we follow in order to built up a bot.

Also we should have the knowledge of the components such as resistors, transistors, oscillators, motor drivers and other electronic components.

Also the Microcontrollers, what type of battery we should use. We have covered all these small points in this manual.

Most of we are confused in making the connection and understanding the connection which is done in our robot for this you should refer this manual.

We also have covered this programming part in our manual for better understanding. So in this manual we will cover all the basic needs and the concepts we require in making our robot.

**CONTENTS**

**Meteorite (Wi-Fi enabled robot)**

Mechanical Section

* MATERIAL REQUIRED
* CHASSIS
* STABILITY
* SELECTION OF WHEEL
* TORQUE & RPM
* MOTOR MOUNTING
* OTHER TIPS

Electronics Section

* Printed Circuit Board Description
* Component Description
* Mounting Instructions

Programming Section

Programming a robot

App development

**Automatic Extension Board**

* Connections and program details

**METEORITE**

**MECHANICAL SECTION**

**BASIC REQUIREMENTS**

These are some of the most fundamental materials which are definitely needed before anyone begins with robot making:

* Screw Drivers
* Spanner
* Drill bit
* Base Material
* Wheels + Castors
* DC Geared Motor (as per your need)
* Nails, Screws and Nuts
* Clamps

**CHASSIS**

Chassis is the base of our Robot, to which everything else is attached. It is probably the largest part of a robot, so make sure it is made of a light weight rigid material. Thus it is a part on which other mountings such as batteries, wheels, mechanism is done.

**CHASSIS**

WOODEN

ALUMINIUM

FIBER

CHASSIS MATERIAL USED IN OUR ROBOT :**MDF**



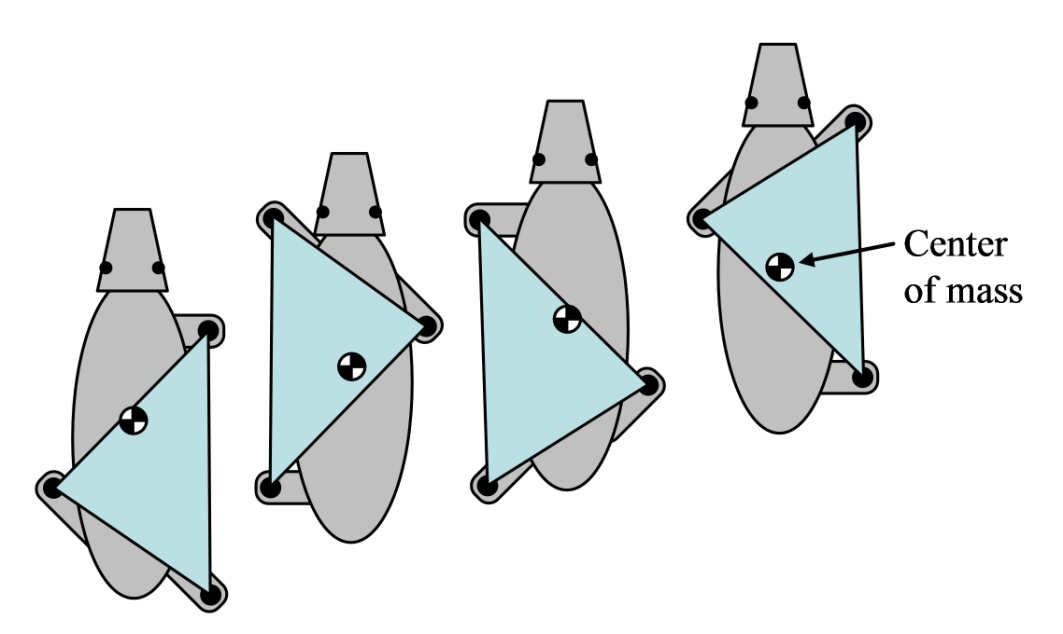
**Properties of Material:**

* Strength to Carry Weight
* Should be Light
* Can be easily worked on (drilled, cut)
* Easily Available

Materials which can be used for building Robots:

1. Balsa Wood : Light weight, good choice for flying bots
2. Aluminum : Light and Strong
3. Acrylic Sheet : Cheaper than Aluminum, easy to work.
4. MDF (Med. Density Fiber) or any other easily available material.

**STABILITY**

The centre of gravity(cog) of robot should lie inside the polygon formed by the wheels.

Let us take an example of a three wheeler to understand this concept. Let the polygon formed by the wheels be an equilateral. Let initially the COG is located at the centroid, if we start increasing weight equally on the two adjacent corners of polygon, the normal on the third corner becomes zero.

* The COG should be as low as possible for better stability while accelerating/decelerating.
* Moving the COG can allow the bot to move across wider gaps, climb steeper slopes and get over or onto higher steps.

**SELECTION OF WHEELS**

While buying wheels, motor consideration must be taken into account. For a start, the is torque and velocity. Large diameter wheels give your robot low torque but high velocity. So if you already have a very strong motor, then you can use wheels with larger diameters. Servo’s already have good torque, so you should use larger diameter wheels. But if your motor is weak (such as if it does not have any gearbox), you want to use such a much smaller diameter wheel. This will make your robot slower, but atleast it has enough torque to go up a hill!!

The texture of your wheel is very terrain dependant. If your wheel is too smooth then it will not have much friction. This is a serious issue with omni wheels. So it is advisory to have normal wheel with good friction.

**Castors**

A caster is an undriven, single, double, or compound wheel that is designed to be mounted to the bottom of a larger object so as to enable that object to be easily moved. They are available in various sizes and are commonly made of stainless steel, rubber or nylon

**CASTERS OPERATE WELL ON SMOOTH AND FLAT SURFACES.**

Casters may be fixed to roll along a straight line path, or mounted on a pivot such that the wheel will automatically align itself to the direction of translation



**TORQUE AND RPM**

No of wheels = 3 with different drive

No of motors = 2

The bot is front wheel driven and has a rear castor wheel.

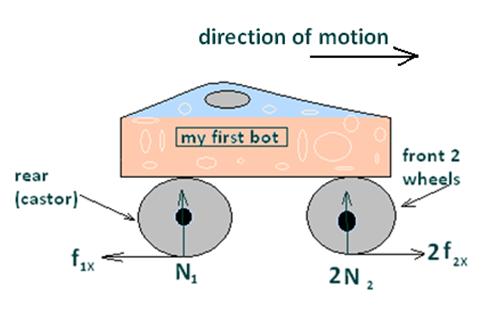
Coefficient of Friction = 0.6 and,

Radius of wheel = 3cm

Weight of Bot (W=mg= 20N)

When the bot is accelerating in the forward direction, the friction acts in the forward direction on the front wheels and on the rear caster wheel the friction acts

in the opposite direction (see fig). Assuming that we have an impending motion i.e. the torque is sufficient to start the bot.



Following are the equations of motion,

N1 + 2\*N2 = W

F2\*r – Τ = 0

Since there is an impending motion hence

F2x = µN

For a motor the impending torque should be able to overcome the static friction torque. After this, the torque required for rotating the wheels of the bot and the RPM of motor will increase, hence the speed of the bot.

N = W/3(considering the weight is equally distributed)

µ\*W/3\*r – Τ = 0

0.6\*(20/3)\*0.03 – Τ = 0

Τ = 0.12 N-m or 1.2 kg-cm

**Calculating the RPM of motor:**

Speed of bot = RPM of motor\*radius of wheels (while moving forward)

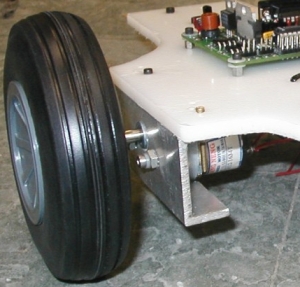
For e.g

0.5(m/s) = ω \* 0.03(m)

ω = 17(rad/s) = 17 \* 60/2∏ (RPM) =160 (RPM)

Still RPM upto 300 is comfortable or rather controllable. If you feel that the RPM of your Bot is difficult to handle then you should choose lower voltage rating.

**MOTOR MOUNTING**

To mount any type of motor to your chassis you will need to use an L shaped bracket. For a DC motor, all you need to do is take a sheet of aluminum, drill two holes in two corners, drill two more holes on the other half to match the motor screw holes, then bend the entire piece in a 90 degree angle.

**OTHER TIPS**

* **Use fewer and simpler parts**. Don’t use unnecessary or over complicated features, or designs you do not have the tools to make or are really hard to make.
* **Use off the shelf parts**. When you purchase a part, it costs money. However there is a good chance that the off-the-shelf part is better than anything you design and build yourself.
* **Do not use more than 2 or 3 different types of screws**. If you can make your entire robot out of only very common 4-40 screws, you are on the right track.

**ELECTRONICS SECTION**

Printed Circuit Board Description

Component list-

1.Node MCU (controller + wi-fi)

2.L293D (motor driver IC)

3.LM324 (Op-amp IC)

4.7805(voltage regulator)

5.DPDT switch (on/off)

6.Potentiometer (variable resistor)

7.1000uf,470uf capacitors (filtering capacitors)

8.motor connectors (green colored connector)

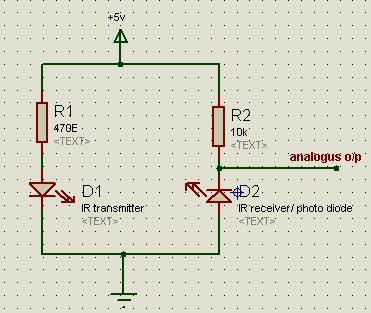
9. 3v6 Zener diode

10. Infrared pair (transmitter and receiver)

11.DC geared motor

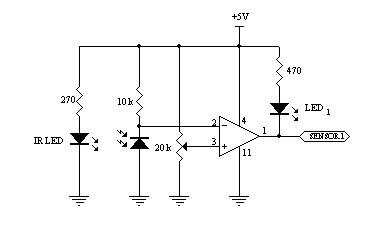
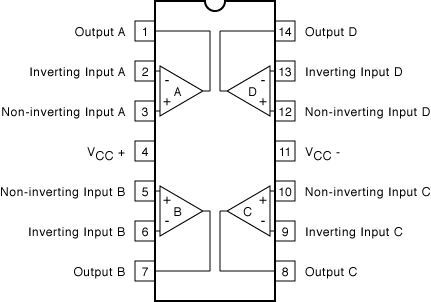
How it works: The Sensor Module

We are providing sensor modules which has 3 pairs of IR Transmitter-Receiver and 2 modules with one pair each. These Sensors can be utilized and operated to receive a non electrical data from the environment and to convert it to an electrical signal. This signal can thus further processed by the brain of our robot. So this is how it works,



Above diagram shows the schematic of one part of sensor module. Here the IR LED is connected in forward bias and the IR Receiver is connected in reverse bias. As per the intensity of light falling on it, it gives changes in the output.

When certain changes in the position of a robot and thus there are the changes in the environment around our robot, the changed conditions gives changes in the output. Like in obstacle detection mode, a considerable change in the output occurs when there is an obstacle in front of it, than when it is free from obstacles. Similarly for line tracing mode, a light reflecting back from the surfaces of different colors the sensor gives different outputs. All these properties of sensors are thus used and given to the comparator.

How it works: The Comparator

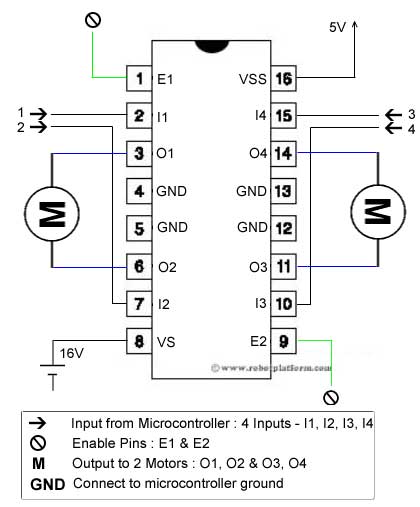
Sensors cannot itself provide output, which can solely be used as a reference to the controlling.

Also every IR Receiver provides a different output even for the same conditions and is Analog in nature. Thus a fundamental signal conditioner is necessary to turn it into identifiable. This job is done using a comparator IC LM324. IC LM324 has 4 comparator in it, so it can compare 4 sensor outputs at a time.

As shown above, one input of the comparator has given to the variable pot connected to the supply voltage, which will provide the reference comparator input. So the output of the sensor is compared with the reference voltage and the respective HIGH and LOW output is given out.

How it works: The Motor Driver

While driving the motor, motors draw large currents from the source, which is in Amperes. Thus connecting the loads directly to the control circuitry is totally impractical. Thus to control the motors from the robot brain, driving loads are needed to be isolated from it, also the high current to drive the motors has to be provided from some other source. These needs can be fulfilled by a DRIVER CIRCUITRY.



In our Robot, we will be using L293D ICs for driving the motors. The schematic diagram of a common motor driver circuit is like above. Each Motor Drives as follows:

* When both i/p are HIGH-Motor Halts
* When both i/p are LOW-Motor Halts
* When i/p 1 is HIGH and i/p 2 is LOW-Motor moves in forward direction
* When i/p1 is LOW and i/p 2 is HIGH-Motor moves in reverse direction

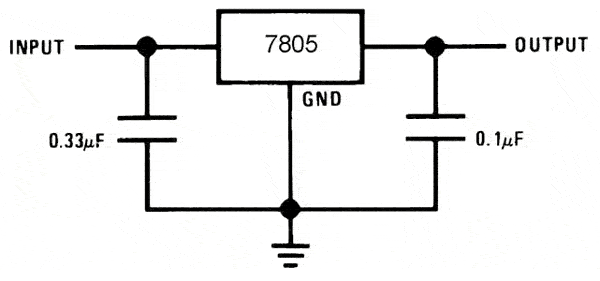
In our Robot, we’re using on Motor Driver IC for each motor by shorting out the terminals of both terminals

of IC’s. This helps to provide high load current to the motors without the IC getting damaged.

How it works: Node MCU

Miscellaneous Components

* **Voltage Regulator**

A fixed voltage regulator, IC7805 will regulate any unstable voltage in the range of 7 volts to 30 V to a constant regulated voltage of 5V. It provides both Line and load regulation to the circuit. Also it has over voltage protection and thermal shutdown in it.

* **Motor Connector**

These High Current Carrying screw connectors are used to connect the motors to the circuit. It makes it convenient to attach the motor to the circuit.

* **Battery**

We will be using a Lead acid battery of 12V, 1.2AH supply rating, a lead acid battery requires less time to charge and also lasts long. Also it has a longer lifetime than other battery types. It is cheaper than Li-ion battery and also easily available.

* **DC Motor**

The working principle behind any DC motor is the attraction and repulsion of magnets. The simplest motors use electromagnets on the shaft, with permanent magnets in the case of the motor that attract and repel the electromagnets. The reason for using electromagnets is so that it is possible to flip their magnetic field (their north and south poles).

So the electromagnet is attracted to one of the permanent magnets. As soon as it reaches the permanent magnet, its north and south poles flip so that it is repelled from that magnet and attracted to the other permanent magnet

Mounting Instructions:

* **Connecting the battery**

The provided Battery is already connected with a power plug to avoid problems. To start the bot you need to plug in the power socket. But before plugging the battery in the on/off switch if it is off or not as plugging the battery in the on the state may give rise to the surge current damaging the circuitry.

* **Calibrating the Sensors**

The sensor module provided are not most of the time pre calibrated to the comparator and to make the robot operate properly their good calibration is necessary and it can be done using the potentiometers shown in the diagram. Adjusting the reference voltage to the comparator is done by varying the potentiometers.

* **Connecting the Motors**

Green connectors as shown above are provided one from one side of each motor driver. First you have to make some space between by unscrewing it and then putting the wire coming from the motors, the screws have to be tightened. While connecting, consider checking the polarity of the motor as it may lead to the motor move in reverse direction. Also check if the screws are tightened or not as during the movement of the robot it may lead to disconnection of the motors.

* **Programming the Robot**

The robot provided is pre-programmed. But if you want to write your own code to make some changes in its functionality or simply for testing

**COMPUTER SECTION**

CONTENTS

* Programming a Microcontroller
* Storing Programs
* Working with Arduino IDE
* Code Explanation

**Programming a Microcontroller**

Nearly all factory-fresh microcontroller arrive without any program in them whatsoever. The configurable pins are usually designed to start as inputs at power-up. Without any output pins, the chip do nothing but listen to the inputs and constantly reset itself as it encounters improper (random garbage) instructions.

It’s up to you, the developer, to give each microcontroller its purpose.

Block diagram of a micro-controller: -

M  
E  
M  
O  
R  
Y

Central Processing Unit

(CPU)

IP/OP

REGISTER

ADC

USART

Timers

**Storing Programs :-**

Most of The Modern Microcontroller contain some form of internal non-volatile memory for storing programs. The program remains inside of the chip even when power is turned off or the chip is removed from the circuit. So, once you’ve programmed a microcontroller, you can

move it from circuit to circuit, power it up, and have the program starts running immediately. this really provides the sense of having a custom types chips. The two most popular reprogrammable non-volatile memory (EEPROM) and flash. These allow the chips to be programmed and reprogrammed over and over again, between ten thousand and a million times or more. The program or other memory **contents** can be read an infinite number of times and remains intact for decades.

We are going to use Devlopment board of microcontroller(WIFI module – Node MCU esp8266 (12e)) .

Working with Arduino IDE

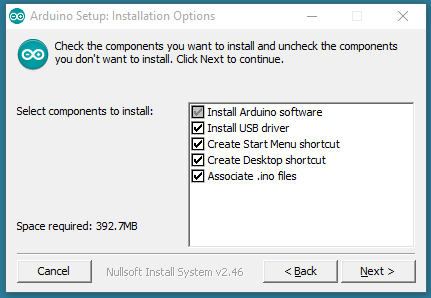
## Install the Arduino Software (IDE) on Windows PCs

This document explains how to install the Arduino Software (IDE) on Windows machines

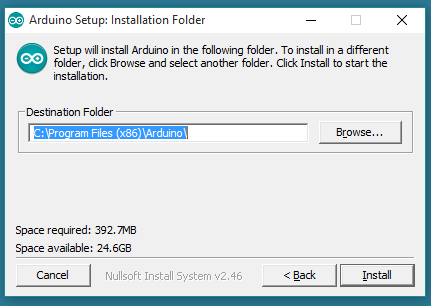
### Download the Arduino Software (IDE)

Get the latest version from the [download page](https://www.arduino.cc/en/Main/Software). You can choose between the Installer (.exe) and the Zip packages. We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers. With the Zip package you need to install the drivers manually. The Zip file is also useful if you want to create a[portable installation](https://www.arduino.cc/en/Guide/PortableIDE).

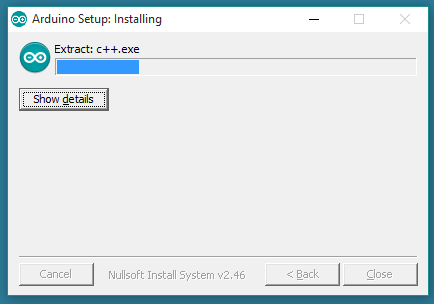
When the download finishes, proceed with the installation and please allow the driver installation process when you get a warning from the operating system.



Choose the components to install



Choose the installation directory (we suggest to keep the default one)



The process will extract and install all the required files to execute properly the Arduino Software (IDE)

### Proceed with board specific instructions

When the Arduino Software (IDE) is properly installed you can go back to the [Getting Started Home](https://www.arduino.cc/en/Guide/HomePage) and choose your board from the list on the right of the page.

Code Functionalities: -

* Motion Modules

**Forward**:

void forward(int a)

{

digitalWrite(m1dir, LOW);

analogWrite(m1pwm, a);

digitalWrite(m2dir, LOW);

analogWrite(m2pwm, a);

}

**Backward:**

void back(int a)

{

digitalWrite(m1dir, HIGH);

analogWrite(m1pwm, a);

digitalWrite(m2dir, HIGH);

analogWrite(m2pwm, a);

}

**Right:**

void dright(int a)

{

digitalWrite(m1dir, HIGH);

analogWrite(m1pwm, 1024-a);

digitalWrite(m2dir, LOW);

analogWrite(m2pwm, a);

}

**Left:**

void dleft(int a)

{

digitalWrite(m2dir, HIGH);

analogWrite(m2pwm, 1024-a);

digitalWrite(m1dir, LOW);

analogWrite(m1pwm, a);

}

/\*m1dir,m2dir are two direction inputs to the motors which decides the direction of the motors .m1pwm,m2pwm are two PWM inputs ,which decides the PWM of the Motors. Value of PWM decides the duty cycle generated. As the modules is ininfinite loop , it will produced continuous PWM to the motors. All the above four modules, Using combinations of the PWM and direction will generate 4 different direction of the motors , that are Forward,Reverse,Backward,Right. \*/

* Obstacle Avoider Module

obst=analogRead(A0);

if(obst<200)

{

stopp();

obst=analogRead(A0);

Serial.println(obst);

delay(10);

client.flush();

}

/\*For Obstacle Avoider, Sensor output is given to the A0 pin and if this pin sets to high then the previously set forward motion will be stop.If the obstacle is out of its way the robot will be set again in the forward motion \*/

* Line Tracing Module: -

void linetrace() {

pinMode(16,INPUT);//lmid d7

pinMode(5,INPUT);//lleft d1

pinMode(15,INPUT);//lright d5

pinMode(A0,INPUT);//obst

lmid=digitalRead(16);//d0

lleft=digitalRead(5);//d1

lright=digitalRead(15);//d8

//Serial.println("linetracedata====================================================");

//Serial.print(lmid);

//Serial.print(lright);

//Serial.print(lleft);

if(lleft==1&&lmid==1&&lright==1){

//all high 111

//ctr++;

//if(ctr%2==0)

//left(900);

//else

forward(800);

yield();

}

else if(lleft==1&&lmid==1&&lright==0){

//110

digitalWrite(m1dir, LOW);

analogWrite(m1pwm, 750);

digitalWrite(m2dir, HIGH);

analogWrite(m2pwm, 350);

ai=1;

yield();

}

else if(lleft==1&&lmid==0&&lright==1){

//101

forward(800);

yield();

}

else if(lleft==1&&lmid==0&&lright==0){

//100

digitalWrite(m2dir, HIGH);

analogWrite(m2pwm, 350);

digitalWrite(m1dir, LOW);

analogWrite(m1pwm, 750);

ai=1;

yield();

}

else if(lleft==0&&lmid==1&&lright==1){

//011

digitalWrite(m1dir, HIGH);

analogWrite(m1pwm, 350);

digitalWrite(m2dir, LOW);

analogWrite(m2pwm, 750);

ai=0;

yield();

}

else if(lleft==0&&lmid==1&&lright==0){

//010

forward(700);

yield();

}

else if(lleft==0&&lmid==0&&lright==1){

//001

digitalWrite(m1dir, HIGH);

analogWrite(m1pwm, 350);

digitalWrite(m2dir, LOW);

analogWrite(m2pwm, 750);

ai=0;

yield();

}

else{

//000

if(ai = 0)

{

digitalWrite(m2dir, LOW);

analogWrite(m2pwm, 750);

digitalWrite(m1dir, HIGH);

analogWrite(m1pwm, 350);

}

else if(ai = 1)

{

digitalWrite(m1dir, LOW);

analogWrite(m1pwm, 750);

digitalWrite(m2dir, HIGH);

analogWrite(m2pwm, 350);

}

yield();

}

yield();

}

/\*In this module the robot follows the white line .Conditional IF ,ELSE statement is used to check the position of the Robot. Conditional If Else Statement are used to check the position of the robot. Input is obtained on digital pins. It thus with respect to the position with reference to the line is deciding the direction of motors using the Above speed modules.It even have AI.\*/

* **AUTOMATED HOME EXTENSION BOARD**

void homeauto(){

if(cmd=="m"||cmd=="n"||cmd=="o"||cmd=="M"||cmd=="N"||cmd=="O")

{

oldc=cmd;

}

else

{

return;

}

if (oldc == "n"){

inv1=!inv1;

digitalWrite(5,inv1);

}

else if (oldc == "o"){

inv2=!inv2;

digitalWrite(16,inv2);

}

else if (oldc == "m"){

inv3=!inv3;

digitalWrite(15,inv3);

}

else if (oldc == "N"){

inv1=!inv1;

digitalWrite(5,inv1);

}

else if (oldc == "O"){

inv2=!inv2;

digitalWrite(16,inv2);

}

else if (oldc == "M"){

inv3=!inv3;

digitalWrite(15,inv3);

}

sendBackEcho(oldc);

//oldc = "\0";

delay(1);

client.flush();

yield();

}

/\*Automated home extension board is based on IOT(Internet of Things).It just check the command send and each and every command is given a unique letter by comaparing these letters we specify the button to switch ON or OFF. \*/

* Reference Links: -
* [www.societyofrobots.com](http://www.societyofrobots.com)
* [www.electronicsteacher.com](http://www.electronicsteacher.com)
* [www.extremelectronics.co.in](http://www.extremelectronics.co.in)
* [www.avrfreaks.com](http://www.avrfreaks.com)
* [www.robotics.org](http://www.robotics.org)
* [www.robots.net](http://www.robots.net)
* [www.active-robots.com](http://www.active-robots.com)
* [www.robotshop.com](http://www.robotshop.com)
* [www.instructable.com](http://www.instructable.com)
* [www.hackaday.com](http://www.hackaday.com)
* [www.wired.com](http://www.wired.com)
* www.arduino.cc
* developer.android.com
* www.esp8266.com

Checklist for items in Kit:

* 2 Motors
* 2 Wheels
* 2 Clamps
* 1 Chassis
* 1 Caster Wheel
* 1 Line Tracer/Edge Detector Module
* 1 Obstacle Detecting Sensor Module
* 1 Motherboard
* 1 Node MCU
* 12 V Battery
* One 6 pin Relimate Connector wire
* One 3 pin Relimate connector wire
* One 2 pin Power Relimate
* 5 spacers- 35mm
* 4 spacers- 50mm
* 4 spacers- 9mm

**Contact Us:**

Electronics:

Ritvik Joshi: 8805356252

Aniruddha Damle: 9823178270

Software:

Devavrat Joshi: 9881494971

Mechanical:

Sumedh Raghoji: 8411917140

Mohit Malpani: 9850094710