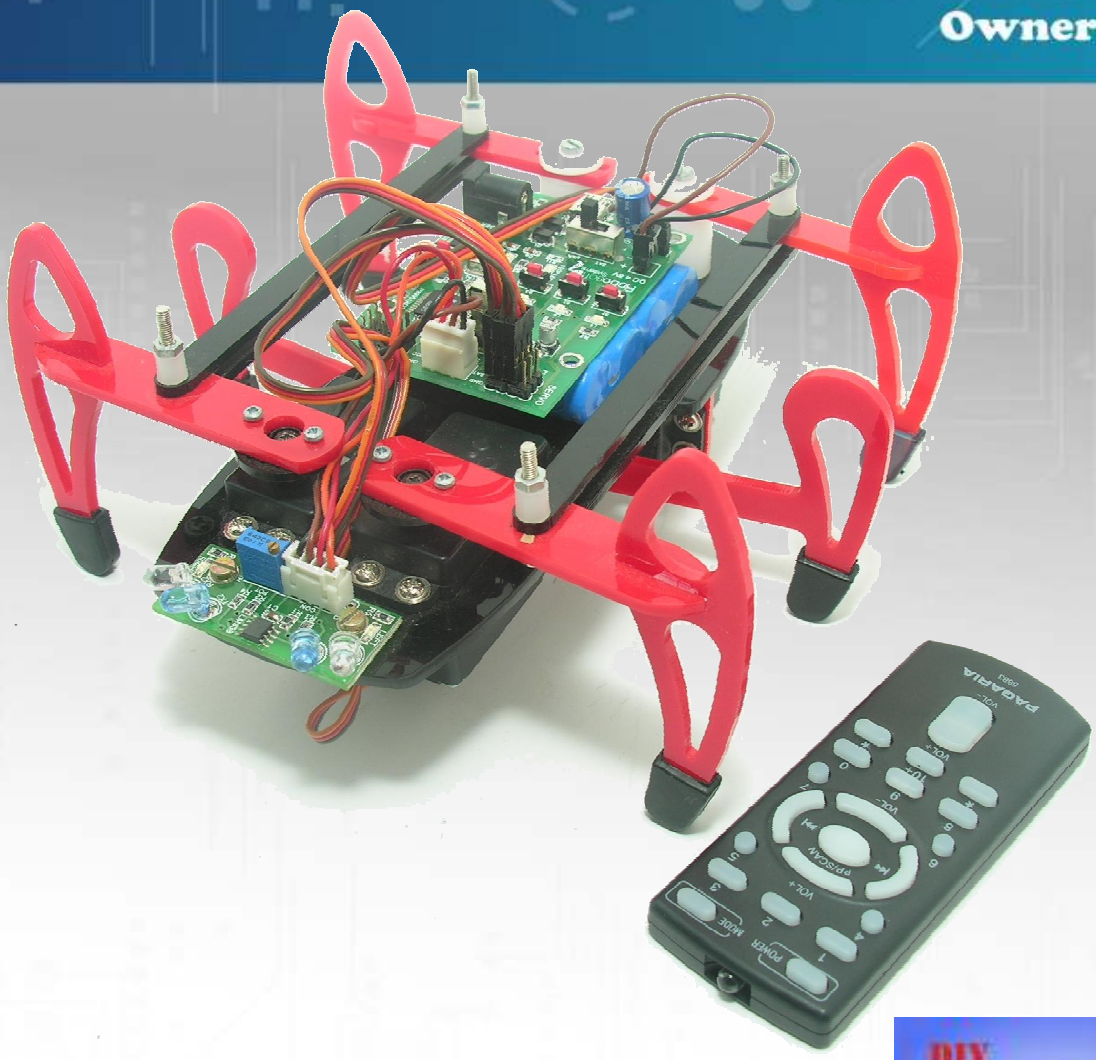
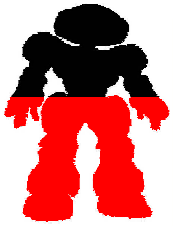
Robosoft Systems

HexaBOT

Owners Manual



Robosoft Systems



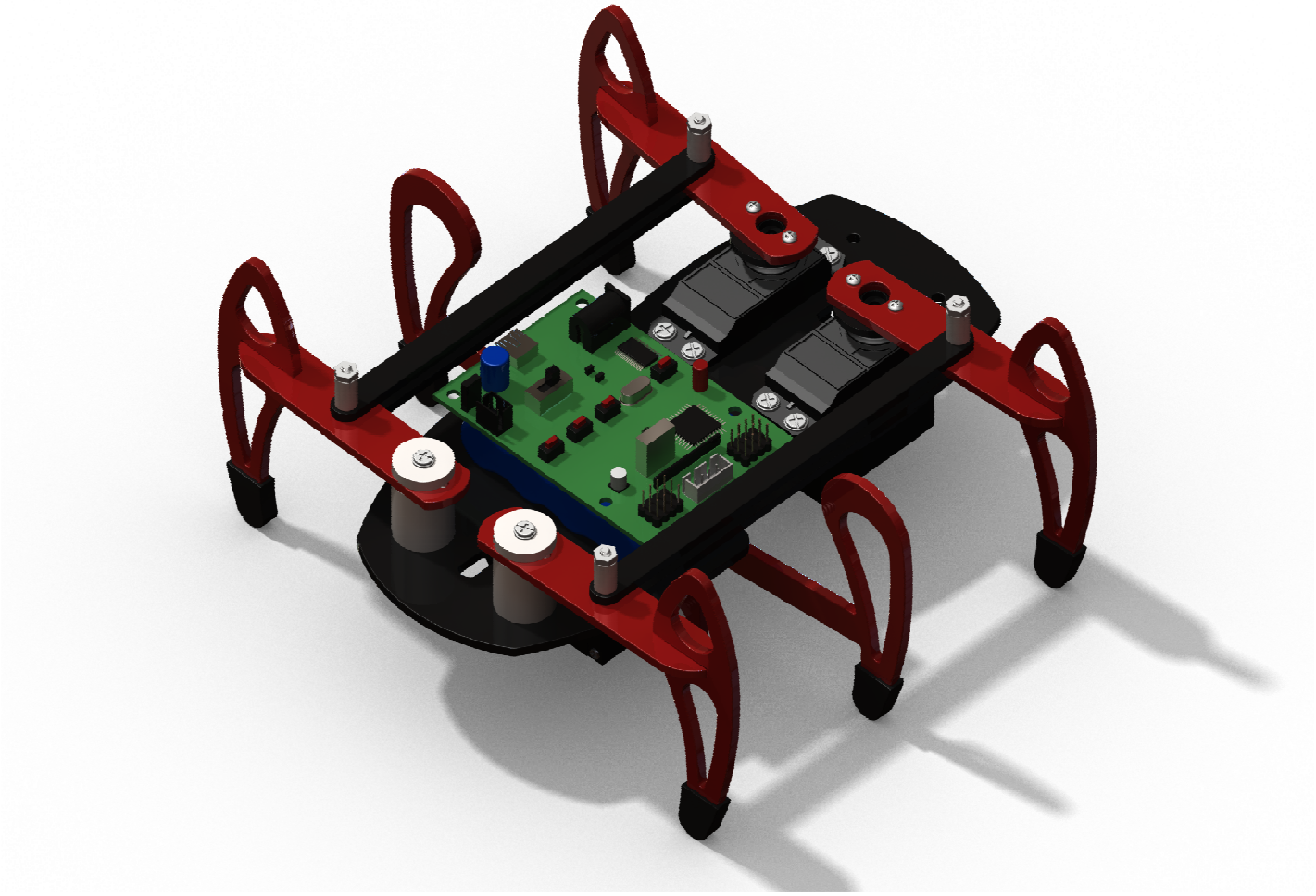
Skill Level:
Expert

Soldering
Not Required

4 hrs

HexaBOT

Owner's Manual



HexaBOT is not a toy.

Robosoft Systems® employs high performance motors and electronic equipments.

Robosoft Systems® cannot be held liable for damages or injuries caused by the improper or un recommended use of HexaBOT.

All advanced modifications or deviations from the directions contained herein are considered to be at the risk of the HexaBOT

This is the HexaBOT Owners Manual v2.00 for the robot HexaBOT.

For more information go to http://www.robosoftsystems.co.in

or contact us at support@robosoftsystems.co.in

Conventions

This symbol indicates that improper use can lead to physical injury or damage to the robot itself. It urges you to exercise precaution.

This symbol indicates that there is some additional information or tip that you can use.

This symbol indicates that you should make a note of the point.

Important Note

This manual refers to the product HexaBOT. All the shown components will not be available if the product is bought in parts. Nevertheless, you can refer to the specific section as that of the parts that you have bought to know the assembly details. The components shown in this manual may not be the same as those in the packaging and may change without notice.

Please note that the manual is not comprehensive and cannot be considered as a complete reference for any theories presented here. Please refer to the compact disc that has accompanied the product for a more detailed overview of the various technologies and programming language that has been put to use in this product.

For further details and supported accessories, please visit our website http://www.robosoftsystems.co.in/roboshop.

Contents

About Robosoft Systems	01
HexaBOT Features	02
Introduction	03
Safety Instruction	04
Package Content	05
Construction of HexaBOT	07
Servo Motors	18
Know Your Board	22
How to Do Connection	24
How to Charge Battery	26
How to fix Batteries	27
Programming Instructions	28
Programming Motion of HexaBOT	32
Navigation using IR Remote	36
HexaBOT PRO	37

About Robosoft Systems ®

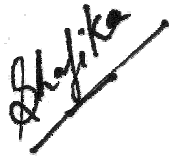
Robosoft Systems ® is a young & dynamic organization. It all started with a vision back in 2008 when a group of engineers bitten by the entrepreneurship bug came together for transforming their ideas into reality, & it had to be in the vibrant field of Robotics just like the young minds themselves.

Just two years into its inception & the company is creating waves in the design and manufacture of various robotics kits, components and niche industrial products. In such a short span we already have a wide ranging portfolio with an untiring focus on quality and providing more value for money. On the technical front the company has had path breaking success lead by a dedicated team of R&D engineers. The company has been launching products defying cutting edge technology. This pursuit of excellence has won the company many accolades. The Indian media has taken note of our achievements and have featured us in prestigious platforms such as Business Standard, Economic Times, TOI and ET NOW.

We have a dream of inculcating robotics as an education platform to prepare the children of India for a highly technical world. To this end we have a range of edutainment products – we call them as - Construction or Do It Yourself - kits. We have with these kits, conducted robotics workshops in various schools and technical colleges around the country. Through these workshops the kits have gone through a long product development cycle and have become finished products in all respects. These kits share a common thread of introducing robotics to the technology enthusiast and to break the notion – that robotics is for the geeky elite. We at Robosoft believe that Robotics is an integrator of science & applied technology and this belief is reflected in our robotic kits.

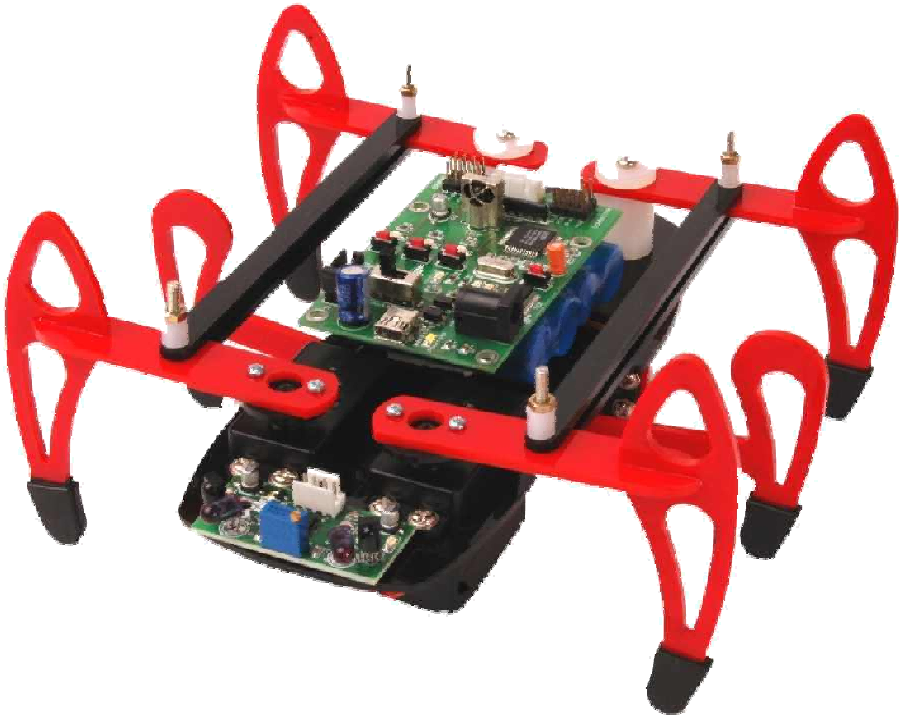
Thus we humbly bring to you, a range of engaging and curiosity inducing products. Our kits range from wheeled to legged motion and from mechanical to programming modules. Through this door awaits a world of robotics wonder – turn the knob to enter.

Managing Partner



Robosoft Systems®

HexaBOT Features



- **Philips P89V51RD2 SMD µc on board.**
- **Autonomous robot.**
- **Completely mobile robot.**
- **ISP (In System Programming) support.**
- **Easy USB interface.**
- **On board provision for battery charging.**
- **Accessories: battery, adaptor, software cd.**
- **Compact size.**
- **Separate detachable TSOP based obstacle avoiding module.**
- **Three servo motors for accurate movement.**
- **6V DC battery for continuous operation.**
- **IR remote for wirelessly controlled motion.**

Introduction

This manual has been written keeping in mind robotic enthusiasts from all age groups. It will take you right through the building steps of the HexaBOT to controlling it.

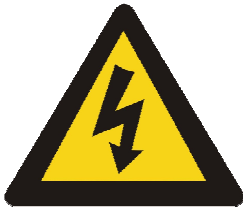
The HexaBOT is a basic level robot from Robosoft Systems®. It helps you to get a feel of what robotics is all about. The HexaBOT has been developed by bringing together the laws & concepts of Physics, Electronics & Mechanics hence it helps one to inherit this knowledge as one is implementing it.

The HexaBOT is basically a robot which uses legged locomotion. It is quite evident from the name that it will be six legs for locomotion. The robot can work as manual controlled or it can be autonomous also. Speciality of HexaBOT is ,It is using only 3 servo motors to control 6 legs. MCS51 (P89V51RD2) based controller based board is used for controlling servo motors hence to control HexaBOT. The TSOP sensor based obstacle avoiding module helps machine roam around room and avoiding obstacles. HexaBOT can also be controlled using IR remote. Your microcontroller board can be programmed using flash tool. It is easy to install and easy to use. Simple USB interface allows board to connect with PC or Laptop. Prior knowledge to MCS51 systems will allow user to use the controller board for embedded systems development.

Statutory Warning:

This is a highly addictive robotics kit; please do your school homework before starting.

Safety instructions



Be careful while using the screw driver, place the machine on a platform and then tighten the screws. Don't tighten the screws keeping the assembly in your hands, as the screw driver may slip and you may injure your hands.

Don't work on your machine with wet hands. The machine is having electronics parts & battery, hence there are chances that you may get an electric shock(It may cause severe injuries or loss of life).

The components on the electronic boards are having soldering joints at the bottom end which are vulnerable to short circuit, to avoid such a situation don't place the circuits on a conducting surface when the supply voltage is there, otherwise there can be a short circuit between the soldering joints causing permanent damage.

The battery provided to you is having two wires, please don't short the wires together, if you do so that will discharge the battery quickly. This can damage the battery in the long run reducing its life.

The microcontroller board is very fragile. Please take appropriate care when handling the microcontroller board. It may break if you apply more pressure than you ought to.

Package contents

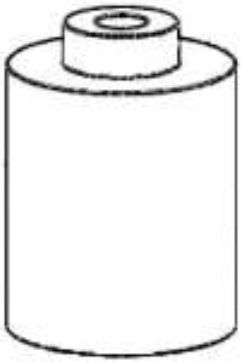

X 22

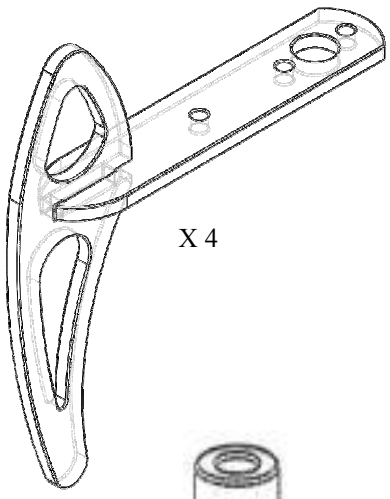

X 2


X 3

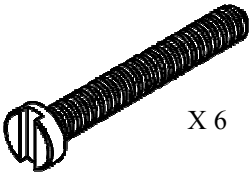

X 6

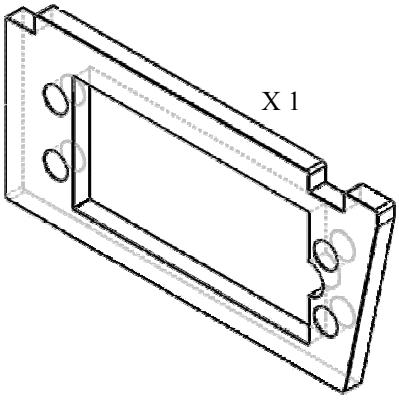

X 16

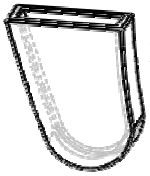

X 2


X 4

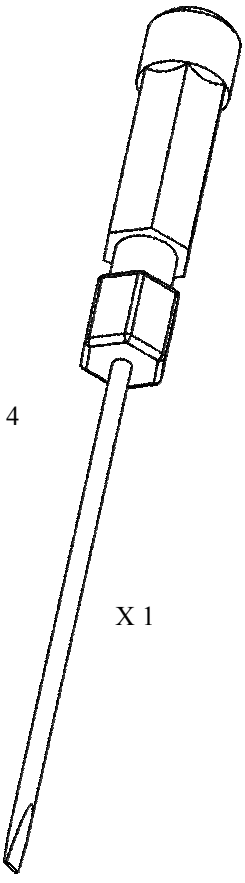

X 3

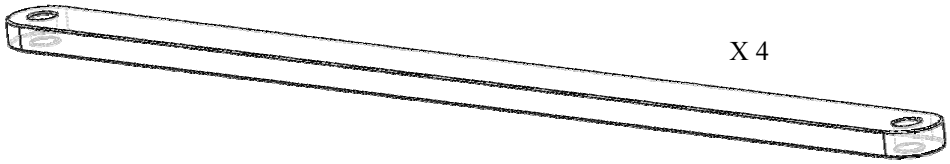

X 6

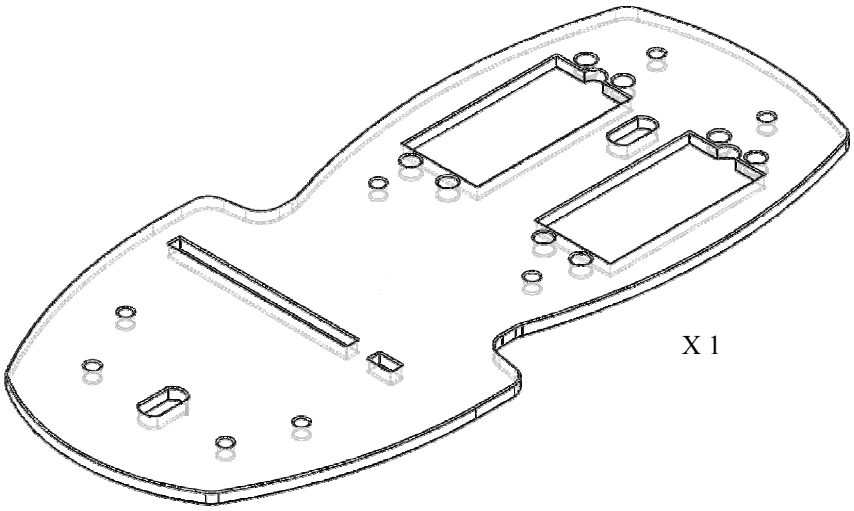

X 1


X 6

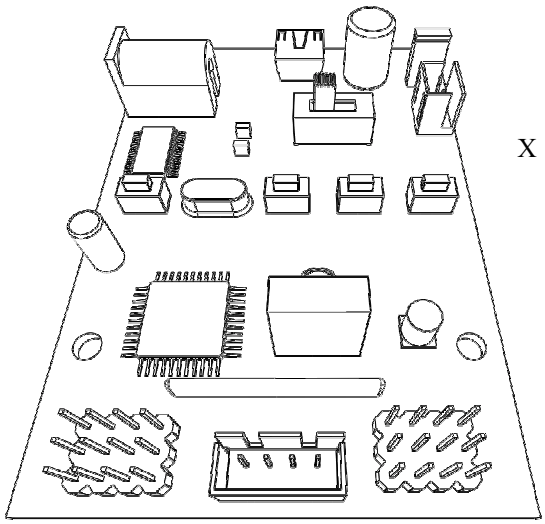

X 4


X 1

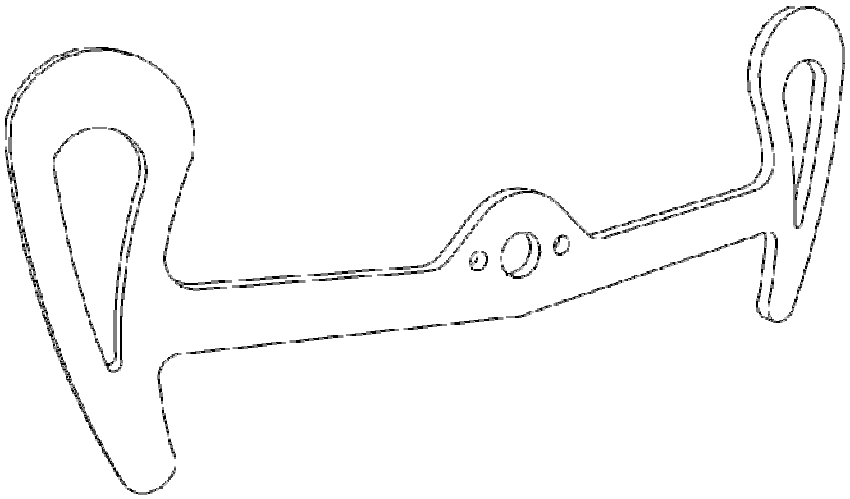

X 4



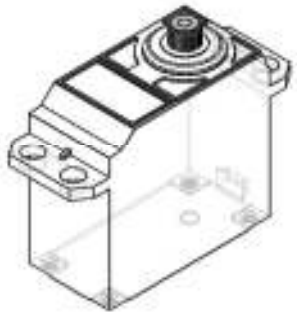
X 1



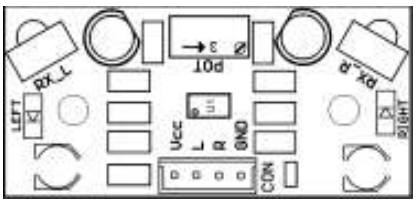
X 1



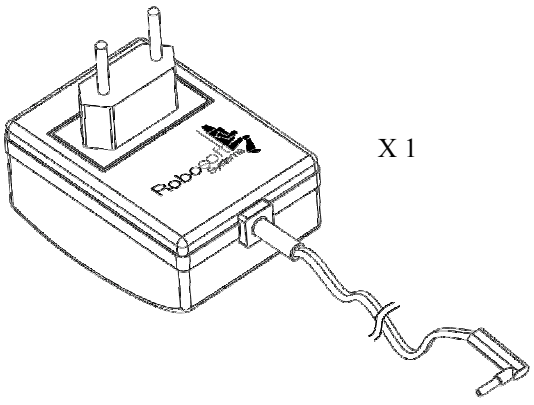
X 1



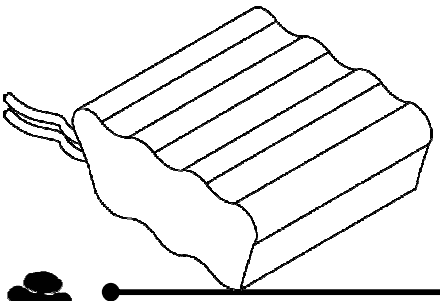
X 3



X 1



X 1



X 1

Construction Of HexaBOT

Below we list all the steps for assembling the machine. If any explanation is required, then we will give them at appropriate places. Legs of HexaBOT are pre assembled and fixed with fevi kwick. If your package contents legs in un fixed then follow the procedure.

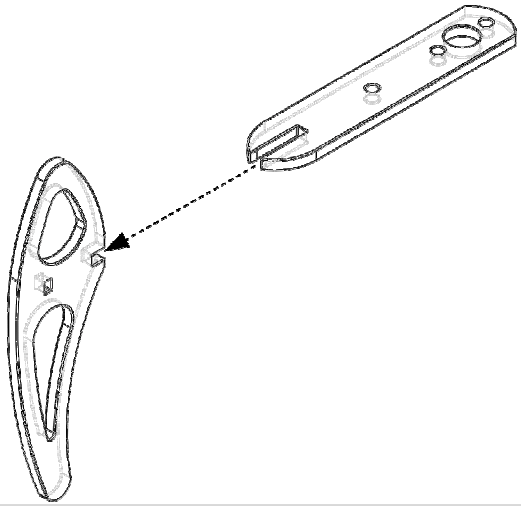


Fig 1

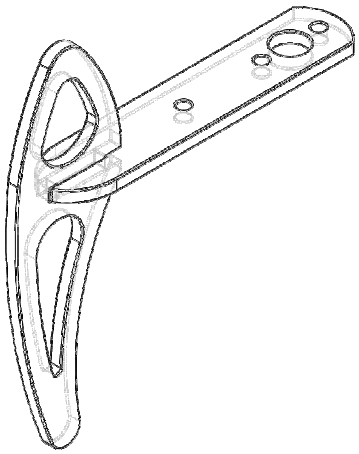


Fig 2

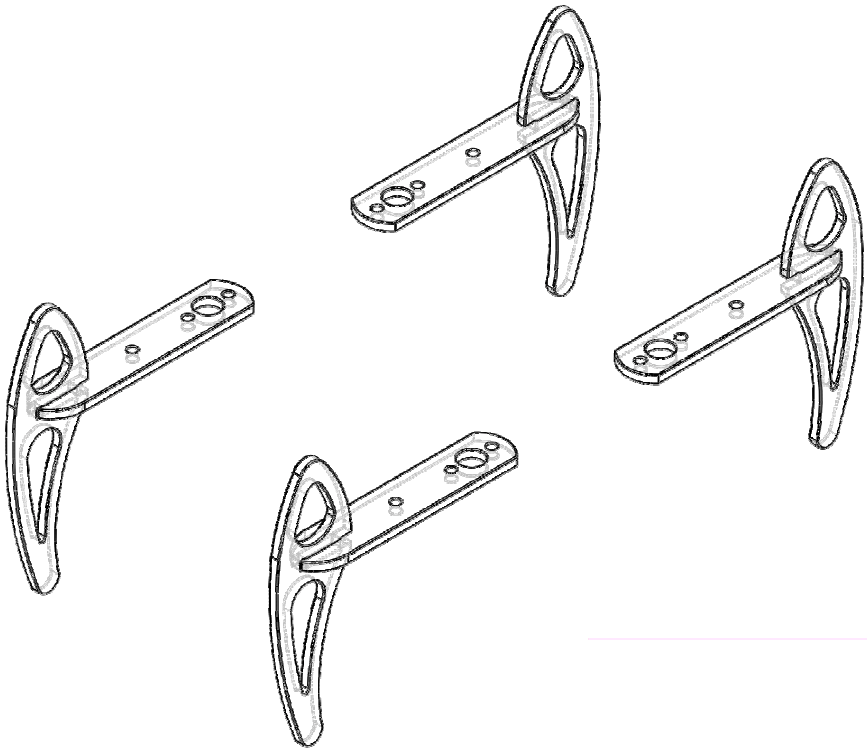


Fig 3

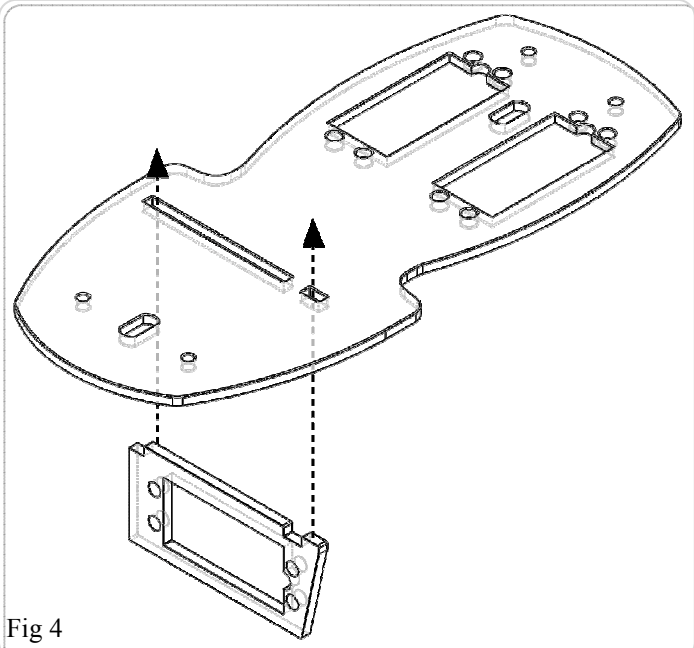


Fig 4

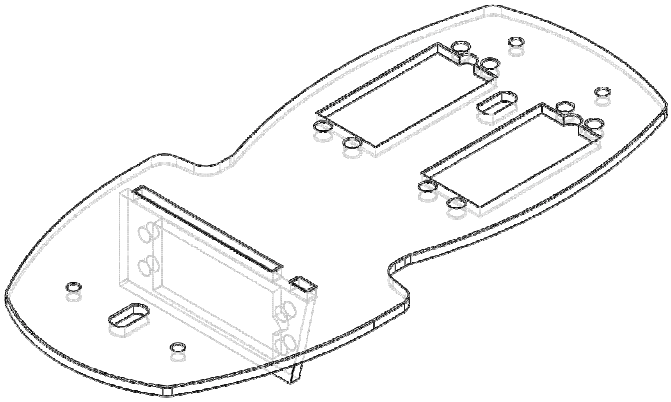


Fig 5

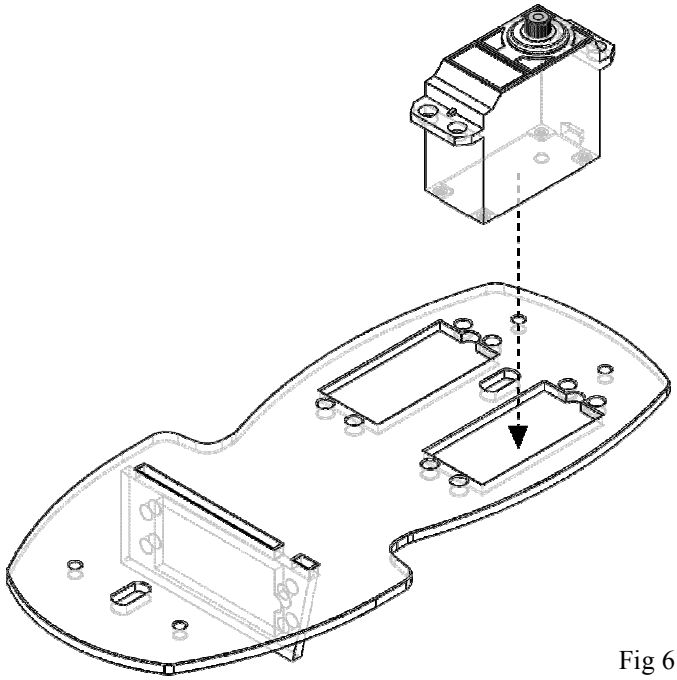


Fig 6

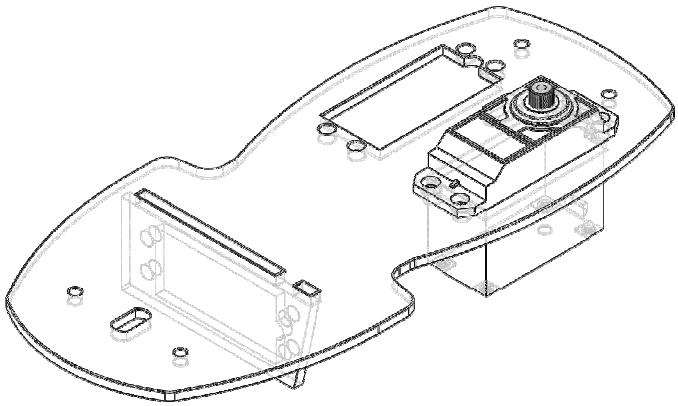


Fig 7

In the steps that we have shown, you must have noticed that we have not used any screws to fix the legs of the HexaBOT. To fix these legs, you have to use any industrial strength adhesive like FeviKwik®. But please take care to fix the legs in exactly 90°. If you fail to do so, then your robot will not follow a straight line while moving forward or backward. Also take care not to glue your fingers to the acrylic parts.

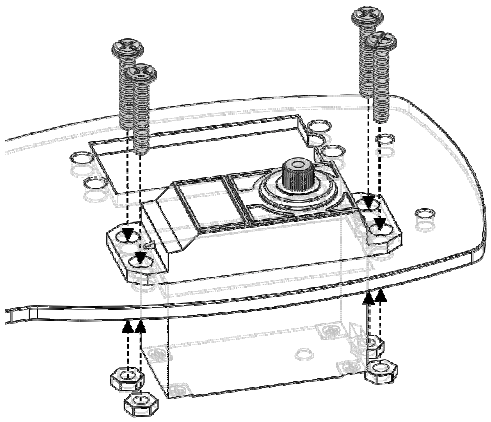


Fig 8

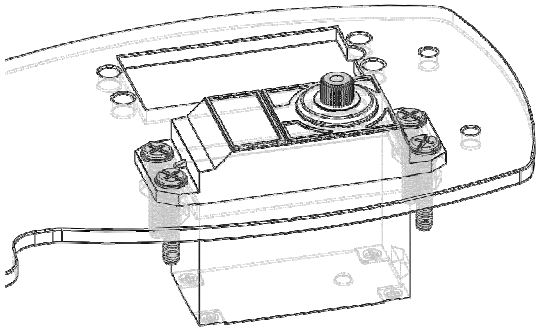


Fig 9

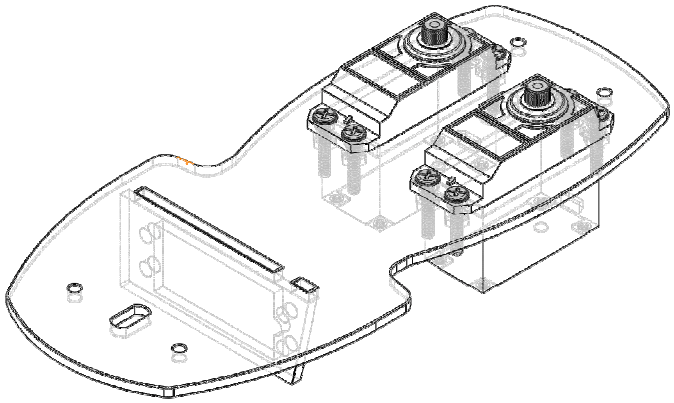


Fig 10

To fix these motors, you have to keep in mind the direction that you are putting them in. You must have noticed that there is only one side of the motor which has the wires coming out of it. The set of three wires originate through a rubber stub. On the board, we have provided notches on one side of the motor holes. The rubber stubs will go through those notches. Do not exert more pressure than required.

The easiest way to glue the acrylic components together is to first attach the components with your hands in exactly the way you want and then apply FeviKwik® on the corners where the components meet each other.

Be extra careful when you glue the red acrylic rectangular frame to the black acrylic body. Sometimes the rectangular part refuses to fit snugly into the black body. Give very light pressure when you try to fit that component in. Once when you have put that in, just apply FeviKwik® from the top layer. Also you have to keep in mind that the shaft of the motors will come on the opposite side of the Robosoft Systems® logo.

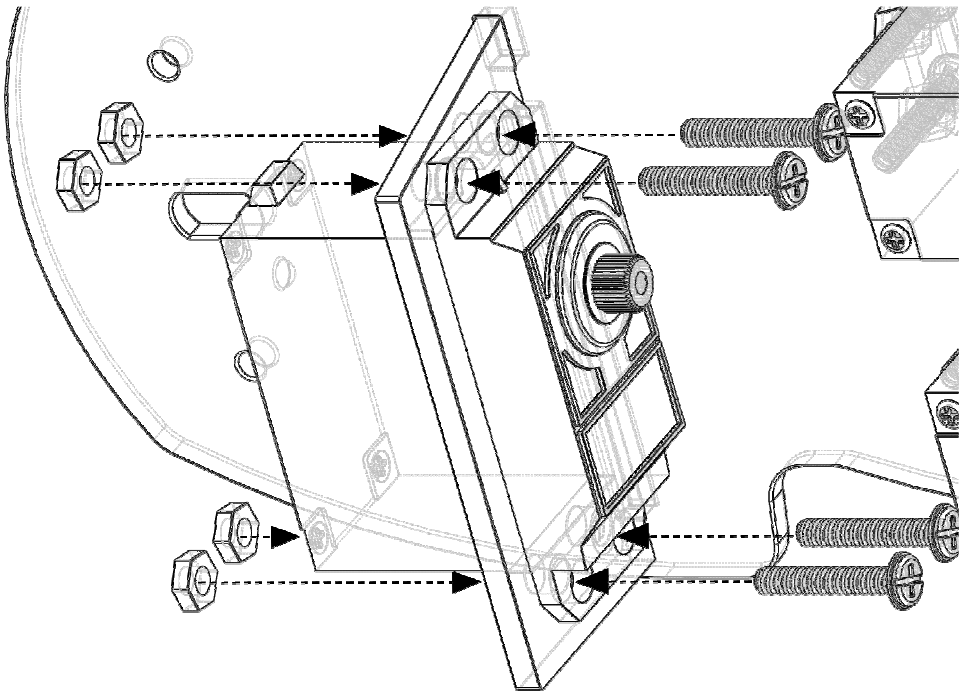


Fig 11

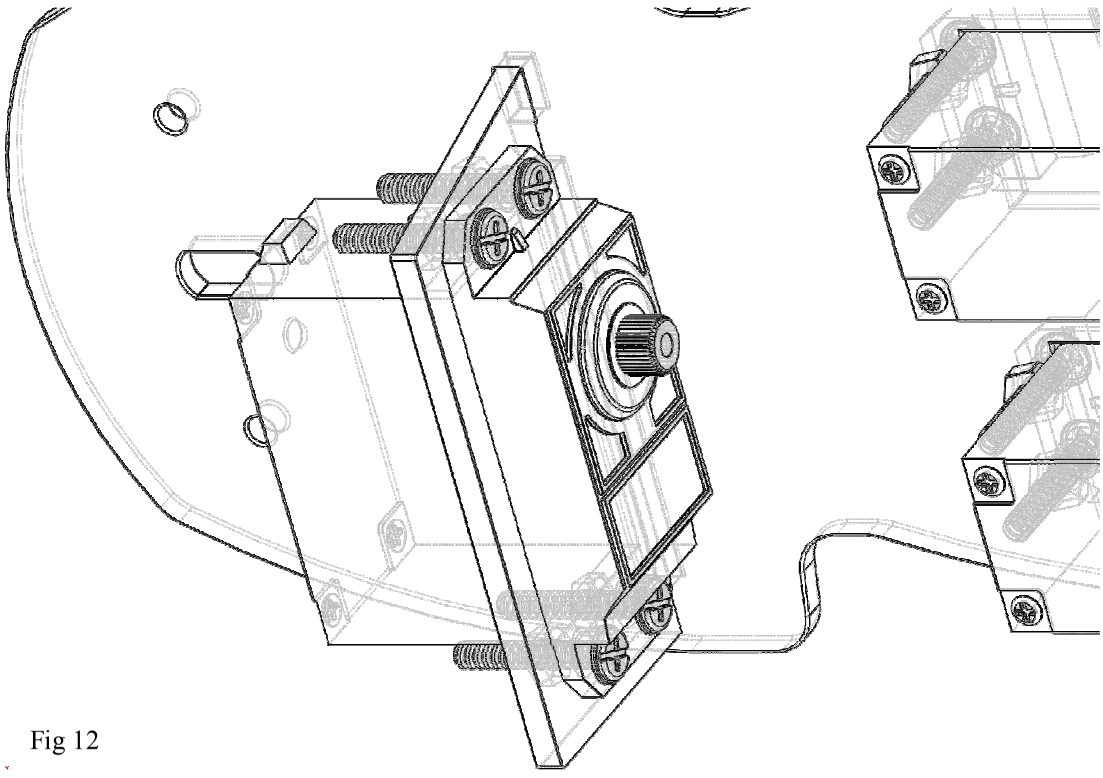


Fig 12

Besides we have shown two figures which gives you an exploded view from the bottom of the robot. It shows you how to mount the motor from the bottom of the robot into the red rectangular acrylic frame. You have to keep in mind that the shaft of the motor will face the other two motors. Similar to the other two motors, we have provided a notch for the wire of the motor to go in.

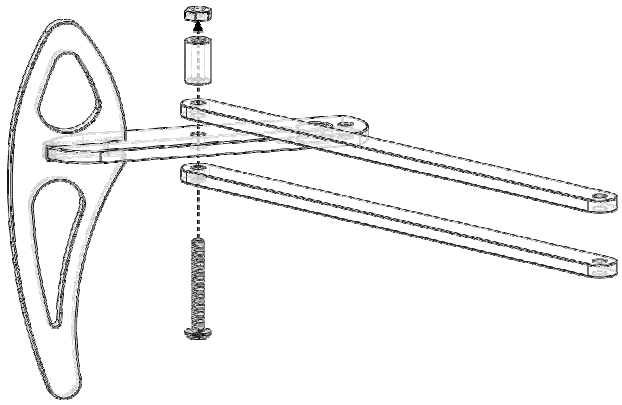


Fig 13

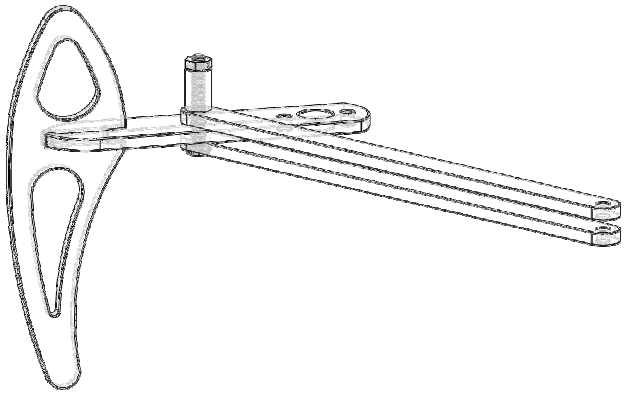


Fig 14

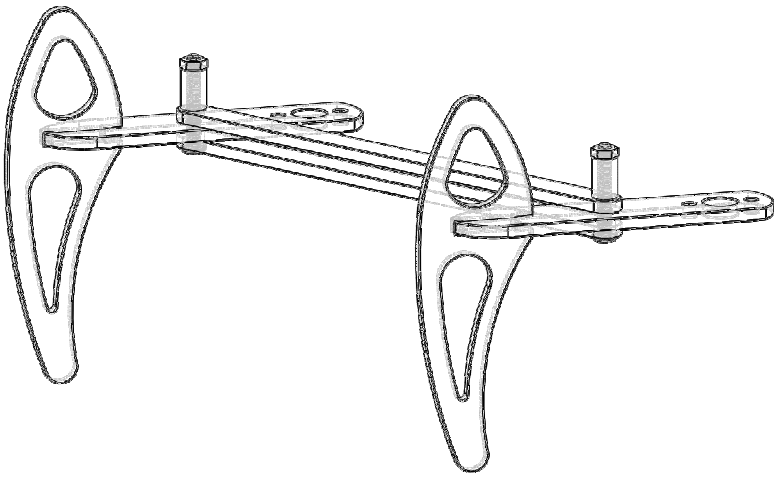


Fig 15

Here we have listed put the steps to mount the black acrylic strips on the legs. For this you will need the smaller white plastic spacers. The screws that you will require, are, 3X15mm and 3mm nuts. The screws will come from the bottom and the spacers from the top. After that you will have to tighten the nuts from above. But while tightening the nuts, do not over tighten them; otherwise the robot will lose flexibility.

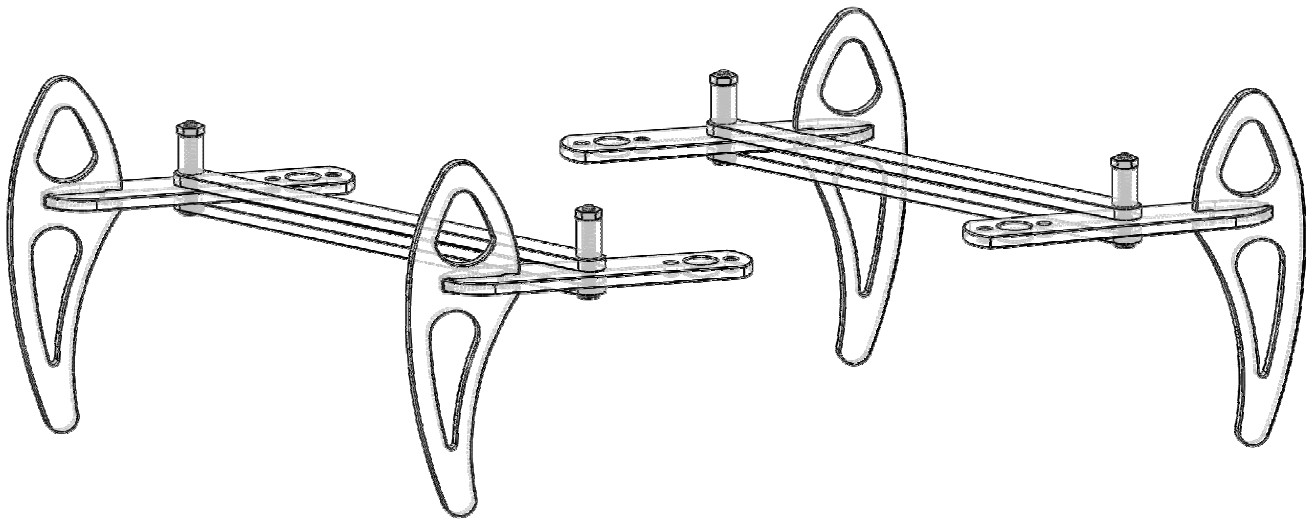


Fig 16

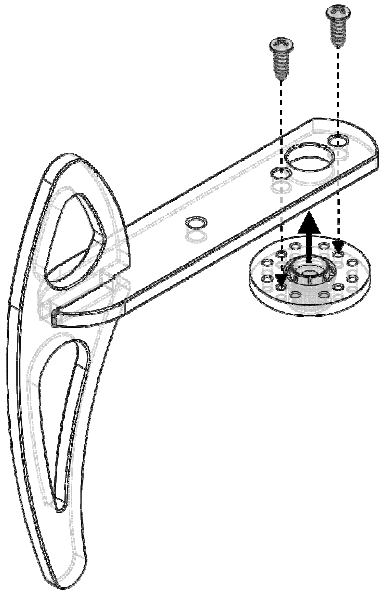


Fig 17

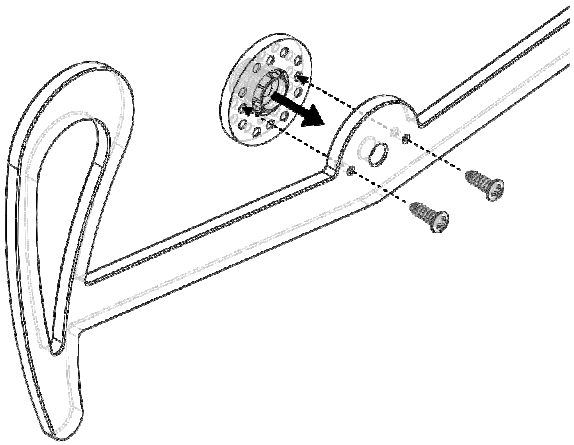


Fig 18

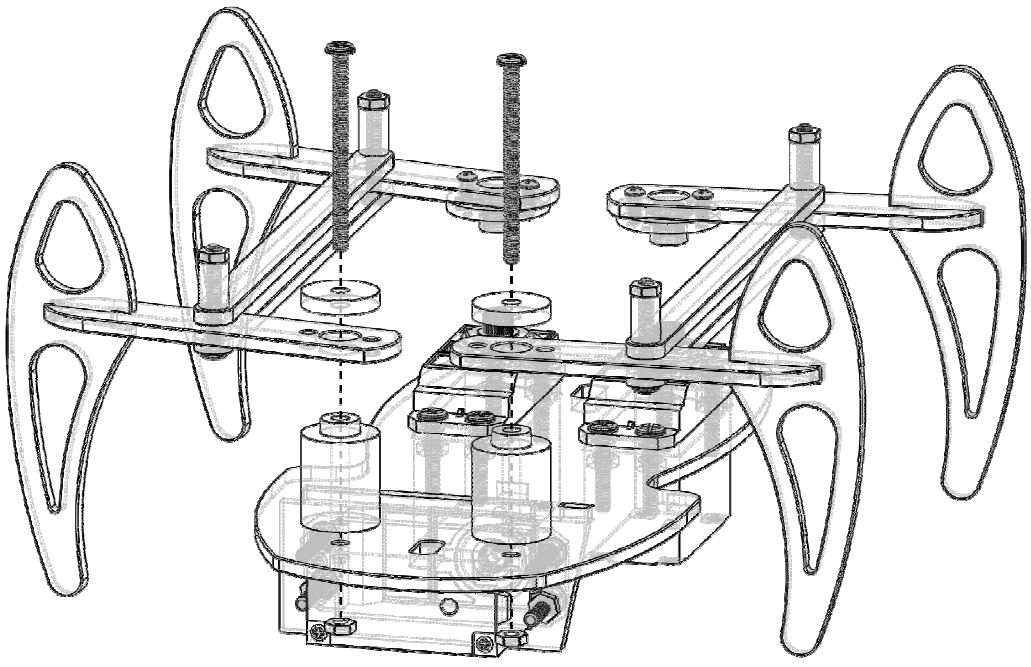


Fig 19

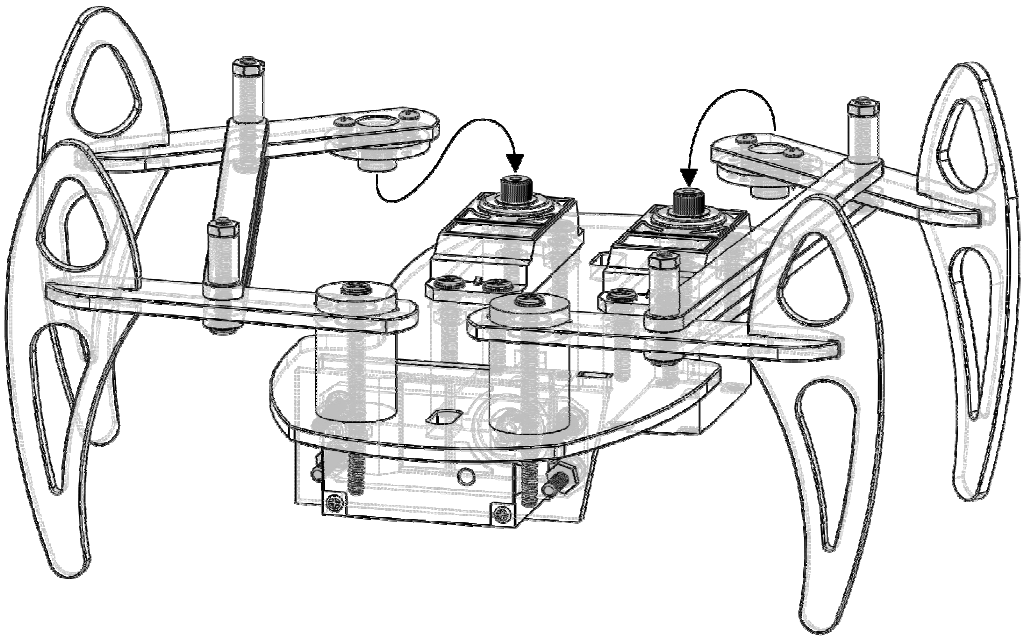


Fig 20

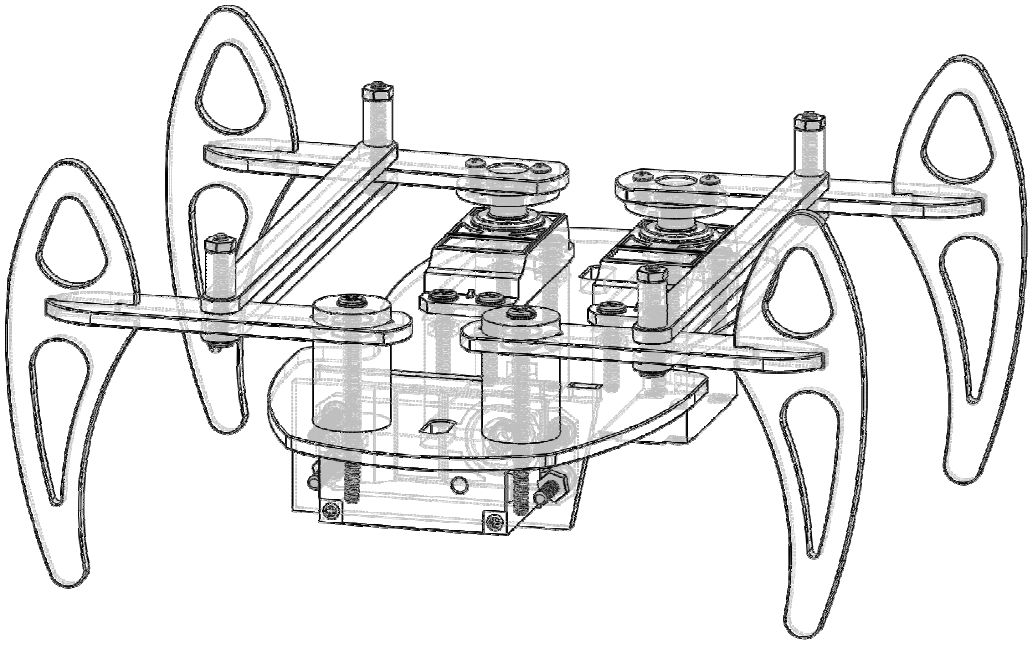


Fig 21

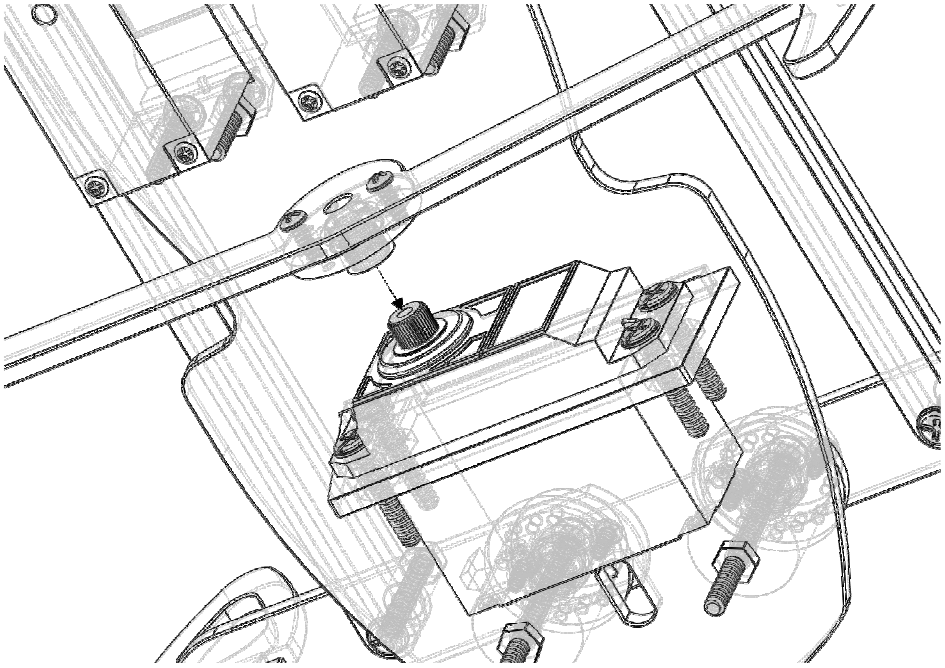


Fig 22

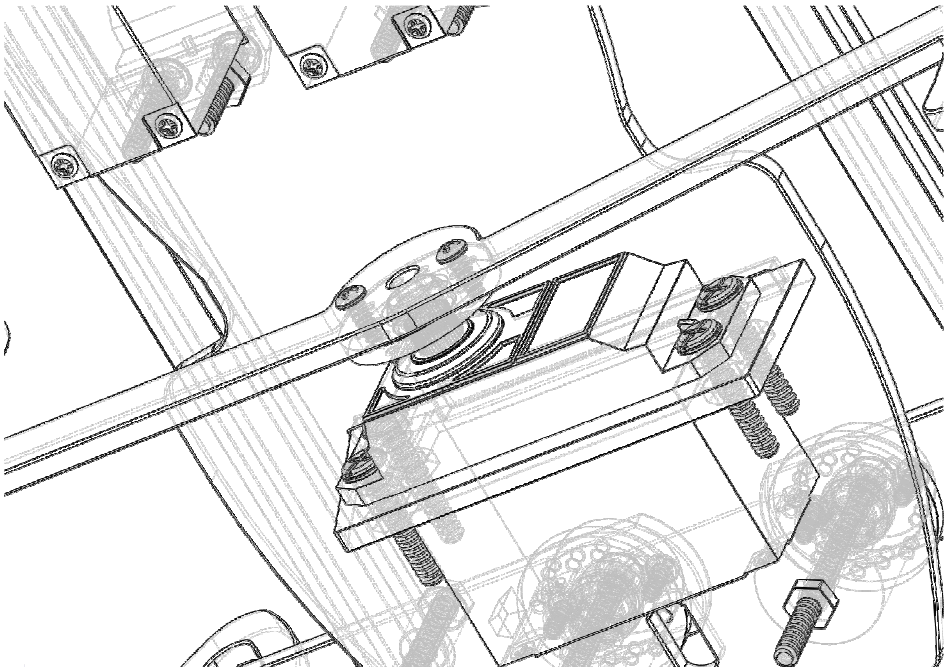


Fig 23

The fig. 22 and 40 shows how you can attach the bottom leg to the bottom servo motor.

The fig. 24 and 25 show the final steps in assembling the machine. After everything else is complete, you will have to put the black rubber shoes onto the legs of the HexaBOT. Fig. 42 shows the final assembled view of the machine as everything else is complete. Only the thing that is remaining is to mount the electronic board and the battery to the machine and your machine will be ready. You will just need to program the machine to perform various kinds of motions.

Fig 24

Fig 25

Important note

Once you finished with all mechanical assembly the last thing you have to do is fix Black servo screws. But if you fix it when servo is in wrong position then you can damage your servo and your HexaBOT assembly too. To avoid that follow these instruction.

1.Make sure you have not fix black screw in your servo motor shaft.

2. Remove Black rings from motor shaft. Don't remove self threading screw holding HexaBOT legs to plastic ring.

3. Download Servo Rest code from resources CD to your microcontroller board.

4.Once you load that program and connected battery to board then all motor will be at 90 degree. That is considered as resting position for all servos.

5. Then only fix all black servo screws such that while in rest left and right leg perpendicular to HexaBOT and middle legs both legs touching ground.

To know more about connection refer know your Board section and how to do connection section. Excessive pressure on servo can damage your servo.

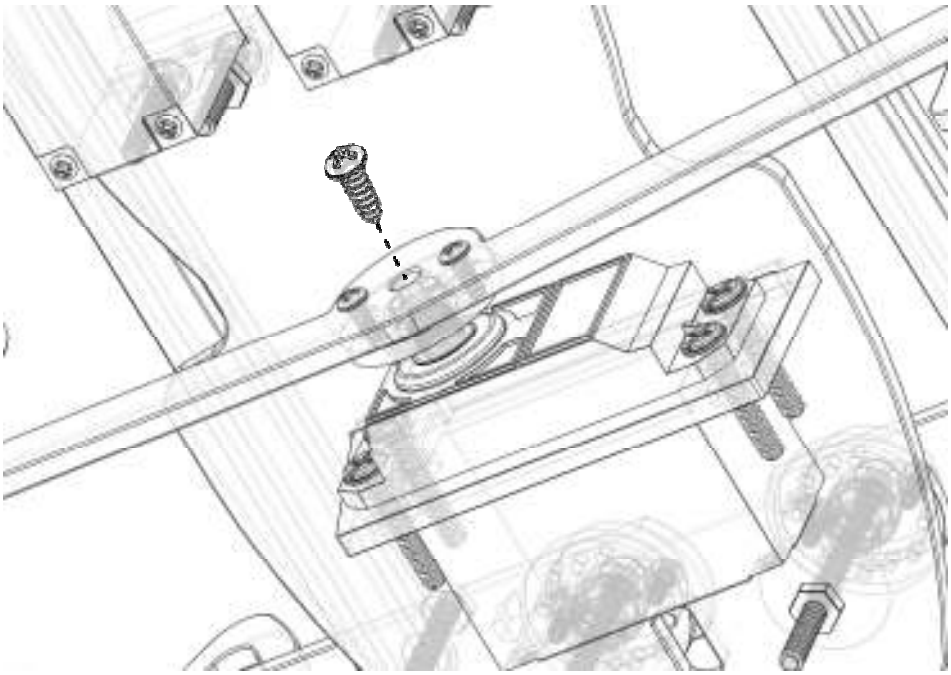


Fig 26

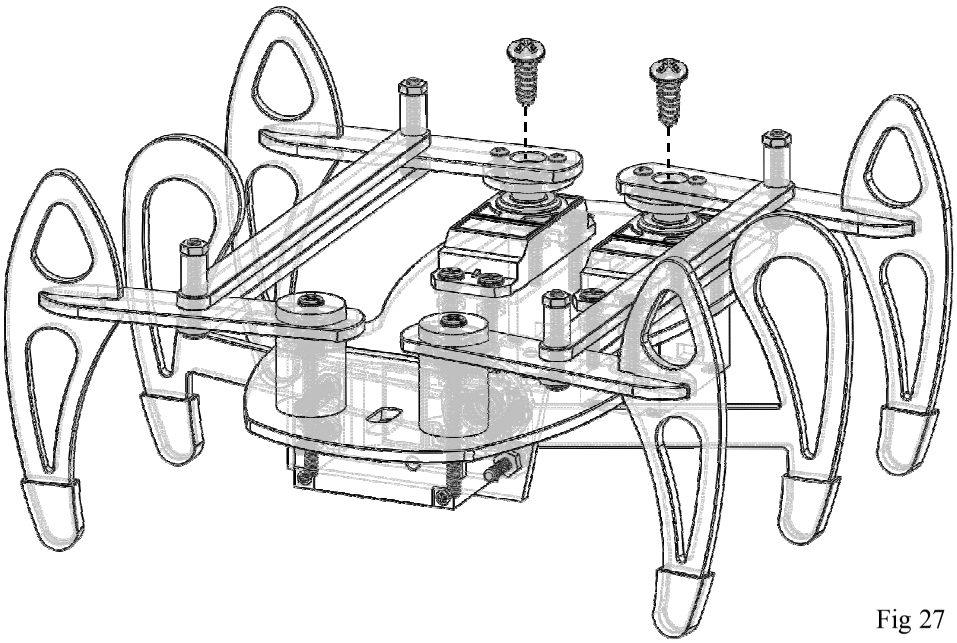


Fig 27

This is a last step in mechanical assembly of your HexaBOT .But make sure you have read instruction of previous page. And all 3 servos are in resting mode. A single mistake can damage your whole HexBOT. So be care full while doing this assembly.

Servo Motors

A "servo" is a generic term used for an automatic control system. It comes from the Latin word "servus" - slave. In practical terms, that means a mechanism that you can set and forget, and which adjusts itself during continued operation through feedback. Disk drives, for example, contain a servo system ensuring that they spin at a desired constant speed by measuring their current rotation, and speeding up or slowing down as necessary to keep that speed.

A Servo motor consists of the following components:

- A DC motor
- Gears with an output shaft
- Position sensing mechanism
- Control circuitry

The control circuitry indicates to the servo the position that the output shaft *should have*. The position-sensing mechanism tells the servo what position the shaft *currently has*. The control circuitry notes the difference between the desired position and the current position, and uses the motor to "make it so". If the difference in position is large, the motor moves rapidly to the correct position; if the difference is small, the adjustment is more subtle.

Fig 28

Fig 29

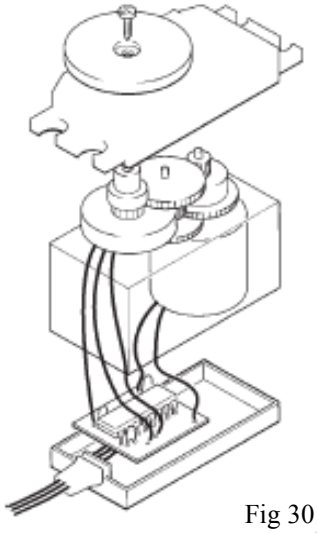


Fig 30

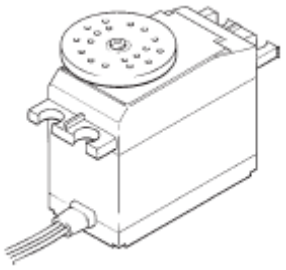


Fig 31

The figures the line diagram schematic and the exploded view of a servo motor. You can see the *control circuitry* and *position sensing mechanism* inside the servo motor. The position sensing is nothing but a potentiometer which gives the relative resistance to the origin at any point of time. The gear train inside the servo motor is also evident in the diagram. Finally we have three wires emanating out of the motor; VCC, Ground and Control. The motor shown can be driven with a maximum voltage of 5V DC.

Controlling Servo motors

Controlling the servo is quite easy. But you cannot drive it directly just by plugging it into 5V DC. You need a train of pulses along the control line. The width of the pulses that you provide determines the angle of rotation of the shaft of the servo motor. This method is called ***Pulse Width Modulation***.

To drive the servo, the pulses that you need to provide should have a minimum delay of 20ms. That is the off time between the pulses should have a minimum duration of 20ms. This off time may or may not be accurate. You cannot offset it by too much, but. You can exceed the 20ms mark somewhat but you cannot go below that. Now how do you think you drive the servo motor to various angles? Do you know that you cannot drive a stepper motor continuously in circles.* You can only make it rotate to several predefined angles like 0°, 45°, 90°, 135° and 180°. You would require to pass pulses of different widths, though! Below, we have listed the pulse width required to move the servo to several angles:

- 0° ———— 0.5 ms
- 45°———— 1.0 ms
- 90°———— 1.5 ms
- 135°———— 2.0 ms
- 180°———— 2.5 ms

Please refer overleaf for a graphical overview of the pulse width.

If you notice, in each case, the off time of the signals remain the same: 20ms. This 20ms off time ensures that the pulses are repeated 50 times in one second. After you ensure this, you may vary the on time of the pulses from 0.5ms to 2.5ms to vary the angle of rotation of the servo motor shaft. Beware that the on time of the pulses should be maintained accurately, otherwise the servo won't rotate at all. You have to keep on providing the same pulse width until you want a change in rotation. After that you have to provide the new pulse width to maintain the position of the servo.

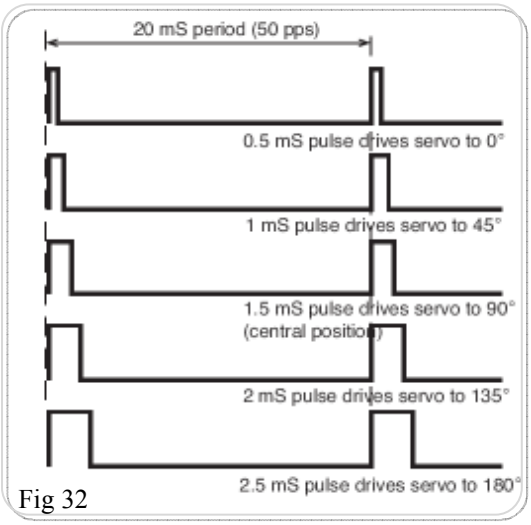


Fig 32

You have to understand that there is no easy way to make the servo rotate continuously. A servo is used only in those cases where you have fixed scope of rotations and you require accurate and exact positioning. Suppose that you require that your motor rotate only 45° and then stop over there. For this to happen, you have to provide a continuous pulse train with an *off time* of ***20ms*** and an *on time* of ***1ms***. This situation can also be attained with a stepper motor, but in a different way. But the difference will be evident when you actually turn off the system and subject the motor to nay free rotations. In a stepper motor you do not have any reference as to where your motor shaft is exactly positioned. But a servo motor always knows where its shaft is positioned. So if you again turn the system on and start giving the same pulse width of 1ms, the motor will get back to its original 45° position.

There are servo motors available which can actually rotate 360° continuously. But these motors are very costly and are practically not feasible for small time projects. There is one another way in which you can make a normal servo motor rotate continuously. If you open a servo motor, you will find a hardware lock which actually prevents the motor rotate more than 180°. You have to break that lock which will allow the motor to move to any angle you want. If you want the motor to come back to its reset position (0°) you have to resume pulses of 0.5ms width.

The advantage of this motor is that it can be operated even without a microcontroller. To achieve you can use two 555 timer IC to generate the desired pulses. Why two? This is because one 555 timer IC will generate a pulse with 20ms on time and the other one to invert it and also to introduce a specific on time in those pulses. The other method is to generate PWM using a microcontroller.

Interfacing Servo motors

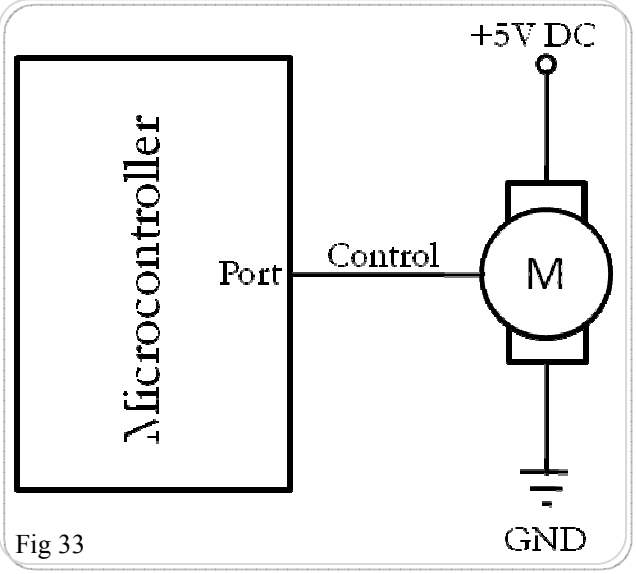


Fig 33

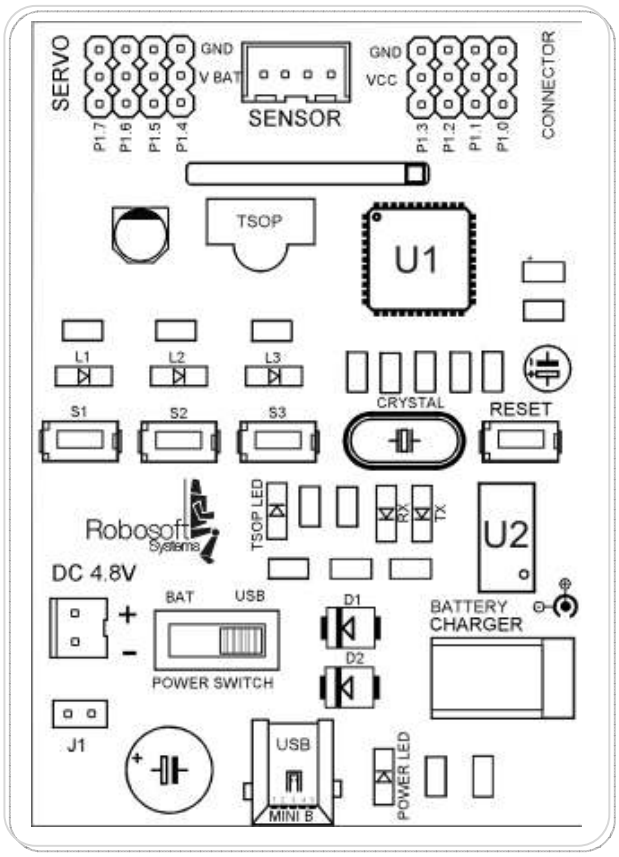
Beside we have shown the functional diagram of how to connect a servo motor to a microcontroller. A servo motor has three leads, VCC, GND and Control. We have connected VCC to +5V DC, GND to Ground and Control to any port pin of the microcontroller which may be free. The servo motor draws the required electricity from an external source other than the microcontroller. So we do not require to connect a motor driver circuit or IC to the motor. In the next section we will look at the way about how to generate PWM through the microcontroller.

Generating PWM

A microcontroller has four I/O ports. Each I/O port has eight pins. That means there are thirty two pins available for I/O. These I/O ports can be independently set to either input or output. They are called bit-addressable. This means that if one pin of a specific port has been marked for output then the other pins of the same port can be used for input. Now to connect the servo motor to a pin of the microcontroller we need to configure that pin in output mode. We have an inbuilt timer circuit inside the microcontroller. A timer is a circuit which can be used to count up or down or to generate time delay. So we use an inbuilt timer to generate a signal which has an on time of 20ms. We keep the output port pin low during that time. Then we reset that timer and again use it to generate another signal which has an on time of 0.5ms. We keep the output port pin high during that time. Thus if we analyze the output port pin, we will have a continuous signal which has an on time of 0.5ms and an off time of 20ms. When this signal reaches the servo motor on the control wire, it makes the servo to remain steady at 0° (reset) position. Even if you try to move the shaft of the motor with your hand during this time, the motor will oppose the force that you apply. Now if you want to move the shaft of the motor to the 90° position, you have to make the timer generate the delays so that the output port pin stays low for 20ms and high for 1.5ms. The use of timers and start and stop of them are further explained in detail in the disc that accompanies this product. Please refer to them.

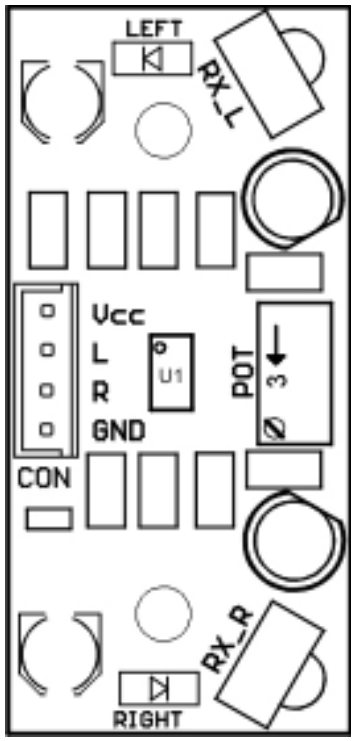
Know your board

The microcontroller board



Beside, we have shown a schematic of the actual electronic board that has been provided with the HexaBOT. You can say that this is the brain of the robot. It contains all the controller IC's and other circuitry so that you can write your own programs and make the HexaBOT walk according your wish. You have to mount the board on the HexaBOT using a double-sided tape, so that the board does not fall of from the robot when it walks. Below we have listed some of the important components that are worth listing.

Symbol	Devices	Symbol	Devices
SENSOR	Obstacle Avoidance	TSOP LED	IR data reception indicator
SERVO	Servo Motors Connector	U2	FT232
CONNECTOR	Other Sensors	RX ,TX	UART Trans-receive indicator
U1	P89V51RD2	USB	USB Cable Connector
L1,L2,L3	Output LED	J1	Jumper for Battery
S1,S2,S3	Input Keys	POWER LED	Power Indictor LED
TSOP	IR Remote Sensor	POWER SWITCH	Supply Selection Switch
RESET	Controller RESET	BATTERY CHARGER	Charger Connector



Dual TSOP is a handy sensor module used for obstacle detection. This module works on the principle of IR transmitter using IR LEDs and IR receiver using TSOP sensors. The distance for detection can be adjusted by tuning the pot. 555 IC is used to generate a frequency of 38khz which will be transmitted using the IR LEDs.

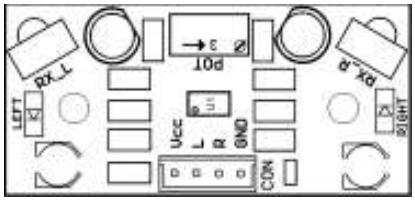
SERVO MOTOR	PORT PINS	IDs
SERVO A	P1.4	1
SERVO B	P1.5	2
SERVO C	P1.6	3
SERVO D	P1.7	4

SYMBOL	DEVICES
IR LED	LEFT and RIGHT IR transmitters
RX_L, RX_R	TSOP receivers
POT	SENSITIVITY
LEFT and RIGHT	OBSTACLE INDICATOR
U1	555 IC

SERVO A

1

RIGHT MOTOR



SERVO

PORT PINS — YELLOW

GND — BLACK

Vcc — RED

4 3 2 1

SERVO B

2

CENTRE MOTOR

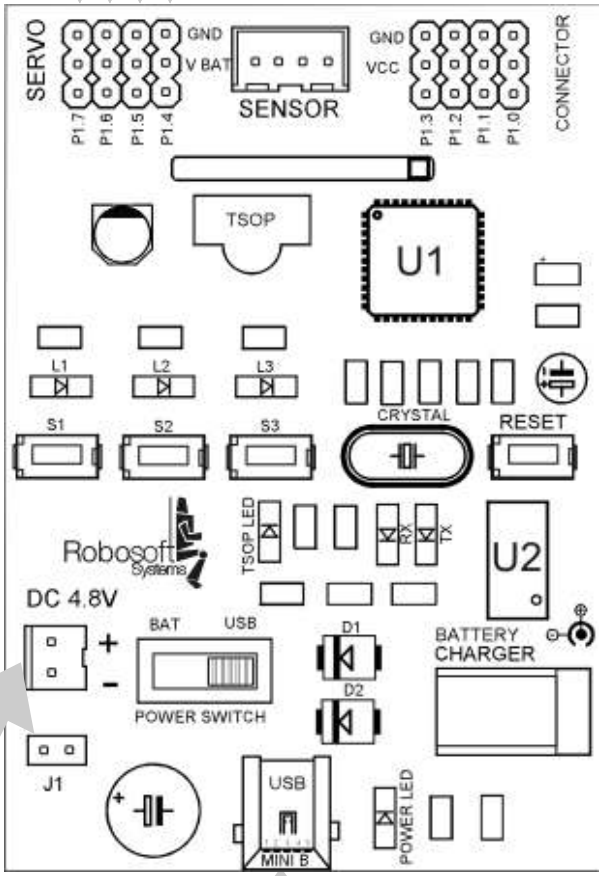
SERVO C

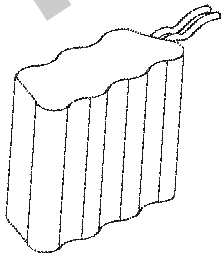
3

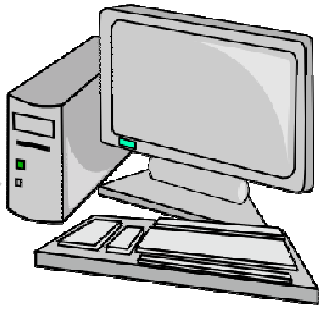
LEFT MOTOR

SERVO D

4







How to Do Connections

Step 1 : Connecting Servo Motors

Servo Motor has 3 pin connector. Out of which Black wire is Ground, Red wire is VCC and Yellow is signal cable. so when your connecting servo motor make sure it proper color wire going into proper slot. If you connect servo in reverse order your servo may get damaged. We have written all sample code with connection shown as on previous page. So at start up we recommend that do connection as shown if fig.

Step 2 : Obstacle Detecting Module

Obstacle sensor is connected with main board using relimate connector. We have provided 4 wire connector, which is having 4 pin female relimate connector at both the sides. The connector can be fixed in only one way because of slot on it. So you don't have to worry about polarity.

Step 3 : Connecting Battery

Before connecting battery make sure whichever sensor module and Servo Motors you have connected is in proper way. While connecting battery first put switch Sw1(Power Switch) in USB state. And jumper J1 removed. Your board can be powered using either USB or Battery.USB provides supply to controller board and that power is not sufficient to drive servo motors. That is reason why we are connecting battery on board.

Before switching to battery mode first time make sure you have uploaded Servo Rest Code in your HexaBOT and all black screws and plastic disc of servo shaft has removed. Before connecting jumper make sure all theses points.

1. **Servo Motor connected in proper polarity.**
2. **Code for all rest downloaded on board**
3. **Plastic ring on servo not mounted .**

Polarity of battery is important. Black wire of battery is Ground and other wire that is either Red or Brown will be the Vcc. If you want to switch for other battery then make sure the battery should provide maximum current output of 1300mA.And It should be 4 battery pack(1.2V X 4). Connecting high voltage battery to servo can damage your servo motors.

If all above condition are true then only connect jumper as soon as you connect battery servo is drive to 90 degree. Now fix all legs such that side legs remain perpendicular to robot and middle legs both sides touching ground. Jumper can be used as a switch between battery and your board. So when jumper there battery is connected if jumper removed battery disconnects.

How to Charge Battery

Step 1 : Connect Battery

If you are charging battery make sure it is properly connected on board. It should be connected in correct polarity and firmly fixed. If there is any loose connection then your battery wont charge. So make sure it is fixed properly.

Step 2 : Remove Jumper

When your charging your battery your jumper J1 must be removed. If it is connected then it will keep on supplying current to servo motors and board. So your battery will charge and discharge at same time. So it is important that you should remove jumper.

Whenever the HexaBOT is not in use, it is a good practice to remove jumper. It will extend runtime of your battery.

Step 3 : Switch Position

While charging battery your Power Switch (Power selector switch) should be at USB mode.

Step 3 : Connect Adopter

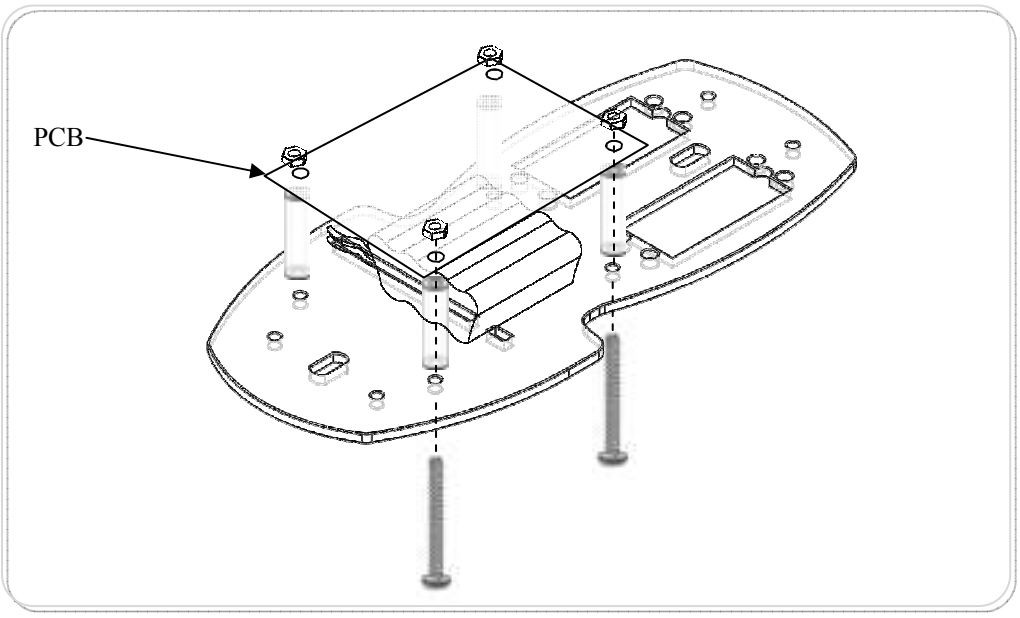
Once all of the above condition are true then you can connect adopter provided by Robosoft systems for charging. To charge fully discharge battery over night charging is necessary. The charging process is slow. Slow charging will extend your battery Life Time and Run time also!!

How to fix batteries

Once you finished with all the mechanical assembly now next task is to fix battery and PCB on HexaBOT.

There are 4 holes on main acrylic body of HexaBOT. You have to mount board such a way that battery will be sandwiched between board and base. After Placing all the component use longest screws to match the holes of PCB and acrylic Board. Screws should go through spacers placed in between board and acrylic body.

In fig. Only 2 screw are shown. You have to fix all 4.



While fixing 4 screws you may have to open middle servo motor because screw position is just on top of servo motor.

How to write program for your HexaBOT

First and important thing is you have to include library function to your project which contains all predefined function routine for your Servo Motor control and IR remote reception. So if you want to control Servo Motor or IR Remote or both then you have to include lib file "robosoft.lib" to your project and add include header file "robosoft.h" in your main c file.

Lets start with Introduction to robosft.lib file and its uses.

robosoft.lib file defines 4 major predefined functions

1. **void Servo_Init (void)**
2. **void Servo_Control (unsigned char Servo_No, unsigned char Angle)**
3. **void Remote_Init (void)**
4. **unsigned char IR_REMOTE (void)**

1. void Servo_Init(void)

This function has to be called at beginning of your main function to initialise servo motors.You don't have to pass any parameter to this function and it also don't return any parameter.

E.g Servo_Init( ); //Initialise Servo Motors//all servo at 90 degree

2. void Servo_Control (unsigned char Servo_No, unsigned char Angle)

This function is used to control servo motors. You have to pass 2 parameters to control servo motor. You have 4 servo motor connection available on board out of which you have to use any 3 to control your hexaBOT. Each servo is defined with its id as 1,2,3 or 4. For detail check know your board section. As you know servo motor can move from 0 degree to 180 degree. So while controlling servo motor you have to pass two parameters. That is servo number and Angle for that particular servo motor. After initiation all the servos will be set to 90 degree angle.

Eg. Servo_Control(1 , 95); //Move servo 1 to position 95 degree

3. void Remote_Init(void)

This function has to be called at beginning of your main function to initialise IR remote. You don't have to pass any parameter to this function and it also don't return any parameter.

E.g Remote_Init( ); //Initialise IR remote

2. unsigned char IR_REMOTE (void)

Whenever you call this function IR remote will go for scanning for around 100 msec. If any key pressed then its equivalent value will be return by function. If no key is pressed then function will return zero. The values for each buttons are defined in the header file of "robosoft.lib" .

#define NUM_1	1	#define NUM_0	10
#define NUM_2	2	#define NUM_10	11
#define NUM_3	3	#define FWD	12
#define NUM_4	4	#define BWD	13
#define NUM_5	5	#define RIGHT	14
#define NUM_6	6	#define LEFT	15
#define NUM_7	7	#define STOP	16
#define NUM_8	8	#define PW	17
#define NUM_9	9	#define MD	18

So whenever you press any key its respective value will be return. Lets assume if you press MODE key then IR_REMOTE function will return value 18.

Important NOTE :

Please note that this library is written for P89V51RD2 and it will not run for any other controller. P89V51RD2 has 3 timers and 1 PCA timers. Since we are using all PCA timer pins for controlling servo motors make sure you are not using or affecting its values in your code. Similarly Timer is also consumed by library file so you are not allowed to access the register for that timer. f you are trying to modify the values for PCA timer or Timer 2 your HexaBOT will fail to perform.

How to Add Library :

Step1:

First thing is you have to copy both **"robosoft.lib"** and **"robosoft.h"** file to current project folder.

And when you are adding file to source group add your **" *.c "** file containing main function and **"robosoft.lib"** file also.

Step2:

You have to include library in main c file also just below your reg51f file declaration.

i.e. ***#include <reg51f.h>***

#include "robosoft.h"

Step3:

Before using any function related to Servo and IR remote you have to initialise servo motors and ir remote after declaration of main function

i.e. ***void main (void)***

{

Servo_Init( );

Remote_Init( );

//…….. And so on

//… after this you can use servo control and ir remote function any time...

Lets try sample program :

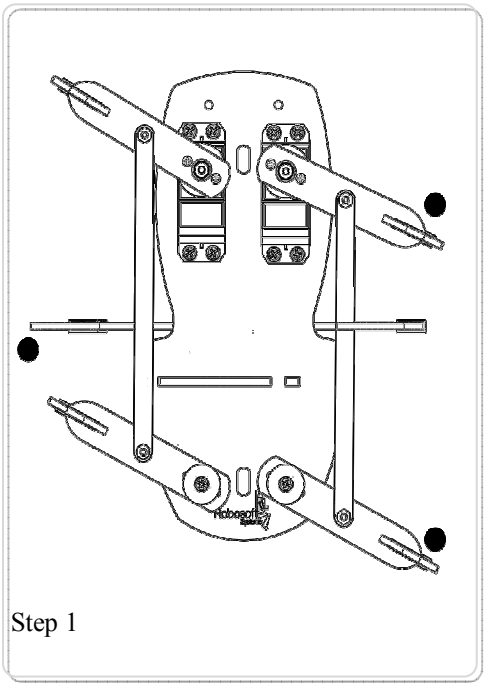
Problem Statement if control servo 1 with IR remote if "UP" key is pressed angle should increase If "DOWN" key is pressed angle should decrease.

```
#include  <reg51f.h>          // header file for P89V51RD2
#include "robosoft.h"         //include servo and IR remote function

void main ( void )            //main function starts here
{
unsigned char  key = 0;       //variable assigned to store IR remote key
Unsigned char angle = 90;     //variable assigned to store servo angle
Servo_Init( );                //Initialise servo motors
Remote_Init( );               //Initialise IR remote control
Servo_Control( 1 , angle);    //Move servo 1 to angle position i.e. 90 here
while ( 1 )                   //repeat code continuously
  {
        key =  IR_REMOTE( ) ;     //Scan IR remote once

        if( key == UP )           //if UP key is pressed
        {
        If( angle < 180)          //if angle is less then 180
        angle++;                  //then increment angle
        Servo_Control( 1 , angle);//and move servo to that angle
        }

        If(key == DOWN)           //if DOWN key is pressed
        {
        If( angle > 0)            //if angle is greater then 0
        Angle- -;                 //then decrement angle
        Servo_Control( 1 , angle);//and move servo to that angle
        }
  }
}
```

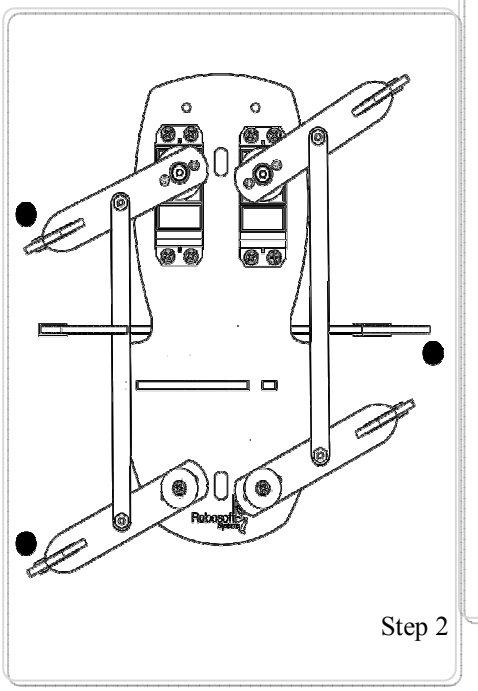
Programming the motions of HexaBOT

The flowchart beside shows the steps that has to be followed to make the HexaBOT go forward. The robot has three motors: right, left and center.

The flowchart is self-explanatory. But still we will explain the forward motion. For the other movement, you are required to figure out the details for yourself by looking at the flowchart. First of all the right motor is moved backward, then the left motor is moved forward. Lastly the center motor makes the right leg move up. After this the controller calls a delay, which is necessary. Without delay the servos will not respond. In the next step the right motor is moved forward, then the left motor is moved backward. Lastly the center motor makes the left leg go up. The controller calls a delay again and the entire process is terminated and the control returns to the calling program. If the calling program decides to call the forward procedure once again, then all the steps that has just been explained are repeated once more before the control returns to the calling program once again.



Step 1



Step 2

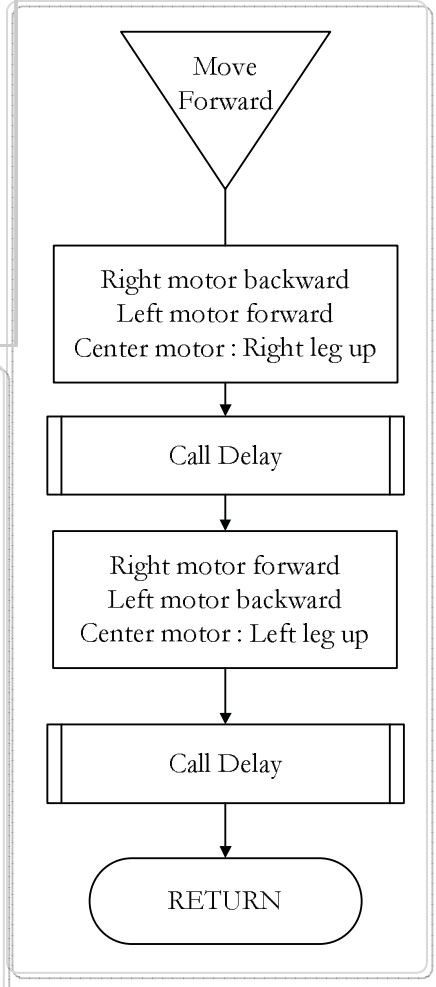


Fig 9

Moving backward

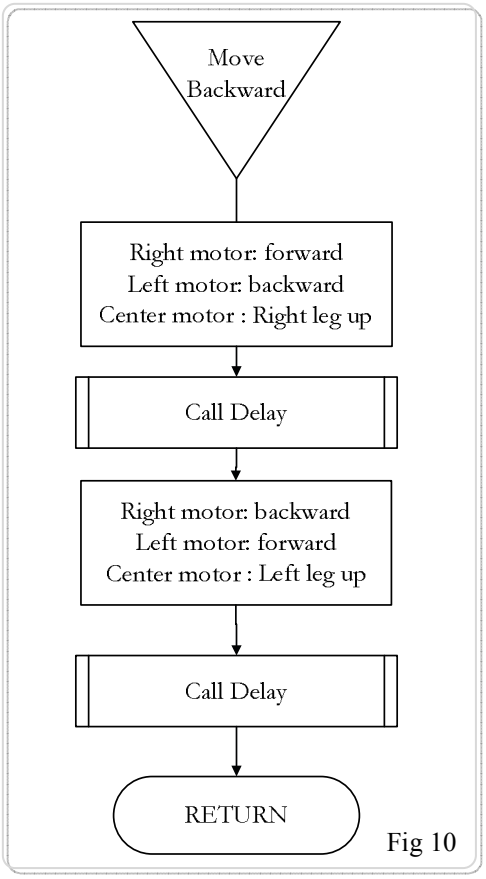
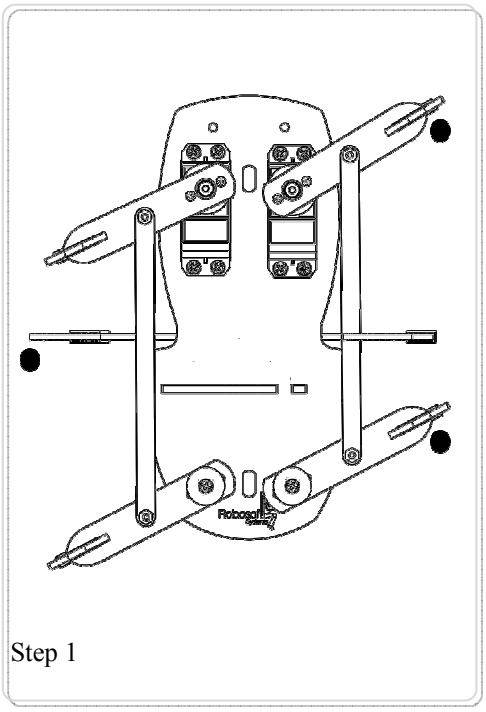
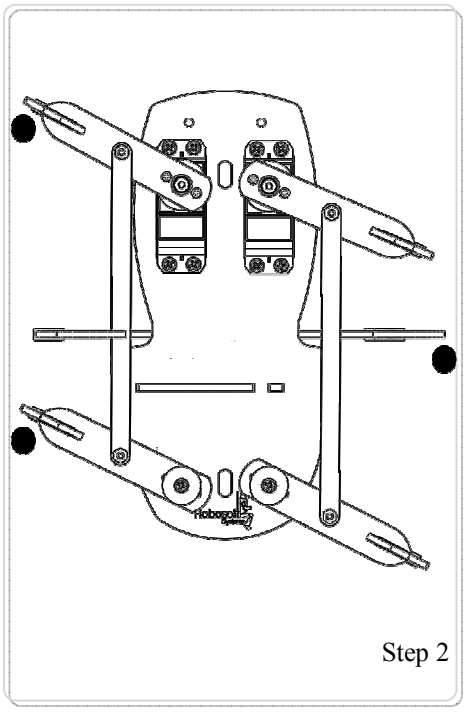


Fig 10


Step 1


Step 2

The flowchart shows how you can call the Move Forward procedure from the main program.

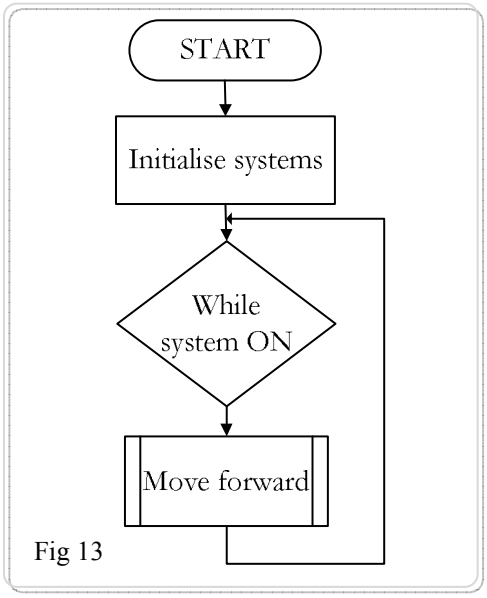
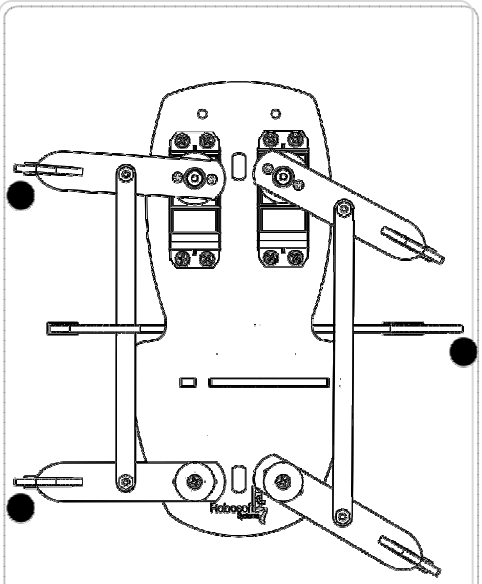
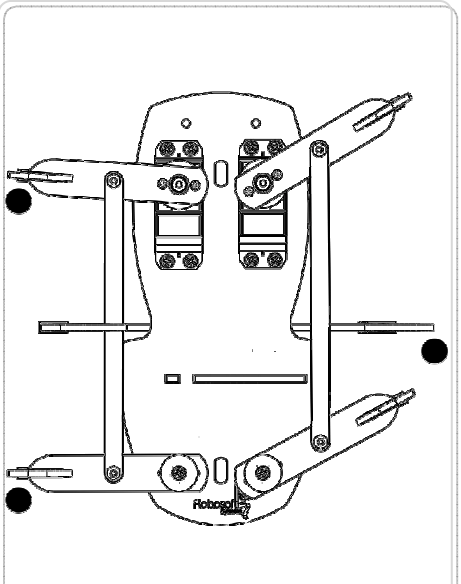


Fig 13

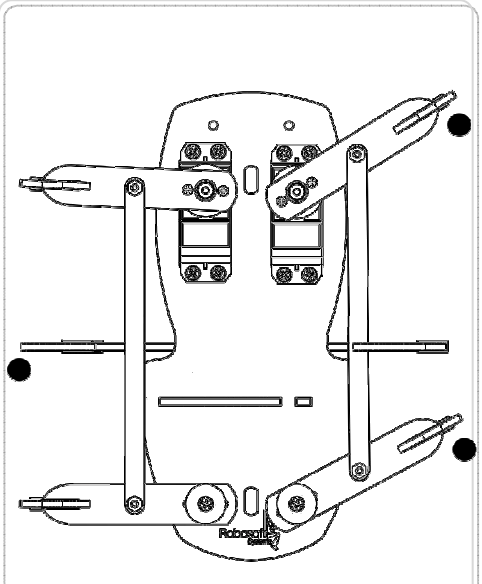
Turning left



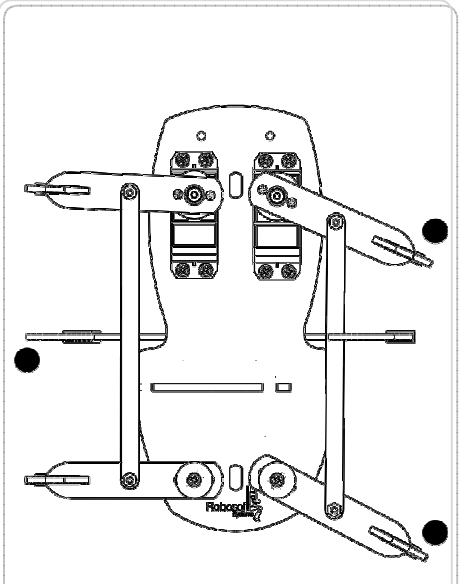
Step 1



Step 2



Step 3



Step 4

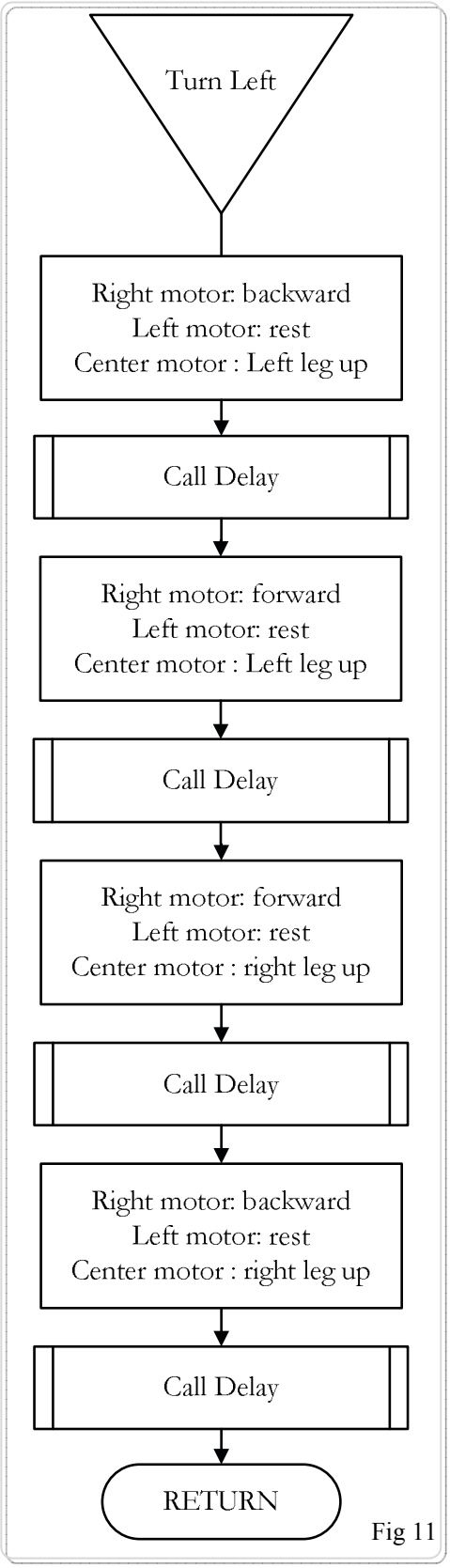


Fig 11

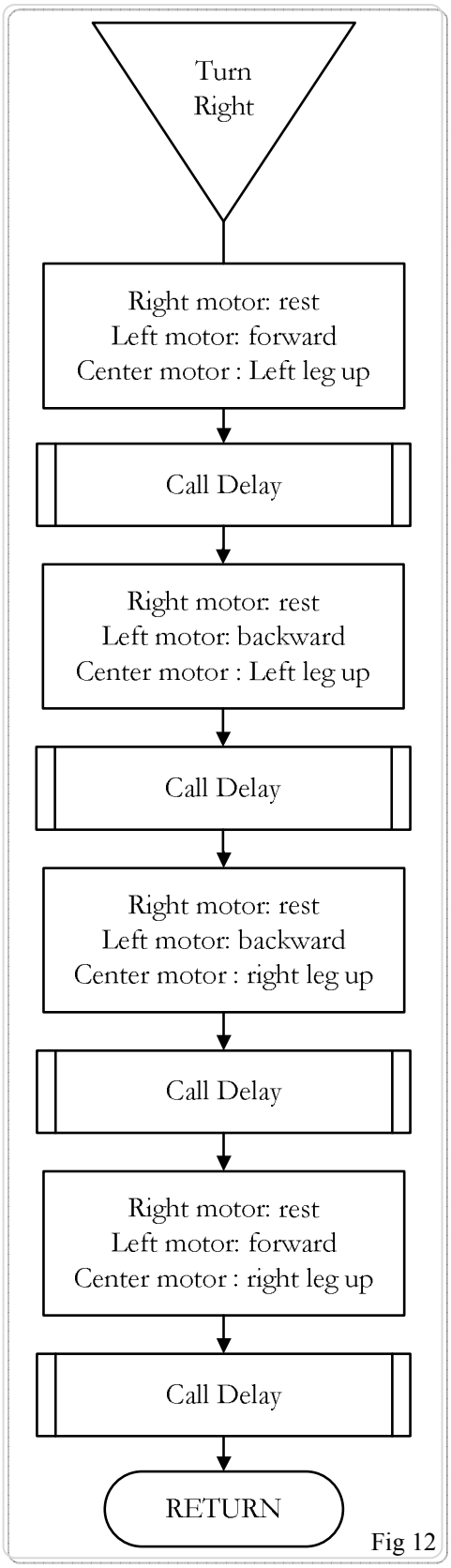
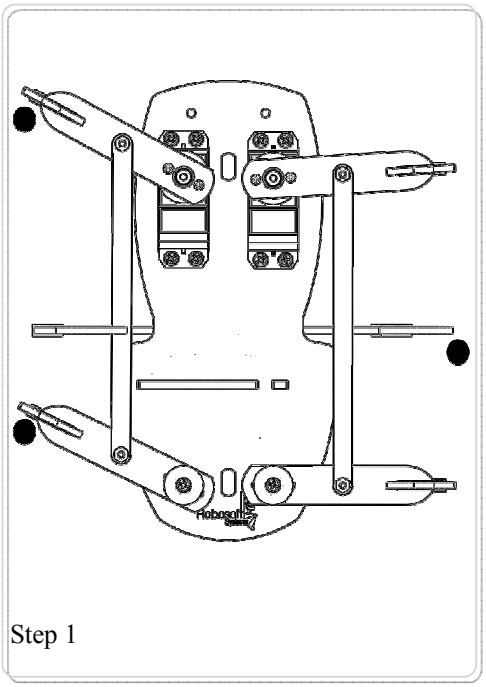
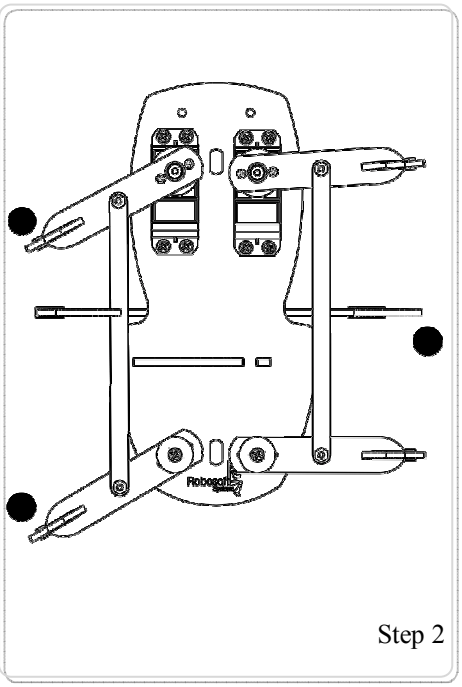


Fig 12

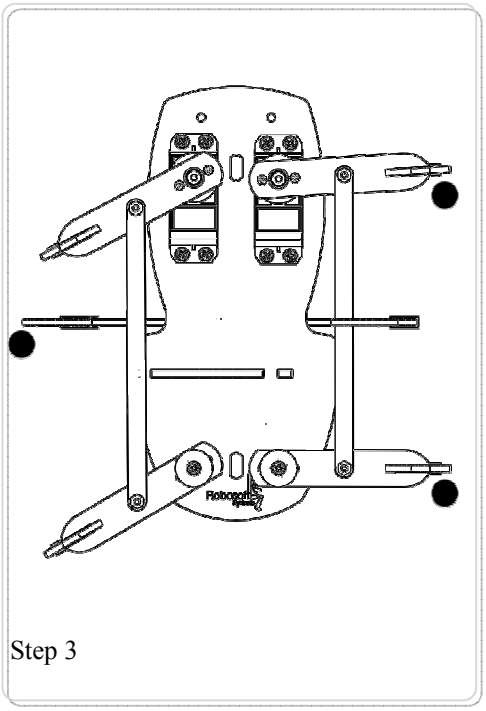
Turning right



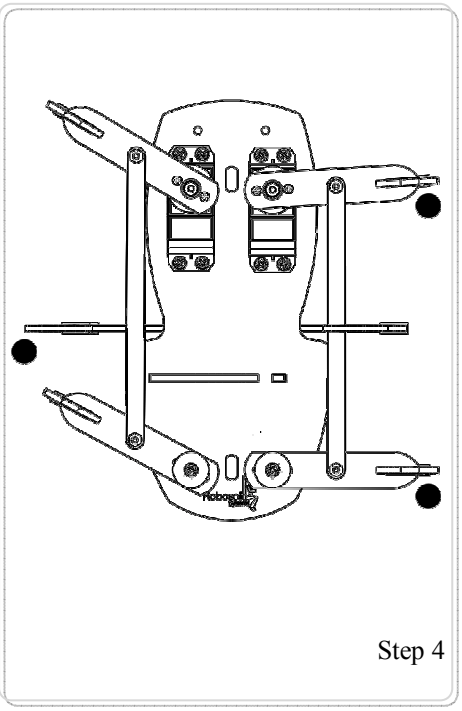
Step 1



Step 2



Step 3



Step 4

Navigation with IR remote

Fig 16

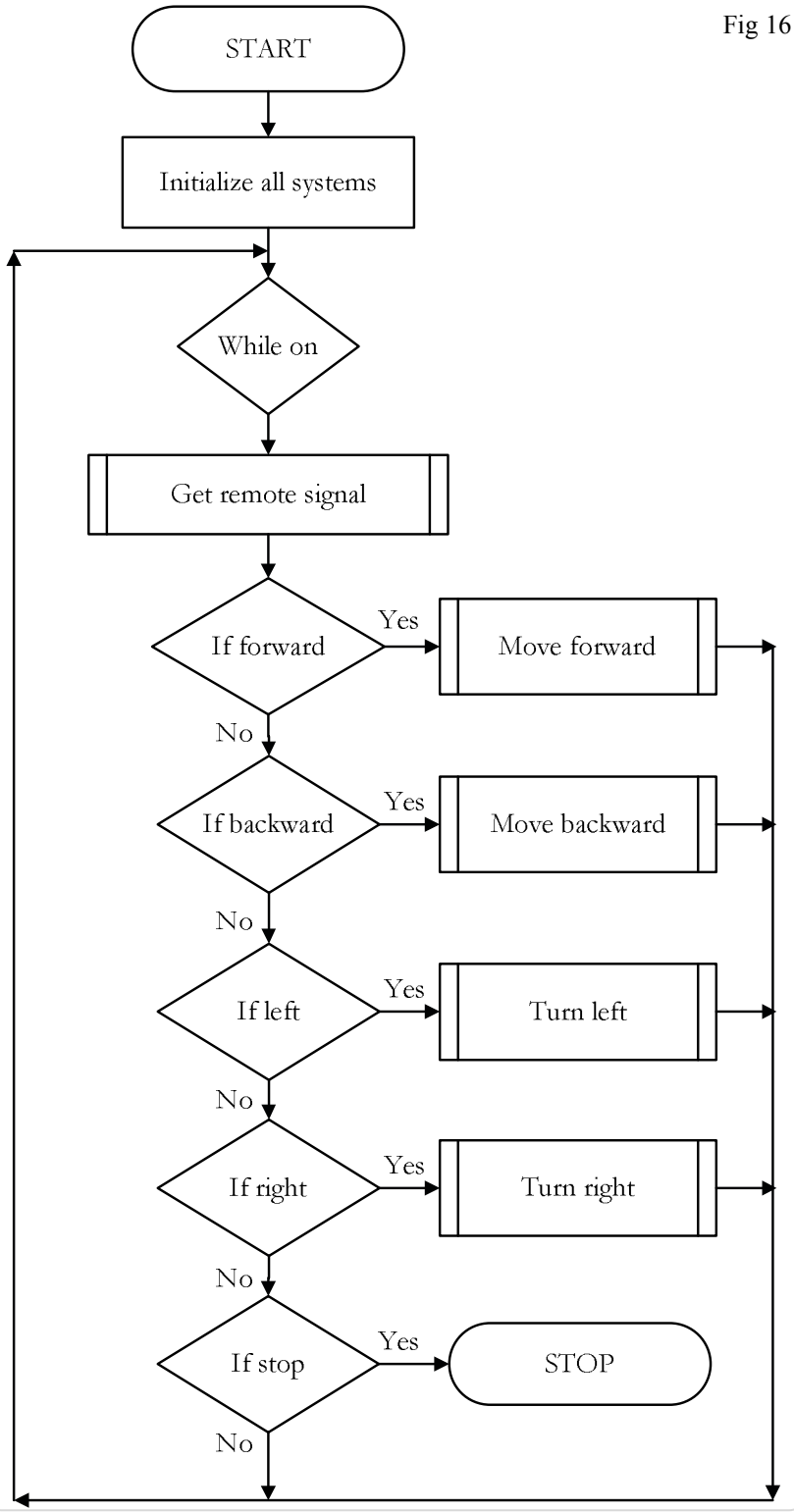
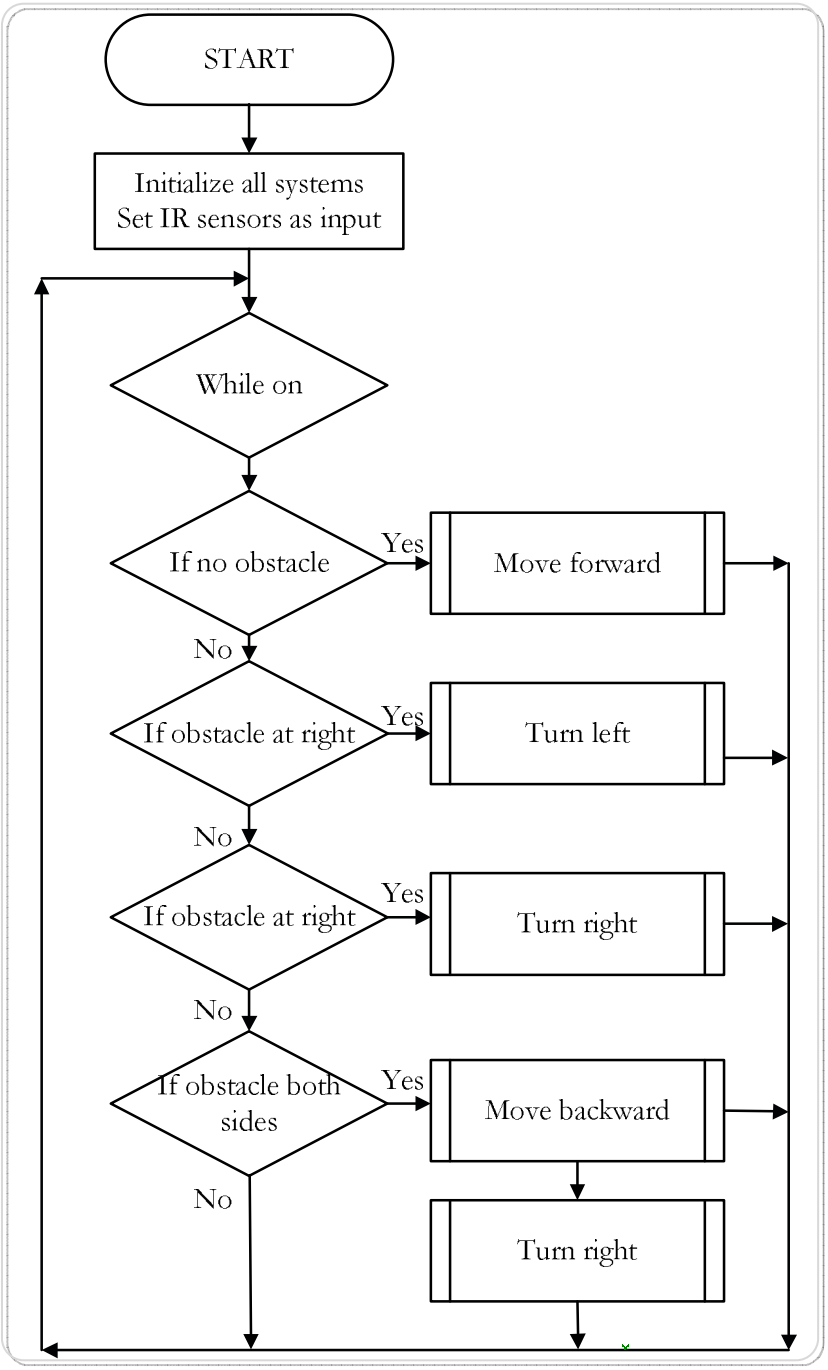
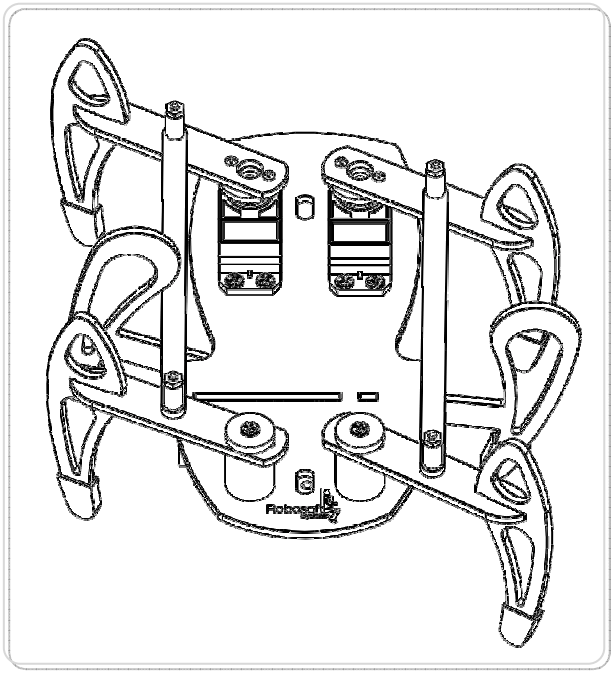


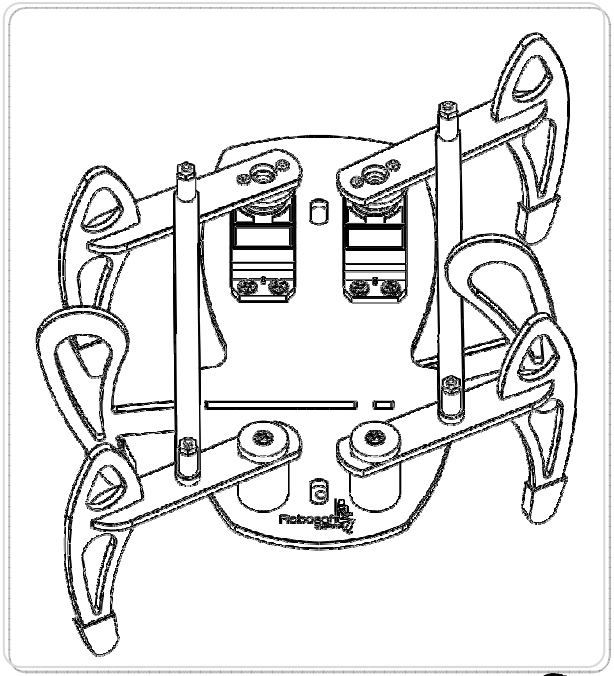
Fig 17

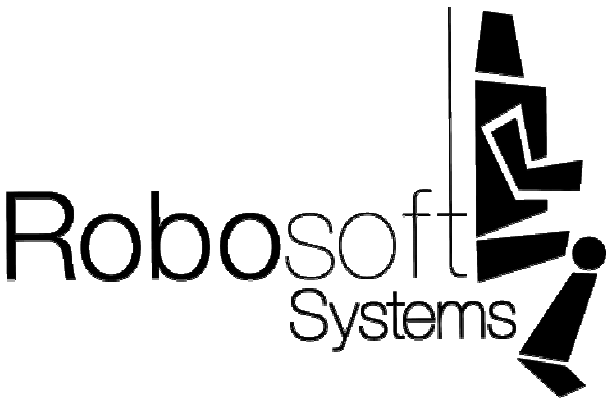
HexaBOT PRO

Want to go pro now? Let's try to make our HexaBOT work autonomously, shall we? That means we will make use of the obstacle detector module and make the HexaBOT work by itself and avoid obstacles by itself. For detailed program, please









A-61, 1 st Floor, Raj Industrial Complex,
Military Road, Marol,
Andheri (East),
Mumbai – 400059
India

Contact Information
Office: (91-22) 66751507
Office: (91-22) 29207086
Mobile: (91) 9930776661

Email: info@robosoftsystems.co.in

Visit us at
www.robosoftsystems.co.in