Introduction

The human brain is the central [organ](https://en.wikipedia.org/wiki/Organ_(anatomy)) of the human system .The brain sense robot car is interesting to all type of people, to measure their concentration and piece level of mind with our brain sense head band. Our bio-sensor band digitize brain wave signals to power the user- interface of computers, game and health application. The brain sense measure brainwaves from the forehead.

Building the Chassis

Before programming anything, we’ll build the chassis for the robot. Basically it’s a traditional rover robot structure with two Dc motors in the back and one caster in the front. To make it suitable for mind-controlling needs, we’ll add a [Bluetooth](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) for wireless communication and LED indications on the top. We use a Screw Shield for the Arduino, to make adding components and wires easy. Fig 1-1 shows the design of the chassis. Here’s how all the major components will work together to create a working robot:

Arduino

This is the brains of the project. It is essentially a small embedded computer with a brain (a microcontroller), as well as header pins that can connect to inputs (sensors) and outputs (actuators).

Chassis

This holds everything together. It’s essentially the platform for the robot.

DC Motors

These are motors that can be connected directly to the Arduino without the need for any additional hardware (such as a motor shield). The Arduino communicates with them by sending pulses to control speed and direction.

Caster wheel

Because we’ll be turning the robot by varying the speed and direction of the servos, which are fixed in place, we need one wheel that pivots nicely. A furniture caster is perfect for this, and the robot ends up being able to rotate in place.

Screw Shield

Screw Shield adds “wings” with terminal blocks to both sides of Arduino. Terminal blocks have screws, so you can attach one or more wires firmly

to any pin. This prevents wires from popping out, which makes building

and coding the robot much easier.

[Brain Sense](https://www.pantechsolutions.net/sensors/brainsense)

[Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) measures your brain Waves from EEG signal and transmits the results of attention and meditation to Arduino.

Tools and Parts

Here we list the parts and tools needed to make the robot

PARTS

☞[Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) sensor  or Neurosky mindwave mobileas a mind reading device

☞Arduino Uno

☞([Bluetooth HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module))

☞DC motor

☞AA Battery(6v-4nos)

☞Battery Holder

☞Wheels

☞USB connector(B- type)

☞Male and Female Headers

☞DC motor Driver(L293D)

☞L-brackets with 3 mm holes

☞Free Wheel

☞Connecting wires and jumpers

TOOLS

☞Hot glue gun

☞Soldering iron and solder

☞Pliers

☞Screwdriver

☞Driller

☞Screws

☞Multimeter.

DC Motors

DC motors will be moving the wheels of our robot. The most usual type of DC have limited rotation. They are used when you need to turn the motor to a specific angle. In our robot, we only need to control speed and direction. And, of course, the motor needs to be able to turn freely. Continuous rotations DC are made for this. Almost any servo can be modified to continuous rotation, but it’s easier to buy a ready-made version.

The Parallax (Futaba) continuous rotation DC is perfect for our needs. This allows identical centring of two DC effortlessly. You’ll notice how handy this is later, when we program the movements for the robot.

Attaching DC Motors

We’re going to use regular L-brackets to attach the DC motors. Attach two brackets to each servo with screws if you can’t find suitable L-brackets, you can make them from metal strips. For example, you could salvage some strips from an old typewriter, drill holes that match your servo, and bend them to a 90° angle in the middle.

Chassis

For the chassis you’ll need something that is robust enough to hold the robot together and can be shaped easily. Plywood, acrylic, or metal plate (Figure 1-3) works well.

Even if you don’t have a sign-making store as your neighbour, you could find other useful material thrown away. For example, you shouldn’t have any trouble finding a metal plate from computer parts or plywood from furniture. Choose a material that is easily available for you and that is comfortable to process with your tools and skills.

Making Holes

The easiest way to get the holes where you want them is to hold the target object as a stencil and mark spots with a pen. (Figure 1-9).

First, mark the DC motor   bracket places on both sides of the robot. The back wheel will be mounted centred in the back of the chassis. Place the Arduino on the top of the Robot so that the screw holes don’t overlap the back wheels’ holes. One more holes needed for battery holders placements.

Next we’ll drill the holes, but first, its hammer time. Always use a nail punch and a hammer to make a small starting hole before drilling metal.

Wheels

There’s plenty of choice for the wheels (Figure 1-5). Just don’t pick any that are too heavy, and make sure that you’ll be able to make the holes for the DC motor horn attachment. For our robot, we salvaged tires from a remote controlled car.

Drill two 3 mm holes through both DC motor horns and the wheel rim. Secure the DC motor horns to the wheel rims with 3x10 mm screws. Push the DC motor horns on the DC motor shaft and tighten them in place with 3x18 mm woodscrews.

Battery Holder

A simple 4×A battery holder placed on the back of the chassis. Drill the hole in the middle Put a 3x10mm screw through the hole and mounts the battery holder to the chassis.

[Bluetooth](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module)

[Bluetooth module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) is used to communicate with brain sense and Arduino via wireless manner. One [Bluetooth module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) is connected with top of our chassis and another one is connected with brain sense band

[Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) Band

Our [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) band contains EEG module (TGAM1-R2.8) and [Bluetooth module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) (hc-05) with a electrode. It is placed on forehead of the user to measure their brainwaves and to power the user interface of robot.

Screw Shield Holds Wires in Place

Loose wires are probably the most annoying thing while building a prototype. Arduino’s header pins can’t hold the jumper wires securely. Especially when your robot is moving, wires pop out constantly. Luckily the Screw Shield pretty much solves this problem.

Screw Shield adds “wings” with terminal blocks to both sides of Arduino. Terminal blocks have screws so you can attach one or more wires firmly to any pin.

You have to do some soldering when building the Screw Shield, but it will pay off later on. As an added bonus, you’ll get nice and relatively easy soldering practice.

This completes your robot chassis! You now have a moving robot platform. In the next chapter, you’ll be coding the mind-controlled robot part by part. What other projects could you do with your new robot chassis

The code that controls your robot is simple. Well, it’s simpler than one would expect for a sci-fi trick like mind control.

A simplified, pseudo code version of your robot would be as follows:

void loop ()

{

while (line Detected ()) turn (); //

forward (get Attention ()); // }

Then the robot goes forward. The speed is read from the EEG headband. Rinse and repeat. In this coding part of the book, you’ll write each part of the code that makes your robot tick.

Moving

Continuous rotation DC motors are motors you can easily control. With continuous rotation DC motor, you can only control the speed (forward, stop, back).

There are three wires going to a DC motor: red (positive), black (ground), to control a DC motor.

Connect DC motor

Let’s start by connecting DC motor to Arduino. To keep the solders in your DC motor intact,

Now connect the wires to Arduino. You can connect both the left and right DC motor away.

Then Arduino data line (GPIO) is given to motor driver (L2930) and then it is given to pins of motor driver. After that output of driver is given to input of the motor

Note: Every GND symbol means a connection to Arduino GND pin, which is also connected to the black 0V GND wire of the battery. This saves the trouble of having a circuit diagram full of wire going to ground.

  -------------------------------------------------------------------------------------------------------------

Configuration settings

Steps to interface [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) with Arduino

Find the Serial number of your [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) Headband

1.Select your [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) Headband

Right click →     Properties →         Bluetooth →

Then know your Unique Identifier Number (Note it)

Example: This band Unique no =98d3,31,302424

Diagram for HC-05 [Bluetooth module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) with Arduino

Method 1 for AT Commands

                                    Fig2.3:Connection of Arduino with [Bluetooth](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module)for pairing Brainsense or mindwave mobile with [Bluetooth](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module)

------------------------------------------------------------------------------------

Note: Here the Arduino Board used only for USB to UART converter.

So don’t want to download any source code (Sketch) to board.

Modes of HC-05 [Bluetooth module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module):

How to [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) Enter into Command mode:

Press and Hold the Push switch in [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) at few secs, when its power ON (In between OFF state to ON state).

If [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) in Command mode means, the LED blink at low toggle (blinking slowly).

Baud rate for Command Mode

    In command mode the [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) works in 38400 baud rate (fixed Baud rate).

How to [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) Enter into Data mode:

   In default state, the [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) works as a Data mode operation.

   If [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) in Data mode means, the LED blink at High toggle.

Baud rate for Data Mode

    In Data mode the [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) works in variable Baud rate.

AT Commands for [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) Initialize for [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense)

          First Enter into command mode, after then enter the following AT commands for [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module)to communicate our [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) Headband.

☞.AT+NAME=”PANTECH”

☞.AT+UART=”57600,0,0”

☞.AT+ROLE=”1”

☞.AT+PSWD=”1234”

☞.AT+CMODE=”0”

☞.AT+BIND=”XXXX,YY,ZZZZZZ”(MindwaveUniqueNumber)

☞.AT+IAC=”9E8B33”

☞.AT+CLASS=”0”

☞.AT+INQM=”1,9,48”

Verification commands for [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module)

1.AT

          OK

2.AT+UART?

                                      +UART:57600,0,0

                                        OK

2.AT+ROLE?

                                      +ROLE:1

                                      OK

3.AT+PSWD?

                                      +PSWD:0000

                                      OK

4.AT+CMODE?

                                      +CMODE:0

                                        OK

5.AT+BIND?

                                      +BIND:9cb7:0d:89e51c

                                      OK

6.AT+IAC?

                                      +IAC:9E8B33

                                       OK

7.AT+INQM?

                                      +INQM:1,9,48

                                       OK

----------------------------------------------------------------------------------------------------------------

Note: For using above AT commands don’t use “(Double quotes) only enter In-between commands from double quotes.

After the end of every command press “Enter Key”

----------------------------------------------------------------------------------------------------------------

Open serial communication software

(Here we have to use Flash magic software)

1.Select your Port and set Baud rate is 38400

Enter the AT com

2.Enter the AT commands (Verification)

Some Tips about [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) Configuration

1. [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) configure as Mater device and [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) Headband act as Slave device

☞And the [HC-05 Module](https://www.pantechsolutions.net/module-ic-s/hc-05-bluetooth-module) was configuring as Specific address mode. Therefore it’s Pair with only the specific device.

☞Here the specific device is our [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) Headband (Using BIND AT command, we configured the [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) unique number to HC-05).

☞So the [Brain Sense](https://www.pantechsolutions.net/sensors/brainsense) and HC-05 module automatically paired, when we switch ON the both device.

NeuroSky Protocol

How do you begin when faced with an unknown protocol? We did some Googling, and found that the NeuroSky reference documentation and more was available on the NeuroSky website. For example, see “Mind Wave and Arduino” (NeuroSky 2011), “Mindset Communications Protocol” (NeuroSky 2010), and “arduino\_tutorial” in the NeuroSky wiki ([*http://developer.neuro*](http://developer.neuro/) *sky.com/docs/doku.php?id=start*).

The NeuroSky dongle communicates over the serial port. We have wires going to Arduino RX and TX.

Every packet is of this format: sync (170 170), payload length, payload, and a checksum (see Table2.1)

Table 2-1. NeuroSky packet

|  |  |  |
| --- | --- | --- |
| Meaning | Example | Comment |
| Sync | 170 170 | Always the same |
| Payload length | 4 | Most common length in  this project |
| Payload | 486 | 4 attention, value  86/100 = 86% |
| Payload cont. | 20 | 2 poor signal, value 0  signal OK |
| Checksum | 163 | 255–(sum of each payload  byte) |

Even though the reference documentation saved us the tedious work of reverse engineering the protocol, there were surprises around the corner. Coping with serial buffer overflow, long delays caused by moving parts, and doing voltage level changes without additional circuits were problems to solve. See “Mindset Communications Protocol” and “Mind Wave Dongle Communications Protocol.” Some code samples on the Web use everyday decimal format (170), but some use hexadecimal (0xaa or just AA). We’ve compiled a list of the most common code numbers You can also convert numbers with Python. Just start Python at the command line and you’ll get an interactive console where you can type things like *hex(170)*or *0xaa*. Or you can use *http://www.wolframalpha.com*or Google. Most of the NeuroSky protocol control characters we use are outside ASCII. The last ASCII printable character is 126 tilde *~*; the last ASCII character is 127 DEL.

// //

// Program : Mindwave with Arduino\_Robo Car //

// Interfacing : HC-05 Bluetooth Module //

// Output : Robo Control using Attention //

#define BAUDRATE 57600

#define LED 13

#define ATT\_TH 40

int Mot\_L\_Plus = 3;

int Mot\_L\_Minus = 5;

int Mot\_R\_Plus = 6;

int Mot\_R\_Minus = 9;

byte payloadData[32] = {0};

byte checksum=0,generatedchecksum=0,Poorquality,Attention;

int Att\_Avg=0,Plength,Temp;

int k=0,j=0;

void setup()

{

Serial.begin(BAUDRATE); // USB

pinMode(LED, OUTPUT);

pinMode(Mot\_L\_Plus, OUTPUT);

pinMode(Mot\_L\_Minus, OUTPUT);

pinMode(Mot\_R\_Plus, OUTPUT);

pinMode(Mot\_R\_Minus, OUTPUT);

}

byte ReadOneByte() // One Byte Read Function

{

int ByteRead;

while(!Serial.available());

ByteRead = Serial.read();

return ByteRead;

}

void loop() // Main Function

{

while (1)

{

if(ReadOneByte() == 170) // AA 1 st Sync data

{

if(ReadOneByte() == 170) // AA 2 st Sync data

{

Plength = ReadOneByte();

if(Plength == 32) // Big Packet

{

generatedchecksum = 0;

for(int i = 0; i < Plength; i++)

{

payloadData[i] = ReadOneByte(); //Read payload into memory

generatedchecksum += payloadData[i] ;

}

generatedchecksum = 255 - generatedchecksum;

checksum = ReadOneByte();

if(checksum == generatedchecksum) // Varify Checksum

{

Poorquality = payloadData[1];

Attention = payloadData[29];

Serial.print(Attention);

Serial.print("\n");

Attention\_Fun(Attention);

}

}

}

}

}

}

void Robot\_Forword()

{

digitalWrite(Mot\_L\_Plus, LOW);

digitalWrite(Mot\_L\_Minus, HIGH);

digitalWrite(Mot\_R\_Plus, LOW);

digitalWrite(Mot\_R\_Minus, HIGH);

}

void Robot\_Stop()

{

digitalWrite(Mot\_L\_Plus, LOW);

digitalWrite(Mot\_L\_Minus, LOW);

digitalWrite(Mot\_R\_Plus, LOW);

digitalWrite(Mot\_R\_Minus, LOW);

}

void Attention\_Fun(byte data1)

{

if (Poorquality == 0)

{

if (j<3)

{

Temp += data1;

j++;

}

else

{

Temp += data1;

Att\_Avg = Temp/4;

Serial.print("Att\_Avg");

Serial.print(Att\_Avg);

Serial.print("\n");

if (Att\_Avg >= ATT\_TH)

{

digitalWrite(LED, HIGH);

Robot\_Forword();

}

else

{

digitalWrite(LED, LOW);

Robot\_Stop();

}

j=0;

Temp=0;

}

}

else

{

Robot\_Stop();

Serial.print("Signal Poor Quality");

Serial.print("\n");

}

}