//The motor and speaker pins in the Arduino are being defined

int speakerPin = 12;

int right\_motor\_speed\_pin = 3;

int right\_motor\_forward\_pin = 4;

int right\_motor\_backward\_pin = 5;

int left\_motor\_speed\_pin = 8;

int left\_motor\_forward\_pin = 9;

int left\_motor\_backward\_pin = 10;

//The pins are stated to be Output pins, meaning they are pins that the Arduino controls.

void setup(){

Serial.begin(115200);//Serial Monitor is set at 115200 baud rate

Serial.write(0xc2);//This instruction is used for pairing the RF dongle with the Mindwave.

//Initializing all the morot and speaker Arudino PINs.

pinMode(speakerPin, OUTPUT);

 pinMode(right\_motor\_speed\_pin, OUTPUT);

 pinMode(right\_motor\_forward\_pin, OUTPUT);

 pinMode(right\_motor\_backward\_pin, OUTPUT);

 pinMode(left\_motor\_speed\_pin, OUTPUT);

 pinMode(left\_motor\_forward\_pin, OUTPUT);

 pinMode(left\_motor\_backward\_pin, OUTPUT);

}

//The function below is used to determine whether the robot should turn or go in a straight line.

//If the attention is above 50, then the function will return the same value.

//If it is below .50, then the function will return the negative value which will make one of the motors to turn.

float calcDirection (float attention) {

 if (attention>.40){

 return attention;

 }

 else {

 return -attention;

 }

}

//The ON function is used for making a PIN High.

void on(int pin){

digitalWrite(pin, HIGH);

}

//The OFF function is used for making a PIN Low.

void off(int pin){

digitalWrite(pin, LOW);

}

//Main Program

void loop(){

float att = getAttention(); //Get the attention value from Mindwave.

if (att > 0){

Serial.println("The value of att");

Serial.print(att);

off(right\_motor\_backward\_pin);

on(right\_motor\_forward\_pin);

analogWrite(right\_motor\_speed\_pin, (255\*att)); // Move the robot forward.

off(left\_motor\_backward\_pin);

on(left\_motor\_forward\_pin);

//Move or turn the robot depending on the attention value. If the attention value is greater than 50 the robot moves forward else turns.

analogWrite(left\_motor\_speed\_pin, ((calcDirection(att))\*255));

 }

}

//This function is used for getting one byte data from transmitted by the EEG headset to the dongle

byte readOneByte()

{

while (!Serial.available()) {

 delay(5);

};

return Serial.read();

}

//This function is used for getting the attention value from the Neurosky Headset.

float getAttention() {

byte generatedChecksum = 0;

byte checksum = 0;

int payLoadLength = 0;

byte PayLoadData[64] = { 0 };

int poorQuality = 0;

float attention = 0;

Serial.flush();

if (170 != readOneByte())return -1;

if (170 != readOneByte())return -1;

//Get how long the data is sent. The first byte of the data sent from the headset tells how many bytes are transmitted from the Headset. If 4 bytes of data is transmitted, then the first byte will be 4.

payLoadLength = readOneByte();

 if (payLoadLength > 169 ) return -2;

generatedChecksum = 0;

//Read all the bytes returned from the headset and store it in PayLoadData array. Also generate the check sum from the data being read.

 for (int i = 0; i < payLoadLength; i++) {

 PayLoadData[i] = readOneByte();

 generatedChecksum += PayLoadData[i];

 }

 generatedChecksum = 255 - generatedChecksum;

 //Get the CheckSum from the data stream transmitted by the headset.

 checksum = readOneByte();

//If the calculated Check sum (Generated CheckSum) is not equal to the check sum transmitted by the Headset, then return error code -3.

if (checksum !=generatedChecksum) return -3;

//The following loop analyzes the payload read to see if it is related Attention or Meditation

 for (int i = 0; i < payLoadLength; i++) {

 switch (PayLoadData[i]) {

 case 0xD0:

//If the data read is Hexa D0, this shows the EEG headset is paired with the RF dongle

 sayHeadsetConnected();

break;

 case 4:

 i++;

attention = PayLoadData[i]; //If the data read is Hexa 4, the value of the next byte will be the attention value.

 break;

 case 2:

 i++;

 poorQuality = PayLoadData[i];

 if (200 == poorQuality) {

 return -4;

 }

 break;

 case 0xD1: Serial.println("Headset not found.");

 case 0xD2: Serial.println("Headset Disconnected");

 case 0xD3: Serial.println("Request Denied") ;

 case -70:

 wave(speakerPin, 900, 500);

 return -5;

 break;

 case 0x80:

 i = i + 3;

 break;

 case 0x83:

 i = i + 25;

 break;

 }

 }

 return (float) attention/ 100; //Return the value of attention which we are interested in.

}

//This function is called when the headset has paired with the RF dongle.

void sayHeadsetConnected() {

wave(speakerPin, 440, 40);

delay(25);

wave(speakerPin, 300, 20);

wave(speakerPin, 540,40);

delay(25);

wave(speakerPin, 440, 20);

 wave(speakerPin, 640, 40);

 delay(25);

 wave(speakerPin, 540, 40);

 delay(25);

}

//Functon for playing the frequency note for the duration passed as the argument.

void wave(int pin, float frequency, int duration) {

 float period = 1/frequency \* 1000 \* 1000;

long int startTime = millis();

 while (millis()-startTime < duration) {

 digitalWrite(pin, HIGH);

 delayMicroseconds(period / 2);

 digitalWrite(pin, LOW);

 delayMicroseconds(period / 2);

 }

}