Analogue DC Controller using a TV Remote

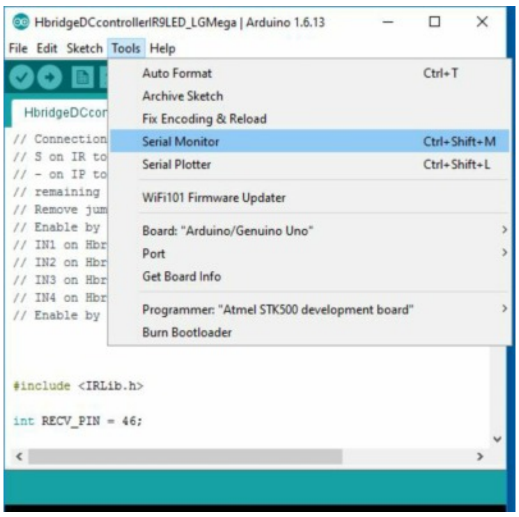
This project began when a fellow Club Member said that he would like to be able to operate his modular layout from the front rather than from ‘behind the scenes’. The first idea was to use Bluetooth for the remote control but experiments showed that ‘pairing’ was not 100% reliable. The next choice was infra-red. Initially a cheap infra-red remote control from eBay was used when it was realised that a discarded TV remote would do the trick equally as well and provide more control buttons.

The main components comprise an Arduino, an Hbridge shield and a 12 volt regulated power supply. Whilst development was carried out on an Arduino, the final model used an Arduino Nano which is an Arduino equivalent but makes a smaller, neater final product.

The job of the Arduino is to interpret the pulses coming from the infra-red remote into voltage changes to drive the trains. However, the current taken by a loco is too high for the output of the Arduino and so a device has to used that handles this current whilst taking the commands from the Arduino itself. The L298 Hbridge is a simple and very inexpensive solution. This device is designed to work a 2-phase stepper motor and therefore has two output pairs of connections (one for each stepper motor phase). These outputs are short circuit protected, can handle the current required by a loco with ease and can be divided into two independent DC Analogue outputs. i.e. two tracks can be powered from the one device. A further advantage that this device provides is a regulated 5 volt supply for the Arduino Nano.

IR Remote Codes

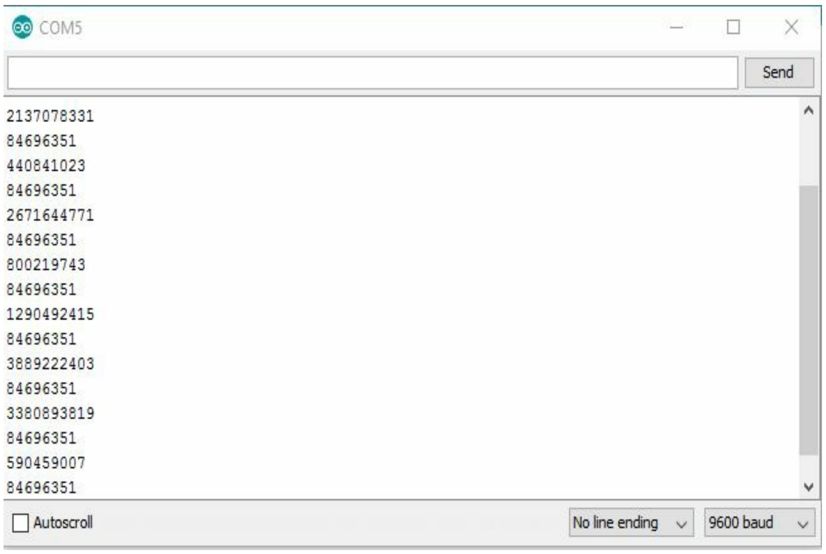
TV Infra-red remotes output a stream of numbers, different for each button pressed. The first task is to discover what each of these numbers are. To do this, a simple program (or ‘sketch’ to use the Arduino parlance) is required. The Arduino programming software, downloadable free and safely from https://www.arduino.cc/en/Main/Software for Windows, Linux or Mac, has a facility called the ‘Serial Monitor’ that will allow these codes to be viewed. From this a list of what numbers are produced by each button can be prepared. Once these codes are known then it will be possible to proceed to writing the sketch for the IR Train Controller.


HbridgeDCcontrollerIR9LED_LGMega | Arduino 1.6.13
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 Firmware Updater
Board: "Arduino/Genuino Uno"
Port
Get Board Info
Programmer: "Atmel STK500 development board"
Burn Bootloader
HbridgeDCcon

```
// Connection
// S on IR to
// - on IP to
// remaining
// Remove jum
// Enable by
// IN1 on Hbr
// IN2 on Hbr
// IN3 on Hbr
// IN4 on Hbr
// Enable by

#include <IRLib.h>

int RECV_PIN = 46;
```



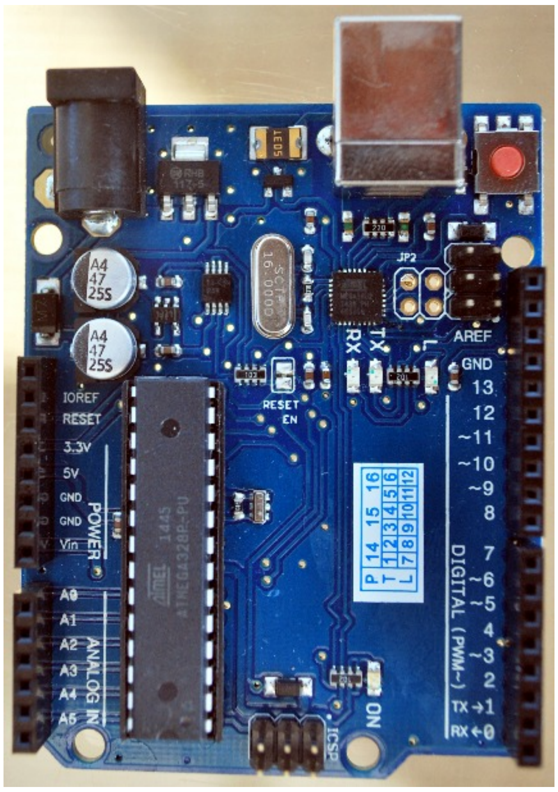
```
2137078331
84696351
440841023
84696351
2671644771
84696351
800219743
84696351
1290492415
84696351
3889222403
84696351
3380893819
84696351
590459007
84696351
```

There is often a repeating code that outputs even when a button is not being pressed and these should be ignored. It will be obvious which is the correct number at the point of pressing the relevant button.

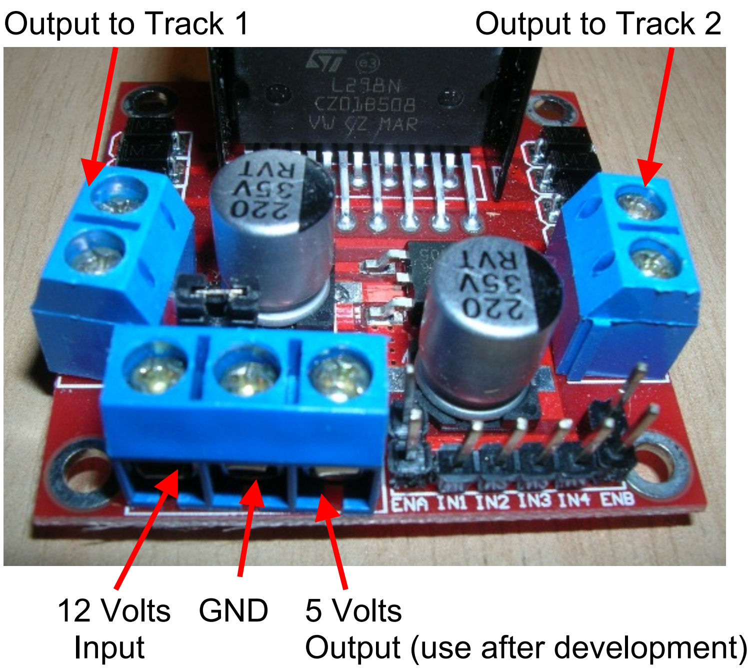
The Development Hardware

During development, the Arduino receives its power from the USB connection to the computer. A separate 12 volt regulated power supply should power the Hbridge. After development the Hbridge will be used to power the 5 volts required by the Arduino.

The Arduino



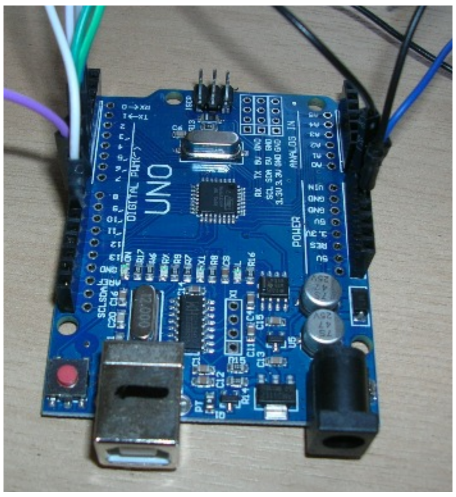
The Hbridge

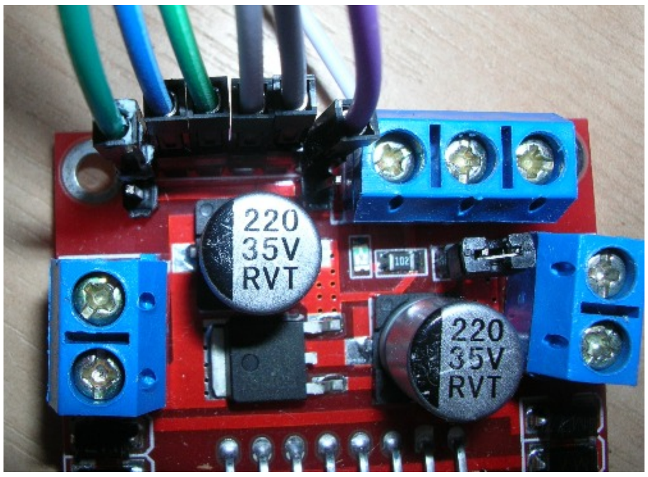


The IR Detector

Connections

1. S on IR Detector to Arduino pin 11
2. - (minus) on IR Dectector to Arduino Pin GND (left)
3. Middle pin on IR Detector to Arduino Pin 5v
4. Remove jumpers ENA and ENB on Hbridge
5. Hbridge ENA to Arduino Pin 10
6. Hbridge IN1 to Arduino Pin 5
7. Hbridge IN2 to Arduino Pin 6
8. Hbridge IN3 to Arduino Pin 8
9. Hbridge IN4 to Arduino Pin 7
10. Hbridge ENB to Arduino Pin 9
11. Hbridge GND to Arduino GND (Right)



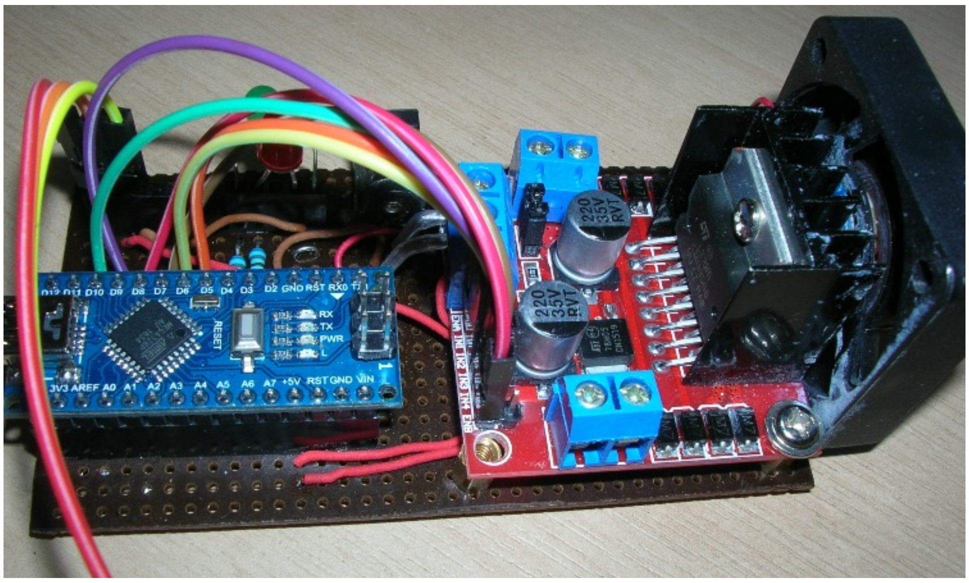


The Production Model

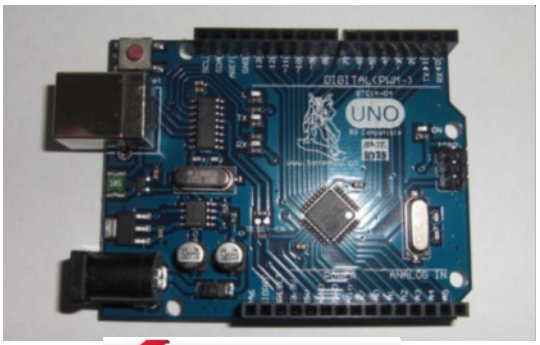
The production model uses an Arduino Nano, a smaller, cheaper replacement. Mounted on socket pins onto strip board (e.g. veroboard). The 5 volt feed available on the Hbridge (once it is fed with over 7v) is used to power the Nano. The developed program is uploaded to the Nano via the miniature USB port.

LED's were added to show which channel is the current operations output to be controlled. These will have a 1 k resistor in series with the LED and be connected to pins 3 and 4 on the Nano. The LED cathodes will go to GND.

Although strictly unnecessary a small 12 volt miniature fan was added (super glued) in case the unit is situated in an enclosed space such as a building on the layout.



Parts List



UNO R3 ATmega328P, CH340T USB Arduino Compatible Microcontroller Board

£5.45

Dual Bridge L298N Stepper Motor Driver Controller Board Module

£3.27

Nano V3.0 For Arduino with CH340G 5V 16M compatible ATmega328P

£3.33

Remote Control and Sensor

£2.59

10pcs Dupont Cables M-F, M-M, F-F Jumper Breadboard Wire GPIO Ribbon

£1.20 per type

AC 100-240V to DC 12V/2A Converter UK Regulated Power Supply Adapter Transformer = 5.5 mm x 2.1 mm plug

£6.79

Sketch for reading IR Remote Codes in Serial Monitor

```
#include <IRLib.h>

int RECV_PIN = 11;

IRrecv My_Receiver(RECV_PIN);
IRdecode My_Decoder;
IRdecodeHash My_Hash_Decoder;

void setup() {
  My_Receiver.enableIRIn(); // Start the receiver
  Serial.begin(9600);
  delay(2000);while(!Serial);//delay for Leonardo
}

void loop() {
  if (My_Receiver.GetResults(&My_Decoder)) {      //Puts results in My_Decoder
    My_Hash_Decoder.copyBuf(&My_Decoder);      //copy the results to the hash decoder
    My_Decoder.decode();
    My_Hash_Decoder.decode();
    Serial.println(My_Hash_Decoder.hash, DEC);
    delay(1000);
    My_Receiver.resume();
  }
}
```

Sample Code for IR Controller (Sketch)

```
// Connections
// S on IR to pin 11
// - on IP to Pin GND
// remaining pin on IR to 5v
// Remove jumpers on each ebale on Hbridge
// Enable by IN1 to pin 10
// IN1 on Hbridge to Pin 5
// IN2 on Hbridge to Pin 6
// IN3 on Hbridge to Pin 8
// IN4 on Hbridge to Pin 7
// Enable by IN4 to Pin 9

// Call IR Library
#include <IRLib.h>

int RECV_PIN = 11;

IRrecv My_Receiver(RECV_PIN);
IRdecode My_Decoder;
IRdecodeHash My_Hash_Decoder;

// ******* Define the button codes for LG Remote
#define RIGHT_ARROW   2340092731
#define LEFT_ARROW    2086679999
#define OK            3733801123
#define UP_ARROW      1674690463
#define DOWN_ARROW    3179222947
#define BUTTON_0      3509629087
#define BUTTON_1      719194275
#define BUTTON_2      2137078331
#define BUTTON_3      440841023
#define BUTTON_4      2671644771
#define BUTTON_5      800219743
#define BUTTON_6      1290492415
#define BUTTON_7      3889222403
#define BUTTON_8      3380893819
#define BUTTON_9      590459007
#define RED           3576004571
#define GREEN         2158120515
#define YELLOW        255616091
#define BLUE          533969343
#define PLUS          2651459839
#define MINUS         2398047107
#define PROGUP        2250803747
#define PROGDN        379378719
#define QMENU         84696351
#define MENU          2900869695
#define GUIDE         1307419706
#define RETURN        1073074659
#define INFO          3693363464
#define AVMODE        84696351
#define FAV           2183527131
#define MUTE          1907962371
#define LIST          1161620931
#define QVIEW         1743457375
//#define TV
//#define STB
//#define POWER
```

```
int motorPin = 9 ;
int motorValue9 = 0;
int motorValue10 = 0;
int startmotorValue = 0;
int delaytime = 25 ;
int fwdbkwd9 = 0 ;
int fwdbkwd10 = 0 ;
// setup speed table
int speed0 = 0;
int speed1 = 75 ;
int speed2 = 97 ;
int speed3 = 119 ;
int speed4 = 141 ;
int speed5 = 163 ;
int speed6 = 185 ;
int speed7 = 207 ;
int speed8 = 229 ;
int speed9 = 255 ;
int increment = 5;
// Arduino pins from Hbridge
int In1 = 5 ;
int In2 = 6 ;
int In3 = 7 ;
int In4 = 8 ;
long unsigned mvalue  ;

void setup() {
  // put your setup code here, to run once:
  My_Receiver.enableIRIn(); // Start the receiver
  Serial.begin(9600);
  delay(2000); while (!Serial); //delay for Leonardo
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(In1, OUTPUT);
  pinMode(In2, OUTPUT);
  pinMode(In3, OUTPUT);
  pinMode(In4, OUTPUT);
  pinMode(10, OUTPUT);

  digitalWrite(In1, LOW);
  digitalWrite(In2, HIGH);
  digitalWrite(In3, LOW);
  digitalWrite(In4, HIGH);

  setPwmFrequency(9, 256 ) ;
  setPwmFrequency(10, 256 ) ;
}

void loop() {
  mvalue = 0;
  if (My_Receiver.GetResults(&My_Decoder)) {        //Puts results in My_Decoder
    My_Hash_Decoder.copyBuf(&My_Decoder);       //copy the results to the hash decoder
    My_Decoder.decode();
    My_Hash_Decoder.decode();
    mvalue = (My_Hash_Decoder.hash) ;
    //Serial.println(My_Hash_Decoder.hash, DEC);
    delay(1000);
    My_Receiver.resume();
  }
```

```
if (mvalue == RED )  {
  motorPin = 9 ;
  digitalWrite(3, HIGH) ;     // turn Channel A LED on
  digitalWrite(4, LOW);       // turn Channel B LED of
  //Serial.println(motorPin); // Do something interesting with this value
  //    mvalue = 0;
}
if (mvalue == BLUE )  {
  motorPin =  10 ;
  digitalWrite(3, LOW) ;    // turn Channel A LED off
  digitalWrite(4, HIGH);    // turn Channel B LED on
  //    Serial.println(motorPin); // Do something interesting with this value
  //   mvalue = 0;
}
//mvalue=BUTTON_7 ;
if (motorPin == 9) {
  switch (mvalue) {
    case LEFT_ARROW :
      if (fwdbkwd9 == 1) {
        startmotorValue = motorValue9 + 1 ;
        while (motorValue9 > 0) {
          analogWrite(motorPin, motorValue9 - 1) ;
          motorValue9 = motorValue9 - 1 ;
          delay(delaytime);
        }
        digitalWrite(In1, HIGH);
        digitalWrite(In2, LOW);
        fwdbkwd9 = 0 ;
        while (motorValue9 < startmotorValue) {
          analogWrite(motorPin, motorValue9 + 1);
          motorValue9 = motorValue9 + 1 ;
          delay(delaytime);
        }
      }
      break;
    case  RIGHT_ARROW :
      if (fwdbkwd9 == 0) {
        startmotorValue = motorValue9 + 1;
        while (motorValue9 > 0) {
          analogWrite(motorPin, motorValue9 - 1) ;
          motorValue9 = motorValue9 - 1 ;
          delay(delaytime);
        }
        digitalWrite(In1, LOW);
        digitalWrite(In2, HIGH);
        fwdbkwd9 = 1 ;
        while (motorValue9 < startmotorValue) {
          analogWrite(motorPin, motorValue9 + 1);
          motorValue9 = motorValue9 + 1 ;
          delay(delaytime);
        }
      }
      break;
    case UP_ARROW :
      motorValue9 = motorValue9 + increment ;
      analogWrite(motorPin, motorValue9) ;
      mvalue = 0;
      break;
    case DOWN_ARROW :
      motorValue9 = motorValue9 - increment ;
  analogWrite(motorPin, motorValue9) ;
  mvalue = 0 ;
  break;
case BUTTON_1 :
  if (motorValue9 > speed1) {
    while (motorValue9 > speed1) {
      analogWrite(motorPin, motorValue9 - 1);
      motorValue9 = motorValue9 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue9 < speed1) {
      analogWrite(motorPin, motorValue9 + 1);
      motorValue9 = motorValue9 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_2 :
  if (motorValue9 > speed2) {
    while (motorValue9 > speed2) {
      analogWrite(motorPin, motorValue9 - 1);
      motorValue9 = motorValue9 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue9 < speed2) {
      analogWrite(motorPin, motorValue9 + 1);
      motorValue9 = motorValue9 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_3:
  if (motorValue9 > speed3) {
    while (motorValue9 > speed3) {
      analogWrite(motorPin, motorValue9 - 1);
      motorValue9 = motorValue9 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue9 < speed3) {
      analogWrite(motorPin, motorValue9 + 1);
      motorValue9 = motorValue9 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_4  :
  if (motorValue9 > speed4) {
    while (motorValue9 > speed4) {
      analogWrite(motorPin, motorValue9 - 1);
      motorValue9 = motorValue9 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue9 < speed4) {
      analogWrite(motorPin, motorValue9 + 1);
      motorValue9 = motorValue9 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_5 :
  if (motorValue9 > speed5) {
    while (motorValue9 > speed5) {
      analogWrite(motorPin, motorValue9 - 1);
      motorValue9 = motorValue9 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue9 < speed5) {
      analogWrite(motorPin, motorValue9 + 1);
      motorValue9 = motorValue9 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_6 :
  if (motorValue9 > speed6) {
    while (motorValue9 > speed6) {
      analogWrite(motorPin, motorValue9 - 1);
      motorValue9 = motorValue9 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue9 < speed6) {
      analogWrite(motorPin, motorValue9 + 1);
      motorValue9 = motorValue9 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_7 :
  if (motorValue9 > speed7) {
    while (motorValue9 > speed7) {
      analogWrite(motorPin, motorValue9 - 1);
      motorValue9 = motorValue9 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue9 < speed7) {
      analogWrite(motorPin, motorValue9 + 1);
      motorValue9 = motorValue9 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_8 :
  if (motorValue9 > speed8) {
    while (motorValue9 > speed8) {
      analogWrite(motorPin, motorValue9 - 1);
      motorValue9 = motorValue9 - 1 ;
```

```
          delay(delaytime);
        }
      }
      else {
        while (motorValue9 < speed8) {
          analogWrite(motorPin, motorValue9 + 1);
          motorValue9 = motorValue9 + 1 ;
          delay(delaytime);
        }
      }
      break;
    case BUTTON_9 :
      while (motorValue9 < speed9) {
        analogWrite(motorPin, motorValue9 + 1);
        motorValue9 = motorValue9 + 1 ;
        delay(delaytime);
      }
    case BUTTON_0 :
      if (motorValue9 > speed0) {
        while (motorValue9 > speed0) {
          analogWrite(motorPin, motorValue9 - 1);
          motorValue9 = motorValue9 - 1 ;
          delay(delaytime);
        }
      }
      break;
    case OK  :
      analogWrite(motorPin, 0);
      motorValue9 = 0 ;
      break;
  }
}
// ********************************************************* motorPin=10
if (motorPin == 10) {
  switch (mvalue) {
    case LEFT_ARROW :
      if (fwdbkwd10 == 1) {
        startmotorValue = motorValue10 + 1 ;
        while (motorValue10 > 0) {
          analogWrite(motorPin, motorValue10 - 1) ;
          motorValue10 = motorValue10 - 1 ;
          delay(delaytime);
        }
        digitalWrite(In3, HIGH);
        digitalWrite(In4, LOW);
        fwdbkwd10 = 0 ;
        while (motorValue10 < startmotorValue) {
          analogWrite(motorPin, motorValue10 + 1);
          motorValue10 = motorValue10 + 1 ;
          delay(delaytime);
        }
      }
      break;
    case RIGHT_ARROW :
      if (fwdbkwd10 == 0) {
        startmotorValue = motorValue10 + 1;
        while (motorValue10 > 0) {
          analogWrite(motorPin, motorValue10 - 1) ;
          motorValue10 = motorValue10 - 1 ;
          delay(delaytime);
      }
      digitalWrite(In3, LOW);
      digitalWrite(In4, HIGH);
      fwdbkwd10 = 1 ;
      while (motorValue10 < startmotorValue) {
        analogWrite(motorPin, motorValue10 + 1);
        motorValue10 = motorValue10 + 1 ;
        delay(delaytime);
      }
    }
    break;
  case UP_ARROW :
    motorValue10 = motorValue10 + increment ;
    analogWrite(motorPin, motorValue10) ;
    mvalue = 0;
  case DOWN_ARROW :
    motorValue10 = motorValue10 - increment ;
    analogWrite(motorPin, motorValue10) ;
    mvalue = 0 ;
    break;
  case BUTTON_1 :
    if (motorValue10 > speed1) {
      while (motorValue10 > speed1) {
        analogWrite(motorPin, motorValue10 - 1);
        motorValue10 = motorValue10 - 1 ;
        delay(delaytime);
      }
    }
    else {
      while (motorValue10 < speed1) {
        analogWrite(motorPin, motorValue10 + 1);
        motorValue10 = motorValue10 + 1 ;
        delay(delaytime);
      }
    }
    break;
  case BUTTON_2 :
    if (motorValue10 > speed2) {
      while (motorValue10 > speed2) {
        analogWrite(motorPin, motorValue10 - 1);
        motorValue10 = motorValue10 - 1 ;
        delay(delaytime);
      }
    }
    else {
      while (motorValue10 < speed2) {
        analogWrite(motorPin, motorValue10 + 1);
        motorValue10 = motorValue10 + 1 ;
        delay(delaytime);
      }
    }
    break;
  case BUTTON_3 :
    if (motorValue10 > speed3) {
      while (motorValue10 > speed3) {
        analogWrite(motorPin, motorValue10 - 1);
        motorValue10 = motorValue10 - 1 ;
        delay(delaytime);
      }
    }
  else {
    while (motorValue10 < speed3) {
      analogWrite(motorPin, motorValue10 + 1);
      motorValue10 = motorValue10 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_4 :
  if (motorValue10 > speed4) {
    while (motorValue10 > speed4) {
      analogWrite(motorPin, motorValue10 - 1);
      motorValue10 = motorValue10 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue10 < speed4) {
      analogWrite(motorPin, motorValue10 + 1);
      motorValue10 = motorValue10 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_5 :
  if (motorValue10 > speed5) {
    while (motorValue10 > speed5) {
      analogWrite(motorPin, motorValue10 - 1);
      motorValue10 = motorValue10 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue10 < speed5) {
      analogWrite(motorPin, motorValue10 + 1);
      motorValue10 = motorValue10 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_6 :
  if (motorValue10 > speed6) {
    while (motorValue10 > speed6) {
      analogWrite(motorPin, motorValue10 - 1);
      motorValue10 = motorValue10 - 1 ;
      delay(delaytime);
    }
  }
  else {
    while (motorValue10 < speed6) {
      analogWrite(motorPin, motorValue10 + 1);
      motorValue10 = motorValue10 + 1 ;
      delay(delaytime);
    }
  }
  break;
case BUTTON_7 :
  if (motorValue10 > speed7) {
    while (motorValue10 > speed7) {
      analogWrite(motorPin, motorValue10 - 1);
```

```
          motorValue10 = motorValue10 - 1 ;
          delay(delaytime);
        }
      }
      else {
        while (motorValue10 < speed7) {
          analogWrite(motorPin, motorValue10 + 1);
          motorValue10 = motorValue10 + 1 ;
          delay(delaytime);
        }
      }
      break;
    case BUTTON_8 :
      if (motorValue10 > speed8) {
        while (motorValue10 > speed8) {
          analogWrite(motorPin, motorValue10 - 1);
          motorValue10 = motorValue10 - 1 ;
          delay(delaytime);
        }
      }
      else {
        while (motorValue10 < speed8) {
          analogWrite(motorPin, motorValue10 + 1);
          motorValue10 = motorValue10 + 1 ;
          delay(delaytime);
        }
      }
      break;
    case BUTTON_9 :
      while (motorValue10 < speed9) {
        analogWrite(motorPin, motorValue10 + 1);
        motorValue10 = motorValue10 + 1 ;
        delay(delaytime);
      }
      break;
    case BUTTON_0 :
      if (motorValue10 > speed0) {
        while (motorValue10 > speed0) {
          analogWrite(motorPin, motorValue10 - 1);
          motorValue10 = motorValue10 - 1 ;
          delay(delaytime);
        }
      }
      break;
    case OK :
      analogWrite(motorPin, 0);
      motorValue10 = 0 ;
      break;
  }
 }
 //Serial.println(motorPin);
 //Serial.println(mvalue);
}



void setPwmFrequency(int pin, int divisor) {
 byte mode;
 if (pin == 5 || pin == 6 || pin == 9 || pin == 10) {
   switch (divisor) {
    case 1: mode = 0x01; break;
    case 8: mode = 0x02; break;
    case 64: mode = 0x03; break;
    case 256: mode = 0x04; break;
    case 1024: mode = 0x05; break;
    default: return;
  }
  if (pin == 5 || pin == 6) {
    TCCR0B = TCCR0B & 0b11111000 | mode;
  } else {
    TCCR1B = TCCR1B & 0b11111000 | mode;
  }
 } else if (pin == 3 || pin == 11) {
  switch (divisor) {
    case 1: mode = 0x01; break;
    case 8: mode = 0x02; break;
    case 32: mode = 0x03; break;
    case 64: mode = 0x04; break;
    case 128: mode = 0x05; break;
    case 256: mode = 0x06; break;
    case 1024: mode = 0x7; break;
    default: return;
  }
  TCCR2B = TCCR2B & 0b11111000 | mode;
 }
}
```