Κώδικας για 3τροχο αυτοκινητάκι

Απλός έλεγχος με υπέρυθρο χειριστήριο

```
#include "IRremote.h"
// LAFVIN control
/*-----( Declare Constants )-----*/
int receiver = 3; // pin 1 of IR receiver to Arduino digital pin 3
int val;
/*-----( Declare objects )-----*/
IRrecv irrecv(receiver); // create instance of 'irrecv'
decode_results results; // create instance of 'decode_results'
/*-----( Declare Variables )-----*/



//Motor Connections
//Change this if you wish to use another diagram
#define EnA 10
#define EnB 5
// #define EnB A3
// #define EnA A2

#define In1 9
#define In2 8
#define In3 7
#define In4 6
```

```
void setup()
{
// All motor control pins are outputs
pinMode(EnA, OUTPUT);
pinMode(EnB, OUTPUT);
pinMode(In1, OUTPUT);
pinMode(In2, OUTPUT);
pinMode(In3, OUTPUT);
pinMode(In4, OUTPUT);
irrecv.enableIRIn(); // Start the receiver
Serial.begin(9600);
Serial.println("Starting!");
}
/*-----( Declare User-written Functions )-----*/
void translateIR() // takes action based on IR code received

// describing Car MP3 IR codes

{

switch(results.value)

{


case 0xFF629D:
// Serial.println(" UP ");
break;

case 0xFFE21D:
// Serial.println(" CH+ ");
break;

case 0xFF22DD:
// Serial.println(" PREV ");
goBack();
break;
```

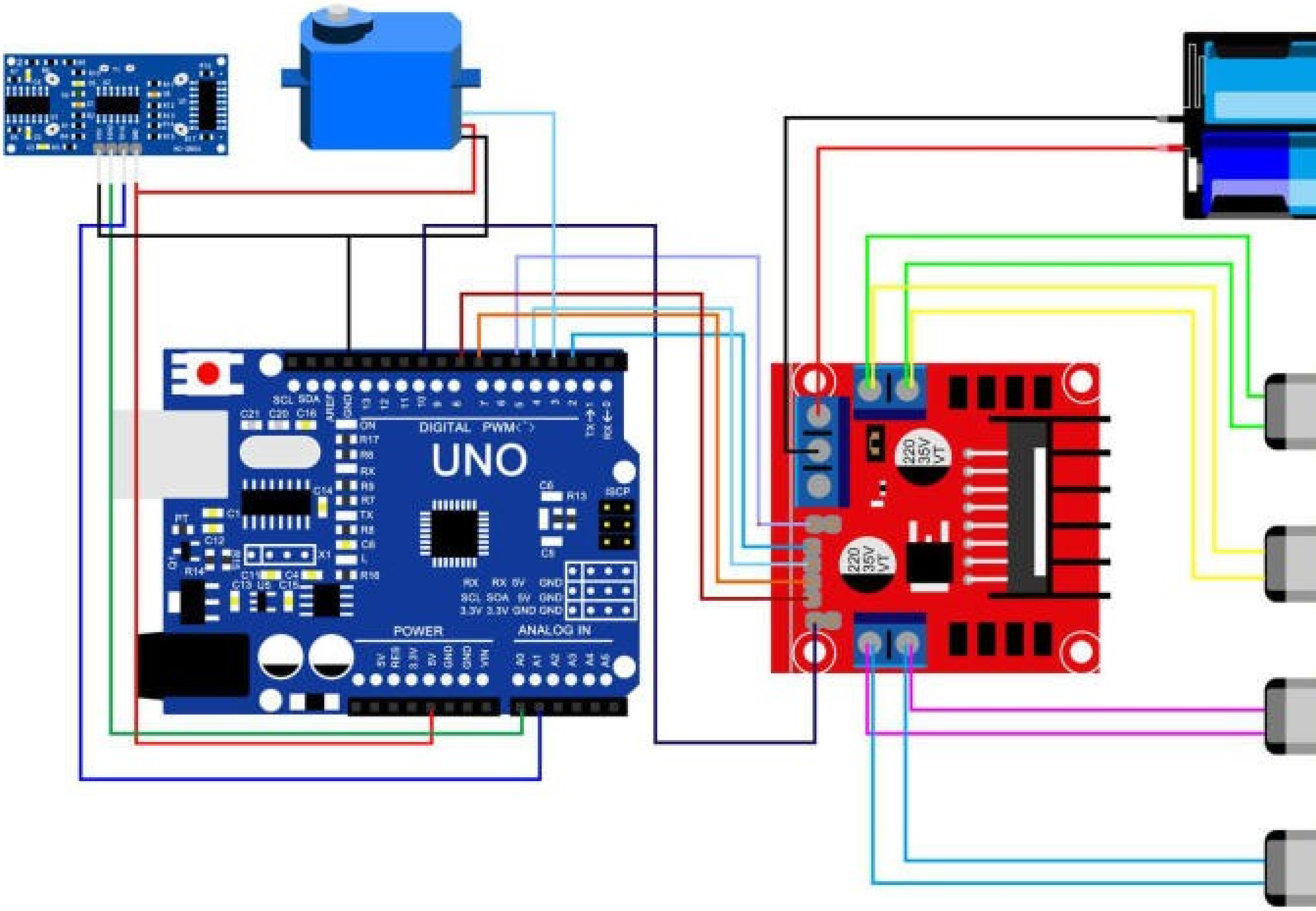
```
case 0xFF02FD:
// Serial.println(" OK ");
stop1();
break;

case 0xFFC23D:
// Serial.println(" PLAY/PAUSE/NEXT ");
goStraight();
break;


case 0xFFA857:
// Serial.println(" DOWN ");
break;

case 0xFF6897:
// Serial.println(" 1 ");
a();
break;

case 0xFF9867:
// Serial.println(" 2 ");
break;
b();
case 0xFFB04F:
// Serial.println(" 3 ");
break;

case 0xFF30CF:
// Serial.println(" 4 ");
break;

case 0xFF18E7:
// Serial.println(" 5 ");
break;

case 0xFF7A85:
// Serial.println(" 6 ");
```

```
break;

case 0xFF10EF:
// Serial.println(" 7 ");
break;

case 0xFF38C7:
// Serial.println(" 8 ");
break;

case 0xFF5AA5:
// Serial.println(" 9 ");
break;

case 0xFF42BD:
// Serial.println(" * ");
break;

case 0xFF4AB5:
// Serial.println(" 0 ");
break;

case 0xFF52AD:
// Serial.println(" # ");
break;

// default:
// Serial.println(" PREV ");

}

delay(500);

} //END translateIR

// If you use the IRrecvDemo Sketch (above) and count the 21 buttons from left to right and top to bottom,
// the codes received are these: (NOTE: Receiving "FFFFFFFF" means "repeat" if you hold the button down.)
```

```
//run both motors in the same direction
void goStraight()
{
/* Serial.println("Insert value:");
while (Serial.available()==0) { //Wait for user input

}
val=Serial.parseInt(); */

// turn on motor A
digitalWrite(In1,HIGH );
digitalWrite(In2, LOW);

// turn on motor B
digitalWrite(In3, HIGH);
digitalWrite(In4, LOW);
// set speed to 150 out 255
analogWrite(EnA, 150);
Serial.println("Forward");
Serial.print("EnA=");
Serial.println(analogRead(EnA));
// set speed to 150 out 255
analogWrite(EnB, 150);
Serial.print("EnB=");
Serial.println(analogRead(EnB));
delay(1000);
// now turn off motors
digitalWrite(In1, LOW);
digitalWrite(In2, LOW);
digitalWrite(In3, LOW);
digitalWrite(In4, LOW);
}

void goBack()
{
// turn on motor A
digitalWrite(In1, LOW);
digitalWrite(In2, HIGH);
Serial.println("Backward");
Serial.print("EnA=");
Serial.println(analogRead(EnA));
// turn on motor B
digitalWrite(In3, LOW);
digitalWrite(In4, HIGH);
// set speed to 150 out 255
analogWrite(EnA, 250);
// set speed to 150 out 255
analogWrite(EnB, 250);
Serial.print("EnB=");
Serial.println(analogRead(EnB));
delay(1000);
// now turn off motors
digitalWrite(In1, LOW);
digitalWrite(In2, LOW);
digitalWrite(In3, LOW);
digitalWrite(In4, LOW);
}

void stop1()
{
// now turn off motors
digitalWrite(In1, LOW);
digitalWrite(In2, LOW);
digitalWrite(In3, LOW);
digitalWrite(In4, LOW);
}

void a() {
analogWrite(EnA, 150);
}

void b() {
analogWrite(EnB, 150);
}

void loop()
```

```
{
if (irrecv.decode(&results)) // have we received an IR signal?

{
// Serial.println(results.value, HEX); UN Comment to see raw values
translateIR();
irrecv.resume(); // receive the next value
}

// goStraight();
delay(1000);
}
```


Αποφυγή εμποδίου



Κώδικας

```
int inputPin=A0; // ultrasonic module ECHO to A0
int outputPin=A1; // ultrasonic module TRIG to A1
#define Lpwm_pin 5 //pin of controlling speed---- ENA of motor driver board
#define Rpwm_pin 10 //pin of controlling speed---- ENB of motor driver board
int pinLB=2; //pin of controlling turning---- IN1 of motor driver board
int pinLF=4; //pin of controlling turning---- IN2 of motor driver board
int pinRB=7; //pin of controlling turning---- IN3 of motor driver board
int pinRF=8; //pin of controlling turning---- IN4 of motor driver board
unsigned char Lpwm_val = 200; //initialized left wheel speed at 250
unsigned char Rpwm_val = 200; //initialized right wheel speed at 250
int Car_state=0; //the working state of car
int servopin=3; //defining digital port pin 3, connecting to signal line of servo motor
int myangle; //defining variable of angle
int pulsewidth; //defining variable of pulse width
unsigned char DuoJiao=60; //initialized angle of motor at 60°

void servopulse(int servopin,int myangle) //defining a function of pulse
{
pulsewidth=(myangle*11)+500; //converting angle into pulse width value at 500-2480
digitalWrite(servopin,HIGH); //increasing the level of motor interface to upmost
delayMicroseconds(pulsewidth); //delaying microsecond of pulse width value
digitalWrite(servopin,LOW); //decreasing the level of motor interface to the least
delay(20-pulsewidth/1000);
}
void Set_servopulse(int set_val)
{
for(int i=0;i<=10;i++) //giving motor enough time to turn to assigning point
servopulse(servopin,set_val); //invokimg pulse function
}
void M_Control_IO_config(void)
{
pinMode(pinLB,OUTPUT); // /pin 2
pinMode(pinLF,OUTPUT); // pin 4
pinMode(pinRB,OUTPUT); // pin 7
pinMode(pinRF,OUTPUT); // pin 8
pinMode(Lpwm_pin,OUTPUT); // pin 11 (PWM)
pinMode(Rpwm_pin,OUTPUT); // pin10(PWM)
}
void Set_Speed(unsigned char Left,unsigned char Right) //function of setting speed
{
analogWrite(Lpwm_pin,Left);
analogWrite(Rpwm_pin,Right);
}
void advance() // going forward
{
digitalWrite(pinRB,LOW); // making motor move towards right rear
digitalWrite(pinRF,HIGH);
digitalWrite(pinLB,LOW); // making motor move towards left rear
digitalWrite(pinLF,HIGH);
Car_state = 1;
}
void turnR() //turning right(dual wheel)
{
digitalWrite(pinRB,LOW); //making motor move towards right rear
digitalWrite(pinRF,HIGH);
digitalWrite(pinLB,HIGH);
digitalWrite(pinLF,LOW); //making motor move towards left front
Car_state = 4;
}
void turnL() //turning left(dual wheel)
{
digitalWrite(pinRB,HIGH);
digitalWrite(pinRF,LOW ); //making motor move towards right front
digitalWrite(pinLB,LOW); //making motor move towards left rear
digitalWrite(pinLF,HIGH);
Car_state = 3;
}
void stopp() //stop
{
digitalWrite(pinRB,HIGH);
digitalWrite(pinRF,HIGH);
digitalWrite(pinLB,HIGH);
digitalWrite(pinLF,HIGH);
Car_state = 5;
}
void back() //back up
```

```
{
digitalWrite(pinRB,HIGH); //making motor move towards right rear
digitalWrite(pinRF,LOW);
digitalWrite(pinLB,HIGH); //making motor move towards left rear
digitalWrite(pinLF,LOW);
Car_state = 2;
}



void Self_Control(void)//self-going, ultrasonic obstacle avoidance
{
int H;

Set_servopulse(DuoJiao);

H = Ultrasonic_Ranging(1);
delay(300);
if(Ultrasonic_Ranging(1) < 35)
{
stopp();
delay(100);
back();
delay(50);
}

if(Ultrasonic_Ranging(1) < 60)
{
stopp();
delay(100);
Set_servopulse(5);
int L = ask_pin_L(2);
delay(300);
Set_servopulse(177);
int R = ask_pin_R(3);
delay(300);

if(ask_pin_L(2) > ask_pin_R(3))
{
```

```
back();
delay(100);
turnL();
delay(400);
stopp();
delay(50);
Set_servopulse(DuoJiao);
H = Ultrasonic_Ranging(1);
delay(500);
}

if(ask_pin_L(2) <= ask_pin_R(3))
{
back();
delay(100);
turnR();
delay(400);
stopp();
delay(50);
Set_servopulse(DuoJiao);
H = Ultrasonic_Ranging(1);
delay(300);
}
if (ask_pin_L(2) < 35 && ask_pin_R(3)< 35)
{
stopp();
delay(50);
back();
delay(50);
}
}
else
{
advance();
}
}
int Ultrasonic_Ranging(unsigned char Mode)//function of ultrasonic distance detecting ,  MODE=1,  displaying,  no displaying under other situation
```

```
{
int old_distance;
digitalWrite(outputPin, LOW);
delayMicroseconds(2);
digitalWrite(outputPin, HIGH);
delayMicroseconds(10);
digitalWrite(outputPin, LOW);
int distance = pulseIn(inputPin, HIGH); // reading the duration of high level
distance= distance/58; // Transform pulse time to distance
if(Mode==1){
Serial.print("\n H = ");
Serial.print(distance,DEC);
return distance;
}
else return distance;
}
int ask_pin_L(unsigned char Mode)
{
int old_Ldistance;
digitalWrite(outputPin, LOW);
delayMicroseconds(2);
digitalWrite(outputPin, HIGH);
delayMicroseconds(10);
digitalWrite(outputPin, LOW);
int Ldistance = pulseIn(inputPin, HIGH);
Ldistance= Ldistance/58; // Transform pulse time to distance
if(Mode==2){
Serial.print("\n L = ");
Serial.print(Ldistance,DEC);
return Ldistance;
}
else return Ldistance;
}
int ask_pin_R(unsigned char Mode)
{
int old_Rdistance;
digitalWrite(outputPin, LOW);
delayMicroseconds(2);
digitalWrite(outputPin, HIGH); //
delayMicroseconds(10);
digitalWrite(outputPin, LOW);
int Rdistance = pulseIn(inputPin, HIGH);
Rdistance= Rdistance/58; // Transform pulse time to distance
if(Mode==3){
Serial.print("\n R = ");
Serial.print(Rdistance,DEC);
return Rdistance;
}
else return Rdistance;
}

void setup()
{
pinMode(servopin,OUTPUT); //setting motor interface as output
M_Control_IO_config(); //motor controlling the initialization of IO
Set_Speed(Lpwm_val,Rpwm_val); //setting initialized speed
Set_servopulse(DuoJiao); //setting initialized motor angle
pinMode(inputPin, INPUT); //starting receiving IR remote control signal
pinMode(outputPin, OUTPUT); //IO of ultrasonic module
Serial.begin(9600); //initialized serial port , using Bluetooth as serial port, setting
baud
stopp(); //stop
}
void loop()
{

Self_Control();
}
```