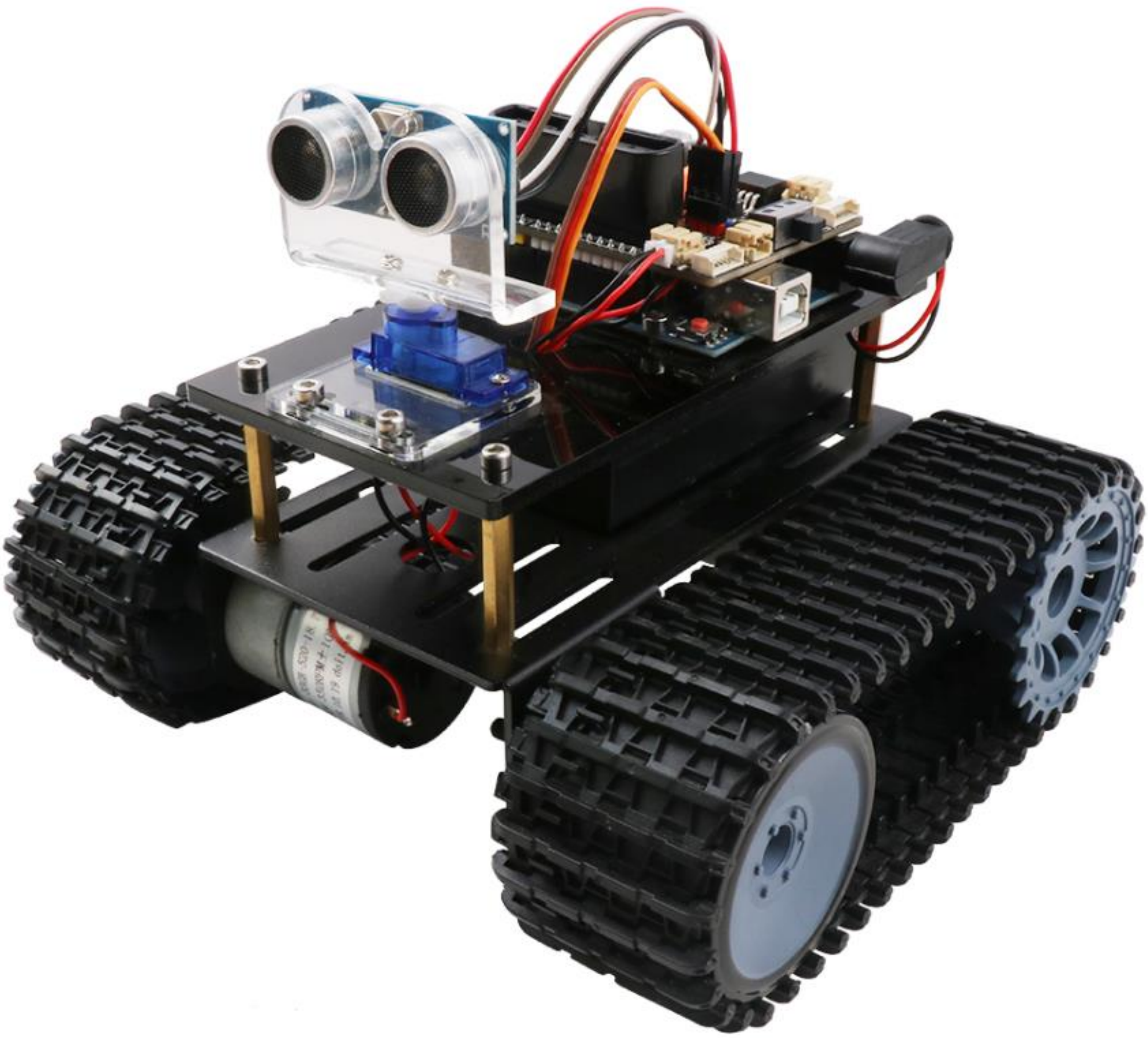
Panther-Tank V3.0 Instruction Manual



Get last update from https://github.com/keywish/keywish-panther-tank

CONTENTS

Chapter 1 PREFACE

1.1 Purpose

Our purpose is to offer a learning platform for DIY lovers, makers and beginners, help to get a better understanding of Arduino, and its expansion system design methods and principles, as well as the corresponding hardware debugging methods. Further deepen the understanding of the design and application of Arduino and its extended system.

The instruction manual mainly introduces the installation, hardware and software for Panther- Tank from the easy to the difficult and complicated. There are two parts for the instruction manual.

The first part mainly introduces how to use the common developing software and the method of downloading,debugging. And the second part mainly introduces about hardware and software. For the hardware part, it mainly introduces the functions, principles of every module, and for the software part, there are many examples for customers, it mainly introduces the applications of the Tank.

There are lots of detailed schematic diagrams and example codes for each module, which is strictly tested to ensure the accuracy and precision. Moreover, you can easily find the library files in the corresponding file folder and download through the UART/simulator to the Arduino uno R3 board for the corresponding functions . You can debug each module according to the course or directly assemble the car to enjoy the fun of a maker.

1.2 Product Introduction

"Tank" is ATMEGA328P-PU as the main control chip, and TB6612FGN is used as a multi-functional crawler car for motor drive chip. Compared with the traditional car, "Tank" is also equipped with wireless control (Bluetooth, infrared remote control). It can automatically avoid obstacles. Of course, Maker can also add or subtract other functions through its own Idea, such as adding automatic tracking, PS2 gamepad, adding wifi control, robotic arm, etc.

"Tank" is equipped with all kinds of materials, technical manuals, routines, etc., and teaches you from entry to proficiency. Every electronic enthusiast can easily get started and realize the functions they want.

Product features

- High power all metal geared motor
- Integral stamping molding kit,easier Installation,tighter
- 2400mAH,7.4v ,rechargeable li-battery,longer battery life,and more dynamic
- 2 RGB turn lights
- Buzzer Turn around reminder

- Infrared remote control
- Android App control

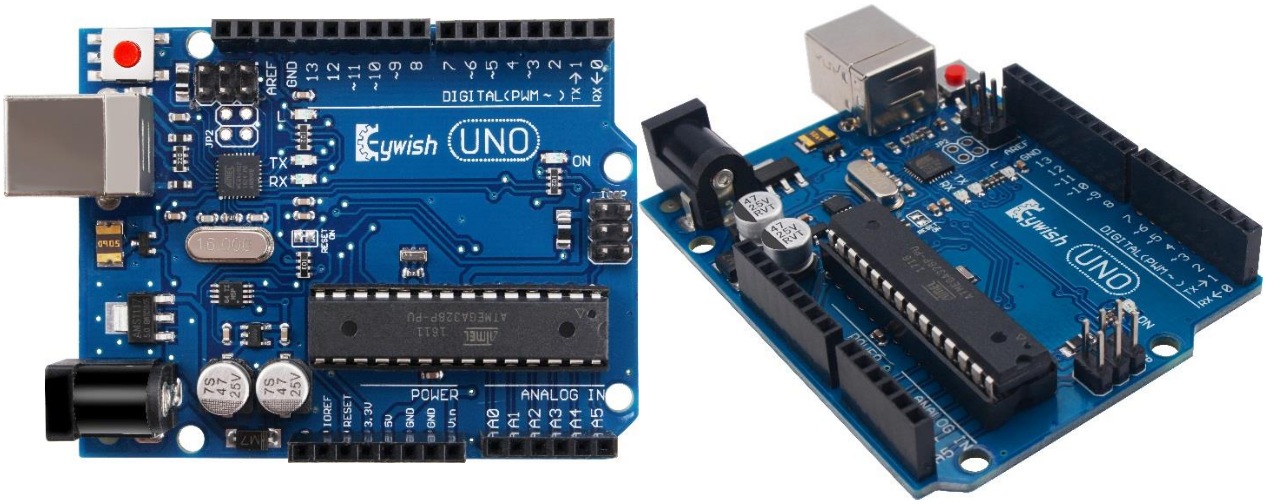
1.3 Product Introduction list:

 1 x Keywish UNO R3 Controller Board	 1 x Motor driver shield	 1 x 18650 li-battery box 1 x Power DC head	 The charger	 1 x Infrared remote	 2 x Velcro 1 x USB Cable
 2 x Driving wheel	 2 x Load-bearing wheel	 1 x Aluminum alloy plate	 1 x Acrylic button plate	 2 x track	 1 x Ultrasound Module
 1 x SG90 servo	 2 x DC Motor 2 x PH2 . 0mmx20cm Wire	1 x SG90 Servo fixed plate	1 x Ultrasonic Holder	 2 x Couplings	 2 x Top wire
 4 x M3*30mm Dual channel copper posts	 16 x M3*10mm screw	 6 x M3*8 Flat-headscrew	 2 x M3*12 Flat-headscrew	 2 x M4*51mm screw	 4 x M4 nut
 6 X M3 nut	 2 x M2 nut	 4 x M4 washer	 2 x M4 Plastic washer	Several cable ties	 1 x M4 Allen wrench
 1 x M3 Allen wrench	 1 x M2 Allen wrench	 1 x Cross sleeve	 1 x 3.0-7.5mm Phillips screwdriver	 20cm Dupont Line (4pin) 1 x Card needle	 1 x CD-ROM with tutorial

Chapter 2 Preparation

2.1 About Arduino uno R3

In Panther-Tank,we use Arduino uno r3 as main control board. Arduino uno r3 features 14 digitalinput/output (6 of them can be used as PWM output), six analog input inputs,one 16 MHz ceramic resonator,one USB connecter,one power adapter, one ICSP,one reset button.



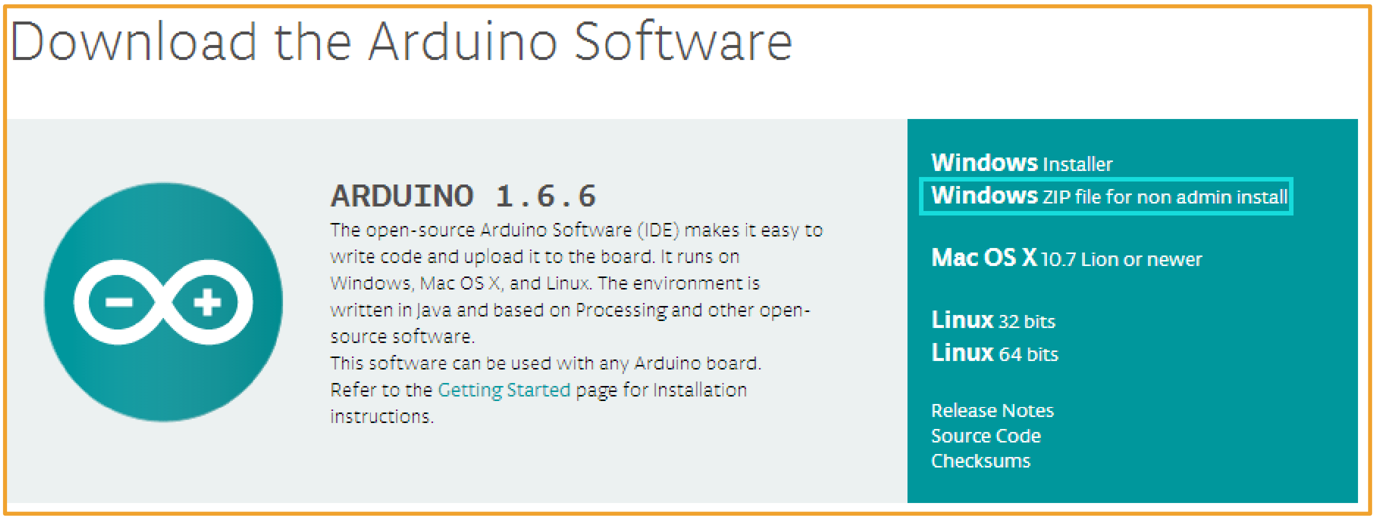
Specifications

- Working voltage: 5V
- Input voltage: 7V~12V DC or USB Power;
- Output voltage: 5V DC output, 3.3V DC output, extern power
- Microcontroller: ATmega328
- Bootloader:Arduino Uno
- Clock rate:16 MHz
- Support USB port agreement and USB charging(no other battery needed)
- Support ISP download
- Digital I/0 port: 14(4 PWM outputs)
- Analog input port: 6
- DC current I/0 port: 40 mA
- DC current 3.3V port: 50mA
- Flash momory: 32KB (ATmega328)(0.5 KB for boot program)
- SRAM:2 KB (ATmega328)
- EEPROM:1 KB (ATmega328)
- Dimensions:75*55*15mm

2.2 Development Software

Download link: https://www.arduino.cc/en/Main/Software

Windows, Linux, Mac are all available for downloading.

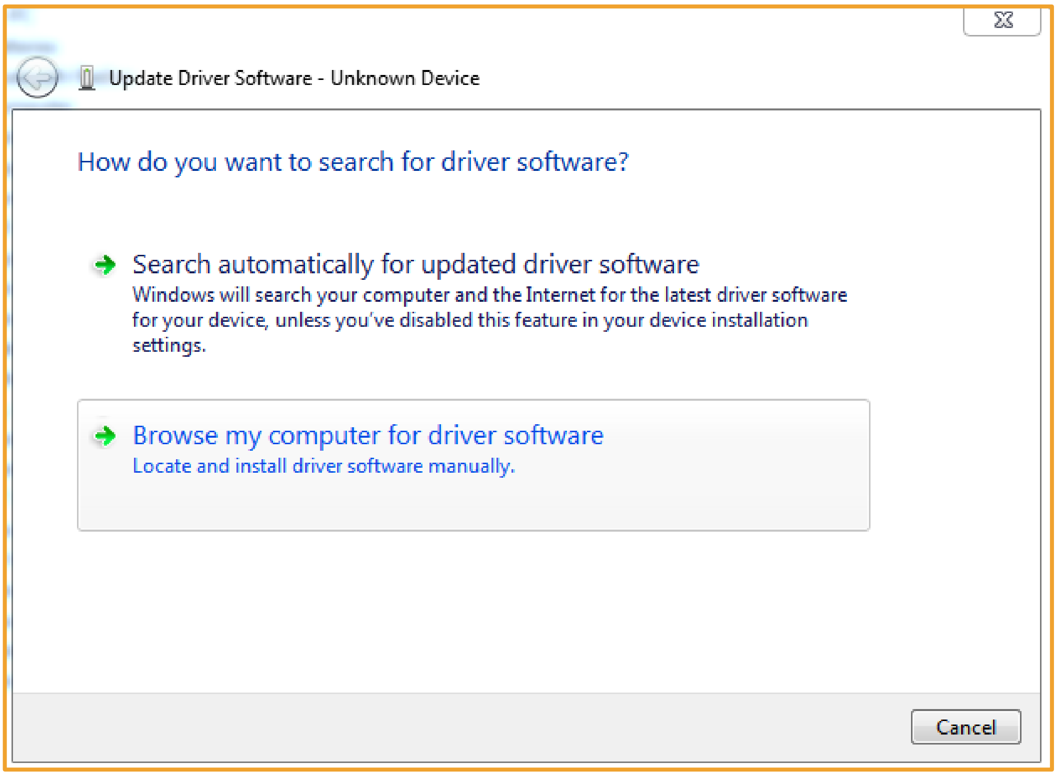


The interface of Arduino IDE is simple and the operation is rather convenient.

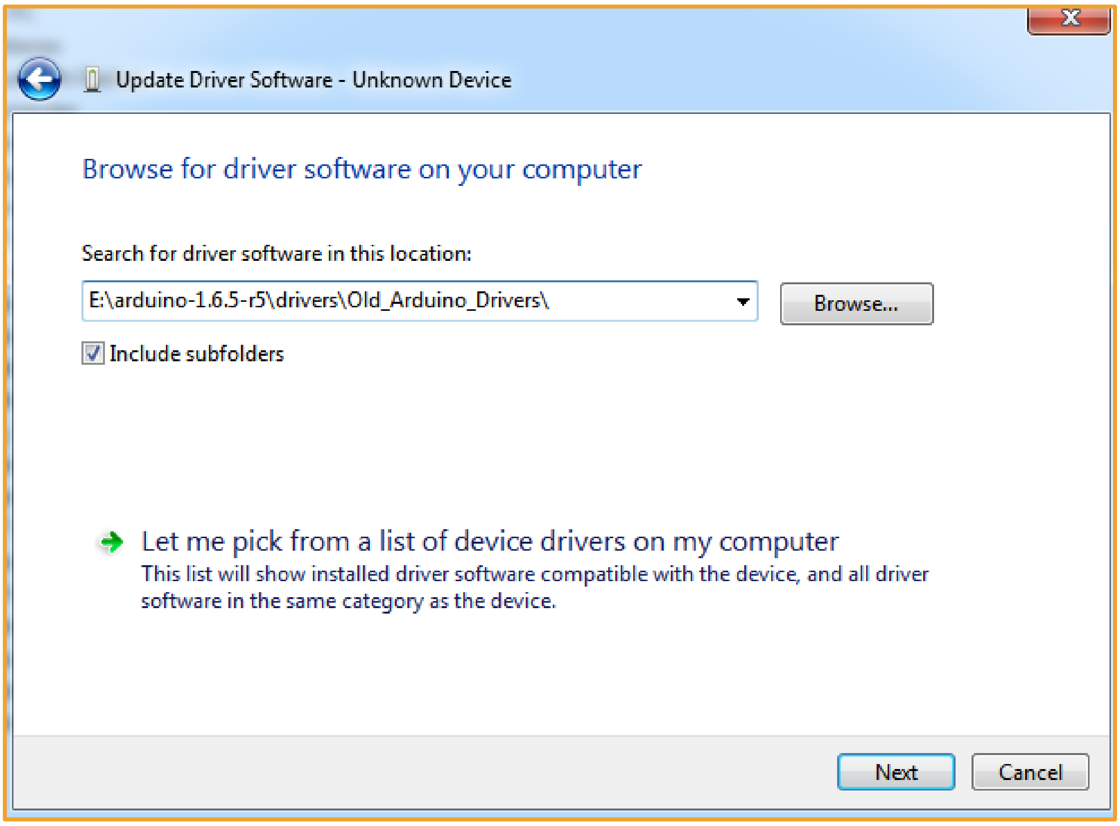
If you want get more,please click https://www.arduino.cc/en/Guide/Environment

Install usb to serial port driver

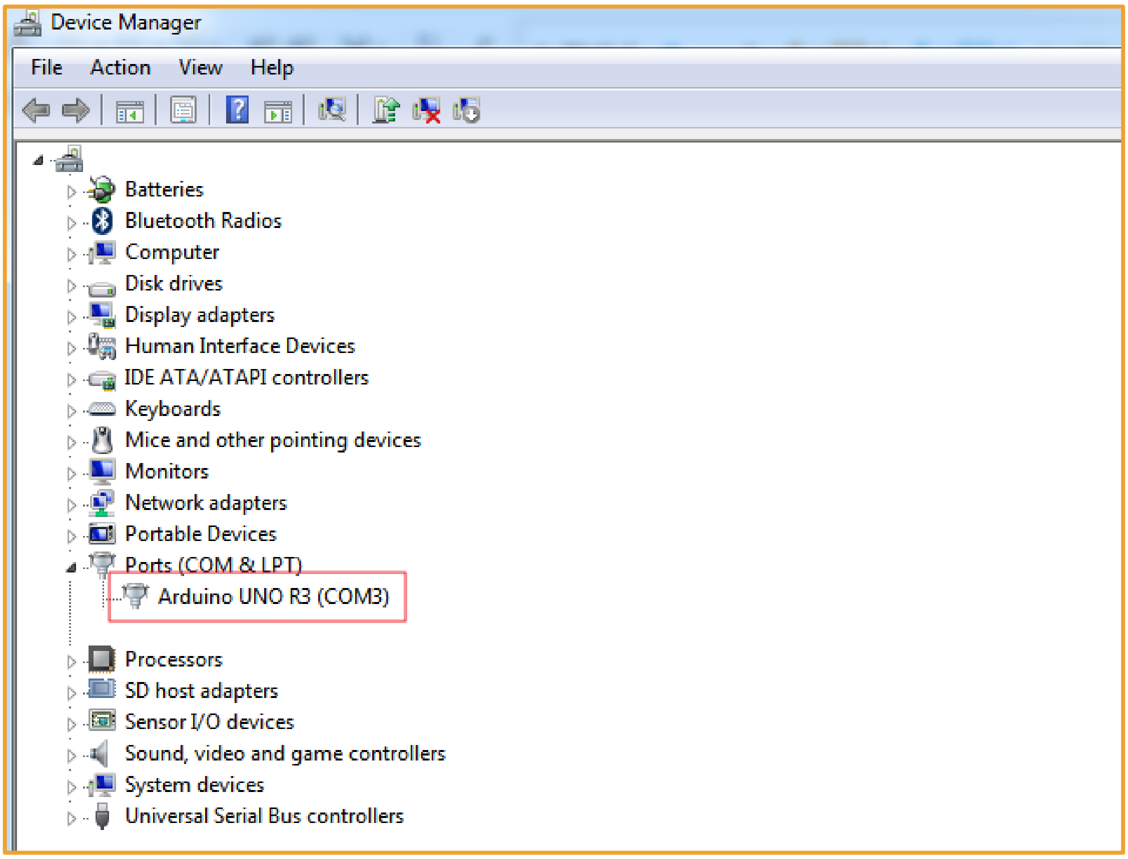
Inserting USB cable will prompt as follows, choosing the specified location to install.



Selecting download Arduino ide file "E:\arduino-1.6.5-r5\drivers\Old_Arduino_Drivers\" Checking the type of USB serial chip on the board, if it is Atmel, then choose the following path; if it is FTDI, you should choose the arduino\drivers\FTDI USB Drivers path.



Clicking the next step, you will be prompted with a successful installation message. Now you can change to equipment management to see Arduino UNO R3.



Chapter 3 Installation

3.1 Tank Assembly

Firstly, we open the box, take out all the components and put it on the table lightly. (Note: There are many devices, be careful when installing to prevent some devices from being lost)

3.1.1 Lower metal backplane copper column and battery installation

Step 1: The installation of the copper plate under the metal bottom plate

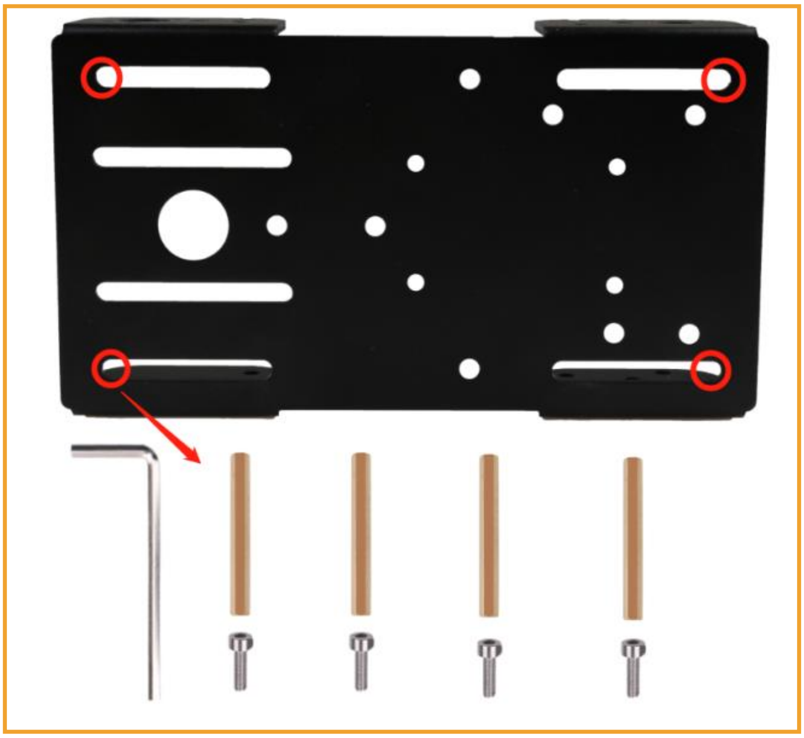


Figure 3.1.1.1 Schematic diagram of copper column installation on lower metal bottom plate

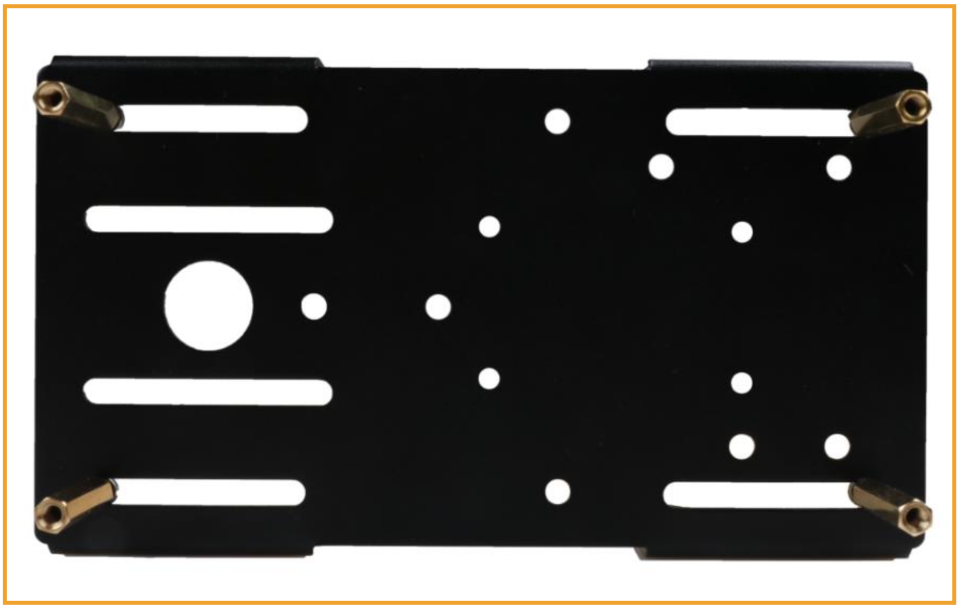
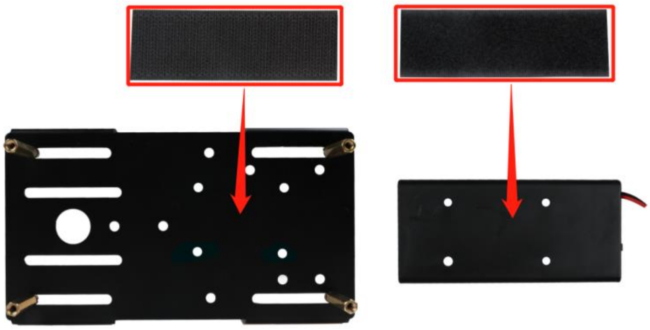


Figure 3.1.1.2 Under metal floor copper column installation effect diagram

Step 2：Battery installation



Note: Attach a piece of Velcro to the back of the battery and attach another to the top of the metal base plate.

Figure 3.1.1.3 Battery box installation diagram

Figure 3.1.1.4 Battery box installation effect diagram

Step 3：Welding DC head

Connect the power cable: Firstly, find the matching two power cables (the same as the wires used by the motor, one red and one black) and connect the two cables to the DC power head. The DC power connector is shown in Figure3.1.1.5The rubber ring marked as "1" can be removed to open the shell and then welding the wires to the +12V and GND, as shown in Fig.3.1.1.6

Figure 3.1.1.5 Power DC Head

Figure 3.1.1.6 Schematic diagram of wire welding

3.1.2 load-bearing wheels、pedrail and motor installation

Step 1：Installation of load-bearing wheels

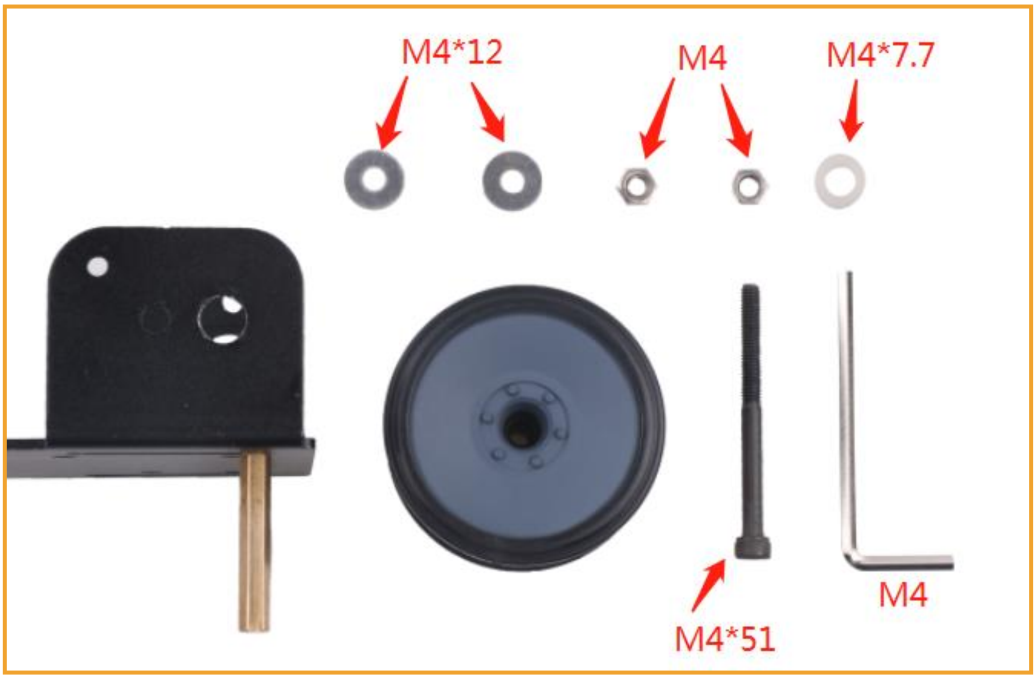


Figure 3.1.2.1 load-bearing wheels installation checklist

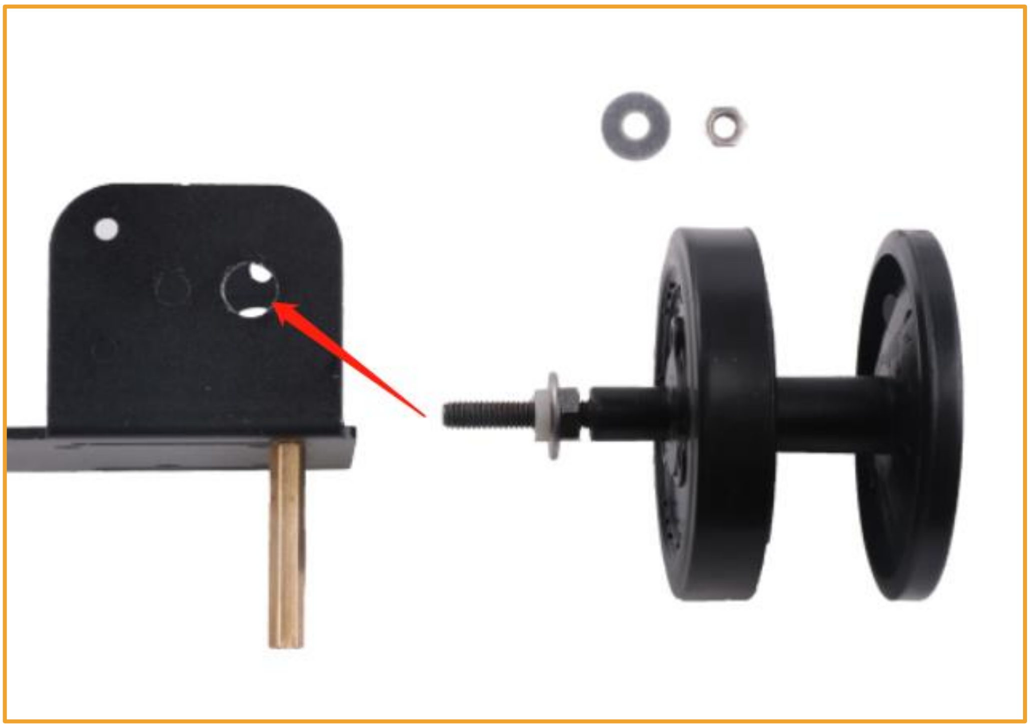
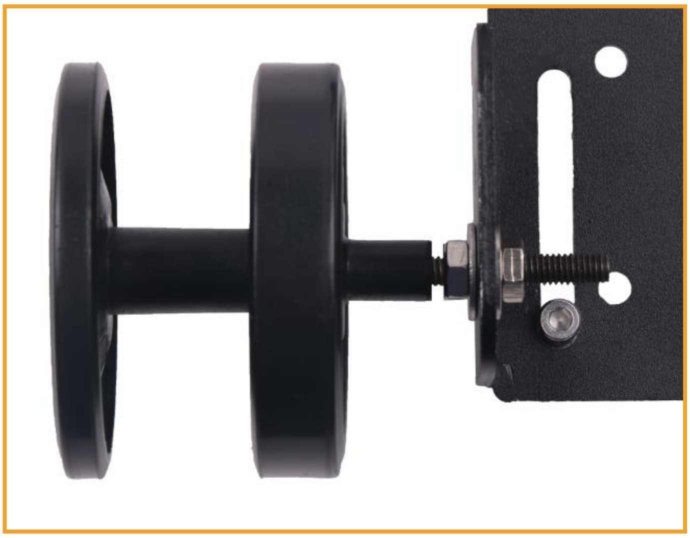


Figure 3.1.2.2 Schematic diagram of load-bearing wheels installation



Note: The load-bearing wheels must be reserved for 1mm from the inner nut.

Figure 3.1.2.3 Effect diagram of load-bearing wheels installation

Step2： Weld motor wire

Figure 3.1.2.4 Schematic diagram of motor wire welding

Figure 3.1.2.5 Effect diagram of motor wire welding

Step 3: Install the motor

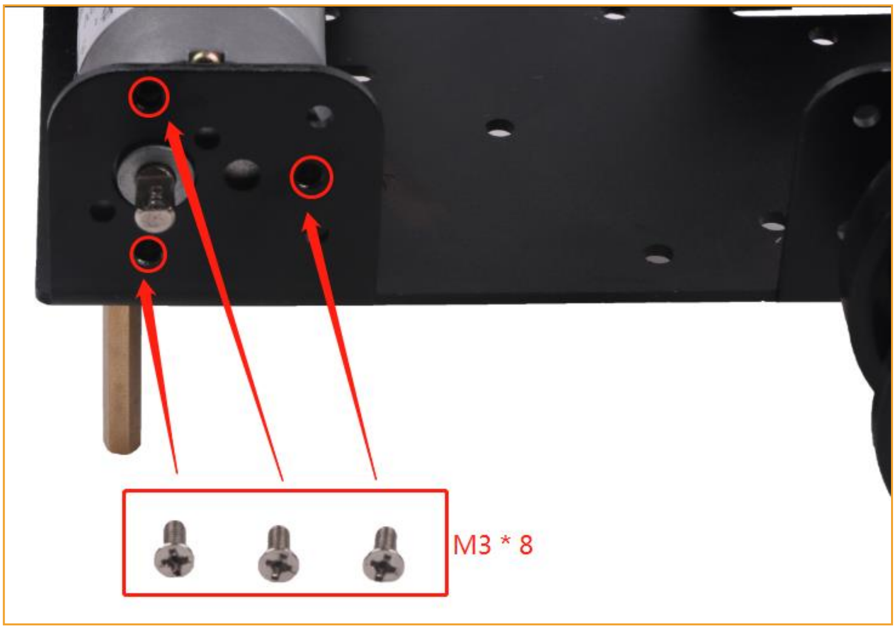


Figure 3.1.2.6 Schematic diagram of motor installation

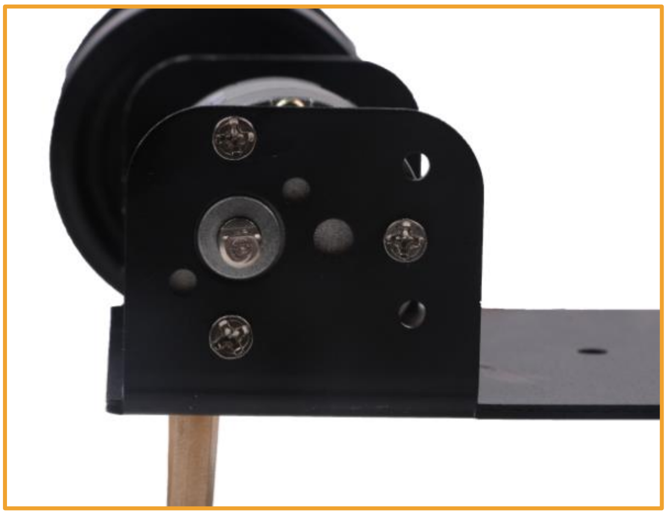


Figure 3.1.2.7 Effect diagram of motor installation

Step 3: Install the coupling

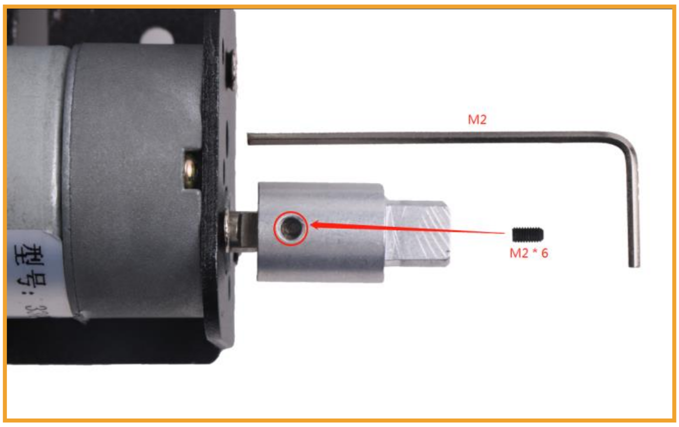
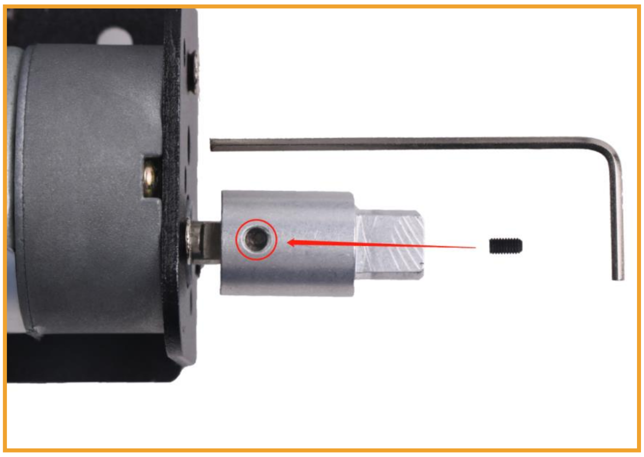


Figure 3.1.2.8 Schematic diagram of coupling installation

Firstly, insert the motor into the circular hole of the coupling, make sure that the hole of the coupling just reaches the flat position of the motor shaft, and screw the black top wire into the hole of the coupling as shown in Figure 3.1.2.9



Note: Do not move the coupling when twisting the black set screw

Figure 3.1.2.9

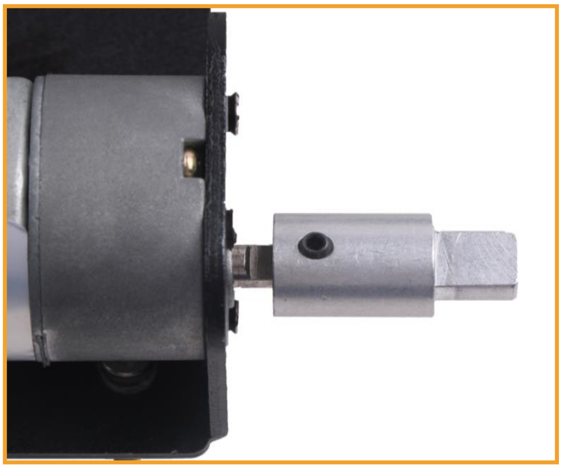


Figure 3.1.2.10 Effect diagram of coupling installation

Step 4: Install the drive wheel

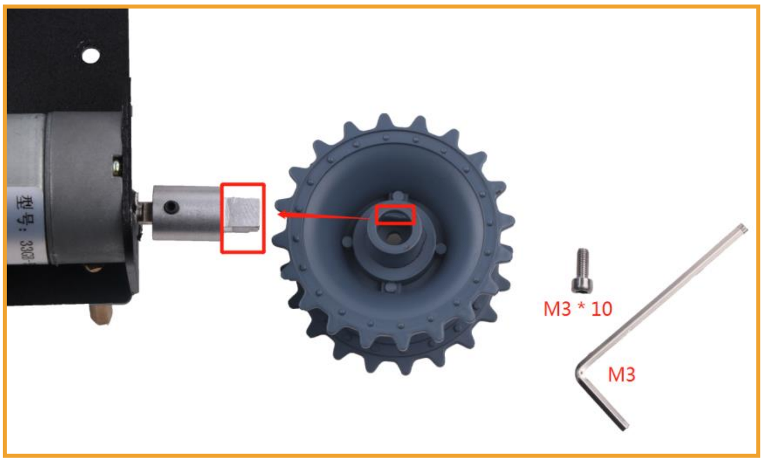


Figure 3.1.2.11 Schematic diagram of drive wheel installation

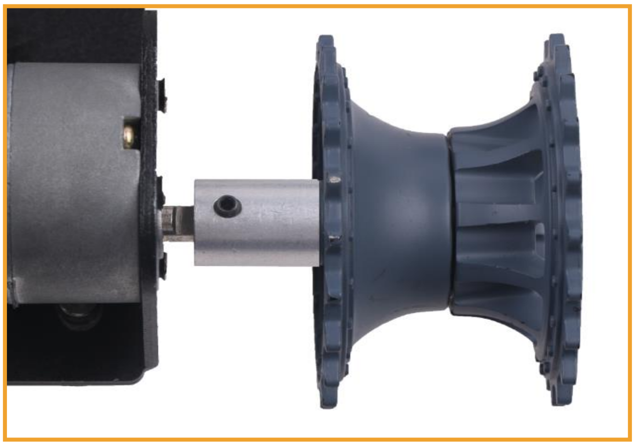


Figure 3.1.2.12 Effect diagram of drive wheel installation

Step 4: Install the pedrail

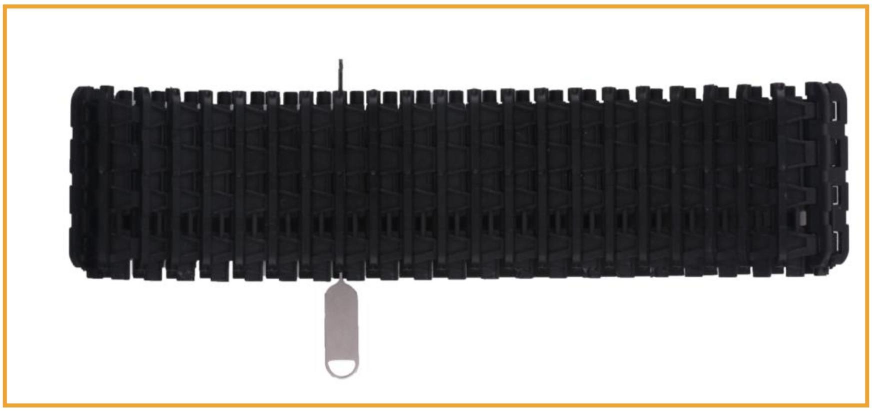
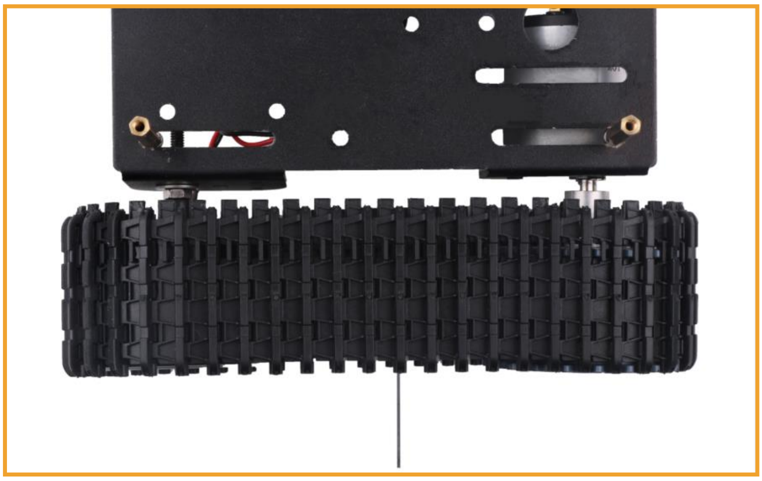


Figure 3.1.2.13 Schematic diagram of the pedrail disassembly



Note: Disassemble the proper size of the pedrai for installation

Figure 3.1.2.14 Schematic diagram of the pedrail installation

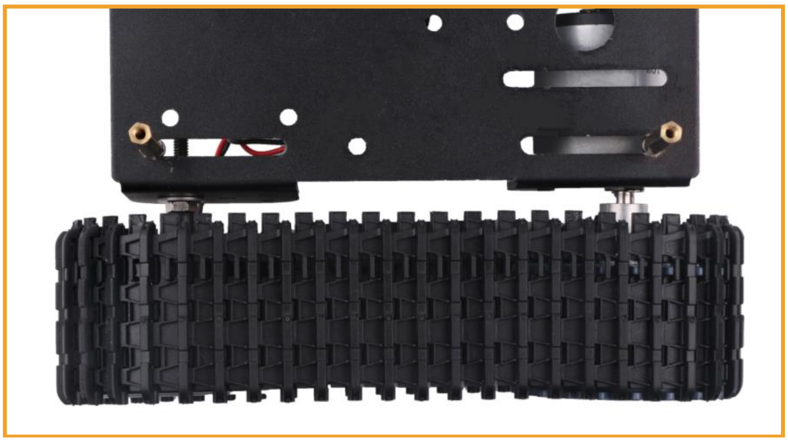


Figure 3.1.2.14 Effect diagram of the pedrail installation

3.1.3 Servo and UNO board installation

Step1: Install UNO board

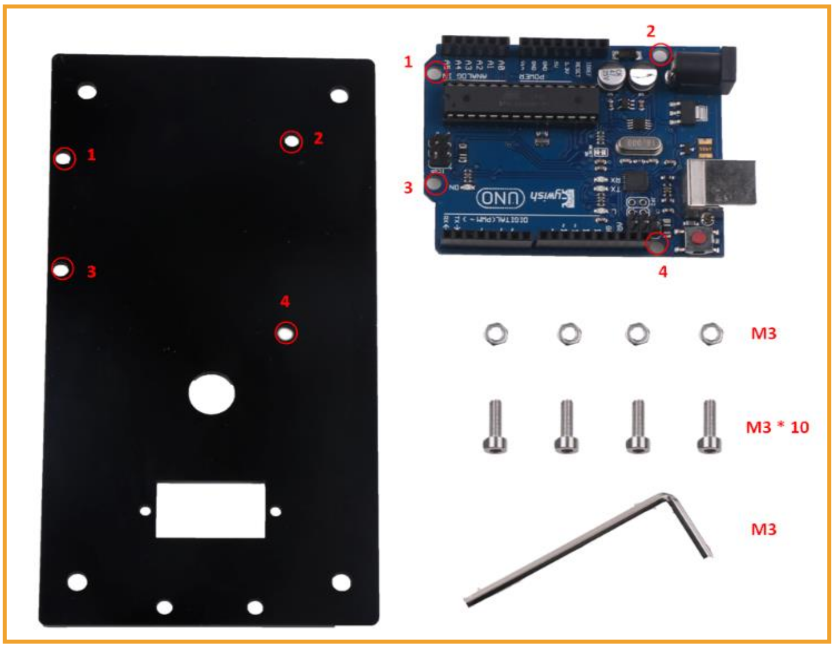


Figure 3.1.3.1 Schematic diagram of UNO board installation

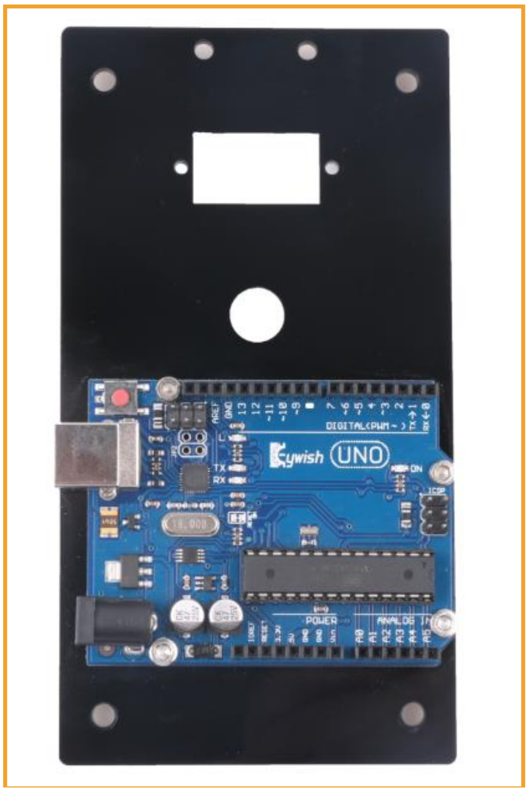


Figure 3.1.3.2 Effect diagram of UNO board installation

Step 2：Install servo

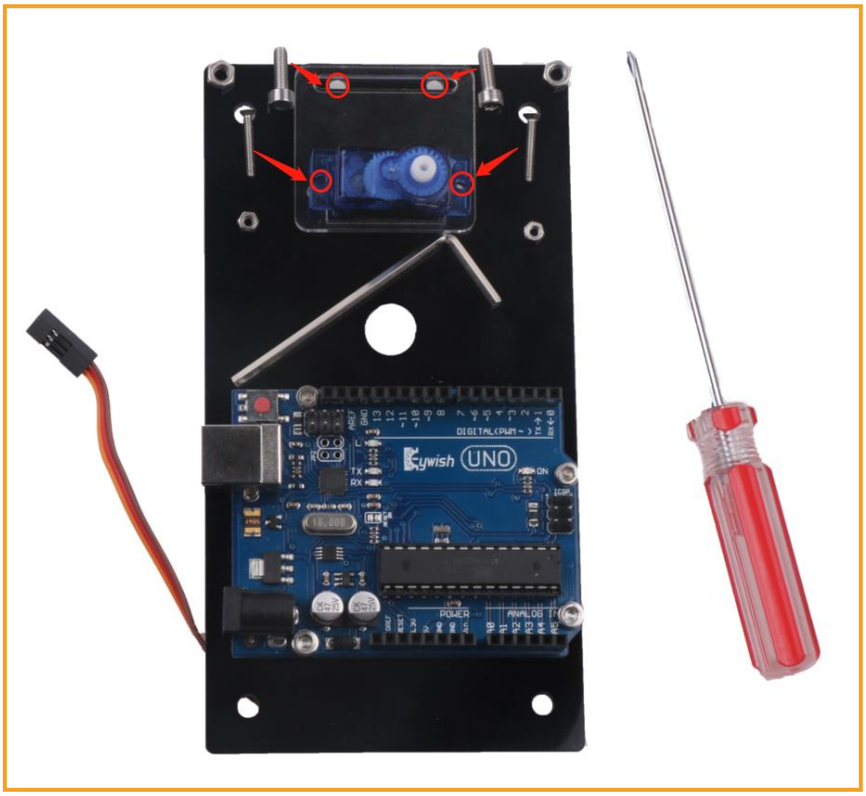


Figure 3.1.3.3 Schematic diagram of servo installation

Figure 3.1.3.4 Effect diagram of servo installation

Step 3: Install the ultrasonic bracket

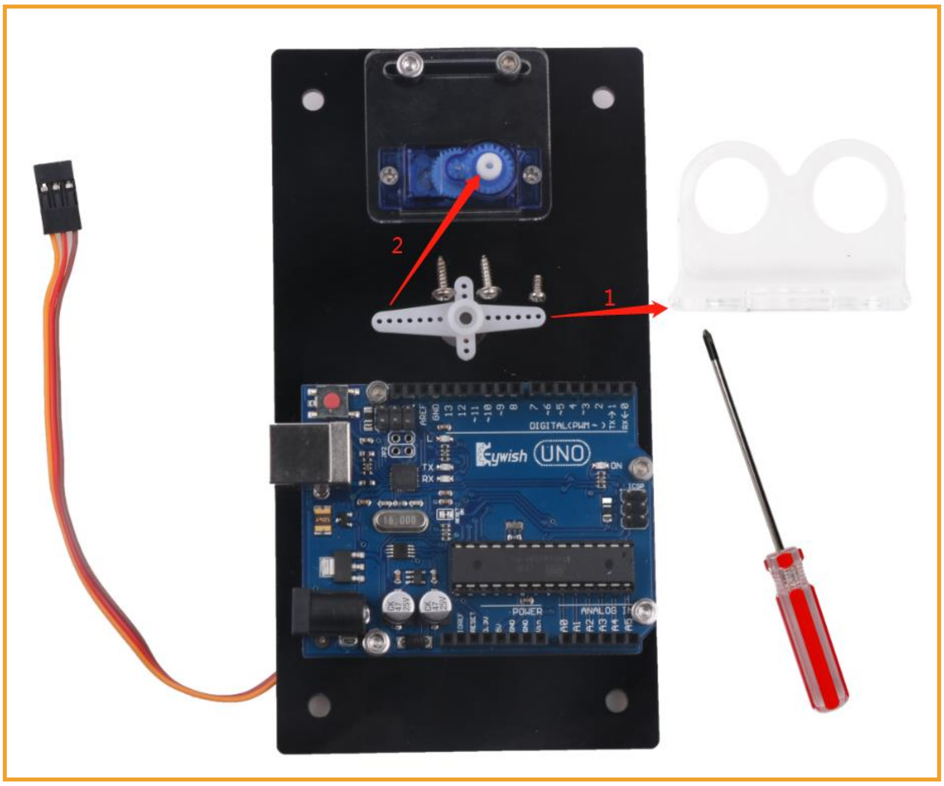


Figure 3.1.3.5 Schematic diagram of ultrasonic bracket installation

Firstly, fix the paddle with the ultrasonic bracket, as shown in Figure 3.1.3.6

Figure 3.1.3.6

Note: The left screw is too long and requires pliers to cut some off.

```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
int pos = 90;    // variable to store the servo position
void setup()
{
myservo.attach(7);  // attaches the servo on pin 13 to the servo object
}
void loop()
{
myservo.write(pos);    // tell servo to go to position in variable 'pos'
delay(15);            // waits 15ms for the servo to reach the position
}
```

Connect the servo to the expansion board and insert the expansion board into the Arduino board. Copy the following program to the compilation environment (**you can also open the CD-ROM in the program Lesson\ModuleDemo\Servo\ServoCorrect\ServoCorrect.ino**). Download the program to the Arduino and ensure that the servo is rotated to 90°. Insert the rudder angle vertically onto the steering shaft and fix it with the screw. As shown in Figure 3.1.3.6, the yellow line is the signal line.

Figure 3.1.3.7 Effect diagram of ultrasonic bracket installation

Step 4: Ultrasonic module and uno expansion board installation

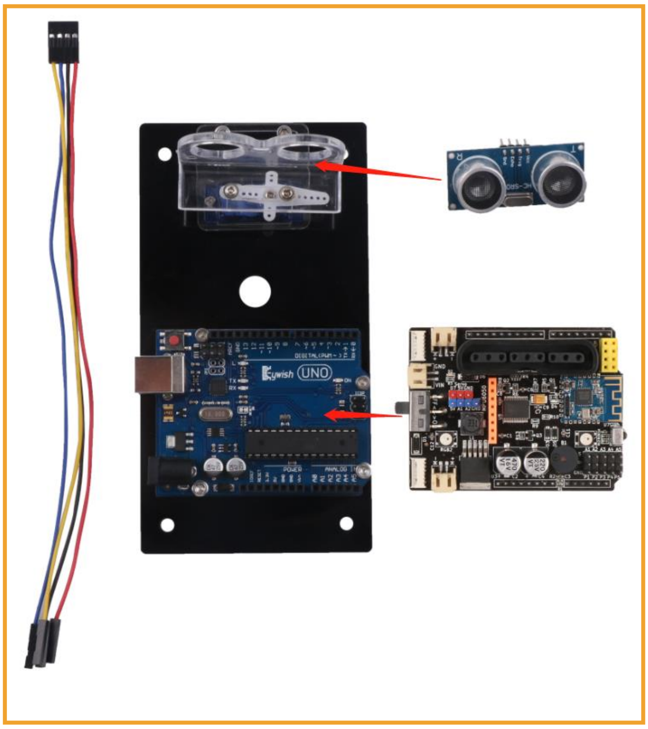


Figure 3.1.3.8 Schematic diagram of ultrasonic module and uno expansion board installation

Figure 3.1.3.9 Effect diagram of ultrasonic module and uno expansion board installation

3.1.4 Upper acrylic floor mounting

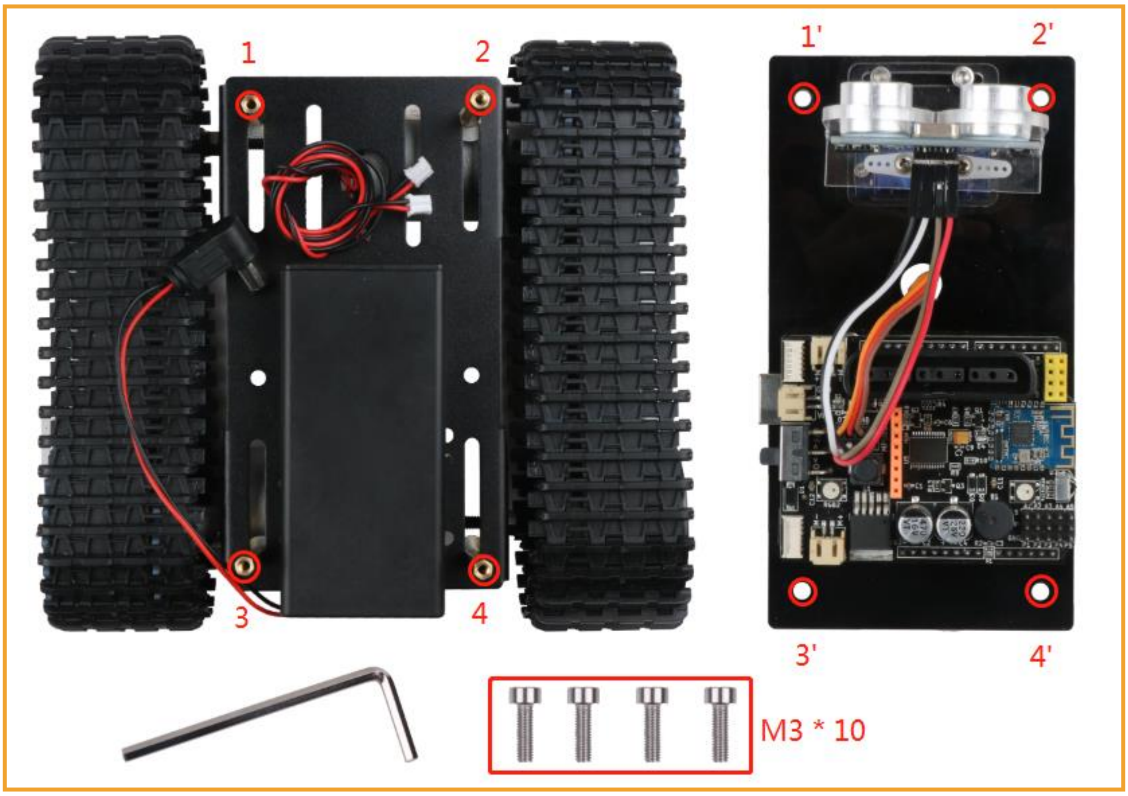


Figure 3.1.4.3 Schematic diagram of upper acrylic baseboard installation

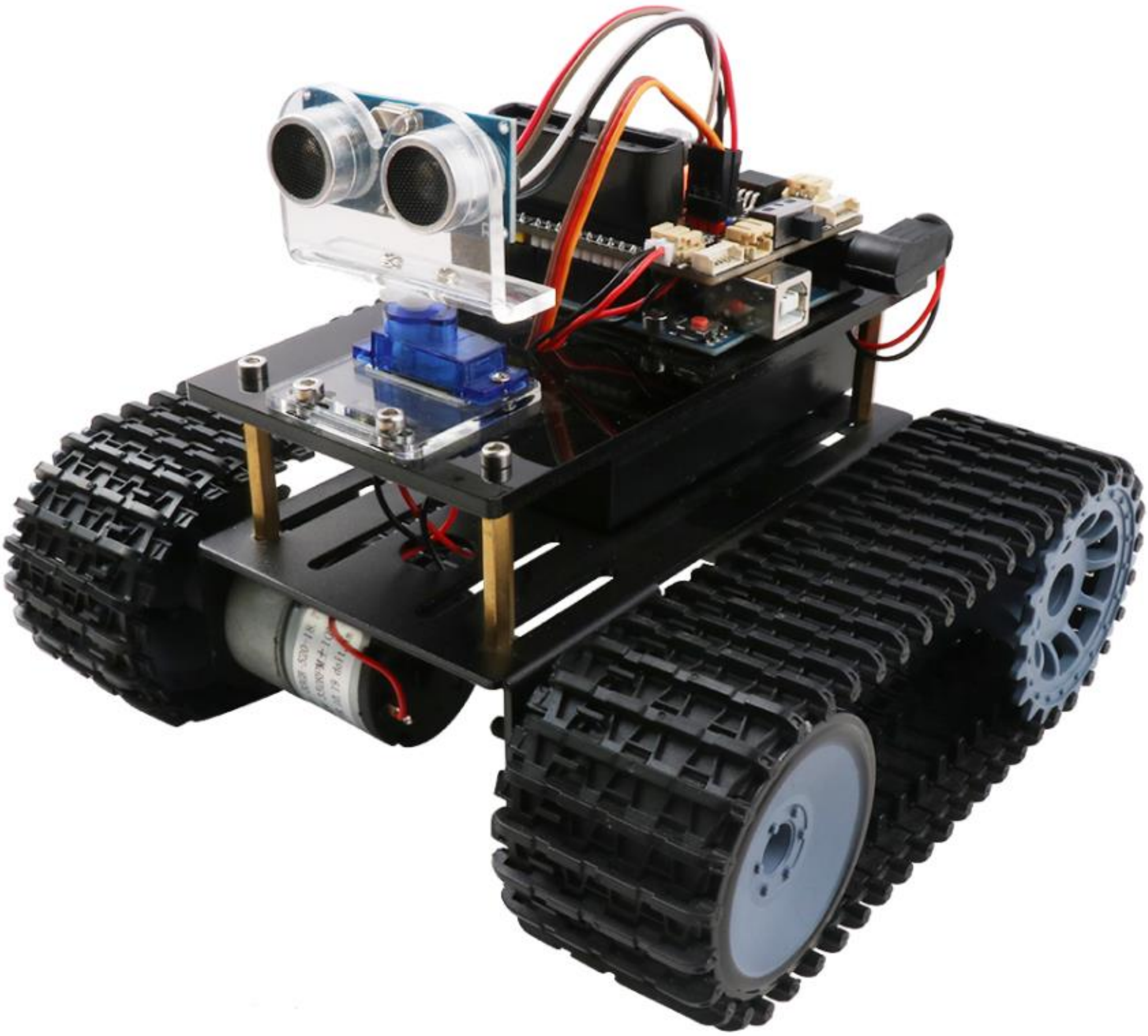
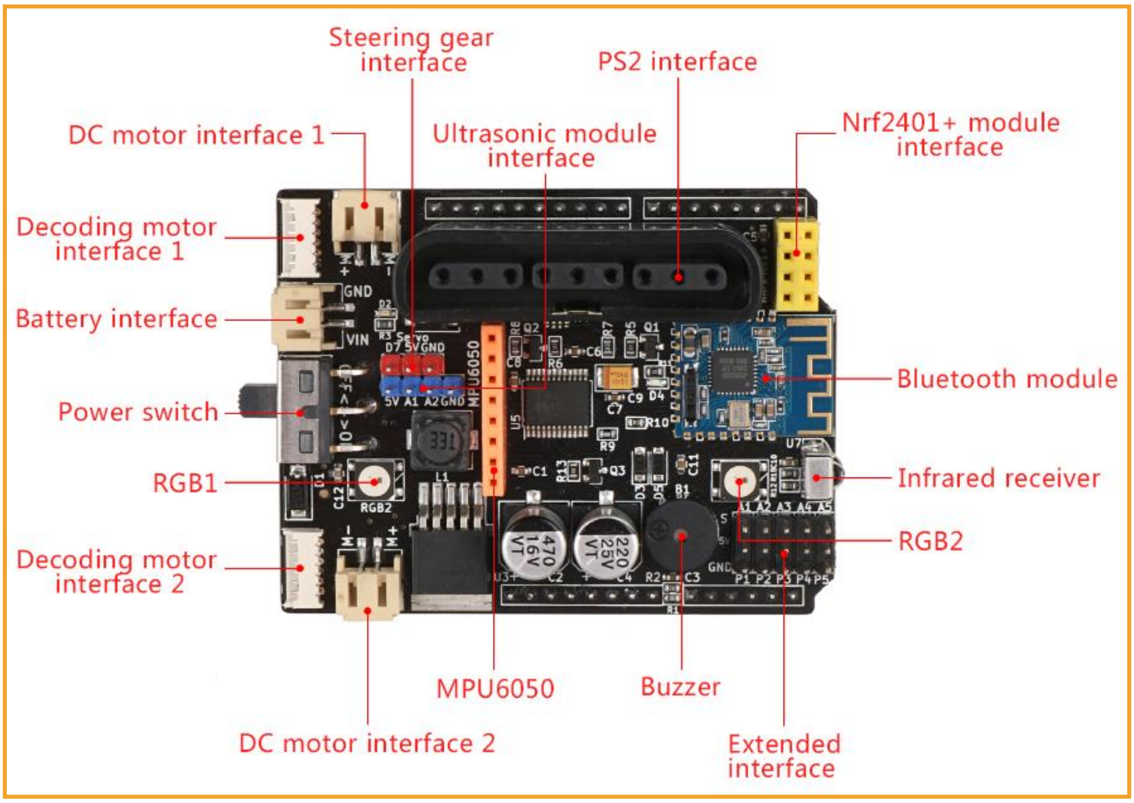


Figure 3.1.4.4 Effect diagram of upper acrylic baseboard installation

So far, the basic assembly of the tank has been completed. We believe you have some basic knowledge of your car’s structure, function and some modules through a short period of time, then you can achieve the corresponding functions only by downloading the program to the development board, each function has a corresponding program in CD, so please enjoy playing. However, if you can read the program and write your own program, there will be more fun, now let's go to the software section!

3.2 Experiment

TB6612FNG motor drive board Frame diagram



Picture 3.2.1 TB6612 motor drive board Frame diagram

TB6612FNG is a motor driver extension board special for Arduino UNO R3, with Bluetooth 4.0 Moudle,MPU6050 Moudle Interface,IR,2 x6 pins encoder deceleration interfaces， 2 x 2Pins PWM DC motor Interfaces， 1 x servo Interface,5 extension Pins,2 x RGB LED ,Buzzer.The Shield will help you handle motor diver and control issues ,give a easier and more intelligent solution when you build a Arduino Car， Robot，Balance car etc;

Features:

- 2 x 2pins PWM DC motor driver interface
- 2x 6pins motor encoder deceleration interface
- 1x 3pins servo interface
- 1x 4pins Needle ultrasonic module interface
- 2x RGB LED light
- 1x mpu6050 module interface
- 1x On Board passive buzzer
- 1x On Board integrated infrared receiver

- 1x PS2 Receiver interface
- 1x NRF24L01 Interface
- 5x Extended interface A1 A2 A3 A4 A5

Specification:

Connect directly to Arduino UNO R3 and powed through UNO motherboard,Working Voltage:6-20V Output current: 1.2A single channel continuous drive current start / peak current: 2A (continuous pulse) / 3.2A (single pulse)

3.2.1 RGB WS2812B Experiment

3.2.1.1 RGB WS2812B Description

The WS2812B is 3 output channels special for LED driver circuit. Its internal includes intelligent digital port data latch and signal reshaping amplification drive circuit. Also includes a precision internal oscillator and a 12V voltage programmable constant current output drive. In the purpose of reduce power supply ripple, the 3 output channels designed to delay turn-on function.

Unlike the traditional RGB, WS281B **is integrated with a WS281B LED drive control special chip, which requires a single signal line to control a LED lamp or multiple LED modules. Features as below:**

- Output port compression 12V.
- Built-in voltage-regulator tube, only a resistance needed to add to IC VDD feet when under 24V power supply.
- 256 Gray-scale adjustable and scan frequency is more than 2KHz.
- Built in signal reshaping circuit, to ensure waveform distortion do not accumulate after wave reshaping to the next driver
- Built-in electrify reset circuit and power-down reset circuit.
- Cascading port transmission signal by single line
- Any two point the distance less than 5 Meters transmission signal without any increase circuit.
- When the refresh rate is 30fps, the cascade number is at least 1024 pixels.
- Send data at speed of 800Kbps.

For more parameters of WS2812,please check the file in CD:"Panther-Tank \Document\WS281B.pdf"

3.2.1.2 RGB WS281B working principle

IC uses single NZR communication mode. After the chip power-on reset, the DIN port receive data from controlle,the first IC collect initial 24bit data then sent to the internal data latch,the other data which reshaping by the internal signal reshaping amplification circuit sent to the next cascade IC through the DO port. After transmission for each chip, the signal to reduce 24bit. IC adopt auto reshaping transmit technology,making the chip cascade number is not limited the signal transmission,only depend on the speed of signal transmission.

The data latch of IC depend on the received 24bit data produce different duty ratio signal at OUTR,OUTG,OUTB port.All chip synchronous send the received data to each segment when the DIN port input a reset signal. It will receive new data again after the reset signal finished. Before a new reset signal received, the control signal of OUTR ,OUTG, OUTB port unchanged. IC sent PWM data that received justly to OUTR, OUTG, OUTB port, after receive a low voltage reset signal the time retain over 280μs.

Pin function and Pin configuration as picture 3.2.1.1

NO	Symbol	Function description
1	VDD	Power supply voltage
2	OUT	Data signal cascade output
3	IN	Data signal input
4	VSS	Ground

WS281B Pin function

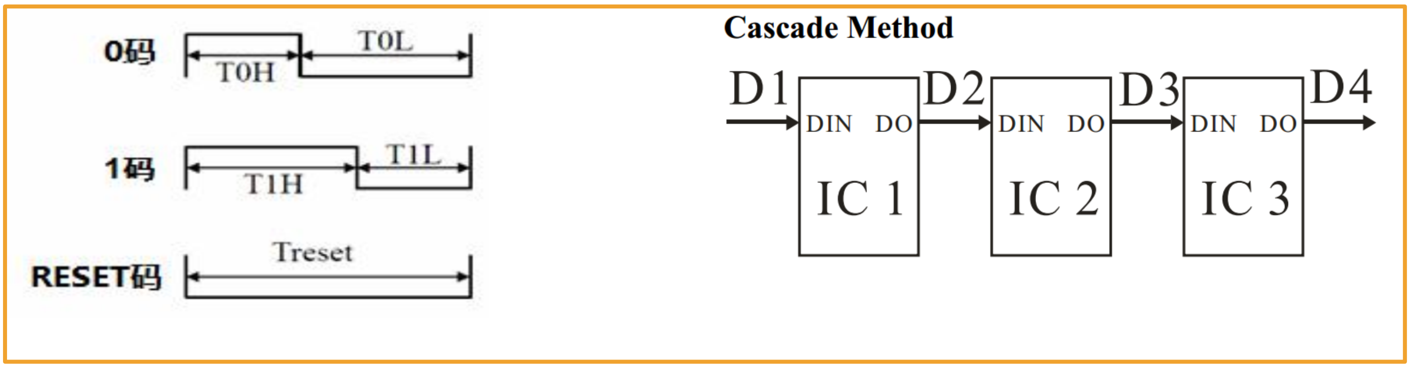
Picture 3.2.1.1 WS281B Pin configuration

3.2.1.3 WS281B drive principle

WS281B The low level is represented by T0, consists of a 0.5μs high level and 2μs low level. The high level is represented by T1, consists of a 2μs high level 0.5μs low level. When low level last more than 50μs there will be a reset signal.

T0H	0 code, high voltage time	0.5μs
T1H	1 code, high voltage time	2μs
T0L	0 code, low voltage time	2μs
T1L	1 code, low voltage time	0.5μs
RES	Frame unit, low voltage time	>50μs

Sequence Chart



Picture 3.2.1.2 Waveform sequence diagram

Composition of 24bit Data:

R7	R6	R5	R4	R3	R2	R1	R0	G7	G6	G5	G4	G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0

Note: Data transmit in order of GRB, high bit data at first.

3.2.1.4 RGB WS281B Test Program

Open the material,path: **“Lesson\ModuleDemo\RGB\RGB_test2\RGB_test2.ino**”

The purpose of this experiment is to show the double breath lamp effect that the RGB1 is gradually brightened, and the RGB2 gradually extinguished separately.Program as follows.

```
#include "Adafruit_NeoPixel.h"
#define RGB_PIN A3
#define MAX_LED 2
int RGB1_val = 0;
int RGB2_val = 255;
bool trig_flag = true;
Adafruit_NeoPixel strip = Adafruit_NeoPixel(MAX_LED, RGB_PIN, NEO_GRB + NEO_KHZ800);
void setup()
{
```

```
    strip.begin();  //default close all led
    strip.show();
}
void loop()
{
    // write 0 ~ 255  ~ 0 value to RGB1
    strip.setPixelColor(0, strip.Color(RGB1_val, 0, 0));
    // write 255 ~ 0 ~ 255 value to RGB2
    strip.setPixelColor(1, strip.Color(0, 0, RGB2_val));
    strip.show();
    if (trig_flag == 1) {
        RGB1_val--;
        RGB2_val++;
    } else {
        RGB1_val++;
        RGB2_val--;
    }
    if (RGB1_val >= 255 || RGB1_val <= 0 ) {
        trig_flag = !trig_flag;
    }
    delay(30);
 }
```

Adafruit_NeoPixel(MAX_LED, RGB_PIN, NEO_RGB + NEO_KHZ800);

The first parameter sets the number of RGB display, the second parameter sets the GPIO port that RGB use, and the third parameter sets the RGB data transmission speed mode.

NEO_KHZ800 800 KHz bitstream (most NeoPixel products w/WS2812 LEDs)

NEO_KHZ400 400 KHz (classic 'v1' (not v2) FLORA pixels, WS281B drivers)

NEO_GRB Pixels are wired for GRB bitstream (most NeoPixel products)

NEO_RGB Pixels are wired for RGB bitstream (v1 FLORA pixels, not v2)

strip.Color(0, 0, RGB2_val)

The three parameters are Red, Green, Blue, three color value components, the range 0~255, synthesis a 24bit value.

strip.setPixelColor(index, strip.Color(R, G, B))

The index parameter represents the serial number of the RGB to light.

strip.setPixelColor(0, strip.Color(RGB1_val, 0, 0))

Light RGB1 to red color.

strip.setPixelColor(1, strip.Color(0, 0, RGB2_val))

Light RGB2 to blue color.

3.2.2 Passive Buzzer

3.2.2.1 Description

The buzzer is an integrated electronic alarm device that uses DC voltage to power electronic products for sound devices.Buzzers are mainly divided into active buzzer and passive buzzer.The main difference between the two type device is as follow :

The ideal signal for active buzzer operation is direct current,usually labeled at VDC, VDD, etc. There is a simple oscillating circuit inside the buzzer. As long as it is energized, it can motivate the molybdenum plate to vibrate. But some active buzzer can work under specific AC signal, while it has high requires of AC signal voltage and frequency, this situation is very rarely .

The passive buzzer does not include internal oscillation circuit, the working signal is a certain frequency of pulse signal .If we give DC signal, the passive buzzer will not work, because the magnetic circuit is constant, the molybdenum sheet cannot vibrate. They are just as shown as follow picture 3.2.2.1 :

Active Buzzer

Passive Buzzer

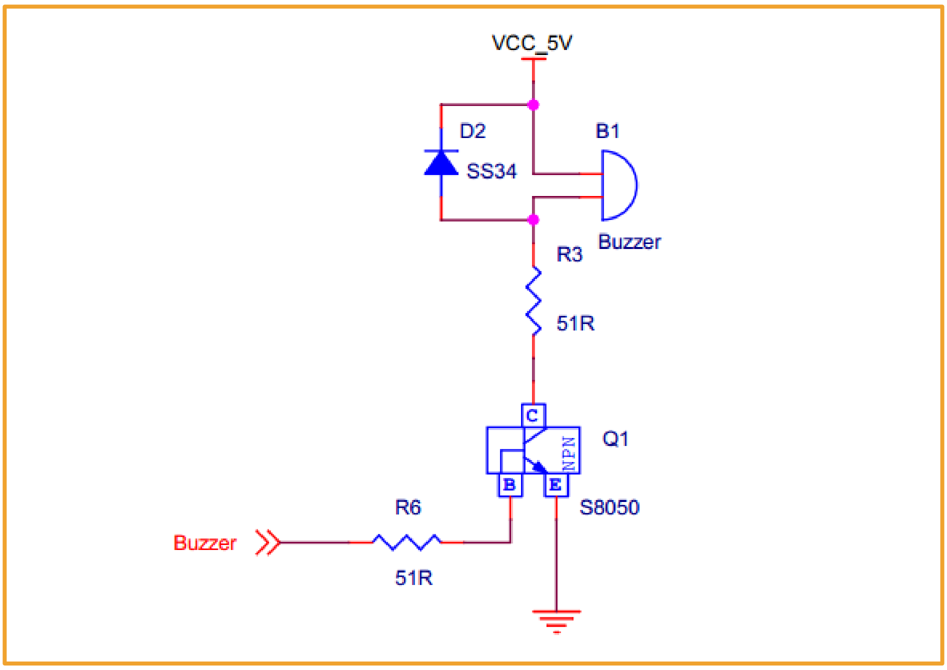
Picturec 3.2.2.1 Ative Buzzer and Passive Buzzer

3.2.2.2 Buzzer working principle

The passive buzzer generates music mainly through the I/O port of the single-chip microcomputer to output different pulse signals of different levels to control the buzzer pronunciation.

For example,if Arduino use12MHzs crystal oscillator, to produce middle tune “Re” sound, it needs 587Hzs audio pulse frequency output .The audio signal pulse cycle is T=1/587=1703.5775us, half cycle time is 852us,the pulse timer needs always count at =852us/1us=852,when it count at 852,the I/O port will reverse the direction, then it get the “Re” sounds in C major scale.

In addition to that, the passive buzzer sound principle is the current through the electromagnetic coil, making the electromagnetic coil generated magnetic field to drive the vibration film audible. Therefore,a certain amount of current is required to drive it, while the ARDUINOI/O pin outputs a lower voltage. Arduino output level can hardly drive the buzzer, so it needs to add an amplifier circuit. And transistor S8050 is used here as an amplification circuit. As shown in picture 3.2.2.2 is the onboard buzzer schematic diagram:



Picture 3.2.2.2 Schematic diagram of onboard buzzer

3.2.2.3 Experiment 1:Alarm Test

Experimental purposes:

Make the buzzer simulate the alarm sound

Experimental principle:

The sound starts with the frequency increasing from 200HZ to 800HZ, then stops for a period of time from 800HZ to 200HZ, and loops experimental code location.

"Lesson\ModuleDemo\Buzzer\AlarmSound\AlarmSound.ino"

```
void setup()
{
   pinMode(9,OUTPUT);
}
void loop()
{
   for(int i = 200; i <= 800; i++)   // 200HZ ~ 800HZ
   {
      tone(9,i);
   }
   delay(1000);                  //Max Frequency hold 1s
   for(int i= 800; i >= 200; i--)   // 800HZ ~ 200HZ
   {
      tone(9,i);
      delay(10);
   }
}
```

Firstly,we use a simple procedure to understand how to use the buzzer, and its sound principle. And to drive a buzzer like singing sound,we need make the buzzer issued frequency and duration of the different sound . Cycle is equal to the reciprocal of the frequency, so you can know the time by frequency, and then by calling the delay function or timer to achieve.Similarly the sound duration can also be achieved through the delay function. So the key factor to make the buzzer sing is to know how much time to prolong! Play music with Arduino, you just need to understand the two concepts of "tone" and "beat" .

The tone means the frequency at which a note should be sung.

The beat means how long a note should be sung.

The commonly used method is "look-up table method", this method is complex where you have to find the corresponding frequency of each note (according to the note, the frequency comparison), and then according to the formula converted to the corresponding time (take half cycle), and then through the delay function implementation. Finally by programming to achieve.

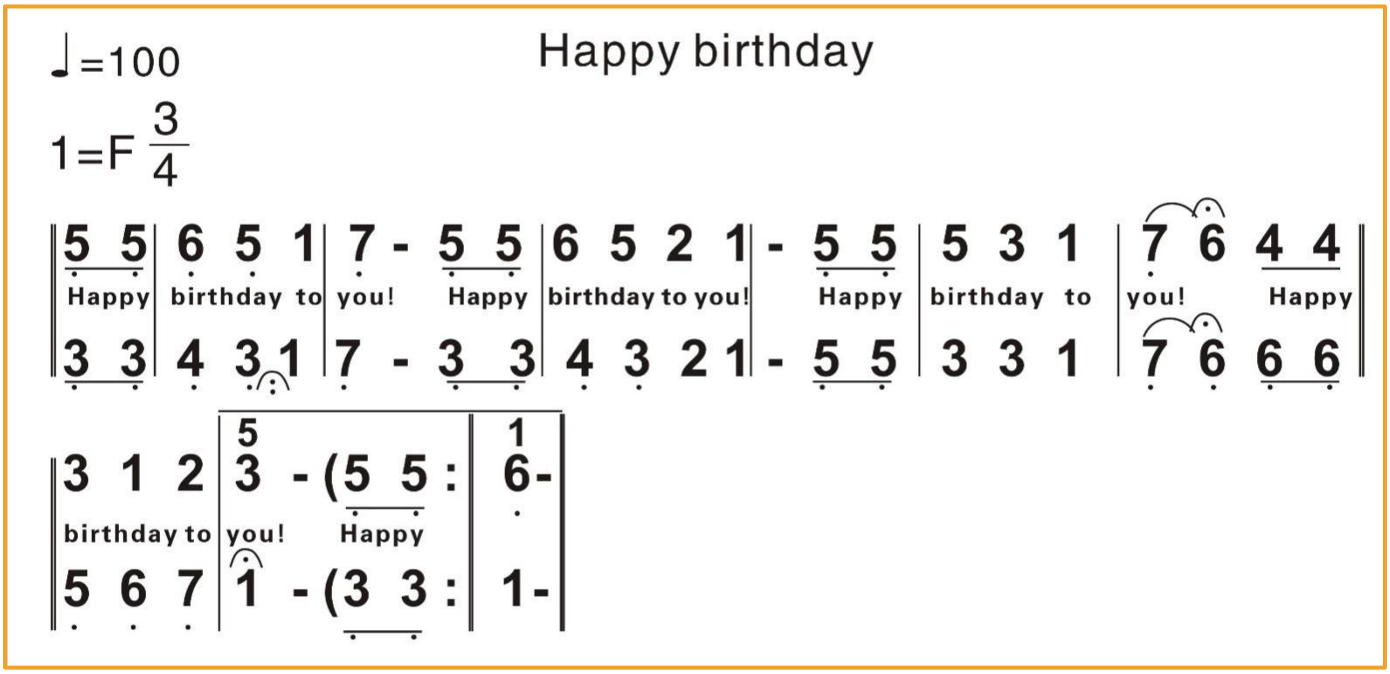
The whole process is like this:

Firstly, according to the score of Happy Birthday song, convert each tone to the corresponding frequency.

For example: picture 3.2.2.3 is the note frequency conversion table, picture 3.2.2.4 is the Happy Birthday song score.

	Musical notes	Corresponding frequency (Hz)	Half cycle(us)
Bass	1	261.63	1911.13
	1.5	277.18	1803.86
	2	293.66	1702.62
	2.5	311.13	1607.06
	3	329.63	1516.86
	4	349.23	1431.73
	4.5	369.99	1351.37
	5	392.00	1275.53
	5.5	415.30	1203.94
	6	440.00	1136.36
	6.5	446.16	1120.66
	7	493.88	1012.38
Alto	1	523.25	955.56
	1.5	554.37	901.93
	2	587.33	851.31
	2.5	622.25	803.53
	3	659.26	758.43
	4	698.46	715.86
	4.5	739.99	675.69
	5	783.99	637.76
	5.5	830.61	601.97
	6	880.00	568.18
	6.5	932.33	536.29
	7	987.77	506.19
Treble	1	1046.50	477.78
	1.5	1108.73	450.97
	2	1174.66	425.66
	2.5	1244.51	401.77
	3	1318.51	379.22
	4	1396.91	357.93
	4.5	1479.98	337.84
	5	1567.98	318.88
	5.5	1661.22	300.98
	6	1760.00	284.09
	6.5	1864.66	268.15
	7	1975.53	253.10

Picture 3.2.2.3 The note frequency conversion table



Picture 3.2.2.4 The score of Happy birthday song

这 Firstly , let’s learn some knowledge about music score and look at the above music score ,the bass is the one which with point under the number, the normal tone without any point .The treble is the one which with point above the number. The bass of tone 5 is 4.5,the treble is 5.5. Other notes are the corresponding truth. There is a “1=F” on the upper left of music score, while the general music score is C ,it is “1=C”.Note, the 1234567(do ,re,mi,fa,so,la,xi,duo) relative is CDEFGAB,not ABCDEFG.So, if the rule is F ,that means 2 is G tone,3 is A tone,……7 is E tone. So, in the situation, the bass 5 corresponds to bass 1.5,the tone need to move to the right or left. If you still don't understand, look at the following:

1 originally corresponding should be c,4 originally should correspond to f.

Then now 1 corresponds to F, which corresponds to 4, then 1.5 corresponds to 4.5, 2 corresponds to 5.

So, bass 5 is actually 4.5, so half cycle is 1803μs.

As to it is based on half-cycle calculations, because the single-chip microcomputer makes a sound by looping resetting the port connected to the buzzer, so it is a half-cycle. Because our product is passive buzzer, the active buzzer is full cycle.

Then according to the above reason, converse the tone one by one, achieve it by delay function. Because the frequency of the conversion of each note is different, you need to using multiple delay functions to achieve accurate tone frequencies one by one. But this is too complicated, and the microcontroller itself is not specifically to sing. The delay function has almost the same frequency in order to adapt to each tone, you need to calculate it by yourself, and different songs have different values, so this is the more troublesome issue.

After we know the frequency of the pitch, the next step is to control the playing time of the notes. Each note plays for a certain amount of time so that it can be a beautiful piece of music. The rhythm of notes is divided into one beat, half beat, 1/4 beat, and 1/8 beat. We stipulate that the time of a clap of notes is 1, half a beat for the 0.5;1/4 Pat for the 0.25;1/8 0.125 ...

Firstly,ordinary notes occupy for 1 shots.

Secondly, underlined notes indicate 0.5 beats; two underlines are quarter beats (0.25)

Thirdly, the notes which followed by a point, which means more 0.5 beats, that is 1+0.5.

Fourthly, the notes followed by a "-", which means more than one beat, that is, 1 +1.

So we can give this beat to each note and play it out. As for the conversion of beats to frequency, it also has corresponding list, just follow table two:

Music beat	1/4 beat delay time	Music	1/8 beat delay time
4/4	125ms	4/4	62ms
3/4	187ms	3/4	94ms
2/4	250ms	2/4	125ms

Table 2: Beat and frequency correspondence table

It is also achieved through the delay function,of course there will be errors. The idea of programming is very simple, firstly convert the note frequency and the time you want to sing into the two arrays. Then in the main programming, through the delay function to reach the corresponding frequency . sing it over, stop for a while, and then sing it, all the conversion is complete, we get the following frequency (Table 3) and beat:

Do 262	Re 294	Mi 330	Fa 349	Sol 392	La 440	Si 494	Do_h 523
Si_h 988	Mi_h 659		La_h 880	Sol_h 784	Fa_h698	Re_h 587	

Table 3: Happy Birthday Song beat table

According to the music score, we can get the frequency of the birthday song:

Sol, Sol, La, Sol, Do_h, Si, Sol, Sol, La, Sol, Re_h, Do_h, Sol, Sol, Sol_h, Mi_h, Do_h, Si, La, Fa_h, Fa_h , Mi_h, Do_h, Re_h, Do_hfloat

The beat is as follows:

0. 5, 0. 5, 1, 1, 1, 1+1, 0. 5, 0. 5, 1, 1, 1, 1+1, 0. 5, 0. 5, 1, 1, 1, 1, 1, 0. 5, 0. 5, 1, 1, 1, 1+1,

Add beats and frequency to the program and download it to Arduino to play.

Happy Birthday music score beat, view table two rhythm and frequency corresponding table 1 beats time is 187*4 = 748ms

Note: The procedure is shown in the: "Lesson\Advanced Experiment\Happy_Birthday \Happy_Birthday.ino"

```
#define Do 262
#define Re 294
#define Mi 330
#define Fa 349
```

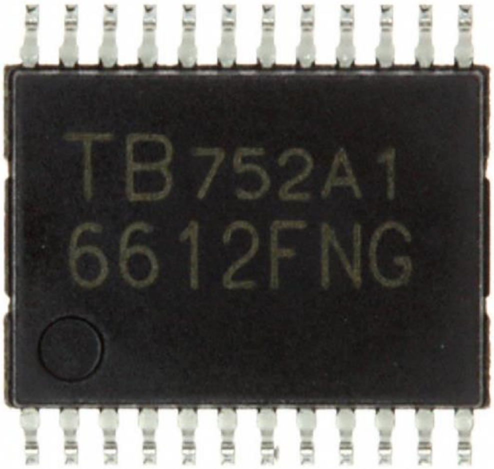
```
#define Sol 392
#define La 440
#define Si 494
#define Do_h 523
#define Re_h 587
#define Mi_h 659
#define Fa_h 698
#define Sol_h 784
#define La_h 880
#define Si_h 988
#include "RGBLed.h"
RGBLed rgbled_A3(7,A3);
int buzzer = 9;   // buzzer pin 9
int length;
// happy birthday Music score
int scale[] = {Sol, Sol, La, Sol, Do_h, Si, Sol, Sol,
             La, Sol, Re_h, Do_h, Sol, Sol, Sol_h, Mi_h,
             Do_h, Si, La, Fa_h, Fa_h, Mi_h, Do_h, Re_h, Do_h };
// Beats time
float durt[]={  0.5, 0.5, 1, 1, 1, 1+1, 0.5, 0.5,
                1, 1, 1, 1+1, 0.5, 0.5, 1, 1,
                1, 1, 1, 0.5, 0.5, 1, 1, 1, 1+1 };
void setup()
{
   pinMode(buzzer, OUTPUT);
   // get scale length
   length = sizeof(scale) / sizeof(scale[0]);
   Serial.begin(9600);
}
void loop()
{
   for(int x = 0; x < length; x++) {
      // Serial.println(scale[x]);
      tone(buzzer, scale[x]);
      rgbled_A3.setColor(0, scale[x] - 425, scale[x] - 500, scale[x] - 95);
      rgbled_A3.show();
      // 1= 3/4F so one Beats is  187*4 = 748ms
      delay(748 * durt[x]);
      noTone(buzzer);
   }
   delay(3000);
}
```

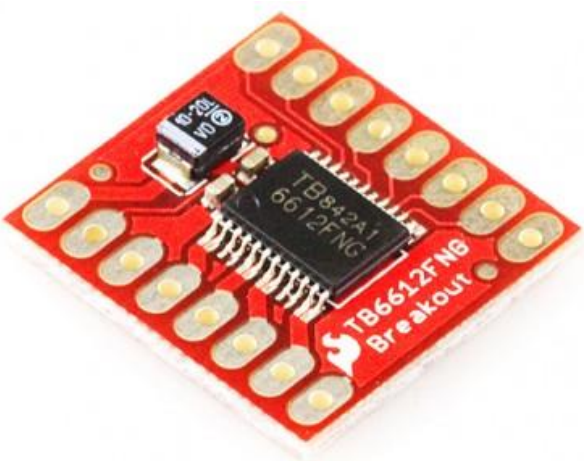
3.2.3 TB6612FNG Drive Principle

3.2.3.1 TB6612FNG Introduction

TB6612FNG is a new type of DC motor drive device produced by Toshiba Semiconductor Corporation. It is much more efficient than the traditional L298N, and its volume is also greatly reduced. Within the rated range, the chip basically does not generate heat, and of course it becomes even more delicate . The most important is that has the very high integration level, Independent two-way control of 2 DC motors, so in the integrated, miniaturized motor control system, it can be used as an ideal motor drive components.

Tb6612FNG has a large current MOSFET-H bridge structure, dual channel circuit output, each channel output highest 1.2 A continuous drive current, start peak current up to 2A/3.2A (continuous pulse/single pulse); 4 motor control modes: forward/reverse/brake/stop; Standby state; PWM support frequency of up to kHz, in-chip low-voltage detection circuit and thermal shutdown protection circuit, a common package for SSOP24 small patches, as shown in picture 3.2.3.1





Picture 3.2.3.1 TB6612 Physical drawings and common module diagrams

Tb6612fng main pin function: ainl/ain2, bin1/bin2, PWMA/PWMB for the control signal input terminal; AO1/A02, B01/B02 for 2-way motor control output end; STBY control pins for normal work/standby state; The VM (4.5~15 V) and VCC (2.7~5.5 V) are motor-driven voltage inputs and logical level inputs respectively.

In addition, TB6612FNG is a MOSFET based H-Bridge integrated circuit, which is more efficient than the transistor H-bridge drive. The output load capacity of L293D is increased by one-fold compared to the drive current of the average 600MA in each channel and the pulse peak current of 1.2 A. Compared with the L298N heat consumption and the peripheral diode continuous circuit, it does not need to add a heatsink, the peripheral circuit is simple, only the external power filter capacitor can directly drive the motor, to reduce the system size. For the PWM signal, it supports up to 100kHz frequency, relative to the above 2 chip 5 kHz and 40kHz also has a greater advantage.

3.2.3.2 Motor Control Unit Design

Unit Hardware Composition

Picture 3.2.3.2 shows the pulse and voltage diagram of the TB6612FNG. SCM Timer generates 4-channel PWM output as AIN1/AIN2 and BIN1/BIN2 control signals, A01 and A02, B01 and B02 control of Motor M1 and M2 in the picture. Using the timer output hardware PWM pulse, the SCM CPU only in the change of PWM duty ratio, the participation operation, greatly reducing the system operation burden and PWM software programming overhead. The input pins PWMA, PWMB and Stby are controlled by the I/0 level of the motor or the braking state as well as the device working status. The circuit uses voltage-V-10μf electrolytic capacitors and 0.1μf capacitors for power filtering, using power MOSFET to provide power reversal protection for VM and VCC.。

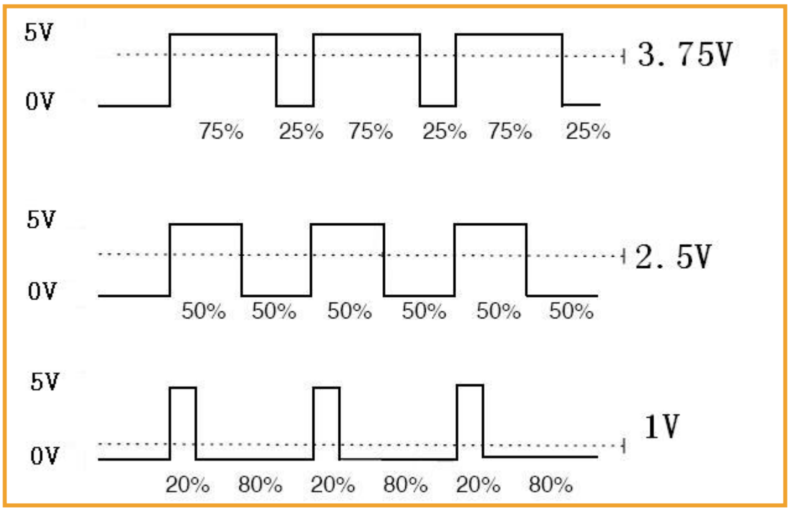
Software Control Software Implementation

Pulse width modulation generates the PWM signal of duty ratio change, and realizes the speed control of the motor by fast switching of the output state of the drive. The size of the PWM duty ratio determines the average output voltage, and then determines the speed of the motor. In general, single polarity and constant frequency PWM modulation method are adopted to ensure the stability of motor speed control. TB6612FNG logical truth-table is shown in tables three. When the device is working, the STBY pin is set to a high level, the IN1 and IN2 are unchanged, the input signal of the PWM pin can be controlled by the motor one-way speed, the PWM pin is a high level, and the input signal of IN1 and IN2 can be controlled by the bidirectional speed of the motor. The control logic of the A and B channels in the table is the same.。

Input				Output		
IN1	IN2	PWM	STBY	01	02	Mode status
H	H	H/L	H	L	L	brake
L	H	H	H	L	H	Reverse
L	H	L	H	L	L	brake
H	L	H	H	H	L	Forward
H	L	L	H	L	L	brake
L	L	H	H	off		stop
H/L	H/L	H/L	L	off		Standby

Table three TB6612FNG logical truth tables

In Arduino, can not output analog voltage, can only output 0 or 5V of digital voltage value, we use the high resolution counter, using the square wave of the duty ratio is modulated to a specific analog signal level coding. The PWM signal is still digital, because at any given moment, the full amplitude of the DC power supply is either 5V (on) or 0V (off). The voltage or current source is added to the simulated load by a repetitive pulse sequence with a pass (on) or a break (off). When the pass is the DC power is added to the load, when the break is the power is disconnected. As long as the bandwidth is sufficient, any analog value can be encoded using PWM. The output voltage value is calculated through the time of the pass and break. Output voltage = (connect time/pulse time) * Maximum voltage value, picture 3.2.8 for pulse change corresponding voltage.



Picture 3.2.3.2 Pulse and Voltage diagram

The Arduino Uno has 6 PWM interfaces, namely the Digital Interface 3, 5, 6, 9, 10, 11. To ensure that the expansion board at the same time to support a variety of cars, here we choose 5, 6 (Timer 2) as the motor PWM control io, detailed IO definition as shown in Figure 3.2.15, can also refer to the supporting information in the **" Schematic\Moto Shield board V3.0.pdf".**

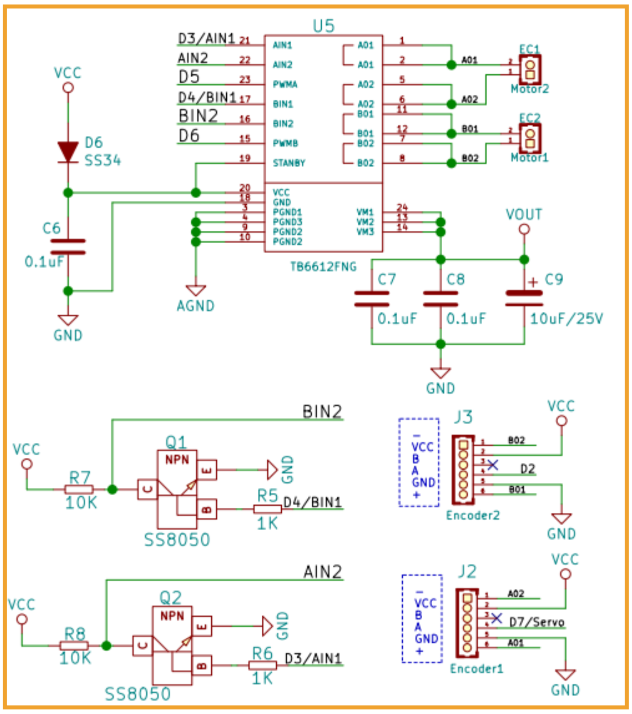


Fig. 3.2.3.2 TB6612FNG Application Circuit diagram

3.2.4.3 Motor Test Procedure

After knowing about these knowledge, we can control the motor, but before the control, we had better test whether the motor is working properly, the best way is to write a test program, download the program to Arduino, to observe whether the motor in accordance with our expectations set speed and direction of rotation. In the supporting information, we provide a small section of the Motor test program (file name "**Lesson\ModuleDemo\TB6612_DC_Motor\TB6612_DC_Motor.ino**"), of course, you can also write your own program. It's very simple,when we download the program to Arduino, we can see the motor: The story two seconds-----reverse two seconds-----stop two seconds.If it cycles like this all the time,indicating that the motor is working properly.

Then you can easily control the car by matching the other modules together.

```
#define AIN1 3
#define BIN1 4
#define PWMA 5
#define PWMB 6
void setup()
{
  Serial.begin(9600);
  pinMode(AIN1, OUTPUT);
  pinMode(BIN1, OUTPUT);
  pinMode(PWMA, OUTPUT);
  pinMode(PWMB, OUTPUT);
}
void loop()
{
  digitalWrite(AIN1,HIGH);
  digitalWrite(BIN1,LOW);
  analogWrite(PWMB,255);
  analogWrite(PWMA,255);
  delay(2000);
  digitalWrite(AIN1,LOW);
  digitalWrite(BIN1,HIGH);
  analogWrite(PWMB,255);
  analogWrite(PWMA,255);
  delay(2000);
  digitalWrite(AIN1,LOW);
  digitalWrite(BIN1,HIGH);
  analogWrite(PWMB,0);
  analogWrite(PWMA,0);
  delay(2000);
}
```

3.2.4 Ultrasonic Obstacle Avoidance

In this product, we will integrate the ultrasonic module and steering engine together and make the two parts working at the same time, which greatly increases the effectiveness of the data and the flexibility of the car, the main working flow: When the power is on, steering engine will automatically rotates to 90 degrees, the MCU will read data from the reflected ultrasonic. If the data is greater than the security value, the car will continue to drive forward, otherwise the car will stop, then the steering engine will rotate 90 degrees to the right. After that, the MCU reads data from the reflected ultrasonic again, the steering engine rotates 180 degrees to the left, then reading data again, the steering engine rotate 90 degrees, the MCU will contrast the two detected data, if the left data is greater than the right data, the car will turn left, otherwise turn right, if the two data are both less than the safety value, the car will turn around.

3.2.4.1 Suite Introduction

1.The steering gear

The steering gear is also called servo motor which is originally used in ships, since it can control the angle continuously through the program, so it has been widely used in intelligent steering robot to achieve all kinds of joint movement, the characteristics steering gear are small volume, large torque, high stability, simple external mechanical design. Either in hardware or software design, the design of steering engine is an important part of car controlling, the steering gear is mainly composed of the following parts in general, steering wheels, gear group, position feedback potentiometer, DC motor and control circuit (shown in Fig.3.2.4.1). Fig.3.2.4.2 shows the most commonly used 9G steering gear now.

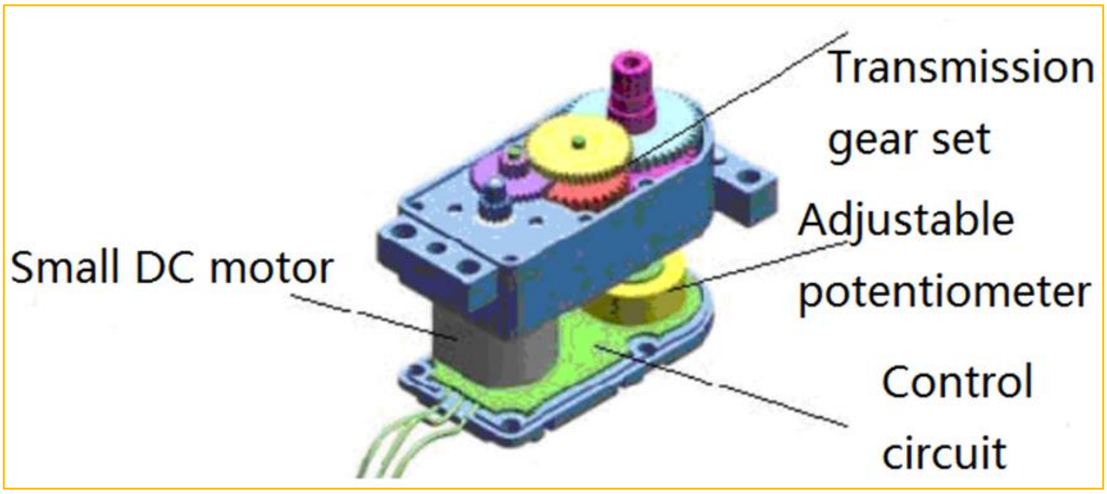


Figure .3.2.4.1 Diagram of Steering Gear

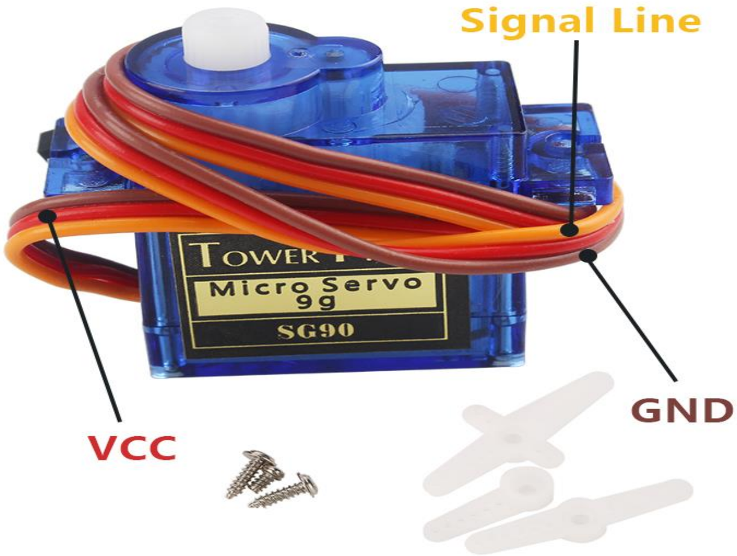


Figure .3.2.4.2 Physical Map of Steering Gea

3.2.4.2 Working Principle

1. Steering gear

The control signal enters the signal modulation chip by the receiver channel, gets the DC bias voltage. The steering gear has a reference circuit which generates a reference signal with a period of 20ms and a width of 1.5ms. Comparing the obtained DC bias voltage with the voltage of the potentiometer and obtaining the output voltage difference. Finally, the positive and negative output voltage difference in the motor driver chip decide the positive and negative rotation of motor. When the speed of motor is certain, the cascade reducer gear will drive potentiometer to rotate so that the voltage difference is reducing to 0, the motor will stop rotating.

When the control circuit receives the control signal, the motor will rotate and drive a series of gear sets, the signal will move to the output steering wheel when the motor decelerates. The the output shaft of steering gear is connected with the position feedback potentiometer, the potentiometer will output a voltage signal to the control circuit board to feedback when the steering gear rotates, then the control circuit board decides the rotation direction and speed of the motor according to the position, so as to achieve the goal. The working process is as follows: control signal→control circuit board→motor rotation→gear sets deceleration →steering wheel rotation→position feedback potentiometer→control circuit board feedback.

The control signal is 20MS pulse width modulation (PWM), in which the pulse width varies linearly from 0.5-2.5MS, the corresponding steering wheel position varies from 0-180 degrees, which means the output shaft will maintain certain corresponding degrees if providing the steering gear with certain pulse width. No matter how the external torque changes, it only changes position until a signal with different is provided as shown in Fig.3.2.4.4. The steering gear has an internal reference circuit which can produce reference signal with 20MS period and 1.5MS width, there is a comparator which can detect the magnitude and direction of the external signal and the reference signal, thereby produce the motor rotation signal.

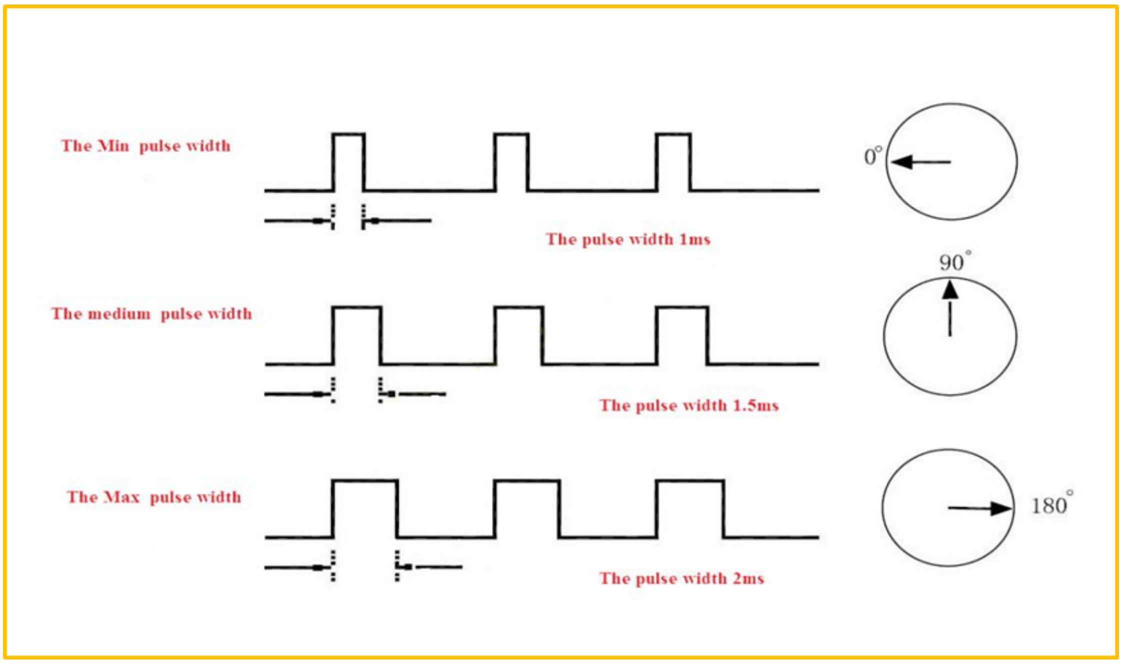


Figure .3.2.4.4 Relationship between the Motor Output Angle and Input Pulse

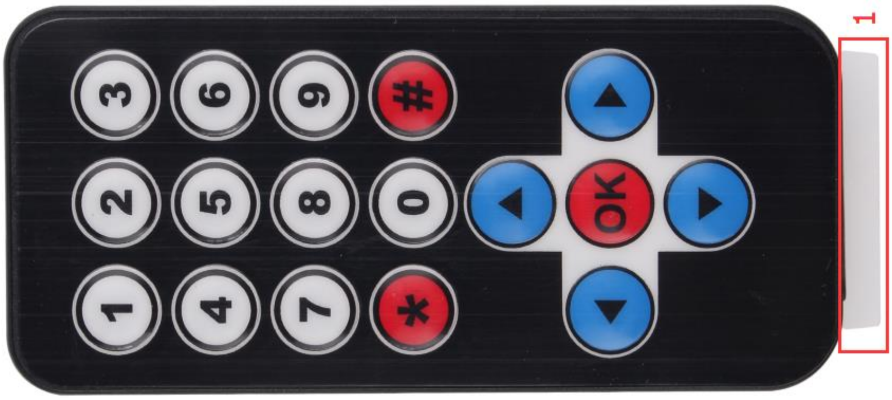
3.2.1.1 Steering test

Program location ： Lesson\ModuleDemo\Servo\ServoTest\ServoTest.ino

3.2.5 Infrared Remote Control

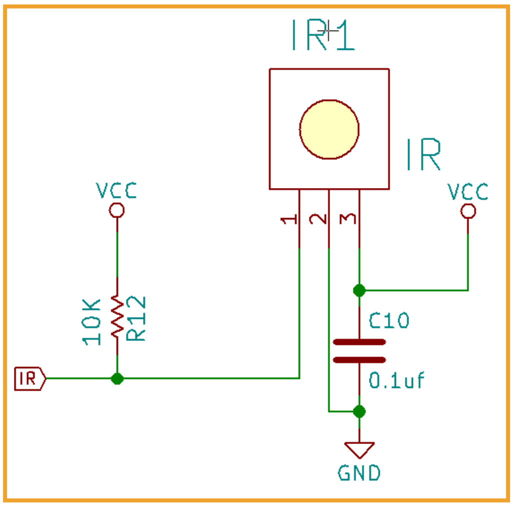
3.2.5.1 Introduction

Infrared remote control is widely used in every field at present.Infrared wireless remote control consists of Mini ultra-thin infrared remote controller (physical map shown in the picture 3.2.22) and integrated 38KHz infrared receiver. Mini ultra-thin infrared remote controller has 17 function keys, and the launch distance is up to 8 meters. Suitable for indoor control of various devices.



Picture 3.2.22 Infrared remote Control physical map

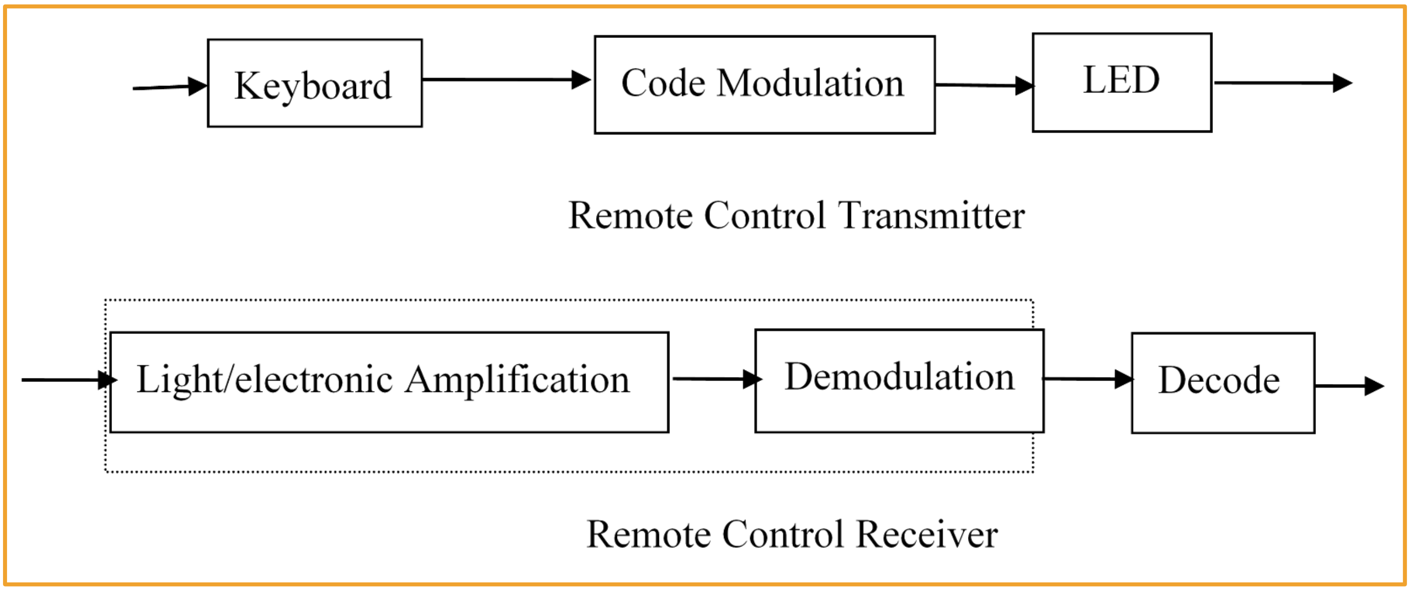
In the " Panther-Tank " car, integrated IR receiver head has been added to the expansion board, just need to plug the expansion board into the Arduino, and in the program defined pins (8th number IO), the IR receiver head has three pins, including power supply feet, grounding and signal output feet. The circuit is shown in the picture 3.2.23. The ceramic capacitor 0.1uf is a decoupling capacitor, which filters out the interference from the output signal. The 1-terminal is the output of the demodulation signal, which is directly connected with the Arduino 8th of the single-chip microcomputer. When the infrared coded signal is emitted, the output of the square wave signal after the infrared joint is processed, and is provided directly to the Single-chip microcomputer, and the corresponding operation is carried out to achieve the purpose of controlling the motor.



Picture.3.2.23 Infrared receiver Head circuit diagram and physical map

3.2.5.2 Working Principle

Remote control system is generally composed of remote control (transmitter), receiver, when you press any button on the remote control, the remote will produce the corresponding coded pulse, output a variety of infrared as the medium of control pulse signal, these pulses are computer instruction code, infrared monitoring diode monitoring to infrared signals, The signal is then sent to the amplifier and the limiter, which controls the pulse amplitude at a certain level, regardless of the distance between the IR transmitter and the receiver. The AC signal enters the bandpass filter, the bandpass filter can pass the load wave of 30KHZ to 60KHZ, through the demodulation circuit and the integral circuit to enter the comparator, the comparator outputs the high and low level, restores the signal waveform of the transmitting end. As shown in the 3.2.24.

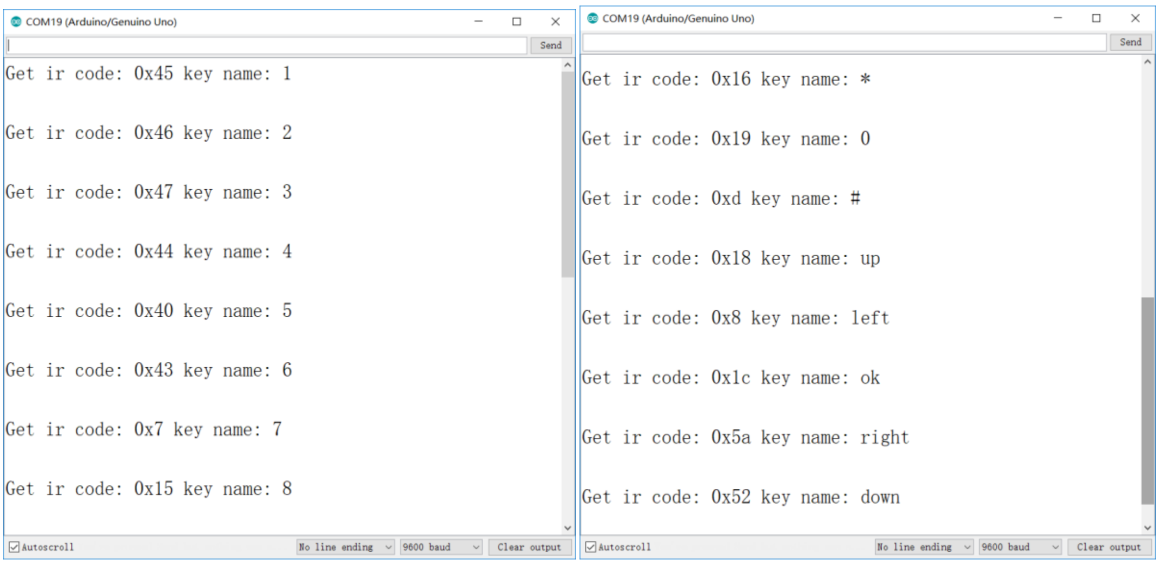


Picture.3.2.24 Infrared emitter and receiver system block diagram

3.2.5.3 Acquiring Infrared remote value

Open the program named “Lesson\ModuleDemo\IrkeyPressed\IrkeyPressed.ino” in the file and download it to the development board. Unplug the transparent plastic piece marked “1” in Figure 3.2.22. Then open the "serial monitor" and use the remote control to align the receiver head and press any key to observe the value displayed in the "serial monitor" and record it for later development as shown in the picture.

```
#include "IRremote.h"
IRremote ir(12);
unsigned char keycode;
char str[128];
void setup() {
    Serial.begin(9600);
    ir.begin();
}
void loop()
{
    if (keycode = ir.getCode()) {
        String key_name = ir.getKeyMap(keycode);
        sprintf(str, "Get ir code: 0x%x key name: %s \n", keycode, (char
*)key_name.c_str());
        Serial.println(str);
    } else {
      //   Serial.println("no key");
    }
    delay(110);
}
```



Picture 3.2.25 Remote coded query

In the picture 3.2.25, we can look at the IR code "0x45" and KeyName "1" two values, where "0x45" is a remote control key code, "1" is the Remote control button function name. Matching remote control all encoded values in **Lesson\ModuleDemo\IrkeyPressed\Keymap.cpp.**

3.2.5.4 Infrared Remote Control Procedure

Program Location:Lesson\Advanced Experiment\PantherTank_IR\ PantherTank_IR.ino

The above programs are infrared remote control reference routines, we can open the supporting program and download to the Development Board, according to the picture 3.2.26 to use remote control.

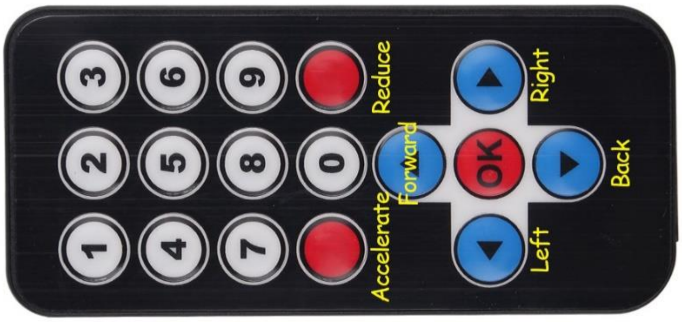


Figure 3.2.26 Infrared remote control instructions

3.2.6 Mobile phone Bluetooth control

3.2.6.1 Module Introduction

Tank support mobile phone Bluetooth app remote control function. The Bluetooth module used in the racing car is the JDY-16 ble module.

JDY-16 transmission module is based on Bluetooth 4.2 protocol standard, the working band for the 2.4GHZ range, modulation mode for the GFSK, the maximum emission power of 0db, the maximum emission distance of 80 meters, the use of imported original chip design, support users through the AT command to modify the device name, service UUID, transmit power, Matching password and other instructions, convenient and quick to use flexible. Module information see Tank\Document\JDY-16-V1.2(English manual).pdf,the module physical map as shown in the picture 3.2.27.

Picture 3.2.27 JDY-16 Module

3.2.6.2 Function Introduction

- BLE high speed transmission, support 8K Bytes rate communication
- There is no limiton of bytes when sending and receiving datas, supporting 115200 baud rate continuous transceiver data .
- Support 3 working modes (please see At+starten instruction function description)
- Support (serial, IO, APP) sleep wake
- Support WeChat Airsync and WeChat applets、applications in micro-credit H5 or factory server communication and communication with APP
- Support 4-Way IO port control (applied to mobile phone control relays or LED lights Out)
- Support high precision RTC clock
- Support PWM function (via UART, IIC, APP, etc.)
- Support UART and IIC communication mode, the default is UART communication
- Ibeacon mode (support for micro-signal shake-roll protocol and Apple ibeacon Protocol)
- Host transmission mode (application of data transmission between modules, host and from machine communication)

3.2.6.3 Bluetooth test

JDY-16 test please see **"Panther-Tank\JDY-16\《JDY-16 Module Test.pdf》"**

3.2.6.3.3Bletooth Control Panther-Tank Principle

Use Bluetooth to control the car, in fact is using the Android app to send instructions to the Arduino serial port via Bluetooth to control the car. Since it involves wireless communication, one of the essential problems is the communication between the two devices. But there is no common "language" between them, so it is necessary to design communication protocols to ensure perfect interaction between Android and Arduino. The main process is: The Android recognizes the control command and package it into the corresponding packet, then sent to the Bluetooth module (JDY-16), JDY-16 received data and send to Arduino, then Arduino analysis the data then perform the corresponding action. The date format that the Android send as below, mainly contains 8 fields.

Protocol Header	Data Length	Device Type	Device Address	Function Code	Control Data	Check Sum	Protocol End Code

In the 8 fields above,we use a structural body to represent.

```
typedef struct
{
    unsigned char start_code;      // 8bit 0xAA
    unsigned char len;
    unsigned char type;
    unsigned char addr;
    unsigned short int function;   // 16 bit
    unsigned char *data;           // n bit
    unsigned short int sum;        // check sum
    unsigned char end_code;        // 8bit 0x55
}ST_protocol;
```

“Protocol Header” means the beginning of the packet, such as the uniform designation of 0xAA.

“Data length”means except the valid data length of the start and end codes of the data.

“Device type”means the type of device equipment

“Device address” means the address that is set for control

“Function code”means the type of equipment functions that need to be controlled,the function types we currently support as follows.

```
typedef enum
{
    E_BATTERY = 1,
    E_LED = 2,
    E_BUZZER = 3,
    E_INFO = 4,
    E_ROBOT_CONTROL_DIRECTION = 5,
    E_ROBOT_CONTROL_SPEED = 6,
    E_TEMPERATURE = 7,
    E_IR_TRACKING = 8,
    E_ULTRASONIC = 9,
```

```
    E_VERSION = 10,
    E_UPGRADE = 11,
}E_CONTOROL_FUNC ;
```

“Data”is about the exact values that we control the racing car,such as speed and angle

“Checksum” is the result of different or calculated data bits of the control instruction.

“Protocol end code ”is the end part of the data bag,when receiving this data,it means that the data pack has been sent,and is ready for receiving the next data pack,here we specified it as 0x55.

For example: A complete data pack can be like this: “AA 07 01 01 06 50 00 5F 55”,

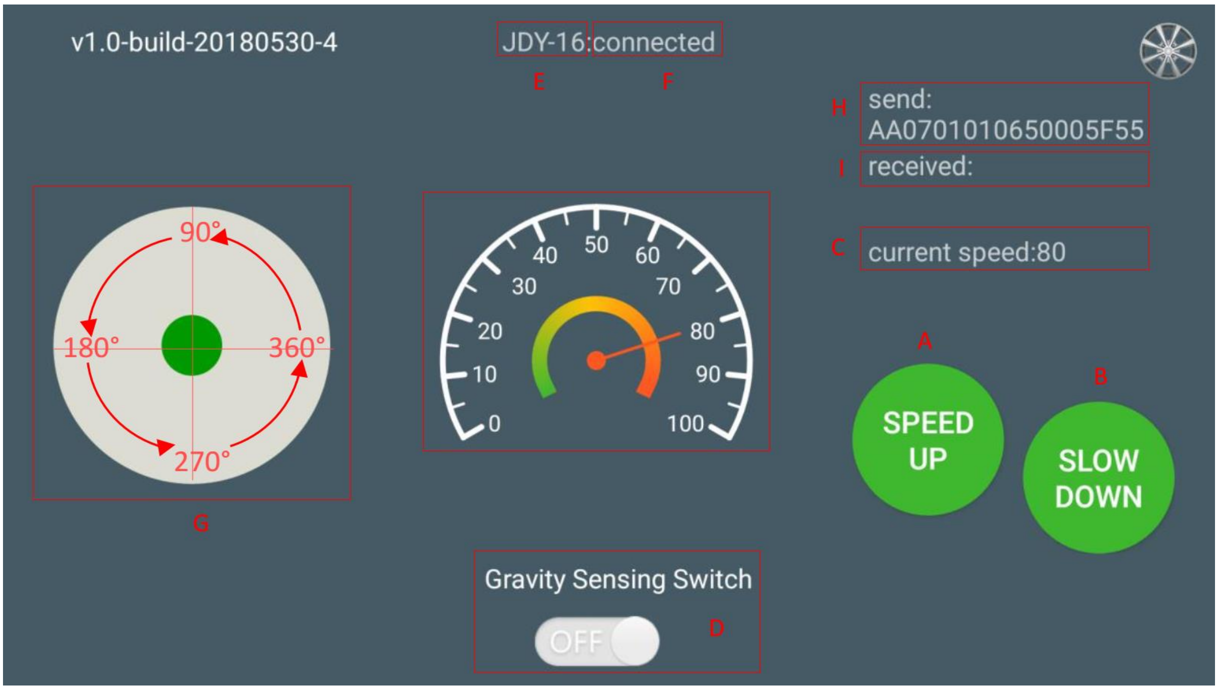
“07” Transmission Data Length 7 bytes

“06” is “Device type”， the device that we specified,such as LED、buzzer;here 06 means the “speed” of transportation,05 means the “direction” of transportation.

"50 (or 0050)" is the control data, where 0x50 is hexadecimal, converted to binary 80, if "Device type" transmits 06, then the data here is the speed value, that is, the speed is 80. If the transfer 05 o'clock is the control direction, that is, the 80° direction (forward).

“005F” is a checksum that is 0x07+0x01+0x01+0x06+0x50=0x5F.

“55” is the end code of the Protocol, indicating the end of data transfer.



Picture 3.2.29 Android Interface diagram

Above the picture 3.2.29

The "A, B" section is a plus deceleration button.

The "C" section includes the dashboard and the digital display area, two parts for synchronized display. Indicates the current speed, that is, the speed of the acceleration and deceleration.

The "D" section is a gravity remote sensing switch, which can be switched to the gravity remote sensing mode. The "E" section indicates the Bluetooth name that is currently connected to.
The "F" section indicates the Bluetooth connection status, and if Bluetooth is not connected, the "disconnected" is displayed here.
The "G" section is the hand shakes the pole, the slippery sway pole may let the car rotate direction.
The "I" section is a data return area, such as the current driving state and speed of the trolley.
The "H" section is for packets sent, such as the data "AA 2B 55", which indicates that the speed is 35 (23 is 16 data, and conversion to 10 is the current speed of 35).
If the data sent is “AA 08 01 01 05 00 5B 00 6A 55”,it means that the car is driving forward(05 is a direction command instruction,91 is the value when 005B converted to binary.from picture 3.2.17, we can know that 91°also means moving forward)

3.2..6.4 Experimental procedure

1.Connect the bluetooth module to the Arduino serial port (this step we ignore directly, our expansion board has connected Bluetooth and Arduino serial port, more convenient, faster, stable).
2. Turn on the mobile bluetooth(Note: Do not connect the car’s bluetooth in the phone settings, you can connect directly in our app) install "appbluetoothcontrolcom.keywish.robot.apk" to you phone and open the app (there is a software installation package in the accessory CD that currently only support Android phones, later will release the iOS version) ,it will automatically connect our car bluetooth.

3.2.6.5 Software Design

Bluetooth part of the program includes many library files, we do not annotate here, please open the file “**Lesson\Panther-Tank\ Panther-Tank.ino**”, download the program to the racing board, according to the above mobile phone APP operation to control racing car. We will continue to improve and increase the function of the app, please pay attention to our github updates.

3.2.7 PS2 Wireless Control (optional)

3.2.7.1 Introduction

Picture 3.2.30 PS2 wireless handle

PS2 handle is composed of two parts, the handle and receiver, the handle needs two section 7th 1.5V batteries, the receiver and arduino control board use the same power supply, the voltage is 3~5v, can not be reversed, can not exceed , overvoltage and reverse connection will cause the receiver to burn out. There is a power switch on the handle,turn handle switch to on, the lamp on the handle will flash if didn’t search the receiver, the handle will enter Standby mode if this station last for some time,the light will go out. At this time, you need to press the "START" button to wake-up handle.

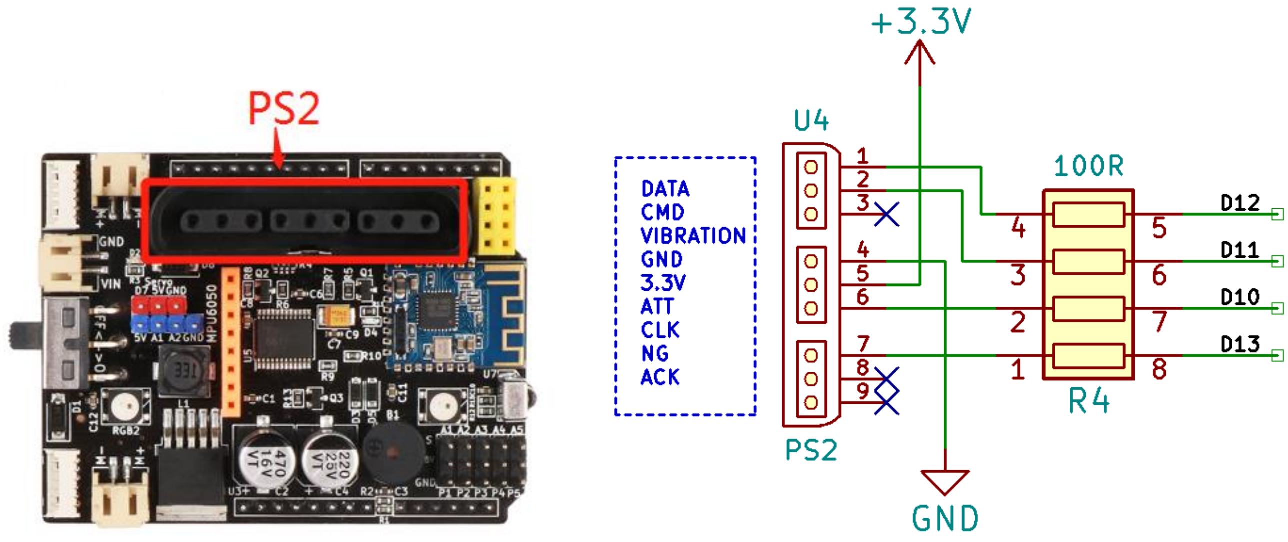
The receiver is connected to the Arduino, and powered by Arduino, such as the picture 3.2.40, in an unpaired condition,a green light will flash.If the handle is open, the receiver is powered, the handle and receiver will automatically pair, then the lamp is always bright, the handle pair succeeds. The key "mode" (handle batch is different, the identification may be "analog", but will not affect the use), you can choose "Red light Mode", "Green mode".

Some users reflect that the handle and receiver can not be paired properly!

Most of the problems are that the receiver wiring is incorrect or the program is incorrect.

Solution:The receiver only receives the power supply (the power cord must be connected correctly), without any data line and clock line, the handle is usually able to pair successfully. Paired success then lights will keep bright, indicating that the handle is good.

Then check if the wiring is correct, Is there any problem with program migration?



Picture 3.2.31 Remote receiver module

There are 9 interfaces on the the receiver, shown in the following table:

1	2	3	4	5	6	7	8	9
DI/DAT	DO/CMD	NC	GND	VDD	CS/SEL	CLK	NC	ACK

Note: Different batches, the appearance of the receiver will be some different, but the pin definition is the same,so don’t worry about the use.

Di/dat: Signal flow, from the handle to the host, this signal is a 8bit serial data, synchronous transmission in the clock down the edge. The reading of the signal is done in the process of changing the clock from high to low.

Do/CMD: Signal flow, from the host to the handle, this signal is relative to DI, the signal is a 8bit serial data, synchronously transmitted to the clock down the edge.

NC: Empty port; .

GND: Ground;

VDD: The 3~5V power supply;

CS/SEL: Providing trigger signals for handles, the level is low during communication;

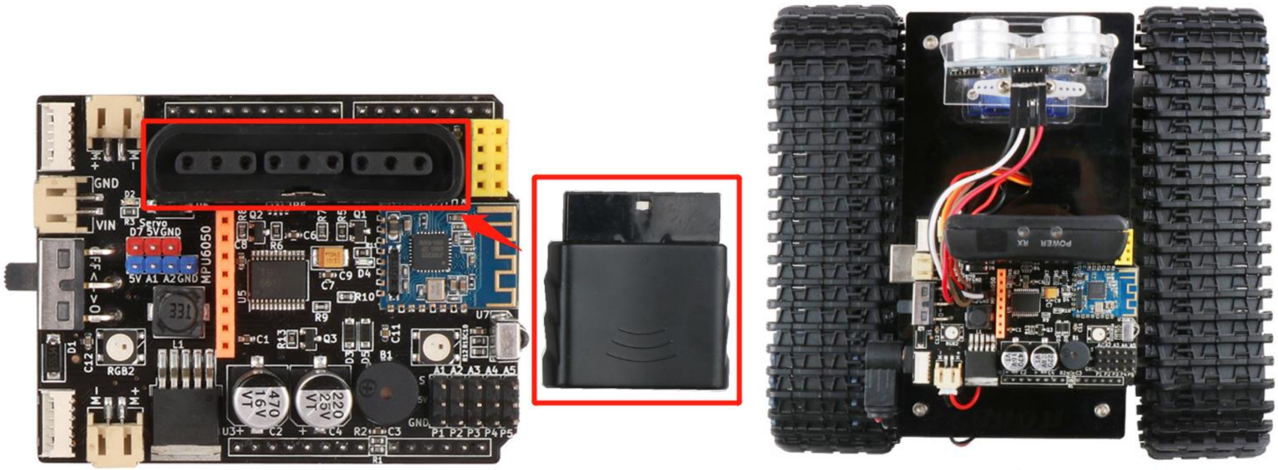
CLK: The clock signal is sent by the host to maintain data synchronization;

NC: Empty port;

ACK: the response signal from the handle to the host. This signal changes to low in the last cycle of each 8-bit data sending, and the CS remains low. If the CS signal do not remain low, the PS host will try another device in about 60 microseconds. The ACK port is not used in programming.

3.2.7.2 Experimental procedures

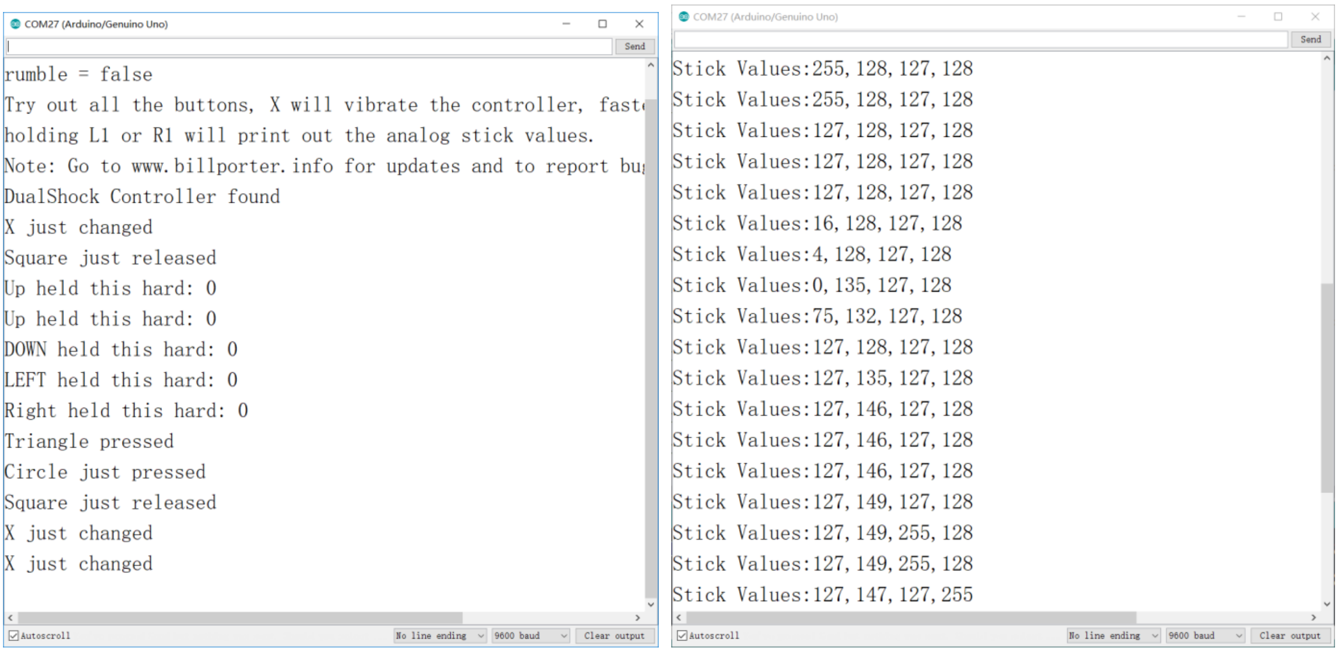
1、 For the sake of wiring simplicity, the PS2 we use can directly plug the PS2 receiver into the socket. As shown in Figure 3.2.33



Picture.3.2.33 Installation of Receiving Head

4、Open the CD"Lesson\ModuleDemo\PS2X\PS2X.ino ",download the program to the Arduino Board, open the PS2 remote control, if the receiver and the remote control has been connected (or the pairing is successful), the receiver’s led will keep light, if not the LED light constantly flashing.

Then we open "serial monitor", press any button on the remote control, you can see the corresponding data in the "serial monitor" ,shown as the picture 3.2.34.



Picture 3.2.34 “Serial monitor” Data Display

3.2.7.3 Software Design

Ps2 Control program in Lesson\Advanced Experiment\PantherTank_PS2\ PantherTank_PS2.ino

In above programs,we just finished the experiment of testing the button. In this experiment we want to implement the PS2 remote control car function. We first define all the button functions as follows:

Picture 3.2.34 Functions of PS2 Handle Buttons

PS2 handle description:

Mark UP: move forward

Mark DOWN: move backward

Mark LEFT: turn left

Mark RIGHT turn right

Mark A: Acceleration

Mark B: Turn left on the steering gear

Mark C: Deceleration

Mark D: steering gear turn right

Mark 3: right joystick (Mark 5) control key. When the R1 is pressed, the right joystick will work.

Mark 4: left joystick (Mark 6) control key. When L1 is pressed, the left joystick will work.

Joystick Right :adjust right rgb led

Joystick Left: adjust left rgb led

Start: power switch