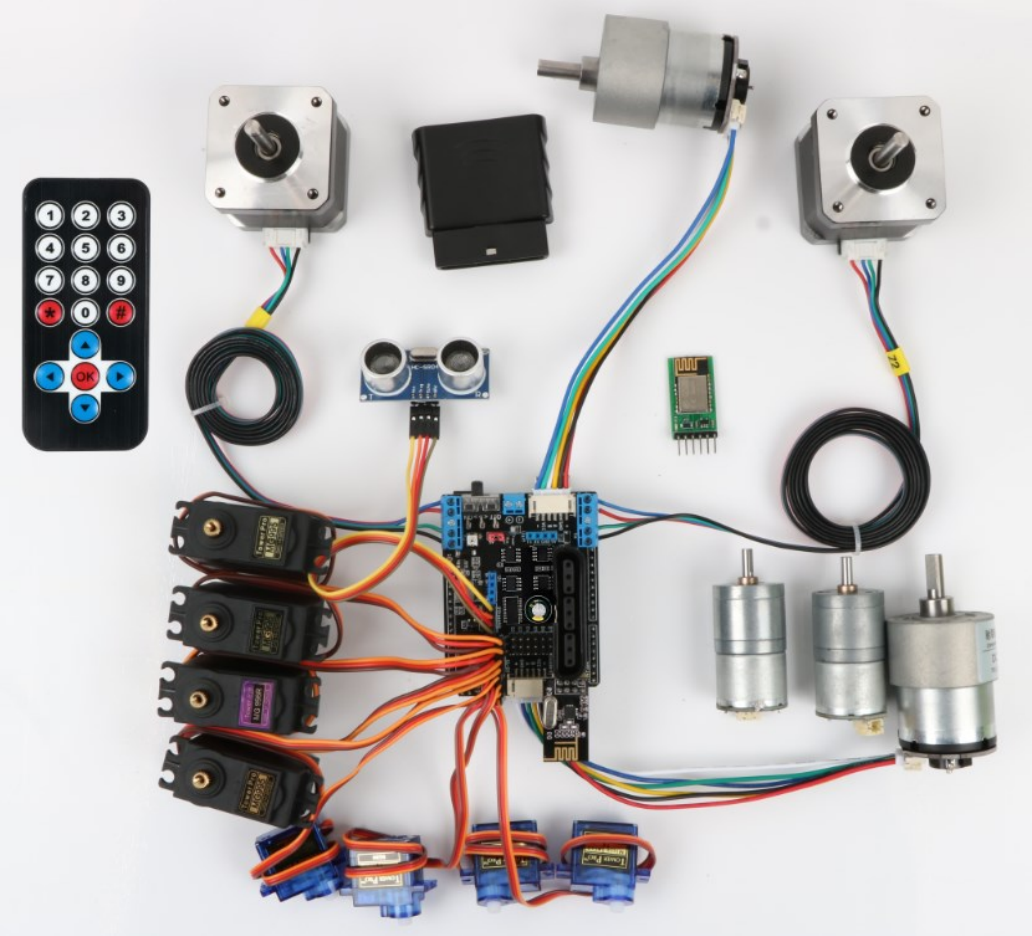
# PS2X&Motor Drive Board Instruction Manual V.1.6



Revision

| Date | Version | Description | Author |
|---|---|---|---|
| **2018/11/27** | V. 1. 0 | Create | **Abbott.Chen** |
| **2019/1/2** | V. 1. 1 | **Rearrange the document frame and fix the error** | **Ken.chen** |
| **2019/1/11** | V. 1. 2 | **Optimize and supplement documentation** | **Abbott.Chen** |
| 2019/1/14 | V. 1. 3 | **Modify the image, the loading code is partially highlighted** | **Ken.Chen** |
| 2019/9/20 | V. 1. 4 | **Modify the picture, modify the code path, increase the 6-way servo to 8** | **Carl.Du** |
| **2019/10/30** | V. 1. 5 | **Increase the use of program functions.** | **Carl.Du** |
| **2019/12/25** | V. 1. 6 | **Increase the program version switching and replace with the motor chip** | **Carl.Du** |

## Catalog

## Overview

PS2X & Motor Driver Board driver can drive 4 DC motors, 2 encoder motors, 2 stepper motors, 8 servos (can be connected to external power supply), the driving current is 1.2A, and the peak is 2A. This driver board is specially designed for Arduino Uno R3 and Arduino mega 2560 motherboards, which can be directly plugged into Arduino Uno and Arduino mega 2560. The motherboard integrates a passive buzzer, 2 RGB LED lights, and an infrared receiver head. Also reserve PS2 socket, Uart interface, I2C interface, ultrasonic obstacle avoidance module socket, 6 IO ports, it is very convenient to connect various sensor modules.

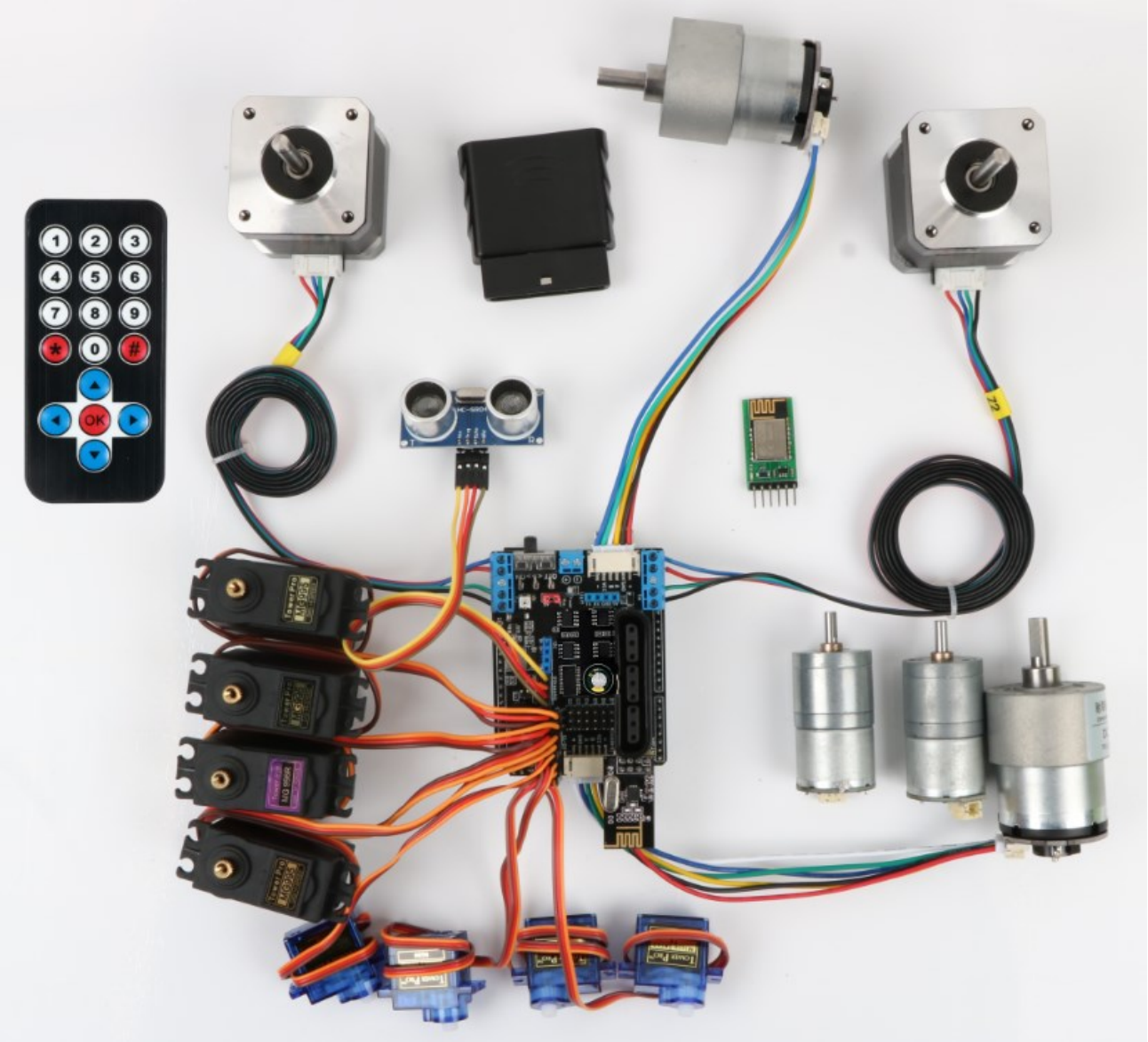
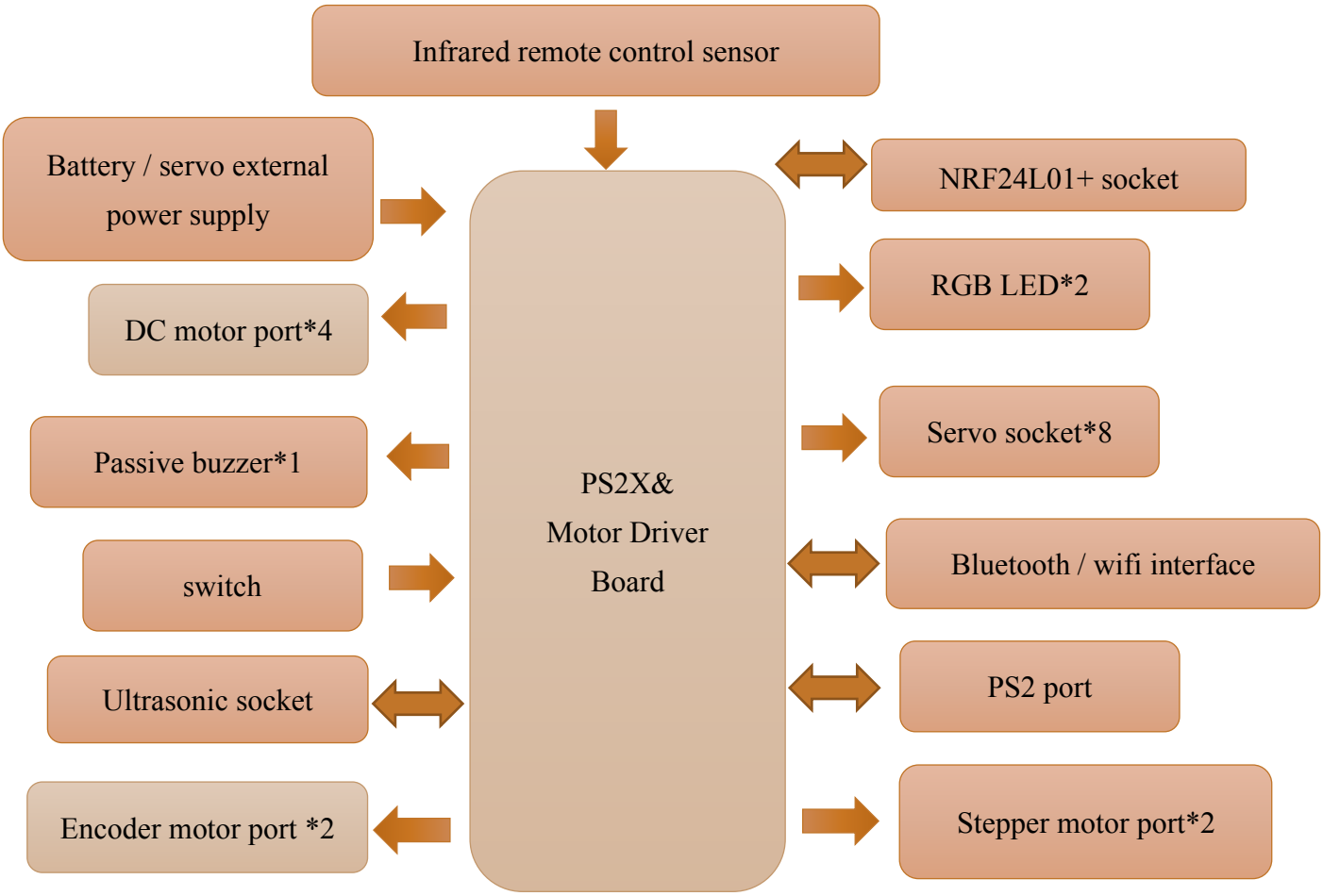



Figure 1-1 Function Display

# PS2X&Motor Driver Board Composition frame



## Driver Board Introduction

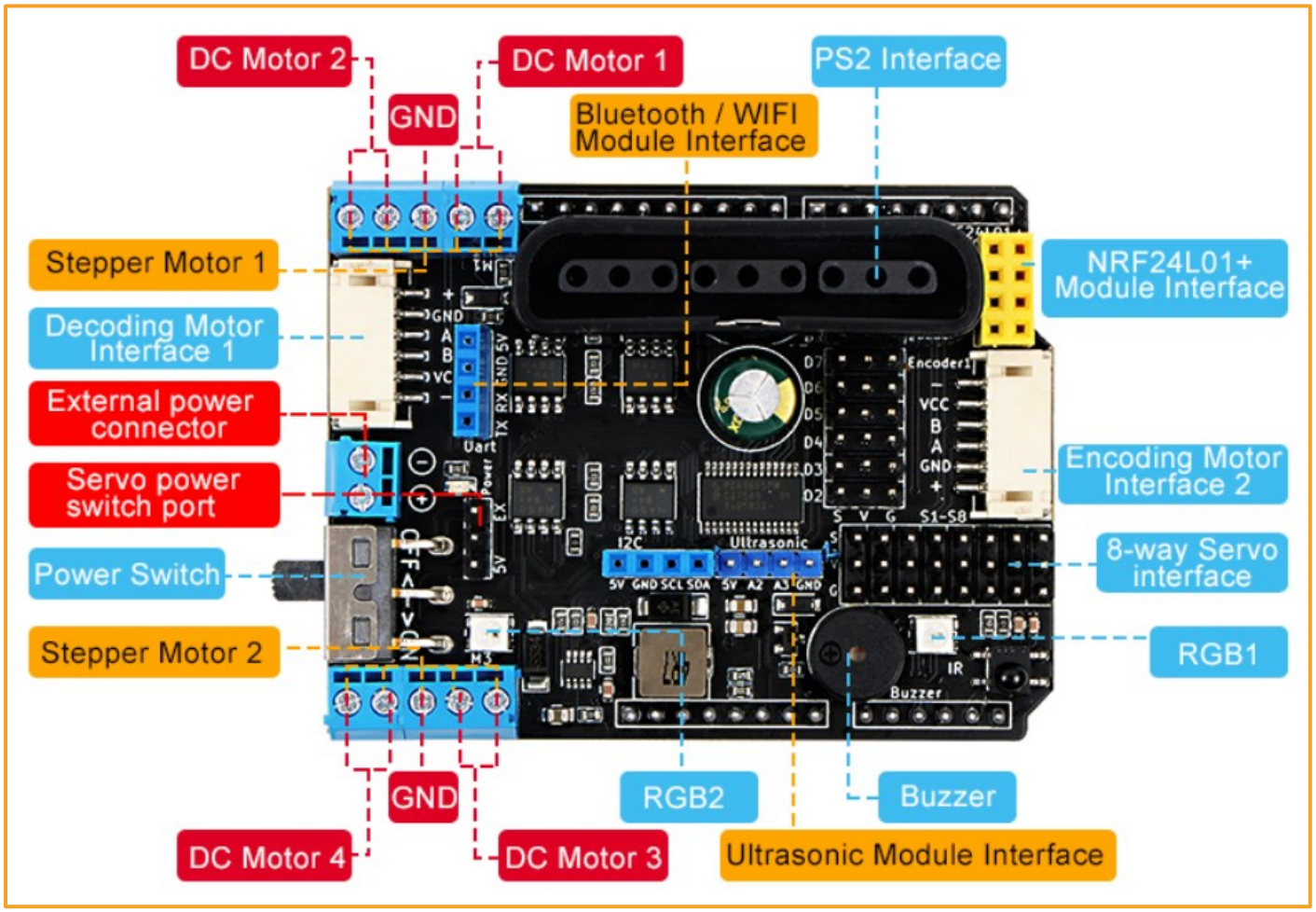



Figure 2-1 PS2X&Motor Driver Board Introduction

## Common problems

**Q：How to power the PS2X&Motor Driver Board?**

A：It can be directly powered by the DC port of Arduino UNO, with a voltage range of 6~35V.

**Q：How many motors does the PS2X&Motor Driver Board can drive?**

A：The PS2X&Motor Driver Board can drive 4 DC motors ,2 stepper motors or 2 encoder motors.

**Q：How does PS2X&Motor Driver Board connect power drive servo?**

A：PS2X&Motor Driver Board 8 servos support external power supply steering gear. The external power supply interface of the servo is connected to the power supply. The jumper cap is connected to EX, and the power supply of the external power supply is just fine.

**Q：I want to upload the sample program to the Arduino board, suggesting that the upload failed. What is the reason?**

A：Before uploading the sample program to the Arduino board, you need to check if the board and the computer are properly connected, then install the driver and try again.

**Q：I want to upload the sample program to the Arduino board, and then the motor does not work after turning on the power. What is the reason?**

A：Firstly, check if the green indicator on the Arduino UNO board is on. If it is not on, it means the power supply is not normal. Check whether the battery voltage is above 6v. Then check whether the motor's wiring port is consistent with the port set in the program. Turn it on again.

**Q：How to install graphical programming library?**

A：Download the relevant tutorial under this link: https://github.com/keywish/motor-drive-board

**Q：Where can we download the program examples?**

A：**Dow**nload the relevant tutorial under this link: https://github.com/keywish/motor-drive-board

**Q：My Arduino will crash when the motor is running. Is the drive broken?**

A：When the motor is running, the power consumption is large. You need to ensure that the battery has sufficient power. You can try to charge the battery before turning it on.

## Driver board power

The power on the driver board can be directly powered through the DC port of Arduino UNO. The input voltage range is 6 ~ 35V, the output voltage of the driver board is 5V, and the maximum output current is 3A.

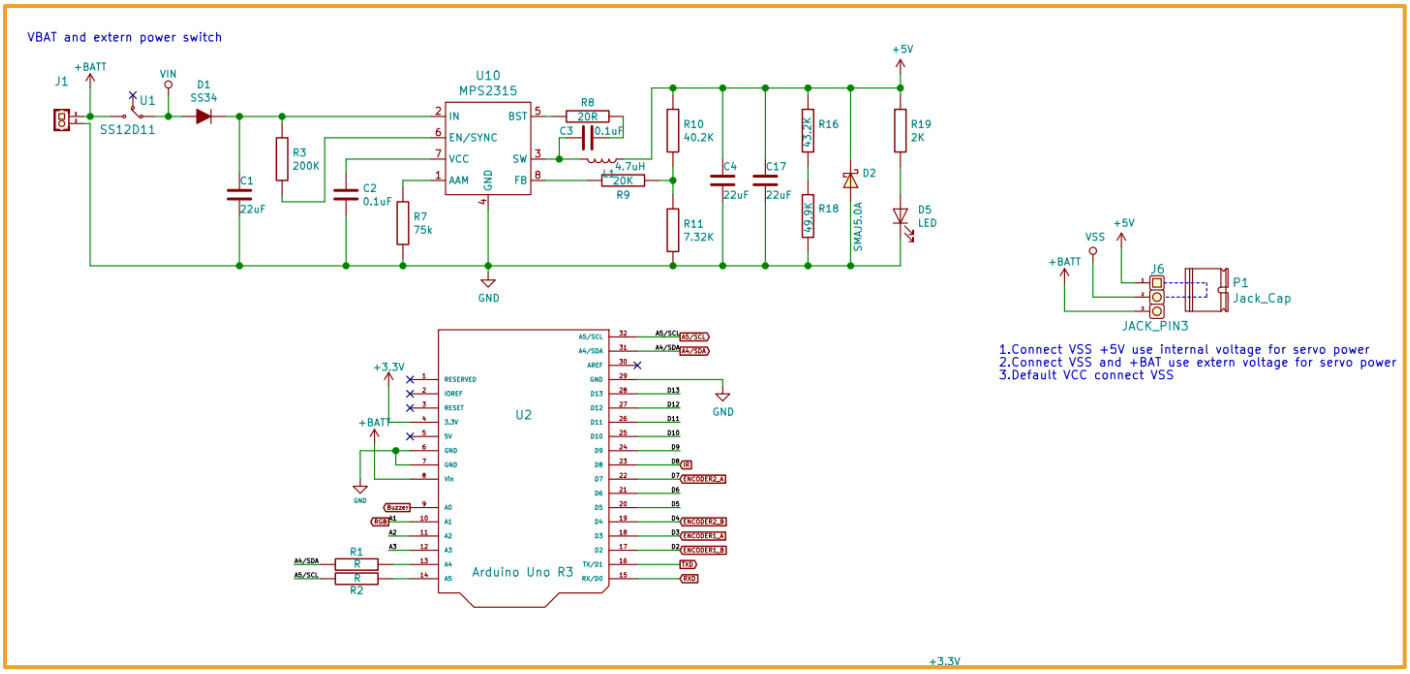



Figure 3-1 Power supply schematic

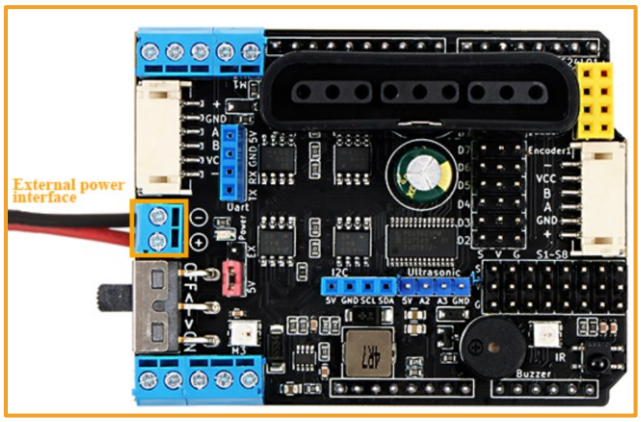



Figure 3-2 External power interface

Power can be supplied to MotorDriverBoard and UNO R3 / MEGA 2560 in two ways. One is to power the motherboard and MotorDrvierBoard through the DC head of UNO / MEGA 2560. The toggle switch needs to be turned ON. The other is to supply power through an external power interface. The external power supply can also supply power to the UNO board, but you need to turn the toggle switch to OFF.

Toggle switch: OFF is to use the external power supply of MotorDrvierBoard to power the entire drive board and motherboard, and it can also be used as an external power supply for the servo; NO is to use the UNO / MEGA 2560 motherboard to power the entire drive board or motherboard. UNO / MEGA 2560 power supply method It cannot be used as an external power supply for the servo.

Servo power:

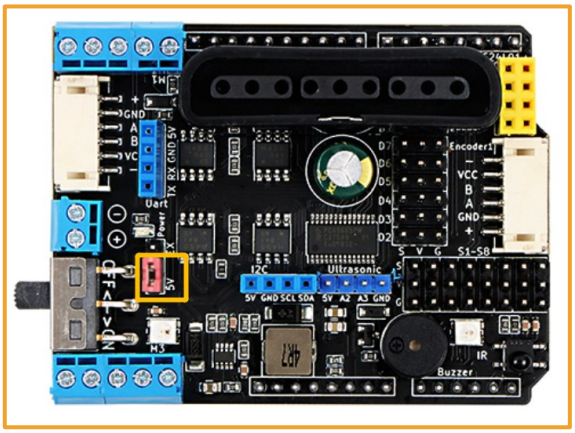


Figure 3-3 Expansion board power supply

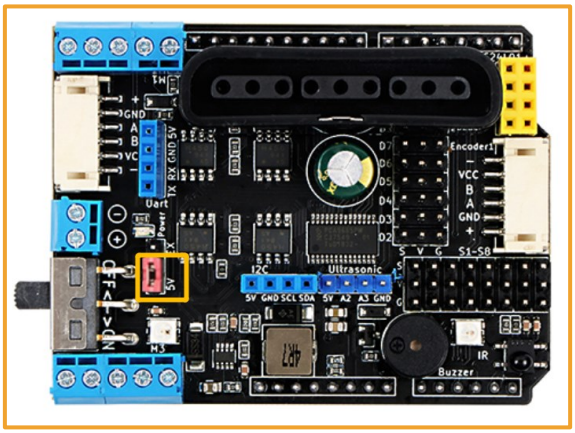


Figure 3-4 External power supply

When the shorting cap is plugged into 5V, the servo uses the step-down chip of the expansion board to supply power. The voltage is 5V and the maximum current can reach 3A.

When the shorting cap is plugged into the EX, the servo uses external power supply. There are two power supply methods for this external power supply, which are through the external power interface. This external power supply is directly from the external power supply.

# DC Motor

## Motor control principle

The PS2X&Motor Driver Board uses the PCA9685 to output the PWM control motor driver chip H450. Now we'll briefly introduce the two chips.

The main parameters of PCA9685 are as follows:

- I2C interface control can control 16-channel PWM and support up to 16 servos or PWM output, 12-bit resolution for per channel (4096 levels)
- Built-in 25MHz crystal oscillator, can be connected to external crystal oscillator or can also not be connected to external crystal oscillator, up to 50MHz.
- Supporting 2.3V-5.5V voltage, maximum withstand voltage is 5.5V

  With power-on reset, software reset and other functions

The device address of the PCA9685 is determined by the pins A0, A1, A2, A3, A4, and A5, and the pin cannot be left floating. Since there are six pins that together determine the device address, there are 64 device addresses. Since the IC keeps the LED All Call address (E0h, 1110 000) and Software Reset address (06h, 0000 0110) after power-on, there are only 62 available device addresses. Therefore, theoretically one I2C interface can control the way 16* 62=992 PWM, the pin control device address is shown in the figure below:

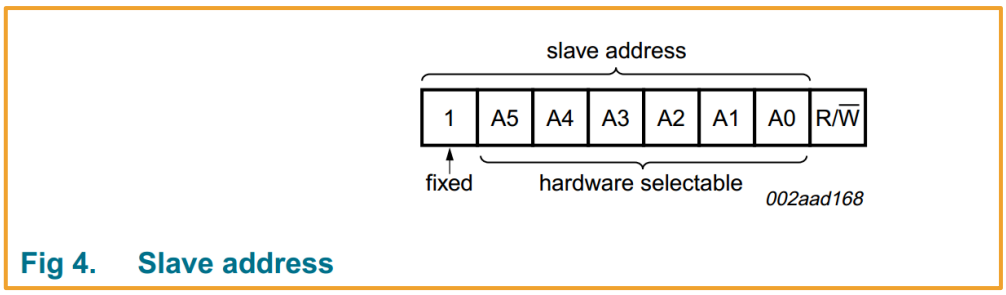



**Fig 4. Slave address**

Figure 4-1 Device address schematic

**For detailed use of the chip, please refer to《MotorDriverBoard\Datasheet\PCA9685.pdf》**

### Electrical drive introduction

PS2X & Motor Driver Board_V5.0 uses four H450 chips to drive the motor. It is a PWM chopper DC brush motor driver. One channel of the motor output block is embedded, manufactured by BiCD process, rated output voltage 50V, maximum current 3A, built-in output MOSFET with low on-resistance (high side + low side = 0.6Ω (typical)) Error detection functions (thermal shutdown (TSD), overcurrent detection (ISD) and undervoltage lockout (UVLO)).

| IN1 | IN2 | OUT1 | OUT2 | Function |
|---|---|---|---|---|
| L | L | OFF (Hi-Z) | OFF (Hi-Z) | Stop |
| | | | | Standby mode after 1 ms |
| H | L | H | L | Forward |
| L | H | L | H | Reverse |
| H | H | L | L | Short brake |

Figure 4-2 Diagram of pulse and voltage

## Driving DC motor

The PS2X&Motor Driver Board has four DC motor interfaces, namely DC motor interface 1, DC motor interface 2, DC motor interface 3 and DC motor interface 4, which can be directly connected to the drive via the terminals. Connect 4 DC motors to DC motor interface 1, 2, 3, 4 (as shown in Figure 4-3). After connecting the motor, connect the battery to the development board, turn on the power switch, and the program starts running. We will see the motor will turn up, and the drive schematic of the DC motor is shown in Figure 4-4.

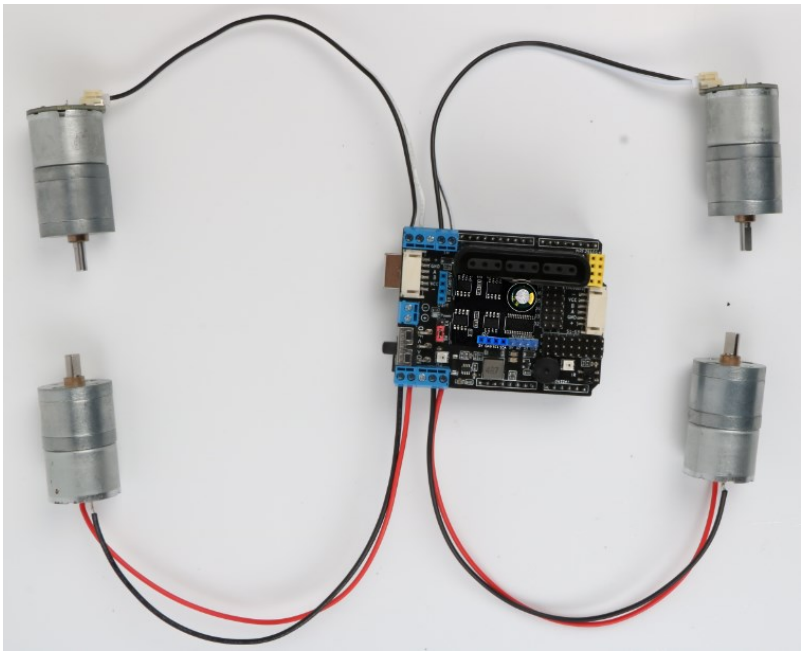


Figure 4-3 DC Motor connection diagram

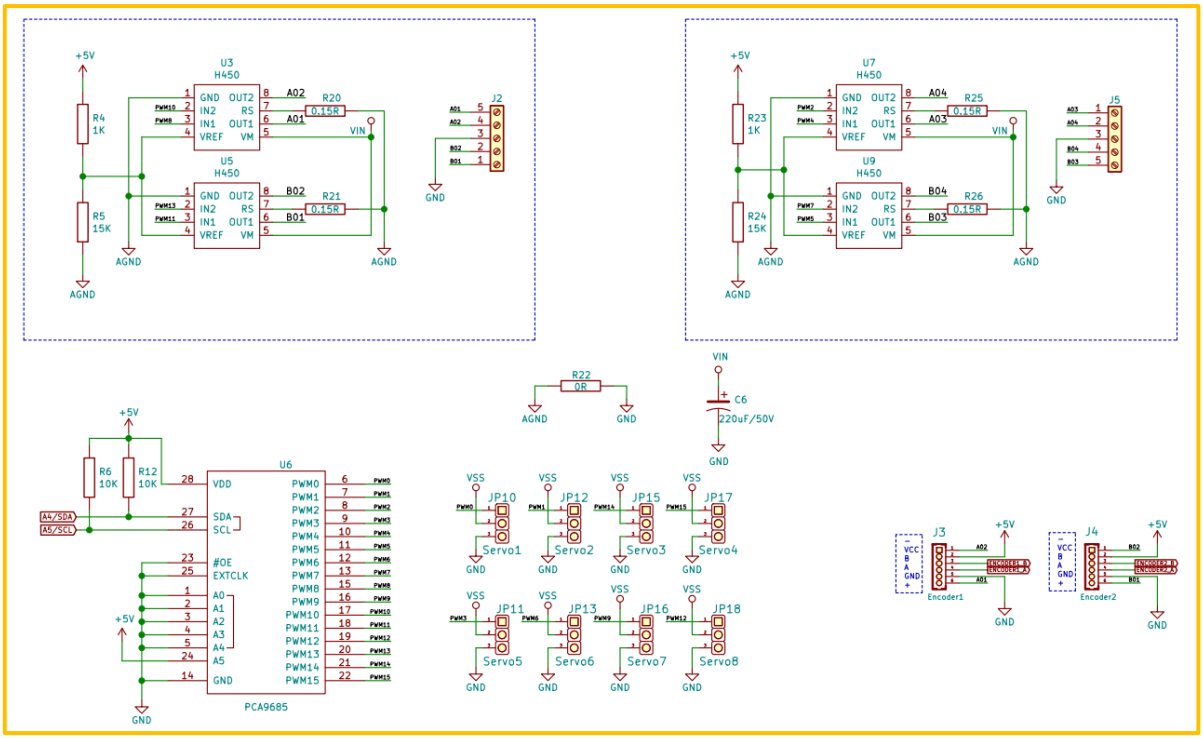



Figure 4-4 Motor drive schematic

## Experimental procedure

Open the IDE, load the file -> **Lesson\AdvancedExperiment\DC_Motor\ DC_Motor.ino**

According to the simple introduction of the program above, I can see that the DC motor test program is to let the M1, M2, M3, and M4 motors rotate for 4 seconds in the forward direction, the motors reverse for 4 seconds, and stop for 4 seconds.

## Program use

```
Emakefun_MotorDriver mMotorDriver
=Emakefun_MotorDriver(0x60,MOTOR_DRIVER_BOARD_V5);
// Initialization library, 0x60 is the I2C address of PCA9685, MOTOR_DRIVER_BOARD_V5
   is MotorDrvierBoar_V5, if your MotorDriverBoard is V4 version, please change
   MOTOR_DRIVER_BOARD_V5 to MOTOR_DRIVER_BOARD_V4.
Eakefun_DCMotor *DCMotor_1 = mMotorDriver.getMotor(M1);  // Initialize
the DC motor M1.
mMotorDriver.begin(50);          // When using DC motors and encoder
motors, the frequency must be set.
DCMotor_1->setSpeed(200);          // Set the motor speed. The speed is
0-255.
DCMotor_1->run(FORWARD);       // Set the motor status, FORWARD forward,
BACKWARD reverse, BRAKE stop.
```

# Stepper Motor

## Principle of stepper motor

Stepper motor is an open-loop control motor that converts electrical pulse signals into angular displacement or line displacement. It is the main actuator in modern digital program control systems and widely used. In the case of non-overloading, the speed and stop position of the motor only depends on the frequency of the pulse signal and the number of pulses.

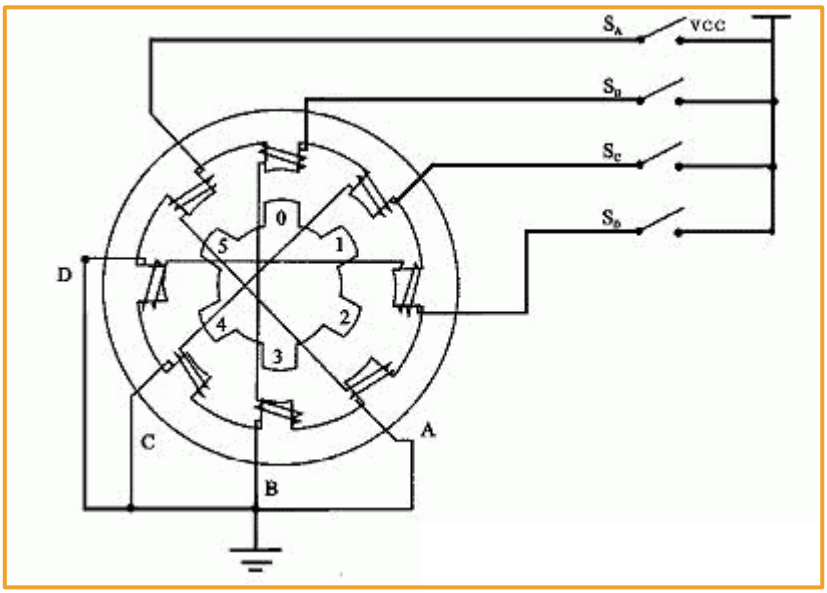



Figure 5-1 stepper motor schematic

As shown in Figure 5-1 above, there are multiple pairs of magnetic poles inside the stepping motor.If the power-on state remains unchanged, the stepper motor will remain in a fixed state. Only by changing the energization state of each pair of poles can the stepper motor continue to rotate. Therefore, the stepper motor cannot be directly connected to the DC or AC power supply, and a dedicated driving power supply (stepper motor driver) must be used. The controller (pulse signal generator) can control the angular displacement by controlling the number of pulses to achieve the purpose of accurate positioning. At the same time, the speed and acceleration of the motor rotation can be controlled by controlling the pulse frequency, thereby achieving the purpose of speed regulation.

## Driving stepper motor

Connect a stepper motor to port 1 and port 2 of the stepper motor as shown in Figure 2-1. The working voltage of the stepper motor is between 5~12V. If the working voltage of the stepper motor is too low or too high, there is a phenomenon that the burned motor or the drive board collapses, and the battery should be fully charged when driving the motor. After connecting the motor to the line (as shown in Figure 5-2), find

the stepper motor test sample program in the sample program (sample program file path: load file -> Lesson\AdvancedExperiment\Stepper\Stepper.ino). Burning the program into the development board, then turn on the power of the driver board, you will find the stepper motor rotates. The schematic diagram of the stepper motor is shown in Figure 4-4.

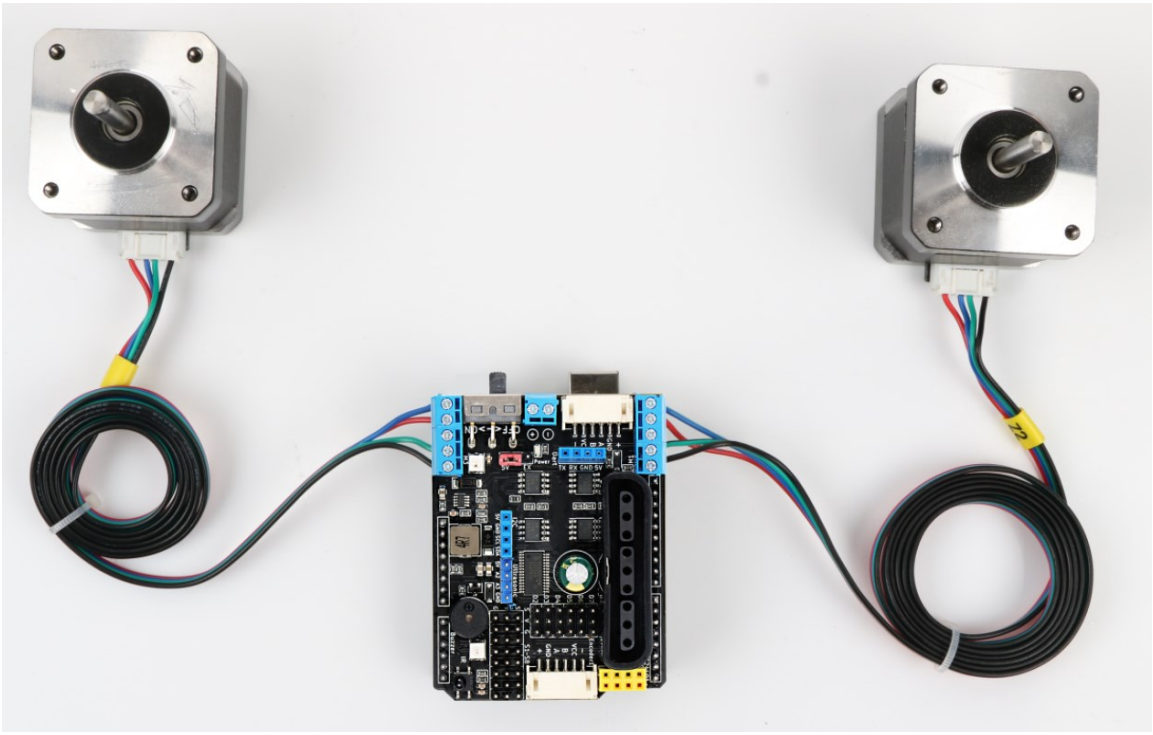


Figure 5-2 Stepper motor connection diagram

Program use

```
Emakefun_MotorDriver mMotorDriver
=Emakefun_MotorDriver(0x60,MOTOR_DRIVER_BOARD_V5);
// Initialize library, 0x60 is the I2C address of PCA9685, MOTOR_DRIVER_BOARD_V5 is
    MotorDrvierBoar_V5, if your MotorDriverBoard is V4 version, please change
    MOTOR_DRIVER_BOARD_V5 to MOTOR_DRIVER_BOARD_V4.
makefun_StepperMotor *StepperMotor_1 = mMotorDriver.getStepper(Steps,
Stepper);            // Initialize the stepper motor, Steps: the number
of steps required per turn, it is recommended to be before 35 ~ 200
StepperMotor_1->setSpeed(Speed);      // Set the rotation speed per
minute, Speed: the number of steps per minute rotation
StepperMotor_1->step(step, direction); // step: number of steps
direction: forward and reverse, FORWARD forward, BACKWARD reverse
StepperMotor_1->release();           //stop。
```

# Encode Motor

## Encoder Introduction

An encoder is a rotary sensor that converts angular or angular velocity into a series of electrical digital pulses. We can measure the displacement or velocity information through the encoder. The encoder is divided into output data types and can be divided into incremental encoders and absolute encoders. From the principle of encoder detection, it can also be divided into optical, magnetic, inductive and capacitive. Commonly used are photoelectric encoders (optical) and Hall encoders (magnetic) and our balanced car trials are Hall encoders.

## Encoder Principle

A Hall encoder is a sensor that converts the amount of mechanical geometric displacement on an output shaft into a pulse or digital quantity by magnetoelectric conversion. The Hall encoder consists of a Hall code disc and a Hall element. The Hall code disc is equidistantly arranged with different magnetic poles on a circular plate of a certain diameter. The Hall code disc is coaxial with the motor. When the motor rotates, the Hall element detects and outputs several pulse signals. In order to judge the steering, two groups of square wave signals having a certain phase difference are generally output. It can be seen that the encoders of both principles aim to obtain the square wave signal of the AB phase output, and the method of use is the same. The following is a simple schematic diagram.

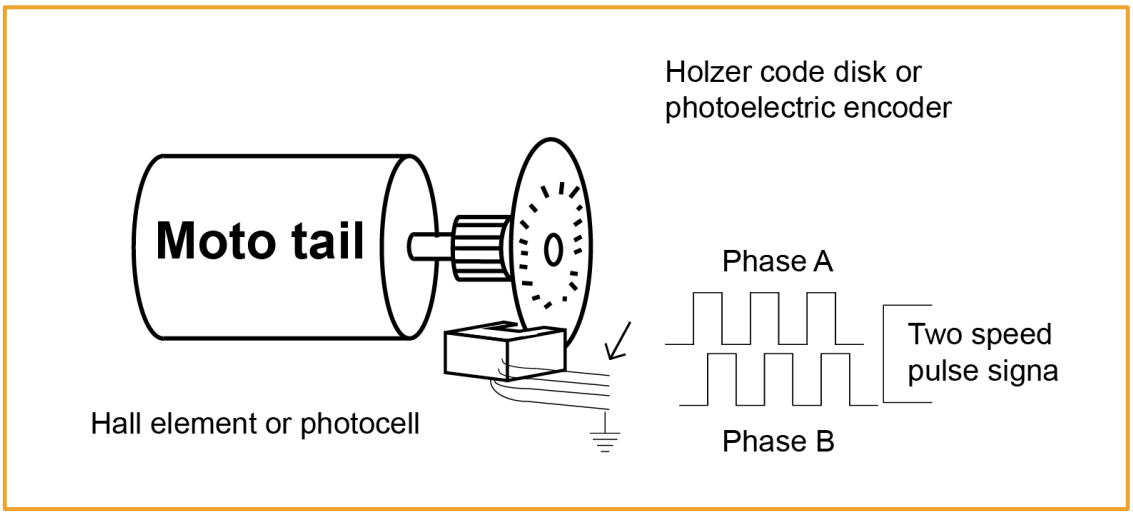



Figure 6-1 Schematic diagram of the Coded Motor

## Encoder wiring instructions

Specific to our encoder motor, we can look at the actual motor encoder. This is an incremental output Hall encoder. The encoder AB phase output, so not only the speed can be measured, but also the steering can

be discerned. According to the wiring instructions in Figure 6-2-4, we only need to supply 5V power to the encoder, and the square wave signal can be output through the AB phase when the motor rotates. The encoder comes with a pull-up resistor, so no external pull-up is required and can be directly connected to the microcontroller IO for reading.

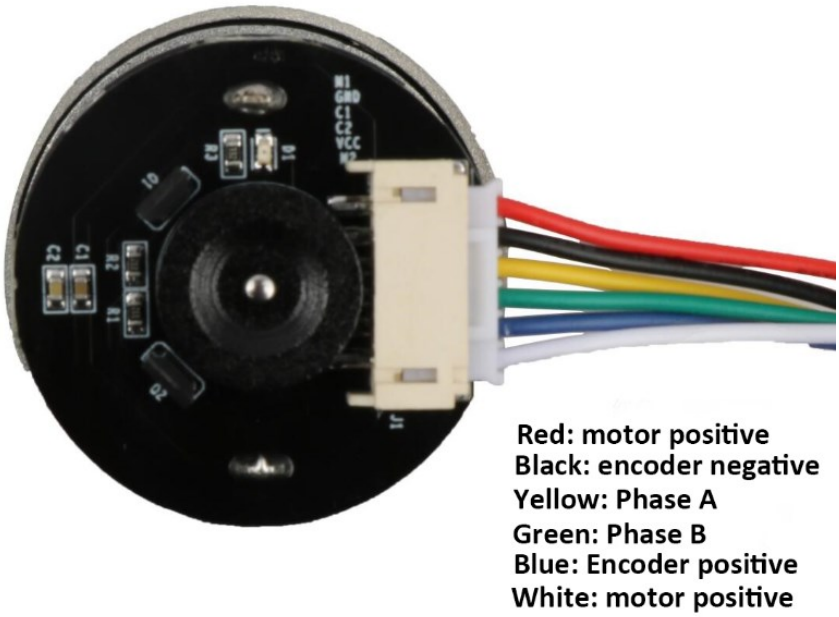



Figure 6-2 Encoder wiring diagram

## Drive Encode Motor

Connect a coding motor to port 1 and port 2 of the encoder motor as shown in Figure 8-3. The working voltage of the encoder motor is between 5 and 12V. After connecting the motor to the line, find the encoded motor test sample program in the sample program (sample program file path: load file -> **Lesson\AdvancedExperiment\Encoder\ Encoder.ino**), burn the program to the development board, and then turn on the driver. The power of the board will find that the two encoder motors will rotate synchronously. The schematic diagram of the encoder motor is shown in Figure 8-1.

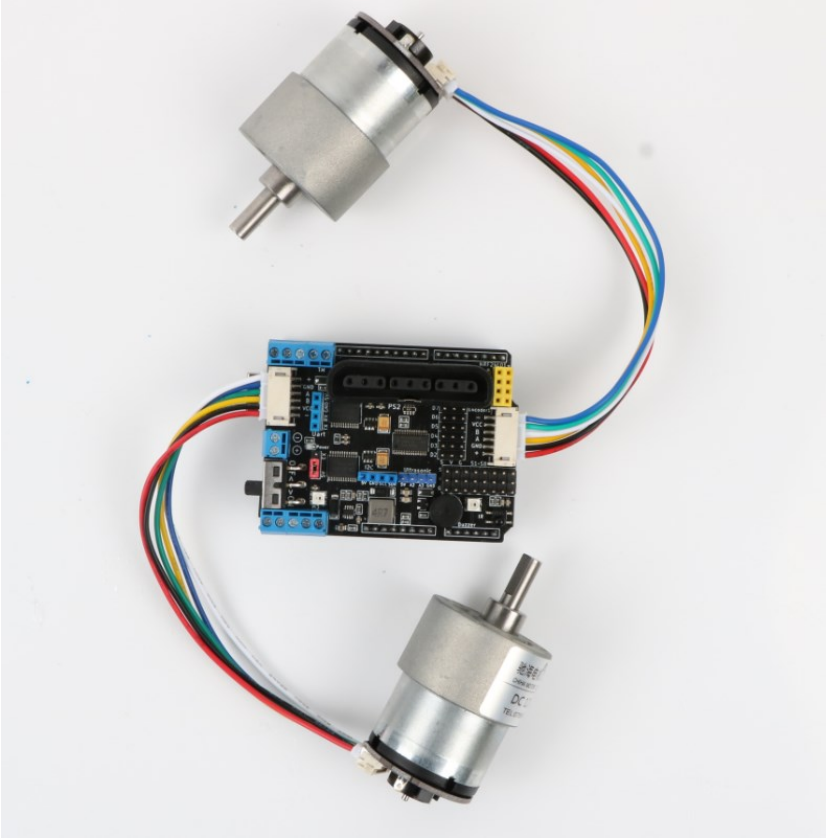


Figure6-3 Coded motor connection diagram

# Servo

## Servo Introduction

The steering gear is also called servo motor. It was first used to realize its steering function on the ship. Because it can continuously control its rotation angle through the program, it is widely used to realize the steering and the various joint movements of the robot. The steering gear is the control of the steering of the trolley. The mechanism has the characteristics of small size, large torque, simple external mechanical design and high stability. Whether in hardware design or software design, the steering gear design is an important part of the car control. Generally speaking, the steering gear is mainly composed of the following The components are composed of steering wheel, reduction gear set, position feedback potentiometer, DC motor, control circuit, etc., as shown in Figure 7-1 and Figure 7-2. Motor Driver Board uses 180 degree SG90 (180 degrees) 9g servo.

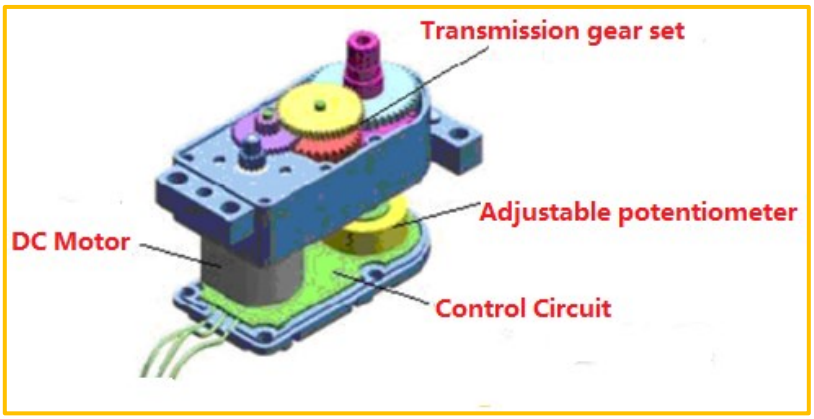



Figure7-1 Diagram of Servo

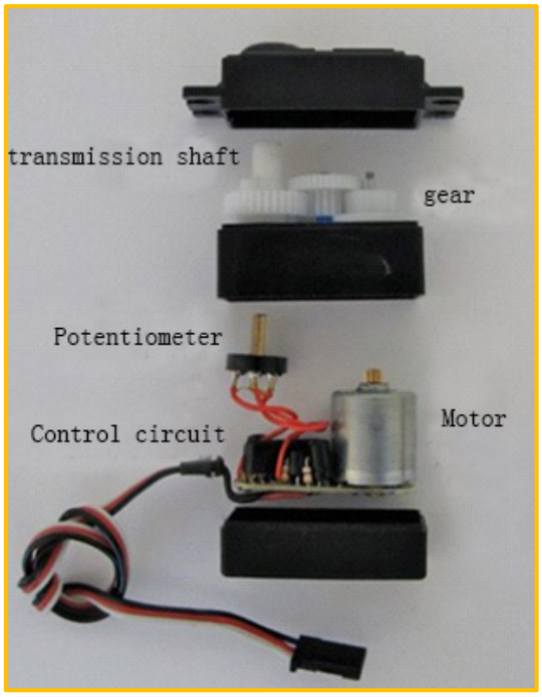



Figure 7-2 Physical image of steering gear composition

## Working principle of servo

The servo control signal enters the signal modulation chip from the channel of the receiver to obtain a DC bias voltage. It has a reference circuit inside, which generates a reference signal with a period of 20ms and a width of 1.5ms, and compares the obtained DC bias voltage with the voltage of the potentiometer to obtain a voltage difference output. Finally, the positive and negative voltage output of the voltage difference to the motor drive chip determines the forward and reverse of the motor. When the motor speed is constant,

the potentiometer is rotated by the cascade reduction gear, so that the voltage difference is 0, and the motor stops rotating.

When the control circuit board receives the control signal from the signal line, the motor is controlled to rotate, the motor drives a series of gear sets, and after deceleration, the drive is transmitted to the output steering wheel. The output shaft of the steering gear and the position feedback potentiometer are connected. When the steering wheel rotates, the position feedback potentiometer is driven. The potentiometer will output a voltage signal to the control circuit board for feedback, and then the control circuit board determines the motor according to the position. The direction and speed of rotation to achieve the target stop. The workflow is: control signal → control circuit board → motor rotation → gear set deceleration → steering wheel rotation → position feedback potentiometer → control circuit board feedback.

The control signal period of the servo is 20MS pulse width modulation (PWM) signal, the pulse width is from 0.5-2.5MS, and the corresponding steering wheel position is 0-180 degrees, which varies linearly. That is to say, give him a certain pulse width, its output shaft will maintain a certain corresponding angle, no matter how the external torque changes, until it is given a pulse signal of another width, it will change the output angle to the new corresponding position is shown in Figure 7-3. There is a reference circuit inside the steering gear to generate a reference signal with a period of 20MS and a width of 1.5MS. There is a comparator that compares the applied signal with the reference signal to determine the direction and size to produce the motor's rotation signal.

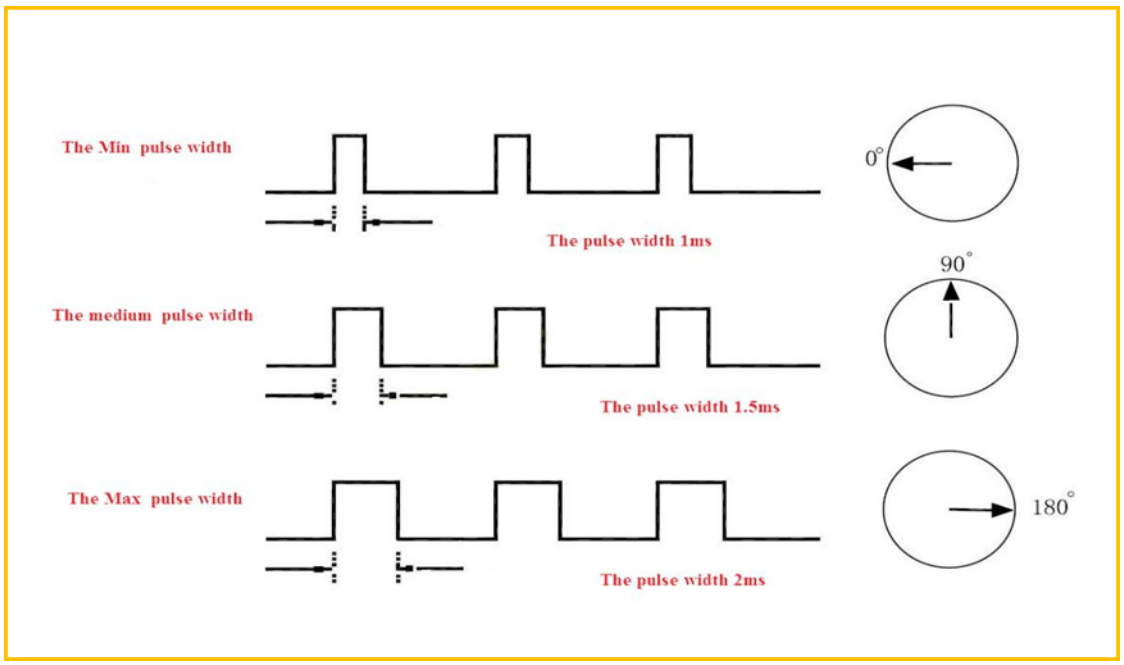



Figure 7-3 Relationship between servo output angle and input pulse

## Drive Servo

The PS2X&Motor Driver Board driver board can drive 8-way servos, the servo pin position on the driver board (pins above the IR receiver), ground pins (G), power pins (V) and signal pins ( S), the three pins respectively correspond to the corresponding pins of the servo (Fig. 7-4), and can also control the servo through I2C communication, as shown in Figure 7-5, (Example program file path: Load file -> **Lesson\AdvancedExperiment\Servo\ Servo.ino**) After burning the sample program, turn on the power of the expansion board, and you can find that the eight servos rotate to 0°, 90°, and 180° at the same time. We can realize the steering of the robot by driving the rotation of the steering gear, or install the sensor probe on the steering gear, so as to adjust the detection direction of the sensor through the steering gear. The schematic diagram of the servo interface is shown in Figure 7-6.

Figure 11-4 MG90 Servo machine physical map

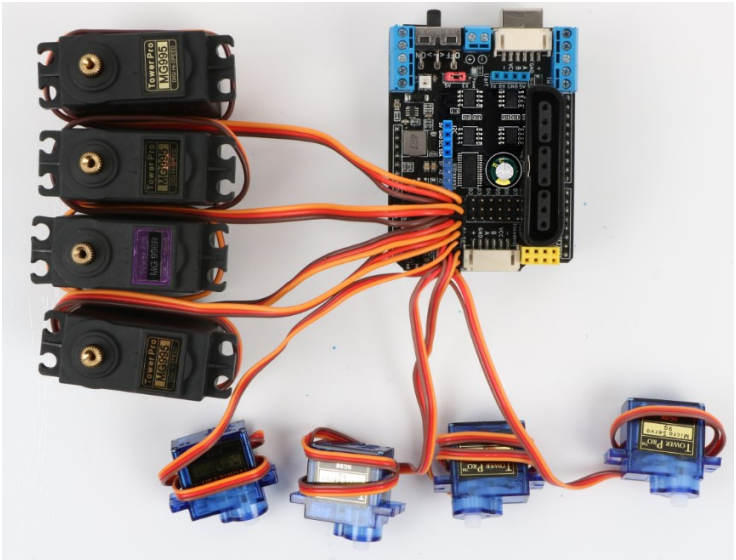


Figure 7-5 Drive Servo connection diagram

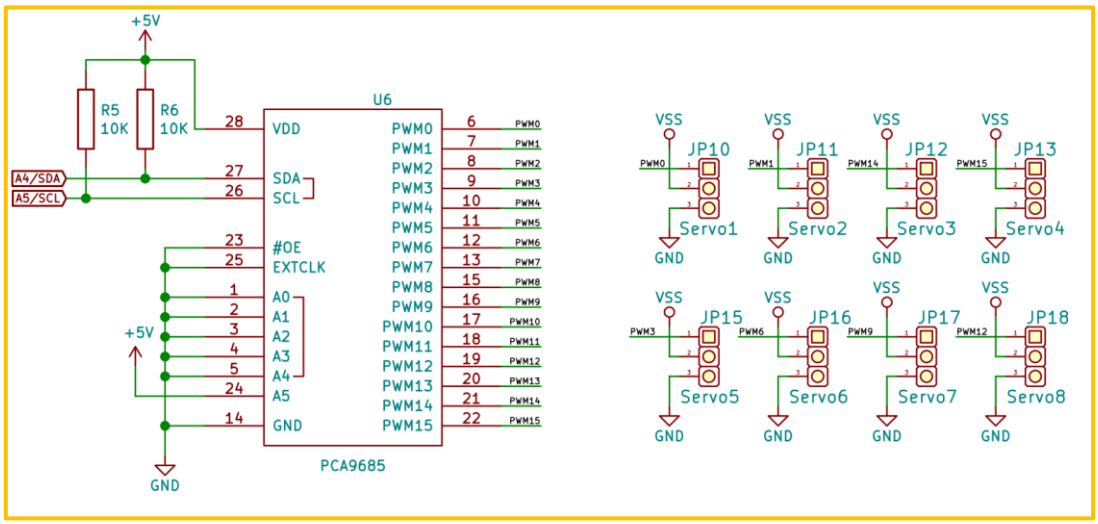



Figure 7-6

Program use

```
Emakefun_MotorDriver mMotorDriver
=Emakefun_MotorDriver(0x60,MOTOR_DRIVER_BOARD_V5);
// Initialization library, 0x60 is the I2C address of PCA9685, MOTOR_DRIVER_BOARD_V5
   is MotorDrvierBoar_V5, if your MotorDriverBoard is V4 version, please change
   MOTOR_DRIVER_BOARD_V5 to MOTOR_DRIVER_BOARD_V4.
Emakefun_Servo *mServo1 = mMotorDriver.getServo(1);// Initialize I2C
servo, 1-8 are I2C servo interface.
mServo1->writeServo(num);   // The steering gear rotates to num degrees,
num: 0 ~ 180 degrees.
```

## How to use an external power supply to drive the servo

The PS2X&Motor Driver Board can not only power the steering gear through the power supply chip to drive the steering gear, but also power the steering gear through an external power supply to drive the steering gear. The voltage of the main control chip of PS2X&Motor Driver Board is generally 5V. When driving the servo, although it can also be driven, some servos will have a rated voltage higher than 5V. If the voltage is supplied with 5V, the steering wheel will rotate. If the force is too small, the accuracy will be affected. If the external voltage is greater than 5V (5.2-7.2V), then the steering gear will be driven, obviously the speed of the steering gear will be much faster, and the force during rotation will be very high. Strong and accurate.

The specific steps are as follows:

1) Short the power supply pin EX pin on the PS2X&Motor Driver Board through the jumper cap;

2) Connect the external power supply to the power connector on the PS2X&Motor Driver Board;

3) Turn the toggle switch to OFF to supply power to the entire board through external power supply.

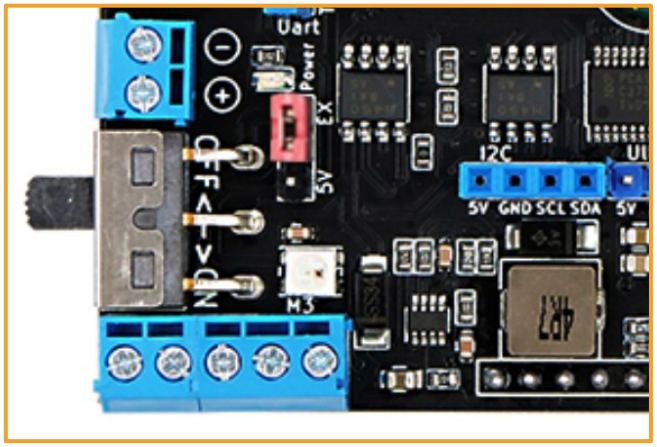



Figure 7-7

# RGB LED light

## RGB WS2812 Introduction

The WS2812 RGB LED lamp is a three-channel drive control. It contains an intelligent digital interface data latch signal shaping and amplifying driver circuit. It also includes a high-precision internal oscillator and a 15V high-voltage programmable constant current output driver. At the same time, in order to reduce the power supply ripple, the three channels have a certain delay turn-on function, which can reduce the circuit ripple installation more easily during frame refresh.

Unlike the traditional RGB LED lamps, the WS2812 RGB LED lamps integrate a dedicated driver control chip to control one LED bead or multiple LED modules with a single signal line. Its main features are as follows:

- Output port withstand voltage 15V
- Built-in voltage regulator tube, only 24V power supply terminal only needs string resistor to IC VDD pin, no need to add regulator tube
- Grayscale adjustment circuit (256 levels of grayscale adjustable)
- Built-in signal shaping circuit, after any IC receives the signal, the waveforms of other various LED lighting products are shaped and re-output to ensure that the waveform distortion of the line will not be accumulated.
- Built-in power-on reset and power-down reset circuit
- The PWM control terminal can achieve 256 levels of adjustment, and the scanning frequency is not less than 400Hz/s.
- Serial interface level connection port, which can receive and decode data through one signal line
- Any two points transmission distance is more than 10 meters without adding any circuit

- When the refresh rate is 30 frames/second, the cascading number of the low-speed mode is not less than 512 points, and the high-speed mode is not less than 1024 points.
- Data transmission speed is 800Kbps mode

## Working principle of WS2812 RGB LED Light

WS2812 RGB LED light data protocol adopts single-line return-to-zero code communication mode. After the power-on reset, the DIN terminal accepts the data transmitted from the controller. The first 24 bits of data sent by the first pixel are extracted and sent. The data latch inside the pixel, the remaining data is shaped and amplified by the internal shaping processing circuit, and then the output is forwarded to the next cascaded pixel through the DO port, and the signal is reduced by 24 bits after each pixel transmission. The pixel adopts automatic shaping and forwarding technology, so that the number of cascaded pixels is not limited by signal transmission, and only the signal transmission speed is limited.

The data latch inside the chip generates different duty cycle control signals at the OUTR, OUTG, and OUTB control terminals according to the received 24bit data. When the DIN terminal inputs the RESET signal, all the chips synchronously send the received data to each section. The chip will re-accept the new data after the end of the signal. After receiving the first 24bit data, the data port is forwarded through the DO port. Before the chip receives the RESET code, the original output of the OUTR, OUTG, and OUTB pins remains unchanged. After receiving a low level RESET code of 50μs or more, the chip outputs the pulse width of the 24bit PWM data just received to the OUTR, OUTG, and OUTB pins. The pin and function of the chip are shown in Figure 8-1 and Table 1.

| No | Symbol | Pin name | Function description |
|---|---|---|---|
| 1 | VDD | Logic power | IC Power Supply |
| 2 | OUT | LED Drive output | Display data cascade output |
| 3 | IN | LED Drive input | Display data input |
| 4 | VSS | Ground | Ground, finally cascaded |

Table 1 WS2812 chip pin function table

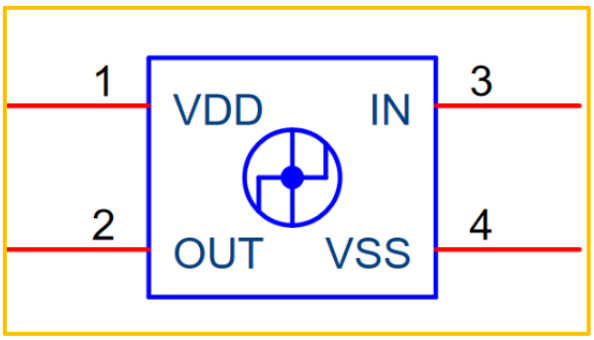



Figure 8-1 Schematic diagram of the pin function of the WS2812 chip

## WS2812 RGB LED Light Driving principle

The WS2812 RGB LED low level is represented by T0, which is 0.5μs high level 2μs low level. T1 consists of a low level of 2μs and a low level of 0.5μs. Low level time above 50μs during reset.

| | | |
|---|---|---|
| T0H | 0 yards, high voltage time | 0. 5μs |
| T1H | 1 yard, high voltage time | 2μs |
| T0L | 0 yards, low voltage time | 2μs |
| T1L | 1 yard, low voltage time | 0. 5μs |
| RES | Reset code, low voltage time | >50μs |

The timing waveform diagram is as follows

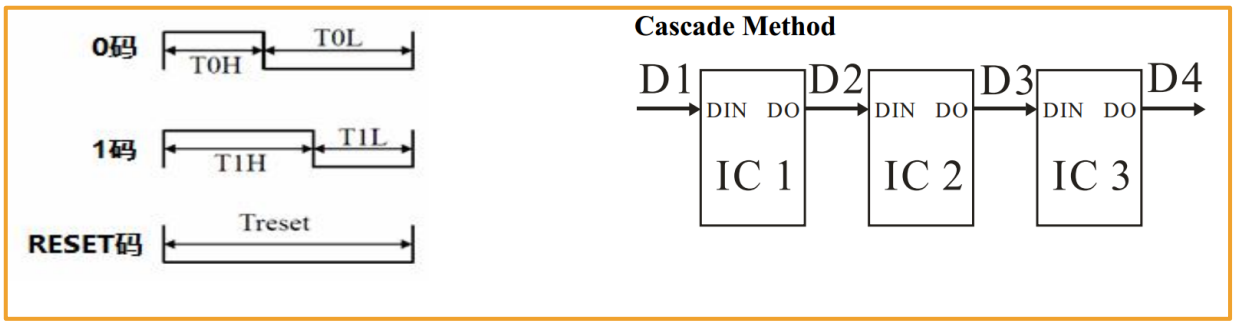



Figure 8-2 Waveform timing diagram and connection method

**24bit** data structure

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

## Drive RGB LED light

There are two RGB LED lights RGB1 and RGB2 on the PS2X&Motor Driver Board. You can control the RGB light-off timing by programming. You can also set the color of the RGB light (example program file path: load file -> **Lesson\AdvancedExperiment\RGB\ RGB.ino**) After burning the sample program, turn on the power switch and you will see the RGB light flashing. The schematic diagram of the RGB LED light is shown in Figure 8-3.

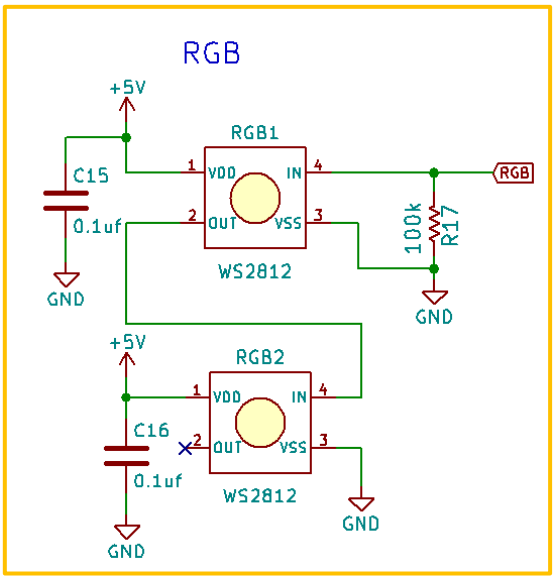



Figure8-3 RGB LED schematic

Program use

```
Emakefun_MotorDriver mMotorDriver
=Emakefun_MotorDriver(0x60,MOTOR_DRIVER_BOARD_V5);
// Initialization library, 0x60 is the I2C address of PCA9685, MOTOR_DRIVER_BOARD_V5
   is MotorDrvierBoar_V5, if your MotorDriverBoard is V4 version, please change
   MOTOR_DRIVER_BOARD_V5 to MOTOR_DRIVER_BOARD_V4.
RGBLed *rgb = mMotorDriver.getSensor(E_RGB);   // Initialize RGB.
rgb->SetRgbColor(E_RGB_INDEX, colour);   // E_RGB_INDEX: E_RGB_ALL: two
RGB E_RGB_RIGHT: right RGB; E_RGB_LEFT: left RGB Colour: display color.
rgb->setBrightness(num);       // Brightness adjustment, num: 0 ~ 255.
```

# Buzzer

## Buzzer Introduction

The buzzer is an integrated structure electronic sounder that is powered by a DC voltage and is used in electronic products as a sounding device. The buzzer is mainly divided into two types: active buzzer and passive buzzer. The main difference between the two is as follows:

The ideal signal for active buzzer operation is direct current, usually labeled VDC, VDD, and so on. Because the buzzer has a simple oscillating circuit inside, it can drive the molybdenum sheet to vibrate as long as it is energized. However, some active buzzers can work under certain AC signals, but the voltage and frequency of the AC signal are very high. This kind of scene is rare.

There is no oscillation circuit inside the passive buzzer, and the working signal is a pulse signal of a certain frequency. If the passive buzzer for the DC signal is not responsive, the molybdenum sheet cannot vibrate because the magnetic path is constant.


Active buzzer Passive buzzer

Figure 9-1 Physical diagram of active buzzer and passive buzzer

## Working principle of buzzer

The passive buzzer generates music mainly by controlling the buzzer sound by outputting high and low pulse signals from the I/O port of the single chip microcomputer. To generate an audio pulse signal, it is necessary to calculate the period (1/frequency) of an audio, and then Dividing this period by 2 is the half-cycle time. The time of this half cycle is counted by the single-chip timer, and the I/O port of the output pulse is inverted every time the timer is counted, so that the frequency of the pulse is obtained on the I/O port.

For example, if the Arduino uses a 12MHzs crystal, the sound of the mid-tone Re is required to output an audio pulse frequency of 587Hzs. The pulse period of the audio signal is T=1/587=1703.5775us, and the half-cycle time is 852us. The device count = 852us / 1us = 852, the I / O port is inverted at 852 every count, the middle adjustment Re of C major is obtained.

In addition, the passive buzzer sounds the principle that the current passes through the electromagnetic coil, causing the electromagnetic coil to generate a magnetic field to drive the diaphragm to sound. Therefore, a certain current is required to drive it, and the voltage output from the Arduino I/O pin is small. The level of the Arduino output basically does not drive the buzzer, because an amplifier circuit needs to be added. Here, the transistor SS8050 is used as an amplifying circuit.

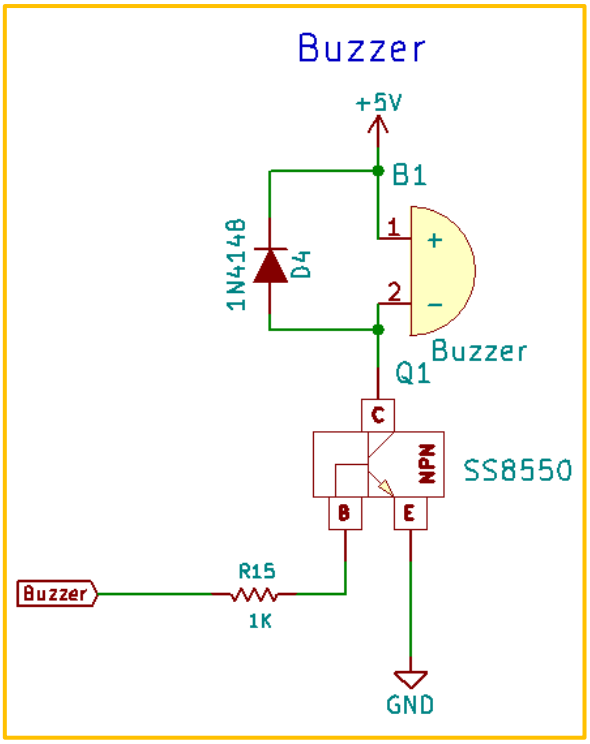



Figure 9-2 Buzzer schematic

## Drive buzzer

The PS2X&Motor Driver Board has a passive buzzer on it that can be programmed to control the buzzer to play a tone or play music (example program file path:

Load File-> **Lesson\AdvancedExperiment\Buzzer\Buzzer.ino**) After burning the sample program, turn on the power switch and the buzzer plays music. The schematic diagram of the buzzer is shown in Figure 9-2. Show.

### Program use

```
Emakefun_MotorDriver mMotorDriver
=Emakefun_MotorDriver(0x60,MOTOR_DRIVER_BOARD_V5);
// Initialization library, 0x60 is the I2C address of PCA9685, MOTOR_DRIVER_BOARD_V5
   is MotorDrvierBoar_V5, if your MotorDriverBoard is V4 version, please change
   MOTOR_DRIVER_BOARD_V5 to MOTOR_DRIVER_BOARD_V4.
Emakefun_Sensor *buzzer = mMotorDriver.getSensor(E_SENSOR_MAX);
mMotorDriver.getSensor(E_BUZZER);   // Initialize buzzer.
buzzer->Sing(S_connection);     // Play sound effects, there are many
    kinds of sound effects, all sound effects in the "Sounds.h" file
    beginning with the macro definition of S are sound effects.
```

# Ultrasonic obstacle avoidance module

## Working Principle of Ultrasonic module

The ultrasonic transmitter emits ultrasonic in a certain direction, and starts timing at the same time as the transmission time. When the ultrasonic travels in the air, it will return immediately when it hits an obstacle. And the ultrasonic receiver stops the timing immediately upon receiving the reflected waves. The propagation speed of the ultrasonic in the air is V, and according to the time difference Δt of the measured transmitted and received echo recorded by the timer, the distance S of the emission point from the obstacle can be calculated, that is:

$$S = V \cdot \Delta t / 2$$

## Drive Ultrasonic Module

The PS2X&Motor Driver Board has a four-wire ultrasonic module interface on the driver board. The four pins are the power supply pin (vcc), the ultrasonic signal transmission pin (A1), the ultrasonic signal receiving pin (A1), and the ground (GND). The four pins are connected to the corresponding pins of the ultrasonic module (as shown in Figure 10-1), as shown in Figure 10-2. (Example program file path: **Load file -> Lesson\AdvancedExperiment\Ultrasonic\Ultrasonic.ino**) After burning the sample program, turn on the power switch and turn on the serial monitor. You will see the distance of the ultrasonic probe continuously printed on the serial monitor. The schematic diagram of the ultrasonic obstacle avoidance module interface is shown in Figure 10-3.

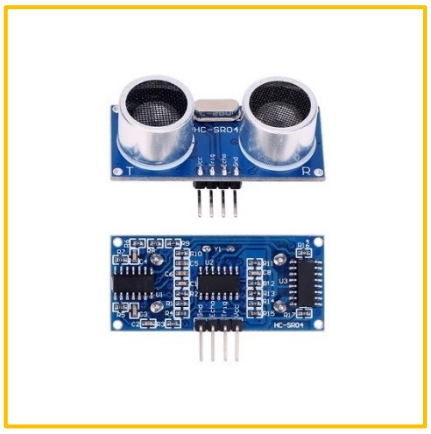


Figure 10-1 Physical diagram of the ultrasonic module

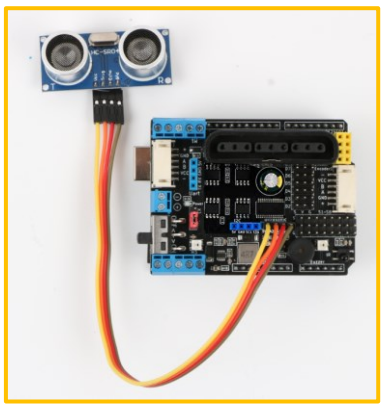


Figure 10-2 Ultrasonic module connection diagram

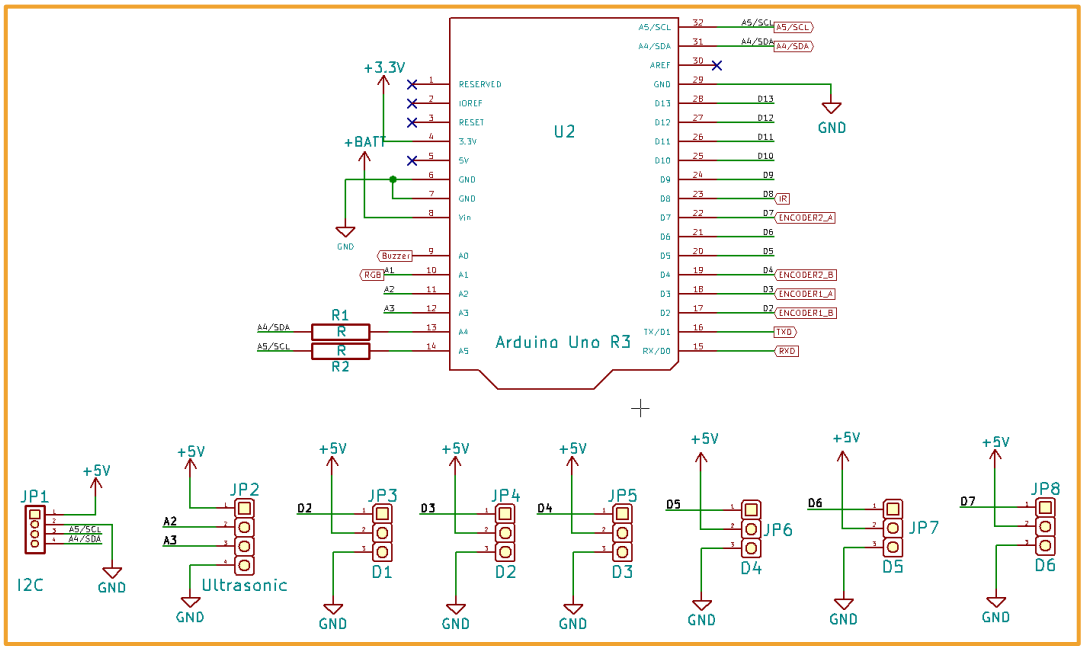



Figure 10-3 Schematic diagram of ultrasonic connection

Program use

```
Emakefun_MotorDriver mMotorDriver
=Emakefun_MotorDriver(0x60,MOTOR_DRIVER_BOARD_V5);
// Initialization library, 0x60 is the I2C address of PCA9685, MOTOR_DRIVER_BOARD_V5
   is MotorDrvierBoar_V5, if your MotorDriverBoard is V4 version, please change
   MOTOR_DRIVER_BOARD_V5 to MOTOR_DRIVER_BOARD_V4.
Emakefun_Sensor *ult = mMotorDriver.getSensor(E_SENSOR_MAX);
//Initialize sensor.
mMotorDriver.getSensor(E_ULTRASONIC); // Initialize ultrasonic ECHO to
A3 TRIG to A2
ult->GetUltrasonicDistance();  // Read ultrasonic distance
```

## Infrared Remote Control

Infrared wireless remote control consists of Mini ultra-thin infrared remote control (shown in Figure 11-1) and integrated 38KHz infrared receiving head. Mini ultra-thin infrared remote control has 17 function keys, the transmission distance can reach 8 meters, which is very suitable for Control a variety of equipment indoors.

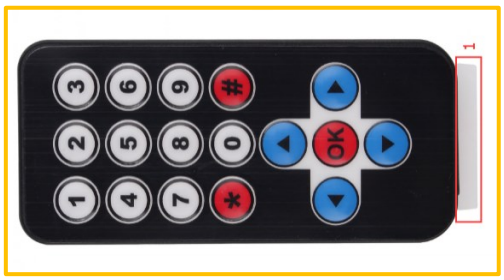



Figure 11-1 Infrared remote control physical map

In the Motor Driver Board of the racing car,the integrated infrared receiver has been added to the expansion board. When it is used, it only needs to be inserted into the Arduino. When there is an infrared coded signal transmission, after being processed by the infrared connector, the output is a square wave signal after the detection and shaping, and is directly supplied to the single chip microcomputer, and the corresponding operation is performed to achieve the purpose of controlling the motor.

## Working Principle

The remote control system generally consists of a remote control (transmitter) and a receiver. When you press any button on the remote control, the remote control generates a corresponding code pulse to output various infrared-based control pulse signals. The pulse is a computer command code. The infrared monitor diode monitors the infrared signal and then sends the signal to the amplifier and the limiter. The limiter controls the pulse amplitude to a certain level regardless of the distance between the infrared transmitter and the receiver. The AC signal enters the bandpass filter. The bandpass filter can pass the load wave of 30KHZ to 60KHZ, enter the comparator through the demodulation circuit and the integration circuit, and the comparator outputs high and low levels to restore the signal waveform of the transmitter. As shown in the 11-2 system fram diagram.

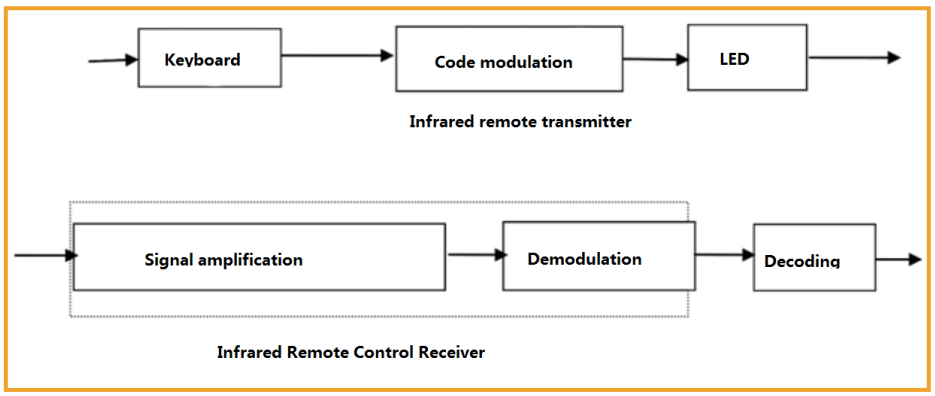



Figure 11-2 Block diagram of the infrared transmitting and receiving system

## Driving infrared remote control

There is an infrared remote control receiving probe on the PS2X&Motor Driver Board (as shown in Figure 11-3). (Example program file path: Load file -> **Lesson\AdvancedExperiment\IrkeyPressed \ IrkeyPressed.ino**) After burning the sample program, turn on the power. Switch, and open the serial monitor, use the remote control to press the different buttons on the infrared receiver on the driver board, you will see different numbers printed on the serial monitor, and the receiver will light red when receiving the infrared signal.Of course, you can also write a program that uses infrared remote control to control the rotation of the motor or the rotation of the servo. It is more intuitive to see the effect of the infrared remote control. The schematic diagram of the infrared remote control is shown in Figure 11-5.


Figure 11-3 Infrared receiver head physical map

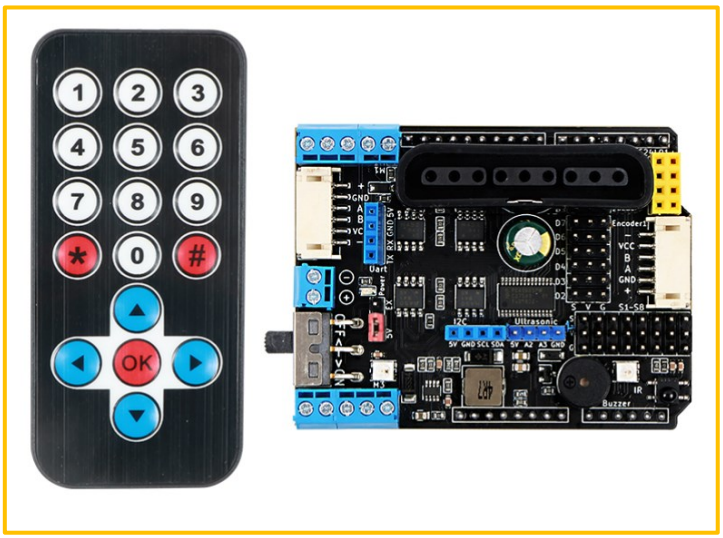



Figure 11-4 Infrared remote control

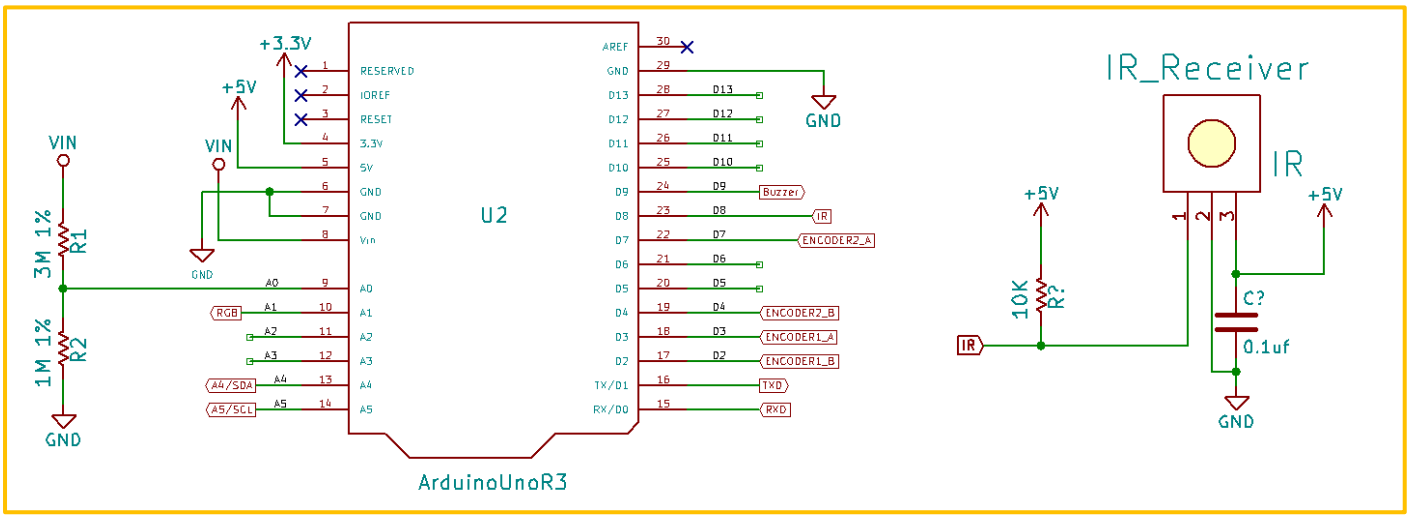



Figure 11-5 Schematic diagram of the infrared receiver connection

### Program use

```
Emakefun_MotorDriver mMotorDriver
=Emakefun_MotorDriver(0x60,MOTOR_DRIVER_BOARD_V5);
// Initialization library, 0x60 is the I2C address of PCA9685, MOTOR_DRIVER_BOARD_V5
    is MotorDrvierBoar_V5, if your MotorDriverBoard is V4 version, please change
    MOTOR_DRIVER_BOARD_V5 to MOTOR_DRIVER_BOARD_V4.
IRremote *ir = mMotorDriver.getSensor(E_IR);  //Initialize the infrared
receiver.
irKeyCode = ir->getCode();  // Get the value of the pressed key
(E_NORMAL_IR_KEYCODE)ir->getIrKey(irKeyCode, IR_TYPE_NORMAL)  // Match
the obtained key value and return the key name
```

# PS2 Remote Control

## PS2 Introduction

The PS2 handle is the remote control handle of Sony game machine, Sony series game host is in the world very popular. The outstanding characteristic is that this kind of handle price is extremely high now. Key-rich, easy to extend to other applications, such as the picture 12-1 is a commonly used PS2 wireless handle.


Figure 12-1 PS2 Wireless Handle

PS2 handle is composed of two parts of the handle and receiver, the handle needs two section 7th 1.5V power supply, the receiver's power supply and controller use the same power supply, the power supply range is 3~5v, can not be reversed or exceed voltage. Overvoltage and reverse connection will cause the receiver to burn out. There is a power switch on the handle, on the open/off off, the handle switch to on.In the case of no search to the receiver, the lamp on the handle flashing constantly, in a certain period of time, has not been found in the receiver, the handle will enter Standby mode, the handle on the lights will be extinguished, at this time, press the "START" button, wake-up handle.

The receiver is connected to the Arduino, and is powered by Arduino. In an unpaired condition, with a green light flashing. The handle is open, the receiver is powered, the handle and receiver will automatically pair, when the lamp is always bright, the handle pair succeeds. The key "mode" (handle batch is different, the above identification may be "analog", but will not affect the use), you can choose "Red light Mode", "Green mode".

There are 9 interfaces at the end of the receiving head, each of which is 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: The batch is different, the appearance of the receiver will be a difference, one on the power light red light, a power supply lights, but the use of the same method, pin definition is the same.

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 and DI relative, 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.

## Drive PS2 remote

There is a PS2 port on the PS2X&Motor Driver Board. The PS2 infrared receiver can be directly plugged into the PS2 port. After the PS2 receiver is properly inserted (as shown in Figure 11-1), the sample program starts to be burned. (Example program file Path: Load File -> **Lesson\AdvancedExperiment\PS2X\PS2X.ino**) After burning the sample program, turn on the power switch and turn the handle switch to ON. When the receiver is not searched, the light on the handle will flash continuously. The receiver has not been searched for a certain period of time. It will enter standby mode, the light on the handle will be extinguished, then press the "START" button to wake up the handle. The handle and the receiver are automatically paired. When the pairing is not successful, the green light of the receiver flashes, and the light on the handle will also flash. After the pairing is successful, the green light on the receiver is always on, and the light on the handle is always on. Press the button “MODE”. "(The handle batch is different, the above logo may be "ANALOG", but it will not affect the use), you can choose "red light mode", "green light mode". After the connection is successful, open the serial monitor, use the handle remote control to press the different buttons on the drive board, you will see the print content on the serial monitor. Of course, you can also write a program that uses PS2 to control the rotation of the motor or the rotation of the servo. It is more intuitive to see the effect of the PS2 remote control. The schematic diagram of the PS2 remote control is shown in Figure 12-3.。

Figure 12-2 PS2 Control

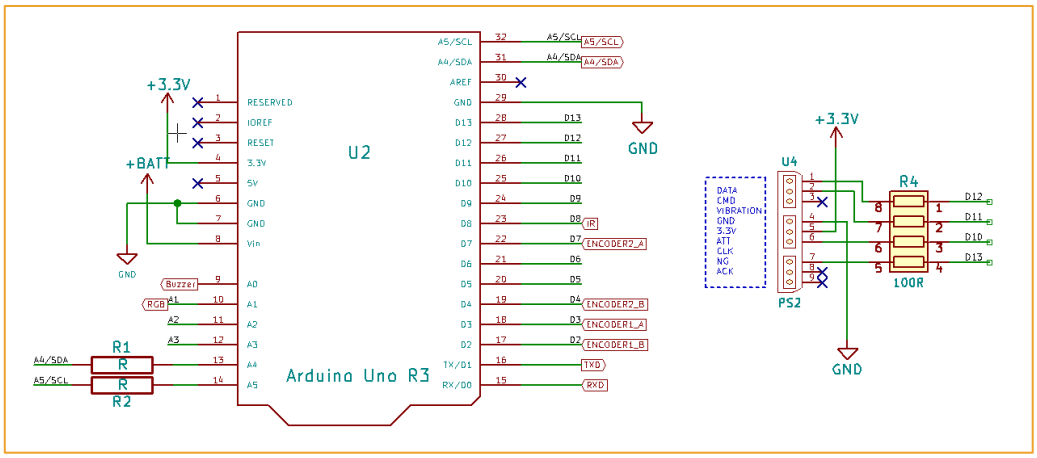



Figure 12-3 Schematic diagram of the PS2 receiver

## Program use

```
Emakefun_MotorDriver mMotorDriver
=Emakefun_MotorDriver(0x60,MOTOR_DRIVER_BOARD_V5);
// Initialization library, 0x60 is the I2C address of PCA9685, MOTOR_DRIVER_BOARD_V5
is MotorDrvierBoar_V5, if your MotorDriverBoard is V4 version, please change
MOTOR_DRIVER_BOARD_V5 to MOTOR_DRIVER_BOARD_V4.
mMotorDriver.getSensor(E_PS2X); // Initialize PS2.
ps2x->ButtonDataByte();  // Whether a key has been pressed and returns false or true
ps2x->Button(PSB_PAD_UP);   // Whether PSB_PAD_UP button is pressed, return false or
true
ps2x->RightHart();       // Read the value of the right joystick angle and return 0 ~
                            360 degrees, or 0XFFFF if there is no joystick;
ps2x-> LeftHart ();       // Read the left joystick angle value and return 0 ~ 360
                            degrees, if there is no joystick, it will return 0XFFFF;
```

# NRF2401

## NRF24L01+ module Introduction

The nRF24L01+ module(as shown in Figure 13-1) is a 2.4G wireless communication module developed by Nordic based on the nRF24L01 chip. Adopt FSK modulation and integrate Nordic's own Enhanced Short Burst protocol. Point-to-point or 1-to-6 wireless communication can be achieved. Wireless communication speed can reach up to 2M (bps). The NRF24L01 has four operating modes: transceiver mode, configuration mode, idle mode, and shutdown mode.

## Drive NRF24L01+ module

Insert the nRF24L01+ module into the corresponding interface on the PS2X&Motor Driver Board driver board (as shown in Figure 13-2). For the stable reception of Nrf24L01 data, it is recommended to connect 10uf capacitor between VCC and GUD as shown in Figure 13-3. **(Sample program file path: Receive load file -> Lesson\AdvancedExperiment\nRF24l01+\Receive \Receive.ino Send load file -> Lesson\AdvancedExperiment\nRF24l01+\Emitter\Emitter.ino)**, which can send and receive data tests to each other through two devices.

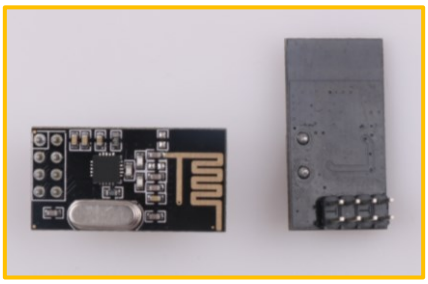


Figure 13-1: Nrf24l01+ physical map

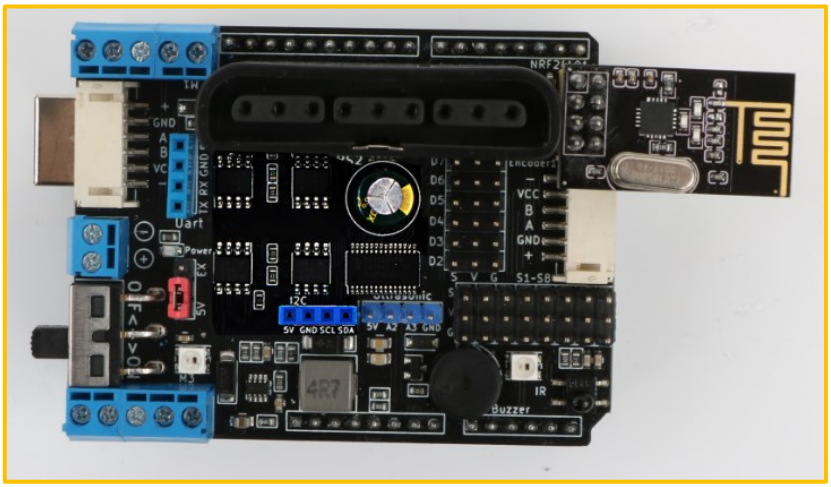


Figure 13-2 Connection diagram of Nrf24l01+

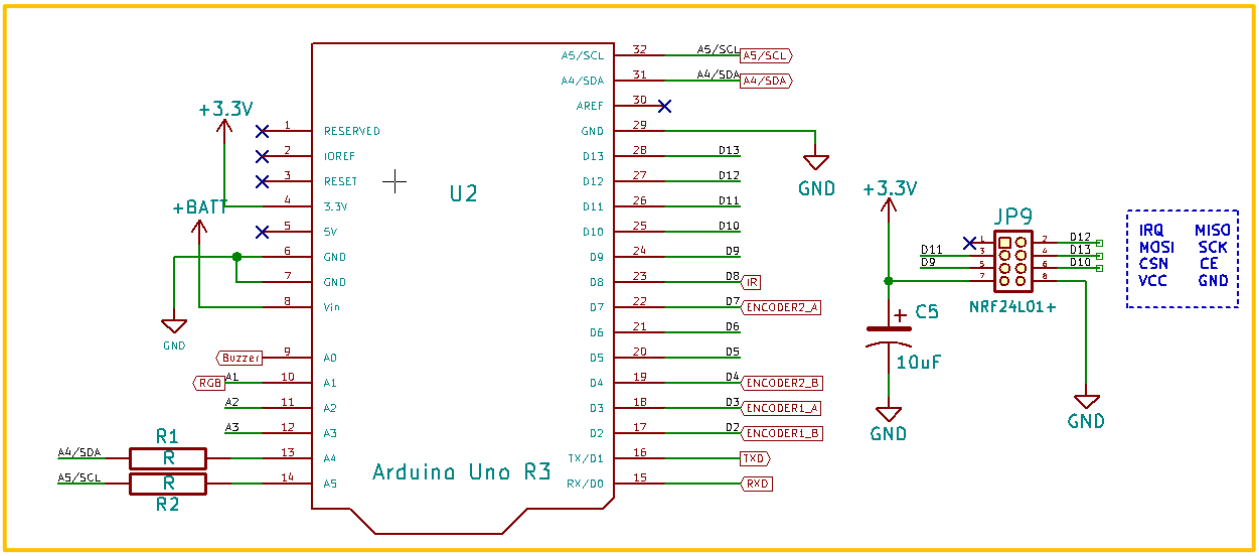



Figure 13-3 Schematic diagram of the Nrf24l01+ connection

## Program use

```
Emakefun_MotorDriver mMotorDriver
=Emakefun_MotorDriver(0x60,MOTOR_DRIVER_BOARD_V5);
// Initialization library, 0x60 is the I2C address of PCA9685, MOTOR_DRIVER_BOARD_V5
   is MotorDrvierBoar_V5, if your MotorDriverBoard is V4 version, please change
   MOTOR_DRIVER_BOARD_V5 to MOTOR_DRIVER_BOARD_V4.
Emakefun_Sensor *Nrf = mMotorDriver.getSensor(E_SENSOR_MAX);
mMotorDriver.getSensor(E_NRF24L01); // Initialize Nrf24L01+.
Nrf->sendNrf24l01((char *)"add",value);  // add: Receiver address value:
Data sent.
NrfData = Nrf->GetNrf24L01((char *)"add");  // add: Set your own address
and return the received data.
```

# Robotic arm

## Robot arm wiring

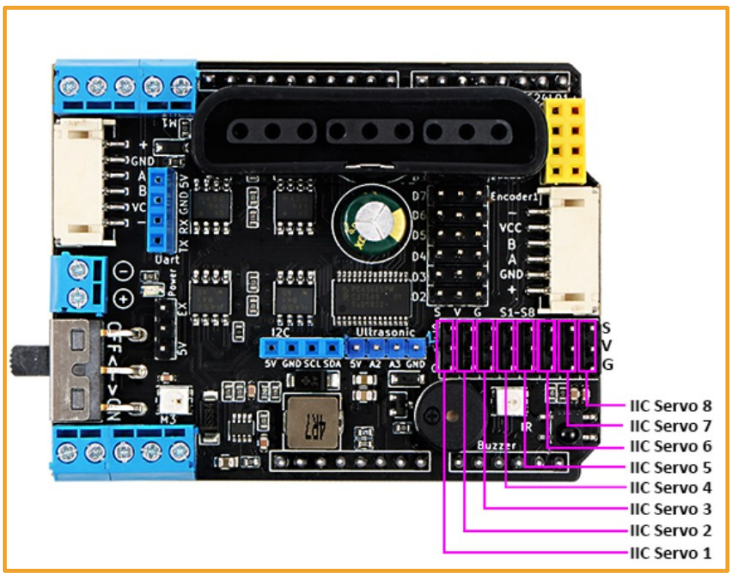



Figure 14-1

The arm of the arm is connected to the IIC servo 1, the left steering gear of the robot arm is connected to the IIC steering gear 2, the right steering gear is connected to the IIC steering gear 3, and the clip steering gear is connected to the IIC steering gear 4.



Figure 14-2

Left rocker 0 – 90°, 270 - 360° to control the IIC servo 1 The angle is slowly reduced, and 91–269° is used to control the IIC steering angle slowly.

Right rocker 0 – 44°, 315 - 360° for controlling IIC steering gear 2 angle slowly decreasing, 135 – 225 degrees for controlling IIC steering gear 2 angle slowly increasing, 45 – 134° for controlling IIC steering gear 3 angle Slowly increase, 225 – 314° control IIC servo 3 angle slowly decreases.

L1 is to increase the control IIC servo 4, and R1 is to control the IIC servo 4 to slowly decrease.

Note: Because of the mechanical arm structure problem, the left and right steering gears have a travel limit. If you are not using well, you can ask the technical support personnel how to adjust. If you understand, you can try it yourself.

## Bluetooth control by robotic arm

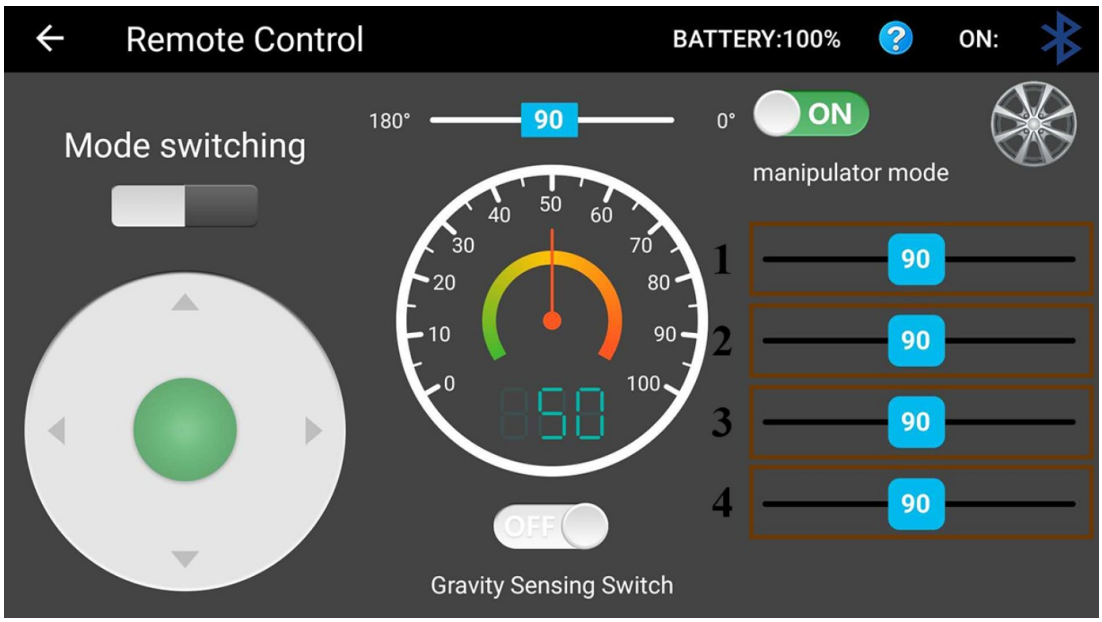
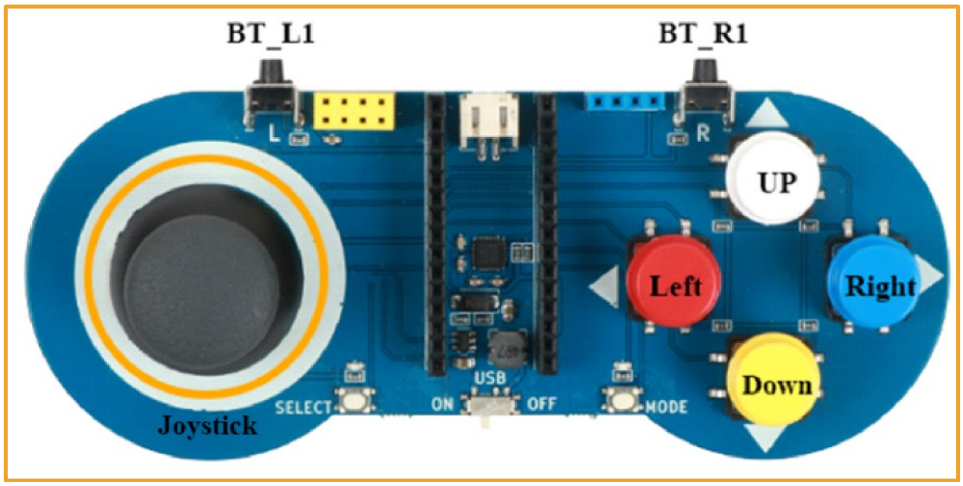



Figure 14-3

1 is sending angle to IIC Servo 1, 2 is sending angle to IIC Servo 2, 3 is sending angle to IIC Servo 3, and 4 is sending angle to IIC Servo 4.

## Robotic arm wireless handle control




Joysticks 0-90 ° and 270-360 ° are used to control the IIC Servo 1 and the angle is gradually decreased, and 91-269 ° are used to control the IIC Servo to increase the angle slowly.

Left controls the angle of IIC Servo 2 to increase slowly.

Right controls IIC Servo 2 angle slowly decreases

The UP controls the IIC Servo 3 angle slowly.

DOWN control IIC Servo 3 angle gradually decreases.

BT_L1 controls the angle of IIC Servo 4 to increase slowly.

BT_R1 controls the angle of IIC Servo to decrease gradually.

Please refer to the manual of the wireless controller for the detailed use of the wireless controller.

# Appendix

## External sensor connection method

In addition to the ultrasonic module socket and the servo jack, the PS2X&Motor Driver Board also reserves 10 sets of expansion pin interfaces, of which 6 are general digital I/0 interfaces and 4 are I2C control interfaces. These expansion interfaces can be used for external connection of various sensor modules, such as infrared obstacle avoidance modules, infrared tracking modules and other common sensor modules that can make your products more powerful and functional.