一、电机

是能把电能转换为机械能的设备装置

常见的电机：
（1）步进电机

一种将电脉冲信号，转换为对应 角速度/线位移 的电机

一次脉冲信号，电机内部的转子就会转动一个角度/一步

（2）舵机

可以精确控制角位移和线位移，但是对于角度有限制

常用于机器人的关节部位

（3）直流电机 <= 本次使用

接通直流电就可以转动，可以控制转动速度

生活中非常常见，俗称为马达

原理：

通电就能转

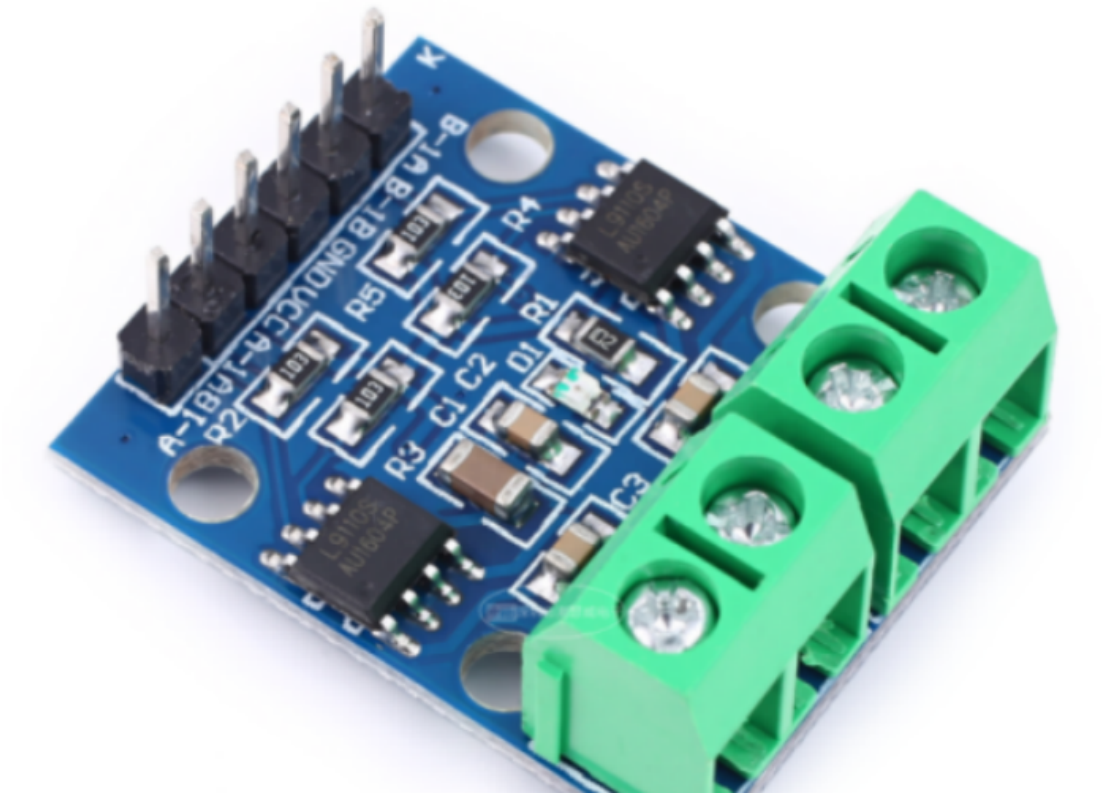
转速和电流大小相关，电流越大，电机转的越快

转动方向和电流方向有关

二、电机驱动

本次项目使用的是直流电机

只需要接直流电就能控制直流电机的转动



Q：为什么不直接将直流电机接到单片机IO口，而要加一个电机驱动呢?

A：单片机IO口输出的电流只有15mA左右，无法满足电机驱动的要求

本次使用的驱动模块是L9110S

工作电压5V ~ 12V之间，可以同时驱动两个直流电机

三、驱动原理

每个直流电机都有两个触点

假设为a、b

```
如果a 为高电平，b 为低电平 => 电机正转
那么a 为低电平，b 为高电平 => 电机翻转

项目有两个直流电机，一共就是四个触点，我们将这四个触点记为
    left_a left_b right_a right_b

    left_a 为高
    left_b 为低
        左边的电机正转，左轮正转

    right_a 为高
    right_b 为低
        右边的电机正转，右轮正转

小车的运动状态即为前进
其他运动同理
```

四、控制原理

两个直流电机通过杜邦线与L9110电机驱动模块连接

该模块有6个引脚，2个绿色的接线端子

2个绿色的接线端子 => 分别连接在了两个直流电机上

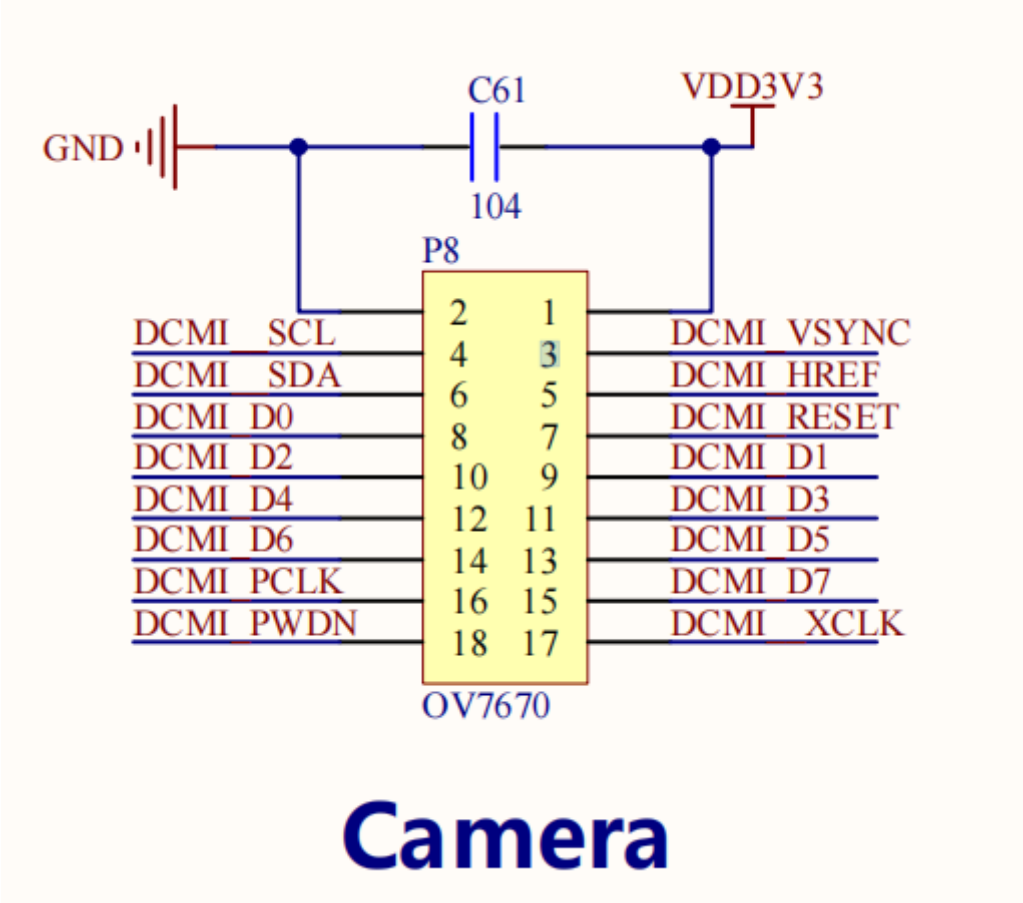
6个引脚 => 黑色排针，暂未连接，需要连接在开发板上

```
B-1A        模块的输入信号，接在开发板的GPIO口
B-1B        模块的输入信号，接在开发板的GPIO口
GND         接在开发板的GND
VCC         接在开发板的5V / 3V3
A-1A        模块的输入信号，接在开发板的GPIO口
A-1B        模块的输入信号，接在开发板的GPIO口
```

只需要使用6根杜邦线连接这6个引脚到开发板上即可

Q：怎么连接到开发板的GPIO呢？连接哪几个GPIO？

A：查看原理图 P4，有一组空闲的GPIO口，对应在板子的下边偏左边的位置



```
实物引脚编号为：
    2   4   6   8   10  12  14  16  18      靠开发板内部
    1   3   5   7   9   11  13  15  17      靠开发板边缘
```

连接好后，根据对应的网络标签，找到对应的GPIO引脚

2	4	6	8	10	12	14	16	18
GND	PD6	PD7	PC6	PC8	PE4	PE5	PA6	PG9
1	3	5	7	9	11	13	15	17
3.3V	PB7	PA4	PG15	PF11	PC9	PB6	PE6	PA8

分析小车的运动轨迹和控制信号的关系：

```
小车前进：

小车后退：

小车左转：

小车右转：

小车停止：
```

五、作业

（1）利用同住和螺丝，将开发板、电机驱动模块，固定在小车底座上

（2）在Keil上编写程序，控制与连接L9110S的GPIO引脚输出电平，观察并记录，在不同输出电平状态下，两个直流电机带动的车轮的转动方向

（3）编写程序，实现小车前进、后退、左转、右转、停止等动作的功能函数

```
void Car_Go()
{
    // LEFT_A 1
    GPIO_Set_Bits(GPIOx, GPIO_Pin_x);
    // LEFT_B 0
    GPIO_Reset_Bits(GPIOx, GPIO_Pin_x);
    // RIGHT_A 1
    GPIO_Set_Bits(GPIOx, GPIO_Pin_x);
    // RIGHT_B 0
    GPIO_Reset_Bits(GPIOx, GPIO_Pin_x);
}

void Car_Back()
{
    // LEFT_A 0
```

```
    // LEFT_B 1
    // RIGHT_A 0
    // RIGHT_B 1
}

void Car_Stop()
{
    // LEFT_A   X
    // LEFT_B   X
    // RIGHT_A  Y
    // RIGHT_B  Y       (X = 0 / 1  Y = 0 / 1)
}

void Car_TurnLeft()
{
    // （1）左轮不转，右轮正转

    // （2）左轮反转，右轮正转
}

void Car_TurnRight()
{
    // （1）左轮正转，右轮不转

    // （2）左轮正转，右轮反转
}

void car_gpio_init()
{
    // step1：使能四个信号控制端连接的GPIO引脚对应分组的时钟

    // step2：配置四个信号控制端连接的GPIO引脚为推挽输出模式

    // step3：设置四个信号控制端连接的GPIO引脚默认电平状态（小车停止）
    Car_Stop();
}

void key_gpio_init()
{
    // step1：使能四个按键对应的GPIO引脚对应分组的时钟

    // step2：配置四个按键对应的GPIO引脚为上拉输入模式

}

int main()
{
    car_gpio_init();
    key_gpio_init();

    while(1)
    {
        // 不断的判断按键的状态

        if(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_0) == 0)
        {
            // S1 按下
            Car_Go();
```

```
            while(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_0) == 0);// 等待S1松开
            Car_Stop();
        }
        else if(GPIO_ReadInputDataBit(GPIOx, GPIO_Pin_x) == 0)
        {
            // S2 按下
            Car_Back();
            while(GPIO_ReadInputDataBit(GPIOx, GPIO_Pin_x) == 0);// 等待S2松开
            Car_Stop();
        }
        else if(...)
        {
            // ...
            while(...);
        }
        else if(...)
        {
            // ...
            while(...);
        }
    }
}
```

效果：四个按键控制四种不同的运动状态，按下按键则发生指定的运动，不安下按键则处于停止状态