基于八路巡线模块的小车巡线

1. 开篇说明

请先阅读四路电机驱动板资料中的《电机介绍以及用法》，了解清楚自己现使用的电机参数、接线方式、供电电压。以免造成烧坏主板或者电机的后果。

2.实验准备

国赛底盘V2四驱版本、4*310电机、12V锂电池、八路巡线模块、四路电机驱动模块、MSPM0机器人扩展板（选配）、MSPM0G3507核心板（亚博）。

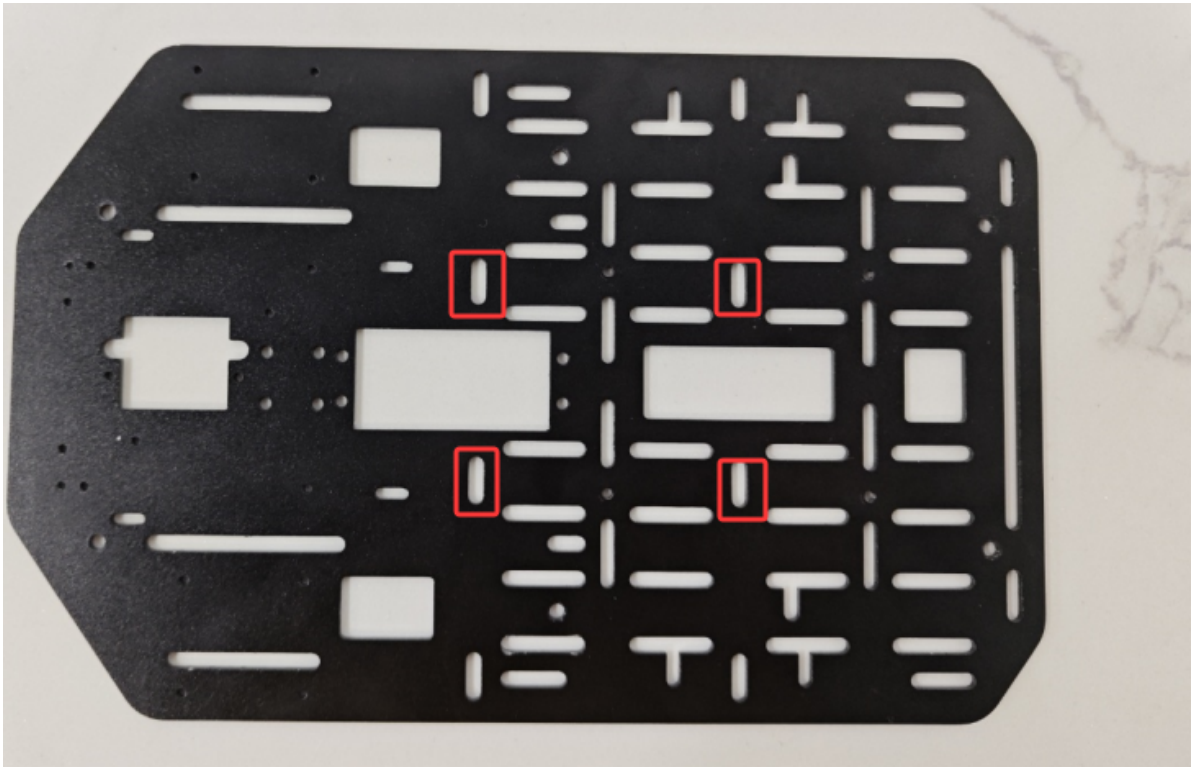
4个电机接口对应小车的关系如下：

M1 -> 左上电机(小车的左前轮)
M2 -> 左下电机(小车的左后轮)
M3 -> 右上电机(小车的右前轮)
M4 -> 右下电机(小车的右后轮)

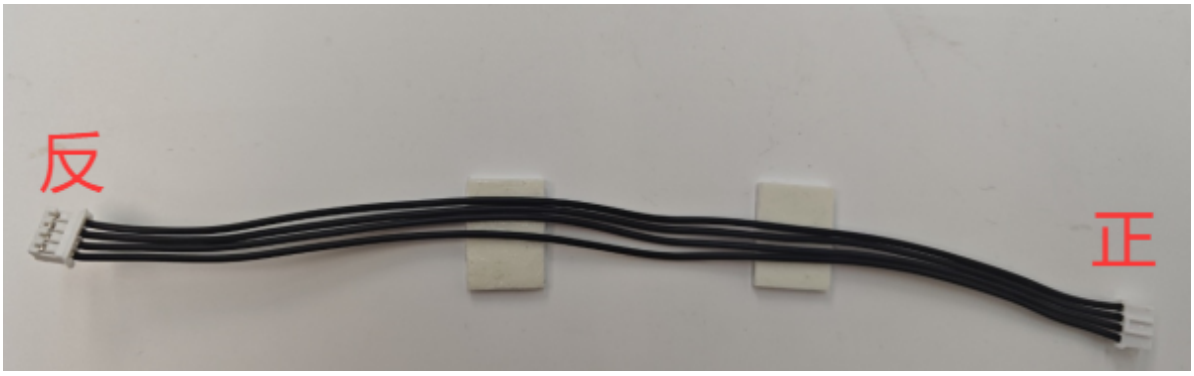
硬件接线：

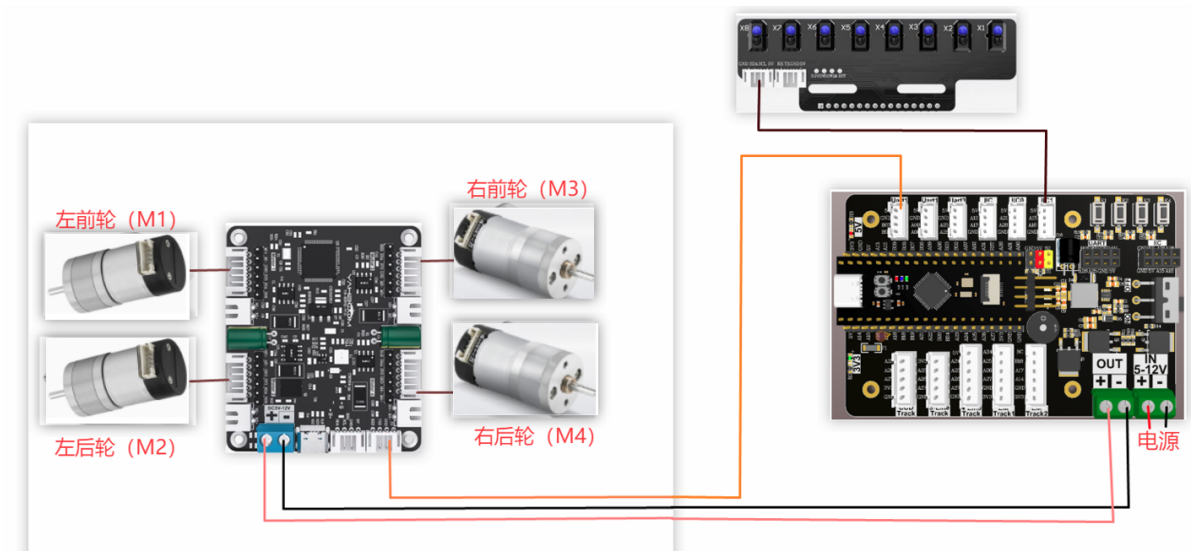
使用MSPM0机器人扩展板接线

在安装接线过程中，如果发现接线长度不够，可以把MSPM0机器人扩展板往前移一点安装，如下图位置。

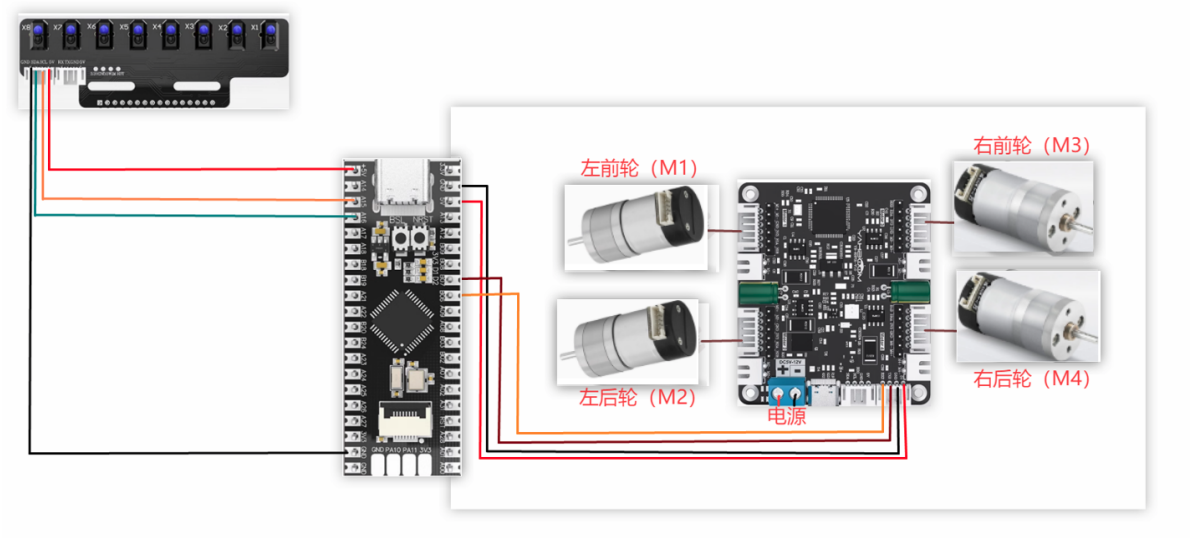


注意：八路巡线模块使用的线材、MSPM0机器人扩展板与四路电机驱动模块连接所用的线材为：PH2.0-4pin排线 双头全黑 反向(200mm)，反向排线座子方向如下图所示





使用MSPM0G3507核心板（亚博）接线



接线引脚

四路电机驱动板	MSPM0G3507核心板（亚博）
RX2	PB6
TX2	PB7
GND	GND
5V	5V

下面以M1电机为例，其他电机依此类推

电机	四路电机驱动板(Motor)
M+	M1+
M-	M1-
GND	GND
VCC	3V3
B	H1A
A	H1B

八路巡线模块	MSPM0G3507核心板（亚博）
5V	5V
SCL	PA15
SDA	PA16
GND	GND

3.关键代码解析

- app_irtracking.c

```
#define IRTrack_Trun_KP (500)
#define IRTrack_Trun_KI (0)
#define IRTrack_Trun_KD (0)

int pid_output_IRR = 0;
u8 trun_flag = 0;


#define IRR_SPEED             300  //巡线速度   Patrol speed

float PID_IR_Calc(int8_t actual_value)
{

    float IRTrackTurn = 0;
    int8_t error;
    static int8_t error_last=0;
    static float IRTrack_Integral;

    error=actual_value;

    IRTrack_Integral +=error;

    //位置式pid    Positional pid
    IRTrackTurn=error*IRTrack_Trun_KP
                          +IRTrack_Trun_KI*IRTrack_Integral
                          +(error - error_last)*IRTrack_Trun_KD;
    return IRTrackTurn;
}

void deal_IRdata(u8 *x1,u8 *x2,u8 *x3,u8 *x4,u8 *x5,u8 *x6,u8 *x7,u8 *x8)
{
    u8 IRbuf = 0xFF;
    IRbuf = IRI2C_ReadByte(0x30);

    *x1 = (IRbuf>>7)&0x01;
    *x2 = (IRbuf>>6)&0x01;
    *x3 = (IRbuf>>5)&0x01;
    *x4 = (IRbuf>>4)&0x01;
    *x5 = (IRbuf>>3)&0x01;
    *x6 = (IRbuf>>2)&0x01;
    *x7 = (IRbuf>>1)&0x01;
    *x8 = (IRbuf>>0)&0x01;
}


void LineWalking(void)
{
    static int8_t err = 0;
    static u8 x1,x2,x3,x4,x5,x6,x7,x8;

    deal_IRdata(&x1,&x2,&x3,&x4,&x5,&x6,&x7,&x8);
```

```
    ///L1为X1，白底灭灯时为1，黑线亮时为0      L1 is X1, 1 when the white background
is off, 0 when the black line is on///


//优先判断是否到直角或锐角  Prioritize whether to right angles or acute angles
     if(x1 == 0 && x2 == 0  && x3 == 0&& x4 == 0 && x5 == 0 && x6 == 1  && x7 ==
1 && x8 == 1) // 0000 0111
    {
        err = -15;
        delay_ms(100);
    }
    else if(x1 == 1 && x2 == 1  && x3 == 1&& x4 == 0 && x5 == 0 && x6 == 0  &&
x7 == 0 && x8 == 0) // 1110 0000
    {
        err = 15;
        delay_ms(100);
    }

    ......

    //剩下的就保持上一个状态        The rest will stay the same.
    pid_output_IRR = (int)(PID_Calc(err));

    Motion_Car_Control(IRR_SPEED, 0, pid_output_IRR);

}
```

- PID_IR_Calc：位置式PID计算，计算出来的结果用于控制小车运动。如果巡线效果不好，可以将KP和KD先置0，然后慢慢增加KP，最后再尝试加KD的值
- deal_IRdata:获取八路巡线模块的数据
- LineWalking：根据读到的八路巡线模块的状态来控制小车调整不同的运动状态

- app_motor.c

```
void Set_Motor(int MOTOR_TYPE)
{
    if(MOTOR_TYPE == 1)
    {
        send_motor_type(1);//配置电机类型 Configure motor type
        delay_ms(100);
        send_pulse_phase(30);//配置减速比 查电机手册得出     Configure the reduction
ratio. Check the motor manual to find out
        delay_ms(100);
        send_pulse_line(11);//配置磁环线 查电机手册得出 Configure the magnetic ring
wire. Check the motor manual to get the result.
        delay_ms(100);
        send_wheel_diameter(67.00);//配置轮子直径,测量得出         Configure the
wheel diameter and measure it
        delay_ms(100);
        send_motor_deadzone(1900);//配置电机死区,实验得出 Configure the motor dead
zone, and the experiment shows
        delay_ms(100);
    }
```

```
    else if(MOTOR_TYPE == 2)
    {
        send_motor_type(2);
        delay_ms(100);
        send_pulse_phase(20);
        delay_ms(100);
        send_pulse_line(13);
        delay_ms(100);
        send_wheel_diameter(48.00);
        delay_ms(100);
        send_motor_deadzone(1600);
        delay_ms(100);
    }

    else if(MOTOR_TYPE == 3)
    {
        send_motor_type(3);
        delay_ms(100);
        send_pulse_phase(45);
        delay_ms(100);
        send_pulse_line(13);
        delay_ms(100);
        send_wheel_diameter(68.00);
        delay_ms(100);
        send_motor_deadzone(1600);
        delay_ms(100);
    }

    else if(MOTOR_TYPE == 4)
    {
        send_motor_type(4);
        delay_ms(100);
        send_pulse_phase(48);
        delay_ms(100);
        send_motor_deadzone(1000);
        delay_ms(100);
    }

    else if(MOTOR_TYPE == 5)
    {
        send_motor_type(1);
        delay_ms(100);
        send_pulse_phase(40);
        delay_ms(100);
        send_pulse_line(11);
        delay_ms(100);
        send_wheel_diameter(67.00);
        delay_ms(100);
        send_motor_deadzone(1900);
        delay_ms(100);
    }
}

void Motion_Car_Control(int16_t V_x, int16_t V_y, int16_t V_z)
{
    float robot_APB = Motion_Get_APB();
    speed_lr = 0;
    speed_fb = V_x;
```

```
    speed_spin = (V_z / 1000.0f) * robot_APB;
    if (V_x == 0 && V_y == 0 && V_z == 0)
    {
        Contrl_Speed(0,0,0,0);
        return;
    }

    speed_L1_setup = speed_fb + speed_spin;
    speed_L2_setup = speed_fb + speed_spin;
    speed_R1_setup = speed_fb  - speed_spin;
    speed_R2_setup = speed_fb  - speed_spin;

    if (speed_L1_setup > 1000) speed_L1_setup = 1000;
    if (speed_L1_setup < -1000) speed_L1_setup = -1000;
    if (speed_L2_setup > 1000) speed_L2_setup = 1000;
    if (speed_L2_setup < -1000) speed_L2_setup = -1000;
    if (speed_R1_setup > 1000) speed_R1_setup = 1000;
    if (speed_R1_setup < -1000) speed_R1_setup = -1000;
    if (speed_R2_setup > 1000) speed_R2_setup = 1000;
    if (speed_R2_setup < -1000) speed_R2_setup = -1000;


 //printf("%d\t,%d\t,%d\t,%d\r\n",speed_L1_setup,speed_L2_setup,speed_R1_setup,s
peed_R2_setup);

    Contrl_Speed(speed_L1_setup, speed_L2_setup, speed_R1_setup,
speed_R2_setup);

}
```

- Set_Motor：这个函数用于存储本店在售电机的参数，通过修改empty.c开头处的MOTOR_TYPE参数，就可以实现一键配置。正常情况下使用本店的电机不要修改此处的代码。如果使用的是自己的电机，就可以任选一个电机，将参数都改成自己所使用的电机的参数。如果不理解这些参数的含义，可以查看文档《4路电机驱动板控制指令》来了解每一项指令的具体含义。
- Motion_Car_Control：控制小车运动

- app_motor_usart.c

```
//发送电机类型    Transmitter motor type
void send_motor_type(motor_type_t data)
{
    sprintf((char*)send_buff,"$mtype:%d#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));

}

//发送电机死区    Send motor dead zone
void send_motor_deadzone(uint16_t data)
{
    sprintf((char*)send_buff,"$deadzone:%d#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));
}
```

```
//发送电机磁环脉冲  Send motor magnetic ring pulse
void send_pulse_line(uint16_t data)
{
    sprintf((char*)send_buff,"$mline:%d#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));
}

//发送电机减速比   Transmitting motor reduction ratio
void send_pulse_phase(uint16_t data)
{
    sprintf((char*)send_buff,"$mphase:%d#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));
}

//发送轮子直径    Send wheel diameter
void send_wheel_diameter(float data)
{
    sprintf((char*)send_buff,"$wdiameter:%.3f#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));
}
```

通过串口发送电机的参数给四路电机驱动板

- empty.c

```
#define MOTOR_TYPE 2   //1:520电机 2:310电机 3:测速码盘TT电机 4:TT直流减速电机 5:L型520电机
                       //1:520 motor 2:310 motor 3:speed code disc TT motor 4:TT DC reduction motor 5:L type 520 motor
int main(void)
{
    USART_Init();//打印串口初始化 Print serial port initialization
    printf("please wait...\r\n");

     //使能DMA通道  Enable DMA Channel
    NVIC_ClearPendingIRQ(UART_1_INST_INT_IRQN);
    DL_DMA_enableChannel(DMA, DMA_CH0_CHAN_ID);
    NVIC_EnableIRQ(UART_1_INST_INT_IRQN);

    //设置电机类型    Set motor type
    Set_Motor(MOTOR_TYPE);

    //修改电机PID，这里的参数是为四驱310底盘配置的，其他底盘需要自己测试修改
    //Modify the motor PID, the parameters here are configured for the 4WD 310 chassis, other chassis need to test and modify their own!
    send_motor_PID(1.9,0.2,0.8);

    printf("Initialization Succeed\r\n");

    while(1)
    {
        LineWalking();//开始八路巡线  Starting eight-way patrols.
    }

}
```

- MOTOR_TYPE：修改成自己在使用的电机类型
- USART_Init: 初始化与四路电机驱动板通信的串口
- Set_Motor：设置电机类型
- send_motor_PID：设置电机的PID
- LineWalking：控制小车进行巡线

4.实验现象

将小车接好线，给MSPM0烧录程序后，把小车放在白底黑线的地图上，这个工程适配的地图是本店售卖的高难度地图，适配的是四驱310的底盘。其他的底盘的巡线效果不好的话，需要自己修改代码中的电机PID与巡线PID。插上电源。（巡线前建议先对模块进行校准），小车将巡黑线驾驶。