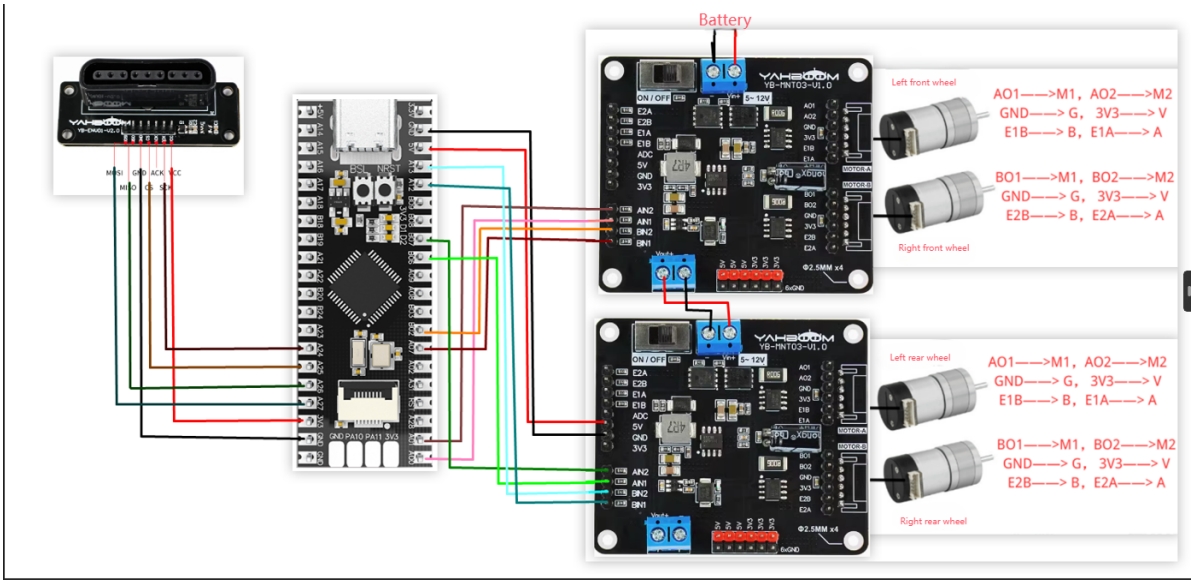
Car control

1. Learning objectives

Control the movement of the car through the handle

2. Hardware connection

Wiring of the handle receiver, MSPM0G3507 and dual-channel motor driver board



Lower driver board (left front wheel, right front wheel)	MSPM0G3507		Upper driver board (left rear wheel, right rear wheel)	MSPM0G3507
AIN1	PA0		AIN1	PB6
AIN2	PA1		AIN2	PB7
BIN1	PA7		BIN1	PA12
BIN2	PB2		BIN2	PA13
PS2 controller receiver	**MSPM0G3507**		5V	5V
MOSI	PA27		GND	GND
MISO	PA26			
GND	GND			
CS	PA25			
ACK	No connection			
SCK	PA24			
VCC	3V3			

3. Program description

- pstwo.c

```
//判断是否为红灯模式 Judge whether it is red light mode
//返回值：0，红灯模式    Return value: 0, red light mode
//其他，其他模式       Others, other modes
u8 PS2_RedLight(void)
{
    CS_L;
    PS2_Cmd(Comd[0]);  //开始命令   Start Command
    PS2_Cmd(Comd[1]);  //请求数据   Request data
    CS_H;
    if( Data[1] == 0X73)   return 0 ;
    else return 1;

}

void Get_PS2data(void)
{
    PS2_LX=PS2_AnologData(PSS_LX);
    PS2_LY=PS2_AnologData(PSS_LY);
    PS2_RX=PS2_AnologData(PSS_RX);
    PS2_RY=PS2_AnologData(PSS_RY);
    PS2_KEY=PS2_DataKey();

//  printf("PS2_LX=%d    ",PS2_LX);
//  printf("PS2_LY=%d    ",PS2_LY);
//  printf("PS2_RX=%d    ",PS2_RX);
//  printf("PS2_RY=%d    ",PS2_RY);
//  printf("PS2_KEY=%d    \r\n",PS2_KEY);

}

void PS2_CarControl(void)
{
    //----摇杆直向控制---//
    //----Joystick vertical control---//
    if(PS2_LX == 128 && PS2_LY == 127)
    {
        Motion_Set_Pwm(0,0,0,0);
    }

    if(PS2_LY>=0&&PS2_LY<70)//直走    Go straight
    {
        Motion_Set_Pwm(200,200,200,200);
    }

    if(PS2_LY<=255 && PS2_LY>200 &&PS2_LX == 128)//后退   Back
    {
        Motion_Set_Pwm(-200,-200,-200,-200);
    }

    if(PS2_LX>=0&&PS2_LX<70)//左旋    Rotate Left
    {
        Motion_Set_Pwm(-400,700,-400,700);
    }
```

```
    if(PS2_LX<=255 && PS2_LX>200 && PS2_LY == 127)//右旋  Rotate Right
    {
        Motion_Set_Pwm(700,-400,700,-400);
    }

    //----摇杆斜向控制---//
    //----Joystick tilt control---//
    if(PS2_LX >= 0&&PS2_LX <= 90 && PS2_LY >= 0&&PS2_LY <= 70)//左前旋转    Left
front rotation
    {
        Motion_Set_Pwm(0,500,0,500);
    }
    if(PS2_LX >= 165&&PS2_LX <= 255 && PS2_LY >= 0&&PS2_LY <= 70)//右前旋转 Right
front rotation
    {
        Motion_Set_Pwm(500,0,500,0);
    }
    if(PS2_LX >= 0&&PS2_LX <= 90 && PS2_LY >= 165&&PS2_LY <= 255)//左后旋转 Left
rear rotation
    {
        Motion_Set_Pwm(0,-500,0,-500);
    }
    if(PS2_LX >= 165&&PS2_LX <= 255 && PS2_LY >= 165&&PS2_LY <= 255)//右后旋转
Right rear rotation
    {
        Motion_Set_Pwm(-500,0,-500,0);
    }

    //---方向按键控制---//
    //---Direction button control---//
    switch(PS2_KEY){
        case KEY_FORWARD:
            Motion_Set_Pwm(200,200,200,200);
        break;

        case KEY_BACK:
            Motion_Set_Pwm(-200,-200,-200,-200);
        break;

        case KEY_SPINRIGHT:
            Motion_Set_Pwm(700,-400,700,-400);
        break;

        case KEY_SPINLEFT:
            Motion_Set_Pwm(-400,700,-400,700);
        break;

        defalut:
        break;
    }
}
```

- PS2_RedLight：Realize the red light mode judgment. When the handle is in non-red light mode, that is, traffic light mode, the car can be controlled to move;
- Get_PS2data：Get the joystick analog value and button value of the controller
- PS2_CarControl：Control the car through the acquired data

- bsp_motor.h

```
#define PWM_M1_A(value)
DL_TimerG_setCaptureCompareValue(PWM_0_INST,value,GPIO_PWM_0_C0_IDX);
#define PWM_M1_B(value)
DL_TimerG_setCaptureCompareValue(PWM_0_INST,value,GPIO_PWM_0_C1_IDX);

#define PWM_M2_A(value)
DL_TimerG_setCaptureCompareValue(PWM_0_INST,value,GPIO_PWM_0_C2_IDX);
#define PWM_M2_B(value)
DL_TimerG_setCaptureCompareValue(PWM_0_INST,value,GPIO_PWM_0_C3_IDX);

#define PWM_M3_A(value)
DL_TimerG_setCaptureCompareValue(PWM_1_INST,value,GPIO_PWM_1_C0_IDX);
#define PWM_M3_B(value)
DL_TimerG_setCaptureCompareValue(PWM_1_INST,value,GPIO_PWM_1_C1_IDX);

#define PWM_M4_A(value)
DL_TimerG_setCaptureCompareValue(PWM_2_INST,value,GPIO_PWM_2_C0_IDX);
#define PWM_M4_B(value)
DL_TimerG_setCaptureCompareValue(PWM_2_INST,value,GPIO_PWM_2_C1_IDX);
```

Define the set duty cycle functions for the 4 motor pins.

- bsp_motor.c

```
// 设置电机速度，speed:±1000, 0为停止   // Set the motor speed, speed: ±1000, 0
means stop
void Motor_Set_Pwm(uint8_t id, int16_t speed)
{
    int16_t pulse = Motor_Ignore_Dead_Zone(speed);
    // 限制输入 Limit Input
    if (pulse >= MOTOR_MAX_PULSE)
        pulse = MOTOR_MAX_PULSE;
    if (pulse <= -MOTOR_MAX_PULSE)
        pulse = -MOTOR_MAX_PULSE;

    switch (id)
    {
    case MOTOR_ID_M1:
    {
        if (pulse >= 0)
        {
            PWM_M1_A(pulse);
            PWM_M1_B(0);
        }
        else
        {
            PWM_M1_A(0);
            PWM_M1_B(-pulse);
        }
        break;
    }
    case MOTOR_ID_M2:
    {
```

```
        pulse = -pulse;
        if (pulse >= 0)
        {
            PWM_M2_A(pulse);
            PWM_M2_B(0);
        }
        else
        {
            PWM_M2_A(0);
            PWM_M2_B(-pulse);
        }
        break;
    }

    case MOTOR_ID_M3:
    {
        if (pulse >= 0)
        {
            PWM_M3_A(pulse);
            PWM_M3_B(0);
        }
        else
        {
            PWM_M3_A(0);
            PWM_M3_B(-pulse);
        }
        break;
    }
    case MOTOR_ID_M4:
    {
        pulse = -pulse;
        if (pulse >= 0)
        {
            PWM_M4_A(pulse);
            PWM_M4_B(0);
        }
        else
        {
            PWM_M4_A(0);
            PWM_M4_B(-pulse);
        }
        break;
    }

    default:
        break;
    }
}
```

Give the input value to the motor to control the forward and reverse rotation and speed of the motor. This control logic is not suitable for all motors and motor driver boards. If you are using the motor driver board and motor mentioned in this tutorial, if the direction of the car is not correct, you need to check the wiring again. If you are using a motor driver board and motor not mentioned in this tutorial, you need to modify it according to your own situation.

- app_motor.c

```
// 控制小车运动，Motor_X=[-1000, 1000]，超过范围则无效。
// Control the movement of the car, Motor_X=[-1000, 1000]. It will be invalid if
it exceeds the range.
void Motion_Set_Pwm(int16_t Motor_1, int16_t Motor_2, int16_t Motor_3, int16_t
Motor_4)
{
    if (Motor_1 >= -MOTOR_MAX_PULSE && Motor_1 <= MOTOR_MAX_PULSE)
    {
        Motor_Set_Pwm(MOTOR_ID_M1, Motor_1);
    }
    if (Motor_2 >= -MOTOR_MAX_PULSE && Motor_2 <= MOTOR_MAX_PULSE)
    {
        Motor_Set_Pwm(MOTOR_ID_M2, Motor_2);
    }
    if (Motor_3 >= -MOTOR_MAX_PULSE && Motor_3 <= MOTOR_MAX_PULSE)
    {
        Motor_Set_Pwm(MOTOR_ID_M3, Motor_3);
    }
    if (Motor_4 >= -MOTOR_MAX_PULSE && Motor_4 <= MOTOR_MAX_PULSE)
    {
        Motor_Set_Pwm(MOTOR_ID_M4, Motor_4);
    }
}
```

Limit the motor input value to not exceed the limit range.

- empty.c

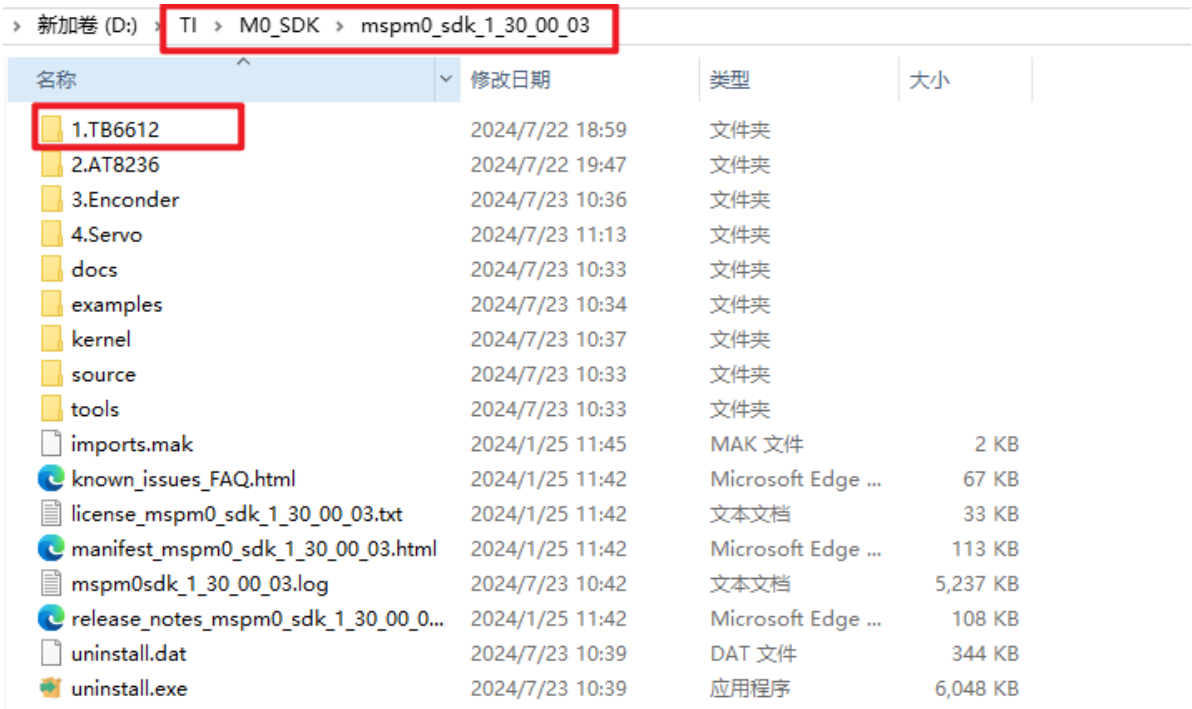
```
int main(void)
{
    extern int AnalogFLAG;

    USART_Init();
    while(1)
    {
        Get_PS2data();
        if(!PS2_RedLight())//切换为模拟量输入模式时，开始控制小车 When switching to
analog input mode, start controlling the car
        {
            PS2_CarControl();
        }
        delay_ms(2);
    }
}
```

After powering on and switching to traffic light mode, start controlling the car's movement.

Note: Be careful when changing the delay function used in this project! Random modification may cause excessive delay or disconnection of the PS2 controller.

Note: The project source code must be placed in the SDK path for compilation.

** For example, the path: D:\TI\M0_SDK\mspm0_sdk_1_30_00_03\1.TB6612**



4. Experimental phenomenon

Burn the line patrol program into MSPM0G3507. Patiently connect the wires according to the wiring diagram. After connecting the wires, patiently check whether the wires are connected correctly. If you do not check, the car may not move at best, or the board may be burned directly. After confirming that everything is correct, turn on the upper and lower drive board switches. Then turn on the handle switch, wait until the indicator light of the handle stops flashing, and the red and green lights of the handle receiver are always on. Press the mode button of the handle to switch to the traffic light mode of the handle. Then you can press the directional key and joystick on the left side of the handle to control the car. The directional key up and down controls forward and backward, and the left and right controls left and right rotation. The joystick can control the car forward and backward, diagonally controls left and right rotation, and left and right controls left and right rotation.