**实验九：树莓派平台-------巡线实验**

1. **实验前准备**

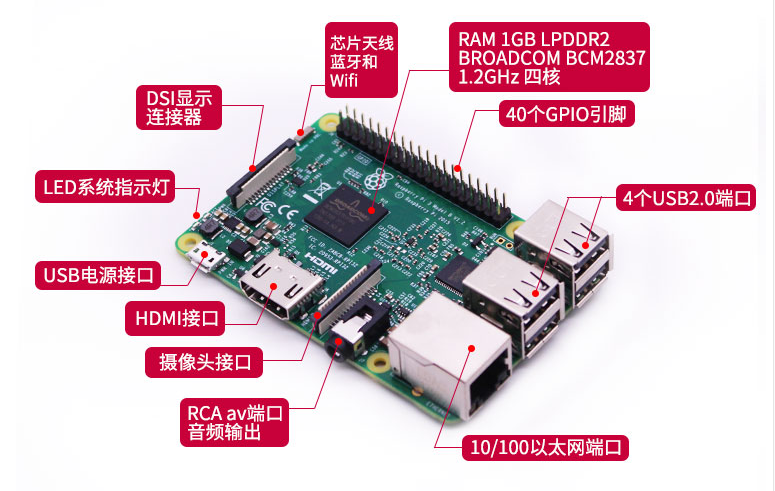
****

图1-1 树莓派主控板

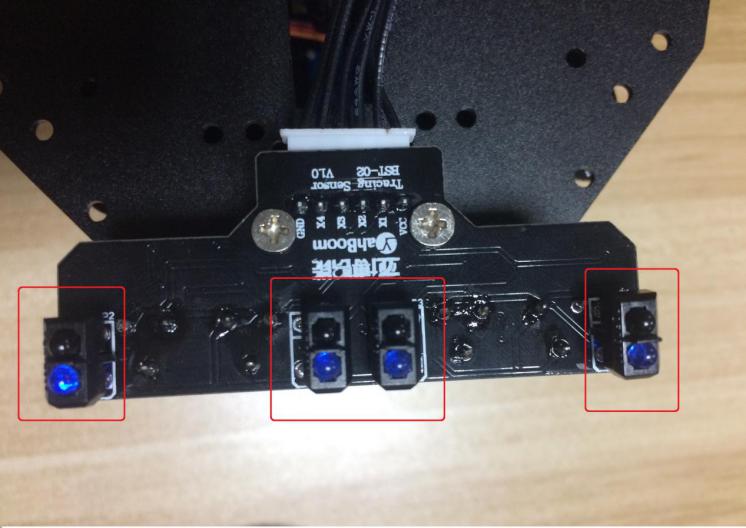


图1-2 4路红外巡线模块

1. **实验目的**

SSH服务登录树莓派系统之后，编译运行巡线程序启动实验后，接着按下按键K2，启动红外巡线功能，小车会自动的巡黑线行走。

1. **实验原理**

红外传感器巡线的基本原理是利用物体的反射性质，我们本次实验是巡黑线行驶，当红外线发射到黑线上时会被黑线吸收掉，发射到其他的颜色的材料上会有反射到红外的接受管上。我们根据这点的不同写相应的代码完成小车巡线功能。我们本次实验采用的是四路红外传感器分别连接在树莓派主控板上的wiringPi编码的21,9,7,1口上。

**4、实验步骤**

4-1.看懂原理图

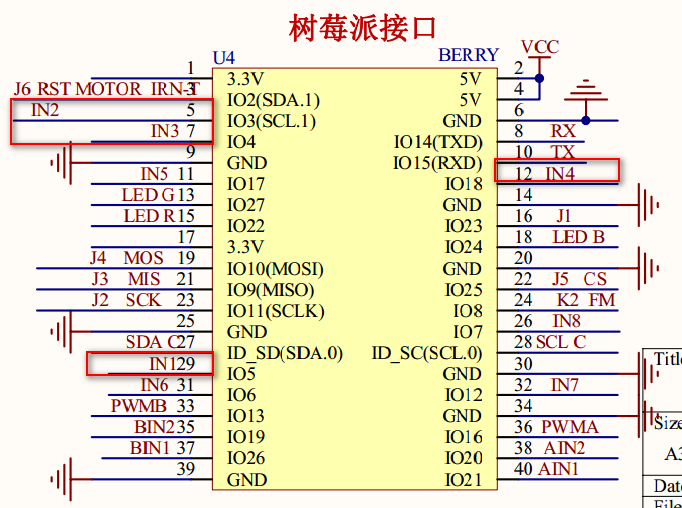


图4-1 树莓派主控板电路图

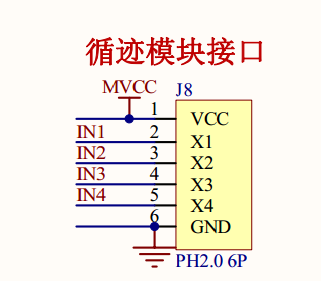


图4-2 4路红外传感器接线头

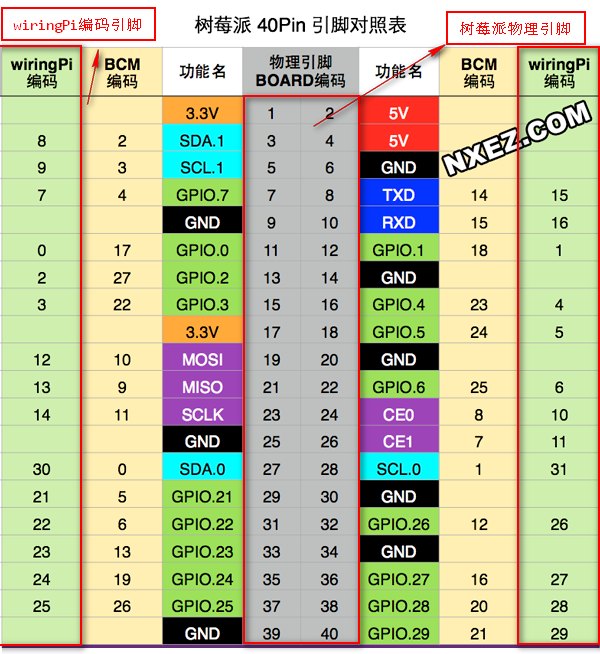


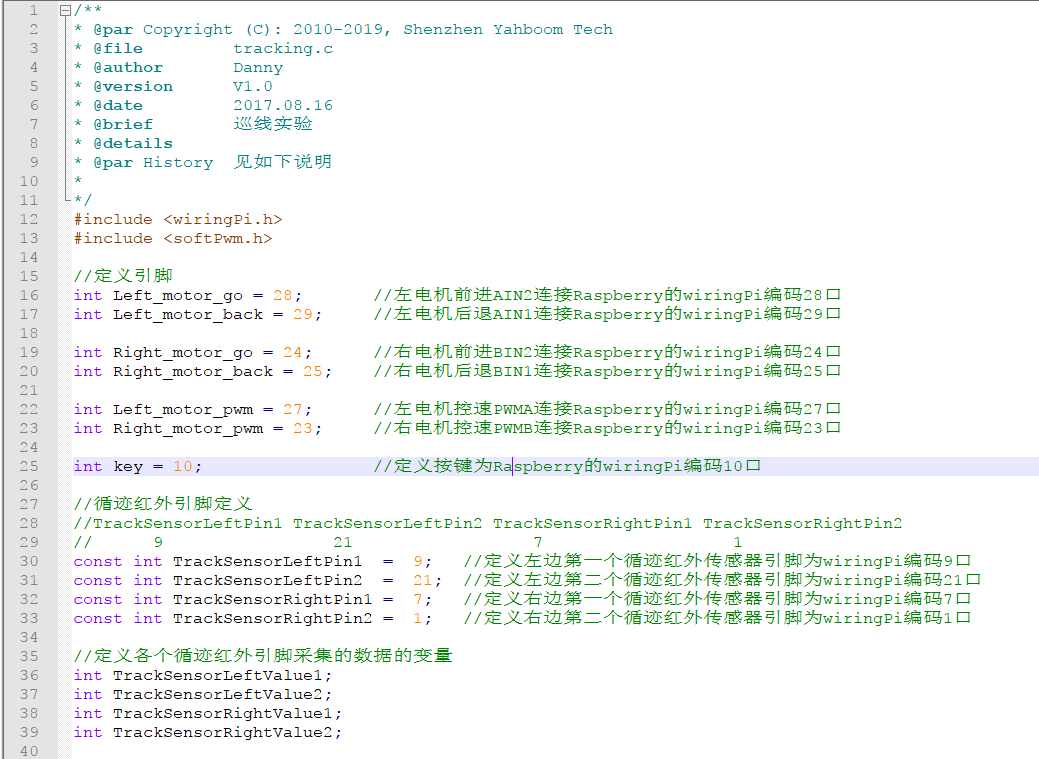
图4-3 树莓派40pin引脚对照表

4-2 由电路原理图可知从左到右4路红外传感器接在树莓派主控板上的wiringPi编码的9,21,7,1上.

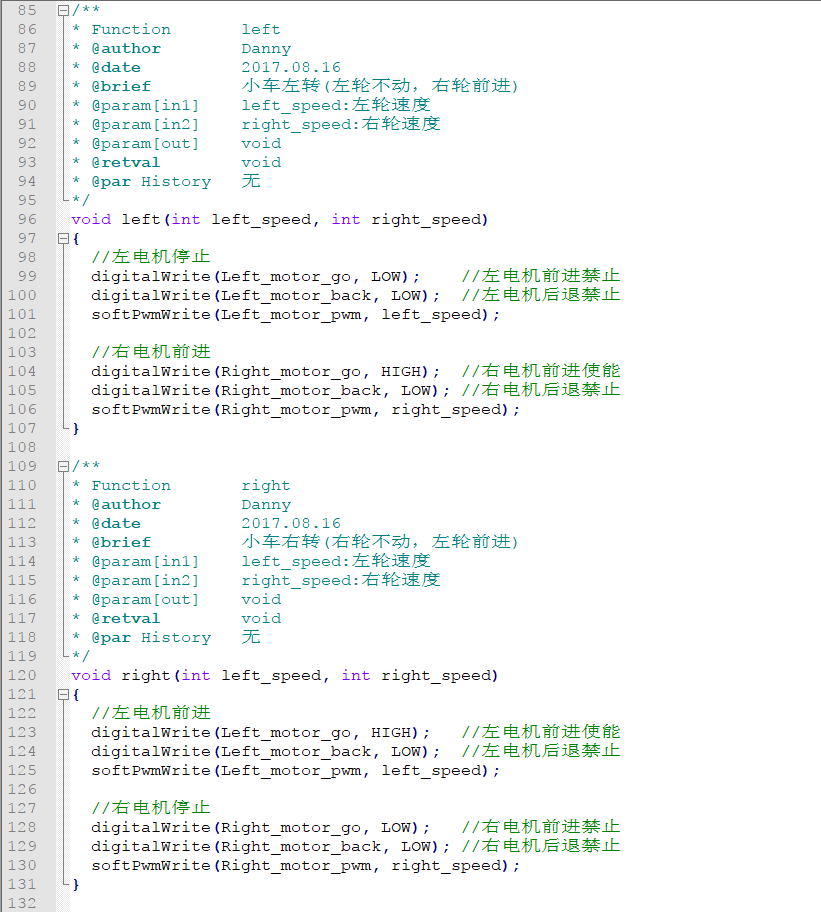
注：

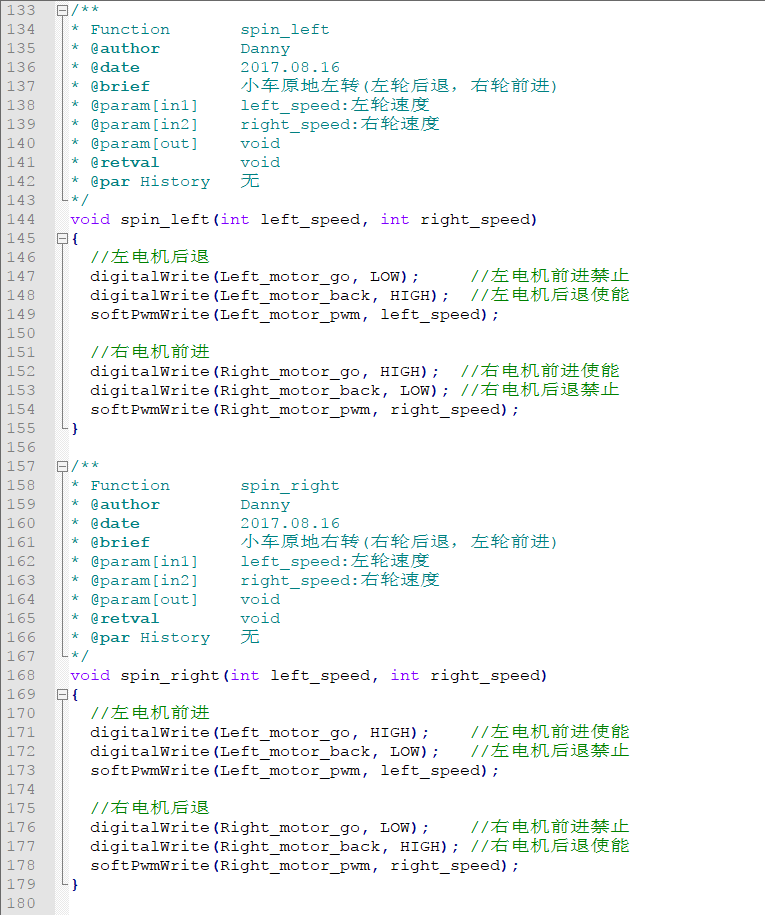
1.本次实验需要调节4路红外循迹模块的电位器使得巡线的灵敏度达到最佳。

4-3 程序代码如下：





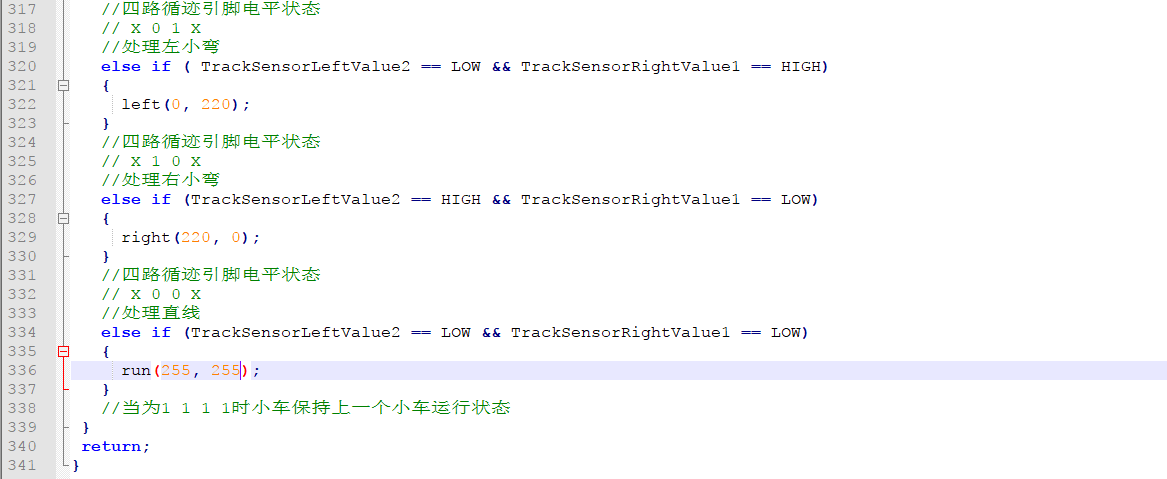








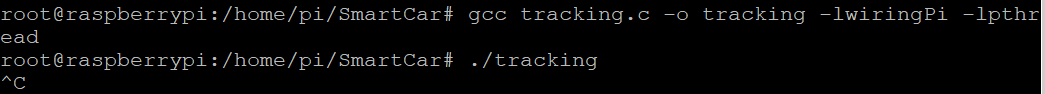




输入：

gcc tracking.c -o tracking -lwiringPi -lpthread

./tracking



接着

./initpin.sh初始化引脚。

代码：

/\*\*

\* @par Copyright (C): 2010-2019, Shenzhen Yahboom Tech

\* @file tracking.c

\* @author Danny

\* @version V1.0

\* @date 2017.08.16

\* @brief 巡线实验

\* @details

\* @par History 见如下说明

\*

\*/

#include <wiringPi.h>

#include <softPwm.h>

//定义引脚

int Left\_motor\_go = 28; //左电机前进AIN2连接Raspberry的wiringPi编码28口

int Left\_motor\_back = 29; //左电机后退AIN1连接Raspberry的wiringPi编码29口

int Right\_motor\_go = 24; //右电机前进BIN2连接Raspberry的wiringPi编码24口

int Right\_motor\_back = 25; //右电机后退BIN1连接Raspberry的wiringPi编码25口

int Left\_motor\_pwm = 27; //左电机控速PWMA连接Raspberry的wiringPi编码27口

int Right\_motor\_pwm = 23; //右电机控速PWMB连接Raspberry的wiringPi编码23口

int key = 10; //定义按键为Raspberry的wiringPi编码10口

//循迹红外引脚定义

//TrackSensorLeftPin1 TrackSensorLeftPin2 TrackSensorRightPin1 TrackSensorRightPin2

// 9 21 7 1

const int TrackSensorLeftPin1 = 9; //定义左边第一个循迹红外传感器引脚为wiringPi编码9口

const int TrackSensorLeftPin2 = 21; //定义左边第二个循迹红外传感器引脚为wiringPi编码21口

const int TrackSensorRightPin1 = 7; //定义右边第一个循迹红外传感器引脚为wiringPi编码7口

const int TrackSensorRightPin2 = 1; //定义右边第二个循迹红外传感器引脚为wiringPi编码1口

//定义各个循迹红外引脚采集的数据的变量

int TrackSensorLeftValue1;

int TrackSensorLeftValue2;

int TrackSensorRightValue1;

int TrackSensorRightValue2;

/\*\*

\* Function run

\* @author Danny

\* @date 2017.08.16

\* @brief 小车前进

\* @param[in1] left\_speed:左轮速度

\* @param[in2] right\_speed:右轮速度

\* @param[out] void

\* @retval void

\* @par History 无

\*/

void run(int left\_speed, int right\_speed)

{

//左电机前进

digitalWrite(Left\_motor\_go, HIGH); //左电机前进使能

digitalWrite(Left\_motor\_back, LOW); //左电机后退禁止

softPwmWrite(Left\_motor\_pwm, left\_speed );

//右电机前进

digitalWrite(Right\_motor\_go, HIGH); //右电机前进使能

digitalWrite(Right\_motor\_back, LOW); //右电机后退禁止

softPwmWrite(Right\_motor\_pwm, right\_speed);

}

/\*\*

\* Function brake

\* @author Danny

\* @date 2017.08.16

\* @brief 小车刹车

\* @param[in] time:延时时间

\* @param[out] void

\* @retval void

\* @par History 无

\*/

void brake(int time)

{

digitalWrite(Left\_motor\_go, LOW);

digitalWrite(Left\_motor\_back, LOW);

digitalWrite(Right\_motor\_go, LOW);

digitalWrite(Right\_motor\_back, LOW);

delay(time \* 100);

}

/\*\*

\* Function left

\* @author Danny

\* @date 2017.08.16

\* @brief 小车左转(左轮不动，右轮前进)

\* @param[in1] left\_speed:左轮速度

\* @param[in2] right\_speed:右轮速度

\* @param[out] void

\* @retval void

\* @par History 无

\*/

void left(int left\_speed, int right\_speed)

{

//左电机停止

digitalWrite(Left\_motor\_go, LOW); //左电机前进禁止

digitalWrite(Left\_motor\_back, LOW); //左电机后退禁止

softPwmWrite(Left\_motor\_pwm, left\_speed);

//右电机前进

digitalWrite(Right\_motor\_go, HIGH); //右电机前进使能

digitalWrite(Right\_motor\_back, LOW); //右电机后退禁止

softPwmWrite(Right\_motor\_pwm, right\_speed);

}

/\*\*

\* Function right

\* @author Danny

\* @date 2017.08.16

\* @brief 小车右转(右轮不动，左轮前进)

\* @param[in1] left\_speed:左轮速度

\* @param[in2] right\_speed:右轮速度

\* @param[out] void

\* @retval void

\* @par History 无

\*/

void right(int left\_speed, int right\_speed)

{

//左电机前进

digitalWrite(Left\_motor\_go, HIGH); //左电机前进使能

digitalWrite(Left\_motor\_back, LOW); //左电机后退禁止

softPwmWrite(Left\_motor\_pwm, left\_speed);

//右电机停止

digitalWrite(Right\_motor\_go, LOW); //右电机前进禁止

digitalWrite(Right\_motor\_back, LOW); //右电机后退禁止

softPwmWrite(Right\_motor\_pwm, right\_speed);

}

/\*\*

\* Function spin\_left

\* @author Danny

\* @date 2017.08.16

\* @brief 小车原地左转(左轮后退，右轮前进)

\* @param[in1] left\_speed:左轮速度

\* @param[in2] right\_speed:右轮速度

\* @param[out] void

\* @retval void

\* @par History 无

\*/

void spin\_left(int left\_speed, int right\_speed)

{

//左电机后退

digitalWrite(Left\_motor\_go, LOW); //左电机前进禁止

digitalWrite(Left\_motor\_back, HIGH); //左电机后退使能

softPwmWrite(Left\_motor\_pwm, left\_speed);

//右电机前进

digitalWrite(Right\_motor\_go, HIGH); //右电机前进使能

digitalWrite(Right\_motor\_back, LOW); //右电机后退禁止

softPwmWrite(Right\_motor\_pwm, right\_speed);

}

/\*\*

\* Function spin\_right

\* @author Danny

\* @date 2017.08.16

\* @brief 小车原地右转(右轮后退，左轮前进)

\* @param[in1] left\_speed:左轮速度

\* @param[in2] right\_speed:右轮速度

\* @param[out] void

\* @retval void

\* @par History 无

\*/

void spin\_right(int left\_speed, int right\_speed)

{

//左电机前进

digitalWrite(Left\_motor\_go, HIGH); //左电机前进使能

digitalWrite(Left\_motor\_back, LOW); //左电机后退禁止

softPwmWrite(Left\_motor\_pwm, left\_speed);

//右电机后退

digitalWrite(Right\_motor\_go, LOW); //右电机前进禁止

digitalWrite(Right\_motor\_back, HIGH); //右电机后退使能

softPwmWrite(Right\_motor\_pwm, right\_speed);

}

/\*\*

\* Function back

\* @author Danny

\* @date 2017.08.16

\* @brief 小车后退

\* @param[in] time：延时时间

\* @param[out] void

\* @retval void

\* @par History 无

\*/

void back(int time)

{

//左电机后退

digitalWrite(Left\_motor\_go, LOW); //左电机前进禁止

digitalWrite(Left\_motor\_back, HIGH); //左电机后退使能

softPwmWrite(Left\_motor\_pwm, 40);

//右电机后退

digitalWrite(Right\_motor\_go, LOW); //右电机前进禁止

digitalWrite(Right\_motor\_back, HIGH); //右电机后退使能

softPwmWrite(Right\_motor\_pwm, 40);

delay(time );

}

/\*\*

\* Function key\_scan

\* @author Danny

\* @date 2017.08.16

\* @brief 按键检测(包含软件按键去抖)

\* @param[in] void

\* @param[out] void

\* @retval void

\* @par History 无

\*/

void key\_scan()

{

while (digitalRead(key)); //当按键没有被按下一直循环

while (!digitalRead(key)) //当按键被按下时

{

delay(10); //延时10ms

if (digitalRead(key) == LOW)//第二次判断按键是否被按下

{

delay(100);

while (!digitalRead(key)); //判断按键是否被松开

}

}

}

/\*\*

\* Function main

\* @author Danny

\* @date 2017.08.16

\* @brief 先调用setup初始化配置里面的按键扫描函数，

\* 循迹模式开启

\* @param[in] void

\* @retval void

\* @par History 无

\*/

void main()

{

//wiringPi初始化

wiringPiSetup();

//初始化电机驱动IO口为输出方式

pinMode(Left\_motor\_go, OUTPUT);

pinMode(Left\_motor\_back, OUTPUT);

pinMode(Right\_motor\_go, OUTPUT);

pinMode(Right\_motor\_back, OUTPUT);

//创建两个软件控制的PWM脚

softPwmCreate(Left\_motor\_pwm,0,255);

softPwmCreate(Right\_motor\_pwm,0,255);

//定义按键接口为输入接口

pinMode(key, INPUT);

//定义四路循迹红外传感器为输入接口

pinMode(TrackSensorLeftPin1, INPUT);

pinMode(TrackSensorLeftPin2, INPUT);

pinMode(TrackSensorRightPin1, INPUT);

pinMode(TrackSensorRightPin2, INPUT);

//调用按键扫描函数

key\_scan();

while(1)

{

//检测到黑线时循迹模块相应的指示灯亮，端口电平为LOW

//未检测到黑线时循迹模块相应的指示灯灭，端口电平为HIGH

TrackSensorLeftValue1 = digitalRead(TrackSensorLeftPin1);

TrackSensorLeftValue2 = digitalRead(TrackSensorLeftPin2);

TrackSensorRightValue1 = digitalRead(TrackSensorRightPin1);

TrackSensorRightValue2 = digitalRead(TrackSensorRightPin2);

//四路循迹引脚电平状态

// 0 0 X 0

// 1 0 X 0

// 0 1 X 0

//以上6种电平状态时小车原地右转，速度为250,延时80ms

//处理右锐角和右直角的转动

if ( (TrackSensorLeftValue1 == LOW || TrackSensorLeftValue2 == LOW) && TrackSensorRightValue2 == LOW)

{

spin\_right(250, 250);

delay(80);

}

//四路循迹引脚电平状态

// 0 X 0 0

// 0 X 0 1

// 0 X 1 0

//处理左锐角和左直角的转动

else if ( TrackSensorLeftValue1 == LOW && (TrackSensorRightValue1 == LOW || TrackSensorRightValue2 == LOW))

{

spin\_left(250, 250);

delay(80);

}

// 0 X X X

//最左边检测到

else if ( TrackSensorLeftValue1 == LOW)

{

spin\_left(150, 150);

//delay(10);

}

// X X X 0

//最右边检测到

else if ( TrackSensorRightValue2 == LOW )

{

spin\_right(150, 150);

//delay(10);

}

//四路循迹引脚电平状态

// X 0 1 X

//处理左小弯

else if ( TrackSensorLeftValue2 == LOW && TrackSensorRightValue1 == HIGH)

{

left(0, 220);

}

//四路循迹引脚电平状态

// X 1 0 X

//处理右小弯

else if (TrackSensorLeftValue2 == HIGH && TrackSensorRightValue1 == LOW)

{

right(220, 0);

}

//四路循迹引脚电平状态

// X 0 0 X

//处理直线

else if (TrackSensorLeftValue2 == LOW && TrackSensorRightValue1 == LOW)

{

run(255, 255);

}

//当为1 1 1 1时小车保持上一个小车运行状态

}

return;

}