《机器人导论实验》报告

|  |  |  |
| --- | --- | --- |
| **第五次实验** | **日期：2022-12-3** | **得分：** |
| **学号：** | **姓名：*Steven*** | **专业：智能科学与技术** |

注意：小组的实验成果可以共享，切忌实验报告完全复制粘贴。

# 实验准备

1. 学习并理解SPI通信原理与PS2端口的结构和原理。
2. 基于PS2端口的通信原理将PS2与STM32主板通信使小车完成实验要求功能。
3. 结合程序学习PS2发送端手柄各个按键的定义返回值学习如何通过按键切换到手柄的不同模式并且如何在不同模式下实现小车的各个功能。

# 实验原理

一、SPI通信协议

PS2采用的是SPI通信协议SPI是一种高速的全双工同步的通信总线并且在芯片的管脚上只占用四根线DI、DO、CS、CLK节约了芯片的管脚同时为PCB的布局上节省空间提供方便主要应用在EEPROMFLASH实时时钟AD转换器还有数字信号处理器和数字信号解码器之间。

SPI分为主、从两种模式一个SPI通讯系统需要包含一个且只能是一个主设备一个或多个从设备。提供时钟的为主设备Master接收时钟的设备为从设备SlaveSPI接口的读写操作都是由主设备发起。当存在多个从设备时通过各自的片选信号进行管理。

SPI是全双工且SPI没有定义速度限制一般的实现通常能达到甚至超过10Mbps。

二、PS2端口介绍

1．DI/DAT：信号流向从手柄到主机此信号是一个8bit的串行数据同步传送于时钟的下降沿。信号的读取在时钟由高到低的变化过程中完成。

2．DO/CMD：信号流向从主机到手柄此信号和DI相对信号是一个8bit的串行数据同步传送于时钟的下降沿。

3．NC：空端口。4GND：电源地

5．VCC：接收器工作电源电源范围35V。

6．CS/SEL：用于提供手柄触发信号。在通讯期间处于低电平。

7．CLK：时钟信号由主机发出用于保持数据同步。

8．NC：空端口。

9．ACK：从手柄到主机的应答信号。此信号在每个8bits数据发送的最后一个周期变低并且CS一直保持低电平如果CS信号不变低约60微秒PS主机会试另一个外设。

# 实验任务

1. 在红灯模式下 通过一个摇杆实现前进和转弯 要求速度与转弯弧度会随摇杆的变化而变化**，**即摇杆拨到最上方速度最大**，**拨到一半速度不是最大值

提示：speed=二次处理后的摇杆值×系数，（比如10、9.5、9……）

2. 在其他模式下，按三角键实现自动圆周运动，按圆圈键实现自动随意运动，按错误键实现原地静止。

# 实验过程与实验结果

本次实验中主要的功能函数是scan\_ps2(void)。其中首先使用了一个if函数，根据PS2\_RedLight()的返回值（ps2\_mode）来判断使用摇杆模式还是按键模式。在每种模式内部，使用一个switch函数，根据PS2\_DataKey()的返回值（key）来判断当前按了哪个按键，并在对应的case内部写按键对应事件的代码。

分析清楚本次代码的结构之后，实现起来就不难了。

1. 摇杆控制前进和转弯

摇杆模式对应ps2\_mode==0。

其中switch(key)的case PSB\_PAD\_UP对应前进事件、case PSB\_PAD\_DOWN对应后退事件、case PSB\_PAD\_LEFT对应左转事件、case PSB\_PAD\_RIGHT对应右转事件。

这里只需简单设置case即可。

|  |
| --- |
| case PSB\_PAD\_UP:  Set\_Pwm\_Motor1(5000); //电机设置  Set\_Pwm\_Motor2(5000);  break;  case PSB\_PAD\_DOWN:  Set\_Pwm\_Motor1(-5000); //电机设置  Set\_Pwm\_Motor2(-5000);  break;  case PSB\_PAD\_LEFT:  CPWM[1] = 1000; //左转一个角度  break;  case PSB\_PAD\_RIGHT:  CPWM[1] = 1500 + LX\_AD \* 1000 / 255; //右转合适角度(从中间开始)  break; |

1. 按钮实现特定功能

这部分的实现原理和上一部分是相同的，不同的是上一步是检测是否触发摇杆，这一步检测是否按了某个按钮。同样是在对应的case中写对应的代码。

|  |
| --- |
| case PSB\_TRIANGLE:  //运动模式=圆周  Set\_Pwm\_Motor1(5000); //电机设置  Set\_Pwm\_Motor2(5000);  CPWM[1] = 2300; //左转一个角度  break;  case PSB\_CROSS:  //运动模式=静止  Set\_Pwm\_Motor1(0); //电机设置  Set\_Pwm\_Motor2(0);  CPWM[1] = 1500; //左转一个角度  break;  case PSB\_SQUARE:  break;  case PSB\_CIRCLE:  //运动模式=随意(自行设定)  while (1)     {  int angle = (int)(1000 \* sin(i \* 3.14159 / 180)) + 1500; // 计算前轮的转向角度  CPWM[1] = angle; // 将转向角度传入底盘控制函数  Set\_Pwm\_Motor1(2500);  Set\_Pwm\_Motor2(2500);  key = PS2\_DataKey();  if (key == PSB\_CROSS)  break; // 决定前进还是后退，并控制角度  Delay\_ms(5); // 信号间隔5ms，此时前轮转向周期约为1.8s，后轮转换一次方向约为5s  i++;  }  break; |

# 实验心得