|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 实践  项目  名称 | 基于stm32的蓝牙小车 | 实践项目类型 | | | | |
| 验证 | 演示 | 综合 | 设计 | 其他 |
|  |  | ＊ | ＊ |  |
| 1. 系统设计方案分析   本次实训所用到的硬件模块有：STM32最小系统板、步进电机、HC-05蓝牙，L298N、蜂鸣器、按键、LED等。该系统以STM32最小系统板为核心，在MDK环境下进行编程，通知控制L298N模块IN1~IN4引脚的高低电平进而控制电机的转动方向，使小车产生不同的转动模式。控制小车的转动模式通过蓝牙模块来实现，利用手机的应用蓝牙调试宝和蓝牙模块相通，进而控制小车的各种运转模式。  蓝牙小车可实现的功能：通过手机蓝牙，控制小车运动，在蓝牙程序中，我们用到了switch-case语句来对小车的功能进行控制，通过手机蓝牙输入a，使小车前进，输入b小车后退，输入c小车右转，输入d小车左转，输入f打开车灯，输入e关闭车灯，为了方便观察蓝牙的输入，我们设置了一个LED灯，当蓝牙有输入时，LED灯就会闪烁200ms，除此之外，我们还为小车设置了车载音乐，我们通过PB11按键控制蜂鸣器来实现车载音乐的响灭。   1. 系统主要功能电路模块分析   步进电机是一种电气设备，能将电脉冲信号转换为角位移或线位移。其具有独特的特性，即接收到一定频率的脉冲信号后会转动指定的步数。步进电机本身到电机轴之间一般会有一级齿轮减速，比如64指减速比表示电机本身的转子转64圈，输出的电机轴才转1圈。  HC-05蓝牙模块则是一种实现无线通信的模块，常用于建立短距离的无线链接。它可以与STM32微处理器等设备连接，通过发送和接收数据来实现控制。  L298N是一款接受高电压的电机驱动器，既可以驱动直流电机，也可以驱动步进电机。一片驱动芯片可以同时控制两个直流减速电机做不同动作，在6V到46V的电压范围内，提供2安培的电流，并且具有过热自断和反馈检测功能。因为L298N模块具有两个H桥，每个H桥将驱动步进电机的电磁线圈之一。通过以特定顺序为这些电磁线圈通电，步进电机的轴可以以小步长精确地向前或向后移动。  蜂鸣器是一种能够发出声音的设备，它通常由电磁线圈和振膜构成，当电流通过电磁线圈时，振膜会以特定频率振动而发出声音。   1. 系统程序设计分析（含各功能模块程序设计流程图） 2. main.c函数：   #include "delay.h"  #include "sys.h"  #include "pwm.h"  #include "usart.h"  #include "stm32f10x\_gpio.h"  #include "stm32f10x\_rcc.h"  #include "LED.H"  void Stop(void);  int main(void)  {  SystemInit();    delay\_init(72);//延时初始化  uart\_init(9600);//波特率9600  LED\_Init();  Key\_Init();  PWM\_Init2(999,719); //左、右电机pwm  Stop(); //防止一上电，小车就开始跑  uint8\_t a=0;  while(1)  {  if(GPIO\_ReadInputDataBit(GPIOB,GPIO\_Pin\_11)==0)  {    delay\_ms(20);  while(GPIO\_ReadInputDataBit(GPIOB,GPIO\_Pin\_11)==0);  delay\_ms(20);  a++;  }  if (a == 1)  {  B\_Music();  }  if (a == 2)  {  a=0;  }  }  }    //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*STOP\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    void Stop(void)//让小车停下的函数  {  TIM\_SetCompare1(TIM3,0);  TIM\_SetCompare2(TIM3,0);    TIM\_SetCompare3(TIM3,0);  TIM\_SetCompare4(TIM3,0); }  2.蓝牙通信.c函数 #include "sys.h"  #include "usart.h"  #include "delay.h"  void uart\_init(u32 bound){  //GPIO端口设置  GPIO\_InitTypeDef GPIO\_InitStructure;  USART\_InitTypeDef USART\_InitStructure;  NVIC\_InitTypeDef NVIC\_InitStructure;  RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_USART1|RCC\_APB2Periph\_GPIOA, ENABLE); //使能USART1¬GPIOA时钟  //USART1\_TX GPIOA.9  GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_9; //PA.9  GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;  GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AF\_PP; //复用推挽输出  GPIO\_Init(GPIOA, &GPIO\_InitStructure);//初始化GPIOA.9    //USART1\_RX GPIOA.10初始化  GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_10;//PA10  GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_IN\_FLOATING;//浮空输入  GPIO\_Init(GPIOA, &GPIO\_InitStructure);//初始化GPIOA.10  //Usart1 NVIC 配置  NVIC\_InitStructure.NVIC\_IRQChannel = USART1\_IRQn;  NVIC\_InitStructure.NVIC\_IRQChannelPreemptionPriority=3 ;//抢占优先级3  NVIC\_InitStructure.NVIC\_IRQChannelSubPriority = 3; //子优先级3  NVIC\_InitStructure.NVIC\_IRQChannelCmd = ENABLE; //IRQ通道使能  NVIC\_Init(&NVIC\_InitStructure); //根据指定的参数初始化vic寄存器    //USART 初始化设置  USART\_InitStructure.USART\_BaudRate = bound;//串口波特率  USART\_InitStructure.USART\_WordLength = USART\_WordLength\_8b;//字长为8位数据格式  USART\_InitStructure.USART\_StopBits = USART\_StopBits\_1;//一个停止位  USART\_InitStructure.USART\_Parity = USART\_Parity\_No;//无奇偶检验位  USART\_InitStructure.USART\_HardwareFlowControl=USART\_HardwareFlowControl\_None;//无硬件数据流控制  USART\_InitStructure.USART\_Mode = USART\_Mode\_Rx | USART\_Mode\_Tx; //收发模式  USART\_Init(USART1, &USART\_InitStructure); //初始化串口1  USART\_ITConfig(USART1, USART\_IT\_RXNE, ENABLE);//开启串口接受中断  USART\_Cmd(USART1, ENABLE); //使能串口1  }  void USART1\_IRQHandler(void)  {  u8 res;  if(USART\_GetITStatus(USART1,USART\_IT\_RXNE)!= RESET)  {  res=USART\_ReceiveData(USART1);  USART\_SendData(USART1,res);  switch(res)  {  case 0x0a://小车前进  GPIO\_SetBits(GPIOA,GPIO\_Pin\_2);//set亮，reset暗  delay\_ms(200);  GPIO\_ResetBits(GPIOA,GPIO\_Pin\_2);//set亮，reset暗    TIM\_SetCompare3(TIM3,600); //PB0 左电机  TIM\_SetCompare4(TIM3,0); //PB1    TIM\_SetCompare1(TIM3,600); //PA6 右电机  TIM\_SetCompare2(TIM3,0); //PA7    delay\_ms(200);Stop();    break;  case 0x0b://小车后退  GPIO\_SetBits(GPIOA,GPIO\_Pin\_2);//set亮，reset暗  delay\_ms(200);  GPIO\_ResetBits(GPIOA,GPIO\_Pin\_2);//set亮，reset暗  TIM\_SetCompare3(TIM3,0); //PB0 左前  TIM\_SetCompare4(TIM3,600); //PB1 左后    TIM\_SetCompare1(TIM3,0); //PA6 右前  TIM\_SetCompare2(TIM3,600); //PA7 右后  delay\_ms(200);Stop();    break;  case 0x0c://小车右转  GPIO\_SetBits(GPIOA,GPIO\_Pin\_2);//set亮，reset暗  delay\_ms(200);  GPIO\_ResetBits(GPIOA,GPIO\_Pin\_2);//set亮，reset暗  TIM\_SetCompare3(TIM3,600); //PB0 左电机前进  TIM\_SetCompare4(TIM3,0); //PB1  TIM\_SetCompare1(TIM3,0); //PA6 右电机前进  TIM\_SetCompare2(TIM3,600); //PA7  delay\_ms(100);Stop();  break;  case 0x0d://小车左转  GPIO\_SetBits(GPIOA,GPIO\_Pin\_2);//set亮，reset暗  delay\_ms(200);  GPIO\_ResetBits(GPIOA,GPIO\_Pin\_2);//set亮，reset暗  TIM\_SetCompare3(TIM3,0); //PB0 左前  TIM\_SetCompare4(TIM3,600); //PB1 左后    TIM\_SetCompare1(TIM3,600); //PA6 右前  TIM\_SetCompare2(TIM3,0); //PA7 右后  delay\_ms(100);Stop();  break;  case 0x0e://开启车灯  GPIO\_ResetBits(GPIOA,GPIO\_Pin\_3);//set亮，reset暗  GPIO\_ResetBits(GPIOA,GPIO\_Pin\_4);//set亮，reset暗  break;  case 0x0f://关闭车灯  GPIO\_SetBits(GPIOA,GPIO\_Pin\_3);//set亮，reset暗  GPIO\_SetBits(GPIOA,GPIO\_Pin\_4);//set亮，reset暗  break;  }  }  }   1. 系统调试结果分析       当蓝牙输入f时，车灯开启，发出蓝色亮光。  五、总结  在PWM定时器的更改的时候，忘记把引脚对应的通道打开，导致小车的电机不能正常驱动。我们在选择电机驱动时，最终选择了L298N，它的驱动电压是要12V的，但我们一开始选择的电源电压是9v的达不到启动电压，电机也没有正常驱动，当我们更换了12V电压的时候电机驱动就能正常运行了。驱动电机一定要注意和STM32共地。LED灯要注意增加电阻来保护灯不被烧坏。在弄蜂鸣器的PWM时，忘记在int main函数里增加蜂鸣器初始化函数，导致蜂鸣器不能正常运作。程序没有用到PID调控，车子的左右重量不平衡会导致车辆直线行驶不稳定。 | | | | | | |