语音中文播报

1. 学习目标

本次课程我们主要学习使用 89C52RC 型号的 51 单片机和语音合成播报模块实现语音中文播报功能。

2. 课前准备

语音合成播报模块采用的是 I2C 通讯，将模块的 SDA，SCL 分别连接 51 的 P2.0 和 P2.1 引脚。VCC 和 GND 分别连接 51 的 5V 和 GND。

3.语音合成播报模块协议

语音播报是通过 I2C 通讯间按照规定的数据帧进行配置和使用的，具体请查看语音合成播报模块通讯协议文档教程。

注意新旧版语音播报模块 IIC 地址不同，新版 IIC 地址为:0x30,旧版为 0x50。如果使用不正常可以尝试修改另一个 IIC 地址测试。

4. 程序

模块操作用定义相应的函数和枚举。举个例子设置合成风格，由模块协议文档中文本合成控制的协议可知，通过写入相应文本符号可以设置相应的功能，如[f?]为设置合成风格，通过设置合成类型的枚举变量进行设置风格的选择，定义相应的设置函数。

作用	控制	详细说明	芯片
合成风格设置	[f?]	？为 0，一字一顿的风格	[f1]
		？为 1，正常合成	

```
typedef enum
{
        Style_Single,//? 为 0， 一字一顿的风格
        Style_Continue//? 为 1， 正常合成
} Style_Type; //合成风格设置 [f?]
void SetStyle(Style_Type style);
```

合成风格设置函数中字符串 f 和选择风格的枚举变量通过调用文本控制函数将字符串和数值变量合成新的符合文本控制格式的字符串，再调用帧发送函数完成设置。

```
void SetStyle(Style_Type style)
{
  TextCtrl('f', style);
    while(GetChipStatus() != ChipStatus_Idle)
    {
      delay(10);
    }
}
```

```
void TextCtrl(char c, int d)
{
  char str[10];
  if (d != -1)
    sprintf(str, "[%c%d]", c, d);
  else
    sprintf(str, "[%c]", c);
  SetBase(str);

}
```

帧发送函数，将字符串按照命令帧的格式，计算对应字符串的高低直接数值，通过结构体中的变量进行命令设置，文本编码设置，这里需要注意需要将对应的编译器设置为对应的编码方式，例如我这里为了后续播报中文，我需要将模块设置为 GB2312，同时需要对编译器的字符串编码也设置为 GB2312 格式，这里使用的为 keil3 默认为中文编码，所以无需设置。

命令帧格式结构	帧头	数据区长度		数据区		
	0xFD	高字节	低字节	命令字	文本编码格式	待合成文本
		0xHH	0xLL	0x01	0x00~0x03	……

```
void SetBase(u8 *str)
{
    u16 size = strlen(str) + 2;

    XFS_Protocol_TypeDef DataPack;

    DataPack.DataHead = DATAHEAD;
    DataPack.Length_HH = size >>8;
    DataPack.Length_LL = size;
    DataPack.Commond = 0x01;
    DataPack.EncodingFormat = 0x00;
    DataPack.Text = str;

    I2C_ByteWrite(DataPack.DataHead );
    I2C_ByteWrite(DataPack.Length_HH);
    I2C_ByteWrite(DataPack.Length_LL);
    I2C_ByteWrite(DataPack.Commond);
    I2C_ByteWrite(DataPack.EncodingFormat);

    I2C_Writes_Bytes(DataPack.Text,strlen(DataPack.Text));
}
```

当语音合成模块正在合成文本的时候，如果又收到一帧有效合成命令帧，芯片会立刻停止当前正在合成的文本，转而合成新收到的文本，这里就需要利用查询芯片状态的命令，只有当检测到芯片空闲的时候再发送新的命令帧。通过查看模块通讯协议文本可知模块对应状态的返回值，相应的也在类中定义了相应的变量。

回传数据类型	回传数据	触发条件
初始化成功	0x4A	芯片初始化成功
收到正确命令帧	0x41	收到正确的命令帧
收到错误命令帧	0x45	收到错误的命令帧
芯片忙碌	0x4E	收到“状态查询命令”，芯片处于合成文本状态回传 0x4E
芯片空闲	0x4F	当一帧数据合成完以后，芯片进入空闲状态回传 0x4F；当芯片收到“状态查询命令”，芯片处于空闲状态回传 0x4F

```
typedef enum
{
        ChipStatus_InitSuccessful = 0x4A,//初始化成功回传
        ChipStatus_CorrectCommand = 0x41,//收到正确的命令帧回传
        ChipStatus_ErrorCommand = 0x45,//收到不能识别命令帧回传
        ChipStatus_Busy = 0x4E,//芯片忙碌状态回传
        ChipStatus_Idle = 0x4F//芯片空闲状态回传
} ChipStatus_Type;//芯片回传
```

根据模块通讯协议可知获取模块的帧协议规则，并定义相应的获取模块状态的函数。

<table>
<tr><th>名称</th><th>发送的数据</th><th>说明</th></tr>
<tr><td>命令字</td><td>0x21</td><td>通过该命令获取相应参数，来判断 TTS 芯片是否处在合成状态，返回 0x4E 表明芯片仍在合成中，返回 0x4F 表明芯片处于空闲状态</td></tr>
<tr><td>参数列表</td><td>无</td><td></td></tr>
<tr><td>命令帧格式结构</td><td colspan="2">
<table>
<tr><td>帧头</td><td colspan="2">数据区长度</td><td>数据区</td></tr>
<tr><td rowspan="2">0xFD</td><td>高字节</td><td>低字节</td><td>命令字</td></tr>
<tr><td>0x00</td><td>0x01</td><td>0x21</td></tr>
</table>
</td></tr>
</table>

```
u8 GetChipStatus()
{
    u8 dat = 0xff;
    u8* pBuffer = 0xff;
    u8 AskState[4] = {0xFD,0x00,0x01,0x21};

    I2C_Writes_Bytes(AskState,4);
    delay(100);

    Start_I2c();                        //启动总线
    I2C_SendByte(I2C_ADDR+1);          //发送器件地址
    if(ack==0)return(0);
    delay(5);//必须加一点延时
    dat=I2C_RcvByte();                  //读取数据

    Ack_I2c(1);                 //发送非应答信号
    Stop_I2c();                 //结束总线
    return(dat);
}
```

设置模块音量，播报发音人等，设置合成播报的文本，检测芯片的状态。当芯片空闲时才可以进行新的设置和文本合成。

```
SetVolume(10);
SetReader(Reader_XiaoPing);
speech_text("你好亚博智能科技",GB2312);
while(GetChipStatus() != ChipStatus_Idle)   //等待芯片空闲
{
  delay(10);
}


SetReader(Reader_XuDuo);
speech_text("欢迎使用亚博智能语音播报模块",GB2312);
while(GetChipStatus() != ChipStatus_Idle)   //等待芯片空闲
{
  delay(10);
}
```

5. 实验现象

程序下载成功以后，模块分别由小萍播报“你好亚博智能科技”，许多播报“欢迎使用亚博智能科技语音合成播报模块”。