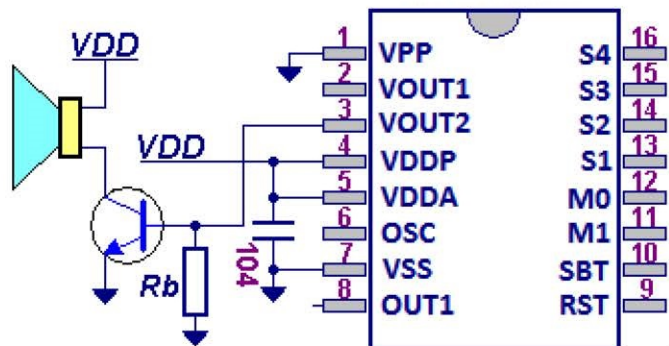
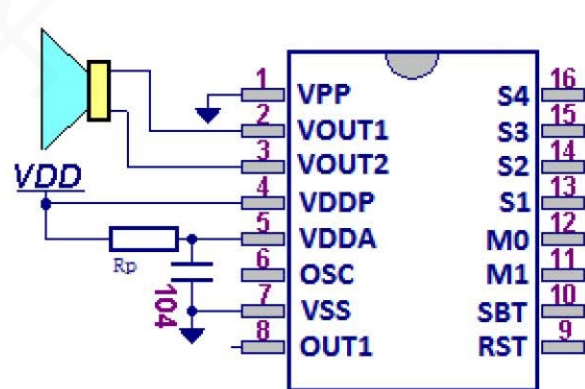


工程项目

一 原理图



$R_b = 220$ to $1k$ ($R_b = 390$ ohm and $Tr = 8050D$ (gain > 150) 为参考值)



$R_p = 56$ ohm (当工作电压大于3.6时需加 R_p 电阻)

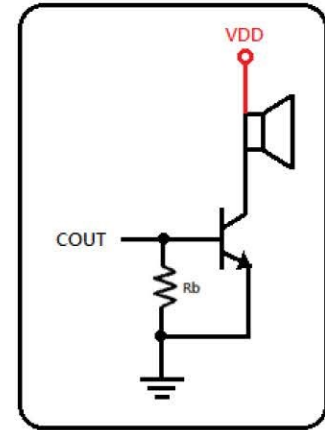
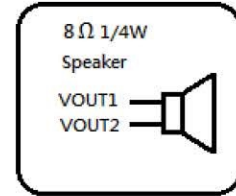
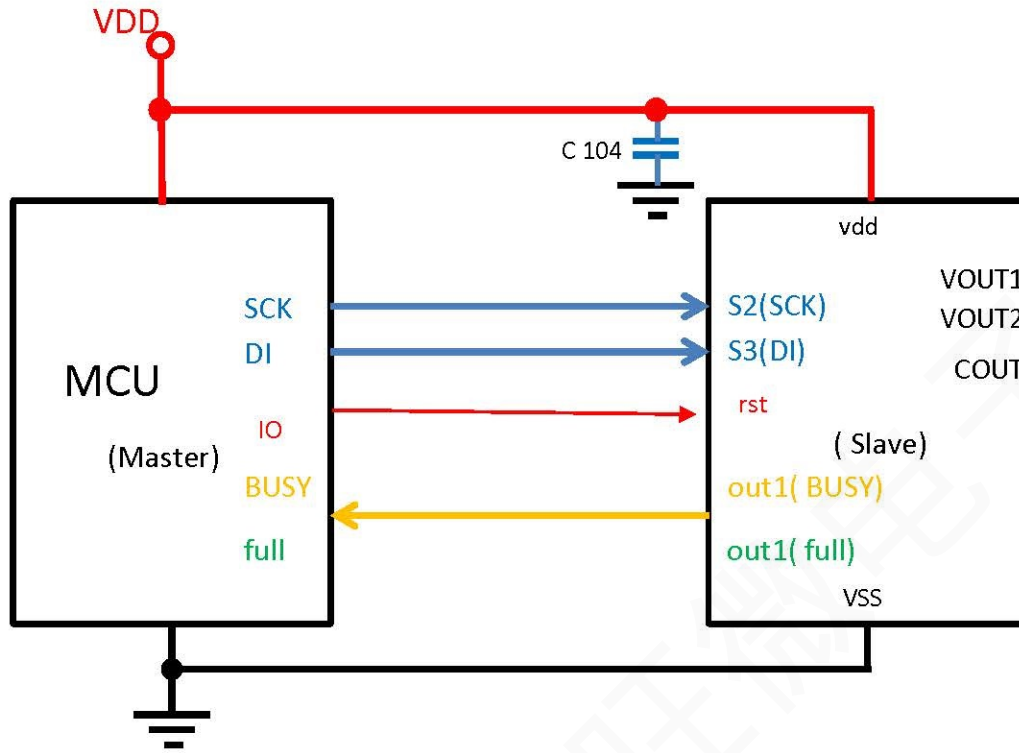
注意：在画pcb线路图时，“104电容”必需尽量靠近语音 ic 引脚。

建议：用主控mcu，控制语音ic的复位脚 RST，每次上电复位一下（1ms以上）。

说明：图1为接三极管dac输出模式，建议采用插件三极管，RB电阻取值建议范围220到1k之间（默认390）。图2为直推喇叭pwm输出模式，Rp = 56 ohm 这个电阻当工作电压大于3.6v时候才需要接（不接时这个脚vdda需接vdd电源）。

二 MCU接线

I2C Mode



脚位S2 :为时钟信号（SCK），DI在SCK为低时改变，DI在上升沿是会被读取。

脚位S3 :为数据信号（DI）。

脚位out1: 为反馈信号busy，full。Busy信号是有声音在播放时输出高电平，无声音播放时输出低电平，full信号是语音ic的缓存地址已经被占用了就输出高电平，缓存地址为空时就输出低电平（一般out1脚位输出的是Busy信号）。

脚位RST: 为复位脚位，建议每次上电时给语音ic复位一下

三 指令介绍

CPU Serial Control Command																
	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
LOAD	1	0	0	1	0	1	094h + address [D9~D0]									
PLAY	1	0	0	1	1	0	098h + address [D9~D0]									
PU1	1	0	1	0	0	1	0A4h + address don't care --- without Ramp-up									
PD1	1	0	1	1	0	1	0B4h + address don't care --- without Ramp-down									
PU2	1	0	1	0	1	0	0A8h + address don't care --- with Ramp-up									
PD2	1	0	1	1	1	0	0B8h + address don't care --- with Ramp-down									
VOL	0	1	0	0	0	1	0	0	0	0	0	0	VOL [3:0]			
VOL++	0	1	0	0	1	0	048h + address don't care									
VOL--	0	1	0	1	0	1	054h + address don't care									

Load指令：此指令为**播放指令**（**094h + address [D9~D0]**，**D9~D0为声音地址码**），这个指令与**play**播放指令的区别在于这个指令没有覆盖功能，只有有等前面正在播放的声音播放完了之后**load**的声音才会被播放出来，此指令和**out1**输出的**full**反馈信号配合使用。

Play指令：此指令为**播放指令**（**098h + address [D9~D0]**，**D9~D0为声音地址码**），这个指令与**load**播放指令的区别在于这个指令有覆盖功能，前面正在播放的声音时用此指令就可以直接替换掉正在播放的声音，此指令和**out1**输出的**busy**反馈信号配合使用。

PU1指令：此指令为**唤醒指令**（**0A4h + address don't care**（**1或0都可以**）），在用**load**指令时，必需先唤醒（**PU1**）之后才能用**load**指令,而**play**指令已经自带有唤醒的头码可以直接使用**play**指令（**pu1为pwm输出模式的唤醒指令**）。

PD1指令：此指令为**休眠指令**（**0B4h + address don't care**（**1或0都可以**）），不需要语音ic工作时发送**PD1**指令之后ic就会进入休眠模式之后，此时的静态电流少于5ua（**pd1为pwm输出模式的唤醒指令**）。

PU2指令：此指令为**唤醒指令**（0A8h + address don't care（**1或0都可以**）），在用load指令时，必需先唤醒（PU2）之后才能用load指令,而play指令已经自带有唤醒的头码可以直接使用play指令（**pu2为dac输出模式的唤醒指令**）。

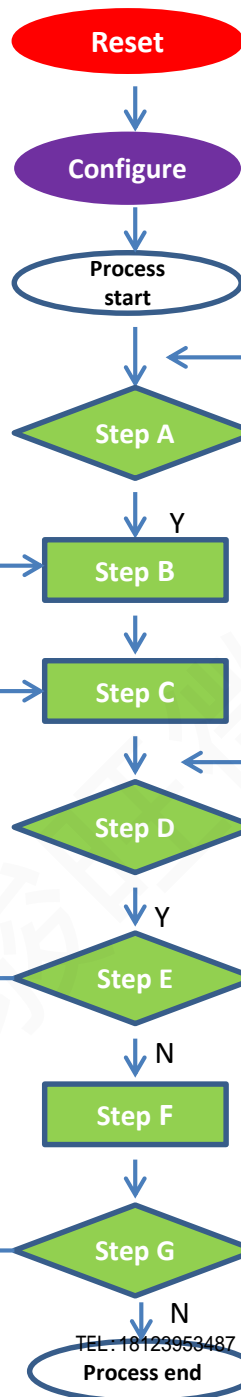
PD2指令：此指令为**休眠指令**（0B8h + address don't care（**1或0都可以**）），不需要语音ic工作时发送PD2指令之后ic就会进入休眠模式之后，此时的静态电流少于5ua（**pd2为pwm输出模式的唤醒指令**）。

VOL指令：此指令为**音量调节指令**，VOL [3:0]为D4~D0为四位的二进制码，最大声音为0000B,最小声音位1111H，此指令可以任意选择输出多大的声音。

Vol++指令：此指令为**音量调节指令**，每次发送此指令音量都会**自加1**。

Vol--指令：此指令为**音量调节指令**，每次发送此指令音量都会**自减1**。

注意：要播放一段声音就是，唤醒+播放+休眠，就可以调出声音了



上电/外部复位

外部复位时间: Typ = 5us

Configure

内部初始化时间: 2ms

Process start

Step A

Step A: 如何你想播发声音.

Step B

Step B: 需先唤醒语音IC, 下唤醒指令之后判断sbt脚位是否完成唤醒 (用指令PU1 / PU2) .

Step C

Step C: 下播放指令+播放声音的地址码 (指令play/load) .

Step D

Step D: 然后去检测out1脚的信号是否完成

Step E

Step E: 如果你想继续播放声音等待out1脚的信号为低就可以下下一个声音地址码.

Step F

Step F: 完成了声音播放后, 进行休眠 (用指令PD1/PD2) , 进入低功耗

Step G

Step G: 如果进入休眠之后想再一次播放声音就还要从step B 这一步开始.

Active CMD : PU1/PU2, Load, Play

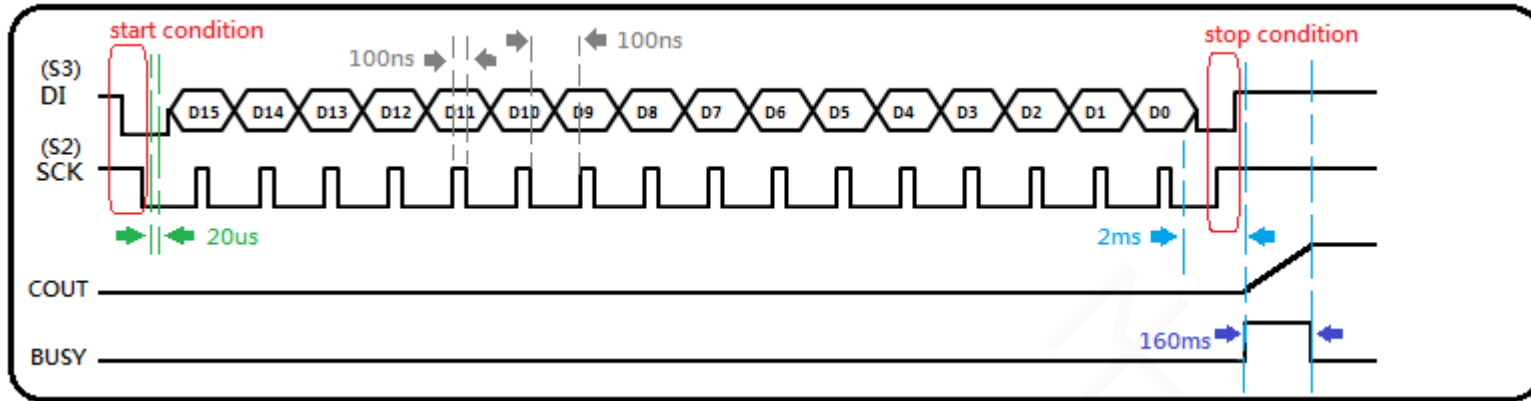
De-Active CMD : PD1/PD2

TEL: 18123953487

QQ: 502725276

Process end

Step B : CMD PU1 (A400h) / PU2 (A800h) 此指令位唤醒指令



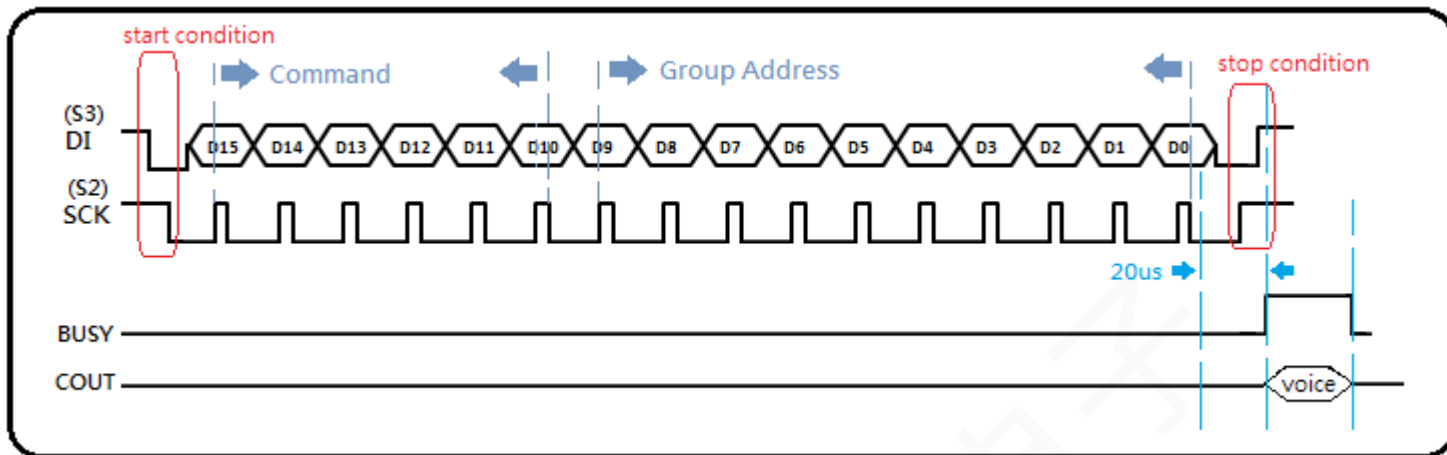
```
void I2Cdata(unsigned int cmd)
```

```
{  
    unsigned int mask = 0;  
    for(mask = 0x8000;mask > 0;mask >>=1)  
    {  
        SCK = 0;  
        DI = (cmd & mask) ? 1:0;  
        SCK = 1;  
    }  
    SCK = 0;  
}
```

```
void SoundChip_Active(unsigned int cmd)
```

```
{  
    SCK = 1;    //开始条件  
    DI = 1;    //  
    DI = 0;    //  
    //-----//  
    Delay20us();    //这里要延时(20us)  
    I2Cdata(cmd);    //下唤醒指令(用指令pu1/pu2)  
    //-----//结束条件  
    DI = 0;    //  
    SCK = 1;    //  
    DI = 1;    //  
    Delay2ms();    //等待状态醒来的状态存在(2ms)  
    while(BUSY);    //wait end of ramp 这个时候busy为高, (需等待  
    160ms)  
}
```


Step C : CMD Load (9400h) + Group Address 此指令为播放指令

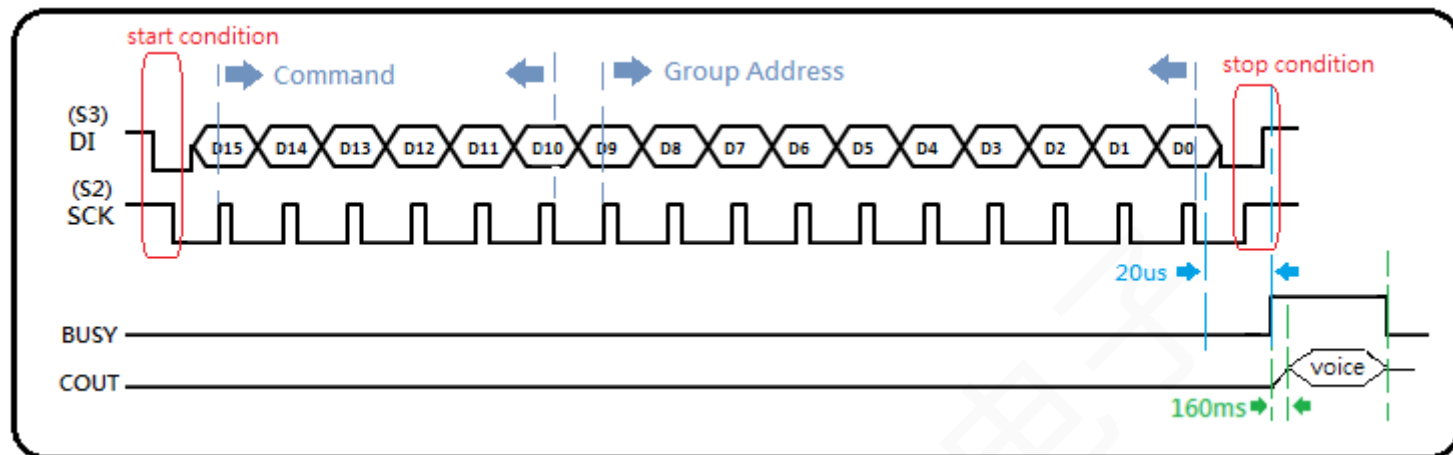


```
void I2Cdata(unsigned int cmd)
{
    unsigned int mask = 0;
    for(mask = 0x8000;mask > 0;mask >>=1)
    {
        SCK = 0;
        DI = (cmd & mask) ? 1:0;
        SCK = 1;
    }
    SCK = 0;
}
```

```
void SoundChip_Load(unsigned int cmd)
{
    unsigned int cmd = 0;
    cmd = 0x9400 + addr;

    SCK = 1;    //开始条件
    DI = 1;    //
    DI = 0;    //
    //-----//
    I2Cdata(cmd);    //下播放指令load
    //-----//结束条件
    DI = 0;    //
    SCK = 1;    //
    DI = 1;    //
    Delay20us();    //需要等待20us之后才能去判断sbt脚位的反馈信号
}
```

Step C : CMD Play (9800h) + Group Address 此指令为播放指令



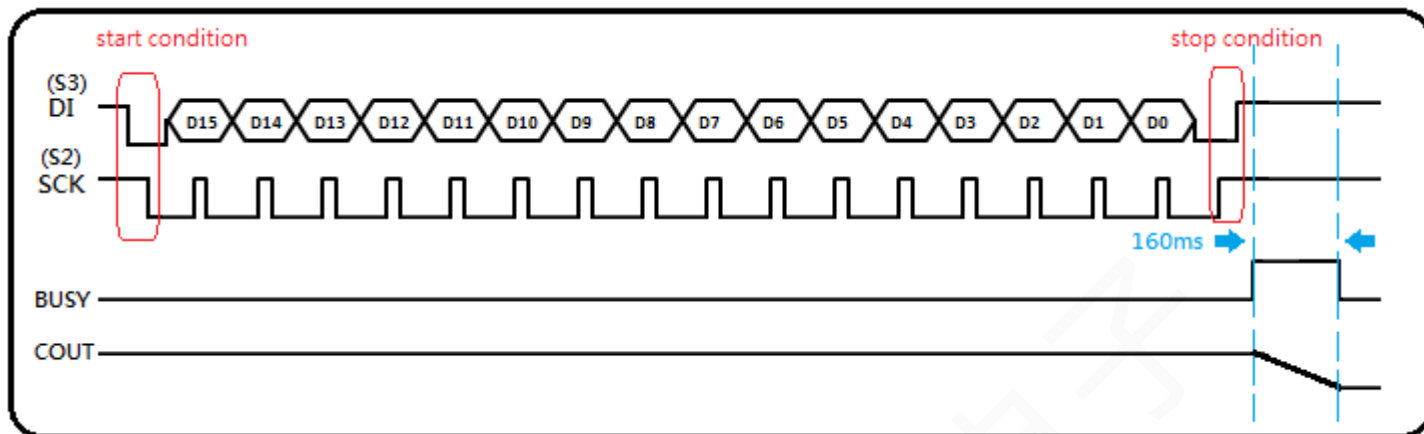
```
void I2Cdata(unsigned int cmd)
```

```
{  
    unsigned int mask = 0;  
    for(mask = 0x8000;mask > 0;mask >>=1)  
    {  
        SCK = 0;  
        DI = (cmd & mask) ? 1:0;  
        SCK = 1;  
    }  
    SCK = 0;  
}
```

```
void SoundChip_Play(unsigned int cmd)
```

```
{  
    unsigned int cmd = 0;  
    cmd = 0x9800 + addr;  
    SCK = 1;    //开始条件  
    DI = 1;    //  
    DI = 0;    //  
    //-----//  
    I2Cdata(cmd); //下播放指令（用play指令）  
    //-----//结束条件  
    DI = 0;    //  
    SCK = 1;    //  
    DI = 1;    //  
    Delay20us(); //需要等待20us之后才能去判断sbt脚位的反馈信号  
}
```

Step F : CMD PD1 (B400h) / PD2 (B800h) 此指令位休眠指令



```
void I2Cdata(unsigned int cmd)
```

```
{
    unsigned int mask = 0;
    for(mask = 0x8000;mask > 0;mask >>=1)
    {
        SCK = 0;
        DI = (cmd & mask) ? 1:0;
        SCK = 1;
    }
    SCK = 0;
}
```

```
void SoundChip_DeActive(unsigned int cmd)
```

```
{
    while(FULL); //wait the last voice group be loaded
    while(BUSY); //wait the end of last voice group

    SCK = 1; //开始条件
    DI = 1; //
    DI = 0; //
    //-----//
    I2Cdata(cmd); //下休眠指令（用指令PD1/PD2）
    //-----//结束条件
    DI = 0; //
    SCK = 1; //
    DI = 1; //
    while(BUSY); //wait end of ramp (需等待160ms之后才会进入
    休眠)
}
```

```

void main()
{
    //以下是 p w m输出模式的用 p l a y 指令的参考例子.
    //
    unsigned char CheckBtn = 0;
    InitPortD();
    InitPortB(); // 初始化mcu io
    Delay10ms(); //等待复位时间&初始化时间.
    while(1)
    {
        //要播放声音.
        PushBtn1(&CheckBtn); //判断是否有按键按下.
        //-----
        if(CheckBtn == 1) //按键按下.
        {
            SoundChip_Active(0xa400); //下唤醒指令，唤醒语音ic，进入工作状态
            //-----
            //Step 2 : CMD Play
            SoundChip_Play(0); // 下播放指令，播放地址位00H的声音。
            while(BUSY); //判断busy信号为低时说明声音已经播放完了， busy为高时说明声音还没播放完.
            SoundChip_Play(1);
            while(BUSY);
            SoundChip_Play(2);
            while(BUSY);
            //-----
            //When you do not want to play voices
            //Step 5 : CMD PD1
            SoundChip_DeActive(0xb400); //等待声音播放完之后，进入休眠下休眠指令，进入休眠之后静态工作电流小于
            // 5 u a

            CheckBtn = 0;
        }
    } //while(1)
}

```

```

void main()
{
    //以下是 p w m输出模式的用 l o a d 指令的参考例子.
    //
    unsigned char CheckBtn = 0;
    InitPortD();
    InitPortB();
    Delay10ms();           //等待复位时间&初始化时间
    while(1)
    {
        //
        PushBtn1(&CheckBtn); //判断是否有按键按下.
        //-----
        if(CheckBtn == 1)     //按键按下.
        {
            //-----
            //Step B : CMD PU1
            SoundChip_Active(0xa400); //下唤醒指令，唤醒语音ic，进入工作状态

            //-----
            //Step C : CMD Load
            SoundChip_Load(0); // 下播放指令，播放地址位00H的声音
            while(FULL);       //当full信号为高时说明，缓存器里的缓存的声音地址已经占有，为低时就没有了。
            SoundChip_Load(1);
            while(FULL);
            SoundChip_Load(2);
            while(FULL);
            //-----
            //When you do not want to play voices
            //Step F : CMD PD1
            SoundChip_DeActive(0xb400); //等待声音播放完之后，进入休眠下休眠指令，进入休眠之后静态工作电流小于
            // 5 u a

            CheckBtn = 0;
        }
    }
}

```