Buildwin
建荣科技
Member of AppoTech Group

AX2226 & AX2227 FAQ

Release Date：2014-10-10

8/F 3D Tech Building,NO.184 Bailian Road,
JiDa Zhuhai,GuangDong,PRC
珠海市吉大白莲路184号立体科技大厦8楼

Tell电话：(0756)3882190
3882191/3882192
Fax传真：(0756)3882193
info@buildwin.com.cn

www.appotech.com
www.buildwin.com.cn

目录

拨打电话后音乐模式调 EQ 全噪音问题（V092、V093）

更新时间：2014-1-26

- **问题现象**

在蓝牙模式下拨打电话后，切到音乐模式播放 U 盘或 SD 卡，按按键调 EQ 后，输出全是噪音，只有 EQ0（EQ 模块是关闭的）声音正常；

通常没有遥控器的方案都没有 EQ 键，默认使用 EQ0，所以不受影响。

- **解决方法**

主要是因为打完电话后没有调用 mp3en_exit()，导致 EQ 模块受影响，程序修改如下：

```
#pragma location="BT_API_SEG"
void bt_exit_hshf(void)
{
#if !BT_CALL_PRIVATE
    dac_fade_out(0);              //先淡出DAC，以消除DAC啪声
    dac_anl_fade_wait();
    dac_disable();
#endif
    //printf("bt_exit_hshf\n");
    sco_farpcm_len = 0;
#if !BT_CALL_PRIVATE
    mp3en_stop();
    mp3en_exit();                 //2014-01-25, 修正接听电话后音乐模式调EQ噪音问题
#endif
    music_unmute();
    dac_enable();
    dac_fade_in(1);
```

双设备时在其他模式拔掉当前设备再切到音乐模式语音播报两次（V092、V093）

更新时间：2014-1-26

- **问题现象**

U 盘和 SD 卡在线时，当前播放 SD 卡，按模式键播 U 盘，再按模式键切到 FM 模式，然后拔掉 U 盘，再切回音乐模式，会播报两次“play by sd card”；

- **解决方法**

在 task_music_enter 中，如果当前设备未被激活，则需要调用 device_change()和 fs_init()，程序修改如下：

```
//音乐任务初始化
#pragma location="TASK_MUSIC_SEG_1"
void task_music_enter(void)
{
    led_mode_off();
    led_busy();

    set_sys_clk(SYS_48M);


    if (device_need_activate(DEVICE_SDMMC) || device_need_activate(DEVICE_UDISK) || !device_is_actived(fs_cur_dev())) {
        draw_wait();
        device_change();                            //扫描设备
        if (fs_need_mounted()) {
            draw_wait();

            fs_init();

        }
    }
```

其他模式下插入 SD 卡仍播 U 盘（V092）

更新时间：2014-1-2

- **问题现象**

V092 版本，程序中定义了 FIRST_PLAY_UDISK 宏，U 盘在线时，在其他模式下插入 SD 卡，进入音量模式后播放 U 盘音乐。

- **解决方法**

在 task_music_enter 中，如果有新设备需要激活则不优先播放 U 盘，修改如下：

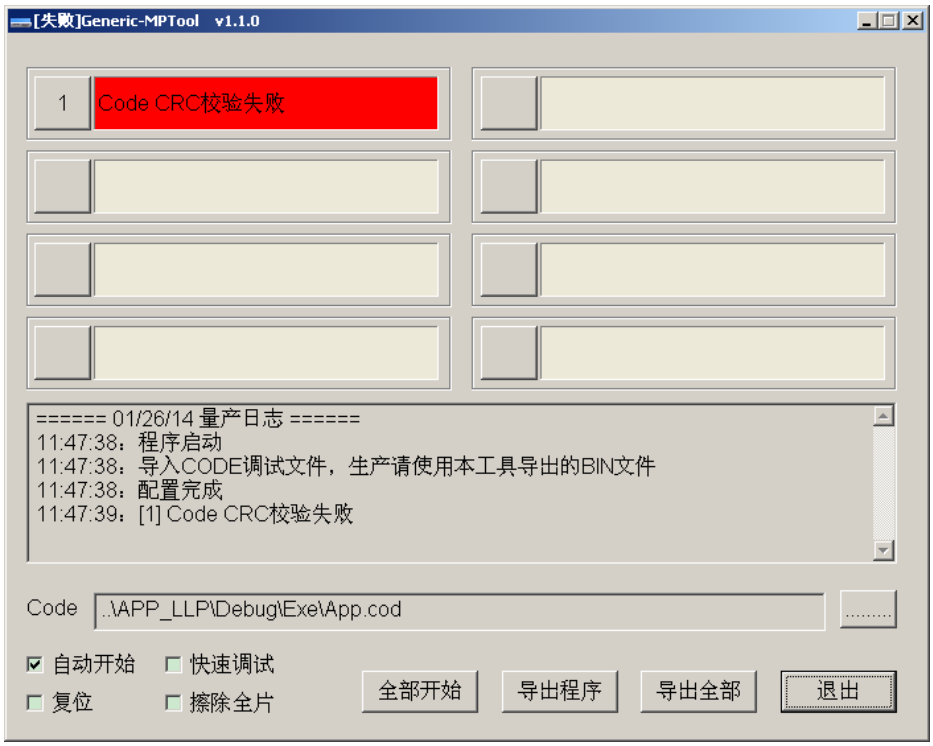
```
306     set_sys_clk(SYS_48M);
307     if (device_need_activate(DEVICE_SDMMC) || device_need_activate(DEVICE_UDISK
308         draw_wait();
309         device_change();                                //扫描设备
310         if (fs_need_mounted()) {
311             draw_wait();
312             //set_sys_clk(SYS_48M);                     //提高主频，加速
313             fs_init();
314             //set_sys_clk(SYS_24M);
315         }
316     }
317 #if FIRST_PLAY_UDISK                                    //2014-1-2, 避免u盘在线时，
318     else if(device_is_actived(DEVICE_SDMMC) && device_is_actived(DEVICE_UDISK))
319         if(fs_cur_dev() != DEVICE_UDISK){
320             fs_sel(DEVICE_UDISK);
321             fs_init();
322         }
323     }
324 #endif
```

MPTool 提示“Code CRC 校验失败”（V092、V093）

更新时间：2014-1-26

- **问题现象**

使用 V1.1.2 之前的 MPTool 下载 cod 文件时，提示“Code CRC 校验失败”。



- **解决方法**

2227 的 Flash 空间要求是 cod 大小+128k byte，旧版本的 MPTool 没有检查是否有预留 128k 空间，存在生产后才发现 Flash 空间不足的风险。在 V094 上添加了预留 128K 空间的设置，会出现旧版本 MPTool 不兼容问题，需要更新到 MPTool V1.1.2。

WAV 语音资源 8k 8bit 音质差问题

更新时间：2014-4-1

- **解决方法**

使用 V093 及以后版本的固件，并且更新“MP3RESTOOL”到 V1.0.0.2。更新后可以支持 8k、16k 和 32k（8bit 或 16bit）的 WAV 语音资源。

- **V092 解决方法**

使用 V092_FIX 版本的 llp_lib.r51，然后修改如下程序：

文件 api_music.h

,MP3\2220SP30037\branches\2227\APP_LLP\api\api_music.h

)14-1-2 14:36:41 10,626 bytes C,C++,C# Source ANSI PC

```
116 };
117
118 extern struct
119 {
120     u32 addr;//资源首地址
121     ul6 len;//资源大小
122     ul6 pos;
123     u8 spr;//采样率
124     u8 bits;//采样精度
125 }mp3res;
126
127 __near_func music_get_status(void);      //获取音乐的播放状态
```

文件 **BtApi.c**

```
1110 #if TASK_USBBT_EN
1111         sbc_pcm_rptr = ua_pcm_buf;
1112         sbc_pcm_wptr = ua_pcm_buf;
1113         sco_samplel = 0;
1114         bt_voice_pcmrd = (void*)sco_farpcm;
1115         bt_voice_pcmwr = (void*)sco_farpcm;
1116         resample_filter_init();                  //USB Dongle由8k转48K需要
1117 #else
1118         AUCON0 = 0x06;  //RST
1119         AUCON9 = 0x04;
1120         player_set_dac(mp3res.spr);              //2014-1-1, 支持多采样率的
1121 #endif
1122         bt_voice_state = BT_VMS_PLAYING;
1123         break;
1124     }
```

```
1175
1176            //2014-1-1, 支持8bit/16bit的WAV语音资源
1177            if(mp3res.bits == 8) {
1178                for(u16 i=0; i != 512; i++){
1179                    while(!(AUCON7 & BIT(7)));
1180                    AUCON5 = ptr[i];
1181                    AUCON5 = 0x00;
1182                    while(!(AUCON7 & BIT(7)));
1183                    AUCON5 = ptr[i];
1184                    AUCON5 = 0x00;
1185                }
1186            } else {
1187                for(u16 i=0; i != 512; i+=2){
1188                    while(!(AUCON7 & BIT(7)));
1189                    AUCON5 = ptr[i+1];
1190                    AUCON5 = ptr[i];
1191                    while(!(AUCON7 & BIT(7)));
1192                    AUCON5 = ptr[i+1];
1193                    AUCON5 = ptr[i];
1194                }
1195            }
1196        }
1197        os_free(ptr);
1198 #endif
```

系统音量为 0 时语音播报不完整（V092）

更新时间：2014-1-2

- **问题现象**

  V092 版本，系统音量为 0 时播报语音，前面约 0.5s 的语音丢失。

- **解决方法**

  修改 music_unmute 函数，系统音量为 0 时，也需要延时。另外，使用 8002 功放时，需要定义 OPA_8002_EN。

```
228 //非静音，正常播放，因为调用了delay_5ms，必须开中断后才能调用
229 void music_unmute(void)
230 {
231     if(sys_ctl.mute_flag || sys_ctl.volume==0) { //避免重复调用时延时久
232         sys_ctl.mute_flag = 0;
233 #if OPA_8002_EN
234         WATCHDOG_CLR();
235         delay_5ms(100);      //使用8002功放时，解MUTE后要延时0.5s左右才有
236 #else
237         delay_5ms(10);       //2014-1-2, 避免某些功放出声音慢
238 #endif
239     }
240 }
241
```

如何生成语音资源

更新时间：2014-2-24

- **解决方法**

1) 把所要添加的音乐文件复制到 MP3RESTOOL 软件所在目录下。注意事项：
   a. 蓝牙模式下使用 WAV 语音资源，非蓝牙模式下使用 MP3 语音资源。
   b. 音乐文件的文件名必须为英文名。
   c. V092 之前版本蓝牙模式下只支持 8k 8bit 的 WAV 语音资源，V093 可以支持 8k、16k、32k 的（8bit 或 16bit）的语音资源；
2) 打开 MP3RESTOOL 软件，点击“生成语音播报表”，自动生成 mp3res.h 和 mp3res.bin 文件；

3) 把 mp3res.bin 拷贝到 APP_LLP\Debug\Exe\bin 目录下，把 mp3res.h 文件拷贝到 APP_LLP\user\mp3res 目录下；
4) 在需要调用语音的地方调用语音播放函数，语音播放函数主要有 3 个 ：

```
void mp3_res_play(u8 music_name);          //播报语音（在非蓝牙模式下）
void mp3_res_play_wait(u8 music_name);     //播报语音，并等待播放完毕（在非蓝牙模式下）
void bt_voice_put(u8 msg);                 //播报语音（在蓝牙模式下）
```

如何添加自定义语音资源

更新时间：2014-1-2

- **解决方法**

在 V093 及以后版本的固件中，更新了 makecode.exe 和资源目录结构，可以通过配置文件中的宏 MP3RES_ID 选择不同的语音资源。例如，SDK 中自带了中文、英文两种语音资源。

1) 在配置文件中定义 MP3RES_ID=1 时，使用中文语音资源

   资源文件位置：APP_LLP\Debug\Exe\bin\mp3res_01.bin
   头文件位置：APP_LLP\user\mp3res\mp3res_01.h
2) 在配置文件中定义 MP3RES_ID=2 时，使用英文语音资源

   资源文件位置：APP_LLP\Debug\Exe\bin\mp3res_02.bin
   头文件位置：APP_LLP\user\mp3res\mp3res_02.h
3) 添加自定义语音资源（以 0x80 为例）

   在配置文件中定义 MP3RES_ID=0x80。
   添加自定义资源文件：APP_LLP\Debug\Exe\bin\mp3res_80.bin

添加自定义资源头文件：APP_LLP\user\mp3res\mp3res_80.h

修改头文件：APP_LLP\user\ user_mp3res.h

```
#ifndef __USER_MP3RES_H_
#define __USER_MP3RES_H_

#if MP3RES_ID == 0x00
    #include "mp3res_00.h"
#elif MP3RES_ID == 0x01      //DEFAULT_CN
    #include "mp3res_01.h"
#elif MP3RES_ID == 0x02      //DEFAULT_EN
    #include "mp3res_02.h"
#elif MP3RES_ID == 0x80      //USER
    #include "mp3res_80.h"
#endif


#endif
```

如何增减 bank 空间

更新时间：2014-2-24

- **解决方法**

1) 这里以由 0x30 改成 0x60 个 bank 为例，修改前后对比如下，

D:\hexscript修改前.hs

```
;*************************************************************
;* Module    : HexScript
;* File      : hexscript.hs
;* Author    : Hanny
;* Email     : coldney@yahoo.com.cn
;* Function  : 用于生成下载的bin文件。修改该文件时，建议进行全
;*************************************************************
load(hex, APP_LLP.a51);                    载入
map(0, 0, 4000);                           映射(0-3fff) ->
set(out, 16, le);                          设置输出位宽为16E
save(hex, APP_LLP.hex);                    输出Hex
mapx(11800, 2000, 800, 10000, 800, 30);    映射0x30个Bank
;map(11000, 1A000, 800);                   蓝牙任务的公共区
map(864800, 1A800, 800);                   DCODE4 蓝牙SCO
map(854800, 1B000, 1800);                  DCODE3 蓝牙
map(844000, 1C800, 1800);                  DCODE2 FM
map(834000, 1E000, 1800);                  DCODE1 USB设备
set(out, 8, le);
save(bin, APP_LLP.bin);

clear(out);
map(FF0000, 0, 20);
save(bin, makecfg.bin);
```

D:\hexscript修改后.hs

```
;*************************************************************
;* Module    : HexScript
;* File      : hexscript.hs
;* Author    : Hanny
;* Email     : coldney@yahoo.com.cn
;* Function  : 用于生成下载的bin文件。修改该文件时，建议进行全
;*************************************************************
load(hex, APP_LLP.a51);                    载入
map(0, 0, 4000);                           映射(0-3fff) ->
set(out, 16, le);                          设置输出位宽为16E
save(hex, APP_LLP.hex);                    输出Hex
mapx(11800, 2000, 800, 10000, 800, 60);    映射0x60个Bank
;map(11000, 32000, 800);                   蓝牙任务的公共区
map(864800, 32800, 800);                   DCODE4 蓝牙SCO
map(854800, 33000, 1800);                  DCODE3 蓝牙
map(844000, 34800, 1800);                  DCODE2 FM
map(834000, 36000, 1800);                  DCODE1 USB设备
set(out, 8, le);
save(bin, APP_LLP.bin);

clear(out);
map(FF0000, 0, 20);
save(bin, makecfg.bin);
```

mapx(11800, 2000, 800, 1000, 800, 60);

首先，前 3 个参数是指将逻辑地址 0x11800 映射到 Flash 物理地址 0x2000，长度 0x800 字节；接着，0x1000 和 0x800 分别是指映射后逻辑地址和物理地址偏移长度；最后，0x60 指总共要映射的次数。

这样，物理地址 0x2000 经过 0x60 次的映射偏移后，就变成 0x2000 + 0x60*0x800=0x32000，也就是说，Flash 空间从 0x2000~0x31fff 是存放 bank 代码。

map(11000, 32000, 800);

将逻辑地址 0x11000 映射到 Flash 物理地址 0x32000，长度 0x800 字节；

经过 12 行 mapx 映射后，0x32000 之前的 Flash 地址都已经存放了内容，所以 13 行 Flash 地址应该改为 0x32000；以此类推，修改后面的 Flash 地址。

2) 修改 APP_LLP.xcl。

```
-D_BANK_START=0x11800
-D_BANK_END=0x11FFF
-D_BANK_STEP=0x10000
-D_BANK_NUM=0x60
```

3) 修改源文件，将 **load_code、load_comm、sbc_load_comm** 这 3 个函数的地址改成 hexscript.hs 的相应地址，对比 hexscript.hs 修改前后的可发现一下公共区地址有变化。

所以，load_comm(0x1a0)要改成 load_comm(0x320)；

sbc_load_code(0x1b0, 3)要改成 sbc_load_code(0x330, 3)；

4) V102 及之后版本，可跳过步骤 3，只需要修改 macro.h 中的 BANK_NUM 宏即可。

如何修改 Timer2 中断

更新时间：2014-5-23

- **解决方法**

V100 的 llp_lib.r51 库添加了修改 Timer2 中断支持，在 APP_LLP 中定义 timer2_isr_do()函数时则用 APP_LLP 的 timer2_isr_do 函数，未定义时则用 LIB 的 timer2_isr_do 函数。该库可以用到 V096/V097/V098/V099。

将 timer2 改成 250us 中断示例：

```
IAR_DATA_A u8 tm2_cnt @ 0x65;//未使用的 DATA
//初始化 timer2 为 250us 中断，要在 timer2_init 之后调用
void timer2_250us(void)
{
    //TMR2PRL = 0x76;          //0x177 为 1ms
    //TMR2PRH = 0x01;
    TMR2PRL = 0x5d;          //0x05d 为 250us
    TMR2PRH = 0x00;
    TMR2CNTL = 0;
    TMR2CNTH = 0;            //Clear CNT
```

```
    tm2_cnt = 0;
}

//在 APP_LLP 中定义 timer2_isr_do 函数
__near_func void timer2_process(void);
__near_func void timer2_isr_do(void)
{
    //my_process();          //250us 中断，可以处理其它事情，但不能阻塞
    if(++tm2_cnt > 3) {      //4*250us = 1ms
        tm2_cnt = 0;
        timer2_process();    //原 timer2 处理函数，必须 1ms 调用一次
    }
}
```

如何修改 UART0 波特率

更新时间：2014-5-23

- **解决方法**

V100 的 llp_lib.r51 库添加了修改 UART0 波特率支持，因为在 LIB 中切时钟时会重设 UART0 波特率，默认是 115200，若要修改波特率，可以在 APP_LLP 中定义 set_clk_div ()函数，使 LIB 切时钟时将 UART0 设为其它波特率。该库可以用到 V096/V097/V098/V099。

以下是波特率为 38400 的例子：

1）添加 UART0 相关函数

```
#define MY_UART_BAUD          38400                                  //定义波特率
#define UART_BAUD             (48000000/8/MY_UART_BAUD-1)      //自动计算
#define UART_BAUDH            ((48000000/8/MY_UART_BAUD-1)>>8)//自动计算

//在 APP_LLP 中定义 set_clk_div 函数
void set_clk_div(u8 sys_v2)
{
    UARTBAUDH = (UART_BAUDH>>sys_v2);
    UARTBAUD = (u8)(UART_BAUD>>sys_v2);
}

//串口初始化
void my_uart_init(void)
{
    uart_disable();          //把调试用的 printf 关掉，之后不能再调用 uart_enable
    UARTBAUD = UART_BAUD;    //UARTBAUD = sysclk/8/波特率 - 1
    UARTBAUDH = UART_BAUDH;
```

```
    UARTCON = 0x90;          //STOP BIT AND ENABLE UART
    UARTSTA = 0x01;
    P0DIR &= ~BIT(1);        //TX
    P0DIR |= BIT(0);         //RX
}
//从串口接收一个字符，在中断查询
__near_func void my_uart_getchar(void)
{
    if((UARTSTA & 0x20) != 0) {
        UARTSTA &= ~0x20;
        put_msg(UARTDATA);  //保存到 RX_BUF，这里用 put_msg 只是用于测试
    }
}

//输出一个字符到串口，前台等待发送完毕
__near_func void my_uart_putchar(char c)
{
    UARTDATA = c;
    while((UARTSTA & 0x10) == 0) {
        WATCHDOG_CLR();
    }
}
```

2）参考“修改 time2 中断”章节，将 timer2 改成 250us 中断。

3）在 timer2_isr_do 函数中查询接收 UART 数据

```
__near_func void timer2_process(void);
__near_func void timer2_isr_do(void)
{
    my_uart_getchar();       //38400 时至少 250us 查询一次
    if(++tm2_cnt > 3) {      //4*250us = 1ms
        tm2_cnt = 0;
        timer2_process();    //原 timer2 处理函数，必须 1ms 调用一次
    }
}
```

4）测试实例（这里是用 put_msg 作为 RX_BUF，测试时要注释掉按键消息和 1s 消息）：

```
void test_init(void)
{
    。。。
    my_uart_init();
    timer2_init();                  //定时器基本功能初始化
    timer2_250us();
```

```
    ...
    flush_msg();
    set_sys_clk(SYS_48M);                    //初始化完毕，设置时钟

    while(1) {
        u8 msg = get_msg();
        if(msg != 0) {
            my_uart_putchar(msg);
        }
        WATCHDOG_CLR();
    }
}
```

2226_S10 开机 MP3BT 检测误判问题修复

更新时间：2014-4-4

- **解决方法**

在 MP3BT_DECT_REUSE_EN 此宏打开时（2226 项目有开机 MP3BT 检测功能的目前此宏都已经打开），检测初始化宏 MP3BT_DECT_INIT()需要打开上拉，检测结束后宏 MP3BT_DECT_END()需要关闭上拉以作他用（2226 项目中用作 SPI_CLK），具体程序修改如下(添加红色部分代码即可)：

```
//上电时 BT/MP3 检测
#define    MP3BT_DECT_INIT()            P0DIR |= BIT(5);P0PU0 |= BIT(5);\
asm("nop");asm("nop");asm("nop");asm("nop");\
asm("nop");asm("nop");asm("nop");asm("nop");
#define    MP3BT_DECT_IS_BT()           (P0 & BIT(5))
#define    MP3BT_DECT_END()             P0PU0 &= ~BIT(5);P0DIR &= ~BIT(5)
```

开关 IE_TM2（小米 2S 播放卡顿、无声）问题

更新时间：2014-3-5

- **问题现象**

连接小米 2S 及部分手机，播放蓝牙音乐时，出现卡顿、无声和断线现象，断开连接后无法再次连接，需要切模式或重启后才能连接。

- **解决方法**

主要是因为处理临界条件时，关掉了次高优先级的 timer2 中断，此时若进入低优先级的软中断和解码中断则会阻塞主循环，导致 timer2 中断延迟打开。timer2 阻塞超过 1ms 后蓝牙 RX 缓冲区会溢出，出现卡顿、断线等现象。

程序中所有关 IE_TM2 的位置都要修改，主要在 led.c、led_5c7s.c、led_7p7s.c 这几个文件，user_setting.c、task.c 有一些背光函数，虽然没有调用，也可以一起修改，避免日后不小心调用了。

若处理临界的时间较短，可以改成关 IE_EA；例如：

```
#pragma location="DISP_LED_SEG"
void led_blue_100ms(void)
{
    bool ie_ea = IE_EA;      //2014-3-5, 修正小米2S及部分手机播歌停顿、无声问题
    IE_EA = 0;               //2014-3-5, 短时间处理，直接关IE_EA
    led_blue.speed = 2;
    led_blue.sp_off = 2;
    led_blue.sp_on = 2;
    led_blue.flag = LED_FLASH;
    IE_EA = ie_ea;
}
```

若处理临界的时间较长，可以改成关除了 UART 中断以外的其他中断，例如：

```
//根据7脚LED的真值表进行映射（每段输出高的IO记为COM端 7*7）COM:0~6  SEG:0~6
#pragma location="DISP_LED_7P7S_SEG"
void led_7p7s_value_set(void)
{
    u8 ie0, ie1;
    bool ie_ea = IE_EA; //2014-3-5, 修正小米2S及部分手机播歌停顿、无声问题
    IE_EA = 0;          //关总中断，保护好IE0/IE1后再打开
    ie0 = IE0;          //保存IE0
    ie1 = IE1;          //保存IE1
    IE0 = 0;            //关掉IE0
    IE1 &= BIT(6);      //关掉IE1，除了UART中断（因为这段处理较长，不能一直关IE_EA，否则会阻塞UART中断）
    IE_EA = ie_ea;      //恢复总中断

    my_memset((u8 *)pin_disp_buf, 0x00, 7);

/************ SEG ****************** COM ******** SEG(out)  *******/
#if LEDSEG_TYPE_SELECT == LEDSEG_7P7S_FM

    if(disp_buf[0] & LED_BIT_A)     pin_disp_buf[1] |= BIT(0); //设为1，扫描时对应此SEG位设为输出(F
    if(disp_buf[0] & LED_BIT_B)     pin_disp_buf[2] |= BIT(0);
    if(disp_buf[0] & LED_BIT_C)     pin_disp_buf[3] |= BIT(0);
    if(disp_buf[0] & LED_BIT_D)     pin_disp_buf[4] |= BIT(0);
    if(disp_buf[0] & LED_BIT_E)     pin_disp_buf[5] |= BIT(0);
    if(disp_buf[0] & LED_BIT_F)     pin_disp_buf[6] |= BIT(0);
    if(disp_buf[0] & LED_BIT_G)     pin_disp_buf[2] |= BIT(1);

    ...

#endif

    IE0 = ie0;          //先恢复IE0，此时会关IE_EA
    IE1 = ie1;          //再恢复IE1
    IE_EA = ie_ea;      //恢复总中断
}
```

SD 升级保留蓝牙地址

更新时间：2014-4-15

- **问题现象**

使用 SD 升级时，蓝牙地址没有自增功能，若在 MPTool 中选择了配置蓝牙地址，升级过后样机蓝牙地址都相同。

- **解决方法**

在 V099 版本中，修改了 SD 升级程序 update_01.bin、update_02.bin，以后需要升级 V099 开发的程序时，将 bin 文件命名为 upd_app.bin 放入 SD 卡中，可以只升级程序，而保留初始量产时 SPIFlash 中的蓝牙地址。详见《SD 卡升级模式.PDF》。

若样机中烧写的是 V099 之前的程序，则只能在 MPTool 中不配置蓝牙地址，SD 升级过后，则使用 OTP 中的随机地址；或者连接 USB，通过 MPTool 重新量产，MPTool V1.1.5 有地址自增功能。

V099 中的 update_01.bin、update_02.bin 可以更新到 V096/V097/V098 的 SDK 中，使新开发的方案也有此功能。对于已生产的方案，更新后可能需要使用强制升级，详见《SD 卡升级模式.PDF》。

SD 升级时功放被 MUTE 无法判断升级结果

更新时间：2014-4-15

- **问题现象**

若修改了 MUTE 引脚（2227 使用 P35、2226 使用 P21），SD 升级时功放没有解 MUTE，无法通过声音判断是否升级成功。

- **解决方法**

在 V099 版本中，改进了 SD 升级程序 update_01.bin、update_02.bin，SD 升级后会自动复位，而且可在 show_sdupd 显示升级结果。

V099 中的 update_01.bin、update_02.bin 可以更新到 V096/V097/V098 的 SDK 中，并添加以下函数，使新开发的方案也有此功能。对于已生产的方案，更新后可能需要使用强制升级，详见《SD 卡升级模式.PDF》。

```
#pragma location="INIT_SEG"
void main(void)
{
    。。。
#if POWER_ON_VOICE
    dac_enable();
    mp3_res_play_wait(RES_MP3_START);
#endif
    show_sdupd();
    run_task();
    while (1) {
    }
}

void show_sdupd(void)
```

```
{
    u8 tmp, tmp1, tmp2;
    tmp1 = *(u8 *)0x3efc;
    *(u8 *)0x3efc = 0x00;
    tmp1 ^= *(u8 *)0x3efd;
    tmp2 = *(u8 *)0x3efe;
    tmp2 ^= *(u8 *)0x3eff;
    tmp = tmp1 & tmp2;
    tmp1 = *(u8 *)0x3efd;
    tmp2 = *(u8 *)0x3efe;
    if(tmp == 0xff && (tmp1 == 0xaa || tmp1 == 0x99) && tmp2 < 10) {
        //printf("sdupd: %02x\n", tmp2);
        switch(tmp2) {
        case 0x00://升级成功
        case 0x03://升级文件与板上程序相同
        case 0x05://强制升级文件与板上程序相同
        case 0x07://强制升级成功
            //printf("sdupd ok\n");
            //dac_enable();
            //mp3_res_play_wait(RES_SDUPD_OK);
            break;
        case 0x01://升级文件大小错误
        case 0x02://升级文件一致性错误
        case 0x04://升级失败
        case 0x06://强制升级失败
            //printf("sdupd fail\n");
            //dac_enable();
            //mp3_res_play_wait(RES_SDUPD_FAIL);
            break;
        case 0x08://升级文件不存在蓝牙信息
            //printf("no btinf\n");
            //dac_enable();
            //mp3_res_play_wait(RES_SDUPD_FAIL);
            break;
        case 0x09://存在两个升级文件
            //printf("two files\n");
            //dac_enable();
            //mp3_res_play_wait(RES_SDUPD_FAIL);
            break;
        }
    }
}
```

如何屏蔽蓝牙通话功能

更新时间：2014-4-4

- **解决方法**

若需要屏蔽手机音频的连接，在 BtApi.c 文件开头处有如下两个宏定义：

```
#define    HEADSET_ENABLE                  1     //手机音频连接使能
#define    A2DP_ENABLE                     1     //媒体音频连接使能
```

把 HEADSET_ENABLE 定义为 0 即可屏蔽手机音频的连接；

若仅仅是需要屏蔽通话功能而不需要屏蔽手机音频的连接，则把强制秘密接听宏 BT_CALL_PRIVATE 打开即可，此宏在 user_config.h 头文件里面有默认定义（默认定义为 0），如下所示：

```
//是否强制使用私密接听（手机端接听）
#ifndef    BT_CALL_PRIVATE
#define    BT_CALL_PRIVATE                 0
#endif
```

此宏在针对具体案子的配置头文件里面一般也有定义，如需要强制秘密接听时，可以直接在 user_config.h 头文件里面把宏 BT_CALL_PRIVATE 定义为 1，也可以在针对具体案子的配置头文件（如 user_config_2226_S10.h）里面直接把宏 BT_CALL_PRIVATE 定义为 1，推荐使用第二种方法，因为在 user_config.h 里面定义的宏绝大部分都是默认功能，尽量不要去修改，以做到 user_config.h 里面的默认配置对每个具体案子都能兼容。

注：1、屏蔽手机音频的连接是指直接把样机的手机音频连接屏蔽掉，样机只提供媒体音频的连接功能，其中手机音频的连接用于通话功能，而媒体音频的连接用于音乐视频等媒体播放功能；

2、强制秘密接听是指把样机的通话功能屏蔽掉，样机与手机连接后所有与通话相关的功能例如接听、挂断、回拨、拒接电话等操作都只能在手机端进行，也不能使用样机的 MIC 及喇叭进行通话，但样机仍然提供手机音频的连接（不屏蔽手机音频的连接时）。

部分手机蓝牙音乐暂停后按播放无响应问题（V099）

更新时间：2014-4-22

- **问题现象**

部分手机如 IPHONE、三星等蓝牙音乐播放暂停后，样机上按播放键无响应，要过几秒后按播放键才有响应。

- **解决方法**

更新一下最新蓝牙库即可。

部分手机通话响铃时有沙沙声问题（V099）

更新时间：2014-4-22

- **问题现象**

  部分手机如 IPHONE 蓝牙上样机后，外面打电话进来响铃时会伴随着沙沙的噪声。

- **解决方法**

  1、在 BtApi.c 文件开头处找到如下代码：

  ```
  extern u8 music_ibuf[0xa08];
  ```

  修改为：

  ```
  IAR_XDATA_A u8 bt_voice_ibuf[0x200] @ 0x52390;
  ```

  2、在 BtApi.c 文件的函数 void bank_process_bt_voice()里面分别找到如下两行代码：

  ```
  u8 *mem = music_ibuf;
  u8 *ptr = music_ibuf;
  ```

  分别修改为：

  ```
  u8 *mem = bt_voice_ibuf;
  u8 *ptr = bt_voice_ibuf;
  ```

蓝牙断线频繁问题修复（V099）

更新时间：2014-5-6

- **问题现象**

  蓝牙音乐播放或通话煲机断线较为频繁。

- **解决方法**

  更新到最新蓝牙库即可；

连接测试盒状态判断及 MUTE 控制（V099）

更新时间：2014-5-6

- **问题现象**

  目前连接测试盒时没有一个统一的标志来判断是否处于连接测试盒状态，而连接测试盒时 BT_STATE 没有正常切换，此时如果根据 BT_STATE <= StateConnected 就直接 MUTE 住功放会导致 1K 测试声音也被 MUTE 住。

- **解决方法**

  1、更新到最新蓝牙库；最新蓝牙库增加了一个变量 Test_Mode_Flag 来标志是否处于连接测试盒状态。

  （1）当 Test_Mode_Flag = 0 时表示不处于连接测试盒状态；

  （2）当 Test_Mode_Flag !=0 时表示处于连接测试盒状态，其中连接测试盒状态又分为三个子状态：

  ①当 Test_Mode_Flag = 1 时表示处于 1K 测试声音播放状态；

  ②当 Test_Mode_Flag = 2 时表示处于按键测试状态；

  ③当 Test_Mode_Flag = 3 时表示处于 MIC 测试状态。

  2、增加了此标志后蓝牙模式下的 MUTE 控制及连接测试盒时灯的闪烁控制判断可以作如下相应修改：

（1）蓝牙模式下的 MUTE 控制修改：在 task_bt.c 文件的 void task_bt_deal_msg(u8 msg)函数里面找到如下 MUTE 控制代码：

```
    case QSYSTEM_1S:            //BT_IS_SLEEP()无需查询太频繁
        if(BT_STATE <= StateConnected && !avdtp_play_status){
#if AUTO_STANDBY_EN
            BT_IS_SLEEP_INIT();
            if(BT_IS_SLEEP()){
                BT_IS_SLEEP_END();
                if(sys_ctl.sleep_cnt == 0) {
                    sys_ctl.sleep_cnt = STANDBY_TIME;
                }
            } else {
                BT_IS_SLEEP_END();
                sys_ctl.sleep_cnt = 0;
            }
#endif
            music_mute();
        } else {
            if(sys_ctl.mute_flag && !key_mute_flag) {
                if(sys_ctl.volume) {
                    music_unmute();
                    dac_fade_in(1);
                }
            }
        }
    ……
```

把上面的 MUTE 判断条件 if(BT_STATE <= StateConnected && !avdtp_play_status) 改为：

```
if(BT_STATE <= StateConnected && ! Test_Mode_Flag)
```

（2）连接测试盒时灯的闪烁控制判断修改：在 task_bt.c 文件的 void task_bt_state(void) 函数里面找到如下连接测试盒时灯的闪烁控制代码：

```
    if(BT_STATE_pre != BT_STATE){
        ……
    }
    else{    //测试盒测试时灯闪烁控制
        if(BT_STATE < StateConnected && avdtp_play_status){ //播放 1K 正弦波中
            led_bt_play();
        }
        else if(Test_Mode_Flag){      //MIC 或按键测试
            led_bt_connect();
        }
    }
```

修改后如下所示：

```
    if(BT_STATE_pre != BT_STATE){
        ……
    }
    else{    //测试盒测试时灯闪烁控制
        if(Test_Mode_Flag == 1){ //播放 1K 正弦波中
            led_bt_play();
        }
        else if(Test_Mode_Flag == 2 || Test_Mode_Flag == 3){      //MIC 或按键测试
            led_bt_connect();
        }
    }
```

播放 TF 卡有杂音问题修复（V096~V099）

更新时间：2014-5-23

- **问题现象**

  音乐模式下播放 TF 卡会有轻微的读卡杂音。

- **解决方法**

  1、 使用最新版本的 OTP（目前是 1.1.8）。

  2、 更新到最新 SDK_V100 版本的 llp_lib 库（适用于 V096~V099 版本 SDK）；

  3、 在 task_fm.c 的 void task_fm_init(void)函数中找到如下代码：

```
#if FMOSC_REUSE_SDCLK
    SD_CLK_DIR_OUT();
    SDCON0 |= BIT(0);                                   //Enable SDC
    SDBAUD = 0;                                 //24MHz / 2 / (SDBAUD + 1) = 12M
    SDCON0 |= BIT(1);                                   //Keep Clk Outing
#endif
```

  把上面直接设置 SD 波特率的地方均改为调用函数 sd_set_rate0(u8 rate)去设置，修改后如下所示：（注：若程序中还有其他地方也是直接设置 SD 波特率寄存器 SDBAUD 的，也要作同样的修改）

```
#if FMOSC_REUSE_SDCLK
    SD_CLK_DIR_OUT();
    SDCON0 |= BIT(0);                                   //Enable SDC
    sd_set_rate0(0);                            //24MHz / 2 / (SDBAUD + 1) = 12M
    SDCON0 |= BIT(1);                                   //Keep Clk Outing
#endif
```

  函数 sd_set_rate0(u8 rate)需要声明一下，如下所示：

```
extern __near_func void   sd_set_rate0(u8 rate);
void task_fm_init(void)
{
```

```
        ......
    }
```

提高 DAC 输出信噪比和改善通话底噪

更新时间：2014-5-12

- **问题现象**

  蓝牙通话过程中底噪偏大。

- **解决方法**

  DACVDD 引脚（AX2226/AX2227 第 8 脚）的电容由 105 改为 10uF，不但可以改善通话底噪，还可以提高播放音乐时 DAC 输出信噪比。

USBDM 复用 IIC 时出现读卡器拷文件出错问题

更新时间：2014-4-15

- **问题现象**

  USBDM 复用 IIC_CLK，且 SD_DAT 复用 IIC_DAT 时，读卡器模式拷文件时会使 IIC_CLK 和 IIC_DAT 同时翻转，存在一定几率使 IIC_DAT 输出 ACK 信号，引起读写卡出错，尤其是拷大文件的时候。

- **解决方法**

  V099 版本中读卡器添加读写卡重试机制，可有效避免上述现象。

  V099 中的 llp_lib.r51 可以更新到 V096/V097/V098 的 SDK 中，使旧版本 SDK 也具有此功能。

USBDM 复用 IIC 时出现退出 FM 失败问题

更新时间：2014-4-15

- **问题现象**

  USBDM 复用 IIC_CLK 时，插着某些 U 盘偶尔会出现退出 FM 失败，导致退出 FM 模式后仍输出 FM 的声音。

- **解决方法**

  通过示波器分析波形发现，退出 FM 模式时，会导致某些 U 盘把 DM 拉成低电平，使 IIC 通讯时无法把 IIC_CLK 拉高。可添加以下程序解决：

```
#pragma location="FM_INIT"
void fm_off(void)
{
#if IICCLK_REUSE_USB && !USB_GPIO_EN
    USB_DPDM_GPIO_EN();
```

```
#endif
#if IICCLK_REUSE_USB
    USB_DP_OUTPUT();        //修正部分 U 盘在线时退出 FM 时出现 FM 关不掉问题
    USB_DP_HIHG();
#endif
    。。。
}
```

SD 复用 IIC 时出现 FM 初始化失败问题

更新时间：2014-5-23

- **问题现象**

AX2227/AX2226 一般是 IIC 复用 SD 引脚，某些 SD 卡在线时，会拉住 SDDAT，导致 FM 初始化失败。

- **解决方法**

在用完 SD 卡时，调用 sd_stop(1)，使 SD 卡停止传输。同时在 FM 初始化前，先推送一段时间的 SDCLK，使 SD 卡释放 SDDAT。

```
#pragma location="TASK_MUSIC_SEG_1"
void task_music_exit(void)
{
    music_stop();                                   //结束 Music 播放
    if(device_is_actived(DEVICE_SDMMC)){
        sd_stop(1);                 //停止 SD 传输，使卡释放 SDDAT，还可降低卡功耗
    }
    //move_ctl.move_flag=0;
    dac_disable();
}

#pragma location="TASK_FM_INIT"
void task_fm_init(void)
{
    set_sys_clk(SYS_24M);               //FM 模式下工作在 24M 系统时钟
    user_set_volex(0);                  //FM 音源较大，一般不需要 3DB 音量补偿
    led_mode_off();
    led_idle();
    。。。
#if IIC_REUSE_SD          //复用 SD 时先输出一段 SDCLK，使 SD 卡释放 SDDAT
    SDCON0 |= BIT(1);                   //Keep Clk Outing
    delay_5ms(2);
    SDCON0 &= ~(BIT(1) | BIT(0));    //Disable SDC & CLK
#endif
```

```
    ...
}
```

SPIFLASH 干扰蓝牙灵敏度的问题

更新时间：2014-5-23

- **问题原因**

GD(博冠) SPI Flash 对蓝牙信号干扰，其他牌子没有发现该问题。暂时 SPI Flash 原厂没有给出解决方案。

- **解决方法**

1）更换 SPI Flash 牌子。

2）SPI Flash DATA 引脚串 2K 电阻，降低干扰。

3）Layout 时，SPI FLASH 靠近主控布局，远离蓝牙 IC。SPI Flash 地线与主控地相连，避免与蓝牙地有共用回路。

RTC6218 收音在清晰台有哒哒干扰声

更新时间：2014-5-23

- **问题原因**

RTC6218 左右声道直接相连，FM 芯片发热。

- **解决方法**

RTC6218 不支持左右声道直接相连，可以通过 1K 电阻相连或只使用单声道。

更新 btlib_20140610 后出现蓝牙断音

更新时间：2014-6-16

- **问题原因**

2014-6-10 更新的“btlib_20140610.rar”及 2011-6-12 更新的“APP_LLP_V099 支持 V100 的蓝牙库.rar”在改善通话效果时，没有处理好，导致蓝牙播放音乐出现断音现象。

V099 和 V100 更新该蓝牙库后会出现此问题，其他版本不兼容该蓝牙库。

- **解决方法**

使用“btlib_20140616.rar”的蓝牙库，将 btlib.r51 替换到 APP_LLP 目录中。

改善通话效果（V099~V100）

更新时间：2014-6-16

- **解决方法**

1、V100 可以直接使用“btlib_20140616.rar”蓝牙库，将 btlib.r51 替换到 APP_LLP 目录中，改善通话效果。

2、若 V099 要使用“btlib_20140616.rar”，还要相应修改 APP_LLP，可参考“APP_LLP_V099 支持 V100 的蓝牙库_20140616.rar”，主要有以下 4 个文件：BtApi.c，btApi.h，task_bt.c，user_timer.c。

4、 V099 以前版本的 SDK 不支持更新该蓝牙库。

如何微调蓝牙晶振频偏

更新时间：2014-7-7

- **解决方法**

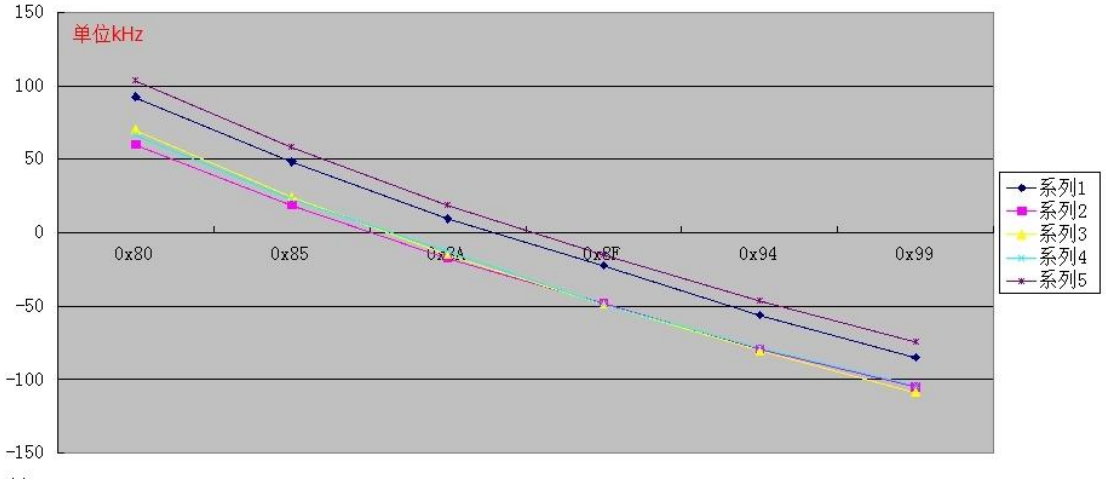
通过修改 RF 参数可以实现频偏的微调，位置在 BT_RFINFO_TABL[]中 0x42 偏移地址上的 bit[4:0]，其中 bit[7]=1，所以该字节取值范围是 0x80~0x9f。

```
IAR_CONST u8 BT_RFINFO_TABLE[128] = {//spi_bt_rfinfo_read
    // 信息头
    'C', 'O', 'N', 'W', 'I', 'S', 'E', '-', '6', '6', '3', '9', 'v', '1', '.', '0',
    // 蓝牙名字
    BT_NAME,
    // 蓝牙地址
    0x56,0x34,0x12,0x99,0x22,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    // 蓝牙 RF 信息
    0x0F,0x40,0x8F,0x41,0x04,0x06,0x04,0x04,0x00,0x80,0x00,0x00,0x80,0x01,0x20,0xD6,
    0x93,0x00,0xF7,0x3F,0x00,0x00,0x00,0x00,0x1F,0x00,0x00,0x00,0x00,0x00,0x9E,0xE4,
    0xFF,0x1E,0x2C,0x1F,0x06,0x42,0x14,0x7E,0x00,0x04,0x07,0x0E,0x1E,0x28,0x00,0x01,
    0x00,0x00,0xF4,0xF1,0x0C,0xC3,0x00,0x16,0x44,0x00,0x2D,0x1C,0x1C,0x24,0xFF,0xFE
};
```

下列数据是实际测量的值，参数由 0x80 至 0x9f 调整，呈现线性变化，参数越大频偏值往更负的方向偏。

参考值	0x80	0x85	0x8A	0x8F	0x94	0x99	0x9F
1	92.6	48.3	9.6	-22.3	-56.1	-84.8	X
2	60	19.2	-17	-47.7	-78.9	-104.9	X

3	70.2	24.8	-14.8	-48.7	-80	-108.4	X
4	66.5	23	-12.7	-48.8	-78.6	-104.1	X
5	104.1	58.6	19.3	-14.8	-46.4	-73.7	-104.7



通话过程中，偶尔出现蓝牙断开后自动回连

更新时间：2014-7-8

- **解决方法**

1) 先排除发射天线不匹配导致，天线不匹配时，一般通信距离较短，而且会出现播歌断续现象。

2) 在 btapi.c 中添加宏定义：#define SCO_SUPERVISION_CHECK，使下面这段代码生效。

```
__near_func void Process_Agc_Data(void)
{
    ......
    if(bt_uart_txlen == 0)
    {
    #ifndef SCO_SUPERVISION_CHECK
        bt_uart_tx(uartTxQ3, 4 + 24*2);    // 24 samples
    #else
    //定义 SCO_SUPERVISION_CHECK 则通话时检测是否通信变差有导致断线的可能并修复。
        extern u8 sco_supervision_cnt;
        if(sco_supervision_cnt < 2) {
            bt_uart_tx(uartTxQ3, 4 + 24*2);    // 24 samples
        }
        ......
    #endif
    }
}
```

手机响铃时回连会导致重启（V100、V102）

更新时间：2014-7-11

- **问题现象**

  手机来电响铃时，开机回连，会导致样机重启。

- **解决方法**

```
void TurnOnRingTone(void)
{
//printf("ON :%d%d\n", HSFRingActive, HSFCallActive);
#ifdef MP3_PLAY
    if(mWorkingMode != WORK_MODE_BT)
    {
        switch_mode_save = WORK_MODE_MP3;
        switch_to_btmode();
    }
#endif
    if(HSHFAPI.RFCOMMConList[0].state != AP_STATE_RFCOMM_FINISHED\
    || HandfreeState != HF_STATE_CONNECT_FINISHED) {    //连接过程中不允许响铃
        return;
    }
    if(HSFRingActive == 0){
        HSFRingActive    = 1;
        g_InComingCall = 1;//响铃
#if !BT_CALL_PRIVATE
        bt_voice_put(BT_VOICE_RING);
#endif
    }
}
```

动态降噪无效问题（V102）

更新时间：2014-7-8

- **问题现象**

  定义宏 BT_DYMANIC_SUPPRESSION=1 时，手机暂停播放也没有淡出 DAC 音量。

- **解决方法**

  打开动态降噪时，变量 sound_detect_ctl.fade_en 只有等于 0x81，才会控制 DAC，这里有一处笔误，需要将 fade_en 修改为 u8 类型。

```
typedef struct {
    void *ptr;                  //检测起始地址
```

```
    u16 samples;            //检测样点数
    u8 delay_cnt;           //检测延时
    u8 fade_en;             //淡入淡出控制使能
} type_sound_detect_ctl;
```

手机音量为 0 时不 MUTE 功放（V102）

更新时间：2014-7-8

- **问题原因**

  由于在 V102 中删除了宏 VOL_CTRL_MUTE。

- **解决方法**

1) 要定义宏 BT_DYMANIC_SUPPRESSION=1 打开动态降噪功能。

2) 修正 V102 动态降噪无效问题。

3) 在 user_io.c 中添加一些代码，并且定义宏 VOL_CTRL_MUTE=1。

```
#if VOL_CTRL_MUTE
    #define HSHF_VOL_IS_ZERO()   (sys_ctl.hshf_vol == 0\
                                  && sys_ctl.voice_play_flag == 0)   //通话音量为 0 时 MUTE 功放
    #define DAC_VOL_IS_ZERO()    (ATCON0 & BIT(0))   //动态降噪（DAC 音量为 0）时 MUTE 功放
#else
    #define HSHF_VOL_IS_ZERO()    0      //通话时解 MUTE 功放
    #define DAC_VOL_IS_ZERO()     0      //动态降噪（DAC 音量为 0）不影响 MUTE 功放
#endif

//判断是否需要 MUTE 功放，在中断调用
__near_func bool is_need_mute(void)
{
    //系统 MUTE、音量为 0 或者非播放状态时，可以 MUTE 功放
    return (sys_ctl.mute_flag || sys_ctl.volume == 0 || sys_ctl.need_mute || DAC_VOL_IS_ZERO())? true :
false;
}

__near_func void mute_ctl_auto(void)
{
    if((BT_STATE != StateNone) && (HSFCallActive | HSFRingActive | scoflag)) {
        if(HSHF_VOL_IS_ZERO()) {
            MUSIC_MUTE();
        } else {
            MUSIC_UNMUTE();        //通话过程不 MUTE
        }
    } else {
        if((sys_ctl.voice_play_flag == 0) && is_need_mute()) {
```

```
                MUSIC_MUTE();
            } else {
                MUSIC_UNMUTE();
            }
        }
}
```

4) 调用 dac_fade_in_2 时，由原来的 is_need_mute 改为 is_sys_mute。

```
__near_func void sound_detect_process(void)
{
    if(sound_detect_ctl.delay_cnt == 0) {
        sound_detect_ctl.delay_cnt = 4;        //5ms 检测 1 次
        u16 v_pow = voice_maxpow(sound_detect_ctl.ptr, sound_detect_ctl.samples);
        //printf("%04x ",v_pow);
        sound_detect(v_pow);
        if(sound_detect_ctl.fade_en == 0x81){
            if (sound_flag()) {
                //uart_putchar('1');
                if(!is_sys_mute()) {            //只需要判断是否系统 mute 即可
                    dac_fade_in_2(1);
                }
            } else {
                //uart_putchar('0');
                dac_fade_out_2();
            }
        }
    } else if(sound_detect_ctl.delay_cnt < 5) {
        sound_detect_ctl.delay_cnt--;
    }
}
```

IPHONE 电量偶尔显示不准问题

更新时间：2014-7-18

- **问题现象**

iphone 连接蓝牙时一般会显示设备电量，第一次连接时电量显示正常，断开重连后，电量显示不准确。

- **解决方法**

1) 将 dev_bat 修改为全局变量，

```
u8 dev_bat = 0xff;          //蓝牙显示电量;
```

2) 修改以下函数

```
void task_bt_state(void)
```

```
{
    if(BT_STATE_pre != BT_STATE)
    {
        BT_STATE_pre = BT_STATE;
        dev_bat = 0xff;   //状态变化时更新电量显示
        //WARNING_MSG("State: %02x\n", BT_STATE);
        ......
    }
    ......
}
#pragma location="TASK_BT_SEG"
void task_bt_1s_msg(void)
{
    if(IS_APPLE_BT_DEVICE) {                    //苹果设备显示电量
        u8 send_bat_to_iphone(u8 batval);
        static u8 dev_bat = 0;
        if(sys_ctl.bat_val != dev_bat) {
            if(!send_bat_to_iphone(sys_ctl.bat_val ? (2*sys_ctl.bat_val-1) : 0)) {
                dev_bat = sys_ctl.bat_val;
            }
        }
}
......
}
```

IPHONE 电量显示一直为 0 问题（V103）

更新时间：2014-7-18

- **问题现象**

  使用 iphone 连接设备时，电量一直显示为 0；设备回连 iphone 时则电量显示正常。V102 及之前版本不存在此问题。

- **解决方法**

  将“btlib_ 20140718.rar”中的蓝牙库 btlib.r51 替换到 V103SDK 中。

如何控制苹果设备媒体音量（V103、V104）

更新时间：2014-8-8

- **解决方法**

  V104 添加了 AVRCP 1.4 支持，可以实现控制苹果设备媒体音量功能，通过苹果设备也可以控制音箱音量。

SDK 中默认是关闭此功能，可以在 btapi.h 中打开：

```
#define BT_FT_APPLE_VOL_CTRL_EN     0x40   //苹果音量控制：0x40 支持，0x00 不支持
```

注意：1）只能在苹果设备播音乐时才能控制音量；

2）频繁调音量时可能引起音乐断续，建议放慢连发按键消息；

3）iphone 音量范围是 0~0x7f，分成 16 级，要转换为系统音量，VOLUME_MAX 最好能被整除，如 16 或 32；

4）改为 AVRCP1.4 后建议重做 BQB 认证；

若要在 V103 上支持，可以将 V104 的 btlib.r51 替换到 V103 中，然后添加如下代码：

1) 在 btapi.c 添加宏定义和声明：

```
#define BT_FT_APPLE_VOL_CTRL_EN     0x40   //苹果音量控制：0x40 支持，0x00 不支持
#define BT_FT_APPLE_DEVICE          0x01
#define BT_FT_APPLE_VOL_CTRL        0x02
#define IS_APPLE_VOL_CTRL           (BT_FT_APPLE_VOL_CTRL_EN\
                                    && (bt_user_feature & BT_FT_APPLE_VOL_CTRL))
#define IS_APPLE_BT_DEVICE          (BT_FT_APPLE_DEVICE_EN\
                                    && (bt_user_feature & BT_FT_APPLE_DEVICE))
void AVRCPChangeVolume(void);
```

2) 在 btapi.c 添加以下函数：

```
#if BT_FT_APPLE_VOL_CTRL_EN
#pragma location="BT_AVCTP_RX"
void a2dpvol_set_hook(u8 vol)
{
    /*iphone 端调设备音量，其中 iphone 音量范围是 0~0x7f，要转换为设备音量，VOLUME_MAX 最好能被整除，如 16 或 32 */
    vol = (vol+2)/8;
    if(vol != sys_ctl.volume) {
        user_set_volume(vol);
        show_volume();
    }

    //vol = (vol+(0x80/VOLUME_MAX-1))/(0x80/VOLUME_MAX);//被整除时用此公式计算
    //if(vol != sys_ctl.volume) {
    //      user_set_volume(vol);
    //      show_volume();
    //}
}

#pragma location="BT_AVCTP_RX"
u8 a2dpvol_get_hook(void)
```

```
{
    /*iphone 获取设备音量，要转换为 iphone 音量，VOLUME_MAX 最好能被整除*/
    u8 vol = sys_ctl.volume*8+2;
    if(vol < 7) {
        return 0;
    } else if(vol > 0x77) {
        return 0x7f;
    } else {
        return vol;
    }

    //vol = sys_ctl.volume*(0x80/VOLUME_MAX);   //被整除时用此公式计算
    //if(vol > 0x7f) {
    //      vol = 0x7f;
    //}
    //return vol;
}
#endif
```

3) 在 task_bt_deal_msg()添加消息处理：

```
        //音量加减
    case K_VOL_DOWN:
    case KL_VOL_DOWN:
    case KH_VOL_DOWN:
case KH_PREV_S10:
    ......
        user_set_volume(user_volume_dec(sys_ctl.volume));
        show_volume();

        if(IS_APPLE_VOL_CTRL) {
            AVRCPChangeVolume();
        }
        break;

    case K_VOL_UP:
    case KL_VOL_UP:
    case KH_VOL_UP:
    case KH_NEXT_S10:
        ......
        user_set_volume(user_volume_inc(sys_ctl.volume));
        show_volume();

        if(IS_APPLE_VOL_CTRL) {
```

```
                AVRCPChangeVolume();
            }
            break;
```

CW6680E/CW6639 改善 RF 的一致性

更新时间：2014-8-15

- **问题现象**

  改善 RF 的一致性

- **解决方法**

  按下图修改 RF 参数配置表

```
#pragma constseg="BT_API_CONST"
IAR_CONST u8 BT_RFINFO_TABLE[128] = {//spi_bt_rfinfo_read
    // 信息头
    'C',  'O', 'N', 'W', 'I', 'S', 'E', '-', '6', '6', '3', '9', 'v', '1', '.', '0',

    // 蓝牙名字
    BT_NAME,

    // 蓝牙地址
    0x56,0x34,0x12,0x99,0x22,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

    // 蓝牙RF信息,eeprom address: [0x80~0xBF]
    0x0F,0x40,0x8F,0x41,0x04,0x06,0x04,0x04,0x00,0x80,0x00,0x00,0x80,0x01,0x20,0xD6,
    0x93,0x00,0xF7,0x3F,0x00,0x00,0x00,0x00,0x1F,0x00,0x00,0x00,0x00,0x00,0x1E,0x90,
    0xFF,0x1E,0x2C,0x1F,0x06,0x42,0x14,0x7E,0x00,0x04,0x07,0x0E,0x1E,0x28,0x00,0x01,
    0x00,0x00,0xF4,0xF1,0x0C,0xC3,0x00,0x16,0x44,0x00,0x29,0x1C,0x1C,0x24,0xFF,0xFE
};
```

FM 播放音量调节提示音后声音变小问题

更新时间：2014-8-19

- **问题现象**

  FM 不经过主控时，第一次播放音量调节提示音无声，播放完音量调节提示音后 FM 的声音变小。

- **解决方法**

  FM 不经过主控时，播放 DAC 前先调用 fm_off 和 dac_recover；播完后调用 dac_pull_down，再重新初始化 fm。

  参考下面代码修改相关函数：

```
#pragma location="TASK_FM_SEG_1"
void task_fm_menu_music(u8 menu_music_num)
{
#if !FM_VOUT_THROUGH_MCU
  //fm_set_vol(0);
```

```
    fm_off();     //播放提示音时需要关闭 FM，否则 FM 会把 DAC 声音拉低
    dac_recover();
#endif
    mp3en_exit();
    dac_enable();

    mp3_res_play_wait(menu_music_num);

    dac_disable();
    task_fm_mp3en_init();  //恢复正常 FM 播放
#if FM_FREQ_SHOW
    mp3en_start(false);
    music_freq_init(FREQ_ADC);
#if LCD_THEME
    memset(disp_freq_show_buf, 0, sizeof(disp_freq_show_buf));
#endif
#endif
    load_code(DCODE_FM, 3);
    task_fm_load();    //由于 fm_ch[]是保存在 DCODE_RAM 的，播放提示音后会被清零，
需要重新 LOAD 一下

#if !FM_VOUT_THROUGH_MCU
dac_pull_down();
    fm_init();
    t_fm.ch_sel = param_read8(PARAM_FM_CHSEL);  //读取当前台号
    t_fm.freq = param_read16(PARAM_FM_FREQ);   //读取当前频率
    task_fm_fix();        //校正频道
    fm_set_freq(t_fm.freq); //设置频率
    //fm_unmute();  //设置频率函数里面有 unmute
    fm_set_vol(sys_ctl.volume);
#endif
}

#pragma location="TASK_FM_SEG_1"
void task_fm_menu_music_continue(u8 menu_music_num)
{
#if !FM_VOUT_THROUGH_MCU
    //fm_set_vol(0);
    fm_off();     //播放提示音时需要关闭 FM，否则 FM 会把 DAC 声音拉低
    dac_recover();
#endif
    mp3en_exit();
```

```
    dac_enable();
#if VOL_MAXMIN_MUSIC && !VOL_MUSIC_ONCE
    while(sys_ctl.vol_maxmin_music_flag != 1)
#endif
    {
        mp3_res_play_wait(menu_music_num);
    }

    dac_disable();
    //task_fm_mp3en_init(); //要连续播放提示音，暂时不恢复正常 FM 播放
#if FM_FREQ_SHOW
    mp3en_start(false);
    music_freq_init(FREQ_ADC);
#if LCD_THEME
    memset(disp_freq_show_buf, 0, sizeof(disp_freq_show_buf));
#endif
#endif
    load_code(DCODE_FM, 3);
    task_fm_load();    //由于 fm_ch[]是保存在 DCODE_RAM 的，播放提示音后会被清零，
需要重新 LOAD 一下

#if !FM_VOUT_THROUGH_MCU
dac_pull_down();
    fm_init();
    t_fm.ch_sel = param_read8(PARAM_FM_CHSEL);  //读取当前台号
    t_fm.freq = param_read16(PARAM_FM_FREQ);    //读取当前频率
    task_fm_fix();          //校正频道
    fm_set_freq(t_fm.freq); //设置频率

    task_fm_mp3en_init(); //恢复正常 FM 播放
    //fm_unmute();  //设置频率函数里面有 unmute
    fm_set_vol(sys_ctl.volume);
#endif
}

#pragma location="TASK_FM_SEG_1"
void fm_play_vol_music(void)
{
#if !VOL_MUSIC_ONCE
    if(!sys_ctl.vol_maxmin_music_flag){
        sys_ctl.vol_maxmin_music_flag = 2;
    }
```

```
    task_fm_menu_music_continue(RES_MP3_VOL_MAXMIN);
    flush_msg();
#else
    if(!sys_ctl.vol_maxmin_music_flag){
        sys_ctl.vol_maxmin_music_flag = 2;
        task_fm_menu_music(RES_MP3_VOL_MAXMIN);
    }
#endif
}

//任务消息处理
#pragma location="TASK_FMPLAY_SEG"
__near_func void task_fm_deal_msg(u8 msg)
{
#if VOL_MAXMIN_MUSIC
    if(sys_ctl.vol_maxmin_music_flag == 1){
        #if !VOL_MUSIC_ONCE && FM_VOUT_THROUGH_MCU
            task_fm_mp3en_init();
        #endif

        sys_ctl.vol_maxmin_music_flag = 0;
        flush_msg();
        return;
    }
#endif
```

蓝牙音量调节提示音偶尔无声问题（V100~V103）

更新时间：2014-8-19

- **问题现象**
  蓝牙模式下在音量最大时短按音量+键偶尔会没有音量调节提示音。
- **解决方法**
  在 BtApi.c 的 bank_process_bt_voice()函数中按以下两点修改：

  1、删掉以下两处红色部分代码：

```
    #if VOL_MAXMIN_MUSIC
            if(bt_voice_msg == RES_WAV_VOL_MAXMIN){
                bt_voice_delay_cnt = 100;
        } else
    #endif
```

```
{
        bt_voice_delay_cnt = OPA_UNMUTE_DELAY; //功放解 MUTE 延时，可根据实际情况调整
}
.....
      case BT_VMS_DELAY:
            if(bt_voice_delay_cnt){//printf("BT_VMS_DELAY\n");
                  return;
            } else {//printf("BT_VMS_INIT\n");
#if VOL_MAXMIN_MUSIC
                  if(bt_voice_msg == RES_WAV_VOL_MAXMIN){
                        if(!sys_ctl.vol_maxmin_music_flag || sys_ctl.vol_maxmin_music_flag == 1){
                              bt_voice_outptr = bt_voice_inptr;
                              bt_voice_state = BT_VMS_PLAYEND;
                              break;
                        }
                  }
#endif
                  bt_voice_state = BT_VMS_INIT;
            }
            break;
```

2、增加以下一处红色部分代码：

```
                  if(msg){
                        bt_voice_msg = msg;
                        bt_voice_state = BT_VMS_INIT;
#if VOL_MAXMIN_MUSIC
                        if(msg == RES_WAV_VOL_MAXMIN){
                              if(sys_ctl.vol_maxmin_music_flag == 2){
                                    bt_voice_delay_cnt = 120;
                                    bt_voice_state = BT_VMS_DELAY;
                              } else {
                                    bt_voice_outptr = bt_voice_inptr;
                                    bt_voice_state = BT_VMS_PLAYEND;
                              }
                        }
#endif
```

音乐模式下播放完音量调节提示音后状态恢复

更新时间：2014-8-19

- **问题现象**

  音乐模式下播放完音量调节提示音后音乐会恢复到播放状态。

- **解决方法**

  按下面几个截图增加或注释掉相关代码：

```
//音乐任务的一些信息初始化
#pragma location="TASK_MUSIC_SEG_1"
void task_music_play_init(void)
{
#if VOL_MAXMIN_MUSIC
    if(!sys_ctl.vol_maxmin_music_flag)
#endif
    {
        u_msc.pause = 0;                        //清除暂停状态
    }
    t_msc.speed = 0;                            //设置播放速度

#if AB_REPEAT_EN
    t_msc.auto_repeat_start = 0;
    //umsc_sound_detect(t_msc.auto_repeat);
    //sound_set_flag(0, 0);
#endif

    music_set_speed(0);

    t_msc.play_sta = PLAYER_NORMAL;             //正常播放状态
}
```

```
//用于在播放歌曲时插入语音菜单的播放，播放完语音菜单后继续播放原来的歌曲
void task_music_menu_music(u8 menu_music_num)
{
#if CONSTANT_WARNING_VOLUME_EN
    user_change_volume(WARNING_VOLUME);
#endif
    sys_ctl.voice_play_flag = 1;
    music_unmute();

    type_music_point pt;
    music_get_point(&pt);                       //设置返回播放点
    mp3_res_play(menu_music_num);
    while (music_get_status() > STATUS_PLAY_STOPPING);

    sys_ctl.voice_play_flag = 0;
    user_set_volume(sys_ctl.volume);

    music_init();
    task_music_play_init();
    music_jump(&pt);                            //恢复播放
    if(!u_msc.pause){
        music_play();
    } else {
        music_pause();
    }
}
```

```
//音乐任务消息处理
#pragma location="TASK_MUSIC_SEG"
void task_music_deal_msg(u8 msg)
{
#if VOL_MAXMIN_MUSIC
    if(sys_ctl.vol_maxmin_music_flag == 1){
        #if !VOL_MUSIC_ONCE
            music_init();
            task_music_play_init();
            music_jump(&pt);     //恢复播放
            if(!u_msc.pause){
                music_play();
            } else {
                music_pause();
            }
        #endif

        sys_ctl.vol_maxmin_music_flag = 0;
        flush_msg();
        return;
    }
#endif
```

```
#if VOL_MAXMIN_MUSIC
    if(u_msc.pause){
        //music_play();
        //u_msc.pause = 0;
    }
    if(!sys_ctl.volume){
        music_play_vol_music();
    }
#endif
```

```
#if VOL_MAXMIN_MUSIC
    if(u_msc.pause){
        //music_play();
        //u_msc.pause = 0;
    }
    if(sys_ctl.volume == VOLUME_MAX){
        music_play_vol_music();
    }
#endif
```

FM 模式下播放完音量调节提示音后状态恢复

更新时间：2014-8-19

- **问题现象**

  FM 模式下播放完音量调节提示音后 FM 会恢复到正常播放状态。

- **解决方法**

  1、按下面三个截图增加红色部分代码：

```
#pragma location="TASK_FM_SEG_1"
void task_fm_menu_music(u8 menu_music_num)
{
#if !FM_VOUT_THROUGH_MCU
    fm_set_vol(0);
#endif
    mp3en_exit();
    dac_enable();

    mp3_res_play_wait(menu_music_num);

    dac_disable();
    task_fm_mp3en_init();    //恢复正常FM播放
    if(sys_ctl.mute_flag){
        dac_fade_out(0);
    }
#if FM_FREQ_SHOW
    mp3en_start(false);
    music_freq_init(FREQ_ADC);
#if LCD_THEME
    memset(disp_freq_show_buf, 0, sizeof(disp_freq_show_buf));
#endif
#endif
```

```
#if VOL_MAXMIN_MUSIC
#pragma location="TASK_FM_SEG_1"
void fm_play_vol_music(void)
{
#if !VOL_MUSIC_ONCE
    if(!sys_ctl.vol_maxmin_music_flag){
        sys_ctl.vol_maxmin_music_flag = 2;
    }
    task_fm_menu_music_continue(RES_MP3_VOL_MAXMIN);
    flush_msg();
#else
    if(!sys_ctl.vol_maxmin_music_flag){
        sys_ctl.vol_maxmin_music_flag = 2;
        task_fm_menu_music_continue(RES_MP3_VOL_MAXMIN);
    }
#endif
}
#endif
```

```
//任务消息处理
#pragma location="TASK_FMPLAY_SEG"
__near_func void task_fm_deal_msg(u8 msg)
{
#if VOL_MAXMIN_MUSIC
    if(sys_ctl.vol_maxmin_music_flag == 1){
        #if !VOL_MUSIC_ONCE
            task_fm_mp3en_init();
            if(sys_ctl.mute_flag){
                dac_fade_out(0);
            }
        #endif

        sys_ctl.vol_maxmin_music_flag = 0;
        flush_msg();
        return;
    }
#endif
```

2、FM 的音量调节消息处理修改如下：

```
        //调节音量
    case K_VOL_DOWN:
    case KL_VOL_DOWN:
    case KH_VOL_DOWN:
    case KH_PREV_S10:
#if FM_SET_VOL_CHANGE_BANK
        user_set_volume(user_volume_dec(sys_ctl.volume));
#else
          user_set_volume_fm(user_volume_dec_fm(sys_ctl.volume));
#endif

#if !FM_VOUT_THROUGH_MCU
        fm_set_vol(sys_ctl.volume);
#endif

#if VOL_MAXMIN_MUSIC
        if(!sys_ctl.volume){
            fm_play_vol_music();
        }
#endif

#if KU_PLAY_FM_MUTE_EN || HARD_IR_EN || SOFT_IR_EN
    #if VOL_MAXMIN_MUSIC
        if(!sys_ctl.vol_maxmin_music_flag)
    #endif
        {
```

```
                if(sys_ctl.mute_flag) {
                    if(sys_ctl.volume) {
                        music_unmute();
                    #if FM_VOUT_THROUGH_MCU
                        dac_fade_in(0);
                    #endif
                        sys_ctl.sleep_cnt = 0;
                    }
                }
            }
#endif
            show_volume();
            break;
        case K_VOL_UP:
        case KL_VOL_UP:
        case KH_VOL_UP:
        case KH_NEXT_S10:
#if FM_SET_VOL_CHANGE_BANK
            user_set_volume(user_volume_inc(sys_ctl.volume));
#else
              user_set_volume_fm(user_volume_inc_fm(sys_ctl.volume));
#endif

#if !FM_VOUT_THROUGH_MCU
            fm_set_vol(sys_ctl.volume);
#endif

#if VOL_MAXMIN_MUSIC
            if(sys_ctl.volume == VOLUME_MAX){
                fm_play_vol_music();
            }
#endif

#if KU_PLAY_FM_MUTE_EN || HARD_IR_EN || SOFT_IR_EN
    #if VOL_MAXMIN_MUSIC
            if(!sys_ctl.vol_maxmin_music_flag)
    #endif
            {
                if(sys_ctl.mute_flag) {
                    if(sys_ctl.volume) {
                        music_unmute();
                    #if FM_VOUT_THROUGH_MCU
```

```
                    dac_fade_in(0);
                #endif
                    sys_ctl.sleep_cnt = 0;
                }
            }
        }
#endif
        show_volume();
        break;
```

用 RC 时钟源时蓝牙播歌断线问题（V105~V109）

更新时间：2014-10-10

- **问题现象**
  部分芯片选用 RC 作为时钟源时，蓝牙播歌存在断线的情况。
- **解决方法**
  使用 V110 SDK 中的 llp_lib.r51，替换到 V105~V109 中，重新编译。