驱动软件对外接口

1 DIO 卡

1.1 参数类型枚举

```
typedef enum _EnableStatus
{
    UNABLE_STATUS = 0, 不使能
    ENABLE_STATUS = 1  使能
} EnableStatus;
```

说明：使能枚举

```
typedef enum _OutputMode{
    NULL_OUTPUT_MODE = 0x00, 无模式
    LEVEL_OUTPUT_MODE = 0x01, 电平模式
    PWM_OUTPUT_MODE = 0x02,  PWM 模式
    BIT_OUTPUT_MODE = 0x03   BIT 数据流模式
} OutputMode;
```

说明：DIO 输出模式枚举

```
typedef enum _OutputImpedanceMode
{
    HIGH_IMPEDANCE_MODE = 0, 高阻模式
    PULL_MODE = 0X01,  拉模式
    PUSH_MODE = 0X02,  推模式
    PUSH_AND_PULL_MODE = 0X03 推拉模式
} OutputImpedanceMode;
```

说明：输出阻抗模式

```
typedef enum _VoltageLevel
{
    LOW_LEVEL = 0x00,  低电平
    HIGH_LEVEL = 0x01, 高电平
    INVALID_LEVEL
} VoltageLevel;
```

说明：电平枚举

```
typedef enum _OutputReferenceVoltage
{
```

```
    OUTPUT_5V_REF = 0X01, 输出 5v
    OUTPUT_12V_REF = 0X02, 输出 12v
    OUTPUT_OUTSIDE_REF = 0X03 输出外部参考
} OutputReferenceVoltage;
```
说明：输出参考电压

```
typedef enum _ReferenceClock
{
    REFCLK_100_MHZ = 0, 100M 参考时钟
    REFCLK_20_MHZ = 1  20M 参考时钟
} ReferenceClock;
```
说明：参考时钟

1.2 参数数据结构体

```
typedef struct _DOLevelConfigure
{
    OutputImpedanceMode eMode; 输出阻抗
    OutputReferenceVoltage eRef;       输出参考电压
} DOLevelConfigure;
```
说明：输出电平配置

```
typedef struct _DOPWMConfigure
{
    double dFreq;              频率 Hz
    double dDuty;              占空比 0~1
    OutputImpedanceMode eMode; 输出阻抗
    OutputReferenceVoltage eRef; 输出参考电压
} DOPWMConfigure;
```
说明：输出 PWM 配置

```
typedef struct _DIPWMConfigure
{
    double dDurationTime;      采样时间 ms
    ReferenceClock eRefClk;    参考时钟（保留，目前为自适应频率）
} DIPWMConfigure;
```
说明：输入 PWM 配置

```
typedef struct _PWMProperty
{
    double dFreq;              频率 Hz
    double dDuty;              占空比 0~1
} PWMProperty;
```
说明：PWM 属性

```
typedef struct _DOBitConfigure
```

```
{
    OutputImpedanceMode eMode; 输出阻抗
    long lRate;         速率 hz
    OutputReferenceVoltage eRef; 输出参考电压
} DOBitConfigure;
```
说明：输出 BIT 流配置

1.3 函数接口

```
int OpenDio (const char* strDev, void **handle);
```
说明： 打开 DIO 对应设备文件，获取对应设备文件指针
参数： strDev DIO 对应设备文件，如“/dev/dio_3_in”，表示打卡 3 槽 DIO 输入部分
        handle 对应设备文件指针
返回值：正常为 0
备注：

```
int GetDioVersion (void *handle,
                   char* pStr,
                   int iLength,
                   int* pActLength);
```
说明： 获取当前 DIO 卡逻辑固件版本信息
参数： handle 资源设备对应指针
        pStr 日期版本字符串首地址，用户申请字符数组之数组地址
        iLength 用户申请的字节数组长度
        pActLength 获取的版本信息实际有效字节数组长度
返回值：正常为 0
备注：

```
int CloseDio (void* handle);
```
说明： 关闭 DIO 对应设备文件
参数： handle 资源设备对应指针
返回值：正常为 0
备注：

```
int SetDoChannelEnable (void *handle,
                        int iChannel,
                        EnableStatus eStatus);
```
说明： 设置 DIO 卡指定通道输出使能
参数： handle 资源设备对应指针
        iChannel 输出通道号,范围 0~23
        eStatus 使能状态，ENABLE_STATUS 使能输出
返回值：正常为 0
备注：

```
int SetDoAllChannelEnable (void *handle, unsigned int iStatus);
```
说明：　设置 DIO 卡全通道输出使能
参数：　handle 资源设备对应指针
　　iStatus 全通道使能状态，通道按位表示，0bit 位为 0 通道，1 为使能，0 为未使能，如参数 0x03 表示输出通道 0 与 1 使能输出，其余 30 个输出通道均未使能输出
返回值：正常为 0
备注：

```
int GetDoAllChannelEnable (void *handle, unsigned int* pStatus);
```
说明：　获取全通道使能状态
参数：　handle 资源设备对应指针
　　pStatus 全通道使能状态，通道按位表示，0bit 位为 0 通道，1 为使能，0 为未使能，如获取 0x03 表示输出通道 0 与 1 为使能输出状态，其余 30 个输出通道均未使能输出
返回值：正常为 0
备注：

```
int SetDoMode (void *handle, int iChannel, OutputMode eMode);
```
说明：　设置 DIO 卡指定输出通道模式
参数：　handle 资源设备对应指针
　　iChannel 输出通道号，范围 0~23
　　eMode 输出模式，主要有电平模式，PWM 模式以及 BIT 数据流模式
返回值：正常为 0
备注：

```
int GetDoAllMode (void *handle,
                  unsigned int* pHiMode,
                  unsigned int* pLoMode);
```
说明：　获取所有输出通道模式
参数：　handle 资源设备对应指针
　　pHiMode 通道 16~23 输出模式指针，如获取 0x01，表示输出通道 16 模式为电平模式
　　pLoMode 通道 0~15 输出模式指针　如获取 0x02,表示输出通道 0 模式为 PWM 模式
返回值：正常为 0
备注：

```
int SetDiReferenceVoltage (void *handle,
                           int iChannel,
                           int iVoltage);
```
说明：　设置输入通道比较参考电压
参数：　handle 资源设备对应指针

iChannel 通道号，范围 0~31，0~7 为一组比较电压参考值，8~15 为一组比较电压参考值，16~23 为一组比较电压参考值，24~31 为一组比较电压参考值，如输入通道参数 1 与 7，其实下发的是同一组比较电压值

iVoltage 参考电压，单位毫伏，数值在 0~25V 之间

返回值：正常为 0

备注：

```
int GetDiLevel (void *handle, int iChannel, int* iVoltage);
```

说明：获取指定输入通道电平值

参数：handle 资源设备对应指针

iChannel 通道号

iVoltage 输入电平，0 为低，1 为高

返回值：正常为 0

备注：

```
int GetDiAllLevel (void *handle, int* iLevel);
```

说明：获取所有输入通道输入电平

参数：handle 资源设备对应指针

iLevel 所有通道的输入电平，通道按位表示，0bit 位为 0 通道，1 为高电平，0 为低电平，如获取 0x01 表示 0 通道为高电平输入，其余 31 个通道为低电平输入

返回值：正常为 0

备注：

```
int SetDoLevelConfigure (void *handle,
                         int iChannel,
                         DOLevelConfigure stDOLevelCfg);
```

说明：设置指定输出通道电平配置

参数：handle 资源设备对应指针

iChannel 输出通道号，0~23

stDOLevelCfg 输出电平配置结构

返回值：正常为 0

备注：

```
int SetDoLevel (void *handle, int iChannel, VoltageLevel eLevel);
```

说明：设置指定输出通道电平值

参数：handle 资源设备对应指针

iChannel 通道号，范围 0~23

eLevel 输出电平

返回值：正常为 0

备注：

```
int SetDiPWMCaptureEnable (void *handle,
                           int iChannel,
```

```
                                    EnableStatus eStatus);
```
说明： 设置指定输入通道 PWM 捕获使能
参数： handle 资源设备对应指针
        iChannel 通道号，范围 0~31
        eStatus 使能标志
返回值：正常为 0
备注：

```
int SetDiPWMAllCaptureEnable (void *handle, unsigned int iStatus);
```
说明： 设置所有输入通道 PWM 捕获使能
参数： handle 资源设备对应指针
        iStatus 所有输入通道 PWM 采集使能，0bit 位为 0 通道，1 为使能，0 为不使能，如 0x03 表示输入通道 0、1PWM 捕获使能，其余 30 个通道捕获未使能
返回值：正常为 0
备注：

```
int GetDiPWMAllCaptureEnable (void *handle, unsigned int* pStatus);
```
说明： 获取所有输入通道 PWM 捕获使能状态
参数： handle 资源设备对应指针
        pStatus 所有输入通道 PWM 采集使能状态指针，0Bit 位 0 通道，1 位使能，0 为不使能，如 0x01 表示输入通道 0 为捕获使能状态，其余输入通道未捕获使能
返回值：正常为 0
备注：

```
int SetDoPWMConfigure (void *handle,
                       int iChannel,
                       DOPWMConfigure stDOPWMCfg);
```
说明： 设置指定输出通道 PWM 波形配置
参数： handle 资源设备对应指针
        iChannel 输出通道号，范围 0~23
        stDOPWMCfg 输出 pwm 配置数据结构
返回值：正常为 0
备注：

```
int SetDiPWMConfigure (void *handle,
                       int iChannel,
                       DIPWMConfigure stDIPWMCfg);
```
说明： 设置指定输入通道 PWM 波形采集设置
参数： handle 资源设备对应指针
        iChannel 通道号，范围 0~31
        stDIPWMCfg 输入 pwm 配置数据结构
返回值：正常为 0
备注：

```
int GetDiPWMCapture (void *handle, PWMProperty arrPWMProper[32]);
```
说明： 获取所有输入通道 PWM 波形采集数据
参数： handle 资源设备对应指针
        arrPWMProper 波形数据结构体
返回值：正常为 0
备注：

```
int ClearDoOverProtectionStatus (void *handle);
```
说明： 清除板卡过流保护状态标志，过流指示灯恢复为绿色正常
参数： handle 资源设备对应指针
返回值：正常为 0
备注：

```
int SetDoBitConfigure (void *handle,
                       int iChannel,
                       DOBitConfigure stDOBitCfg);
```
说明： 设置指定输出通道 BIT 数据流配置
参数： handle 资源设备对应指针
        iChannel 通道号，范围 0~23
        stDOBitCfg 数据流配置结构
返回值：正常为 0
备注：

```
int WriteDoBITData (void *handle, uint32_t* pArr, int iLength);
```
说明： 指定 DIO 卡发送 BIT 数据流
参数： handle 资源设备对应指针
        pArr 数据流缓冲区首地址
        iLength 数据流缓冲区长度，长度须为 2 的整数倍，
返回值：正常为 0
备注：数据流为 uint32_t 类型数组，每 uint32_t 数据，表示在指定通道配置为 bit 流模式下，24 路输出通道同时输出 1bit 数据。数据从低 uint32_t 元素至高 uint32_t 依次输出，如已配置好 0、1 通道输出模式为 bit 流模式，下发数据 0x03，0x01，0x02，0x01，表示通道 0 输出 BIT 流信息 1101，通道 1 输出 BIT 流信息 1010，如下图所示：

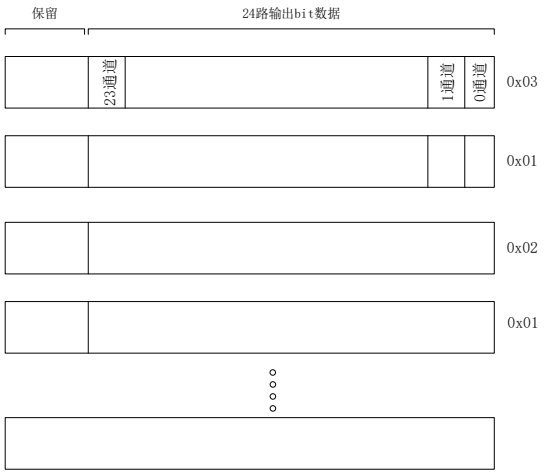


图 1Bit 流

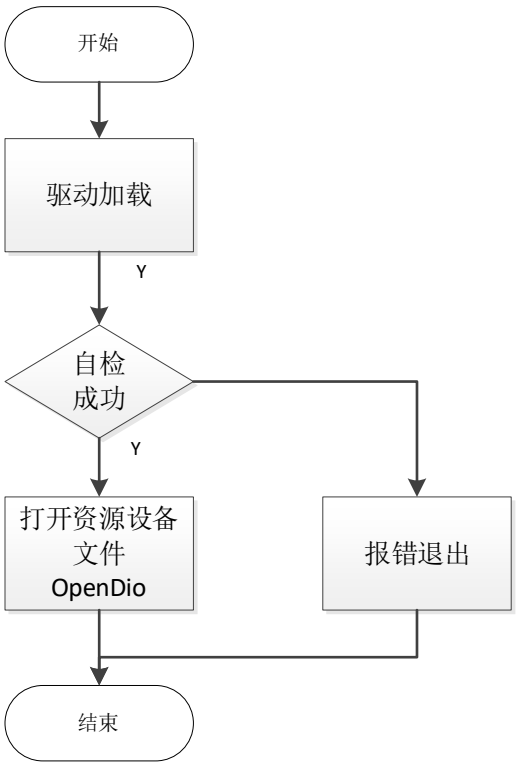
```
int SetDioPpsUpdateCount(void* handle, uint32_t sec);
```
说明： 设置 PPS 秒记数配置
参数： handle 资源设备对应指针
sec pps 秒计数值
返回值：正常为 0
备注：

```
int GetDioBitStreamSendTime (void* handle, uint64_t *mic_sec);
```
说明： 获取 BIT 流发送时间戳
参数： handle 资源设备对应指针
mic_sec 时间戳微秒计数值
返回值：正常为 0
备注：

```
int GetDioPpsSecCount (void* handle, uint32_t *sec);
```
说明： 获取当前 pps 秒计数值
参数： handle 资源设备对应指针
sec pps 秒计数值
返回值：正常为 0
备注：

1.4 工作流程操作

1.4.1 设备初始化与反初始化



1

图 2 设备初始化流程图

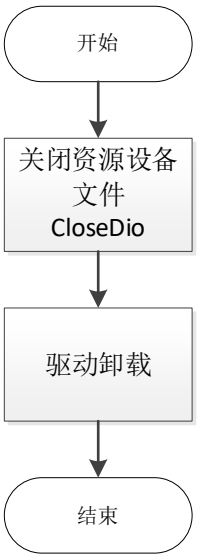


图 3 设备反初始化流程图

1.4.2 DIO 电平输出

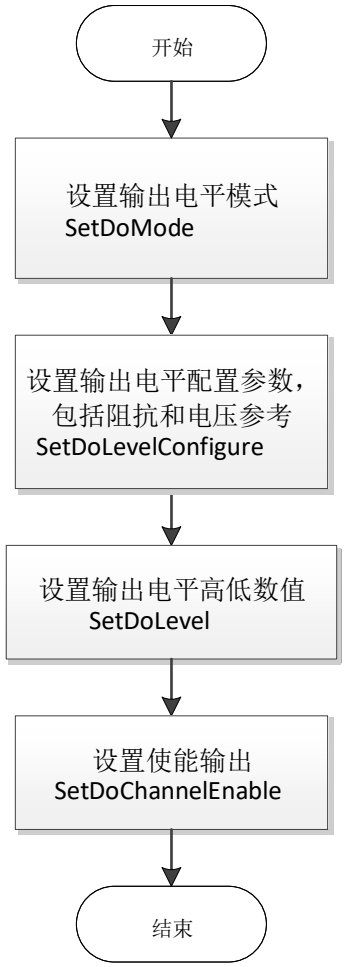


图 4 电平输出流程图

1.4.3 DIO 电平输入

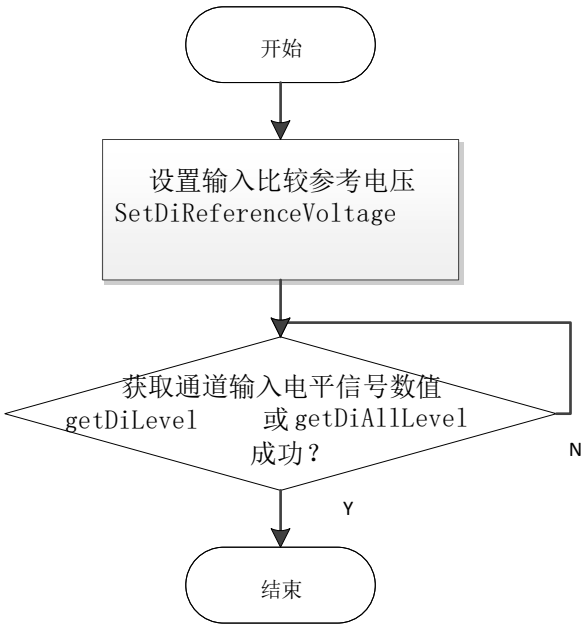


图 5 电平输入流程图

1.4.4 DIO PWM 波形输入

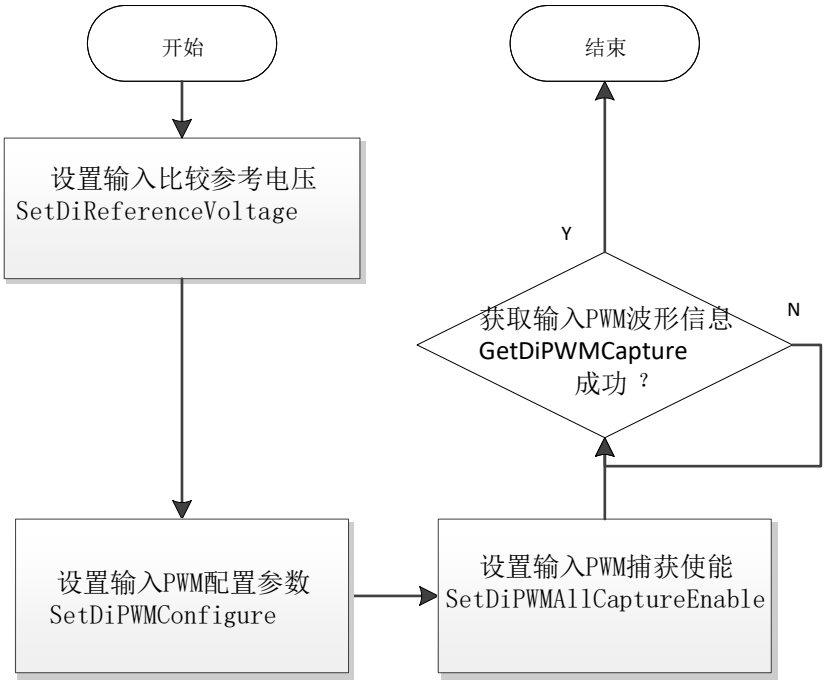


图 6PWM 输入流程图

1.4.5 DIO PWM 波形输出

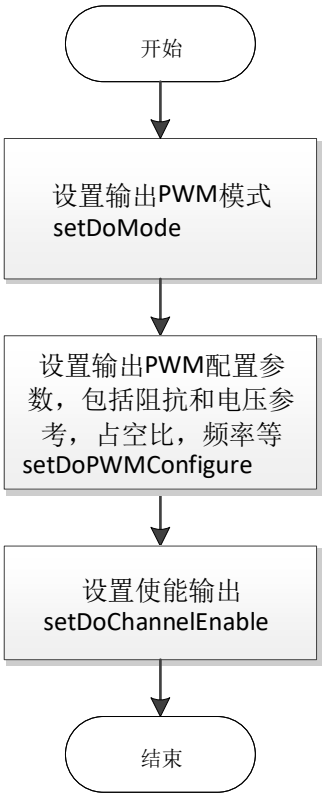


图 7PWM 输出流程图

1.4.6 DIO Bit 流信息输出

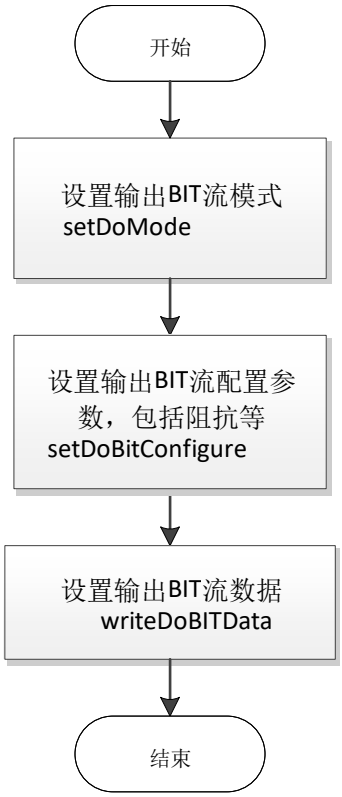


图 8BIT 流输出流程图

2 Audio 卡

2.1 参数类型枚举

```
typedef  enum _SAMPLING_RATE
{
    F48kHz,     采样频率 48000Hz
    F96kHz,     采样频率 96000Hz
    F192kHz     采样频率 192000Hz
    } SAMPLING_RATE;
```

说明：音频输入采样率

```
typedef enum _A2BRATE{
    R48k_400KHZ, 音频帧速 48k IIC 数据速率 400k
    R48k_100KHZ,        音频帧速 48k IIC 数据速率 100k
    R44_1k_400KHZ,      音频帧速 44.1k IIC 数据速率 400k
    R44_1k_100KHZ       音频帧速 44.1k IIC 数据速率 100k
    }A2BRATE;
```

说明：A2B 速率相关配置

```
typedef enum _A2BTXMODE{
    SINGLE,
    MASTERT,
    SLAVE0,
```

```
    SLAVE1
    }A2BTXMODE;
```
说明：A2B 传输模式

```
typedef enum _ERRORTYPE{
    OPEN_ERR = -1,
    WRITE_ERR = -2,
    READ_ERR = -3,
    DDR_EMPTY_ERR = -4,
    INPUT_MODE_ERR = -5,
    OUTPUT_MODE_ERR = -6,
    AWAKEN_ERR = -7,
    A2B_CONFIG_ERR = -8,
} ERRORTYPE;
```
说明：错误码标识

2.2 参数类型结构体

```
typedef  struct __AudInChlInfo{
  int mode = 0;          模式单声道 0 立体声 1
  int atten = 0;         固定衰减 1/5  0  无 1
  int sampling;          采样率，例如 48000 表示 48ksps 即每秒 48000 采样
    } AudInChlInfo;
```
说明：获取通道信息

```
typedef struct _ _AudInCfg{
    float in_coef;      增益  默认 1.0
    int IS_IEPE;        是否使用麦克风 1 使用 0 不使用
} AudInCfg;
```
说明：音频输入参数配置

```
typedef struct  _AudOutCfg{
    uint32_t regData;  固定数值，下发 0X200F
    float out_coef;    输出增益  1.0
    SAMPLING_RATE rate;         输出采样率
    } AudOutCfg;
```
说明：音频输出参数配置

2.3 函数接口

```
int OpenAudio(const char* strDev, void** handle);;
```
说明：打开 Audio 对应设备文件，获取对应设备文件指针
参数：strDev Audio 对应设备文件，如“/dev/audio_3_in1”，表示打卡 3 槽 Audio 输入 1 通道部分
        handle 对应设备文件指针

返回值：正常为 0
备注：

```
int GetAudioVersion(void* handle, char* pStr, int iLength, int*
pActLength);;
```
说明：获取当前 Audio 卡逻辑固件版本信息
参数：handle 对应设备文件指针
pStr 日期版本字符串首地址
iLength 用户申请的字节数组长度
pActLength 实际有效字节数组长度
返回值：正常为 0
备注：

```
int CloseAudio(void* handle);
```
说明：关闭 Audio 对应设备文件
参数：handle 对应设备文件指针
返回值：正常为 0
备注：

```
int SetAudiAtten (void* handle, int atten);
```
说明：设置指定输入通道音频衰减
参数: handle 对应设备文件指针
atten 固定衰减 0 1/5 衰减；1 无衰减
返回值：正常为 0
备注：

```
int SetAudiAllRate (void* handle, SAMPLING_RATE samp);
```
说明：设置音频卡音频输入的采样率
参数：handle 对应设备文件指针
samp 采样率（选用枚举值）
返回值：正常返回 0
备注：所有输入通道共用同一采样率

```
int SetAudiMode (void *handle, int mode);
```
说明：设置音频卡指定输入通道音频采集模式
参数：handle 对应设备文件指针
mode 模式 0 单声道 1 立体声
返回值：正常返回 0
备注

```
int SetAudiEnable (void *handle, int en);
```
说明：设置音频卡指定输入通道音频采集使能
参数：handle 对应设备文件指针
　　en 使能 0 禁止输入 1 使能输入
返回值：正常返回 0
备注：接口保留，目前未使用，采集文件配置 SetAudioInputFile 下发时，已内部设置采集使能

```
int GetAudiStatus (void *handle, int*fileSize, int* total_len);
```
说明：查询音频采集文件读写数据状态和缓冲区数据大小状态
参数：handle 对应设备文件指针
　　filesize 当前采集文件大小，字节
　　total_len 缓冲区的数据长度，字节
返回值：正常返回 0
备注：

```
int GetAudiChannelInfo(void *handle, AudInChlInfo *info);
```
说明：查询音频卡指定输入通道配置信息
参数：handle 对应设备文件指针
　　info 通道信息，包括模式，衰减，采样率
返回值：正常返回 0
备注：

```
int RecvAudiData(void *handle, char* pBuf, int iLength, int* iActLen);
```
说明：获取音频采集数据至指定缓冲区
参数：handle 对应设备文件指针
　　pBuf 缓冲区首地址
　　iLength 缓冲区大小
　　iActLen 实际获取数据大小
返回值：正常返回 0
备注：

```
int RecvAudiFile(void *handle, const char* strFile, int sec);
```
说明：获取音频采集数据至指定文件
参数：handle 对应设备文件指针
　　strFile 全路径文件名
　　sec 采集时间 秒
返回值： 正常返回 0
备注：

```
int SetAudiCfg(void* handle, AudInCfg* cfg);
```
说明：设置音频卡音频采集的增益、时间等配置
参数：handle 对应设备文件指针
　　cfg 输入采集配置指针
返回值：正常返回 0
备注：

```
int SetAudoEnable (void *handle, int en);
```
说明：设置音频卡指定输出通道使能
参数：handle 对应设备文件指针
　　en 使能 0 禁止输出 1 使能输出
返回值：正常返回 0
备注：接口保留，目前未使用，播放文件配置 setAudioOutputFile 下发时，已内部设置播放使能

```
int SetAudoMode (void *handle, int mode);
```
说明：设置音频卡指定输出通道模式
参数：handle 对应设备文件指针
　　mode 模式 0 单声道 1 立体声
返回值：正常返回 0
备注

```
int SetAudoAMP (void *handle, int amp);
```
说明：设置音频卡指定输出通道固定增益
参数: handle 对应设备文件指针
　amp 衰减 0 无增益；1 5 倍增益
返回值：正常为 0
备注：

```
int SetAudoLoopPlay (void *handle, int isLoop);
```
说明：设置音频卡指定输出通道是否循环播放
参数：handle 对应设备文件指针
　　isLoop 循环 0 不循环 1 循环
返回值：正常返回 0
备注：当循环时，下发参数 0 不循环，则播放完当前音频后禁止播放

```
int SendAudoData(void* handle, char* pBuf, int iLength, int*pActLength);
```
说明：发送音频数据至指定输出通道播放

参数：handle 对应设备文件指针
pBuf 数据缓冲区首地址
iLength 缓冲区长度
pActLength 实际下发长度
返回值：正常返回 0
备注： 一次下发数据最大为 2M，超过此限，只发送 2M

```
int SendAudoFile(void* handle, const char* strFile);
```
说明：发送音频文件数据至指定输出通道播放
参数：handle 对应设备文件指针
strFile 待发送数据文件
返回值：正常返回 0
备注：

```
int SetAudoCfg(void* handle, AudOutCfg* cfg);
```
说明：设置音频卡指定输出通道采样率等配置
参数：handle 对应设备文件指针
cfg 输出播放配置指针
返回值：正常返回 0
备注：

```
int SetA2BBoardMode (void *handle, int mode);
```
说明：设置 A2B 工作模式
参数：handle 对应设备文件指针
mode 模式 0 从模式 1 主模式
返回值：正常返回 0
备注

```
int SetA2BTxMode (void *handle, A2BTXMODE mode, A2BRATE mRate);
```
说明：设置 A2B 输出模式
参数：handle 对应设备文件指针
Mode 发送模式，保留，默认填 SINGLE
mRate 速率
返回值：正常返回 0
备注

```
int RecvA2BFile (void *handle, char * fileName, int length, float coef);
```
说明：A2B 根据配置接收数据并保存为指定文件
参数：handle 对应设备文件指针
fileName 接收数据保存文件名

length 保存文件大小
coef 接收数据幅度增益

返回值：正常返回 0

备注

```
int SendA2BFile (void *handle, char * fileName, SAMPLING_RATE rate,
                 float coef);
```

说明：A2B 根据配置打开指定文件，并发送数据

参数：handle 对应设备文件指针

fileName 发送数据文件名称

rate 输出采样率，当此 A2B 为主设备时，方生效

coef 发送数据幅度增益

返回值：正常返回 0

备注

2.4 函数接口操作流程

2.4.1 音频采集

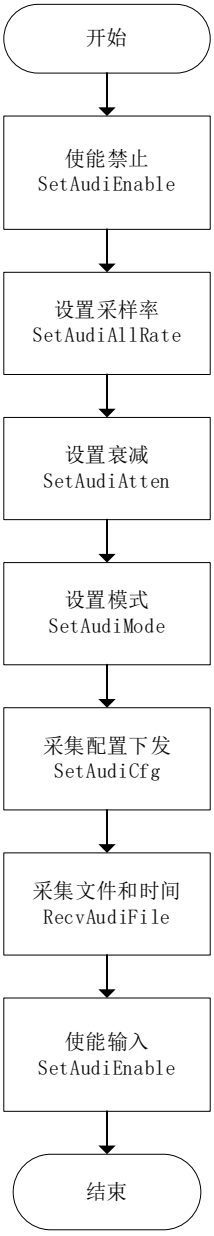


图 9 音频采集

2.4.2 音频输出

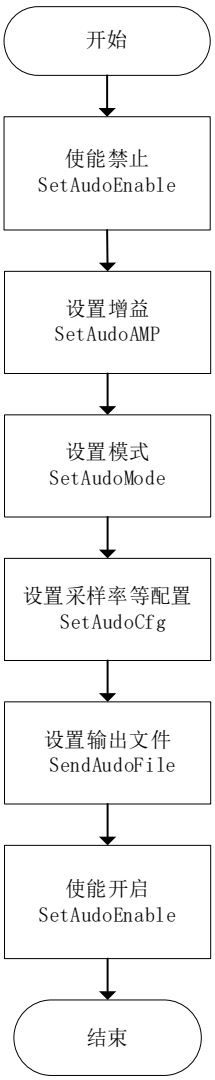


图 10 音频输出

2.4.3 A2B 配置及数据传输

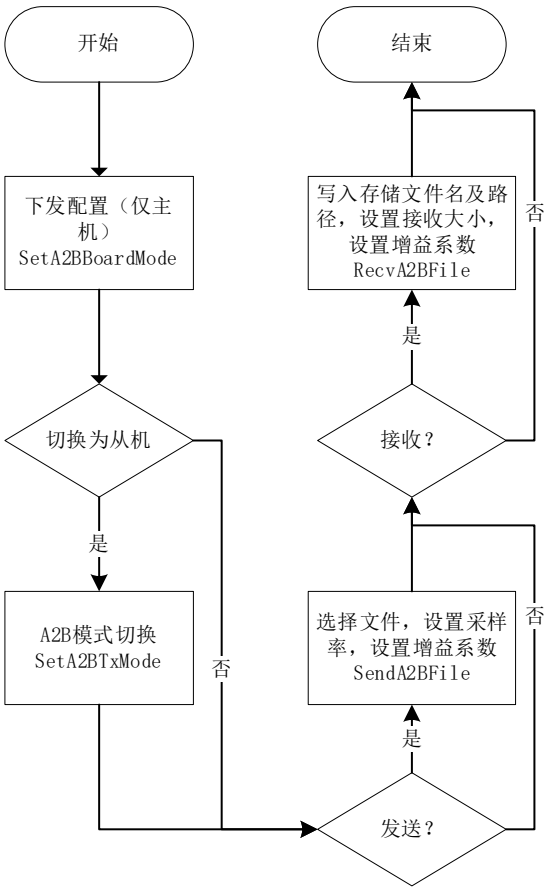


图 11A2B 模式配置以及数据传输