MI AI API

Version 2.07

REVISION HISTORY

Revision No.	Description	Date
2.03	• Initial release	04/12/2018
2.04	• Updated for accuracy	02/23/2019
2.05	• Added MI_AI_SetAedAttr, MI_AI_GetAedAttr, MI_AI_EnableAed, MI_AI_DisableAed, and MI_AI_GetAedResult	03/07/2019
2.06	• Added description regarding audio algorithm	03/25/2019
2.07	• Added MI_AI_SetExtAecChn and updated MI_AI_SetChnParam and MI_AI_GetChnParam	03/30/2019

TABLE OF CONTENTS

1. API 参考

1.1. 概述

音频输入（Audio Input, AI）主要实现配置及启用音频输入设备、获取音频帧数据等功能。

1.2. 功能模块 API

API 名	功能
MI_AI_SetPubAttr	设置 AI 设备属性
MI_AI_GetPubAttr	获取 AI 设备属性
MI_AI_Enable	启用 AI 设备
MI_AI_Disable	禁用 AI 设备
MI_AI_EnableChn	启用 AI 通道
MI_AI_DisableChn	禁用 AI 通道
MI_AI_GetFrame	获取音频帧
MI_AI_ReleaseFrame	释放音频帧
MI_AI_SetChnParam	设置 AI 通道参数
MI_AI_GetChnParam	获取 AI 通道参数
MI_AI_EnableReSmp	启用 AI 重采样
MI_AI_DisableReSmp	禁用 AI 重采样。
MI_AI_SetVqeAttr	设置 AI 的声音质量增强功能相关属性
MI_AI_GetVqeAttr	获取 AI 的声音质量增强功能相关属性
MI_AI_EnableVqe	使能 AI 的声音质量增强功能
MI_AI_DisableVqe	禁用 AI 的声音质量增强功能
MI_AI_ClrPubAttr	清除 AI 设备属性
MI_AI_SaveFile	开启音频输入保存文件功能
MI_AI_SetVqeVolume	设置声音质量增强功能中的音量大小
MI_AI_GetVqeVolume	获取声音质量增强功能中的音量大小
MI_AI_SetAencAttr	设置 AI 编码功能相关属性
MI_AI_GetAencAttr	获取 AI 编码功能相关属性
MI_AI_EnableAenc	使能 AI 编码功能
MI_AI_DisableAenc	禁止 AI 编码功能
MI_AI_SetAedAttr	设置声音事件检测相关属性
MI_AI_GetAedAttr	获取声音事件检测相关属性

API 名	功能
MI_AI_EnableAed	使能声音检测功能
MI_AI_DisableAed	禁止声音检测功能
MI_AI_GetAedResult	获取声音检测结果
MI_AI_SetExtAecChn	设置回声消除功能参考的外部 AI 通道

1.2.1 MI_AI_SetPubAttr

➢ 功能

设置 AI 设备属性。

➢ 语法

MI_S32 MI_AI_SetPubAttr(MI_AUDIO_DEV AiDevId, MI_AUDIO_Attr_t *pstAttr);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
pstAttr	AI 设备属性指针。	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

音频输入设备的属性决定了输入数据的格式，输入设备属性包括工作模式、采样率、采样精度、buffer 大小、每帧的采样点数和通道数目。这些属性应与对接 Codec 配置的时序一致，即能成功对接。

- 工作模式
  音频输入输出目前支持 I²S 主模式、I²S 从模式，但每个音频设备支持的内容可能有出入。
- 采样率
  采样率指一秒中内的采样点数，采样率越高表明失真度越小，处理的数据量也就随之增加。一般来说语音使用 8k 采样率，音频使用 32k 或以上的采样率；设置时请确认对接的 Audio Codec 是否支持所要设定的采样率。
- 采样精度
  采样精度指某个通道的采样点数据宽度，同时决定整个设备的通道分布。采样精度支持 16bit，实际应用中采样精度还受 Audio Codec 限制。
- buffer 大小
  MI_AUDIO_Attr_t 中的 u32FrmNum 项用于配置 AI 中用于接收音频数据的缓存的音频帧帧数，建议配置为 5 以上，否则可能出现采集丢帧等异常。
- 每帧的采样点数
  当音频采样率较高时，建议相应地增加每帧的采样点数目。如要将这些采集到的音频数据送编码，则应保证每帧的持续时长不少于 10ms（例如 16K 的采样频率下每帧的采样点数至少应设置为 160，如果声音有断断续续，可以适当增加每帧的采样点数，参数设置与具体芯片的性能有关），否则解码后声音可能有异常。
- 通道数目
  通道数目指当前输入设备的 AI 功能的信道数目，需与对接的 Audio Codec 的配置保持一致；支持 1 路、2 路、4 路、8 路、16 路。

- 时钟分时通道数目
  时钟分时通道数目决定了主模式下当前输入设备的 AI 支持通道数，与对接的 Audio Codec 及配置的信道数目相关。

➢ 举例

下面的代码实现设置 AI 设备属性及启用 AI 设备。

```
MI_S32 ret;
MI_AUDIO_Attr_t stAttr;
MI_AUDIO_Dev AiDevId =    0;
stAttr.eBitwidth = E_MI_AUDIO_BIT_WIDTH_16;
stAttr.eSamplerate = E_MI_AUDIO_SAMPLE_RATE_8000;
stAttr.eSoundmode = E_MI_AUDIO_SOUND_MODE_MONO;
stAttr.eWorkmode = E_MI_AUDIO_MODE_I2S_SLAVE;
stAttr.u32FrmNum = 5;
stAttr.u32PtNumPerFrm = 160;
stAttr.u32ChnCnt = 2;
/* set public attribute of AI device*/
ret = MI_AI_SetPubAttr(AiDevId, &stAttr);
if(MI_OK != ret)
{
        printf("set ai %d attr err:0x%x\n", AiDevId, ret);
        return ret;
}
/* enable AI device */
ret = MI_AI_Enable(AiDevId);
if(MI_OK != ret)
{
        printf("enable ai dev %d err:0x%x\n", AiDevId, ret);
        return ret;
}
```

1.2.2 MI_AI_GetPubAttr

➢ 功能

获取 AI 设备属性。

➢ 语法

MI_S32 MI_AI_GetPubAttr(MI_AUDIO_DEV AiDevId, MI_AUDIO_Attr_t*pstAttr);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
pstAttr	AI 设备属性指针。	输入

➢ 返回值

返回值
- 0　成功。
- 非 0　失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

- 获取的属性为前一次配置的属性。
- 如果从来没有配置过属性，则返回失败。

➢ 举例

```
MI_S32 ret;
MI_AUDIO_DEV AiDevId = 0;
MI_AUDIO_Attr_t stAttr;
ret = MI_AI_GetPubAttr(AiDevId, &stAttr);
if(MI_OK != ret)
{
    printf("get ai %d attr err:0x%x\n", AiDevId, ret);
    return ret;
}
```

1.2.3 MI_AI_Enable

➢ 功能

启用 AI 设备。

➢ 语法

```
MI_S32 MI_AI_Enable(MI_AUDIO_DEV AiDevId);
```

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

- 必须在启用前配置 AI 设备属性，否则返回属性未配置错误
- 如果 AI 设备已经处于启用状态，则直接返回成功。

➢ 举例

请参见 MI_AI_SetPubAttr 的举例。

1.2.4 MI_AI_Disable

➢ 功能

禁用 AI 设备。

➢ 语法

MI_S32 MI_AI_Disable(MI_AUDIO_DEV AiDevId);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

- 如果 AI 设备已经处于禁用状态，则直接返回成功
- 禁用 AI 设备前必须先禁用该设备下已启用的所有 AI 通道
- 要求在禁用 AI 设备之前，先禁用与之关联、使用 AI 的音频数据的通道和设备，否则可能导致该接口调用失败

➢ 举例

```
MI_S32 ret;
MI_AUDIO_DEV AiDevId = 0;
ret = MI_AI_Disable(AiDevId);
if(MI_OK != ret)
{
printf("disable ai %d err:0x%x\n", AiDevId);
return ret;
}
```

1.2.5 MI_AI_EnableChn

➢ 功能

启用 AI 通道

➢ 语法

MI_S32 MI_AI_EnableChn(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn);

➢ 形参

参数名称	描述	输入/输出

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 支持的通道范围由AI设备属性中的最大通道个数 u32ChnCnt 与声道模式eSoundmode 决定，详见 MI_AUDIO_SoundMode_e定义的描述。	输入

- 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

- 依赖
  - 头文件: mi_ai.h
  - 库文件: libmi.a

※ 注意
  - 启用 AI 通道前，必须先启用其所属的 AI 设备，否则返回设备未启动的错误码

- 举例

无

1.2.6 MI_AI_DisableChn

- 功能

禁用 AI 通道

- 语法

```
MI_S32  MI_AI_DisableChn(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn);
```

- 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, MI_AUDIO_MAX_CHN_NUM)。	输入

- 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

- 依赖
  - 头文件: mi_ai.h
  - 库文件: libmi.a

※ 注意

无

➢ 举例

无

1.2.7 MI_AI_GetFrame

➢ 功能

获取音频帧

➢ 语法

MI_S32 MI_AI_GetFrame(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AUDIO_Frame_t*pstFrm, MI_AUDIO_AecFrame_t *pstAecFrm , MI_S32 s32MilliSec);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 支持的通道范围由AI设备属性中的最大通道个数 u32ChnCnt 与声道模式eSoundmode 决定。	输入
pstFrm	音频帧结构体指针。	输出
pstAecFrm	回声抵消参考帧结构体指针。	输出
s32MilliSec	获取数据的超时时间 -1 表示阻塞模式，无数据时一直等待； 0 表示非阻塞模式，无数据时则报错返回； >0表示阻塞s32MilliSec毫秒，超时则报错返回。	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

- 如果需要获取回声抵消参考帧，pstAecFrm 不能是空指针，如果不想获取回声抵消参考帧 pstAecFrm 置为空指针即可
- AI 模块会缓存音频帧数据，用于用户态获取。缓存的深度通过 MI_AI_SetChnParam 接口设定，默认为 0
- s32MilliSec 的值必须大于等于-1，等于-1 时采用阻塞模式获取数据，等于 0 时采用非阻塞模式获取数据，大于 0 时，阻塞 s32MilliSec 毫秒后，没有数据则返回超时并报错
- 获取音频帧数据前，必须先使能对应的 AI 通道
- 本接口支持 select 操作

➢ 举例

无

1.2.8 MI_AI_ReleaseFrame

- 功能

  释放音频帧

- 语法

  MI_S32 MI_AI_ReleaseFrame(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AUDIO_Frame_t *pstFrm, MI_AUDIO_AecFrame_t *pstAecFrm);

- 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 支持的通道范围由AI设备属性中的最大通道个数 u32ChnCnt 与声道模式eSoundmode 决定。	输入
pstFrm	音频帧结构体指针。	输出
pstAecFrm	回声抵消参考帧结构体指针。	输出

- 返回值

  返回值
  - 0 成功。
  - 非 0 失败，参照错误码。

- 依赖
  - 头文件: mi_ai.h
  - 库文件: libmi.a

※ 注意
- 如果不需要释放回声抵消参考帧，pstAecFrm 置为 NULL 即可

- 举例

  无

1.2.9 MI_AI_SetChnParam

- 功能

  设置 AI 通道参数

- 语法

  MI_S32 MI_AI_SetChnParam(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AI_ChnParam_t *pstChnParam);

- 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 支持的通道范围由AI设备属性中的最大通道个数 u32ChnCnt 与声道模式eSoundmode 决定。	输入
pstChnParam	音频通道参数结构体指针。	输入

- 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

- 依赖
  - 头文件: mi_ai.h
  - 库文件: libmi.a

※ 注意

无

- 举例

无

1.2.10 MI_AI_GetChnParam

- 功能

获取 AI 通道参数

- 语法

MI_S32 MI_AI_GetChnParam(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AI_ChnParam_t *pstChnParam);

- 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 支持的通道范围由AI设备属性中的最大通道个数 u32ChnCnt 与声道模式eSoundmode 决定。	输入
pstChnParam	音频通道参数结构体指针。	输出

- 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

无

➢ 举例

无

1.2.11 MI_AI_EnableReSmp

➢ 功能

启用 AI 重采样

➢ 语法

MI_S32 MI_AI_EnableReSmp(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AUDIO_SampleRate_e eOutSampleRate);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 支持的通道范围由AI设备属性中的最大通道个数 u32ChnCnt 与声道模式eSoundmode 决定。	输入
eOutSampleRate	音频重采样的输出采样率。	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a libSRC_LINUX.so

※ 注意

- 在启用 AI 通道之后，调用此接口启用重采样功能。
- 允许重复启用重采样功能，但必须保证后配置的属性与之前配置的属性一样。
- 在禁用 AI 通道之后，如果重新启用 AI 通道，并使用重采样功能，需调用此接口重新启用重采样。

➢ 举例

以 AI 从 32K 到 8K 的重采样为例，配置如下：

```
/* dev attr of ai */
MI_AUDIO_SampleRate_e eOutSampleRate;
        stAioAttr.u32ChnCnt = 2;
stAioAttr.eBitwidth = E_MI_AUDIO_BIT_WIDTH_16;
stAioAttr.eSamplerate = E_MI_AUDIO_SAMPLE_RATE_32000;
stAioAttr.eSoundmode = E_MI_AUDIO_SOUND_MODE_MONO;
        stAioAttr.u32FrmNum = 30;
stAioAttr.u32PtNumPerFrm = 320*4;
eOutSampleRate = AUDIO_SAMPLE_RATE_8000;
ret = MI_AI_EnableReSmp(AiDev, AiChn, eOutSampleRate);
if (MI_OK != ret)
{
    printf("func(%s) line(%d): failed, ret:0x%x\n", __FUNCTION__,__LINE__, ret);
    return ret;
}
```

1.2.12 MI_AI_DisableReSmp

➢ 功能

禁用 AI 重采样

➢ 语法

MI_S32 MI_AI_DisableReSmp(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 支持的通道范围由AI设备属性中的最大通道个数 u32ChnCnt 与声道模式eSoundmode 决定。	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a libSRC_LINUX.so

※ 注意

- 不再使用 AI 重采样功能的话，应该调用此接口将其禁用。
- 要求在调用此接口之前，先禁用使用该 AI 设备相应通道音频数据的通道，否则可能导致该接口调用失败。

➢ 举例

无

1.2.13 MI_AI_SetVqeAttr

➢ 功能

设置 AI 的声音质量增强功能相关属性

➢ 语法

MI_S32 MI_AI_SetVqeAttr(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AUDIO_DEV AoDevId, MI_AO_CHN AoChn, MI_AI_VqeConfig_t *pstVqeConfig);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
AoDevId	用于回声抵消的AO设备号。	输入
AoChn	用于回声抵消的AO通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
pstVqeConfig	音频输入声音质量增强配置结构体指针	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

- 启用声音质量增强功能前必须先设置相对应 AI 通道的声音质量增强功能相关属性。
- 设置 AI 的声音质量增强功能相关属性前，必须先使能对应的 AI 通道。
- 相同 AI 信道的声音质量增强功能不支持动态设置属性，重新设置 AI 通道的声音质量增强功能相关属性时，需要先关闭 AI 通道的声音质量功能，再设置 AI 通道的声音质量增强功能相关属性。

➢ 举例

无。

1.2.14 MI_AI_GetVqeAttr

➢ 功能

获取 AI 的声音质量增强功能相关属性。

➢ 语法

MI_S32 MI_AI_GetVqeAttr(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AI_VqeConfig_t *pstVqeConfig);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
pstVqeConfig	音频输入声音质量增强配置结构体指针	输出

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

无。

➢ 举例

无。

1.2.15 MI_AI_EnableVqe

➢ 功能

使能 AI 的声音质量增强功能。

➢ 语法

MI_S32 MI_AI_EnableVqe(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖
- 头文件: mi_ai.h
- 库文件: libmi.a libAPC_LINUX.so

※ 注意
- 启用声音质量增强功能前必须先启用相对应的 AI 通道。
- 多次使能相同 AI 通道的声音质量增强功能时，返回成功。
- 禁用 AI 通道后，如果重新启用 AI 通道，并使用声音质量增强功能，需调用此接口重新启用声音质量增强功能。

➢ 举例

无。

1.2.16 MI_AI_DisableVqe

➢ 功能

禁用 AI 的声音质量增强功能。

➢ 语法

MI_S32 MI_AI_DisableVqe(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖
- 头文件: mi_ai.h
- 库文件: libmi.a libAPC_LINUX.so

※ 注意
- 不再使用 AI 声音质量增强功能时，应该调用此接口将其禁用。
- 多次禁用相同 AI 通道的声音质量增强功能，返回成功。

➢ 举例

无。

1.2.17 MI_AI_ClrPubAttr

- 功能

  清除 AI 设备属性。

- 语法

  MI_S32 MI_AI_ClrPubAttr(MI_AUDIO_DEV AiDevId);

- 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入

- 返回值

  返回值
  - 0　成功。
  - 非 0　失败，参照错误码。

- 依赖
  - 头文件: mi_ai.h
  - 库文件: libmi.a

※ 注意
  - 清除设备属性前，需要先停止设备。

- 举例

  无。

1.2.18 MI_AI_SaveFile

- 功能

  开启音频输入保存文件功能

- 语法

  MI_S32 MI_AI_SaveFile(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AUDIO_SaveFileInfo_t *pstSaveFileInfo);

- 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
pstSaveFileInfo	音频保存文件属性结构体指针。	输入

➢ 返回值

返回值 { 0 成功。 / 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

- 此接口仅用于 dump AI 中 VQE 处理前后的文件，没有使能 VQE 功能时，使用该接口 dump AI 数据无效。调用后会在指定目录下写出三个指定大小文件。AudIn.pcm 为 VQE 处理前的输入帧，RefIn.pcm 为 VQE 处理前的回声抵消参考帧，VqeOut.pcm 为 VQE 处理后的输出帧

➢ 相关主题

无。

1.2.19 MI_AI_SetVqeVolume

➢ 功能

设置声音质量增强功能中的音量大小

➢ 语法

MI_S32 MI_AI_SetVqeVolume(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_S32 s32VolumeDb);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
s32VolumeDb	声音质量增强功能中的音量大小（以Db 为单位）。	输入

➢ 返回值

返回值 { 0 成功。 / 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

无。

➢ 举例

无。

1.2.20 MI_AI_GetVqeVolume

➢ 功能

获取声音质量增强功能中的音量大小

➢ 语法

MI_S32 MI_AI_GetVqeVolume(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_S32 *ps32VolumeDb);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
ps32VolumeDb	声音质量增强功能中的音量大小（以Db 为单位）。	输出

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

无。

➢ 举例

无。

1.2.21 MI_AI_SetAencAttr

➢ 功能

设置 AI 编码功能相关属性

➢ 语法

MI_S32 MI_AI_SetAencAttr (MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AI_AencConfig_t *pstAencConfig);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
pstAencConfig	音频编码配置结构体指针	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a libg711.so libg726.so

※ 注意

无。

➢ 举例

无。

1.2.22 MI_AI_GetAencAttr

➢ 功能

获取 AI 编码功能相关属性

➢ 语法

MI_S32 MI_AI_GetAencAttr (MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AI_AencConfig_t *pstAencConfig);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
pstAencConfig	音频编码配置结构体指针	输出

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a libg711.so libg726.so

※ 注意

无。

1.2.23 MI_AI_EnableAenc

- 功能

  使能 AI 编码功能。

- 语法

  MI_S32 MI_AI_EnableAenc (MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn);

- 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入

- 返回值

  返回值：
  - 0　成功。
  - 非 0　失败，参照错误码。

- 依赖
  - 头文件: mi_ai.h
  - 库文件: libmi.a libg711.so libg726.so

※ 注意

  无。

1.2.24 MI_AI_DisableAenc

- 功能

  禁用 AI 编码功能。

- 语法

  MI_S32 MI_AI_DisableAenc (MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn);

- 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入

- 返回值

  返回值：
  - 0　成功。
  - 非 0　失败，参照错误码。

➢ 依赖
- 头文件: mi_ai.h
- 库文件: libmi.a libg711.so libg726.so

※ 注意

无。

1.2.25 MI_AI_SetAedAttr

➢ 功能

设置 AI 声音事件检测功能。

➢ 语法

MI_S32 MI_AI_SetAedAttr(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AI_AedConfig_t *pstAedConfig);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
pstAedConfig	声音事件检测配置结构体指针	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖
- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

无。

1.2.26 MI_AI_GetAedAttr

➢ 功能

获取 AI 声音事件检测功能配置。

➢ 语法

MI_S32 MI_AI_GetAedAttr(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AI_AedConfig_t *pstAedConfig);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
pstAedConfig	声音事件检测配置结构体指针	输出

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a

※ 注意

无。

1.2.27 MI_AI_EnableAed

➢ 功能

使能 AI 声音事件检测功能。

➢ 语法

MI_S32 MI_AI_EnableAed(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a libAED_LINUX.so

※ 注意

无。

1.2.28 MI_AI_DisableAed

➢ 功能

禁止 AI 声音事件检测功能。

➢ 语法

MI_S32 MI_AI_DisableAed(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖

- 头文件: mi_ai.h
- 库文件: libmi.a libAED_LINUX.so

※ 注意

无。

1.2.29 MI_AI_GetAedResult

➢ 功能

获取 AI 声音事件检测结果。

➢ 语法

MI_S32 MI_AI_GetAedResult(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AI_AedResult_t *pstAedResult);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
pstAedResult	声音事件检测结果结构体指针	输出

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖
- 头文件: mi_ai.h
- 库文件: libmi.a libAED_LINUX.so

※ 注意

无。

1.2.30 MI_AI_SetExtAecChn

➢ 功能

设置回声消除功能参考的 AI 通道。

➢ 语法

MI_S32 MI_AI_SetExtAecChn(MI_AUDIO_DEV AiDevId, MI_AI_CHN AiChn, MI_AI_CHN AiAECSndChn);

➢ 形参

参数名称	描述	输入/输出
AiDevId	音频设备号	输入
AiChn	音频输入通道号。 取值范围：[0, AUIDO_MAX_CHN_NUM]。	输入
AiAECSndChn	参考的AI通道号	输入

➢ 返回值

返回值
- 0 成功。
- 非 0 失败，参照错误码。

➢ 依赖
- 头文件: mi_ai.h

※ 注意

无。

2. AI 数据类型

AI 模块相关数据类型定义如下：

MI_AUDIO_DEV	定义音频输入/输出设备编号
MI_AUDIO_MAX_CHN_NUM	定义音频输入/输出设备的最大通道数
MI_AI_CHN	定义音频输入通道
MI_AUDIO_SampleRate_e	定义音频采样率
MI_AUDIO_Bitwidth_e	定义音频采样精度
MI_AUDIO_Mode_e	定义音频输入输出工作模式
MI_AUDIO_SoundMode_e	定义音频声道模式
MI_AUDIO_Attr_t	定义音频输入输出设备属性结构体
MI_AI_ChnParam_t	定义通道参数结构体
MI_AUDIO_Frame_t	定义音频帧数据结构体
MI_AUDIO_AecFrame_t	定义回声抵消参考帧信息结构体
MI_AUDIO_SaveFileInfo_t	定义音频保存文件功能配置信息结构体
MI_AI_VqeConfig_t	定义音频输入声音质量增强配置信息结构体
MI_AUDIO_HpfConfig_t	定义音频高通滤波功能配置信息结构体
MI_AUDIO_HpfFreq_e	定义音频高通滤波截止频率
MI_AI_AecConfig_t	定义音频回声抵消配置信息结构体
MI_AUDIO_AnrConfig_t	定义音频语音降噪功能配置信息结构体
MI_AUDIO_AgcConfig_t	定义音频自动增益控制配置信息结构体
MI_AUDIO_EqConfig_t	定义音频均衡器功能配置信息结构体
MI_AI_AecConfig_t	定义音频回音消除功能配置信息结构体
MI_AI_AencConfig_t	定义音频编码功能配置信息结构体
MI_AUDIO_AlgorithmMode_e	定义音频算法的运行模式
MI_AI_AedConfig_t	定义声音事件检测功能配置信息结构体
MI_AUDIO_AedSensitivity_e	定义声音事件检测的灵敏度
MI_AI_AedResult_t	定义声音事件检测的结果
MI_AI_ChnGainConfig_t	定义音频通道增益设置结构体

2.1. MI_AUDIO_DEV

➢ 说明

定义音频输入/输出设备编号。

➢ 定义

typedef MI_S32 MI_AUDIO_DEV

※ 注意事项

无。

➢ 相关数据类型及接口

无。

2.2. MI_AUDIO_MAX_CHN_NUM

➢ 说明

定义音频输入/输出设备的最大通道数。

➢ 定义

#define MI_AUDIO_MAX_CHN_NUM 16

※ 注意事项

无。

➢ 相关数据类型及接口

无。

2.3. MI_AI_CHN

➢ 说明

定义音频输入通道。

➢ 定义

typedef MI_S32 MI_AI_CHN

※ 注意事项

无。

➢ 相关数据类型及接口

无。

2.4. MI_AUDIO_SampleRate_e

➢ 说明

定义音频采样率。

➢ 定义

```
typedef enum
{
    E_MI_AUDIO_SAMPLE_RATE_8000 =8000, /* 8kHz sampling rate */
    E_MI_AUDIO_SAMPLE_RATE_16000 =16000, /* 16kHz sampling rate */
    E_MI_AUDIO_SAMPLE_RATE_32000 =32000, /* 32kHz sampling rate */
    E_MI_AUDIO_SAMPLE_RATE_48000 =48000, /* 48kHz sampling rate */
    E_MI_AUDIO_SAMPLE_RATE_INVALID,
}MI_AUDIO_SampleRate_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_SAMPLE_RATE_8000	8kHz 采样率
E_MI_AUDIO_SAMPLE_RATE_16000	16kHz 采样率
E_MI_AUDIO_SAMPLE_RATE_32000	32kHz 采样率
E_MI_AUDIO_SAMPLE_RATE_48000	48kHz 采样率

※ 注意事项

这里枚举值不是从 0 开始，而是与实际的采样率值相同。

➢ 相关数据类型及接口

MI_AUDIO_Attr_t。

2.5. MI_AUDIO_Bitwidth_e

➢ 说明

定义音频采样精度。

➢ 定义

```
typedef enum
{
    E_MI_AUDIO_BIT_WIDTH_16 =0, /* 16bit width */
    E_MI_AUDIO_BIT_WIDTH_24 =1, /* 24bit width */
    E_MI_AUDIO_BIT_WIDTH_MAX,
}MI_AUDIO_BitWidth_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_BIT_WIDTH_16	采样精度为 16bit 位宽
E_MI_AUDIO_BIT_WIDTH_24	采样精度为 24bit 位宽

※ 注意事项

目前软件只支持 16bit 位宽。

- 相关数据类型及接口
  无。

2.6. MI_AUDIO_Mode_e

- 说明
  定义音频输入输出设备工作模式。

- 定义

```
typedef enum
{
    E_MI_AUDIO_MODE_I2S_MASTER, /* I2S master mode */
    E_MI_AUDIO_MODE_I2S_SLAVE, /* I2S slave mode */
    E_MI_AUDIO_MODE_TDM_MASTER, /* TDM master mode */
    E_MI_AUDIO_MODE_MAX,
}MI_AUDIO_Mode_e;
```

- 成员

成员名称	描述
E_MI_AUDIO_MODE_I2S_MASTER	I2S 主模式
E_MI_AUDIO_MODE_I2S_SLAVE	I2S 从模式
E_MI_AUDIO_MODE_TDM_MASTER	TDM 主模式

※ 注意事项
主模式与从模式是否支持会依据不同的使用场景而有区别。

- 相关数据类型及接口
  MI_AUDIO_Attr_t

2.7. MI_AUDIO_SoundMode_e

- 说明
  定义音频声道模式。

- 定义

```
typedef enum
{
    E_MI_AUDIO_SOUND_MODE_MONO =0, /* mono */
    E_MI_AUDIO_SOUND_MODE_STEREO =1, /* stereo */
    E_MI_AUDIO_SOUND_MODE_QUEUE =2, /*all data in One chn */
    E_MI_AUDIO_SOUND_MODE_BUTT,
}MI_AUDIO_SoundMode_e
```

➢ 成员

成员名称	描述
E_MI_AUDIO_SOUND_MODE_MONO	单声道。
E_MI_AUDIO_SOUND_MODE_STEREO	双声道。
E_MI_AUDIO_SOUND_MODE_QUEUE	所有音频数据按顺序排列到 1 个通道里面，针对音频采集时使用

※ 注意事项

对于双声道模式，只应对左声道（即编号小于设备属性中通道数 u32ChnCnt 一半的通道）进行操作，SDK 内部会自动对右声道也进行相应的操作。

➢ 相关数据类型及接口

MI_AUDIO_Attr_t

2.8. MI_AUDIO_AencType_e

➢ 说明

定义音频编码类型。

➢ 定义

```
typedef enum
{
    E_MI_AUDIO_AENC_TYPE_G711A = 0,
    E_MI_AUDIO_AENC_TYPE_G711U,
    E_MI_AUDIO_AENC_TYPE_G726,
    E_MI_AUDIO_AENC_TYPE_INVALID,
}MI_AUDIO_AencType_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_AENC_TYPE_G711A	G711A 编码。
E_MI_AUDIO_AENC_TYPE_G711U	G711U 编码。
E_MI_AUDIO_AENC_TYPE_G726	G726 编码。

※ 注意事项

无

➢ 相关数据类型及接口

MI_AUDIO_AencG726Config_t

2.9. MI_AUDIO_G726Mode_e

➢ 说明

定义 G726 工作模式。

➢ 定义

```
typedef enum
{
    E_MI_AUDIO_G726_MODE_16 = 0,
    E_MI_AUDIO_G726_MODE_24,
    E_MI_AUDIO_G726_MODE_32,
    E_MI_AUDIO_G726_MODE_40,
    E_MI_AUDIO_G726_MODE_INVALID,
}MI_AUDIO_G726Mode_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_G726_MODE_16	G726 16K 比特率模式。
E_MI_AUDIO_G726_MODE_24	G726 24K 比特率模式。
E_MI_AUDIO_G726_MODE_32	G726 32K 比特率模式。
E_MI_AUDIO_G726_MODE_40	G726 40K 比特率模式

※ 注意事项

无

➢ 相关数据类型及接口

MI_AUDIO_AencG726Config_t

2.10. MI_AUDIO_I2sFmt_e

➢ 说明

I2S 格式设定。

➢ 定义

```
typedef enum
{
    E_MI_AUDIO_I2S_FMT_I2S_MSB,
    E_MI_AUDIO_I2S_FMT_LEFT_JUSTIFY_MSB,
}MI_AUDIO_I2sFmt_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_I2S_FMT_I2S_MSB	I2S 标准格式，最高位优先
E_MI_AUDIO_I2S_FMT_LEFT_JUSTIFY_MSB	I2S 左对齐格式，最高位优先

※ 注意事项

无

➢ 相关数据类型及接口

MI_AUDIO_I2sConfig_t

2.11. MI_AUDIO_I2sMclk_e

➢ 说明

I2S MCLK 设定

➢ 定义

```
typedef enum{
    E_MI_AUDIO_I2S_MCLK_0,
    E_MI_AUDIO_I2S_MCLK_12_288M,
    E_MI_AUDIO_I2S_MCLK_16_384M,
    E_MI_AUDIO_I2S_MCLK_18_432M,
    E_MI_AUDIO_I2S_MCLK_24_576M,
}MI_AUDIO_I2sMclk_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_I2S_MCLK_0	关闭 MCLK
E_MI_AUDIO_I2S_MCLK_12_288M	设置 MCLK 为 12.88M
E_MI_AUDIO_I2S_MCLK_16_384M	设置 MCLK 为 16.384M
E_MI_AUDIO_I2S_MCLK_18_432M	设置 MCLK 为 18.432M
E_MI_AUDIO_I2S_MCLK_24_576M	设置 MCLK 为 24.576M

※ 注意事项

无

➢ 相关数据类型及接口

MI_AUDIO_I2sConfig_t

2.12. MI_AUDIO_I2sConfig_t

➢ 说明

定义 I2S 属性结构体。

➢ 定义

```
typedef struct MI_AUDIO_I2sConfig_s
{
    MI_AUDIO_I2sFmt_e eFmt;
    MI_AUDIO_I2sMclk_e eMclk;
    MI_BOOL bSyncClock;
}MI_AUDIO_I2sConfig_t;
```

➢ 成员

成员名称	描述
eFmt	I2S 格式设置。 静态属性。
eMclk	I2S MCLK 时钟设置。 静态属性。
bSyncClock	AI 同步 AO 时钟，暂未支持，需设置成 FALSE。 静态属性。

※ 注意事项

无。

➢ 相关数据类型及接口

MI_AUDIO_Attr_t

2.13. MI_AUDIO_Attr_t

➢ 说明

定义音频输入输出设备属性结构体。

➢ 定义

```
typedef struct MI_AUDIO_Attr_s
{
    MI_AUDIO_SampleRate_e eSamplerate; /*sample rate*/
    MI_AUDIO_BitWidth_e eBitwidth; /*bitwidth*/
    MI_AUDIO_Mode_e eWorkmode; /*master or slave mode*/
    MI_AUDIO_SoundMode_e eSoundmode; /*momo or stereo*/
    MI_U32 u32FrmNum; /*frame num in buffer*/
    MI_U32 u32PtNumPerFrm; /*number of samples*/
    MI_U32 u32CodecChnCnt; /*channel number on Codec */
    MI_U32 u32ChnCnt;
    union{
        MI_AUDIO_I2sConfig_t stI2sConfig;
    }WorkModeSetting;
}MI_AUDIO_Attr_t;
```

➢ 成员

成员名称	描述
eSamplerate	音频采样率。 静态属性。
eBitwidth	音频采样精度(从模式下,此参数必须和音频 AD/DA 的采样精度匹配）。 静态属性。
eWorkmode	音频输入输出工作模式。 静态属性。
eSoundmode	音频声道模式。 静态属性。
u32FrmNum	缓存帧数目。 取值范围：[2, MAX_AUDIO_FRAME_NUM]。 静态属性。
u32PtNumPerFrm	每帧的采样点个数。 取值范围为：128, 128*2,…,128*N。 静态属性。
u32CodecChnCnt	支持的 codec 通道数目，即决定了 codec 到 AIO 的 I2S/PCM 时序（时分复用关系），取值范围 1、2、4、8、16（最大取值为 MI_AUDIO_MAX_CHN_NUM）。与对接的 codec 和 u32ChnCnt 相关，要求 u32CodecChnCnt 大于等于 u32ChnCnt。
u32ChnCnt	支持的通道数目，实际可使能的最大通道数。取值：1、2、4、8、16。（输入最多支持 MI_AUDIO_MAX_CHN_NUM

成员名称	描述
	个通道，输出最多支持 2 个通道）
MI_AUDIO_I2sConfig_t stI2sConfig;	设置 I2S 工作属性

※ 注意事项

无。

➢ 相关数据类型及接口

MI_AI_SetPubAttr

2.14. MI_AI_ChnParam_t

➢ 说明

定义通道参数结构体。

➢ 定义

```
typedef struct MI_AI_ChnParam_s
{
    MI_AI_ChnGainConfig_t stChnGain;
    MI_U32 u32Reserved;
} MI_AI_ChnParam_t
```

➢ 成员

成员名称	描述
stChnGain	音频通道的增益设定。
u32Reserved	保留，未使用。

※ 注意事项

无。

➢ 相关数据类型及接口

MI_AI_SetChnParam
MI_AI_GetChnParam

2.15. MI_AUDIO_Frame_t

➢ 说明

定义音频帧结构体。

➢ 定义

```
typedef struct MI_AUDIO_Frame_s
{
    MI_AUDIO_BitWidth_e eBitwidth; /*audio frame bitwidth*/
    MI_AUDIO_SoundMode_e eSoundmode; /*audio frame momo or stereo mode*/
    void *apVirAddr[MI_AUDIO_MAX_CHN_NUM];
    MI_U64 u64TimeStamp;/*audio frame timestamp*/
    MI_U32 u32Seq; /*audio frame seq*/
    MI_U32 u32Len; /*data lenth per channel in frame*/
```

```
    MI_U32 au32PoolId[2];
}MI_AUDIO_Frame_t;
```

➢ 成员

成员名称	描述
eBitwidth	音频采样精度
eSoundmode	音频声道模式。
pVirAddr[MI_AUDIO_MAX_CHN_NUM]	音频帧数据虚拟地址。
u64TimeStamp	音频帧时间戳。 以 μs 为单位
u32Seq	音频帧序号。
u32Len	音频帧长度。 以 byte 为单位。
u32PoolId[2]	音频帧缓存池 ID。

※ 注意事项

- u32Len（音频帧长度）指单个声道的数据长度。
- 单声道数据直接存放，采样点数为 u32PtNumPerFrm，长度为 u32Len；立体声数据按左右声道分开存放，先存放采样点为 u32PtNumPerFrm、长度为 u32Len 的左声道数据，然后存放采样点为 u32PtNumPerFrm，长度为 u32Len 的右声道数据。

➢ 相关数据类型及接口

无。

2.16. MI_AUDIO_AecFrame_t

➢ 说明

定定义音频回声抵消参考帧信息结构体。

➢ 定义

```
typedef struct MI_AUDIO_AecFrame_s
{
    MI_AUDIO_Frame_t stRefFrame; /* aec reference audio frame */
    MI_BOOL bValid; /* whether frame is valid */
}MI_AUDIO_AecFrame_t;
```

➢ 成员

成员名称	描述
stRefFrame	回声抵消参考帧结构体。
bValid	参考帧有效的标志。 取值范围： TRUE：参考帧有效。 FALSE：参考帧无效，无效时不能使用此参考帧进行回声抵消。

※ 注意事项

无。

➢ 相关数据类型及接口

无。

2.17. MI_AUDIO_SaveFileInfo_t

➢ 说明

定义音频保存文件功能配置信息结构体。

➢ 定义

```
typedef struct MI_AUDIO_SaveFileInfo_s
{
    MI_BOOL bCfg;
    MI_U8 szFilePath[256];
    MI_U32 u32FileSize; /*in KB*/
} MI_AUDIO_SaveFileInfo_t
```

➢ 成员

成员名称	描述
bCfg	配置使能开关。
szFilePath	音频文件的保存路径
u32FileSize	文件大小，取值范围[1, 10240]KB。

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_SaveFile

2.18. MI_AI_VqeConfig_t

➢ 说明

定义音频输入声音质量增强配置信息结构体。

➢ 定义

```
typedef struct MI_AI_VqeConfig_s
{
    MI_BOOL bHpfOpen;
    MI_BOOL bAecOpen;
    MI_BOOL bAnrOpen;
    MI_BOOL bAgcOpen;
    MI_BOOL bEqOpen;
    MI_S32 s32WorkSampleRate;
    MI_S32 s32FrameSample;
    MI_AUDIO_HpfConfig_t stHpfCfg;
    MI_AI_AecConfig_t stAecCfg;
    MI_AUDIO_AnrConfig_t stAnrCfg;
    MI_AUDIO_AgcConfig_t stAgcCfg;
    MI_AUDIO_EqConfig_t stEqCfg;
}MI_AI_VqeConfig_t;
```

➢ 成员

成员名称	描述
bHpfOpen	高通滤波功能是否使能标志。
bAecOpen	回声抵消功能是否使能标志。
bAnrOpen	语音降噪功能是否使能标志。
bAgcOpen	自动增益控制功能是否使能标志
bEqOpen	均衡器功能是否使能标志
s32WorkSampleRate	工作采样频率。该参数为内部功能算法工作采样率。取值范围：8KHz/16KHz。默认值为 8KHz。
s32FrameSample	VQE 的帧长，即采样点数目。只能设置 128
stHpfCfg	高通滤波功能相关配置信息。
stAecCfg	回声抵消功能相关配置信息。
stAnrCfg	语音降噪功能相关配置信息。
stAgcCfg	自动增益控制相关配置信息。
stEqCfg	均衡器相关配置信息。

※ 注意事项

无。

➢ 相关数据类型及接口

无。

2.19. MI_AUDIO_HpfConfig_t

➢ 说明

定义音频高通滤波功能配置信息结构体。

➢ 定义

```
typedef struct MI_AUDIO_HpfConfig_s
{
    MI_AUDIO_AlgorithmMode_e eMode;
    MI_AUDIO_HpfFreq_e eHpfFreq; /*freq to be processed*/
} MI_AUDIO_HpfConfig_t;
```

➢ 成员

成员名称	描述
eMode	音频算法的运行模式。
eHpfFreq	高通滤波截止频率选择。 80：截止频率为 80Hz； 120：截止频率为 120Hz； 150：截止频率为 150Hz。 默认值 150。

※ 注意事项

无。

➢ 相关数据类型及接口

MI_AI_VqeConfig_t

2.20. MI_AUDIO_HpfFreq_e

➢ 说明

定义音频高通滤波截止频率。

➢ 定义

```
typedef enum
{
    E_MI_AUDIO_HPF_FREQ_80 = 80, /* 80Hz */
    E_MI_AUDIO_HPF_FREQ_120 = 120, /* 120Hz */
    E_MI_AUDIO_HPF_FREQ_150 = 150, /* 150Hz */
    E_MI_AUDIO_HPF_FREQ_BUTT,
} MI_AUDIO_HpfFreq_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_HPF_FREQ_80	截止频率为 80Hz。
E_MI_AUDIO_HPF_FREQ_120	截止频率为 120Hz。
E_MI_AUDIO_HPF_FREQ_150	截止频率为 150Hz。

※ 注意事项

默认配置为 150Hz

➢ 相关数据类型及接口

MI_AI_VqeConfig_t

2.21. MI_AI_AecConfig_t

➢ 说明

定义音频回声抵消配置信息结构体。

➢ 定义

```
typedef struct MI_AI_AecConfig_s
{
    MI_BOOL bComfortNoiseEnable;
    MI_S16 s16DelaySample;
    MI_U32 u32AecSupfreq[6];
    MI_U32 u32AecSupIntensity[7];
    MI_S32 s32Reserved;
} MI_AI_AecConfig_t;
```

➢ 成员

成员名称	描述
bComfortNoiseEnable	是否添加噪音。 0：不添加； 1：添加。

成员名称	描述
s16DelaySample	采样点样本延迟个数。 仅在 AEC 为立体声处理时有效，默认值为 0。
u32AecSupfreq	回声消除保护频率范围，后 1 个数据必须大于等于前 1 个数据。 如：u32AecSupfreq[0] = 10，则：u32AecSupfreq[1]必须大于等于 10。 当前采样率对应的最高频率平均分成 127 份，频率范围则是对应多少份组成一个频带。 如：当前采样率为 16K，对应的最大频率为 8K，每一份为 8000 / 127 ≈ 63Hz，在推荐值{4 6 36 49 50 51}的设定下，保护范围为 {0~4 * 63Hz, 4~6 * 63Hz, 6~36 * 63Hz, 36~49 * 63Hz, 49~50 * 63Hz, 50~51 * 63Hz, 51-127 * 63Hz} = {0~252Hz, 252~378Hz, 378~2268Hz, 2268~3087Hz, 3087~3150Hz, 3150~3213Hz, 3213Hz~8000Hz} 范围[1, 127]；步长 1 推荐值{4, 6, 36, 49, 50, 51}
u32AecSupIntensity	回音消除保护力度，数值越小保护效果越强。此参数与 u32AecSupfreq 相对应，u32AecSupIntensity[0]对应 0~u32AecSupfreq[0], u32AecSupIntensity[1]对应 u32AecSupfreq[0]~ u32AecSupfreq[1]，以此类推。 范围[0, 15]；步长 1 推荐值{5, 4, 4, 5, 10, 10, 10}

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_VqeConfig_t

2.22. MI_AUDIO_AnrConfig_t

➢ 说明

定义音频语音降噪功能配置信息结构体。

➢ 定义

```
typedef struct MI_AUDIO_AnrConfig_s
{
    MI_AUDIO_AlgorithmMode_e eMode;
    MI_U32      u32NrIntensity;
    MI_U32      u32NrSmoothLevel;
    MI_AUDIO_NrSpeed_e eNrSpeed;
} MI_AUDIO_AnrConfig_t;
```

➢ 成员

成员名称	描述
eMode	音频算法的运行模式 注：Anr 的模式选择将会在一定程度上影响 Agc 的功能
u32NrIntensity	降噪力度配置，配置值越大降噪力度越高，但同时也会带来细节音的丢失/损伤。 范围[0,30]；步长 1 默认值 20。
u32NrSmoothLevel	平滑化程度，值越大越平滑 范围[0,10]；步长 1 默认值 10。
eNrSpeed	噪声收敛速度，低速，中速，高速 默认值中速。

※ 注意事项

在 Anr 和 Agc 都有使能的情况下，当 Anr 设定为 user mode 时，Agc 会对音频数据做频域处理，会评估出语音信号再做相应的增减，而当 Anr 设定为 default/music mode 时，Agc 会对音频数据做时域处理，对全频段的数据进行增减。

➢ 相关数据类型及接口

MI_AI_VqeConfig_t

2.23. MI_AUDIO_NrSpeed_e

➢ 说明

定义噪声收敛速度

➢ 定义

```
typedef enum
 {
        E_MI_AUDIO_NR_SPEED_LOW,
        E_MI_AUDIO_NR_SPEED_MID,
        E_MI_AUDIO_NR_SPEED_HIGH
}MI_AUDIO_NrSpeed_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_NR_SPEED_LOW	低速。
E_MI_AUDIO_NR_SPEED_MID	中速。
E_MI_AUDIO_NR_SPEED_HIGH	高速。

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_VqeConfig_t

2.24. MI_AUDIO_AgcConfig_t

- 说明

  定义音频自动增益控制配置信息结构体。

- 定义

```
typedef struct MI_AUDIO_AgcConfig_s
{
    MI_AUDIO_AlgorithmMode_e eMode;
    AgcGainInfo_t stAgcGainInfo;
    MI_U32       u32DropGainMax;
    MI_U32       u32AttackTime;
    MI_U32       u32ReleaseTime;
    MI_S16       s16Compression_ratio_input[5];
    MI_S16       s16Compression_ratio_output[5];
    MI_S32       s32TargetLevelDb;
    MI_S32       s32NoiseGateDb;
    MI_U32       u32NoiseGateAttenuationDb;
} MI_AUDIO_AgcConfig_t;
```

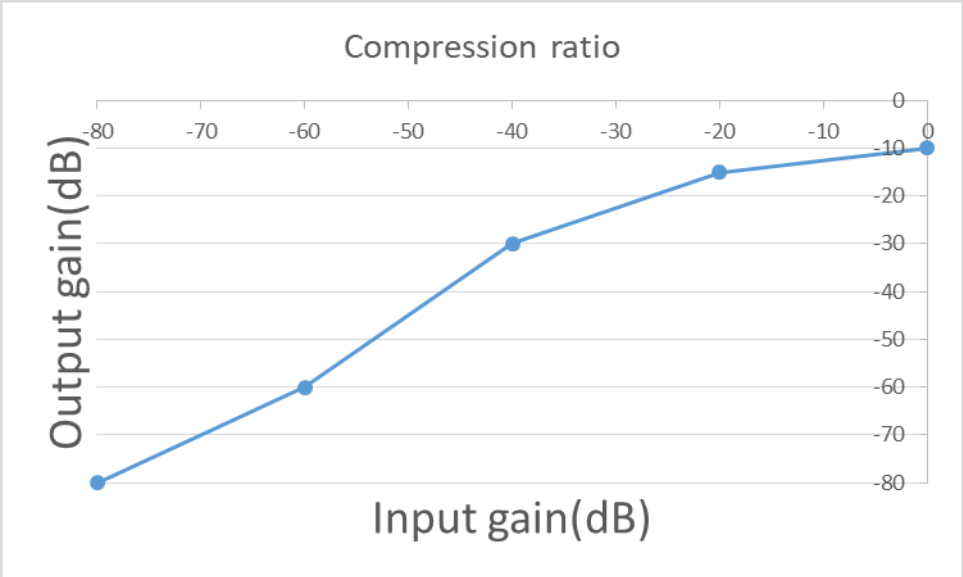
- 成员

成员名称	描述
eMode	音频算法的运行模式
stAgcGainInfo	定义 AGC 增益的最大、最小和初始值
u32DropGainMax	增益下降的最大值，防止输出饱和 范围[0,60]；步长 1 默认值 55。
u32AttackTime	增益下降时间区间长度，以 16 毫秒为 1 单位 范围[1,20]；步长 1 默认值 0。
u32ReleaseTime	增益上升时间区间长度，以 16 毫秒为 1 单位 范围[1,20] ；步长 1 默认值 0。
s16Compression_ratio_input[5]	输入压缩比，必须配合 s16Compression_ratio_output 使用，透过多个转折点实现多斜率的曲线 范围[-80,0]dBFS；步长 1
s16Compression_ratio_output[5]	输出压缩比，必须配合 s16Compression_ratio_input 使用，透过多个转折点实现多斜率的曲线 范围[-80,0]dBFS；步长 1
s32TargetLevelDb	目标电平，经过处理后的最大电平门限 范围[-80,0]dB；步长 1 默认值 0。
s32NoiseGateDb	噪声底值 范围[-80,0]；步长 1 注：当值为-80，噪声底值将不起作用 默认值-55。
u32NoiseGateAttenuationDb	当噪声底值起效果时，输入源的衰减百分比 范围[0,100]；步长 1 默认值 0。

※ 注意事项

在 Anr 和 Agc 都有使能的情况下，当 Anr 设定为 user mode 时，Agc 会对音频数据做频域处理，会评估出语音信号再做相应的增减，而当 Anr 设定为 default/music mode 时，Agc 会对音频数据做时域处理，对全频段的数据进行增减。

而 s16Compression_ratio_input 和 s16Compression_ratio_output 则需要根据所需要的增益曲线来设定。如下面的折线图所示，在输入增益为-80~0dB 划分为四段斜率，-80dB~-60dB 范围内保持原来的增益，斜率为 1，-60dB~-40dB 范围内需要稍微提高增益，斜率为 1.5，-40dB~-20dB 范围内斜率为 1.25，-20dB~0dB 范围内斜率为 0.25。根据曲线的转折点对 s16Compression_ratio_input 和 s16Compression_ratio_output 设置，若不需要那么多段曲线，则将数组不需要的部分填 0。



➢ 相关数据类型及接口

MI_AI_VqeConfig_t

2.25. AgcGainInfo_t

➢ 说明

AGC 增益的取值

➢ 定义

```
typedef struct AgcGainInfo_s{
    MI_S32    s32GainMax;
    MI_S32    S32GainMin;
    MI_S32    s32GainInit;
}AgcGainInfo_t;
```

➢ 成员

成员名称	描述
s32GainMax	增益最大值 范围[0,30]；步长 1 默认值 15。
s32GainMin	增益中间值 范围[-20,30]；步长 1 默认值 0。

成员名称	描述
s32GainInit	增益最小值 范围[-20,30]；步长 1 默认值 0。

※ 注意事项
无

➢ 相关数据类型及接口
MI_AI_VqeConfig_t

2.26. MI_AUDIO_EqConfig_t

➢ 说明
定义音频均衡器功能配置信息结构体。

➢ 定义

```
typedef struct MI_AUDIO_EqConfig_s
{
    MI_AUDIO_AlgorithmMode_e eMode;
    MI_S16        s16EqGainDb[129];
} MI_AUDIO_EqConfig_t;
```

➢ 成员

成员名称	描述
eMode	音频算法的运行模式
s16EqGainDb[129]	均衡器增益调节取值,将当前采样率的频率范围分成 129 个频率范围来进行调节 范围[-50,20]；步长 1 默认值 0。 如：当前采样率为 16K，对应的最高频率为 8K，8000 / 129 ≈ 62Hz,则单个调节的频率范围为 62Hz,将 0-8K 划分成{0-1＊62Hz，1-2＊62Hz,2-3＊62Hz,…,128-129＊62Hz} = {0-62Hz，62-124Hz，124-186Hz，…，7938-8000Hz}，每段对应一个增益值

※ 注意事项
无

➢ 相关数据类型及接口
MI_AI_VqeConfig_t

2.27. MI_AI_AencConfig_t

➢ 说明
定义音频编码功能配置信息结构体。

➢ 定义

```
typedef struct MI_AI_AencConfig_s
{
    MI_AUDIO_AencType_e    eAencType;
    union
    {
        MI_AUDIO_AencG711Config_t    stAencG711Cfg;
        MI_AUDIO_AnecG726Config_t    stAencG726Cfg;
    };
}MI_AI_AencConfig_t;
```

➢ 成员

成员名称	描述
eAencType	音频编码类型。
stAencG711Cfg	G711 编码相关配置信息。
stAencG726Cfg	G726 编码相关配置信息。

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_SetAencAttr

2.28. MI_AUDIO_AencG711Config_t

➢ 说明

定义音频编码功能配置信息结构体。

➢ 定义

```
typedef struct MI_AUDIO_AencG711Config_s
{
    MI_AUDIO_SampleRate_e eSamplerate;
    MI_AUDIO_SoundMode_e eSoundmode;
}MI_AUDIO_AencG711Config_t;
```

➢ 成员

成员名称	描述
eSamplerate	音频采样率。
eSoundmode	音频声道模式。

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_SetAencAttr

2.29. MI_AUDIO_AencG726Config_t

➢ 说明

定义音频编码功能配置信息结构体。

➢ 定义

```
typedef struct MI_AUDIO_AencG726Config_s
{
    MI_AUDIO_SampleRate_e eSamplerate;
    MI_AUDIO_SoundMode_e eSoundmode;
    MI_AUDIO_G726Mode_e   eG726Mode;
}MI_AUDIO_AencG726Config_t;
```

➢ 成员

成员名称	描述
eSamplerate	音频采样率。
eSoundmode	音频声道模式。
eG726Mode	G726 工作模式。

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_SetAencAttr

2.30. MI_AUDIO_AlgorithmMode_e

➢ 说明

音频算法运行的模式。

➢ 定义

```
typedef enum
{
    E_MI_AUDIO_ALGORITHM_MODE_DEFAULT,
    E_MI_AUDIO_ALGORITHM_MODE_USER,
    E_MI_AUDIO_ALGORITHM_MODE_MUSIC,
    E_MI_AUDIO_ALGORITHM_MODE_INVALID,
}MI_AUDIO_AlgorithmMode_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_ALGORITHM_MODE_DEFAULT	默认运行模式 当使用该模式时，将使用算法的默认参数
E_MI_AUDIO_ALGORITHM_MODE_USER	用户模式 当使用该模式时，需要用户重新设定所有参数
E_MI_AUDIO_ALGORITHM_MODE_MUSIC	音乐模式 仅有 Anr 具有此模式，当为此模式时，Agc 不会进行 speech enhancment（语音增强）处理

※ 注意事项

在 Anr 和 Agc 都有使能的情况下，当 Anr 设定为 user mode 时，Agc 会对音频数据做频域处理，会评估出语音信号再做相应的增减，而当 Anr 设定为 default/music mode 时，Agc 会对音频数据做时域处理，对全频段的数据进行增减。

➢ 相关数据类型及接口

MI_AUDIO_HpfConfig_t，MI_AUDIO_AnrConfig_t，MI_AUDIO_AgcConfig_t，MI_AUDIO_EqConfig_t

2.31. MI_AI_AedConfig_t

➢ 说明

声音事件检测功能配置信息结构体。

➢ 定义

```
typedef struct MI_AI_AedConfig_s
{
    MI_BOOL bEnableNr;
    MI_AUDIO_AedSensitivity_e eSensitivity;
    MI_S32 s32OperatingPoint;
    MI_S32 s32VadThresholdDb;
    MI_S32 s32LsdThresholdDb;
}MI_AI_AedConfig_t;
```

➢ 成员

成员名称	描述
bEnableNr	是否启用声音事件检测的降噪功能
eSensitivity	声音事件检测功能的灵敏度
s32OperatingPoint	操作点 范围[-10, 10]，步长为 1 默认值为 0 注：提高操作点将会降低误报率，减小操作点将会降低漏测率
s32VadThresholdDb	Vad 的门槛值(dB) 范围[-80, 0]，步长为 1 默认值为-40
s32LsdThresholdDb	Lsd 的门槛值(dB) 范围[-80, 0]，步长为 1 默认值为-15

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_SetAedAttr，MI_AI_GetAedAttr

2.32. MI_AUDIO_AedSensitivity_e

➢ 说明

声音事件检测的灵敏度

➢ 定义

```
typedef enum
{
    E_MI_AUDIO_AED_SEN_LOW,
    E_MI_AUDIO_AED_SEN_MID,
    E_MI_AUDIO_AED_SEN_HIGH,
    E_MI_AUDIO_AED_SEN_INVALID,
}MI_AUDIO_AedSensitivity_e;
```

➢ 成员

成员名称	描述
E_MI_AUDIO_AED_SEN_LOW	低灵敏度
E_MI_AUDIO_AED_SEN_MID	中等灵敏度
E_MI_AUDIO_AED_SEN_HIGH	高灵敏度

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_AedConfig_t

2.33. MI_AI_AedResult_t

➢ 说明

声音事件检测结果结构体。

➢ 定义

```
typedef struct MI_AI_AedResult_s
{
    MI_BOOL bAcousticEventDetected;
    MI_BOOL bLoudSoundDetected;
}MI_AI_AedResult_t;
```

➢ 成员

成员名称	描述
bAcousticEventDetected	是否检测到声音事件
bLoudSoundDetected	是否检测到高分贝声音

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_GetAedResult

2.34. MI_AI_ChnGainConfig_t

➢ 说明

音频通道增益设置结构体。

➢ 定义

```
typedef struct MI_AI_ChnGainConfig_s
{
    MI_BOOL bEnableGainSet;
    MI_S16 s16FrontGain;
    MI_S16 s16RearGain;
}MI_AI_ChnGainConfig_t;
```

➢ 成员

成员名称	描述
bEnableGainSet	是否使能增益设置
s16FrontGain	前级增益
s16RearGain	后级增益

※ 注意事项

无

➢ 相关数据类型及接口

MI_AI_ChnParam_t

3. 错误码

AI API 错误码如表 3-1 所示：

表 3-1 AI API 错误码

宏定义	描述
MI_AI_ERR_INVALID_DEVID	音频输入设备号无效
MI_AI_ERR_INVALID_CHNID	音频输入信道号无效
MI_AI_ERR_ILLEGAL_PARAM	音频输入参数设置无效
MI_AI_ERR_NOT_ENABLED	音频输入设备或通道没有使能
MI_AI_ERR_NULL_PTR	输入参数空指标错误
MI_AI_ERR_NOT_CONFIG	音频输入设备属性未设置
MI_AI_ERR_NOT_SUPPORT	操作不支持
MI_AI_ERR_NOT_PERM	操作不允许
MI_AI_ERR_NOMEM	分配内存失败
MI_AI_ERR_NOBUF	音频输入缓存不足
MI_AI_ERR_BUF_EMPTY	音频输入缓存为空
MI_AI_ERR_BUF_FULL	音频输入缓存为满
MI_AI_ERR_SYS_NOTREADY	音频输入系统未初始化
MI_AI_ERR_BUSY	音频输入系统忙碌
MI_AI_ERR_VQE_ERR	音频输入 VQE 算法处理失败
MI_AI_ERR_AENC_ERR	音频输入编码算法处理失败
MI_AI_ERR_AED_ERR	声音检测算法处理失败