

# Rockchip麦克风阵列音频算法调试说明文档

---

## 前言

### 概述

本文提供一个麦克风阵列结构设计参考文档，工程师可以参照相关内容进行麦克风阵列设计。

### 产品版本

芯片名称	内核版本
全系列	通用

### 读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

# 目录

## Rockchip麦克风阵列音频算法调试说明文档

1. 概述
  - 1.1 音频算法特性
2. 音频算法说明
  - 2.1 音频算法模块说明
    - 2.1.1 辅助模块
    - 2.1.2 AEC模块
    - 2.1.3 Beamforming模块
  - 2.2 音频算法流程
    - 2.2.1 唤醒流程
    - 2.2.2 通话流程
  - 2.3 音频算法运行效果展示
    - 2.3.1 AEC模块效果
    - 2.3.2 BF模块效果
3. 音频算法接口调用说明
4. 音频算法参数调试说明
  - 4.1 全局模块参数设置
  - 4.2 AEC模块参数设置
  - 4.3 延时估计子模块参数设置
  - 4.4 BF模块参数设置
  - 4.5 ANR模块参数设置
  - 4.6 去混响模块参数设置
  - 4.7 AES模块参数设置
  - 4.8 AGC模块参数设置
  - 4.9 舒适噪声参数设置
  - 4.10 DTD模块参数设置
  - 4.11 推荐参数示例

# 1. 概述

---

## 1.1 音频算法特性

多麦克风阵列算法，包括回声消除AEC、波束形成BF、语音降噪ANR、自动增益AGC等。主要包括以下应用场景：

- 回声消除AEC用于消除麦克采集到的扬声器播放的声音
- 波束形成BF用于增强语音，抑制相干噪声和环境噪声
- 语音降噪ANR用于消除麦克采集到的环境噪音
- 自动增益AGC用于增强语音电平信号

# 2. 音频算法说明

---

多麦克风声学处理主要完成回声消除和语音增强任务。

## 2.1 音频算法模块说明

完整的算法包括的流程模块如图1所示，包括如下部分：

### 2.1.1 辅助模块

这些模块包括高通滤波（High pass-Filter）、预加重（pre-emphasis）、去加重（de-emphasis）等模块，其中，高通滤波模块用于滤除低频电路噪音，预加重用于对语音高频能量提升，增强传输过程高频信噪比，在输出端再通过去加重进行还原。

### 2.1.2 AEC模块

AEC模块包括MDF模块和Delay模块。

- MDF模块属于线性回声消除算法，用于消除回声
- Delay模块为延迟估计模块，用于估计回采信号与麦克风信号之间的延迟

### 2.1.3 Beamforming模块

Beamforming（以下简称BF）模块包括FAST\_AEC模块、WAKEUP模块、ANR模块、Dereverb模块、AES模块、AGC模块、CNG模块、DTD模块、Howling模块、EQ模块以及DOA模块。

- FAST\_AEC模块：线性回声消除算法，用于抑制残留回声
- WAKEUP模块：RK定制唤醒，主要完成特定唤醒词唤醒机器
- ANR模块：降噪模块，用于抑制环境噪音、风噪、冲击噪声

- Dereverb模块：去混响模块，用于消除环境混响
- AES模块：残留回声消除算法，用于进一步抑制残留回声；
- GSC模块：波束模块，输入为多麦音频时默认开启，当额外开启Fix/GSC/CHN\_SELECT时，启用固定方向波束/波束后滤波/定制化扩展mic；
- AGC模块：自动增益模块，用于控制语音电平信号
- CNG模块：舒适度噪声模块，用于添加舒适度噪声
- DTD模块：双讲检测模块，用于检测语音通话状态
- Howling模块：去啸叫模块，用于去除自激产生的啸叫；
- EQ模块：均衡器模块，用于调整频段增益；
- DOA模块：声源定位模块，用于定位声源方位；
- AINR模块：AI降噪模块，用于抑制除人声外的噪声；

## 2.2 音频算法流程

多麦克风阵列算法提供两种不同的语音通路。

### 2.2.1 唤醒流程

该流程通过唤醒进行机器应答以及定位，输出唤醒人所在位置的音频，具体流程如图1所示。

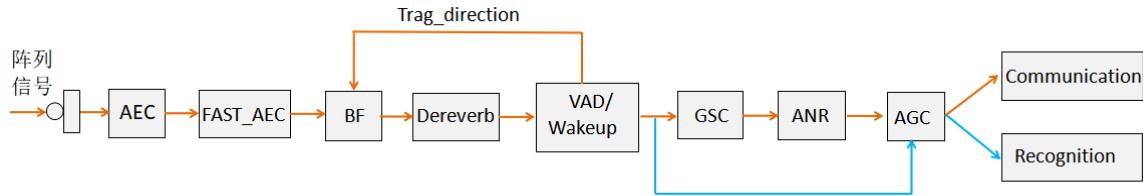


图 1 唤醒流程系统框图

1. 阵列采集阵列信息后，先通过AEC模块消除线性回声，BF内置FAST\_AEC的进行进一步的回声抑制；
2. 经过固定波束输出增强后的语音；
3. 接入Dereverb模块去除残留混响；
4. 通过Wakeup模块定位出声源位置；
5. 将定位的语音语音送入GSC后处理模块进行后滤波，（注：蓝色箭头云端识别流程，直接将定位语音送到AGC输出）；
6. 再将处理后的语音送到ANR模块进一步进行降噪；
7. 最后送到AGC模块进行语音增益，送到远端进行通话。

### 2.2.2 通话流程

该流程不进行唤醒，除了Wakeup模块外其余模块均与唤醒流程相似，具体流程如图2所示。

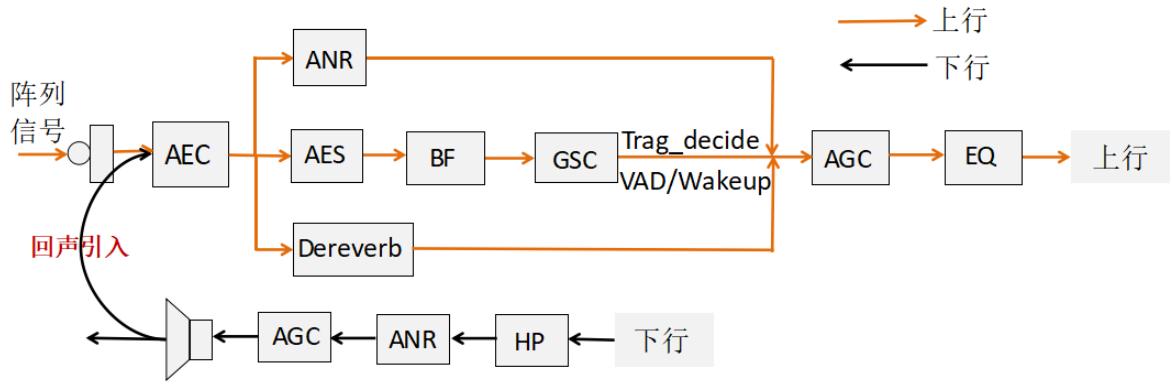


图 2 通话流程系统框图

1. 阵列采集阵列信息后，先通过AEC模块消除线性回声；
2. 接入ANR/AINR、Dereverb、AES模块分别进行噪声、混响估计以及残留回声消除；
3. 经过波束输出增强后的语音，同时支持声源定位以及定向拾音；
4. 最后送到AGC、EQ模块进行语音增益，送到远端进行通话。

## 2.3 音频算法运行效果展示

### 2.3.1 AEC模块效果

多麦克风通话算法提供两种不同的语音通路，以下为2中不同通路产生的效果，注该测试音频采用WV2215音箱，麦克风采用MSM261D4030H1CPM，测试环境为瑞芯微演播室，音箱的频响和失真曲线如下所示：

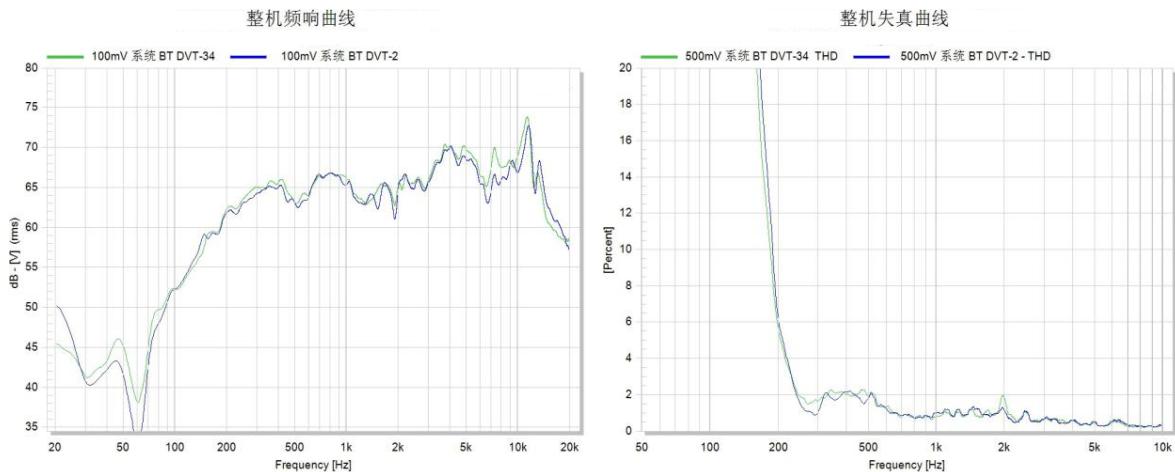


图 3 WV2215频响和失真曲线

采用唤醒加噪声方式进行算法效果统计，左通道为原始输入，右通道为算法处理后输出。其中，左图均为频域图，右图为纯噪声区间内的时域统计结果。

- 线性回声消除效果

以下为线性回声处理结果，需要注意的是该音频并非在专业声学环境测量，存在环境噪声，结果仅做参考。

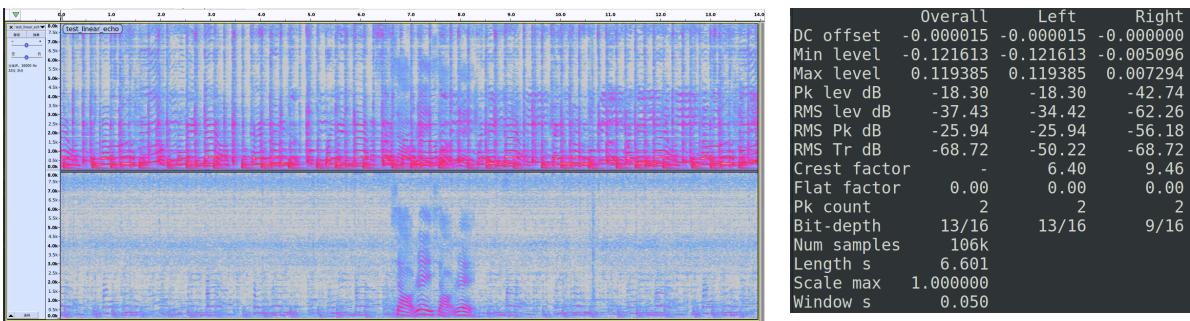


图 4 线性回声消除系统框图

该场景下，回声消除平均在30dB左右。

- 非线性回声消除效果

以下为非线性回声处理结果，需要注意的该音频并非在专业声学环境测量，结果仅做参考。

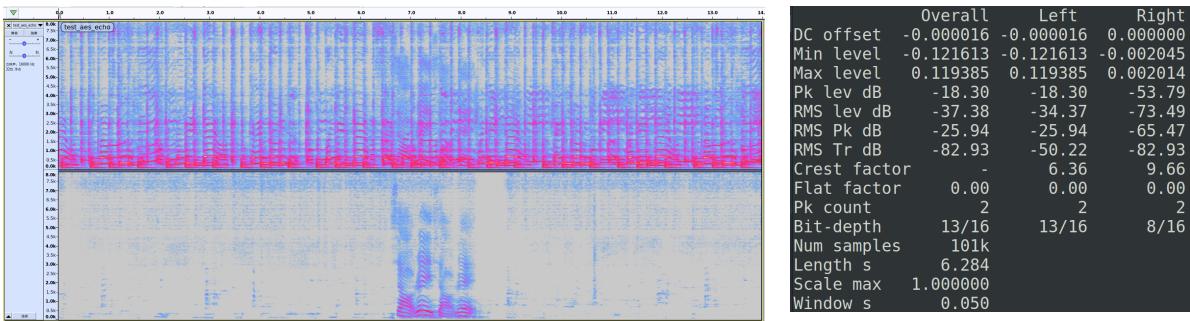


图 5 非线性回声消除系统框图

该场景下，回声消除平均在40dB左右。

### 2.3.2 BF模块效果

以下为干扰测试，测试场地为瑞芯微演播室，测试声源1m处分贝值为70dB，测试声源距离麦阵5m，干扰声源1m处分贝值为64dB，干扰声源距离麦阵1m：

- 唤醒流程效果

以下为通用环境下唤醒测试，结果仅做参考：

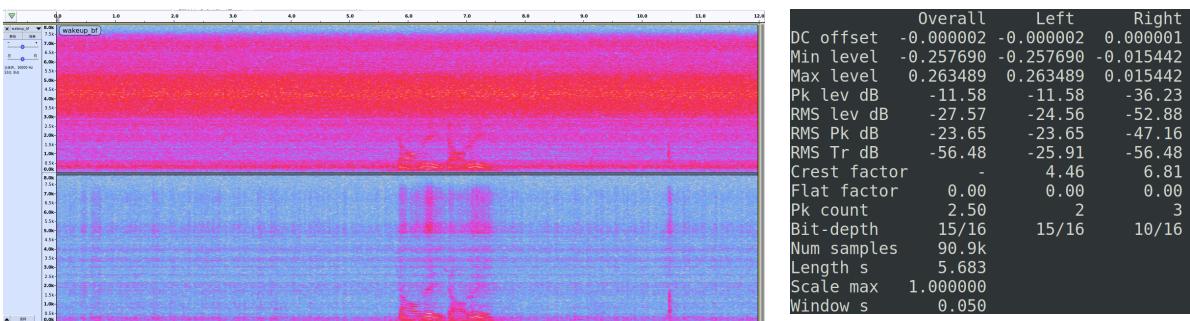


图 6 唤醒流程效果图

该场景下，背景噪声消除平均在28dB左右。

注：语音唤醒只用了固定波束以免造成语音的损伤。

- 通话流程效果

以下为通用环境下通话测试，结果仅做参考：

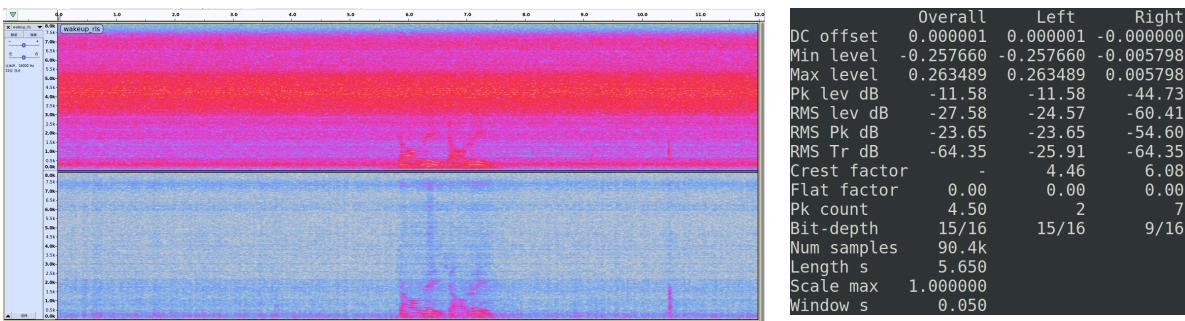


图 7 通话流程效果图

该场景下，背景噪声消除平均在36dB左右。

### 3. 音频算法接口调用说明

该库的调用过程如下：

#### 1. 设置参数：

该步骤用于对麦克风算法开放的音频参数进行设置，其中，AEC相关参数的设置需要根据产品音频腔体、器件特点以及使用需求等进行调整，使用时需要根据每个产品的特点设置参数，初步集成时可使用默认参数，然后根据效果调整参数。参数设置需要通过rkaudio\_preprocess.h文件调整，将该文件路径名作为初始化的输入参数。

#### 2. 初始化：

该步骤用于通话开启后，逐帧处理声音信号前。初始化接口函数名为rkaudio\_preprocess\_init，具体调用如下所示：

```
st_ptr = rkaudio_preprocess_init(mSampleRate, mBitPerSample, num_src_channel,
num_ref_channel, &param);
```

其中，mSampleRate采样率支持8kHz、16kHz、24kHz、32kHz、44.1kHz、48kHz，mBitPerSample音频位深为16bit，num\_src\_channel为mic通道数，num\_ref\_channel为回采通道数，param为输入参数，该参数通过rkaudio\_preprocess.h配置将在后面进行详细说明。

初始化完成算法内部参数初始化和相关内存申请，若初始化成功则返回st\_ptr结构体，如果初始化失败，结构体将为NULL。

#### 3. 帧处理：

该步骤即是实时处理通话语音信号，其单位为帧，当固定每帧16ms、采样率为16k采样时，每帧包含256个样点。帧处理包括接口函数为rkaudio\_preprocess\_short，具体调用如下所示：

```
out_size = rkaudio_preprocess_short(st_ptr, (short*)in, out, in_size / 2,
&is_wakeup);
```

其中，st\_ptr为初始化后生成的结构体，in为输入音频，单位为byte，out为输出音频，单位为short，in\_size为一帧读入的byte数，即`256 * num_channel * 2`。

#### 4. 销毁释放：

该步骤是在通话结束后，用于释放算法内存变量，接口函数为rkaudio\_preprocess\_destory(st\_ptr)和rkaudio\_param\_deinit(&param)。

## 4. 音频算法参数调试说明

多麦克风阵列算法通过rkaudio\_preprocess.h文件进行参数配置。

### 4.1 全局模块参数设置

参数设置主要分为**AEC模块**和**BF模块**，其中AEC模块通过rkaudio\_aec\_param\_init()函数配置，BF模块通过rkaudio\_preprocess\_param\_init()函数配置。

下面是对输入音频各种通道数的基础设置。

```
#define NUM_CHANNEL          4 /* 设置总通道数, 即mic通道以及回采通道之和 */
#define NUM_REF_CHANNEL       2 /* 设置回采通道数, 该处设置2回采通道 */
#define NUM_DROP_CHANNEL      0 /* 设置不处理的回采通道数, 置0没有丢弃, 置为N(N>0)时丢弃回
采通路的最后N路 */
#define REF_POSITION          0 /* 回采位置, 0为回采通道位于mic数据最前面, 1则在最后面 */
static short Array[NUM_CHANNEL] = { 0, 1, 2, 3}; /* 设置通道顺序, 用于不规则麦克风调整
*/
```

### 4.2 AEC模块参数设置

AEC模块通过rkaudio\_aec\_param\_init()函数配置，主要参数如下：

其中各子模块开关主要通过**model\_aec\_en**控制，主要包括：*EN\_DELAY*、*EN\_ARRAY\_RESET*。具体可看下面代码注释。

```
param->pos           /* 回采通道所在位置, 通过宏REF_POSITION赋值 */
param->drop_ref_channel /* 不处理的回采通道数, 通过宏NUM_DROP_CHANNEL赋值 */
param->model_aec_en   /* 置0表示不开除了线性AEC以外的子模块, 置EN_DELAY开启延迟估计,
软回采必须开启, 置EN_ARRAY_RESET开启输入音频通道顺序重排功能 */
param->delay_len      /* 延迟估计不开时为默认硬回采延迟点, 延迟估计打开时为起始延迟估计点
*/
param->look_ahead     /* 缓存麦克风端数据长度, 除非已知采集麦克风信号晚于回采信号可配置,
其余情况默认置0 */
param->Array_list      /* 配置音频通道顺序, 通过数组Array赋值 */
param->filter_len      /* 配置线性AEC滤波器长度, 设置越长对于回声估计越准确, 但收敛速度相
应越慢且算力越大。一般可配置为2/4/6/8 */
param->delay_para = rkaudio_delay_param_init() /* 延时估计子模块参数设置 */
```

### 4.3 延时估计子模块参数设置

delay模块通过rkaudio\_delay\_param\_init()函数配置，主要参数如下：

```

param->MaxFrame      /* 延时估计最长估计帧数, 一般场景建议设为16 */
param->LeastDelay    /* 延时估计最短估计帧数, MaxFrame没有特别大时建议设为0 */
param->JumpFrame     /* 初始跳过帧数, 一般设为12 */
param->DelayOffset   /* offset帧数, 用于防止延时估计过大导致麦克风信号晚于回采信号, 一般设为1 */
param->MicAmpThr     /* mic端最小能量阈值, 低于该值不更新延时估计结果, 一般设为50 */
param->RefAmpThr     /* ref端最小能量阈值, 低于该值不更新延时估计结果, 一般设为50 */
param->StartFreq      /* 延时估计起始频段的频率, 一般选择语音集中频段, 根据实际收音效果调整, 一般设为500 */
param->EndFreq        /* 延时估计终止频段的频率, 一般选择语音集中频段, 根据实际收音效果调整, 一般设为4000 */
param->SmoothFactor   /* 平滑系数, 一般取值为[0.95, 1), 越高延时估计结果越稳定, 越低延时估计结果更新越快, 一般设为0.97/0.99 */

```

## 4.4 BF模块参数设置

BF模块子模块主要包括: *EN\_Fastaec*、*EN\_Wakeup*、*EN\_Dereverberation*、*EN\_AES*、*EN\_Agc*、*EN\_AnR*、*EN\_STDT*、*EN\_CNG*、*EN\_EQ*、*EN\_HOWLING*、*EN\_DOA*、*EN\_WIND*、*EN\_AINR*等。具体可看下面代码注释。

```

typedef enum RKPreprocessEnable_
{
    EN_Fastaec = 1 << 0,           /* 线性残留回声消除模块, 该功能用于唤醒通路进一步消除回声 */
    EN_Wakeup = 1 << 1,            /* 唤醒模块 */
    EN_Dereverberation = 1 << 2,   /* 去混响模块 */
    EN_Nlp = 1 << 3,              /* 非线性残留回声消除模块, 该功能用于通话通路进一步消除回声 (改模块已弃用) */
    EN_AES = 1 << 4,               /* 非线性残留回声消除模块 */
    EN_Agc = 1 << 5,               /* 自动增益控制模块 */
    EN_AnR = 1 << 6,               /* 噪声抑制模块 */
    EN_GSC = 1 << 7,               /* 波束后滤波模块, 用于进一步消除噪声, 视实际效果决定是否打开 */
    GSC_Method = 1 << 8,             /* 关闭则GSC采用LMS方法, 打开GSC采用RLS方法, RLS算力高于LMS算法 */
    EN_Fix = 1 << 9,                /* 启用固定方向波束 */
    EN_STDT = 1 << 10,              /* 单双讲检测模块 */
    EN_CNG = 1 << 11,               /* 舒适度噪声模块 */
    EN_EQ = 1 << 12,                /* EQ调节模块 */
    EN_CHN_SELECT = 1 << 13,         /* 用于定制化扩展MIC需求场景 */
    EN_HOWLING = 1 << 14,             /* 去啸叫模块, 视实际场景需求决定是否打开 */
    EN_DOA = 1 << 15,                /* 声源定位模块, 用于估计声源方向, 视实际场景需求决定是否打开 */
    EN_WIND = 1 << 16,                /* 去风噪模块, 视实际场景需求决定是否打开 */
    EN_AINR = 1 << 17,                /* AI降噪模块, 若开启该模块则ANR模块将强制关闭 */
} RKPreprocessEnable;

```

通过rkaudio\_preprocess\_param\_init()函数进行BF模块总配置, 主要参数如下:

```

param->model_bf_en /* 子模块开关, 置0表示bf模块在非多麦情况下没有开启任何模块, 当在多麦情况下会默认跑波束模块, 其余子模块可根据需要自行开关 */
param->Targ = 0; /* 当EN_Fix打开时, 设置拾音方向 */
param->ref_pos = REF_POSITION; /* 回采通道所在位置, 通过宏REF_POSITION赋值 */

```

```

param->num_ref_channel = NUM_REF_CHANNEL; /* 回采总通道数, 通过宏NUM_REF_CHANNEL赋值 */
param->drop_ref_channel = NUM_DROP_CHANNEL; /* 不处理的回采通道数, 通过宏NUM_DROP_CHANNEL赋值 */
param->anr_para = rkaudio_anr_param_init_tx(); /* ANR模块参数配置 */
param->dereverb_para = rkaudio_dereverb_param_init(); /* 去混响模块参数配置 */
param->aes_para = rkaudio_aes_param_init(); /* AES模块参数配置 */
param->dtd_para = rkaudio_dtd_param_init(); /* DTD模块参数配置 */
param->agc_para = rkaudio_agc_param_init(); /* AGC模块参数配置 */
param->cng_para = rkaudio_cng_param_init(); /* CNG模块参数配置 */
param->eq_para = rkaudio_eq_param_init(); /* EQ模块参数配置 */
param->howl_para = rkaudio_howl_param_init_tx(); /* 噤叫模块参数配置 */
param->doa_para = rkaudio_doa_param_init(); /* DOA模块参数配置 */

```

## 4.5 ANR模块参数设置

降噪模块用于语音降噪，消除环境噪声，通过rkaudio\_anr\_param\_init()函数配置：

```

param->noiseFactor      /* 噪声估计水平, 一般取值范围在[0.5f, 1.5f], 越大噪声消除越干净 */
param->swU               /* 噪声估计帧长时间片, 一般取值范围[2, 20], 设置越大对平稳噪声估计越准确, 对突变噪声收敛越慢; 反之对突变噪声收敛越快但容易造成语音过消 */
param->PsiMin            /* 语音存在概率判断阈值, 取值范围[0.0f, 1.0f], , 设置越大, 噪音消除效果越强 */
param->PsiMax            /* 语音存在概率判断阈值, 取值范围[0.0f, 1.0f], 设置越大, 噪音消除效果越强 */
param->fGmin              /* 噪声消除最小值, 取值范围[0.0f, 1.0f], 设置越小, 噪音消除效果越强 */
*/

param->Sup_Freq1          /* 恒频噪声抑制点1, 取值范围[0, 1/2*sample rate], 始终抑制设置点噪声 */
param->Sup_Freq2          /* 恒频噪声抑制点2, 取值范围[0, 1/2*sample rate], 始终抑制设置点噪声 */
param->Sup_Energy1         /* 恒频噪声抑制点1对应的能量阈值, 一般取值100000, 当频点能量超出该值时会对恒频噪声的估计进行更新 */
param->Sup_Energy2         /* 恒频噪声抑制点1对应的能量阈值, 一般取值100000, 当频点能量超出该值时会对恒频噪声的估计进行更新 */

param->InterV             /* 取帧间隔, 一般取值范围[2, 20], 设置越大估计的噪声越平稳, 类似swU功能 */
param->BiasMin             /* 噪声补偿, 一般取值范围[1.0f, 2.0f], 设置越大噪声消除效果越强 */
param->UpdateFrm           /* 快速收敛长度, 一般设置为15, 在开头15帧快速收敛 */
param->NPreGammaThr        /* 纯噪声概率估计阈值, 一般取值范围(3.0f, 5.0f), 设置越大噪声消除效果越强 */
param->NPreZetaThr         /* 纯噪声概率估计阈值, 一般取值范围(1.0f, 3.0f), 设置越大噪声消除效果越强 */
param->SabsGammaThr0       /* 语音不存在概率估计阈值, 一般取值范围(0.5f, 1.5f), 设置越大噪声消除效果越强 */
param->SabsGammaThr1       /* 语音不存在概率估计阈值, 一般取值范围(2.0f, 4.0f), 设置越大噪声消除效果越强 */
param->InfSmooth            /* 输入信号平滑系数, 取值范围(0.0f, 1.0f) */
param->ProbSmooth           /* 语音存在概率平滑系数, 取值范围(0.0f, 1.0f) */
param->CompCoeff             /* 噪声补偿, 一般取值范围[1.0f, 2.0f], 对于低于平均噪声估计的频段进行补偿, 设置越大噪声消除效果越强 */
param->PrioriMin            /* 先验信噪比最小值, 取值范围(0.0f, 0.5f), 设置越大噪声消除效果越弱 */

```

```

param->PostMax          /* 后验信噪比最大值, 一般设为40 */
param->PrioriRatio      /* 先验信噪比平滑系数, 取值范围(0.0f, 1.0f), 设置越大噪声消除效果越
强 */
param->PrioriRatioLow   /* 先验信噪比平滑系数, 取值范围(0.0f, 1.0f), 设置越大噪声消除效果越
强 */
param->SplitBand        /* 分频段index, 取值范围[0, 32], 对于低于该index的先验信噪比用
PrioriRatioLow进行平滑, 反之用PrioriRatio */
param->PrioriSmooth     /* 先验信噪比平滑系数, 取值范围(0.0f, 1.0f), 设置越大噪声消除效果越
强 */
param->TranMode         /* 冲击噪声开关, 置0表示关闭, 置1表示使用检测消除方法, 置2表示使用估
计消除方法 */

```

## 4.6 去混响模块参数设置

去混响模块用于混响消除, 该模块通过rkaudio\_dereverb\_param\_init()配置:

```

param->rlsLg           /* RLS滤波器阶数, 用于唤醒模块去混响, 一定范围内, 阶数越大, 去混响效果越
强, 算力消耗越大, 建议取值: 4 */
param->curveLg         /* 分布曲线阶数, 用于通话模块去混响, 阶数越大, 去混响效果越强, 越容易过消,
建议取值: 10 */
param->delay            /* RLS滤波器延时, 决定早期混响保留程度, 建议取值: 2 */
param->forgetting       /* RLS滤波器遗忘因子, 因子越小, rls去混响能力越强, 越容易过消, 建议范
围:0.98-0.999 */
param->T60              /* 混响时间估计值 (单位: s) , 越大, 去混响能力越强, 但是越容易过消除, 低混
响场景建议取值0.68, 中高混响场景建议取值1.5 */
param->coCoeff          /* 互相干性调整系数, 防止过消除, 越小能力越强, 低混响场景建议取值2, 中高混
响场景建议取值1 */

```

## 4.7 AES模块参数设置

AES模块用于非线性处理过程, 该模块通过rkaudio\_aes\_param\_init()配置。

```

param->Beta_Up          /* 上升速度, 该值越大则非线性抑制越强, 建议0.001-0.01之间 */
param->Beta_Down         /* 下降速度, 该值越大则非线性抑制越弱, 建议0.001-0.01之间 */
param->Beta_Up_Low       /* 低频段上升速度, 特性以及取值同Beta_Up */
param->Beta_Down_Low    /* 低频段下降速度, 特性以及取值同Beta_Down */
param->low_freq          /* 低频分段 */
param->high_freq         /* 高频分段 */
param->THD_Flag          /* 置1开启谐波失真抑制, 置0则关闭 */
param->HARD_Flag         /* 置1开启Hard模式, 对于回声消除抑制更强, 但也有可能造成语音过消 */
int i, j;

/**
 * 3列分别对应低频分段、中频分段、高频分段 (由上面的low_freq以及high_freq设置
 * 第一行表示各频段残留回声的比例最小值, 一般建议取值范围[1.0f, 5.0f], 设置越大, 回声消除效果
越强
 * 第二行表示各频段消除残留回声的比例, 一般建议取值范围[1.0f, 3.0f], 设置越大, 回声消除效果越
强
 */
param->LimitRatio[2][3]

/**

```

```

    * 谐波抑制频段，最多设置4个频段，不需要时该行可以全置0
    * 每一行表示一个频段，第一列表示频段起始点，第二列表示频段终止点。取值范围均为[0, 1/2*sample
rate]
 */
param->ThdSplitFreq[4][2]

/**
 * 谐波抑制程度，对应上表的四个频段，配置的对应消除比例，当上面有全0行时，该表对应行的设置也将无
效
 * 从第一列到最后一列分别表示频段内有可能成为二次到十一次谐波时消除相应基频的比例，一般建议取值范
围
 * [0.001f, 1.0f]，设置越大，消除效果越强
 */
param->ThdSupDegree[4][10]

/**
 * 配置为Hard模式的频段以及计算平均增益的频段，第一列表示起始，第二列表示终止，取值范围均为
 * [0, 1/2*sample rate]，配置为Hard模式的频段最多设置4个，最后一行为计算平均增益的频段，
 * 该频段将在下一个参数HardThreshold中进行使用
 */
param->HardSplitFreq[5][2]

/**
 * Hard模式开启阈值，4个阈值分别对应上个参数配置的4个频段
 * 使用平均增益频段和该参数配置的阈值进行对比，当平均增益低于阈值时，对应的频段会开启Hard模式
 */
param->HardThreshold[4]

```

## 4.8 AGC模块参数设置

AGC模块用于增强语音，通过rkaudio\_agc\_param\_init()函数配置：

```

param->attack_time = 400.0; /* 建议范围：20~600ms。触发时间，即AGC增益上升所需要的时间
*/
param->release_time = 400.0; /* 建议范围：20~600ms。施放时间，即AGC增益下降所需要的时间
*/
param->max_gain = 30.0; /* 建议范围：0~50dB。最大增益，同时也是线性段增益 */
param->max_peak = -3.0; /* 取值范围：0~-87dB。经AGC处理后，输出语音的最大能量 */
param->fRth0 = -65.0f; /* 取值范围：0~-87dB。噪声门阈值，低于该值的语音不被增益 */
param->fRth1 = -60.0f; /* 取值范围：0~-87dB。扩张段结束能量dB阈值，高于该值的语音
以max_gain增益 */
param->fRth2 = -35.0f; /* 取值范围：0~-87dB。压缩段起始能量dB阈值，高于该值的语音
增益逐渐下降至0dB */

```

注：压缩段必须满足以下条件：**fRth2 + max\_gain** 必须小于**0dB**且小于**max\_peak**，否则会造成截幅。

## 4.9 舒适噪声参数设置

舒适噪声CNG参数设置用于在通话过程中添加舒适白噪声，仅在录音处理中存在，且该模块通常关闭。

```
param->fSmoothAlpha /* 取值范围: 0.0f~1.0f, 施加舒适噪声平滑度, 默认0.92 */
param->fSpeechGain /* 取值范围: 0.0f~1.0f, 施加舒适噪声语音纹理模拟程度, 默认0.3 */
param->fGain        /* 建议范围: 0~50dB, 施加舒适噪声幅度比例, 默认2 */
param->fMpy         /* 建议范围: 0~50dB, 白噪随机数生成幅度, 默认5 */
```

## 4.10 DTD模块参数设置

DTD参数设置用于判断语音处于单讲还是双讲态。

```
param->ksiThd_high = 0.70f /* 单双讲判决阈值, ceil阈值, 高于该阈值判断为单讲 */
param->ksiThd_low = 0.50f; /* floor阈值, 低于该阈值判断为单讲 */
```

注：该值为起始阈值，目前算法采用自适应更新。

## 4.11 推荐参数示例

用于唤醒推荐参数如下：

```
param->model_bf_en = EN_Wakeup | EN_Fastaec | EN_STDT | EN_Agc;
```

用于语音通话推荐参数如下：

```
param->model_bf_en = EN_Fastaec | EN_STDT | EN_AES | EN_Anr | EN_Dereverberation
| EN_Agc;
```