XR872 Internal Codec User Guide

Revision 1.0

Oct 10, 2019

Declaration

THIS DOCUMENTATION IS THE ORIGINAL WORK AND COPYRIGHTED PROPERTY OF XRADIO TECHNOLOGY ("XRADIO"). REPRODUCTION IN WHOLE OR IN PART MUST OBTAIN THE WRITTEN APPROVAL OF XRADIO AND GIVE CLEAR ACKNOWLEDGEMENT TO THE COPYRIGHT OWNER.

THE PURCHASED PRODUCTS, SERVICES AND FEATURES ARE STIPULATED BY THE CONTRACT MADE BETWEEN XRADIO AND THE CUSTOMER. PLEASE READ THE TERMS AND CONDITIONS OF THE CONTRACT AND RELEVANT INSTRUCTIONS CAREFULLY BEFORE USING, AND FOLLOW THE INSTRUCTIONS IN THIS DOCUMENTATION STRICTLY. XRADIO ASSUMES NO RESPONSIBILITY FOR THE CONSEQUENCES OF IMPROPER USE (INCLUDING BUT NOT LIMITED TO OVERVOLTAGE, OVERCLOCK, OR EXCESSIVE TEMPERATURE).

THE INFORMATION FURNISHED BY XRADIO IS PROVIDED JUST AS A REFERENCE OR TYPICAL APPLICATIONS, ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS DOCUMENT DO NOT CONSTITUTE A WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. XRADIO RESERVES THE RIGHT TO MAKE CHANGES IN CIRCUIT DESIGN AND/OR SPECIFICATIONS AT ANY TIME WITHOUT NOTICE.

NOR FOR ANY INFRINGEMENTS OF PATENTS OR OTHER RIGHTS OF THE THIRD PARTIES WHICH MAY RESULT FROM ITS USE. NO LICENSE IS GRANTED BY IMPLICATION OR OTHERWISE UNDER ANY PATENT OR PATENT RIGHTS OF XRADIO.THIRD PARTY LICENCES MAY BE REQUIRED TO IMPLEMENT THE SOLUTION/PRODUCT. CUSTOMERS SHALL BE SOLELY RESPONSIBLE TO OBTAIN ALL APPROPRIATELY REQUIRED THIRD PARTY LICENCES. XRADIO SHALL NOT BE LIABLE FOR ANY LICENCE FEE OR ROYALTY DUE IN RESPECT OF ANY REQUIRED THIRD PARTY LICENCE. XRADIO SHALL HAVE NO WARRANTY, INDEMNITY OR OTHER OBLIGATIONS WITH RESPECT TO MATTERS COVERED UNDER ANY REQUIRED THIRD PARTY LICENCE.

Revision History

Version	Date	Summary of Changes
1.0	2019-10-10	Initial Version

Table 1- 1 Revision History

Contents

Figures

1 宏定义说明

注：凡是使能类型的宏定义，默认 0 为不使能，非 0 为使能（建议非 0 时定义为 1）

1.1 调试宏定义说明

1.1.1 XRADIO_CODEC_DBG_EN

驱动 debug 调试打印使能，使能后驱动中所有调用到 XRADIO_CODEC_DBG 宏定义的调试信息在 codec 使用过程中会被打印出来，方便调试，如下图 1-1 所示为播放时的调试打印：

```
$
$ audio play 16000 2 /silence.pcm
<ACK> 200 OK
[WRN] switch to high speed error, use DS: 25 MHz
[FS INF] mmc init
[FS INF] mount success
[cmd] CMD:drv audio play (samplerate)16000 (channel)2 (file)/silence.pcm
[XRADIO_INTERNAL_CODEC] --->xradio_dai_set_sysclk
[XRADIO_INTERNAL_CODEC] SYSCLK source select AUDIO_PLL
[XRADIO_INTERNAL_CODEC] Xradio Codec PLL freq_in match:40000000, freq_out:24576000
[XRADIO_INTERNAL_CODEC] --->xradio_dai_set_fmt
[XRADIO_INTERNAL_CODEC] --->xradio_dai_hw_params
[XRADIO_INTERNAL_CODEC] Sample rate-[16000], channel numbers-[2], sample resolution-[16]
[XRADIO_INTERNAL_CODEC] --->xradio_dai_set_volume
[XRADIO_INTERNAL_CODEC] LINEOUT set volume Level-[8]
[XRADIO_INTERNAL_CODEC] Route(play): lineout Enable
[XRADIO_INTERNAL_CODEC] --->xradio_codec_open

Xradio Codec Reg:
Reg[0x00] :0x80000003;  Reg[0x04] :0x60004000;  Reg[0x08] :0x0080800c;  Reg[0x0c] :0x00000000;
Reg[0x10] :0x00000000;  Reg[0x14] :0x00000000;  Reg[0x18] :0x00000400;  Reg[0x1c] :0x00800000;
Reg[0x20] :0x00000000;  Reg[0x24] :0x00000000;  Reg[0x28] :0x00000000;  Reg[0x2c] :0x00fffac1;
Reg[0x30] :0x01000000;  Reg[0x34] :0x00000000;  Reg[0x38] :0x003c202a;  Reg[0x3c] :0x00000009;
Reg[0x40] :0x44554055;  Reg[0x44] :0x00010208;

[cmd] Play on.
[XRADIO_INTERNAL_CODEC] Tx: play start...
$ audio end
[cmd] audio task end
<ACK> 200 OK
[XRADIO_INTERNAL_CODEC] --->xradio_codec_close
[XRADIO_INTERNAL_CODEC] --->xradio_dai_hw_free
[XRADIO_INTERNAL_CODEC] xradio_codec_reset, reset all register to their default value
[cmd] Play end.
$
```

图 1-1 XR872 Internal Codec 播放调试信息打印

如上图所示，其中带有“[XRADIO_INTERNAL_CODEC]”前缀的打印为 XR872 Internal Codec 驱动的调试信息打

印。建议调试过程中使能此宏，调试完成后可关闭。

配置建议：此宏可根据调试需要来打开或者关闭；默认关闭；

1.1.2 XRADIO_CODEC_ERR_EN

驱动 error 调试打印使能，使能后驱动中所有调用到 XRADIO_CODEC_ERR 和 XRADIO_CODEC_IT_ERR 宏定义的调试信息在 codec 使用过程中会被打印出来，方便驱动出错时调试查找问题。

另外，驱动中还有一个 XRADIO_CODEC_ALWAYS 宏定义，使用此宏定义的调试信息始终会打印出来，不受使能控制，如配置音量、路径时的打印。

配置建议：此宏可根据调试需要来打开或者关闭；默认打开；

1.2 功能配置宏定义说明

1.2.1 XRADIO_DEFAULT_PLAY_VOLUME

默认播放音量，声卡注册时会初始化播放音量为 VOLUME_INVALID，此时如果应用层未设置过播放音量，则播放时会使用此处定义的默认播放音量，应用层设置过播放音量后则会使用应用设置的播放音量。

配置建议：采用默认值即可；默认值为 VOLUME_LEVEL31；

1.2.2 XRADIO_DEFAULT_RECORD_GAIN

默认录音增益，声卡注册时会初始化录音音量为 VOLUME_INVALID，此时如果应用层未设置过录音音量，则录音时会使用此处定义的默认录音音量，应用层设置过录音音量后则会使用应用设置的录音音量。

配置建议：采用默认值即可；默认值为 VOLUME_GAIN_0dB；

1.2.3 XRADIO_PLAY_UNDERRUN_THRESHOLD

播放 underrun 阈值配置，DMA underrun 达到此阈值后会停止播放。

配置建议：采用默认值即可；默认值为 3；

1.2.4 XRADIO_RECORD_OVERRUN_THRESHOLD

录音 overrun 阈值配置，DMA overrun 达到此阈值后会停止录音。

配置建议：采用默认值即可，默认值为 3；

1.2.5 XRADIO_CODEC_MALLOC

Xradio Internal Codec 驱动中申请内存宏定义接口。

配置建议：采用默认值即可；默认值为 HAL_Malloc；

1.2.6 XRADIO_CODEC_FREE

Xradio Internal Codec 驱动中释放内存宏定义接口。

配置建议：采用默认值即可；默认值为 HAL_Free；

1.2.7 XRADIO_CODEC_MEMCPY

Xradio Internal Codec 驱动中 memcpy 宏定义接口。

配置建议：采用默认值即可；默认值为 HAL_Memcpy；

1.2.8 XRADIO_CODEC_MEMSET

Xradio Internal Codec 驱动中 memset 宏定义接口。

配置建议：采用默认值即可；默认值为 HAL_Memset；

2 基本接口

2.1 Xradio_Codec_Priv 结构体

struct Xradio_Codec_Priv 结构体为 Xradio Internal Codec 驱动中使用的私有数据结构，其对应定义的全局指针变量为 xradio_codec_priv，在 Codec 注册时会为其申请内存，具体定义如下代码段所示：

```
//Xradio codec priv struct
struct Xradio_Codec_Priv {
    //codec status contrl
    bool isCodecInit;
    bool isTxInit;
    bool isRxInit;
    volatile bool txRunning;
    volatile bool rxRunning;

    //buffer control
    uint8_t *txBuf;
    uint8_t *rxBuf;
    uint8_t *writePointer;
    uint8_t *readPointer;
    uint32_t txBufSize;
    uint32_t rxBufSize;

    //DMA control
    uint8_t *txDmaPointer;
    uint8_t *rxDmaPointer;
    DMA_Channel   txDMAChan;
    DMA_Channel   rxDMAChan;
    DMA_DataWidth tx_data_width;
    DMA_DataWidth rx_data_width;
    volatile uint8_t txHalfCallCount;
    volatile uint8_t rxHalfCallCount;
    volatile uint8_t txEndCallCount;
    volatile uint8_t rxEndCallCount;

    //Semaphore control
    HAL_Semaphore txReady;
    HAL_Semaphore rxReady;
    bool isTxSemaphore;
    bool isRxSemaphore;
} ;

static struct Xradio_Codec_Priv *xradio_codec_priv;
```

2.1.1 Codec 状态控制变量

- **isCodecInit**：标志 Codec 是否已经初始化，在 xradio_internal_codec_init 和 xradio_internal_codec_deinit 中使用；

- **isTxInit**：标志 Codec 是否已经播放 open；在 xradio_codec_open 和 xradio_codec_close 中配置；
- **isRxInit**：标志 Codec 是否已经录音 open，在 xradio_codec_open 和 xradio_codec_close 中配置；
- **txRunning**：标志 Codec 是否已经 trigger DMA 播放，在 xradio_codec_dma_trigger 中配置；
- **rxRunning**：标志 Codec 是否已经 trigger DMA 录音，在 xradio_codec_dma_trigger 中配置；

2.1.2 buffer 控制变量

- *txBuf：播放 buffer 指针，在 xradio_codec_open 时申请；
- *rxBuf：录音 buffer 指针，在 xradio_codec_open 时申请；
- *writePointer：下一次播放写入 buffer 的指针；
- *readPointer：下一次录音读取 buffer 的指针（此变量目前暂未使用）；
- txBufSize：以 byte 为单位的播放 buffer 大小；
- rxBufSize：以 byte 为单位的录音 buffer 大小；

2.1.3 DMA 控制变量

- *txDmaPointer：当前 DMA 播放搬运数据指针；
- *rxDmaPointer：当前 DMA 录音搬运数据指针；
- txDMAChan：DMA 播放通道；
- rxDMAChan：DMA 录音通道；
- tx_data_width：DMA 播放数据宽度；
- rx_data_width：DMA 录音数据宽度；
- TxHalfCallCount:DMA 播放半中断计数；
- rxHalfCallCount：DMA 录音半中断计数；
- txEndCallCount：DMA 播放全中断计数；
- rxEndCallCount：DMA 录音全中断计数；

2.1.4 信号量控制变量

- txReady：DMA 播放信号量；
- rxReady：DMA 录音信号量；
- isTxSemaphore：DMA 播放信号量是否正在被占用标志；

❖ isRxSemaphore：DMA 录音信号量是否正在被占用标志；

2.2 常量数组

2.2.1 xradio_pll_params[]

Xradio Internal Codec AUDIO_PLL 配置参数数组，数组元素类型为 struct pll_param 结构体常量，如下代码段所示：

```
struct pll_param {
    uint32_t hosc_freq;
    Audio_Clk_Freq pll_freq_out;
    uint32_t pllParam;
    uint32_t pllPatParam;
};
static const struct pll_param xradio_pll_params[] = {
    {HOSC_CLOCK_26M, AUDIO_CLK_24M, PRCM_AUD_PLL24M_PARAM_HOSC26M, PRCM_AUD_PLL24M_PAT_PARAM_HOSC26M},
    {HOSC_CLOCK_26M, AUDIO_CLK_22M, PRCM_AUD_PLL22M_PARAM_HOSC26M, PRCM_AUD_PLL22M_PAT_PARAM_HOSC26M},
    {HOSC_CLOCK_24M, AUDIO_CLK_24M, PRCM_AUD_PLL24M_PARAM_HOSC24M, PRCM_AUD_PLL24M_PAT_PARAM_HOSC24M},
    {HOSC_CLOCK_24M, AUDIO_CLK_22M, PRCM_AUD_PLL22M_PARAM_HOSC24M, PRCM_AUD_PLL22M_PAT_PARAM_HOSC24M},
    {HOSC_CLOCK_40M, AUDIO_CLK_24M, PRCM_AUD_PLL24M_PARAM_HOSC40M, PRCM_AUD_PLL24M_PAT_PARAM_HOSC40M},
    {HOSC_CLOCK_40M, AUDIO_CLK_22M, PRCM_AUD_PLL22M_PARAM_HOSC40M, PRCM_AUD_PLL22M_PAT_PARAM_HOSC40M},
    {HOSC_CLOCK_52M, AUDIO_CLK_24M, PRCM_AUD_PLL24M_PARAM_HOSC52M, PRCM_AUD_PLL24M_PAT_PARAM_HOSC52M},
    {HOSC_CLOCK_52M, AUDIO_CLK_22M, PRCM_AUD_PLL22M_PARAM_HOSC52M, PRCM_AUD_PLL22M_PAT_PARAM_HOSC52M},
};
```

struct pll_param 结构体各变量定义如下：

- hosc_freq：PLL 外部晶振输入时钟源频率；
- pll_freq_out：PLL 输出频率，此处为 24.576M 或者 22.5792M；
- pllParam：PLL 配置参数；
- pllPatParam：PLL Pat 配置参数（目前此参数暂时不用）；

2.2.2 xradio_sample_rate[]

Xradio Internal Codec 不同采样率与具体的寄存器配置值映射关系数组，数组元素为 struct real_val_to_reg_val 结构体常量；如下代码段所示：

```
struct real_val_to_reg_val {
    uint32_t real_val;
    uint32_t reg_val;
};

static const struct real_val_to_reg_val xradio_sample_rate[] = {
    {48000, 0},
    {32000, 1},
    {24000, 2},
    {16000, 3},
    {12000, 4},
    {8000,  5},

    {44100, 0},
    {22050, 2},
    {11025, 4},
```

```
    {192000,6},
    {96000, 7},
};
```

struct real_val_to_reg_val 结构体各变量定义如下：

- ✓ real_val：实际值，此处为实际要配置的采样率；
- ✓ reg_val：寄存器值，此处为对应采样率的寄存器值；

2.2.3 xradio_amic_pga_gain[]

Xradio Internal Codec 不同 AMIC PGA 增益与寄存器配置值映射关系数组，数组元素为 struct real_val_to_reg_val 结构体常量；如下代码段所示：

```
static const struct real_val_to_reg_val xradio_amic_pga_gain[] = {
    {VOLUME_GAIN_0dB,  0},
    {VOLUME_GAIN_21dB, 1},
    {VOLUME_GAIN_24dB, 2},
    {VOLUME_GAIN_27dB, 3},
    {VOLUME_GAIN_30dB, 4},
    {VOLUME_GAIN_33dB, 5},
    {VOLUME_GAIN_36dB, 6},
    {VOLUME_GAIN_39dB, 7},
};
```

2.2.4 xradio_linein_pga_gain[]

Xradio Internal Codec 不同 LINEIN PGA 增益与寄存器配置值映射关系数组，数组元素为 struct real_val_to_reg_val 结构体常量；如下代码段所示：

```
static const struct real_val_to_reg_val xradio_linein_pga_gain[] = {
    {VOLUME_GAIN_MINUS_3dB,  0},
    {VOLUME_GAIN_0dB,  1},
    {VOLUME_GAIN_3dB,  2},
    {VOLUME_GAIN_6dB,  3},
    {VOLUME_GAIN_9dB,  4},
    {VOLUME_GAIN_12dB, 5},
    {VOLUME_GAIN_18dB, 6},
    {VOLUME_GAIN_24dB, 7},
};
```

2.2.5 xradio_lineout_gain[]

Xradio Internal Codec 不同 LINEOUT PGA 增益与寄存器配置值映射关系数组，数组元素为 struct real_val_to_reg_val 结构体常量；如下代码段所示：

```
static const struct real_val_to_reg_val xradio_lineout_gain[] = {
    {VOLUME_GAIN_0dB,        31},
    {VOLUME_GAIN_MINUS_3dB,  29},
    {VOLUME_GAIN_MINUS_6dB,  27},
```

```
    {VOLUME_GAIN_MINUS_9dB,  25},
    {VOLUME_GAIN_MINUS_12dB, 23},
    {VOLUME_GAIN_MINUS_15dB, 21},
    {VOLUME_GAIN_MINUS_18dB, 19},
    {VOLUME_GAIN_MINUS_21dB, 17},
    {VOLUME_GAIN_MINUS_24dB, 15},
    {VOLUME_GAIN_MINUS_27dB, 13},
    {VOLUME_GAIN_MINUS_30dB, 11},
    {VOLUME_GAIN_MINUS_33dB, 9},
    {VOLUME_GAIN_MINUS_36dB, 7},
    {VOLUME_GAIN_MINUS_39dB, 5},
    {VOLUME_GAIN_MINUS_42dB, 3},
};
```

2.3 基本读写寄存器接口

2.3.1 xradio_codec_reg_read

- **原型**：static int xradio_codec_reg_read(uint32_t reg)；
- **传参**：reg：寄存器地址；
- **功能**：读取 codec 寄存器值；
- **返回值**：读取到的寄存器值；

2.3.2 xradio_codec_reg_write

- **原型**：static int xradio_codec_reg_write(uint32_t reg, uint32_t val)；
- **传参**：reg：寄存器地址；val：要写入的寄存器值；
- **功能**：写入 codec 寄存器值；
- **返回值**：HAL_OK：写入成功；

2.3.3 xradio_codec_reg_update_bits

- **原型**：static int xradio_codec_reg_update_bits(uint32_t reg, uint32_t mask, uint32_t val)；
- **传参**：reg：寄存器地址；mask：需要更新的 bit mask；val：需要更新的 bit mask 对应值；
- **功能**：更新 codec 寄存器的某几 bit；
- **返回值**：HAL_OK：更新成功；

2.4 DMA 配置接口

2.4.1 xradio_codec_dma_trigger

- **原型：**static void xradio_codec_dma_trigger(Audio_Stream_Dir dir, bool enable)；
- **传参：**dir：Audio_Stream_Dir 类型音频流方向；enable：DMA 启动/停止 trigger 控制；
- **功能：**trigger DMA 以及配置 Codec 与 DMA 相关的控制；
- **返回值：**无；

2.4.2 xradio_codec_dma_threshold_check

- **原型：**static int xradio_codec_dma_threshold_check(Audio_Stream_Dir dir)；
- **传参：**dir：Audio_Stream_Dir 类型音频流方向；
- **功能：**检查 DMA xrun 是否达到阈值，达到则停止 DMA 播放录音，未达到则直接返回；
- **返回值：**HAL_OK：DMA xrun 未达到阈值；HAL_ERROR：DMA xrun 达到阈值，停止 DMA 播放/录音；

2.4.3 xradio_codec_dma_half_callback

- **原型：**static void xradio_codec_dma_half_callback(void *arg)；
- **传参：**arg：DMA 当前占用的信号量；
- **功能：**DMA 半中断回调函数，更新半中断计数、释放占用信号量、检查 DMA xrun 次数、更新 DMA 当前读写指针；
- **返回值：**无

2.4.4 xradio_codec_dma_end_callback

- **原型：**static void xradio_codec_dma_end_callback(void *arg)；
- **传参：**arg：DMA 当前占用的信号量；
- **功能：**DMA 全中断回调函数，更新全中断计数、释放占用信号量、检查 DMA xrun 次数、更新 DMA 当前读写指针；
- **返回值：**无；

2.4.5 xradio_codec_dma_init

- **原型**：static int xradio_codec_dma_init(Audio_Stream_Dir dir, DMA_Channel channel)；
- **传参**：dir：Audio_Stream_Dir 类型音频流方向；channel：DMA 通道；
- **功能**：初始化 DMA 配置参数；
- **返回值**：HAL_OK：初始化成功；HAL_ERROR：初始化失败；

2.5 基本硬件配置接口

2.5.1 xradio_codec_reset

- **原型**：static void xradio_codec_reset(void)；
- **传参**：无
- **功能**：Xradio Internal Codec 硬件复位，恢复到上电初始状态，包括 codec 相关的所有寄存器值；
- **返回值**：无；

2.5.2 xradio_codec_hw_common_init

- **原型**：static void xradio_codec_hw_common_init(Audio_Stream_Dir dir)；
- **传参**：dir：Audio_Stream_Dir 类型音频流方向；
- **功能**：Xradio Internal Codec 公共硬件初始化配置，如各模拟电压 enable 等；
- **返回值**：无；

2.5.3 xradio_codec_hw_common_deinit

- **原型**：static void xradio_codec_hw_common_deinit(Audio_Stream_Dir dir)；
- **传参**：dir：Audio_Stream_Dir 类型音频流方向；
- **功能**：Xradio Internal Codec 公共硬件反初始化配置，如各模拟电压 disable 等；
- **返回值**：无；

2.5.4 xradio_codec_set_audio_pll

- **原型**：static int xradio_codec_set_audio_pll(Audio_Clk_Freq pll_freq_out)；
- **传参**：pll_freq_out：Audio_Clk_Freq 类型 PLL 输出频率；

- **功能**：获取外部晶振输入频率，配置 PLL 倍频参数，使能 AUDIO PLL 输出；
- **返回值**：HAL_OK：配置成功；other：配置失败；

2.6 基本通路配置接口

2.6.1 xradio_codec_set_amic

- **原型**：static void xradio_codec_set_amic(bool enable)；
- **传参**：enable：AMIC enable/disable 配置参数；
- **功能**：enable/disable AMIC 通路；
- **返回值**：无；

2.6.2 xradio_codec_set_linein

- **原型**：static void xradio_codec_set_linein(bool enable)；
- **传参**：enable：LINEIN enable/disable 配置参数；
- **功能**：enable/disable LINEIN 通路；
- **返回值**：无；

2.6.3 xradio_codec_set_dmic

- **原型**：static void xradio_codec_set_dmic(bool enable)；
- **传参**：enable：DMIC enable/disable 配置参数；
- **功能**：enable/disable DMIC 通路；
- **返回值**：无；

2.6.4 xradio_codec_set_lineout

- **原型**：static void xradio_codec_set_lineout(bool enable)；
- **传参**：enable：LINEOUT enable/disable 配置参数；
- **功能**：enable/disable LINEOUT 通路；
- **返回值**：无；

2.7 Ioctl 接口

2.7.1 xradio_codec_ioctl_pcm_read

- **原型**：static int xradio_codec_ioctl_pcm_read(uint8_t *buf, uint32_t size)；
- **传参**：buf：读取 pcm 数据的 buf；size：读取 pcm 数据的大小；
- **功能**：读取 pcm 数据；
- **返回值**：大于 0：读取成功，返回读取到的数据量大小；小于等于 0：读取失败；

2.7.2 xradio_codec_ioctl_pcm_write

- **原型**：static int xradio_codec_ioctl_pcm_write(uint8_t *buf, uint32_t size)；
- **传参**：buf：写入 pcm 数据的 buf；size：写入 pcm 数据的大小；
- **功能**：写入 pcm 数据；
- **返回值**：大于 0：写入成功，返回已写入的数据量大小；小于等于 0：写入失败；

2.7.3 xradio_codec_ioctl_set_adda_direct

- **原型**：static int xradio_codec_ioctl_set_adda_direct(Audio_Device device, Audio_Dev_State state)；
- **传参**：device：AUDIO_Device 类型音频设备；state：Audio_Dev_State 类型设备状态；
- **功能**：配置 Xradio Internal Codec ADDA 直通通路；
- **返回值**：HAL_OK：配置成功；other：配置失败；

3 codec_dai_ops 接口

Xradio Internal Codec 的 codec_dai_ops 接口封装如下代码段所示：

```
/*** codec dai ops ****/
static const struct codec_dai_ops xradio_codec_dai_ops = {
    .set_sysclk = xradio_dai_set_sysclk,
    .set_fmt    = xradio_dai_set_fmt,
    .set_volume = xradio_dai_set_volume,
    .set_route  = xradio_dai_set_route,
    .hw_params  = xradio_dai_hw_params,
    .hw_free    = xradio_dai_hw_free,
};
```

3.1 xradio_dai_set_sysclk

- **原型：** static int xradio_dai_set_sysclk(Codec_Sysclk_Src sysclk_src, Codec_Pllclk_Src pllclk_src, uint32_t pll_freq_in, uint32_t sample_rate)；
- **传参：**sysclk_src：系统时钟源选择；pllclk_src：PLL 时钟源选择，未使用；pll_freq_in：PLL 输入频率；sample_rate：采样率；
- **功能：**配置 Xradio Internal Codec 系统时钟；
- **返回值：**HAL_OK：配置成功；other：配置失败；

3.2 xradio_dai_set_fmt

- **原型：**static int xradio_dai_set_fmt(uint32_t fmt)；
- **传参：**fmt：包括主从角色配置、I2S/LJ/RJ/PCM 格式配置、BCLK/LRCK 极性配置，fmt 为 bit mask 组合配置，未使用；
- **功能：**未使用，此接口实现为空；
- **返回值：**HAL_OK：配置成功；

3.3 xradio_dai_set_volume

- **原型：**static int xradio_dai_set_volume(Audio_Device device, uint16_t volume)；
- **传参：**device：AUDIO_Device 类型音频设备；volume：配置音量；
- **功能：**配置 Xradio Internal Codec 对应通道的音量增益；
- **返回值：**HAL_OK：配置成功；other：配置失败；

3.4 xradio_dai_set_route

- **原型**：static int xradio_dai_set_route(Audio_Device device, Audio_Dev_State state)；
- **传参**：device：AUDIO_Device 类型音频设备；state：Audio_Dev_State 类型设备状态；
- **功能**：打开/关闭 Xradio Internal Codec 对应通道；
- **返回值**：HAL_OK：配置成功；other：配置失败；

3.5 xradio_dai_hw_params

- **原型**：static int xradio_dai_hw_params(Audio_Stream_Dir dir, struct pcm_config *pcm_cfg)；
- **传参**：dir：Audio_Stream_Dir 类型音频流方向；pcm_cfg：struct pcm_config 结构体类型指针；
- **功能**：配置 Xradio Internal Codec 的采样率、通道数、采样精度、buffer 大小、DMA 搬运数据宽度以及调用 xradio_codec_hw_common_init 接口进行公共硬件配置；
- **返回值**：HAL_OK：配置成功；other：配置失败；

3.6 xradio_dai_hw_free

- **原型**：static int xradio_dai_hw_free(Audio_Stream_Dir dir)；
- **传参**：dir：Audio_Stream_Dir 类型音频流方向；
- **功能**：停止播放/录音时调用 xradio_codec_hw_common_deinit 接口进行相关硬件清除操作，播放和录音均停止时调用 xradio_codec_reset 接口进行硬件复位操作并 disable AUDIO PLL 输出；
- **返回值**：HAL_OK：配置成功；

4 codec_ops 接口

Xradio Internal Codec 的 codec_ops 接口封装如下代码段所示：

```
/*** codec ops ****/
static const struct codec_ops xradio_codec_ops = {
    .open  = xradio_codec_open,
    .close = xradio_codec_close,

    .reg_read  = xradio_codec_reg_read,
    .reg_write = xradio_codec_reg_write,

    .ioctl = xradio_codec_ioctl,
};
```

4.1 xradio_codec_open

- ✓ **原型：**static int xradio_codec_open(Audio_Stream_Dir dir)；
- ✓ **传参：**dir：Audio_Stream_Dir 类型音频流方向；
- ✓ **功能：**Codec open 时进行相关配置操作，包括申请 DMA buffer、请求 DMA 通道、配置 DMA 参数等；
- ✓ **返回值：**HAL_OK：配置成功；other：配置失败；

4.2 xradio_codec_close

- ✓ **原型：**static int xradio_codec_close(Audio_Stream_Dir dir)；
- ✓ **传参：**dir：Audio_Stream_Dir 类型音频流方向；
- ✓ **功能：**Codec close 时进行相关关闭操作，包括 DMA 反初始化、释放 DMA 通道、释放 DMA buffer 等；
- ✓ **返回值：**HAL_OK：配置成功；other：配置失败；

4.3 xradio_codec_ioctl

- ✓ **原型：**static int xradio_codec_ioctl(uint32_t cmd, uint32_t cmd_param[], uint32_t cmd_param_len)；
- ✓ **传参：**cmd：Codec_Ioctl_Cmd 类型命令；cmd_param[]：命令参数数组指针；cmd_param_len：命令参数数组长度；
- ✓ **功能：**对 Xradio Internal Codec 进行相关 ioctl 命令操作，如 pcm_read、pcm_write 等；
- ✓ **返回值：**HAL_INVALID：无效 ioctl 命令；other：相应命令的执行结果；

5 codec_driver 接口

Xradio Internal Codec 的 codec_driver 接口如下代码段所示：

```
/*** codec driver ****/
static struct codec_driver xradio_internal_codec_drv = {
    .name = XRADIO_INTERNAL_CODEC_NAME,
    .codec_attr = XRADIO_CODEC_INTERNAL,

    .init = xradio_internal_codec_init,
    .deinit = xradio_internal_codec_deinit,

    .dai_ops = &xradio_codec_dai_ops,
    .codec_ops = &xradio_codec_ops,
};
```

5.1 xradio_internal_codec_init

- ✓ **原型：**static int xradio_internal_codec_init(void)；
- ✓ **传参：**无；
- ✓ **功能：**Xradio Internal Codec 模块全局硬件初始化，包括模块时钟 gating、复位的释放和 MCLK enable 等；
- ✓ **返回值：**HAL_OK：配置成功；

5.2 xradio_internal_codec_deinit

- ✓ **原型：**static void xradio_internal_codec_deinit(void)；
- ✓ **传参：**无；
- ✓ **功能：**Xradio Internal Codec 模块全局硬件反初始化， 包括模块时钟 gating、复位的锁定和 MCLK disable 等；
- ✓ **返回值：**无；

6 注册接口

6.1 xradio_internal_codec_register

- **原型：**HAL_Status xradio_internal_codec_register(void)；
- **传参：**无；
- **功能：**Xradio Internal Codec 注册接口，包括申请私有数据内存空间、插入 Codec 链表等；
- **返回值：**HAL_OK：注册成功；HAL_ERROR：注册失败；

此接口不会在外部直接调用，会先在 HAL_SndCard 层封装后再统一给到外部调用，封装后的接口如下代码段所示：

```
HAL_Status HAL_SndCard_CodecRegisterInternal(void)
{
    return xradio_internal_codec_register();
}
```

接口调用如下代码段所示：

```
__weak HAL_Status board_soundcard_init(void)
{
    /* Codec register */
#if PRJCONF_INTERNAL_SOUNDCARD_EN
    HAL_SndCard_CodecRegisterInternal();
#endif

    ......

    return HAL_OK;
}
```

6.2 xradio_internal_codec_unregister

- **原型：**HAL_Status xradio_internal_codec_unregister(void)；
- **传参：**无；
- **功能：**Xradio Internal Codec 反注册接口，包括删除 Codec 链表项、释放私有数据内存空间等；
- **返回值：**HAL_OK：反注册成功；

此接口不会在外部直接调用，会先在 HAL_SndCard 层封装后再统一给到外部调用，封装后的接口如下代码段所示：

```
HAL_Status HAL_SndCard_CodecUnregisterInternal(void)
{
    return xradio_internal_codec_unregister();
}
```

接口调用如下代码段所示：

```
__weak HAL_Status board_soundcard_deinit(void)
{

    ......

    /* Codec unregister */
#if PRJCONF_INTERNAL_SOUNDCARD_EN
    HAL_SndCard_CodecUnregisterInternal();
#endif

    ......

    return HAL_OK;
}
```