TinaLinux

AVS 使用指南

1.0

2019.07.02

文档履历

版本号	日期	制/修订人	内容描述
1.0	2019.07.02	曾治金	创建

目录

1. TinaLinux AVS 使用指南

1.1 概述

1.1.1 编写目的

介绍 TinaLinux 平台上 AVS SDK 的简易使用，实现 Alexa Demo 开发。

1.1.2 适用范围

文档基于 Allwinner R18 搭载 AC108、GMEMS 方案编写，适合 TinaLinux 的 AVS 开发和测试人员阅读，当前仅支持 tulip-noma 方案。

1.2 整体介绍

1.2.1 AVS 基础介绍

AVS-SDK 是对接 Alexa 服务的 Device-CPP-SDK，由亚马逊负责开发维护，在 github 上发布。作为设备端对接，只需要关注三件事情：1. AFE（音频采集处理）输出的数据灌入 Shared Data Stream。2. WakeWord Detection 对接对应的唤醒算法。AVS-SDK 中已经对接好 Sensory 与 Kitt.AI，只要把对应的算法实现放进入，指定路径就可以使用。3. AVS-SDK 处理好云端返回来的结果，通过 Gstreamer 去播放。GStreamer 播放的兼容性，需要设备端解决。

1.2.2 libtutuclear 介绍

Libtutuclear 是 GMEMS 的三麦音频前端处理算法库，主要的功能有：- 噪声压制 - 回音消除（mono、stereo）- 识别音频流输出，适合机器做识别，用于云端识别 - 识别音频流输出，没有加任何的数字增益，适合做能量检测，用于 ESP 处理 - 通话音频流输出，适合人耳听，用于通话 - DOA 信息输出

1.2.2.1 PRM 文件

libtutuclear 通过 PRM 文件配置，在库初始化时，把 PRM 文件解析成数据传给库。

1.2.2.2 主要接口

```
W16 TUTUClear_OneFrame(void *pTUTUClearObject, W16 *ptAECRef, W16 *ptMIC, W16 *ptLOut, TUTUClearStat_t *ptTUTUClearStat);

W16 TUTUClear_OneFrame_32b(void *pTUTUClearObject, W32 *pw32AECRef, W32 *pw32MIC, W32 *pw32LOut, TUTUClearStat_t
    *ptTUTUClearStat);
```

1.2.3 libconfigutils 介绍

libconfigutils 是全志科技提供的平台适配层，用来解决平台差异化问题，通过配置文件去让一套应用代码适配所有的平台。目前配置功能包括：通用需求：- 音频采集，包括单颗 AC108、两个 AC108，AudioCodec 配合做回路等 - 音频前处理，音频格式（采样率、采样精度）转化，GMEMS 算法处理 - privacyMute 控制 - 按键检测。

特定方案需求：- 耳机插拔检测 - LED 阵列控制 - 采集音频 loopback

1.2.4 libpron-lite 介绍

亚马逊提供的唤醒词识别算法库。

1.2.4.1 libconfigutils 配置文件说明

```
{
    "platform" : "auto", 平台自动选择，也可以指定某个方案，如tulip-mozart。
    "tulip-mozart" : { tulip-mozart方案的配置
```

```
"voice-loopback" : 1, 使能通话音频loopback
"detector" : "amazon-lite", /*amazon-lite*/ 选择唤醒算法，目前支持一种。
"provider" : "ac108", 采用AC108来前端采集
"filter" : "tutuclear", 采用tutuclear来做音频前端处理，名字可以随意。
"tutuclear" : { tutuclear的差异配置，会和后面的基础配置做替换
    "base" : "tutuclear", 继承于tutuclear的基础配置
    "prm-file" : "/etc/avs/tutuClearA1_ns4wakeup_mono.prm",
    "signal-channel-0" : "ac108-0", ac108的第0路做tutuclear麦克风输入的第0路
    "signal-channel-1" : "ac108-1", 同上，根据硬件连接配置
    "signal-channel-2" : "ac108-2",
    "reference-channel-0" : "ac108-3", ac108的第3路做tutuclear回路参考输入的第0路
    "reference-channel-1" : ""
},
"ac108" : { ac108的差异配置，会和后面的基础配置做替换
    "device" : "hw:sndac1082003b,0", 声卡号
    "base" : "ac108",
    "channels" : 4, 单颗AC108 4通道采集
},
"mute" : { privacyMute配置，通过sys/gpio控制mute电路
    "mute-enable" : "active", 遍历gpio、amixer列表中active配置
    "mute-disable" : "disactive", 遍历gpio、amixer列表中disactivepriva配置
    "gpio" : [
        {
            "pin" : 100, gpio的pin号
            "active" : 1,
            "disactive" : 0,
        }
    ],
    "amixer" : []
},
"button" : [ 按键控制，目前只设计了三个按键，耳机插拔的伪按键，可以根据需求增加，但是也需要增加相应的代码。
    {
        "name" : "volume-up", 按键名，需要和代码对应，不能改
        "input" : "sunxi-keyboard", 按键的input name
        "code" : 115, 按键的key code
    }
],
"audio-jack" : { 耳机插拔事件控制，不需要的方案可以删除
    "plugout" : "active", /*TODO*/ 拔出，遍历gpio、amixer列表中active配置
    "plugin" : "disactive", /*TODO*/ 插入，遍历gpio、amixer列表中disactive配置
    "gpio" : [
        .....
    ],
    "amixer" : [
        {
            "device" : "hw:audiocodec", amixer目标声卡
            "iface" : "MIXER",
            "name" : "'Left Output Mixer DACL Switch'", amixer 名字
            "active" : false, /*plugout*/ active值，可以是int、bool、string
            "disactive" : true, /*plugin*/
        }
    ],
```

```
        },
        "gobal-volume" : { 暂时不用
            "volume" : [0, 50, 80, 100, 130, 160, 190, 210, 220, 250],
            "device" : "hw:audiocodec",
            "iface" : "MIXER",
            "name" : "'AudioCodec Master PCM softvol'",
            "value" : [0,255],
        }
    },
    "tulip-mozart-new" : { 另外一份配置，解释一下差异化
        ........
        "provider" : ["ac108", "audiocodec"], ac108做3mic采集，audiocodec adc做回路采集
        "filter" : "tutuclear",
        "tutuclear" : {
            "base" : "tutuclear",
            "signal-channel-0" : "ac108-0", ac108的第0路做tutuclear麦克风输入的第0路
            "signal-channel-1" : "ac108-1",
            "signal-channel-2" : "ac108-2",
            "reference-channel-0" : "audiocodec-0", audiocodec的第0路做tutuclear回路参考输入的第0路
            "reference-channel-1" : "audiocodec-1"
        },
        "audiocodec" : {
            "base" : "audiocodec",
            "sample-rate" : 48000, audiocodec特性，要用48K采样率
            "output-data-file" : "" 调试，保存音频原始数据
        },
    },
    "tulip-noma" : {
        "led-ring" : { led阵列配置
            "direction" : {
                "angle-start" : 307.5, 位置校准，第一个led在麦克风坐标的角度，参考gmems的麦克风坐标文档。
                "angle-step" : 30.0,
            }
        }
    },

    下面是基础配置
    "tutuclear" : {
        "type" : "filter",
        "prm-file" : "/etc/avs/tutuClearA1_ns4wakeup_stereo.prm",
        "filter-block" : 160,
        "input-sample-rate" : 16000,
        "input-sample-bits" : 32,
        "input-channels" : 5,
        "output-sample-rate" : 16000,
        "output-sample-bits" : 32,
        "output-channels" : 3,
        "signal-channel-0" : "1",
        "signal-channel-1" : "3",
        "signal-channel-2" : "5",
        "reference-channel-0" : "6",
        "reference-channel-1" : "7", /*mono AEC: file "" here*/
```

```
        "output-data-file" : ""
    },

    "file" : { 把wav文件当成数据输入，适用于保存的音频数据做算法测试
        "type" : "provider",
        "name" : "file",
        "format" : "wav", /*pcm*/
        "channels" : "8",
        "sample-rate" : "16000",
        "sample-bits" : "16",
        "path" : "/tmp/xxxx.wav"
    },
    "ac108" : { AC108的采集参数设置
        "type" : "provider",
        "name" : "ac108",
        "device" : "hw:sndac10810035,0",
        "channels" : 8,
        "period-size" : 1024,
        "period" : 4,
        "sample-rate" : 16000,
        "sample-bits" : 24,
        "mode" : "normal", normal模式，正常的数据；encode模式，通道错乱模式，如R16
        "output-data-file" : ""
    },
    "audiocodec" : {
        "type" : "provider",
        "name" : "audiocodec",
        "device" : "hw:audiocodec,0",
        "channels" : 2,
        "period-size" : 1024,
        "period" : 4,
        "sample-rate" : 16000,
        "sample-bits" : 16,
        "output-data-file" : ""
    },
    唤醒算法的参数设置，如模型路径，阈值等等。
    "sensory" : {
        "model" : "/etc/avs/thfft_alexa_a_enus_v3_1mb.snsr",
        "operating-point" : "8"
    },
    "amazon-lite" : {
        "model" : "/etc/avs/D.en-US.alexa.bin",
        "detect-threshold" : 500
    },
    "tutudetect" : {
    },
    "tulip-noma-test-file" : { 这些是测试配置，可以把文件的数据当成一个输入，测试前段算法性能
        "voice-loopback" : 0,
        "detector" : "sensory", /*sensory or amazon-lite or tutudetect*/
        "provider" : "file123",
        "filter" : "tutuclear11",
        "tutuclear11" : {
```

```
            "base" : "tutuclear",
            "signal-channel-0" : "file123-1",
            "signal-channel-1" : "file123-5",
            "signal-channel-2" : "file123-7",
            "reference-channel-0" : "file123-2",
            "reference-channel-1" : "file123-3"
        },
        "file123" : {
            "base" : "file",
            "name" : "file123",
            "path" : "/mnt/UDISK/ac108-2010-01-01-01-08-33-34-414300.wav",
        },
    }
}
```

1.3 编译配置

menuconfig 配置如下：

```
avs --->
        -*- avs-sdk..... avs library
        -*- libconfigutils..... ConfigUtils for All
        -*- libpryon-lite..... Amazon pryon-lite keyword detect lib
        -*- libsensory..... Sensory keyword detect lib
        -*- libtutuclear..... tutu clear, audio front engine
        <*> sampleapp..... avs sdk SampleApp only
```

1.4 Alexa Demo 运行方法

SampleApp 是基于 AVS-SDK/SampleApp 的二次开发的软件包，主要包括：- SampleApp，基于 AVS-SDK/SampleApp 做了简单的修改，对接了 AC08、GMEMS 前端，耳机插拔切换逻辑，Mute 控制，以及 Button 的管理控制。- sample-wakeup，离线唤醒测试用例，可以用来测试前端音频性能。- fetch-device-sn，把 chipid 自动填入 AlexaClientSDKConfig.json 的工具。

1.4.1 执行命令

1.4.1.1 fetch-device-sn

fetch-device-sn /etc/avs/AlexaClientSDKConfig.json - 参数/etc/avs/AlexaClientSDKConfig.json 为 AVS SDK 的配置文件

1.4.1.2 SampleApp

SampleApp /etc/avs/AlexaClientSDKConfig.json /etc/avs DEBUG0

- 参数/etc/avs/AlexaClientSDKConfig.json 为 AVS SDK 的配置文件
- /etc/avs 为 Allwinner 平台适配层的配置文件路径
- DEBUG0 是调试打印级别，可设置为 DEBUG0~9

1.4.1.3 sample-wakeup

sample-wakeup /etc/avs/config.json

- 参数 /etc/avs/config.json 是 Allwinner 平台适配层的配置文件

2. Declaration

This document is the original work and copyrighted property of Allwinner Technology (“Allwinner”). Reproduction in whole or in part must obtain the written approval of Allwinner and give clear acknowledgement to the copyright owner. The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. 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 Allwinner. This datasheet neither states nor implies warranty of any kind, including fitness for any particular application.tates nor implies warranty of any kind, including fitness for any particular application.