

# 鸿鹄芯片离线识别集成工作概述

Revision Record 修订记录

修改时间	修订版本	修改内容	备注	修改人
201209	1.0.1	初始版本		王瑛

## 一 DSP 资源分布

core0 负责处理唤醒事务，并将音频信号做 wpe 处理，放到循环 buffer 中供 core1 使用  
目前 wpe buffer 深度 140 帧 10ms 音频数据，双 mic 信号。  
即 mic0 0-159 mic1 0-159 排列。  
core1 负责处理 vad（判断识别的起点和尾点）  
卷积 + attention：离线识别模型。

## 二 项目代码获取

icode 地址：  
<http://console.cloud.baidu-int.com/devops/icode/repos/baidu/speech-chip/AIoT/tree/master>  
目前有两个主要分支：  
master 是空调扇项目，该项目使用 doavad  
目前发布到 1.1.1 版本  
jiuyang\_gruvad 是豆浆机项目，该项目使用 gruvad  
目前发布到 1.0.3 版本，即将发布 1.0.4 版本

## 三 集成工作

算法移植同学将提供如下文件：

1 模型文件

将模型文件放置到如下位置

tool/gen\_header/build/optest/

2 生成头文件步骤

cd tool/gen\_header/header

rm -rf \*

sh ../gen\_header.sh

再次执行 sh ../gen\_header.sh

head 目录有如下文本文件

a2a\_att\_info.txt a2a\_conv\_info.txt a2a\_vad\_info.txt

修改其中的配置信息，即可实现模型参数的地址分配

例如 a2a\_vad\_info.txt

存储位置 op 模型参数长度

10 a2a\_vad\_op\_5 608 10--- 代表 dram

2 a2a\_vad\_op\_6 8192 2 --- 代表存储在 sram 空间

10 a2a\_vad\_op\_8 104

2 a2a\_vad\_op\_9 4096

10 a2a\_vad\_op\_11 192

**批注 [MO用1]：**为啥要执行两次  
先解析出配置文件，再解析

**批注 [MO用2]：**1.需要 demo 跑一下  
2.可修改的参数有哪些，修改经验  
一般不需要该参数，当前算力够

```
10 a2a_vad_op_12    4096
10 a2a_vad_op_13    208
10 a2a_vad_op_14    4096
10 a2a_vad_op_16    640
10 a2a_vad_op_17    4096
10 a2a_vad_op_18    928
10 a2a_vad_op_19    4096
10 a2a_vad_op_21    1152
10 a2a_vad_op_22    2048
10 a2a_vad_op_25    33280
10 a2a_vad_op_26    512
10 a2a_vad_op_27    3456
10 a2a_vad_op_28    2560
10 a2a_vad_op_29    512
10 a2a_vad_op_30    3456
10 a2a_vad_op_31    2560
10 a2a_vad_op_32    512
10 a2a_vad_op_33    3456
10 a2a_vad_op_34    1152
10 a2a_vad_op_36    128
10 a2a_vad_op_37    288
10 a2a_vad_op_39    32
10 a2a_vad_op_40    288
10 a2a_vad_op_42    32
```

修改后执行 `../gen_header.sh 1`  
生成 dram 文件， 将\*.h 拷贝到 `europa_mbf_core1/dram_only/`

修改后执行 `../gen_header.sh 2`  
生成 sram 文件， 将\*.h 拷贝到 `europa_mbf_core0/sram_only/`

3 同步引擎修改：

目前 pc 上运行的代码已经非常接近板上运行的环境。

引擎包含：手动 merge 修改到 `europa_mbf_core1` 对应文件

`a2a_engine_api.cpp`

`a2a_asr_engine.cpp`

`a2a_asr_engine.h`

`a2a_engine_api.h`

4 wpe 相关

wpebuffer 深度修改：`common/europa_driver/wak_asr_cfg.h`

```
113 #define WPE_BUF_DEPTH    140
```

该数据和唤醒库相关，目前唤醒库是 140 帧

5 版本号：`common/europa_driver/wak_asr_cfg.h`

```
30 #define BD_ALIGN __attribute__((aligned(16)))
```

```
31 #define VERSION_NAME    "V1.0.4"
```

**批注 [M0用3]：**什么情况下需要修改深度  
识别算力不够的话需要更多的 buffer

32 #define VERSION\_NAME\_HEX

0x31303444

批注 [M0用4]: 手动转化？

44 是 D ？啥意思

D 是豆浆机

S 是空调扇

## 四 编译

cd europa\_mbf\_core1/build/

build.sh 参数描述

参数 1：clean 表示需要 clean 掉 core0 现有编译。重新编译 core0，  
其他表示不需要重新编译 core0

参数 2：clean 表示需要 clean 掉 core1 现有编译。重新编译 core1，  
其他表示不需要重新编译 core1

参数 3: master：表示编译 masterboot 版本  
slave：表示编译灌测版本

参数 4: 1: 表示每次唤醒均使用首次模式  
0: 表示唤醒使用正常模式

参数 5: 版本号，目前格式为 1.0.4

参数 6: 版本号对应的 hex 格式，例如 0x313034 对应参数 5 的编码（ascii）

批注 [M0用5]: 1.什么是首次模式

首次模式不好唤醒，但是精度较高（唤醒首尾），前期  
vad 不好的时候用

现在 vad 比较好用了，都使用正常模式

1 编译 master boot 版本

bash build.sh clean clean master 0 1.0.4 0x313034

参考 log 如下：

```
Compile done.
out_filename: flash_v1.0.4D.bin
----- CORE0 USAGE -----
core0_iram:      using len: 93.398K remain len: 34.602K
core0_dram_data: using len: 400.688K
               dram_bss: using len: 48.688K remain len: 46.625K
----- CORE1 USAGE -----
core1_iram:      using len: 94.867K remain len: 33.133K
core1_dram_data: using len: 143.781K
               dram_bss: using len: 22.266K remain len: 329.953K
----- SRAM USAGE -----
sram0:          using len: 0.000K remain len: 64.000K
sram1:          using len: 225.875K remain len: 30.125K
sram2:          using len: 665.922K remain len: 38.078K
sram3:          using len: 372.500K remain len: 11.500K
sram4:          using len: 94.500K remain len: 1.500K
total:          using len:1358.797K remain len: 145.203K
done
```

2 编译 灌测版本

bash build.sh clean clean slave 0 1.0.4 0x313034

```

Compile done.
ver: 1.0.5 -- build time: 201014
out_filename: flash_slv_v1.0.4.N.bin
----- CORE0 USAGE -----
core0_iram:      using len: 93.711K remain len: 34.289K
core0_dram_data: using len: 400.969K
               dram_bss: using len: 40.453K remain len: 54.578K
----- CORE1 USAGE -----
core1_iram:      using len: 95.742K remain len: 32.258K
core1_dram_data: using len: 145.422K
               dram_bss: using len: 22.328K remain len: 328.250K
----- SRAM USAGE -----
sram0:           using len: 0.000K remain len: 64.000K
sram1:           using len: 225.875K remain len: 30.125K
sram2:           using len: 666.172K remain len: 37.828K
sram3:           using len: 380.500K remain len: 3.500K
sram4:           using len: 94.500K remain len: 1.500K
total:           using len: 1367.047K remain len: 136.953K
done

```

3 todo :

master 版本分为 lcd （demo 版本） 和 uart 版本（发给豆浆机的串口命令）

目前需要手动修改配置文件实现，后续也将添加到编译脚本中

common/europa\_driver/register.h

```

56 #if USE_ESP32_SPI_I2S
57 #undef USE_LCD
58 #undef USE_SEND2HOST
59 #else
60 #define USE_LCD 1
61 #define USE_SEND2HOST 0
62 #if USE_LCD

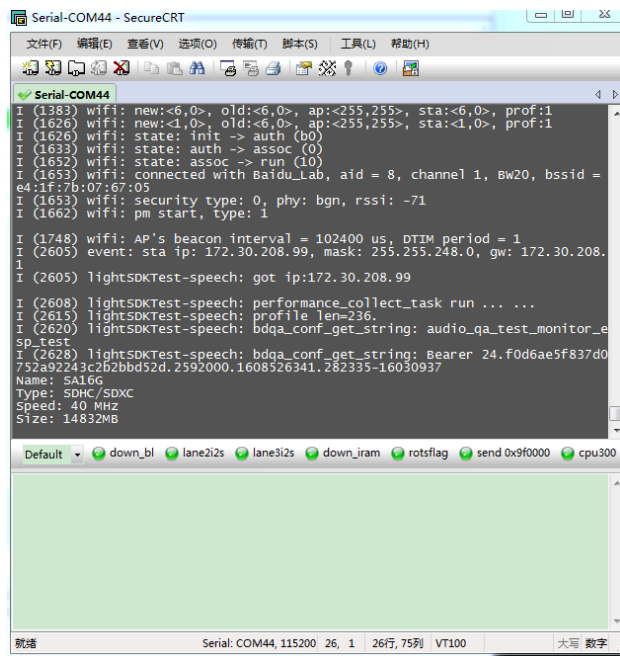
```

## 五 灌测

目前采用 du1906 平台实现灌测，该平台用于哪吒的灌测环境

批注 [M0用6]: 需要将 dsp 的串口接出来





```
I (1383) wifi: new:<6,0>, old:<6,0>, ap:<255,255>, sta:<6,0>, prof:1
I (1626) wifi: new:<1,0>, old:<6,0>, ap:<255,255>, sta:<1,0>, prof:1
I (1626) wifi: state: init -> auth (b0)
I (1633) wifi: state: auth -> assoc (0)
I (1632) wifi: state: assoc -> run (10)
I (1653) wifi: connected with Baldu_Lab, aid = 8, channel 1, BW20, bssid =
e4:1f:7b:07:67:05
I (1653) wifi: security type: 0, phy: bgn, rssi: -71
I (1662) wifi: pm start, type: 1

I (1748) wifi: AP's beacon interval = 102400 us, DTIM period = 1
I (2605) event: sta ip: 172.30.208.99, mask: 255.255.248.0, gw: 172.30.208.
1
I (2605) lightsDKTest-speech: got ip:172.30.208.99

I (2608) lightsDKTest-speech: performance_collect_task run ... ..
I (2615) lightsDKTest-speech: profile len=236.
I (2620) lightsDKTest-speech: bdqa_conf_get_string: audio_qa_test_monitor_e
sp_test
I (2628) lightsDKTest-speech: bdqa_conf_get_string: Bearer 24.f0d6ae5f837d0
752a92243c2b2bbd52d.2592000.1608526341.282335-16030937
Name: SA16G
Type: SDHC/SDXC
Speed: 40 MHz
Size: 14832MB
```

表示启动成功

## 2 加载 dsp 固件

输入 opus, 即开始加载 dsp 固件。

关闭 esp 串口 com44

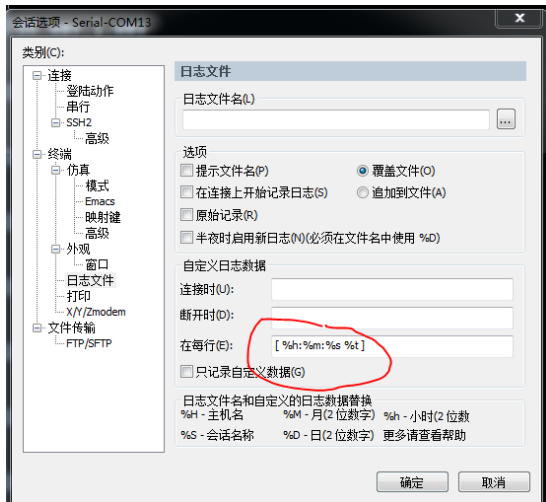
打开 dsp 串口, 会看到 dsp 启动过程

## 3 开始灌测

选择 dsp 串口文件保存路径

选择在每行前加入时标。





执行 sh uart\_test\_all2.sh jiuyang\_wz65dbA.lst no  
参数 1 文件列表  
参数 2 no 表示每次测试不复位 dsp，如需复位则须输入 reset

4 结果分析  
使用 find\_err 脚本进行分析

```
~/gitbase/nn_lib jiuyang_gru_2att_new/builds bash ~/bin/find_err.sh jiuyang_wz65dba_201209 11 jiuyang_wz65dbA.lst 33 time
file      : jiuyang_wz65dba_201209
result_file : 11
list      : jiuyang_wz65dbA.lst
output    : 33
find over
total: 248
begin with 1 count 248
good count 233
error count 15
total      248
rate = 93.95161200 %
```

参数 1: dsp 串口 log 文件  
参数 2 : 分析结果文件  
参数 3 : 灌测文件列表  
参数 4: 分析的 log 文件  
参数 5: 固定为 time

批注 [M0用7]: 去掉上面串口增加的时间戳，写死即可

六 验证及发布

需要发布三个版本 lcd 版本，uart 版本，灌测版本  
目前 uart 版本暂时不会更新，  
灌测版本需要灌测 aj45, aj90, wz45, wz90, longting, rbb, wz65dbA  
供 1532 条，大约需要 5.5 小时。  
目前结果如下：

批注 [M0用8]: 灌测结果是直接生成的吗

测试集	aj45	aj90	wz45	wz90	longting	rbb	wz65dbA
PC成功率	0.9866	0.9866	0.9866	0.9792	0.939	0.980	0.9395
灌测成功率	0.9766	0.9799	0.9866	0.9792	0.918	0.980	0.9395

lcd 版本需要人工验证一下，目前有 12 种识别结果。需要验证正例，负例。

验证完毕后即可发布。