




# ESP32

## ESP-SR 用户手册



Release master  
乐鑫信息科技  
2025 年 01 月 08 日



# Table of contents

<b>Table of contents</b>	<b>i</b>
<b>1 入门指南</b>	<b>3</b>
1.1 概述	3
1.2 准备工作	3
1.2.1 必备硬件	3
1.2.2 必备软件	3
1.3 编译运行一个示例	3
<b>2 AFE 声学前端</b>	<b>5</b>
2.1 AFE 声学前端算法框架	5
2.1.1 概述	5
2.1.2 使用场景	5
2.1.3 选择 AFE Handle	8
2.1.4 输入音频	8
2.1.5 输出音频	9
2.1.6 使能唤醒词识别 WakeNet	9
2.1.7 使能回声消除算法 AEC	9
2.1.8 资源消耗	9
2.2 乐鑫麦克风设计指南	10
2.2.1 麦克风电器性能推荐	10
2.2.2 麦克风结构设计建议	10
2.2.3 麦克阵列设计建议	10
2.2.4 麦克风结构密封性建议	10
2.2.5 回声参考信号设计建议	11
2.2.6 麦克风阵列一致性验证	11
<b>3 唤醒词</b>	<b>13</b>
3.1 WakeNet 唤醒词模型	13
3.1.1 概述	13
3.1.2 WakeNet 的使用	14
3.1.3 资源消耗	15
3.2 乐鑫语音唤醒方案客户定制流程	15
3.2.1 唤醒词定制服务	15
3.2.2 硬件设计与测试服务	16
<b>4 命令词</b>	<b>17</b>
4.1 MultiNet 命令词识别模型	17
4.2 命令词识别原理	17
4.3 命令词格式要求	18
4.4 自定义命令词方法	18
4.4.1 MultiNet6 定义方法:	18
4.4.2 MultiNet5 定义方法:	18
4.4.3 通过调用 API 修改	19
4.5 MultiNet 的使用	20
4.5.1 MultiNet 初始化	20
4.5.2 MultiNet 运行	21

4.5.3	MultiNet 识别结果	21
4.6	资源消耗	21
<b>5</b>	<b>TTS 语音合成模型</b>	<b>23</b>
5.1	简介	23
5.2	简单示例	23
5.3	编程指南	24
5.4	资源消耗	24
<b>6</b>	<b>模型加载</b>	<b>25</b>
6.1	配置方法	25
6.1.1	使用 AFE	25
6.1.2	使用 WakeNet	25
6.1.3	使用 MultiNet	26
6.2	模型使用	26
6.2.1	模型数据存储在 Flash	27
<b>7</b>	<b>性能测试结果</b>	<b>29</b>
7.1	AFE	29
7.1.1	资源消耗	29
7.2	WakeNet	29
7.2.1	资源消耗	29
7.2.2	性能测试	29
7.3	MultiNet	30
7.3.1	资源消耗	30
7.3.2	Word Error Rate 性能测试	30
7.3.3	Speech Commands 性能测试 (空调控制场景)	30
7.4	TTS	30
7.4.1	资源消耗	30
7.4.2	性能测试	30
7.5	NSNET	31
7.5.1	性能测试	31
7.5.2	数据集: array_onemic_nnoise_20230608(按照亚马逊声学认证标准录制测试集)	31
<b>8</b>	<b>测试方法与测试报告</b>	<b>33</b>
8.1	测试场景要求	33
8.2	测试案例设计	33
8.3	乐鑫测试与结果	34
8.3.1	唤醒率测试	35
8.3.2	语音识别率测试	35
8.3.3	误唤醒率测试	35
8.3.4	唤醒打断率测试	35
8.3.5	响应时间测试	36
<b>9</b>	<b>术语表</b>	<b>37</b>
9.1	通用术语	37
9.2	特别术语	37

本文档仅包含针对 ESP32 芯片的 ESP-SR 使用。



# Chapter 1

## 入门指南

乐鑫 [ESP-SR](#) 可以帮助用户基于 ESP32 系列芯片或 ESP32-S3 系列芯片，搭建 AI 语音解决方案。本文档将通过一些简单的示例，展示如何使用 ESP-SR 中的算法和模型。

### 1.1 概述

ESP-SR 支持以下模块：

- [声学前端算法 AFE](#)
- [唤醒词检测 WakeNet](#)
- [命令词识别 MultiNet](#)
- [语音合成](#)（目前只支持中文）

### 1.2 准备工作

#### 1.2.1 必备硬件

- 一款音频开发板，推荐使用 [ESP32-Korvo](#)
- [USB 2.0 数据线](#)（标准 A 型转 Micro-B 型）
- [电脑](#)（Linux）

---

**备注：**目前一些开发板使用的是 USB Type C 接口。请确保使用合适的数据线连接开发板！

---

#### 1.2.2 必备软件

- 下载 [ESP-SKAINET](#)，ESP-SR 将作为 ESP-SKAINET 的组件被一起下载。
- 配置安装 ESP-IDF，推荐使用 [ESP-SKAINET](#) 中包含的版本。安装方法请参考 [ESP-IDF 编程指南](#) 中的 [快速入门](#) 小节。

### 1.3 编译运行一个示例

- 进入 [ESP-SKAINET/examples/cn\\_speech\\_commands\\_recognition](#) 目录。
- 参考该示例目录下的配置和编译说明，运行该示例。

- 该示例为中文命令指令识别示例，通过说唤醒词（Hi, 乐鑫），触发语音指令识别。注意，当一段时间没有语音指令后，语音指令识别功能将关闭，并等待下一次唤醒词触发。

## Chapter 2

# AFE 声学前端

## 2.1 AFE 声学前端算法框架

### 2.1.1 概述

智能语音设备需要在远场噪声环境中，仍具备出色的语音交互性能，声学前端 (Audio Front-End, AFE) 算法在构建此类语音用户界面 (Voice-User Interface, VUI) 时至关重要。乐鑫 AI 实验室自主研发了一套乐鑫 AFE 算法框架，可基于功能强大的 ESP32 系列芯片进行声学前端处理，使用户获得高质量且稳定的音频数据，从而构建性能卓越且高性价比的智能语音产品。

名称	简介
AEC (Acoustic Echo Cancellation)	回声消除算法，最多支持双麦处理，能够有效的去除 mic 输入信号中的自身播放声音，从而可以在自身播放音乐的情况下很好的完成语音识别。
NS (Noise Suppression)	噪声抑制算法，支持单通道处理，能够对单通道音频中的非人声噪声进行抑制，尤其针对稳态噪声，具有很好的抑制效果。
BSS (Blind Source Separation)	盲信号分离算法，支持双通道处理，能够很好的将目标声源和其余干扰音进行盲源分离，从而提取出有用音频信号，保证了后级语音的质量。
MISO (Multi Input Single Output)	多输入单输出算法，支持双通道输入，单通道输出。用于在双麦场景，没有唤醒使能的情况下，选择信噪比高的一路音频输出。
VAD (Voice Activity Detection)	语音活动检测算法，支持实时输出当前帧的语音活动状态。
AGC (Automatic Gain Control)	自动增益控制算法，可以动态调整输出音频的幅值，当弱信号输入时，放大输出幅度；当输入信号达到一定强度时，压缩输出幅度。
WakeNet	基于神经网络的唤醒词模型，专为低功耗嵌入式 MCU 设计

### 2.1.2 使用场景

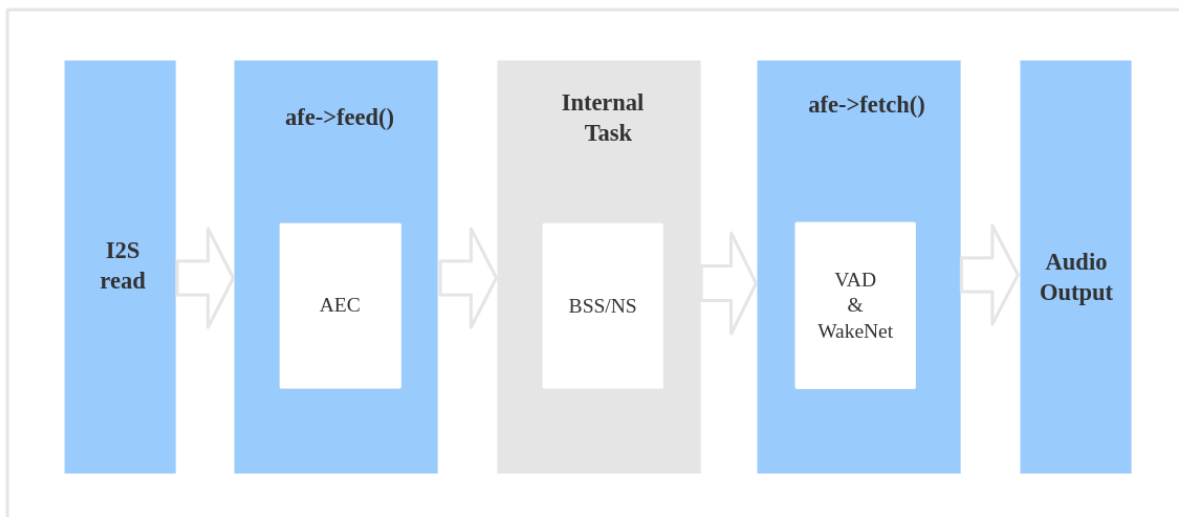
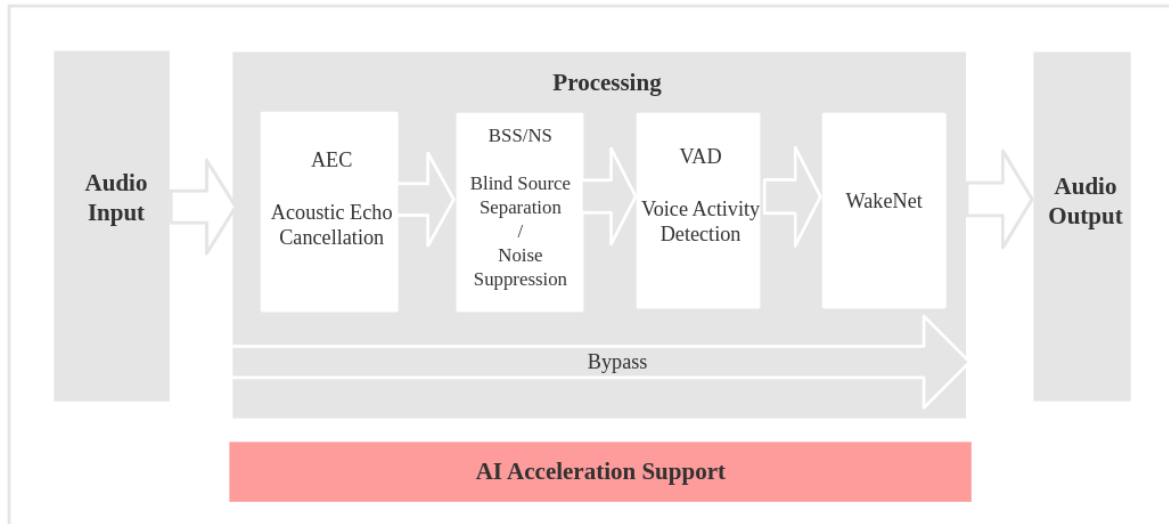
本节将介绍乐鑫 AFE 框架的两个典型使用场景。

#### 语音识别场景

#### 工作流程

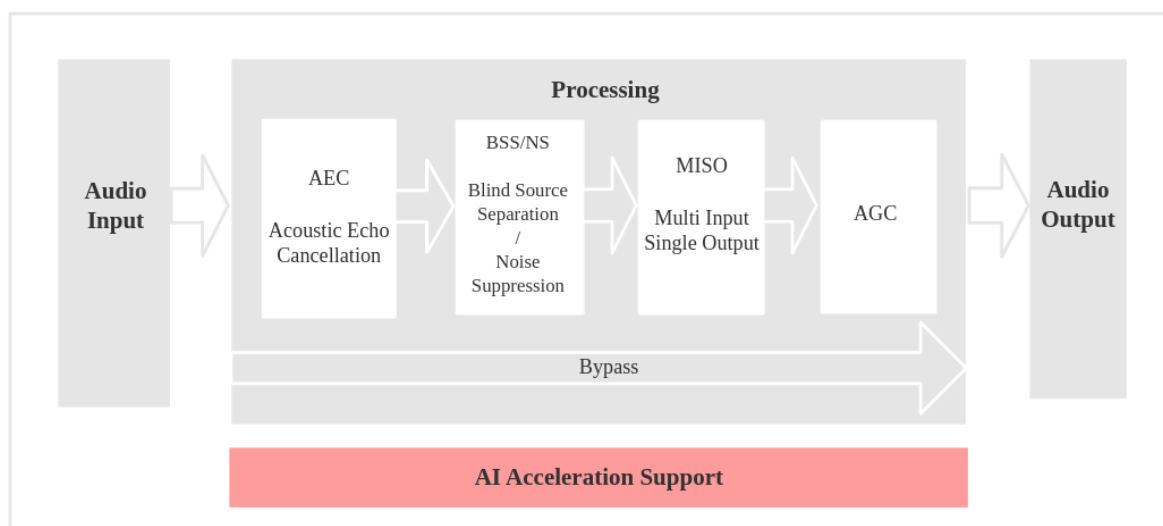
#### 数据流



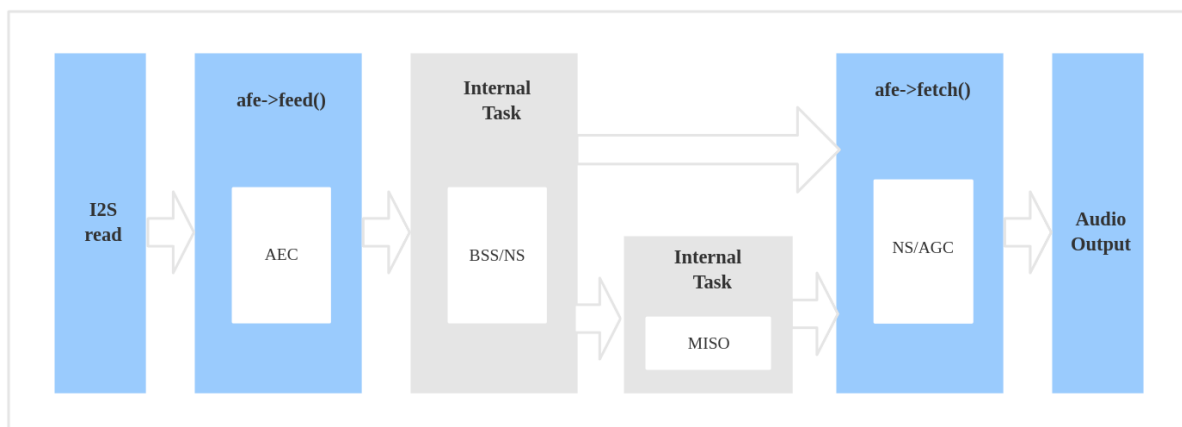


1. 使用 `ESP_AFE_SR_HANDLE()`，创建并初始化 AFE。注意，`voice_communication_init` 需配置为 `false`。
2. 使用 `feed()`，输入音频数据。`feed` 内部会先进行 AEC 算法处理
3. `Feed` 内部进行 BSS/NS 算法处理
4. 使用 `fetch()`，获得经过处理过的单通道音频数据及相关信息。这里，`fetch` 内部可以进行 VAD 处理并检测唤醒词等动作，具体可通过 `afe_config_t` 结构体配置。

### 语音通话场景



### 工作流程



### 数据流

1. 使用 `ESP_AFE_VC_HANDLE()`，创建并初始化 AFE。注意，`voice_communication_init` 需配置为 `true`。
2. 使用 `feed()`，输入音频数据。`feed` 内部会先进行 AEC 算法处理
3. `Feed` 内部进行 BSS/NS 算法处理。若为双麦，还将额外进行 MISO 算法处理。
4. 使用 `fetch()`，获得经过处理过的单通道音频数据及相关信息。这里，可对输出数据进行 AGC 非线性放大，具体增益值可通过 `afe_config_t` 结构体配置。注意，若为双麦，则在进行 AGC 非线性放大前还会进行降噪处理。

备注:

1. `afe_config_t` 结构体中的 `wakenet_init` 和 `voice_communication_init` 不可同时配置为 `true`。
2. `feed()` 和 `fetch()` 对用户可见，其他内部 BSS/NS/MISO 算法处理为 AFE 的内部独立任务，对用户不可见。
3. AEC 算法处理在 `feed()` 中进行。
4. 当 `aec_init` 配置为 `false`，BSS/NS 算法处理在 `feed()` 中进行。

### 2.1.3 选择 AFE Handle

目前，乐鑫 AFE 框架支持单麦和双麦配置，并允许对算法模块进行灵活配置。

- **单麦配置：**
  - 内部 Task 由 NS 算法模块处理
- **双麦配置：**
  - 内部 Task 由 BSS 算法模块处理
  - 此外，如用于语音通话场景（即 `wakenet_init = false` 且 `voice_communication_init = true`），则会再增加一个内部 Task 由 MISO 处理。

获取 AFE handle 的命令如下：

- 语音识别场景

```
esp_afe_sr_iface_t *afe_handle = &ESP_AFE_SR_HANDLE;
```

- 语音通话场景

```
esp_afe_sr_iface_t *afe_handle = &ESP_AFE_VC_HANDLE;
```

### 2.1.4 输入音频

目前，乐鑫 AFE 框架支持单麦和双麦配置，可根据 `esp_afe_sr_iface_op_feed_t()` 的输入音频情况，配置所需的音频通道数。

具体方式为：配置 `AFE_CONFIG_DEFAULT()` 中的 `pcm_config` 结构体成员：

- `total_ch_num`：总通道数
- `mic_num`：麦克风通道数
- `ref_num`：参考回路通道数

注意，在配置时有如下要求：

1. `total_ch_num = mic_num + ref_num`
2. `ref_num = 0` 或 `ref_num = 1`（由于目前 AEC 仅只支持单回路）

在上述要求下，几种支持的配置组合如下：

```
total_ch_num=1, mic_num=1, ref_num=0
total_ch_num=2, mic_num=1, ref_num=1
total_ch_num=2, mic_num=2, ref_num=0
total_ch_num=3, mic_num=2, ref_num=1
```

#### AFE 单麦配置

- 输入音频的格式为 16 KHz、16 bit、双通道（其中 1 个通道为 mic 数据，另 1 个通道为参考回路）。注意，若不需要 AEC 功能，则可只包含 1 个通道输入 mic 数据，而无需配置参考回路（即可配置 `ref_num = 0`）。
- 根据用户配置的算法模块不同，输入音频的帧长将有所差异，具体可通过 `get_feed_chunksize()` 来获取需要的采样点数目（采样点数据类型为 `int16`）。

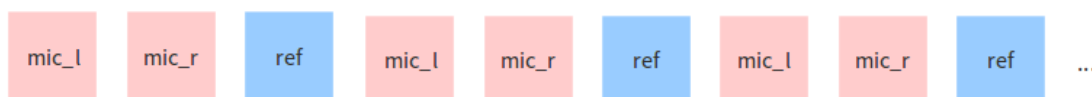
数据排布示意如下：



### AFE 双麦配置

- 输入音频格式为 16 KHz、16 bit、三通道（其中 2 个通道为 mic 数据，另 1 个通道为参考回路）。注意，若不需要 AEC 功能，则可只包含 2 个通道 mic 数据，而无需配置参考回路（即可配置 `ref_num = 0`）。
- 根据用户配置的算法模块不同，输入音频的 **帧长** 将有所差异，具体可通过 `get_feed_chunksize()` 来获取需要填充的数据量（即 `get_feed_chunksize() * total_ch_num * sizeof(short)`）。

数据排布示意如下：



## 2.1.5 输出音频

AFE 的输出音频为单通道数据：

- 语音识别场景：在 WakeNet 开启的情况下，输出有目标人声的单通道数据
- 语音通话场景：输出信噪比更高的单通道数据

### 2.1.6 使能唤醒词识别 WakeNet

在进行 AFE 声学前端处理时，用户可选择是否使能 *WakeNet* 进行唤醒词识别。

当用户在唤醒后需要进行其他操作，比如离线或在线语音识别，这时候可以暂停 WakeNet 的运行，从而减轻 CPU 的资源消耗。此时，仅需调用 `disable_wakenet()`，进入 Bypass 模式。

当后续应用结束后又可以调用 `enable_wakenet()` 再次使能 WakeNet。

ESP32 芯片只支持一个唤醒词，不支持唤醒词切换。

### 2.1.7 使能回声消除算法 AEC

AEC 的使用和 WakeNet 相似，用户可以根据自己的需求来停止或开启 AEC。

- 停止 AEC
 

```
afe->disable_aec(afe_data);
```
- 开启 AEC
 

```
afe->enable_aec(afe_data);
```

### 2.1.8 资源消耗

有关本模型的资源消耗情况，请见[资源消耗](#)。

## 2.2 乐鑫麦克风设计指南

本指南基于乐鑫的 ESP32 系列语音开发板，对于整机 mic 设计做出要求。

### 2.2.1 麦克风电器性能推荐

- 麦克类型：全向型 MEMS 麦克风
- 灵敏度
  - 1 Pa 声压下模拟麦灵敏度不低于 -38 dBV，数字麦灵敏度要求不低于 -26 dB。
  - 公差控制在  $\pm 2$  dB，对于麦克阵列推荐采用  $\pm 1$  dB 公差。
- 信噪比 (SNR)
  - 信噪比：信噪比不低于 62 dB，推荐 > 64 dB
  - 频率响应在 50~16 KHz 范围内的波动在  $\pm 3$  dB 之内
  - 麦克风的 PSRR 应大于 55 dB

### 2.2.2 麦克风结构设计建议

- 麦克孔孔径或宽度推荐大于 1 mm，拾音管道尽量短，腔体尽可能小，保证麦克和结构组件配合的谐振频率在 9 KHz 以上。
- 拾音孔深度和直径比小于 2:1，壳体厚度推荐 1 mm，如果壳体过厚，需增大开孔面积。
- 麦克孔上需通过防尘网进行保护。
- 麦克风与设备外壳之间必须加硅胶套或泡棉等进行密封和防震，需进行过盈配合设计，以保证麦克的密封性。
- 麦克孔不能被遮挡，底部拾音的麦克风需结构上增加底部凸起，保证麦克风与桌面等平面有一定间隙。
- 麦克需远离喇叭等会产生噪音或振动的物体摆放，且与喇叭音腔之间通过橡胶垫等隔离缓冲。

### 2.2.3 麦克阵列设计建议

客户可采用双麦克或三麦克方案：

- 双麦克方案：2 个麦克风之间间距保持 4~6.5 cm，连接 2 个麦克风的轴线应平行于水平线，且 2 个麦克的中心尽量靠近产品水平方向的中心。
- 三麦克方案：3 个麦克风等间距并且成正圆分布（夹角互成 120 度），间距要求 4~6.5 cm。

在选择阵列中的麦克风时，有如下注意事项：

- 麦克类型：全向型硅麦，推荐同一个阵列内的麦克应使用同一厂家的同一型号，不建议混用。
- 灵敏度：麦克阵列中各麦克灵敏度差异在 3 dB 之内。
- 相位差：多麦克阵列中麦克之间的相位差控制在  $10^\circ$  以内。
- 麦克阵列中各麦克的结构设计，推荐采用相同的设计，以保证结构设计的一致性。

### 2.2.4 麦克风结构密封性建议

用橡皮泥等材料封堵麦克拾音孔，密封前后麦克风采集信号的幅度衰减 25 dB 为合格，推荐 30 dB。测试方法：

1. 麦克风正上方 0.5 米处，播放白噪声，麦克风处音量 90 dB。
2. 使用麦克风阵列录制 10s 以上，存储为录音文件 A。
3. 用橡皮泥等材料封堵麦克拾音孔，使用麦克风阵列录制 10s 以上，存储为录音文件 B。
4. 对比两个文件的频谱，需保证 100~8 KHz 频段内整体衰减 25dB 以上。

## 2.2.5 回声参考信号设计建议

- 回声参考信号推荐尽量靠近喇叭侧，推荐从 DA 后级 PA 前级回采。
- 扬声器音量最大时，输入到麦克的回声参考信号不能有饱和失真，最大音量下喇叭功放输出 THD 满足 100 Hz 小于 10%，200 Hz 小于 6%，350 Hz 以上频率，小于 3%。
- 扬声器音量最大时，麦克处拾音的声压不超过 102 dB (1KHz)。
- 回声参考信号电压不超过 ADC 的最大允许输入电压，电压过高需增加衰减电路。
- 从 D 类功放输出引参考回声信号需增加低通滤波器，滤波器的截止频率推荐 > 22 KHz。
- 音量最大播放时，回采信号峰值 -3 到 -5 dB。

## 2.2.6 麦克风阵列一致性验证

要求各个麦克风采样信号幅度相差小于 3 dB，测试方法：

1. 麦克风正上方 0.5 米处，播放白噪声，麦克风处音量 90 dB。
2. 使用麦克风阵列录制 10s 以上，查看各 mic 录音幅度和音频采样率是否一致。



## Chapter 3

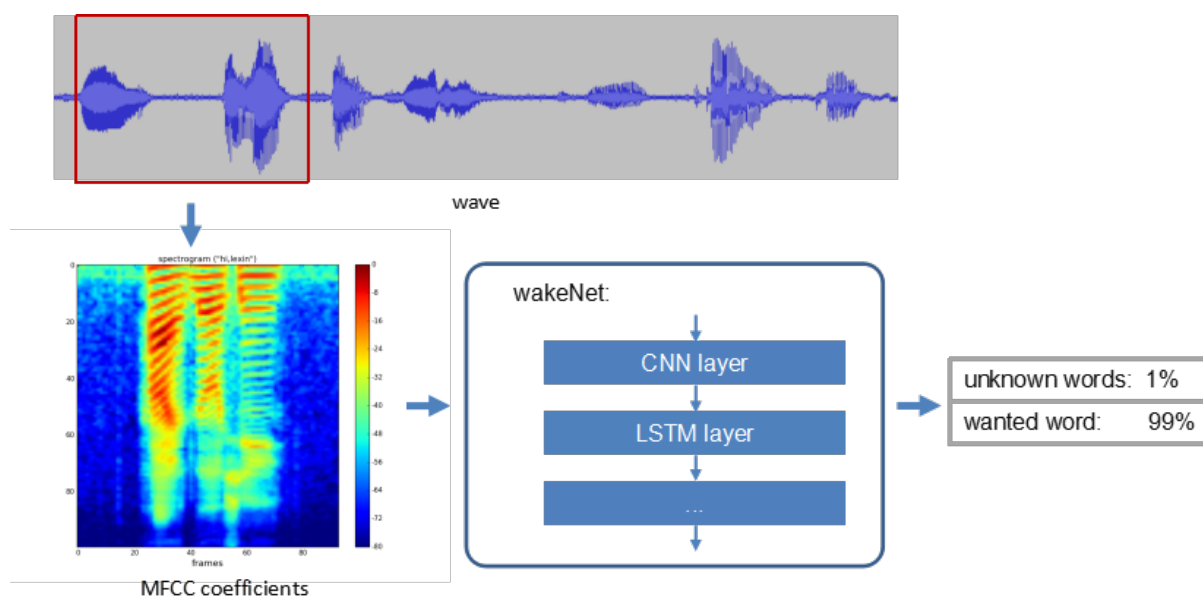
# 唤醒词

### 3.1 WakeNet 唤醒词模型

WakeNet 是一个基于神经网络，为低功耗嵌入式 MCU 设计的唤醒词模型，目前支持 5 个以内的唤醒词识别。

#### 3.1.1 概述

WakeNet 的流程图如下：



- **语音特征 (Speech Feature)** 我们使用 **MFCC** 方法提取语音频谱特征。输入的音频文件采样率为 16 KHz，单声道，编码方式为 signed 16-bit。每帧窗宽和步长均为 30 ms。
- **神经网络 (Neural Network)** 神经网络结构已经更新到第 9 版，其中：
  - WakeNet1、WakeNet2、WakeNet3、WakeNet4、WakeNet6 and WakeNet7 已经停止使用。
  - WakeNet5 应用于 ESP32 芯片。
  - WakeNet8 和 WakeNet9 应用于 ESP32-S3 芯片，模型基于 [Dilated Convolution](#) 结构。

注意，WakeNet5、WakeNet5X2 和 WakeNet5X3 的网络结构一致，但是 WakeNet5X2 和 WakeNet5X3 的参数比 WakeNet5 要多。请参考[资源消耗](#) 来获取更多细节。





- **Keyword Trigger Method** 对连续的音频流，为准确判断关键词的触发，我们通过计算若干帧内识别结果的平均值  $M$ ，来判断是否触发。当  $M$  大于指定阈值，则发出触发的命令。

以下表格展示在不同芯片上的模型支持：

Chip	ESP32			ESP32S3			
	WakeNet 5			WakeNet 8		WakeNet 9	
	WakeNet 5	WakeNet 5X2	WakeNet 5X3	Q16	Q8	Q16	Q8
Hi, Lexin	√	√	√				√
nihaoxiaozhi	√		√				√
nihaoxiaoxin			√				
xiaoaotongxue							√
Alexa				√			√
Hi, ESP							√
Customized word							√

### 3.1.2 WakeNet 的使用

- WakeNet 模型选择  
WakeNet 模型选择请参考 [flash model 介绍](#)。  
自定义的唤醒词，请参考 [乐鑫语音唤醒词定制流程](#)。
- WakeNet 模型运行  
WakeNet 目前包含在语音前端算法 AFE 中，默认为运行状态，并将识别结果通过 AFE fetch 接口返回。  
如果用户无需使用 WakeNet 唤醒，请在 AFE 配置时选择：

```
afe_config.wakenet_init = False.
```

如果用户想临时关闭/打开 WakeNet，请在运行过程中调用：

```
afe_handle->disable_wakenet (afe_data)
afe_handle->enable_wakenet (afe_data)
```

### 3.1.3 资源消耗

有关本模型的资源消耗情况，请见[资源消耗](#)。

## 3.2 乐鑫语音唤醒方案客户定制流程

### 3.2.1 唤醒词定制服务

乐鑫提供 **语音唤醒词定制服务**，详情如下：

- “HI 乐鑫”，“你好小鑫”等官方开放的唤醒词，客户可直接商用
  - 完整列表可见[乐鑫免费商用唤醒词](#)
  - 同时，乐鑫会逐渐开放更多可免费商用的唤醒词
- 除官方开放的唤醒词，乐鑫还可为客户提供 **唤醒词定制服务**，主要分如下两种情况：
  - 客户提供唤醒词语料
    - 需要提供大于 2 万条合格的语料，具体语料需求见[训练语料要求](#)
    - 语料提供给乐鑫后，需要 2~3 周进行模型训练及调优
    - 根据量级收取少量模型定制费用
  - 如果客户不提供唤醒词语料
    - 所有训练语料由乐鑫采集提供
    - 乐鑫需要一定时间收集语料，具体需要分别讨论；语料准备好后，需要 2~3 周进行模型训练及调优
    - 根据量级收取少量模型定制费用，语料采集费用另算
  - 定制的具体时间和费用取决于 **唤醒词定制的数量**以及 **产品量产数量**，详情请联系 [乐鑫销售人员](#)。
- 对于乐鑫唤醒词模型：
  - 目前单个模型最多支持 5 个及以内的唤醒词识别
  - 每个唤醒词通常由 3-6 音节组成，比如“hi 乐鑫”，“Alexa”，“小爱同学”，“你好天猫”等
  - 可多个唤醒模型一起使用，具体需根据客户应用的资源消耗确定
  - 更多详情，请见[WakeNet 唤醒词模型](#)

#### 训练语料要求

客户可自备训练语料或向第三方采购，对于语料有以下要求：

- 语料音频格式要求
  - 采样率 (sample rate): 16 KHz
  - 编码 (encoding): 16-bit signed int
  - 通道数 (channel): mono
  - 格式: wav
- 语料采集要求
  - 采样人数: 最好样本可以大于 500 人，其中男女，年龄分布均衡，儿童不小于 100 人
  - 采样环境: 环境噪声低 (< 40 dB)，建议在语音室等专业环境下录制
  - 录制设备: 高保真麦克风
  - 录制场景:
    - \* 距离麦克风 1 m 处每人录制 15 遍，其中 5 遍快语速，5 遍正常语速，5 遍慢语速；
    - \* 距离麦克风 3 m 处每人录制 15 遍，其中 5 遍快语速，5 遍正常语速，5 遍慢语速
  - 样本命名需体现样本信息：如 female\_age\_fast\_id.wav 或有单独表格记录每个样本的年龄，性别等信息

### 3.2.2 硬件设计与测试服务

语音唤醒效果与硬件设计以及腔体结构有很大关系。因此，请认真阅读以下内容：

1. 硬件设计要求
  - 各类语音音箱类设计：乐鑫可提供 **原理图 / PCB** 等设计参考，客户可以根据自身具体需求设计修改，设计完毕后，乐鑫还可提供审阅服务，避免常见设计问题。
  - 腔体结构：建议有专门的声学人员参与设计，乐鑫不提供 **ID** 设计类参考，客户可参考市面上的主流音箱腔体设计，例如天猫精灵、小度音箱、谷歌音箱等。
2. 硬件设计好后，客户可通过以下简单测试，验证硬件设计效果（下列测试都是基于语音室环境，客户可以根据自身测试环境做调整）
  - 录音测试，验证 mic、codec 录音增益以及失真情况
    - 音源 90 dB，距离 0.1 m 播放样本，调节增益，保证录音样本不饱和
    - 使用扫频文件 (0~20 KHz)，使用 16 KHz 采样率录音，音频不会出现明显频率混叠
    - 录制 100 个语音样本，使用公开的云端语音识别端口识别，识别率达到指定标准
  - 播音测试，验证功率放大器 (PA)、喇叭的失真情况
    - 测试 PA 功率 @1% 总谐波失真 (THD)
  - 语音算法测试，验证 AEC、BFM、NS 效果
    - 首先需要注意下参考信号延时，不同的 AEC 算法有不同的要求
    - 以实际产品场景为测试指标，例如 mic 播放 85DB-90DB 大梦想家.wav，设备回采
    - 保存回声参考信号、回声消除后的信号分析，对比查看 AEC、BFM、NS 等效果
  - DSP 性能测试，验证 DSP 参数是否合适，同时尽可能减少 DSP 算法中的非线性失真
    - 降噪 (Noise Suppression) 算法性能测试
    - 回声消除 (Acoustic Echo Cancellation) 算法性能测试
    - 语音增强 (Speech Enhancement) 算法性能测试
3. 硬件设计完毕后，**可寄送** 1-2 台硬件至乐鑫，乐鑫会基于客户整机做唤醒词性能调优。

## Chapter 4

# 命令词

### 4.1 MultiNet 命令词识别模型

MultiNet 是为了在 ESP32 系列上离线实现多命令词识别而设计的轻量化模型，目前支持 200 个以内的自定义命令词识别。

- 支持中文命令词识别
- 支持用户自定义命令词
- 支持运行过程中增加/删除/修改命令词语
- 最多支持 200 个命令词
- 支持单次识别和连续识别两种模式
- 轻量化，低资源消耗
- 低延时，延时 500 ms 内
- 模型单独分区，支持用户应用 OTA

MultiNet 输入为经过前端语音算法（AFE）处理过的音频（格式为 16 KHz，16 bit，单声道）。通过对音频进行识别，则可以对应到相应的汉字或单词。

请参考 [Models Benchmark](#) 去查看当前不同芯片支持的模型。

用户选择不同的模型的方法请参考 [模型加载](#)。

---

**备注：**其中以 Q8 结尾的模型代表模型的 8 bit 版本，表明该模型更加轻量化。

---

### 4.2 命令词识别原理

命令词识别原理如下图所示：

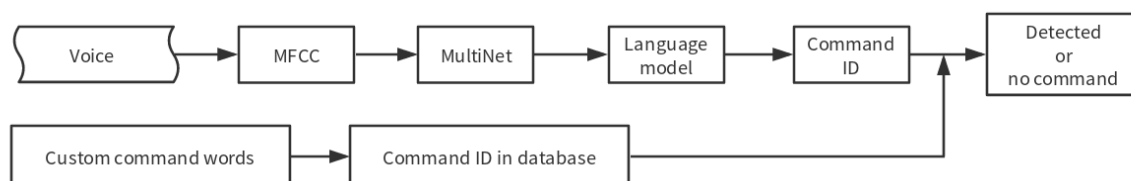


图 1: speech\_command-recognition-system

## 4.3 命令词格式要求

不同版本的 MultiNet 命令词格式不同。命令词需要满足特定的格式，具体如下：

## 4.4 自定义命令词方法

**备注：**不支持中英文混合

不能含有阿拉伯数字和特殊字符

英语自定义命令词方法请参考英文文档

MultiNet 支持多种且灵活的命令词设置方式，可通过在线或离线方法设置命令词，还允许随时动态增加/删除/修改命令词。

MultiNet5 和 MultiNet6 使用汉语拼音作为基本识别单元。比如“打开空调”，应该写成“da kai kong tiao”，请使用以下工具将汉字转为拼音：[tool/multinet\\_pinyin.py](#)。

### 4.4.1 MultiNet6 定义方法：

- 中文通过修改 `model/multinet_model/fst/commands_cn.txt`  
格式如下，第一个数字代表 `command id`，后面为指令的中文拼音，两者由空格隔开，拼音间也由空格隔开，`Command id` 不能为 0

```
# command_id command_sentence
1 da kai kong tiao
2 guan bi kong tiao
```

### 4.4.2 MultiNet5 定义方法：

- 通过 `menuconfig`
  - `idf.py menuconfig > ESP Speech Recognition > Add Chinese speech commands/Add English speech commands`，添加命令词。具体也可参考 `ESP-Skainet` 中的 `example`。  
注意，单个 `Command ID` 可以支持多个短语，比如“打开空调”和“开空调”表示的意义相同，则可以将其写在同一个 `Command ID` 对应的词条中，用英文字符“`,`”隔开相邻词条（“`,`”前后无需空格）。
  - 在代码里调用以下 API：

```
/**
 * @brief Update the speech commands of MultiNet by menuconfig
 *
 * @param multinet          The multinet handle
 *
 * @param model_data       The model object to query
 *
 * @param language         The language of MultiNet
 *
 * @return
 *   - ESP_OK              Success
 *   - ESP_ERR_INVALID_STATE  Fail
 */
esp_err_t esp_mn_commands_update_from_sdkconfig(esp_mn_iface_t_
↪ *multinet, const model_iface_data_t *model_data);
```

```
(Top) → ESP Speech Recognition → Add Chinese speech commands
(da kai kong tiao,kai kong tiao) ID0
(guan bi kong tiao) ID1
(zeng da feng su) ID2
(jian xiao feng su) ID3
(sheng gao yi du) ID4
(jiang di yi du) ID5
(zhi re mo shi) ID6
(zhi leng mo shi) ID7
(song feng mo shi) ID8
(jie neng mo shi) ID9
(chu shi mo shi,chu diao shi qi) ID10
(jian kang mo shi) ID11
(shui mian mo shi) ID12
(da kai lan ya) ID13
(guan bi lan ya) ID14
(kai shi bo fang) ID15
(zan ting bo fang) ID16
(ding shi yi xiao shi) ID17
(da kai dian deng) ID18
(guan bi dian deng) ID19
() ID20
() ID21
() ID22
```

图 2: menuconfig\_add\_speech\_commands

### 4.4.3 通过调用 API 修改

指令还可以通过调用 API 修改，这种方法对于 MultiNet5 和 MultiNet6 都适用。

- 应用新的修改操作，所有添加、移除、修改及清空操作在调用后才会被应用。

```
/**
 * @brief Update the speech commands of MultiNet
 *
 * @Warning: Must be used after [add/remove/modify/clear] function,
 * otherwise the language model of multinet can not be
 * updated.
 *
 * @return
 * - NULL Success
 * - others The list of error phrase which can not be
 * parsed by multinet.
 */
esp_mn_error_t *esp_mn_commands_update();
```

- 添加一条新指令，如果指令格式不正确则返回 ESP\_ERR\_INVALID\_STATE。

```
/**
 * @brief Add one speech commands with command string and command ID
 *
 * @param command_id The command ID
 * @param string The command string of the speech commands
 *
 * @return
 * - ESP_OK Success
 * - ESP_ERR_INVALID_STATE Fail
 */
esp_err_t esp_mn_commands_add(int command_id, char *string);
```

- 移除一条指令，如果该指令不存在则返回 ESP\_ERR\_INVALID\_STATE。

```
/**
```

(下页继续)

(续上页)

```

* @brief Remove one speech commands by command string
*
* @param string The command string of the speech commands
*
* @return
*   - ESP_OK           Success
*   - ESP_ERR_INVALID_STATE  Fail
*/
esp_err_t esp_mn_commands_remove(char *string);

```

- 修改一条指令，如果该指令不存在则返回 ESP\_ERR\_INVALID\_STATE。

```

/**
* @brief Modify one speech commands with new command string
*
* @param old_string The old command string of the speech commands
* @param new_string The new command string of the speech commands
*
* @return
*   - ESP_OK           Success
*   - ESP_ERR_INVALID_STATE  Fail
*/
esp_err_t esp_mn_commands_modify(char *old_string, char *new_string);

```

- 清空所有指令。

```

/**
* @brief Clear all speech commands in linked list
*
* @return
*   - ESP_OK           Success
*   - ESP_ERR_INVALID_STATE  Fail
*/
esp_err_t esp_mn_commands_clear(void);

```

- 打印缓存的指令，只有当调用 esp\_mn\_commands\_update() 缓存指令才会被应用。

```

/**
* @brief Print all commands in linked list.
*/
void esp_mn_commands_print(void);

```

- 打印当前已经被应用的指令。

```

/**
* @brief Print all commands in linked list.
*/
void esp_mn_active_commands_print(void);

```

## 4.5 MultiNet 的使用

MultiNet 命令词识别建议和 ESP-SR 中的 AFE 声学算法模块一起运行，具体请参考[AFE 介绍及使用](#)。当用户配置完成 AFE 后，请按照以下步骤配置和运行 MultiNet。

### 4.5.1 MultiNet 初始化

- 模型加载与初始化，请参考[模型加载](#)
- 设置命令词，请参考[命令词格式要求](#)

## 4.5.2 MultiNet 运行

当用户开启 AFE 且使能 WakeNet 后，则可以运行 MultiNet。但需要注意以下几点要求：

- 传入帧长和 AFE fetch 帧长长度相等
- 支持音频格式为 16 KHz, 16 bit, 单通道。AFE fetch 拿到的数据也为这个格式
- 确定需要传入 MultiNet 的帧长

```
int mu_chunksize = multinet->get_samp_chunksize(model_data);
```

mu\_chunksize 是需要传入 MultiNet 的每帧音频的 short 型点数，这个大小和 AFE 中 fetch 的每帧数据点数完全一致。

- MultiNet 识别  
我们将 AFE 实时 fetch 到的数据送入以下 API：

```
esp_mn_state_t mn_state = multinet->detect(model_data, buff);
```

buff 的长度为 mu\_chunksize \* sizeof(int16\_t)。

## 4.5.3 MultiNet 识别结果

命令行识别必须和唤醒搭配使用，当唤醒后可以运行命令词的检测。

命令行模型在运行时，会实时返回当前帧的识别状态 mn\_state，目前分为以下几种识别状态：

- ESP\_MN\_STATE\_DETECTING  
该状态表示目前正在识别中，还未识别到目标命令词。
- ESP\_MN\_STATE\_DETECTED  
该状态表示目前识别到了目标命令词，此时用户可以调用 get\_results 接口获取识别结果。

```
esp_mn_results_t *mn_result = multinet->get_results(model_data);
```

识别结果的信息存储在 get\_result API 的返回值中，返回值的数据类型如下：

```
typedef struct{
    esp_mn_state_t state;
    int num; // The number of phrase in list, num<=5.
    ↪When num=0, no phrase is recognized.
    int phrase_id[ESP_MN_RESULT_MAX_NUM]; // The list of phrase
    ↪id.
    float prob[ESP_MN_RESULT_MAX_NUM]; // The list of
    ↪probability.
} esp_mn_results_t;
```

其中，

- state 为当前识别的状态
- num 表示识别到的词条数目，num <= 5，即最多返回 5 个候选结果
- phrase\_id 表示识别到的词条对应的 Phrase ID
- prob 表示识别到的词条识别概率，从大到小依次排列

用户可以使用 phrase\_id[0] 和 prob[0] 拿到概率最高的识别结果。

- ESP\_MN\_STATE\_TIMEOUT  
该状态表示长时间未检测到命令词，自动退出。等待下次唤醒。

单次识别模式和连续识别模式：当命令词识别返回状态为 ESP\_MN\_STATE\_DETECTED 时退出命令词识别，则为单次识别模式；当命令词识别返回状态为 ESP\_MN\_STATE\_TIMEOUT 时退出命令词识别，则为连续识别模式；

## 4.6 资源消耗

有关本模型的资源消耗情况，请见[资源消耗](#)。





## Chapter 5

# TTS 语音合成模型

乐鑫 TTS 语音合成模型是一个为嵌入式系统设计的轻量化语音合成系统，具有如下主要特性：

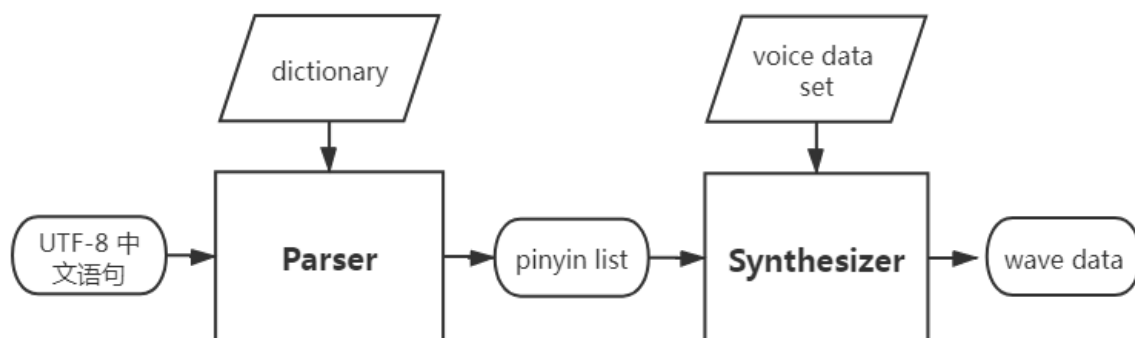
- 目前 仅支持中文
- 输入文本采用 UTF-8 编码
- 输出格式采用流输出，可减少延时
- 多音词发音自动识别
- 可调节合成语速
- 数字播报优化
- 自定义声音集（敬请期待）

### 5.1 简介

乐鑫 TTS 的当前版本基于拼接法，主要组成部分包括：

- 解析器 (Parser)：根据字典与语法规则，将输入文本（采用 UTF-8 编码）转换为拼音列表。
- 合成器 (Synthesizer)：根据解析器输出的拼音列表，结合预定义的声音集，合成波形文件。默认输出格式为：单声道，16 bit @ 16000Hz。

系统框图如下：



### 5.2 简单示例

- [esp-tts/samples/xiaoxin\\_speed1.wav](#) (voice=xiaoxin, speed=1): 欢迎使用乐鑫语音合成，支付宝收款 72.1 元，微信收款 643.12 元，扫码收款 5489.54 元
- [esp-tts/samples/S2\\_xiaole\\_speed2.wav](#) (voice=xiaole, speed=2): 支付宝收款 1111.11 元

## 5.3 编程指南

```

#include "esp_tts.h"
#include "esp_tts_voice_female.h"
#include "esp_partition.h"

/** 1. create esp tts handle */

// initial voice set from separate voice data partition

const esp_partition_t* part=esp_partition_find_first(ESP_PARTITION_TYPE_DATA, ESP_
↳PARTITION_SUBTYPE_DATA_FAT, "voice_data");
if (part==0) printf("Couldn't find voice data partition!\n");
spi_flash_mmap_handle_t mmap;
uint16_t* voicedata;
esp_err_t err=esp_partition_mmap(part, 0, part->size, SPI_FLASH_MMAP_DATA, (const_
↳void*)&voicedata, &mmap);
esp_tts_voice_t *voice=esp_tts_voice_set_init(&esp_tts_voice_template, voicedata);

// 2. parse text and synthesis wave data
char *text="欢迎使用乐鑫语音合成";
if (esp_tts_parse_chinese(tts_handle, text)) { // parse text into pinyin list
    int len[1]={0};
    do {
        short *data=esp_tts_stream_play(tts_handle, len, 4); // streaming synthesis
        i2s_audio_play(data, len[0]*2, portMAX_DELAY); // i2s output
    } while(len[0]>0);
    i2s_zero_dma_buffer(0);
}

```

更多参考，请前往 [esp-tts/esp\\_tts\\_chinese/include/esp\\_tts.h](#) 查看 API 定义，或参考 ESP-Skainet 中 [chinese\\_tts](#) 示例。

## 5.4 资源消耗

有关本模型的资源消耗情况，请见[资源消耗](#)。

## Chapter 6

# 模型加载

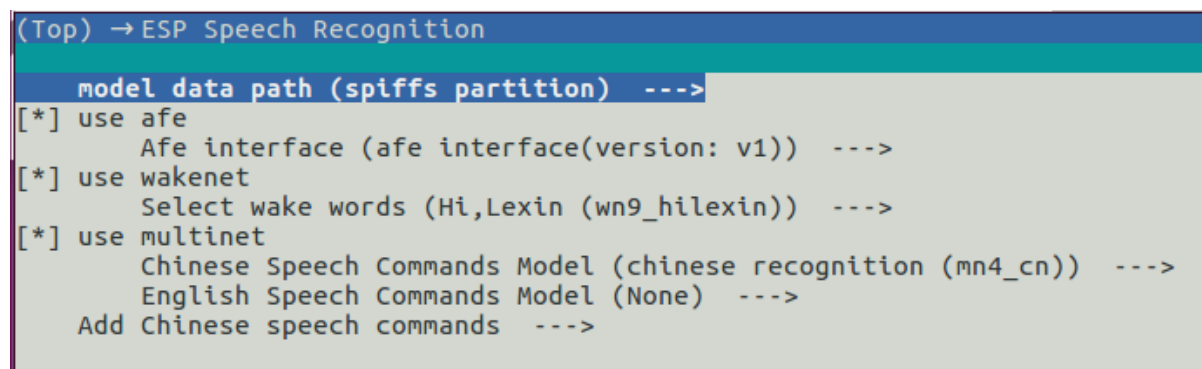
在人工智能行业中，模型是指一个系统或过程的数学表示。它用于基于输入数据做出预测或决策，有许多不同类型的模型，如决策树、神经网络和支持向量机，每种模型都有其优缺点。乐鑫也提供经过训练的 WakeNet 和 MultiNet 模型（数据模型见 [model](#)）。

使用模型前需先将其加载至你的项目，目前 ESP-SR 支持以下模型加载方式：

ESP32：从 Flash 中直接加载

### 6.1 配置方法

运行 `idf.py menuconfig` 进入 ESP Speech Recognition:



```
(Top) → ESP Speech Recognition
model data path (spiffs partition) --->
[*] use afe
    Afe interface (afe interface(version: v1)) --->
[*] use wakenet
    Select wake words (Hi, Lexin (wn9_hilexin)) --->
[*] use multinet
    Chinese Speech Commands Model (chinese recognition (mn4_cn)) --->
    English Speech Commands Model (None) --->
    Add Chinese speech commands --->
```

图 1: overview

#### 6.1.1 使用 AFE

此选项需要打开，用户无须修改，请保持默认配置。

#### 6.1.2 使用 WakeNet

此选项默认打开。当用户只使用 AEC 或者 BSS 等，而无须运行 WakeNet 或 MultiNet 时，请关闭此选项，这将会减小工程固件的大小。

根据 menuconfig 列表选择唤醒词模型，ESP Speech Recognition > Select wake words。括号中为唤醒词模型的名字，在代码中初始化 WakeNet 时需写入对应的名字。

```
(Top) → ESP Speech Recognition → use wakenet → Select wake words
( ) Alexa (wn8_alex)
(X) Hi, Lexin (wn9_hilexin)
( ) xiaoaotongxue (wn9_xiaoaotongxue)
( ) Alexa (wn9_alex)
( ) Hi, ESP (wn9_hiesp)
( ) customized word (wn9_customizedword)
( ) Load Multiple Wake Words
```

如果想加载多个唤醒词，以便在代码中进行唤醒词的切换，首选选择 Load Multiple Wake Words

```
(Top) → ESP Speech Recognition → use wakenet → Select wake words
( ) Alexa (wn8_alex)
( ) Hi, Lexin (wn9_hilexin)
( ) xiaoaotongxue (wn9_xiaoaotongxue)
( ) Alexa (wn9_alex)
( ) Hi, ESP (wn9_hiesp)
( ) customized word (wn9_customizedword)
(X) Load Multiple Wake Words
Load Multiple Wake Words --->
```

然后按照列表选择多个唤醒词：

```
(Top) → ESP Speech Recognition → use wakenet → Select wake words → Load Multiple Wake Words → Load Multiple Wake Words
Espressif IoT Development Framework Config
[*] Hi, Lexin (wn9_hilexin)
[*] xiaoaotongxue (wn9_xiaoaotongxue)
[*] Alexa (wn9_alex)
[ ] Hi, ESP (wn9_hiesp) (NEW)
```

**备注：**ESP32 不支持多唤醒词选项。

更多细节请参考 [WakeNet](#)。

### 6.1.3 使用 MultiNet

此选项默认打开。当用户只使用 WakeNet 或者其他算法模块时，请关闭此选项，将会在一些情况下减小工程固件的大小。

#### 中文命令词识别模型 (Chinese Speech Commands Model)

ESP32 芯片只支持中文命令词识别：

- None
- Chinese single recognition (MultiNet2)

用户按照需求自定义添加命令词，具体请参考 [MultiNet](#)。

## 6.2 模型使用

当用户完成以上的配置选择后，可参考 [ESP-Skainet](#) 应用层仓库中的介绍，进行初始化和使用。

这里主要介绍模型加载在用户工程中的代码实现，用户也可直接参考代码 [src/model\\_path.c](#)。

ESP32 仅支持从 Flash 中直接加载模型数据，因此代码中模型数据会自动按照地址从 Flash 中读取所需数据。为了和 ESP32-S3 进行兼容，ESP32 代码中模型的初始化方法与 ESP32-S3 相同。

## 6.2.1 模型数据存储在 Flash

1. 编写分区表:

```
model, data, data, , SIZE,
```

其中 **SIZE** 可以参考在用户使用 `idf.py build` 编译时的推荐大小, 例如: Recommended model partition size: 500K。

2. 初始化 `partition` 分区: 用户可以直接调用提供的 `esp_srmodel_init(partition_label)` API 来获取 `partition` 中的模型。
  - `partition_label`: 为 `partition table` 中定义的模型的分区, 需要和上述函数的入参保持一致

完成上述配置后, 模型会在工程编译完成后自动生成 `srmodels.bin`, 并在用户调用 `idf.py flash` 时烧写到指定分区。



## Chapter 7

# 性能测试结果

### 7.1 AFE

#### 7.1.1 资源消耗

Algorithm Type	RAM	Average cpu loading(compute with 2 cores)	Frame Length
AEC(HIGH_PERF)	114 KB	11%	32 ms
NS	27 KB	5%	10 ms
AFE Layer	73 KB		

### 7.2 WakeNet

#### 7.2.1 资源消耗

Model Type	Parameter Num	RAM	Average Running Time per Frame	Frame Length
Quantised WakeNet5	41 K	15 KB	5.5 ms	30 ms
Quantised Wak-eNet5X2	165 K	20 KB	10.5 ms	30 ms
Quantised Wak-eNet5X3	371 K	24 KB	18 ms	30 ms

#### 7.2.2 性能测试

Dis-tance	Quiet	Stationary Noise (SNR = 4 dB)	Speech Noise (SNR = 4 dB)	AEC I nterruption (-10 dB)
1 m	98%	96%	94%	96%
3 m	98%	96%	94%	94%

误触发率：12 小时 1 次

备注：我们在测试中使用了 ESP32-S3-Korvo V4.0 开发板和 WakeNet9(Alexa) 模型。



## 7.3 MultiNet

### 7.3.1 资源消耗

Model Type	Internal RAM	PSRAM	Average Running Time per Frame	Frame Length
MultiNet 2	13.3 KB	9KB	38 ms	30 ms

### 7.3.2 Word Error Rate 性能测试

Model Type	aishell test
MultiNet 5_cn	9.5%
MultiNet 6_cn	5.2%

备注：中文使用没有声调的拼音单元去计算 WER。

### 7.3.3 Speech Commands 性能测试 (空调控制场景)

Model Type	Dis- tance	Quiet	Stationary Noise (SNR=5~10dB dB)	Speech Noise (SNR=5~10dB dB)
MultiNet 5_cn	3 m	88.9%	66.1%	67.5%
MultiNet 6_cn	3 m	98.8%	88.3%	88.0%
MultiNet 6_cn_ac	3 m	97.1%	95.1%	96.8%

备注：MultiNet6\_cn\_ac 在空调场景数据集上进行了进一步的微调，所以在空调控制场景具有更好的性能。

## 7.4 TTS

### 7.4.1 资源消耗

Flash image size: 2.2 MB

RAM runtime: 20 KB

### 7.4.2 性能测试

CPU 负载测试 (ESP32 @240 MHz):

Speech Rate	0	1	2	3	4	5
Times faster than real time	4.5	3.2	2.9	2.5	2.2	1.8

## 7.5 NSNET

### 7.5.1 性能测试

### 7.5.2 数据集: array\_onemic\_nnoise\_20230608(按照亚马逊声学认证标准录制测试集)

	dnsmos
nsnet1	2.4
nsnet2	2.71



## Chapter 8

# 测试方法与测试报告

为了保证 DUT 的性能，可通过测试确定 DUT 在以下方面的表现：

- 唤醒率
- 识别率
- 误唤醒率
- 唤醒打断率
- 响应时间

### 8.1 测试场景要求

以上测试需在合适的测试房间中进行，测试房间应满足如下要求：

- **房间大小**
  - 面积：不小于 4 m \* 3.2 m
  - 高度：不低于 2.3 m
- **房间装饰**
  - 地板需配有地毯，天花板需配备常见声学阻尼材料，墙面应有 1 到 2 面墙上挂有窗帘，防止强反射。
  - 房间混响 (RT60) 3 在 [125, 8k] 范围内，要满足 0.2-0.7s 的要求。
  - 不要使用消音室。
- **环境底噪**：必须 < 35 dBA，最好 < 30 dBA
- **温度和湿度要求**：20±10°C，相对湿度为 50%±20%。
- **DUT、外部噪声、人声的放置**：
  - DUT、外部噪声、人声的具体位置和相对位置应根据 DUT 的实际应用场景进行安排。

---

**备注：**RT60、环境底噪和 DUT、外部噪声、人声的放置应在所有测试中保持不变。

---

### 8.2 测试案例设计

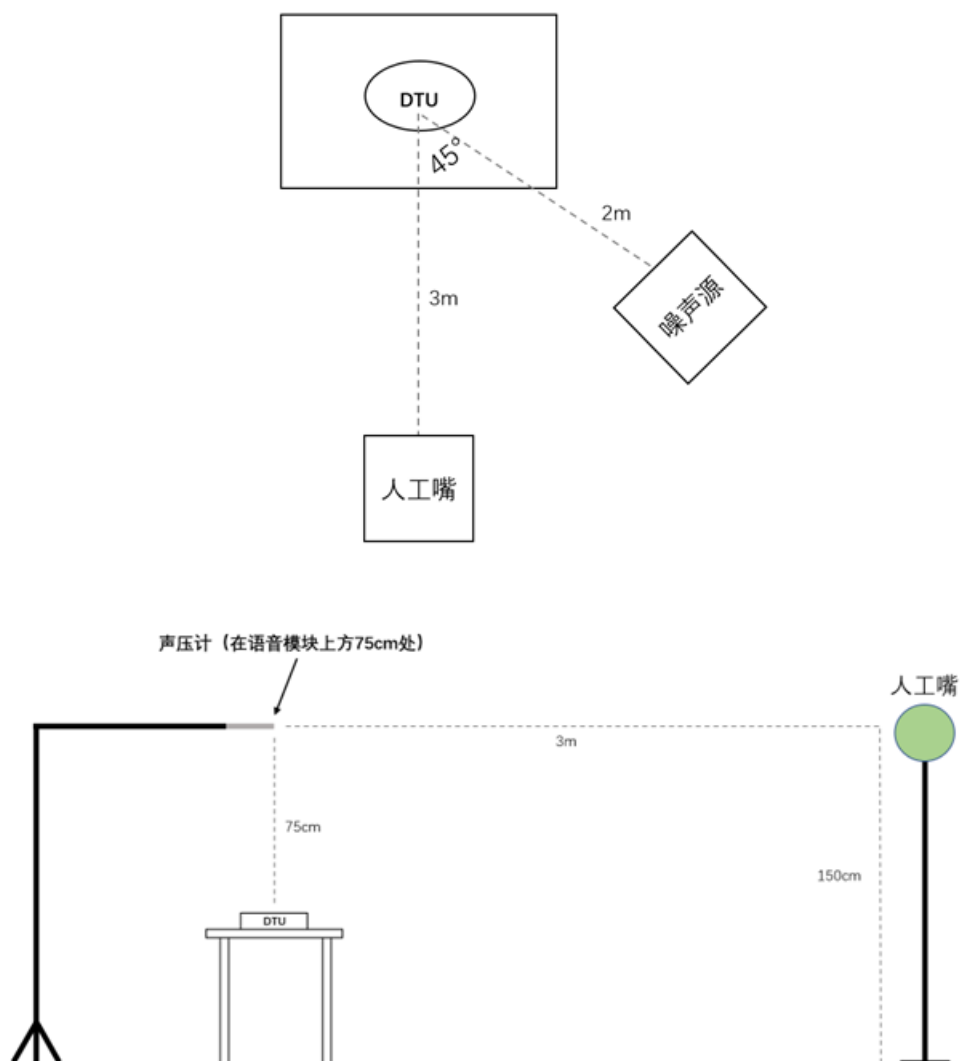
在设计测试案例时，建议按照产品的实际应用场景，考虑 **以下部分或全部参数**：

- **不同噪声源**
  - 白噪声
  - 人类噪声
  - 音乐
  - 新闻
  - ……

- 必要时还可以增加多噪声源的测试场景。
- 不同噪声分贝
  - < 35 dBA
  - 45 dBA
  - 55 dBA
  - 65 dBA
- 不同人声分贝
  - 54 dBA
  - 59 dBA
  - 64 dBA
- 不同 SNR
  - 9 dBA
  - 4 dBA
  - -1 dBA

### 8.3 乐鑫测试与结果

在本章节描述的所有测试中，DUT、外部噪声、人声的具体位置和相对位置如下所述。



具体来说，

- DUT 距离地面高度 0.75 米
- 人工嘴（人声源）距离地面高度 1.5 米，距离 DUT 直线水平距离 3 米
- 噪声源在相对人工嘴 45° 角处，距离地面高度 1.2 米，距离 DUT 直线水平距离 2 米
- 声压计位于 DUT 正上方 0.75 米

### 8.3.1 唤醒率测试

**唤醒率：**指当设备处于待唤醒状态时被唤醒成功的概率。

乐鑫唤醒率测试和结果

测试案例	噪声类型	噪声分贝	人声分贝	SNR	唤醒率
1	/	/	59 dBA	/	99%
2	白噪声	55 dBA	59 dBA	≥4 dBA	99%
3	人声噪声	55 dBA	59 dBA	≥4 dBA	99%

### 8.3.2 语音识别率测试

**语音识别率：**指当设备处于识别状态时，成功识别词表里包含的命令词的概率。

乐鑫语音识别率测试和结果

测试案例	噪声类型	噪声分贝	人声分贝	SNR	语音识别率
1	/	/	59 dBA	/	91.5%
2	白噪声	55 dBA	59 dBA	≥4 dBA	78.25%
3	人声噪声	55 dBA	59 dBA	≥4 dBA	82.77%

### 8.3.3 误唤醒率测试

**误唤醒率：**指设备在产品定义的应用场景下被非唤醒词成功唤醒的概率。

乐鑫误唤醒率测试和结果

测试案例	噪声类型	噪声分贝	测试时长	误唤醒次数
1	音乐	55 dBA	12 小时	1 次
2	新闻	55 dBA	12 小时	1 次

### 8.3.4 唤醒打断率测试

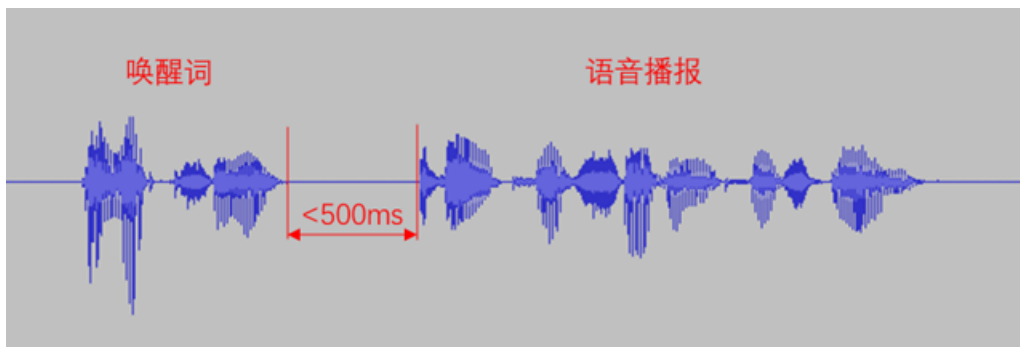
**唤醒打断率：**唤醒打断率是指设备有自噪时，即有 TTS3 播报或播放音频时，被唤醒成功的概率。对于有 AEC 功能的产品需要进行该测试。

乐鑫唤醒打断率测试和结果

测试案例	噪声类型	噪声 / 人声分贝	SNR	唤醒率	语音识别率
1	音乐	69 dBA / 59 dBA	≥10 dBA	100%	96%
2	TTS	69 dBA / 59 dBA	≥10 dBA	100%	96%

### 8.3.5 响应时间测试

**响应时间**：代表 DUT 响应语音命令的时间，具体测量为语音指令与播报之间的时间间隔（见下图）。



#### 乐鑫响应时间测试和结果

测试案例	噪声 / 人声分贝	SNR	响应时间
1	NA / 59 dBA	/	< 500 ms

## Chapter 9

# 术语表

### 9.1 通用术语

ESP-SR 仓库中的大多数术语均与 [乐鑫 ADF 音频应用开发框架](#) 共用，具体请访问 [ADF 中英术语库](#)。

### 9.2 特别术语

ESP-SR 仓库独有术语，请见下方。

语音用户界面 (Voice-User Interface, VUI)