

Voice-on-Spot Reference Design User Guide

Doc ID: VOSRDUG-7p00
Revision 7.0
July 2020

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

1. Table of Contents

1.	Table of Contents	2
2.	List of Tables	3
3.	List of Figures.....	4
4.	Document Revision History	6
5.	Overview	7
6.	Access & Licensing.....	7
7.	IDE Support.....	7
8.	Supported Hardware	8
9.	Project Directory Structure	11
10.	Project Configuration to Select Target Board and Desired Features	11
11.	Setting Other Project Options	16
12.	Building the SDK to Create a Custom Binary for Stand-alone Mode	18
13.	Operation with the Alexa App	19
14.	VoS SDK Tuning Guide	22
15.	AMVoS Profile with Common BLE Test App.....	26
16.	Building an OTA Image	28
17.	OTA image download	30
18.	Audio (Voice) Data Recording using RTT or AMU2S.....	32
19.	Debug Message Output.....	35

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

2. List of Tables

Table 1. Revision History	6
Table 2. Top-level Features of the VoS SDK	11
Table 3. Required Modules in the Four Base Targets.....	12
Table 4. Hardware Selection in Configuration File	14
Table 5. Feature Selection in the Configuration File	14
Table 6. Feature Configuration in the Configuration File.....	15
Table 7. Supported Features List.....	16
Table 8. LED indications of system status	19

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

3. List of Figures

Figure 1. Maya Stand-alone Button.....	8
Figure 2. Maya Board for OKW Enclosure	8
Figure 3. MikroBUS shield board with 3 slots	8
Figure 4. Top-mount Microphone MikroBUS board + MikroBUS shield board + Apollo family EVB	9
Figure 5. Bottom-mount Microphone MikroBUS board + MikroBUS shield board + Apollo family EVB	9
Figure 6. Bottom-mount Microphone MikroBUS board + MikroBUS shield board + Apollo family EVB ...	10
Figure 7. Dual-mode Microphone MikroBUS board + MikroBUS shield board + Apollo family EVB	10
Figure 8. Project target selection for each version	12
Figure 9. Audio chain and detection engine selection	13
Figure 10. KWD Included Files	15
Figure 11. Apollo2 EVB + EM9304 BLE shield	16
Figure 12. configUSE_ALEXA_QUAL_LAYOUT definition in am_vos_sys_config.h	17
Figure 13. AIC processing module selection in project window.....	17
Figure 14. Single MIC supported targets in project window.....	18
Figure 15. Devices Icon in Alexa App.....	20
Figure 16. Device Selection in Alexa App	20
Figure 17. Device Setup Complete in Alexa App	21
Figure 18. VoS DSPC/THF Flow Chart	22
Figure 19. VoS Light Flow Chart.....	22
Figure 20. am_pdm0_isr() in am_vos_isr.c File	23
Figure 21. DSPC Audio Weaver pre-processing.....	23
Figure 22. Input Data to DSPC AWE (Beam Former/SCNR) at am_spp_input_push() in am_vos_awe.c ..	23
Figure 23. Output data from DSPC AWE at am_spp_output_pop() in am_vos_awe.c	24
Figure 24. am_vos_stereo_to_mono_proc () in am_vos_audio.c File	24
Figure 25. Key Word Detection Check in am_vos_engine_process().....	24
Figure 26. Ring Buffer Status Check and Encode Audio Data in am_vos_codec_task()	25
Figure 27. Encoded Audio Data Size Check in am_vos_codec_task()	25
Figure 28. Disabling AMA Protocol.....	25
Figure 29. Building VoS without AMA Protocol.....	26
Figure 30. LightBlue Explorer	26
Figure 31. UUID in LightBlue Properties Page	26
Figure 32. Listen for Notifications in LightBlue	27
Figure 33. Streaming Data in LightBlue	27
Figure 34. OTA project selection on Apollo2.....	28
Figure 35. OTA image generated by Makefile for Apollo2	28
Figure 36. OTA image generated by Makefile for Apollo3	29
Figure 37. Error message due to non-existing make.exe	29
Figure 38. J-Flash Lite address setting for Apollo3	30
Figure 39. iTunes file manager	31
Figure 40. Ambiq OTA app scan result	31
Figure 41. RTT recorder definition	32
Figure 42. RTT recorder definition	32
Figure 43. Recorded data select	32

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

Figure 44. Disable codec and BLE when using RTT recorder	32
Figure 45. System path environment variable	32
Figure 46. Run RTT datalogger batch file	33
Figure 47. Run AMU2S audio recorder	33
Figure 48. configUSE_STDIO_PRINTF definition in am_vos_sys_config.h.....	35
Figure 49. configUSE_PRINTF_UART0 definition in am_vos_sys_config.h	35
Figure 50. Serial port setup	35
Figure 51. Terminal setup.....	35
Figure 52. UART print out using Tera Term.....	35
Figure 53. configUSE_STDIO_PRINTF definition in am_vos_sys_config.h.....	36
Figure 54. configUSE_PRINTF_SWO definition in am_vos_sys_config.h	36
Figure 55. SWO device and clock configuration.....	36
Figure 56. SWO debugging message print out	36

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

4. Document Revision History

Revision Number	Date	Description
1.0	April 2019	Document initial public release
2.0	June 2019	<ul style="list-style-type: none">- OTA supported in IAR / Keil for Apollo2, Apollo2 Blue and Maya- VoS Framework updated to v1.1.2
3.0	August 2019	<ul style="list-style-type: none">- Section 17 (Setting Other Project Options) added- Multiple sections: OPUS audio codec support added
4.0	Dec 2019	<ul style="list-style-type: none">- Cyberon, OVVP features added- LED indication, RTT recorder, Analog MIC description added
5.0	Mar 2020	<ul style="list-style-type: none">- Apollo 3 Plus supported- Retune DSP VS features added
6.0	May 2020	<ul style="list-style-type: none">- DSPC AWE updated to v8.x- Table 4 updated with new build option - USE_DMIC_MB3_VM3011
7.0	July 2020	<ul style="list-style-type: none">- Added Open AI Lab's AID feature- RTT / AMU2S recording guide added.

Table 1. Revision History

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

5. Overview

This document describes example builds in the Voice-on-SPOT (VoS) framework project that demonstrate Sound Pre-Processing (SPP), Key Word Detection (KWD) Engine and other features on the Ambiq Micro Apollo2, Arpollo2 Blue, Apollo3 Blue and Apollo3 Blue Plus MCUs. These examples support the following configurations and features:

- Inclusion or exclusion of audio and other functional features to create baseline and diagnostic builds.
- One or two PDM-driven digital microphones connected to Apollo MCU's stereo PDM interface.
- One or two analog microphones connected to Apollo MCU's ADC interface. (Apollo3 only)
- Capture of an audio buffer output over BLE to Amazon Alexa app for using Alexa Voice Service (AVS).
- Transfer of an audio buffer between Apollo family EVBs and over SEGGER RTT interface to a PC for the generation of a WAV file.
- Ambiq Micro's patented SPOT technology provides up to 10x lower active power consumption than standard Cortex M4 core.

6. Access & Licensing

Example software project requires NDAs and Developers Agreements. See [Project Configuration to Select Target Board and Desired Features](#) for descriptions of audio processing modules.

- **Production IP Licensing**
 - DSP Concepts, Sensory, Cyberon, Retune DSP, Open AI Lab, U-COMM and Ambiq Micro have agreed to a 3rd party software ecosystem licensing model to share IP.
- **Partner Contacts**
 - Ambiq Micro: Marc Miller (mmiller@ambiqmicro.com)
 - DSP Concepts: John Whitecar (jwhitecar@dspconcepts.com)
 - Sensory: Bernie Brafman (bbrafman@sensory.com)
 - Cyberon: Alex Liou (alexliou@cyberon.com.tw)
 - Return DSP: Christopher Welsh (chris@retune-dsp.com)
 - Open AI Lab: Jie Xie (jxie@openailab.com)
 - U-COMM: Eric Lo (el@u-comm.net)

7. IDE Support

VoS SDK is developed based on AmbiqSuite SDK v2.4.2

- **IAR Embedded Workbench IDE**
 - Developed and tested using IAR IDE version "ARM 8.32.3"
- **Keil uVision IDE**
 - uVision v5.27.1.0

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

8. Supported Hardware

- **Apollo3 based button board “Maya” and “Maya OKW” (with or without enclosure)**
 - 1 Vesper analog mic (bottom mounted) and 2 digital mics (top mounted)
 - Digital mic separation: 1cm
 - Configuration definition: ***USE_MAYA***

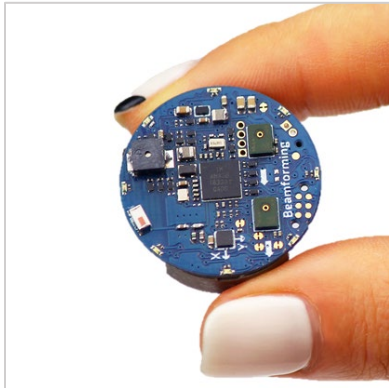


Figure 1. Maya Stand-alone Button

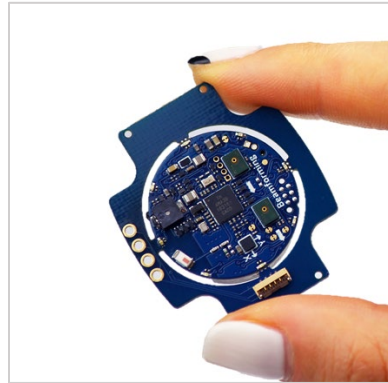


Figure 2. Maya Board for OKW Enclosure

- **MikroBUS Shield Board (AM_MBS_R20)**
 - MikroBUS Shield Board (3 slots – MB1 / MB2 / MB3)
 - Configuration definition: ***USE_MIKRO_DEV_SHIELD_REV1***

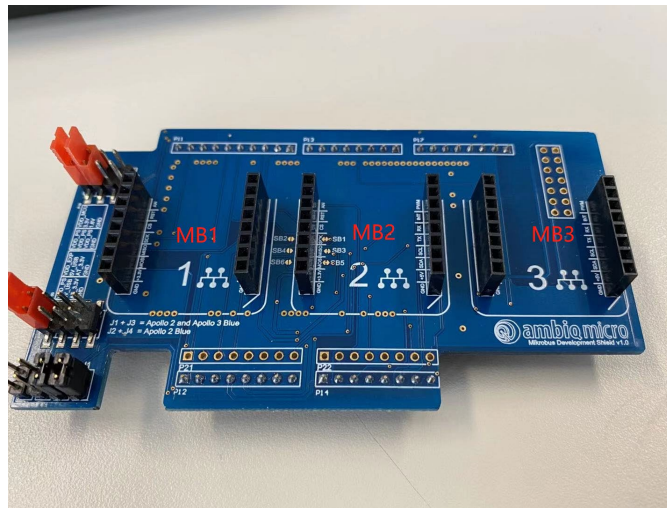


Figure 3. MikroBUS shield board with 3 slots

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- **Microphone MikroBUS board (AM-STD-CL)**
 - Selectable 1-mic or 2-mic configuration
 - If 2 mics, 1cm or 2cm spacing depending on SBx (solder bridge) configuration
 - Default is 2 top-mounted mics with 2cm spacing
 - Configuration definition: ***USE_DMIC_MB3 + USE_MIKRO_DEV_SHIELD_REV1***

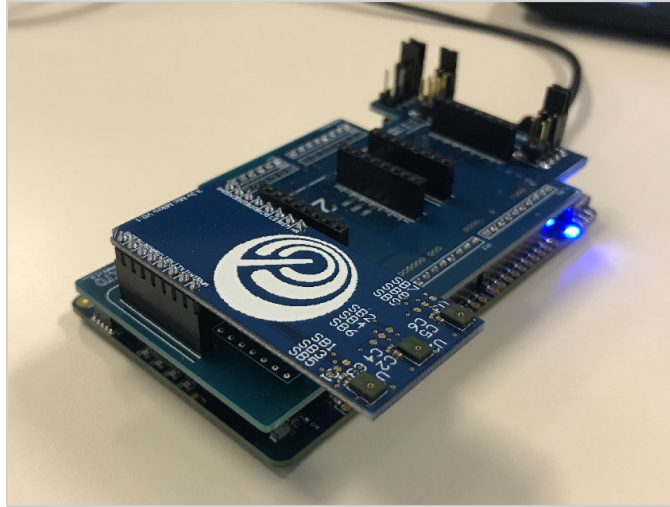


Figure 4. Top-mount Microphone MikroBUS board + MikroBUS shield board + Apollo family EVB

- **Vesper compatible MIC MikroBUS board (AM_VD_CL)**
 - 2 bottom-mounted digital mics (VM3000) with 2cm spacing and an analog mic (VM1010)
 - Support for Wake on Sound (WoS) feature with the VM1010.
 - Configuration definition: ***USE_DMIC_MB3 + USE_WOS_AMIC_MB3 + USE_MIKRO_DEV_SHIELD_REV1***

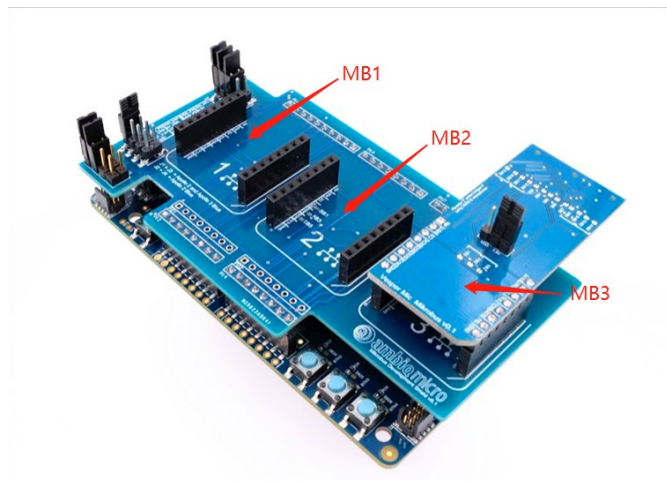


Figure 5. Bottom-mount Microphone MikroBUS board + MikroBUS shield board + Apollo family EVB

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- **Vesper Analog + Digital MIC MikroBUS board (AM_VMB)**
 - 3 bottom-mounted digital mics (VM3000) and 3 analog mics (VM1010)
 - Support for Wake on Sound (WoS) feature with the VM1010.
 - Has FXL6403 GPIO expander to control analog MIC gain.
 - Default setting is 2 x AMIC mode.
 - Configuration definition: **USE_AMIC_VMB + USE_GPIO_FXL6408_MB3 + USE_MIKRO_DEV_SHIELD_REV1**

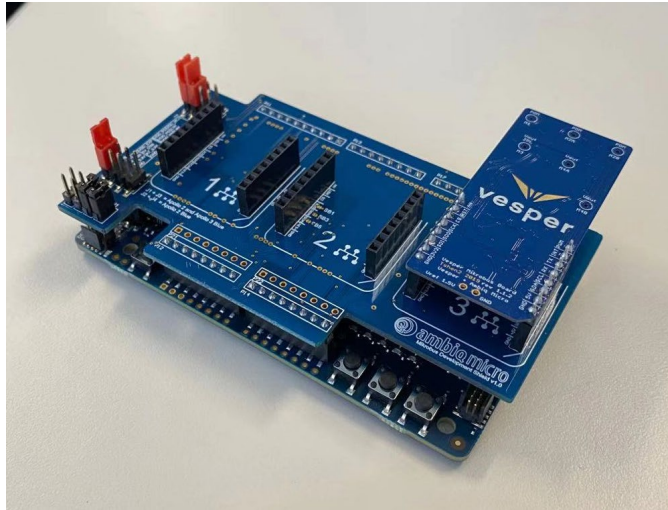


Figure 6. Bottom-mount Microphone MikroBUS board + MikroBUS shield board + Apollo family EVB

- **Vesper Dual-mode MIC MikroBUS board (VM301x)**
 - 1 bottom-mounted VM3011 dual-mode mic (Analog + Digital) and 2 digital mics (VM3000)
 - Support for Wake on Sound (WoS) feature with the VM3011.
 - Default setting is 1 x dual mode MIC enabled.
 - Configuration definition: **USE_DMIC_MB3_VM3011 + USE_MIKRO_DEV_SHIELD_REV1**

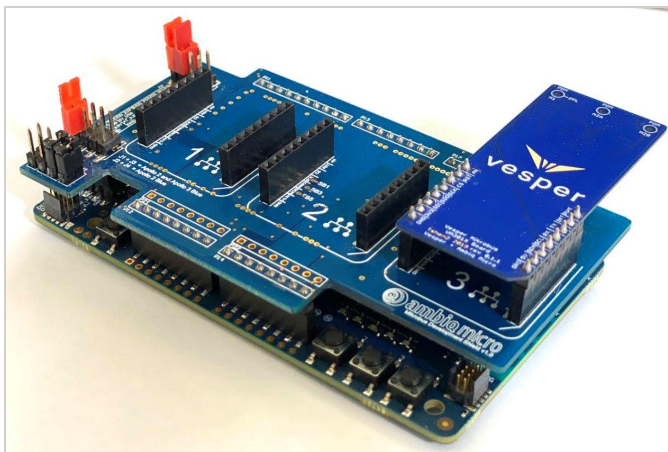


Figure 7. Dual-mode Microphone MikroBUS board + MikroBUS shield board + Apollo family EVB

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

9. Project Directory Structure

- **AmbiqSuite SDK path**
 - <USER_PATH>\AmbiqSuite-KWD
- **VoS Top level path**
 - <USER_PATH>\vos
- **Source Code**
 - vos\sys
- **3rd party Add-ons (DSPC, Sensory, Cyberon, OAL, Return_DSP, U-comm)**
 - vos\third_party\
- **VoS Project**
 - vos\boards\<board_name>\example\vos

10. Project Configuration to Select Target Board and Desired Features

The VoS SDK include the following top-level features:

IP	IP Provider	Description
AB	Ambiq	Universal Audio Buffer (Raw, Encoded Audio data buffer)
AWE	DSPC	AWE (Audio Weaver Engine) audio processing. Includes beam forming (BF) and single-channel noise reduction (SCNR)
THF	Sensory	Truly Hands Free Keyword Detection function - “Alexa” THF models are 64K v3c models
CSpotter	Cyberon	Keyword spotting engine. Supports always listening, non-keyword filtering and keyword spotting.
RDSP	Retune DSP	VoiceSpot – Wake word engine that supports custom voice trigger for target words and phrase.
AID	Open AI Lab	AID.Speech detection engine. (Local CMD phrase)
OPUS	IETF / Xiph.Org	Audio codec that incorporates technology from Skype SILK codec and Xiph.Org’s CELT
mSBC	BlueZ	Modified Sub Band Coding Audio Encoder (Open Source)
OVVP	U-COMM	Own Voice Vibration Pick-up

Table 2. Top-level Features of the VoS SDK

The SDK consists of 7 base projects or workspaces supported in the noted IDE(s):

- DSPC_THF - contains both DSPC audio processing and Sensory Truly Hands Free wake word detection features (IAR & Keil)
- DSPC - contains DSPC audio processing features (IAR & Keil)
- THF - contains Sensory Truly Hands Free features (IAR & Keil)
- DSPC_CSpotter - contains both DSPC audio processing and Cyberon CSpotter features (Keil only)
- CSpotter - contains Cyberon CSpotter KWD features (Keil only)
- RDSP – contains Retune DSP VoiceSpot features (IAR only)

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- AID – contains Open AI Lab AID.Speech features (IAR only)
- Light - contains neither audio pre-processing or keyword detection engine (IAR & Keil). This version support manual trigger (e.g. “push to talk”) and BLE transfer with AMA protocol.

In general, each of these projects shares the same set of files, including the project configuration file, am_vos_sys_config.h. For each of these projects, the configuration file in particular needs to be set correctly for the project to build. Some projects don't require folders of files that are not applicable to the functionality of the project, and are therefore omitted from the project. Alternatively, one could use the VoS project, which is the superset for all other projects, and just “exclude from build” the individual files contained in each of the folders which are not applicable to the specific build.

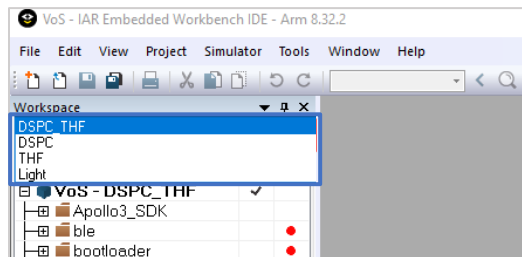


Figure 8. Project target selection for each version

The table below summarizes the modules needed for each target project that are in addition to the base set of files common to all projects.

Target	DSPC AWE	Sensory THF	Cyberon CSpotter	RetunedDSP VS	OAL AID
DSPC_THF	•	•			
DSPC_CSspotter	•		•		
DSPC	•				
THF		•			
CSpotter			•		
RDSP				•	
AID					•
Light					

Table 3. Required Modules in the Four Base Targets

Base **VoS_Light** SDK package (*AmbiqSuite-VoS_v1.x.x_Light.7z*) can build only one target - **Light**.

To build DSPC, THF and CSpotter targets with the VoS_Light package, it needs additional files and libraries. We are providing them as separate add-on packages.

- **DSP Concepts AWE:** *VoS_add-on_DSPC_v1.x.x.7z*
- **Sensory THF:** *VoS_add-on_THF_v1.x.x.7z*
- **Cyberon CSpotter:** *VoS_add-on_CSspotter_v1.x.x.7z*
- **RetunedDSP VS:** *VoS_add-on_RDSP_v1.x.x.7z*
- **OAL AID:** *VoS_add-on_AID_v1.x.x.7z*

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

To build target **DSPC_THF**, it needs the DSPC (*VoS_add-on_DSPC_v1.x.x.7z*) and THF (*VoS_add-on_THF_v1.x.x.7z*) add-on with VoS_Light base package. To build target **DSPC**, it need the DSPC (*VoS_add-on_DSPC_v1.x.x.7z*) add-on based on VoS_Light package.

The subsections that follow describe the settings in the configuration file and the selection of specific files for building each of the 6 base projects, or variations of those projects. Target board and feature selections are selected/enabled by #define's in "am_vos_sys_config.h". These selections are followed by the target selection in the vos\boards\evb_name\vos\inc folder of the VoS project. (e.g., *evb_name* for Apollo3 Blue EVB is apollo3_evb)

- **Audio Signal Chain and Key Word Detection Library Selection in the Configuration File**

- Selected by project target.

```
#if defined (AM_VOS_DSPC)
#define configUSE_AWE 1 // DSPC AWE frame work switch
#else
#define configUSE_AWE 0
#endif

#if defined (AM_VOS_THF)
#define configUSE_Sensory_THF 1 // Sensory detection lib
#else
#define configUSE_Sensory_THF 0
#endif

#if defined (AM_VOS_CSPOTTER) || defined (AM_VOS_CSPOTTER_CES)
#define configUSE_Cyberon_Spotter 1 // Cyberon detection lib
#else
#define configUSE_Cyberon_Spotter 0
#endif
```

Figure 9. Audio chain and detection engine selection

- **EVB and Microphone MikroBUS Board Selection in the Configuration File**

- 1 : Should be 1 when use it with specific board.
- 0 : Should be 0 (not supported)
- • : This could be selected as the user's design requires.

Field in Configuration file	Apollo2 EVB	Apollo2 Blue EVB	Apollo3 EVB (Apollo 3 Plus)	Apollo3 Maya
USE_APOLLO2_EVB	1	0	0	0
USE_APOLLO2_BLUE_EVB	0	1	0	0
USE_APOLLO3_BLUE_EVB	0	0	1	0
USE_MAYA	0	0	0	1
USE_DMIC_MB3	•	•	•	0
USE_DMIC_MB3_VM3011	•	•	•	0
USE_AMIC_MB2	0	0	•	0
USE_AMIC_MB3	0	0	•	0
USE_AMIC_MB3_VMB	0	0	•	0
USE_WOS_AMIC_MB1	•	•	•	0
USE_WOS_AMIC_MB3	•	•	•	0

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

USE_GPIO_FXL6408_MB3	0	0	●	0
USE_EM9304_MIKRO_MB2	●	0	0	0
USE_MIKRO_DEV_SHIELD_REV1	1	1	1	0

Table 4. Hardware Selection in Configuration File

- **USE_DMIC_MB3_VM3011**: VM3011 dual mode MIC (analog for WoS & digital) at MB3.
- **USE_AMIC_MB2**: Single analog MIC at MB2 - debug purpose.
- **USE_AMIC_MB3**: Single analog MIC at MB3 - debug purpose.
- **USE_WOS_AMIC_MB1**: Single VM1010 for WoS (Wake on Sound) at MB1.
- **USE_EM9304_MIKRO_MB2**: EM9304 BLE shield at MB2 (Apollo 2 only).

• Feature Selection in the Configuration File

Field in Configuration file		1	0
configUSE_RTT_RECORDER	RTT audio recorder enabled (SWO)	Disabled	
configUSE_STDIO_PRINTF	Debug print log enabled (UART / RTT / SWO)	No debug log printing.	
configUSE_BLE	Enable audio data transfer via BLE.	Disabled	
configUSE_AUDIO_CODEC	During audio buffer transfer, enable mSBC/OPUS encoding.	Disabled	
configUSE_OVVP_DOUBLE_TAP	Enable OVVP feature and double tap detection using G-sensor.	Disabled	
configUSE_PUTH_TO_TALK	Enable push to talk using on-board button switch.	Disabled	
configUSE_MUTE_MIC	Enable mute MIC using button switch.	Disabled	
configUSE_WOS	Enable Wake on Sound using VM1010, VM3011	Disabled	
configUSE_PREROLL	Buffer pre-roll feature for key word verification.	No pre-roll	
configUSE_AMVOS_AMA	Use AMA protocol to support Alexa app.	Common GATT profile support Read/Write notification.	

Table 5. Feature Selection in the Configuration File

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- **Feature Configuration in the Configuration File**

Field in Configuration file	Description
<i>BUFFERED_SAMPLES_COUNT</i>	Number of audio frames to transfer. (Timeout of streaming) Ex: If audio frame size = 80 samples and duration of audio buffer transfer = 8 sec, then $8 \text{ sec} \times 16\text{KHz audio stream} / 80 \text{ samples} = 1600 \text{ audio frames}$
<i>AUDIO_PREROLL_TIME_MS</i>	If ABX is enabled for UART or BLE, this defines ms of pre-roll data before key word (wake word) is detected from pre-buffer to transfer. Current value is 500 ms

Table 6. Feature Configuration in the Configuration File

- **Included Files for Key Word Detection (KWD) Configuration**

- Select “Alexa” key word detection by including these files in the build:

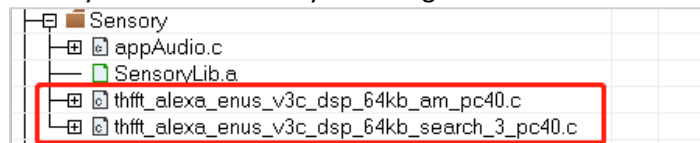


Figure 10. KWD Included Files

- Add or remove unused “c” KWD models from build:
 - Select file in Workspace explorer
 - Right-click to show menu
 - Select “File”
 - Select “Options”
 - Confirm “Exclude from build” field is
 - checked to exclude
 - unchecked to include

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

11. Setting Other Project Options

- **Board Supported Features**

Boards	Common	Features		
	<ul style="list-style-type: none"> - DSPC AWE (QSD, BF, SCNR) - Sensory THF (KWD + CMD phrase) - Wake on Sound - mSBC - Push to Talk - Mute MIC - Pre-roll (AMA) - U-COMM OVVP 	<ul style="list-style-type: none"> - BLE with AMA 	<ul style="list-style-type: none"> - OPUS 	<ul style="list-style-type: none"> - Cyberon CSpotter (KWD + CMD phrase) - Analog MIC input
Apollo2 EVB	•	○	○	
Apollo2 Blue EVB	•	•	○	
Apollo3 Blue EVB	•	•	•	•
Apollo3 Plus EVB	•	•	•	•
Maya (Apollo3 Blue)	•	•	•	•

Table 7. Supported Features List

- Apollo2 EVB support BLE using BLE module shield board.
 - Configuration definition: **USE_EM9304_MIKRO_MB2**

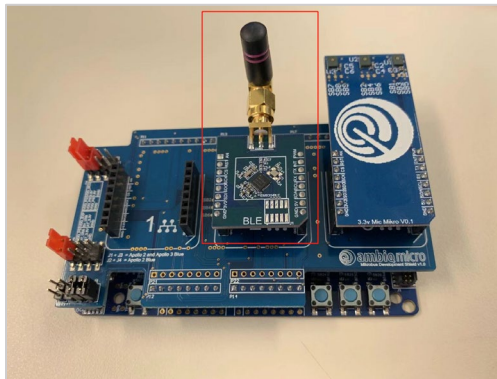


Figure 11. Apollo2 EVB + EM9304 BLE shield

- OPUS can be supported on Apollo2 MCUs without AWE pre-processing. (Apollo2 48MHz / Apollo3 96MHz MCU clock speed with burst mode.)

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- **MIC Configuration**

- 2x Digital MICs

- Default layout: **configUSE_ALEXA_QUAL_LAYOUT** to 1.

```

//*****
// AWE module configuration
//*****
#if configUSE_AWE
#define configUSE_ALEXA_QUAL_LAYOUT      1
#define configUSE_AIC_LITE_LAYOUT        0
#endif

```

Figure 12. configUSE_ALEXA_QUAL_LAYOUT definition in am_vos_sys_config.h

- AIC layout: **configUSE_AIC_LITE_LAYOUT** to 1. Enable DSPC AIC (Adaptive Interference Cancellation) feature using two DMICs. Recommended MIC separation is 5~7 cm. Make AWB file to include project as below. If you want to use AIC layout, set apollo3p_2mic_remote_LMS_only_v4_InitAWB.c file to be included. (Use right click on that file -> Options -> Check or Uncheck "Exclude from build")

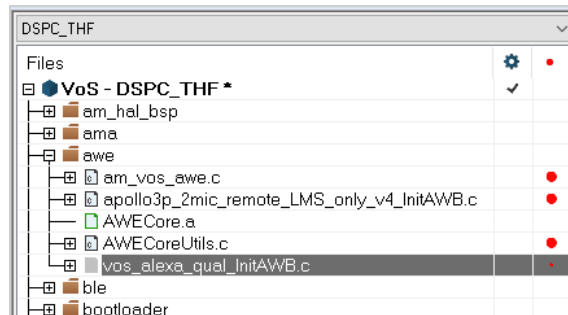


Figure 13. AIC processing module selection in project window

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- Single Digital MIC
 - Latest VoS SDK (v1.2.6) support one digital MIC without audio pre-processing. Only supported with THF, CSpotter, RDSP, Light target.
 - Using right channel setting of DMIC as a default.

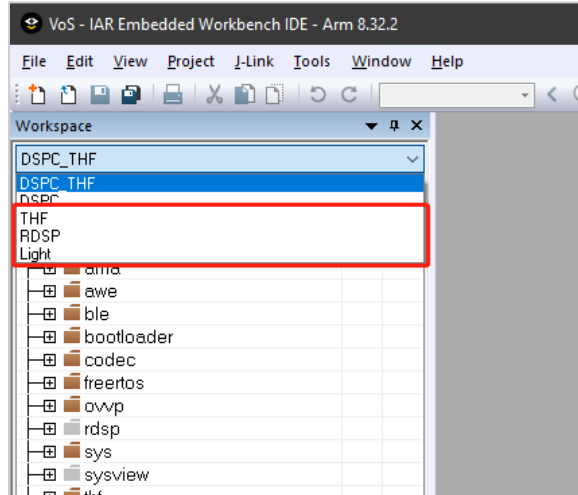


Figure 14. Single MIC supported targets in project window.

12. Building the SDK to Create a Custom Binary for Stand-alone Mode

- **In IAR, build project:**
 - Select Project -> Clean
 - Select Project -> "Rebuild All"
- **Confirm board is connected by USB and powered**
- **In IAR, download bin file:**
 - Select Project -> Download -> "Download active application"
- **Cycle power on EVB after Flashing MCU to initiate operation**

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

13. Operation with the Alexa App

- **EVB (and Maya) LED indication**

Operation	LED	Description
Boot up	All LEDs	LEDs swirl twice
Advertising (not connected to App) or BLE disabled	LED D5 (Apollo2 / Apollo2 Blue / Apollo3 EVB) All LEDs (Maya)	Blinks twice every second. Maya board LEDs blink twice every 3 seconds.
Connected with App	LED D5 (Apollo2 / Apollo2 Blue / Apollo3 EVB) All LEDs (Maya)	Blinks once every second. Maya board LEDs blink once every 3 seconds.
Keyword detected	All LEDs	Swirls one time
CMD phrase detected	LED pattern indicates each command.	e.g., D5 “Louder”, D6 “Pause Music”...
RTT recording started	LED D5 (Apollo2 / Apollo2 Blue / Apollo3 EVB)	Blinks every 300 ms.

Table 8. LED indications of system status

- **iOS App installation**
 - Download the Alexa app from the Apple App Store by searching for “Alexa”.
 - <https://itunes.apple.com/us/app/amazon-alexa/id944011620?mt=8>
<Note> Needs US store account.
- **Android App installation**
 - Download the Alexa app from Google Play Store by searching for “Alexa”.
 - <https://play.google.com/store/apps/details?id=com.amazon.dee.app&hl=en>
<Note> Also needs US PlayStore account.
 - Connect KWD device with mobile phone app
- **Connect KWD device with mobile phone app**
 - With the device programmed and powered up, open the Amazon Alexa app. Tap Device icon on bottom right of screen -> Add new device -> Choose Headphones. Connect to “VoS-xx-AMA-xxxx”. Last four digits are last four characters of the MAC address of device.

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

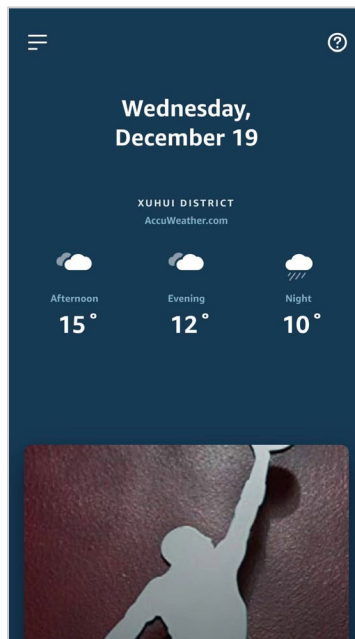


Figure 15. Devices Icon in Alexa App

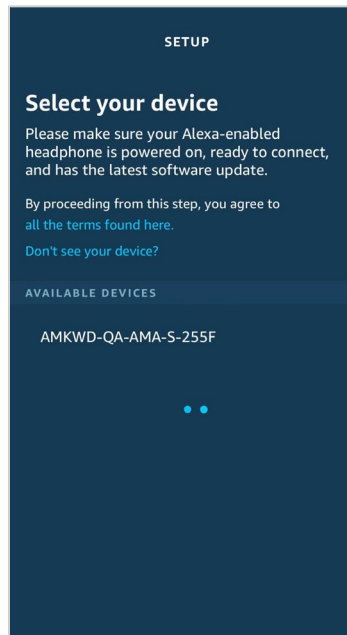


Figure 16. Device Selection in Alexa App

- With the board BLE connection complete, the screen will change to display “Setup Complete”. You are now connected to AVS. If you didn’t log in to Alexa service, you need to log in this time.
- Alexa may not always work in China, so you may need to enable VPN to use this service if you have any network trouble.

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

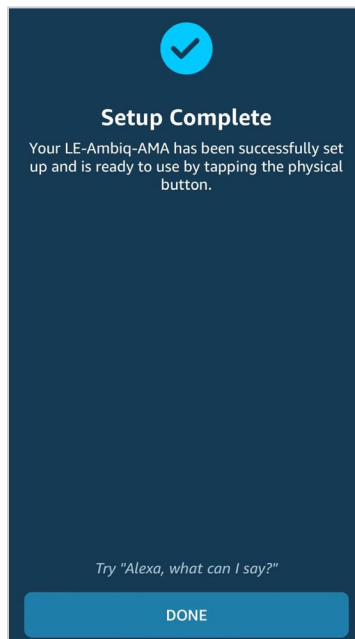


Figure 17. Device Setup Complete in Alexa App

- Say “Alexa, what’s the weather?” The screen will display the activity of the app that transfers data to AVS. The app will respond with the weather. You can ask Alexa for any information, and the full host of queries offered by AVS.
- If you are using DSPC only version or Light version, you can use “Push to Talk” button (default is BTN3 on EVB) to trigger Alexa AI.

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

14. VoS SDK Tuning Guide

- VoS DSPC/THF flow chart**

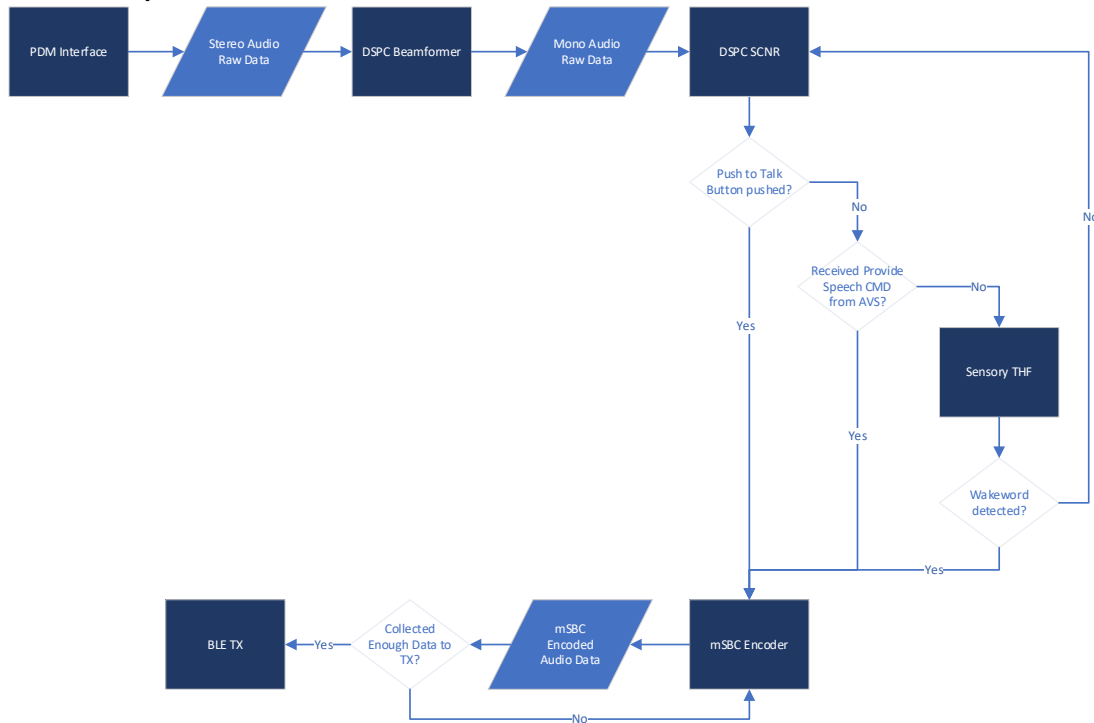


Figure 18. VoS DSPC/THF Flow Chart

- VoS Light flow chart**

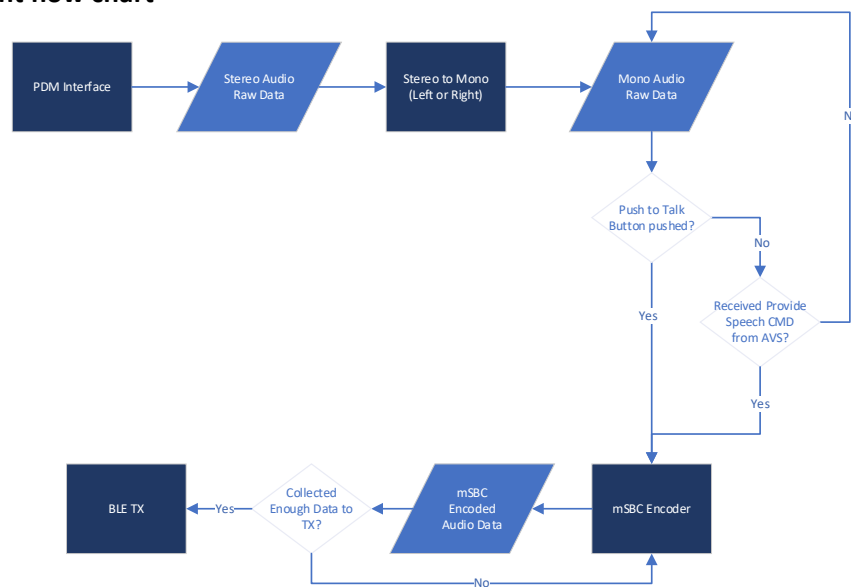


Figure 19. VoS Light Flow Chart

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- **PDM interface**

- am_pdm_isr() function is called each time the PDM interrupt is triggered.
- Puts stereo data in a ring buffer and sends notification to the audio processing task.

```
// Test code for PDM wakeup time measurement
//am_hal_gpio_state_write(LED_SYSTEM, AM_HAL_GPIO_OUTPUT_TOGGLE);
//
// Once our DMA transaction completes, we will disable the PDM and send a
// flag back down to the main routine. Disabling the PDM is only necessary
// because this example only implemented a single buffer for storing FFT
// data. More complex programs could use a system of multiple buffers to
// allow the CPU to run the FFT in one buffer while the DMA pulls PCM data
// into another buffer.
//
if (ui32Status & AM_HAL_PDM_INT_DCMP)
{
    // trigger next transaction
    PDMn(0)->DMATOTCOUNT = AM_SPP_FRAME_SAMPLES * SAMPLE_32BIT; // FIFO unit in bytes

    am_audio_buffer_push(AM_AUDIO_BUFFER_STEREO, g_ui32PDMDataBuffer, PCM_FRAME_SIZE * SAMPLE_32BIT);
    #if configUSE_OVERFLOW_DOUBLE_TAP
```

Figure 20. am_pdm0_isr() in am_vos_isr.c File

- **DSPC Beamformer / SCNR**

- VoS uses DSPC beam former to convert stereo audio to mono output data with directional audio gain control. It is one beam direction and set up for maximum SNR.
- SCNR is processed in DSPC AWE modules after beam forming.



Figure 21. DSPC Audio Weaver pre-processing

- Two audio input and one output can be accessed with below code. It can be reviewed with define **configUSE_AWE**.

```
//
// Data IO of Layout
// Get Current AWE Layout number of channels
// Layout has 1 input and 1 output
//
awe_layoutGetChannelCount(&g_AWEInstance, 0, &fwInCount, &fwOutCount);
if(fwInCount == spp_input->ui32SppInChNum)
{
    for(uint32_t ch_idx=0; ch_idx<fwInCount; ch_idx++)
    {
        awe_audioImportSamples(&g_AWEInstance, (void*)spp_input->SppInputArray[ch_idx], 1, ch_idx, (Sam
    }
}
else
{
    AM_SPP_LOG("Input channel number is not aligned with loading AWB...\n");
    return;
}
```

Figure 22. Input Data to DSPC AWE (Beam Former/SCNR) at am_spp_input_push() in am_vos_awe.c

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

```
//
// Data IO of Layout
// Get Current AWE Layout number of channels
// Layout has 1 input and 1 output
//
awe_layoutGetChannelCount(&g_AWEInstance, 0, &fwInCount, &fwOutCount);
if(fwOutCount == spp_output->ui32SppOutChNum)
{
    for(uint32_t ch_indx=0; ch_indx<fwOutCount; ch_indx++)
    {
        awe_audioExportSamples(&g_AWEInstance, (void*)spp_output->SppOutputArray[ch_indx], 1, ch_indx, (S
    #if AM_configUSE_LOG_SPP
        amu2s_send(Amu2s_spp, spp_output->SppOutputArray[ch_indx], AM_SPP_FRAME_SAMPLES*AM_SPP_OUT_SAMPLE
    #endif // AM_configUSE_LOG_SPP
    }
}
else
{
    AM_SPP_LOG("Output channel number is not aligned with loading AWB...\n");
    return;
}
```

Figure 23. Output data from DSPC AWE at am_spp_output_pop() in am_vos_awe.c

- **Stereo to Mono (without SPP)**

- The am_vos_stereo_to_mono_proc() function converts stereo data to mono audio data.

```
//
// This process only take 1-channel data into MWE
//
void am_vos_stereo_to_mono_proc(int32_t *nLRSample, int16_t *nMonoSample)
{
    int32_t nSample;

    for (nSample = 0; nSample < AM_SPP_FRAME_SAMPLES; nSample++)
    {
        // Without voice pre-processing(e.g. Beamforming), using right MIC's data as a default.
        nMonoSample[nSample] = nLRSample[nSample] >> 16; // Right channel
        //nMonoSample[nSample] = nLRSample[nSample] & 0xFFFF; // Left channel
    }
}
```

Figure 24. am_vos_stereo_to_mono_proc () in am_vos_audio.c File

- Other audio pre-processing functions can be added here. (e.g., Beam Former, Noise Reduction)

- **Sensory THF**

- If “Push to Talk” and “Provided Speech” (manual trigger) are not set, VoS checks for key word detection with the Sensory THF engine.

```
void am_vos_engine_process(int16_t *pi16InputBuffer, int16_t i16InputLength)
{
    errors_t result;

    #if configUSE_PREROLL
        appStruct_T *ap = &appStruct;
        AUDIOINDEX epIndex, tailCount, stIndex;
    #endif // configUSE_PREROLL

    result = SensoryProcessBrick(pi16InputBuffer);

    #if configUSE_OVVP_DOUBLE_TAP
        if(result == ERR_OK && (s_flag > 0))
    #else // configUSE_OVVP_DOUBLE_TAP
        if(result == ERR_OK)
    #endif // configUSE_OVVP_DOUBLE_TAP
    {
        am_vos_reset_detected_flag();
        g_sVosSys.ui8KwdDetectedFlag = 1;
    }
```

Figure 25. Key Word Detection Check in am_vos_engine_process()

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- If THF detects the key word, the SensoryFindStartpoint() and SensoryFindEndpoint() functions are called to calculating the start/end index of the key word.
- **Audio codec encoder (OPUS or mSBC)**
 - After the key word is detected (or “Push to Talk” / “Provided Speech” command is received), the codec task checks the status of the ring buffer and then runs the OPUS or mSBC codec.

```

    AM_CRITICAL_BEGIN_VOS;
    ui32StreamLen = am_app_utils_get_ring_buffer_status(&(g_sAmUtil.sRingBuf[AM_AUDIO_BUFFER_MONO]));

    AM_CRITICAL_END_VOS;

    while(ui32StreamLen)
    {
        //
        // Attempt to clear mono buffer
        //
        if(ui32StreamLen > codecInBufRemaining)
        {
            ...

            #if configUSE_MSBC_BLUEZ || configUSE_SBC_BLUEZ
                am_vos_codec_encode(&(g_sVosSys.sBluezSBCInstance), p_CodecInBuf,
                                    CODEC_IN_RING_BUFF_SIZE, p_CodecOutBuf, CODEC_OUT_RING_BUFF_SIZE);
            #endif // configUSE_MSBC_BLUEZ || configUSE_SBC_BLUEZ

            #if configUSE_OPTIM_OPUS
                am_vos_codec_encode(NULL, p_CodecInBuf, CODEC_IN_RING_BUFF_SIZE, p_CodecOutBuf, CODEC_OUT_RING_BUFF_SIZE);
            #endif // configUSE_OPTIM_OPUS

            am_audio_buffer_nested_push(AM_AUDIO_BUFFER_ENCODED, p_CodecOutBuf, CODEC_OUT_RING_BUFF_SIZE);
        }
    }

```

Figure 26. Ring Buffer Status Check and Encode Audio Data in am_vos_codec_task()

- **BLE TX**
 - Encoded audio data is collected and sent to the Alexa app through BLE when it is at least the designated size. (Current packet size is 114 bytes.)

```

    if(am_app_utils_get_ring_buffer_status(&(g_sAmUtil.sRingBuf[AM_AUDIO_BUFFER_ENCODED])) > 0)
    {
        #if configUSE_BLE
            am_vos_ble_stream_send();
        #else // configUSE_BLE
            am_vos_audio_flush_ring_buffer();
        #endif // configUSE_BLE
    }

```

Figure 27. Encoded Audio Data Size Check in am_vos_codec_task()

- User can disable AMA protocol by defining the below **configUSE_AMVOS_AMA** to 0, then the VoS application is working as common GATT profile and can read/write data using a BLE test app. (e.g. LightBlue™)

```

#define configUSE_AMVOS_AMA 1
#define configUSE_BLE_SECURE_CONNECTION 1 // Using BLE with secured connection.
#define configUSE_BLE_WATCHDOG 1 // Using watchdog timeout feature to
#define configUSE_BLE_Measure_Throughput 0
#define configUSE_BLE_ERROR_SW_RESET 0
#define AM_AMA_DEVICE_TYPE_STR "A3U3B4RTI76K2J" // Ambiq Mirco ID
#endif // configUSE_BLE

```

Figure 28. Disabling AMA Protocol

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

15. AMVoS Profile with Common BLE Test App

- **Build VoS firmware**
 - Define ***configUSE_AMVOS_AMA*** to 0.

```
#define configUSE_AMVOS_AMA 0
#define configUSE_BLE_SECURE_CONNECTION 1 // Using BLE with secured connection.
#define configUSE_BLE_WATCHDOG 1 // Using watchdog timeout feature to
#define configUSE_BLE_Measure_Throughput 0
#define configUSE_BLE_ERROR_SW_RESET 0
#define AM_AMA_DEVICE_TYPE_STR "A3U3B4RTI76K2J" // Ambiq Mirco ID
#define configUSE_BLE
```

Figure 29. Building VoS without AMA Protocol

- **Download and Run general BLE test app.**
 - Recommended app is LightBlue™
 - Run LightBlue™ and connect “AMVOS-EW-xxxx” device.

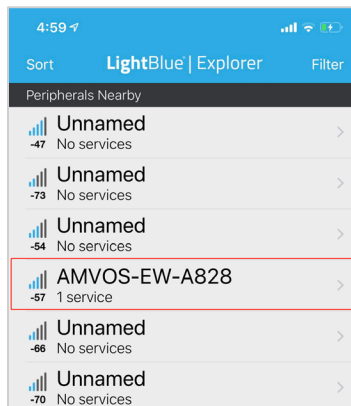


Figure 30. LightBlue Explorer

- **Streaming audio data through BLE**
 - Select the UUID ending with “2E0012” and has “Read Notify” properties.

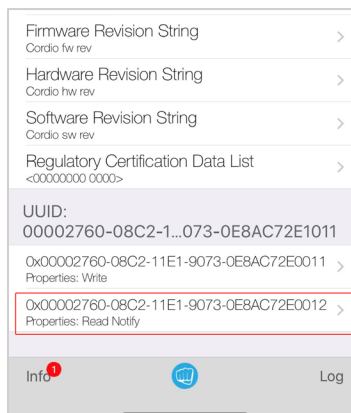


Figure 31. UUID in LightBlue Properties Page

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- Click “Listen for notifications”

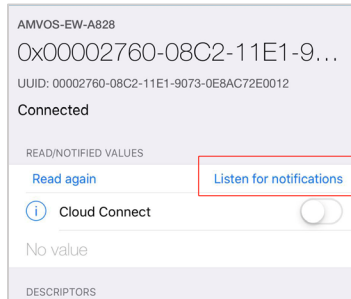


Figure 32. Listen for Notifications in LightBlue

- Send audio data using key word trigger (saying “Alexa,”) or BTN3 (push to talk) button. Then you can see streaming data.

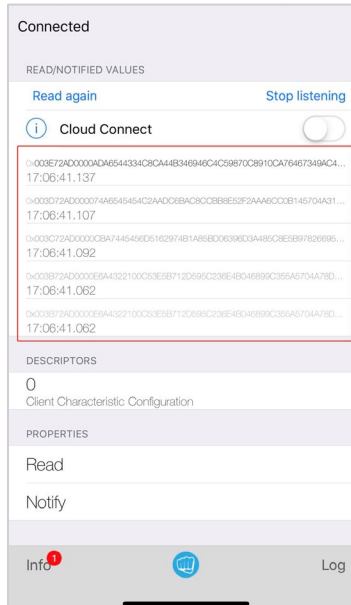


Figure 33. Streaming Data in LightBlue

- If you want to get and verify this dumped data, use “Log” function of this app.

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

16. Building an OTA Image

The OTA feature is supported in the IAR and Keil projects.

- **Apollo2 / Apollo2 Blue EVB**

- Set target to xxx_OTA from the project target menu. (e.g. DSPC_THF_OTA, Light_OTA)

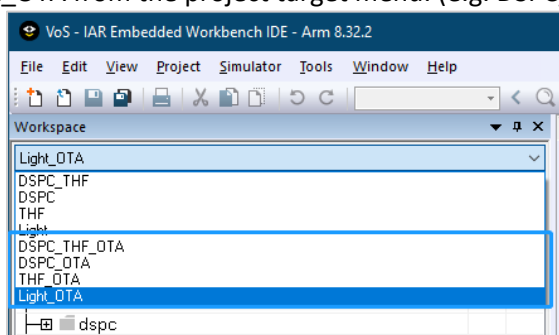


Figure 34. OTA project selection on Apollo2

- Do a “Rebuild All” to create the VoS_xxxx_OTA.bin file.
- Use a Python script located at vos/scripts/vosota/apollo2/ to make two images - Starter (OTA bootloader + binary image) and Update (binary image only)
 - Before running the script, make sure that the path listed in the Makefile for the bin files created above is correct. Makefile’s file path is vos/scripts/vosota/apollo2/Makefile. It should look like below.

```

1  #*****
2  #
3  # Simple Makefile to prepare binaries for VoS OTA.
4  #
5  #*****
6
7  FLAG_ADDR_APOLLO2 = 0x6000
8  LOAD_ADDR_APOLLO2 = 0x8000
9  APPBIN_APOLLO2_BLUE_PROD = ../../boards/apollo2_blue_evb/examples/vos/iar/bin/VoS_Light_OTA.bin
10 BOOTBIN_APOLLO2_BLUE_PROD = ../../boards/apollo2_blue_evb/examples/multi_boot/keil/bin/multi_boot.bin
11 UPDATEBIN_APOLLO2_BLUE = ../../boards/apollo2_blue_evb/examples/vos/iar/bin/VoS_Light_OTA.bin
12
13 : $(APPBIN_APOLLO2_BLUE_PROD) $(BOOTBIN_APOLLO2_BLUE_PROD) $(UPDATEBIN_APOLLO2_BLUE)

```

Figure 35. OTA image generated by Makefile for Apollo2

- Run “Make” command in command prompt, then two binary files will have been generated.
 - starter_binary_apollo2_vos.bin**: OTA bootloader + binary image
 - update_binary_apollo2_vos.bin**: binary image that will be used for OTA reprogramming

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- **Apollo3 EVB / Apollo3 Plus EVB / Maya**

- Apollo3 project's OTA feature is enabled by default. (No need to trigger this by target select.)
- Use a Python script located at vos/scripts/vosota/apollo3/ to make two images - Starter (OTA bootloader + binary image) and Update (binary image only)
 - Before running the script, make sure that the path listed in the vos/scripts/vosota/apollo3/Makefile for the bin file created above is correct. It should look like below.

```

1 #
2 #
3 # Simple Makefile to prepare binaries for AMOTA for Apollo3.
4 #
5 #*****
6 #
7 #
8 # Apollo3-BLUE EVB
9 #UPDATEBIN_APOLLO3_BLUE = ../../boards/apollo3_evb/examples/freertos_amota/iar/bin/freertos_amota.bin
10 #APPBIN_APOLLO3_BLUE = ../../boards/apollo3_evb/examples/freertos_amota/iar/bin/freertos_amota.bin
11 UPDATEBIN_APOLLO3_BLUE = ../../boards/apollo3_evb/examples/key_word_detection/iar/bin/VoS_Light.bin
12 APPBIN_APOLLO3_BLUE = ../../boards/apollo3_evb/examples/key_word_detection/iar/bin/VoS_Light.bin
13
14 : $(APPBIN_APOLLO3_BLUE) $(UPDATEBIN_APOLLO3_BLUE) $(UPDATEBIN_APOLLO3_BLUE_ETHERMIND) $(APPBIN_APOLLO3_BLUE_ETHERMIND)
15 # Apollo3_Gordian

```

Figure 36. OTA image generated by Makefile for Apollo3

- Run “Make” command in command prompt, then two binary files will have been generated.
 - starter_binary_apollo3_vos.bin**: OTA bootloader + 1st firmware image
 - update_binary_apollo3_vos.bin**: 2nd firmware image that will be used for OTA reprogramming

- **Buildtools Installation**

If you don't have **make.exe** to use Makefile and you saw the below error message, then you need to install buildtools for Windows.

```

C:\Work\Ambiq\KWD\KWD_RTOS_Framework_master\vos\scripts\vosota\apollo2>make
'make' is not recognized as an internal or external command,
operable program or batch file.
C:\Work\Ambiq\KWD\KWD_RTOS_Framework_master\vos\scripts\vosota\apollo2>

```

Figure 37. Error message due to non-existing make.exe

- Download **msys-1.0.7z** package from below link.
 - Box Cloud: <https://app.box.com/s/96crrw2muzudqazae87bk1o0kjin4gu7v>
 - Baidu Cloud (for China): <https://pan.baidu.com/s/1cSRxlzvCayd1pPRgXcpLrg> (pwd: tyew)
- Extract and add bin folder (e.g. C:\DevUtils\msys\1.0\bin) to your system path.
- **Python Installation**

Python 3.6 or later also needs to be installed.

 - **pycryptodome** package is needed to build Apollo3 OTA image.
 - Install with “**pip install pycryptodome**” command.
 - If you have any pre-installed crypto related package, please uninstall those first.


```

pip uninstall crypto
pip uninstall pycrypto

```

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

17. OTA image download

- **Download Firmware to the EVB**

- If the EVB has not been updated with the OTA supported image, you must program the MCU with a binary image (.bin) file that contains both the application code and the code that enables OTA reprogramming. The above section did that and, for this demo, that bin file is called `starter_binary_apollo2_vos.bin` or `starter_binary_apollo3_vos.bin`.
- Apollo2 / Apollo2 Blue EVB: The `starter_binary_apollo2_vos.bin` can be programmed using the Segger J-Flash Lite programming utility (<https://www.segger.com/products/debug-probes/j-link/technology/flash-download/>).
- Apollo3 EVB / Apollo3 Plus EVB / Maya: The `starter_binary_apollo3_vos.bin` can be programmed using the Segger J-Flash Lite programming utility. It needs to assign “**Prog. addr.**” to `0x0000C000`. (Apollo2 doesn’t need to change this address.)

After installation of latest J-Link S/W (v6.47d or later), this address is updated automatically when selected Apollo3. (AMA3B1KK)

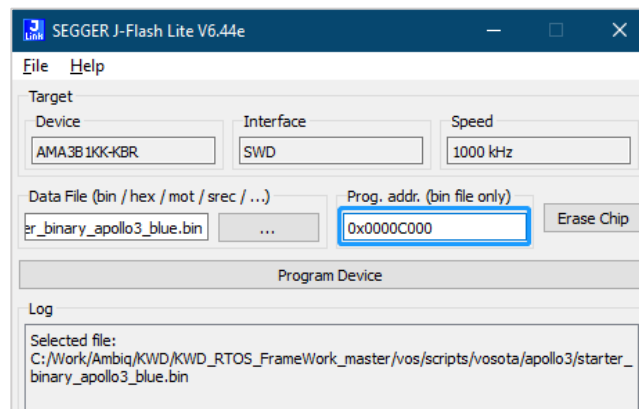


Figure 38. J-Flash Lite address setting for Apollo3

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- **Update the EVB Firmware via OTA**

- This requires that the EVB already has a pre-programmed OTA upgradable image as described above.
- Install the **Ambiq OTA** app from the Apple store (Search with “Ambiq”):
 - <https://itunes.apple.com/us/app/ambiq-ota/id1190453962>
- Add KWD “update” image file (*update_binary_apollo2_vos.bin* or *update_binary_apollo3_vos.bin*) to Ambiq OTA app’s internal storage using iTunes. (or iTools)
 - Connect phone to PC -> Open iTunes -> Hit phone icon under the top menu
-> Select “File Sharing” -> Select “Ambiq OTA” app from the app list
-> Drag and drop the *update_binary_apollo2_vos.bin* file to the “Documents” box

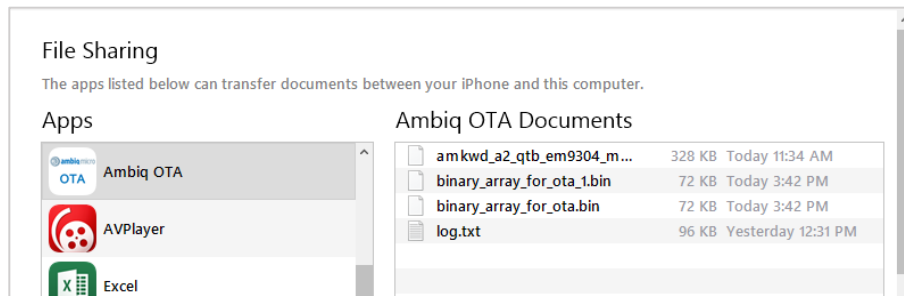


Figure 39. iTunes file manager

- Open Ambiq OTA on your iPhone and skip the short tutorial.
- Wait for connections to show, and select the EVB. (e.g. VoS-xx-AMA-xxxx)

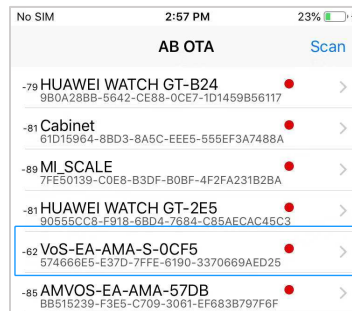


Figure 40. Ambiq OTA app scan result

- Select “Load bin file” and choose the update image file (e.g., *update_binary_apollo2_vos.bin*).
 - Select “Send to device”. This starts firmware update via OTA.
 - Close the app after the update.
- **OTA app for Android**
 - Ambiq OTA apk file is provided in SDK. (AmbiqSuite)
 - File path is “AmbiqSuite-KWD\tools\amota**Application-debug.apk**”

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

18. Audio (Voice) Data Recording using RTT or AMU2S.

The VoS SDK supports RTT (J-Link) and AMU2S (SPI to USB) audio recording. AMU2S is supported on Apollo 3 Blue Plus EVB only.

- **Build image for audio data recording**

- For RTT recording, set **configUSE_RTT_RECORDER** to 1.

```
#define configUSE_SYSVIEWER      0
#define configUSE_SYS_LOG        0
#define configUSE_RTT_RECORDER   1
#define configUSE_AMU2S_RECORDER 0
#define configUSE_STDIO_PRINTF    0
```

Figure 41. RTT recorder definition

- For AMU2S recording, set **configUSE_AMU2S_RECORDER** to 1.

```
#define configUSE_SYSVIEWER      0
#define configUSE_SYS_LOG        0
#define configUSE_RTT_RECORDER   0
#define configUSE_AMU2S_RECORDER 1
#define configUSE_STDIO_PRINTF    0
```

Figure 42. RTT recorder definition

- Select dump data type. (RAW data or SPP processed data)

```
//////////////////////////////////////
#if configUSE_RTT_RECORDER
#define configUSE_RECORD_RAW_PCM      1    // Select which data be recorded
#define configUSE_RECORD_FULL_FILTER  0
```

Figure 43. Recorded data select

- Set **configUSE_BLE** and **configUSE_AUDIO_CODEC** to 0.

```
#define configUSE_BLE      0
#define configUSE_AUDIO_CODEC 0
#define configUSE_LEDS     1
```

Figure 44. Disable codec and BLE when using RTT recorder

- Build and download image to board.

- **Install SW on PC**

- Install J-Link v6.47x or later
 - <https://www.segger.com/downloads/jlink/#J-LinkSoftwareAndDocumentationPack>
 - Add J-Link directory to system path variable: Start -> Type "env" -> Select "Edit system environment variable" -> Environment Variable -> Add path directory of J-Link to current path variable.
- Install python 3.6 or later
 - Also add python directory path to system path variable.

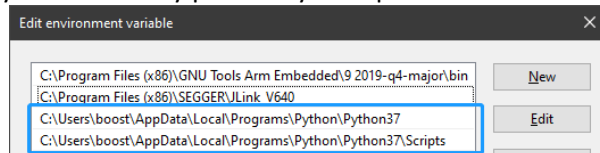
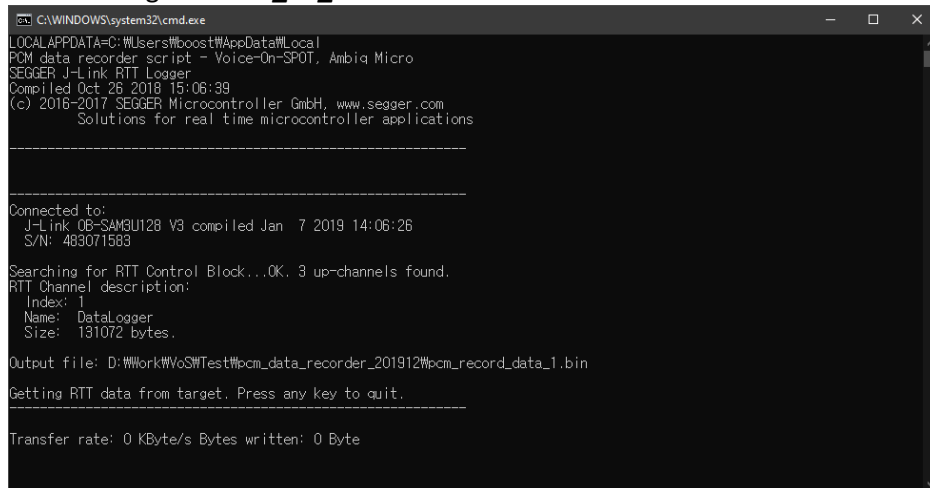


Figure 45. System path environment variable

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- Install necessary modules: docopt, soundfile, numpy, pandas
e.g. Command shell -> ***pip install docopt soundfile numpy pandas***
- RTT and AMU2S PCM recorder script file package
 - ***vos_audio_recorder_200714.7z***: [Box Cloud](#)
 - ***vos_audio_recorder_200714.7z***: [Baidu Cloud \(for China\)](#) pw: 5brv
- **Audio (Voice) data recording**
 - Connect board (VoS firmware programmed with above way) to PC. Or reset the board after programmed.
 - RTT recording: run ***“vos_rtt_recorder.bat”***



```

CA\WINDOWS\system32\cmd.exe
LOCAL APPDATA=C:\Users\#boost\AppData\Local
PCM data recorder script - Voice-On-SPOT, Ambiq Micro
SEGGER J-Link RTT Logger
Compiled Oct 26 2018 15:06:39
(c) 2016-2017 SEGGER Microcontroller GmbH, www.segger.com
Solutions for real time microcontroller applications

-----
Connected to:
J-Link OB-SAM3U128 V3 compiled Jan 7 2019 14:06:26
S/N: 483071583

Searching for RTT Control Block...OK, 3 up-channels found.
RTT Channel description:
Index: 1
Name: DataLogger
Size: 131072 bytes.

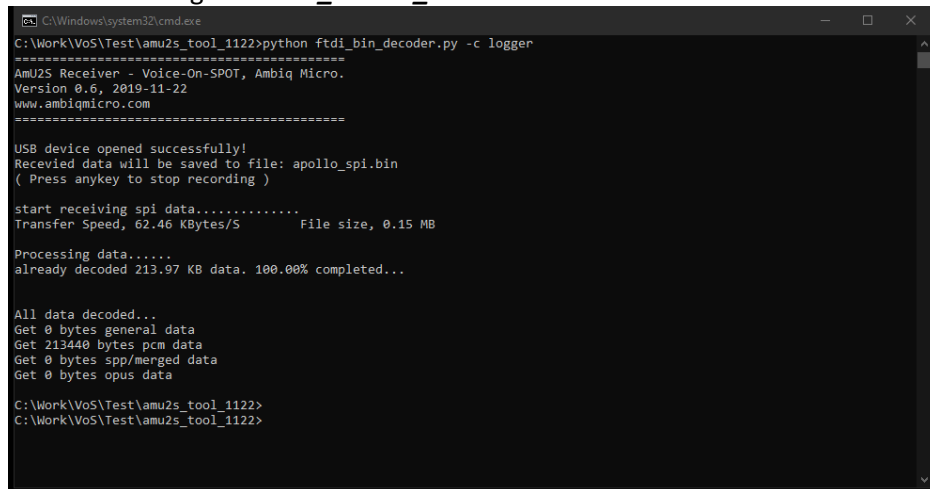
Output file: D:\Work\VoS\Test\pcm_data_recorder_201912\pcm_record_data_1.bin

Getting RTT data from target. Press any key to quit.

-----
Transfer rate: 0 KByte/s Bytes written: 0 Byte
  
```

Figure 46. Run RTT datalogger batch file

- AMU2S recording: run ***“vos_amu2s_recorder.bat”***



```

CA\Windows\system32\cmd.exe
C:\Work\VoS\Test\amu2s_tool_1122>python ftdi_bin_decoder.py -c logger
=====
AMU2S Receiver - Voice-On-SPOT, Ambiq Micro.
Version 0.6, 2019-11-22
www.ambiqmicro.com
=====

USB device opened successfully!
Received data will be saved to file: apollo_spi.bin
( Press anykey to stop recording )

start receiving spi data.....
Transfer Speed, 62.46 KBytes/s      File size, 0.15 MB

Processing data.....
already decoded 213.97 KB data. 100.00% completed...

All data decoded...
Get 0 bytes general data
Get 213440 bytes pcm data
Get 0 bytes spp/merged data
Get 0 bytes opus data

C:\Work\VoS\Test\amu2s_tool_1122>
C:\Work\VoS\Test\amu2s_tool_1122>
  
```

Figure 47. Run AMU2S audio recorder

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- Press BTN4 on Apollo 2/3 EVB to start recording.
- Press any key to stop recording. This generated PCM format file named *"pcm_record_data_x.bin"* (RTT) or *"apollo_spi_log_pcm"* (AMU2S)
- **Convert raw data to WAV file format**
 - Run `pcm_to_wav.py` to convert it to WAV file format.
 - RTT: Command shell -> ***python bin_to_wav.py pcm -i pcm_record_data_1.bin***
 - AMU2S: Command shell -> ***python bin_to_wav.py pcm -i apollo_spi_log_pcm***
 - This generates 2 types of wav files. (before and after normalization.)

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

19. Debug Message Output

- **UART (Supported for all EVBs, not supported for Maya board)**
 - Define ***configUSE_STDIO_PRINTF*** and ***configUSE_PRINTF_UART0*** to 1.

```
#define configUSE_SYSDVIEWER      0
#define configUSE_SYS_LOG        0
#define configUSE_RTT_RECORDER    0
#define configUSE_STDIO_PRINTF    1
```

Figure 48. *configUSE_STDIO_PRINTF* definition in *am_vos_sys_config.h*

```
#if configUSE_STDIO_PRINTF
#define configUSE_PRINTF_UART0    1
#define configUSE_PRINTF_RTT      0
#define configUSE_PRINTF_SWO      0
```

Figure 49. *configUSE_PRINTF_UART0* definition in *am_vos_sys_config.h*

- Make serial port setting as below. (460800 bps / data bit 8 / stop bit 1 / parity none / flow control none)

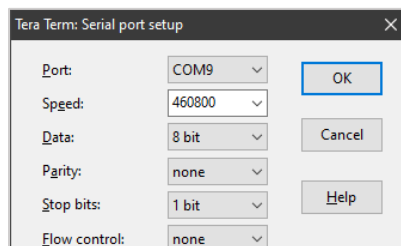


Figure 50. Serial port setup

- Set terminal setup's **New-line** setting as below.

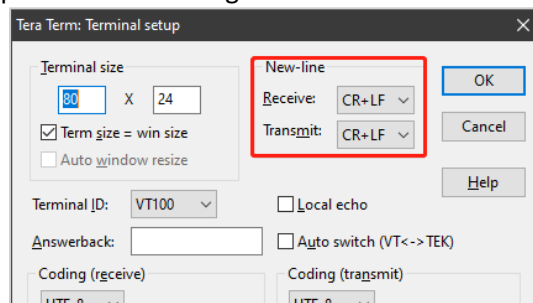


Figure 51. Terminal setup

- UART debugging log is shown as below.

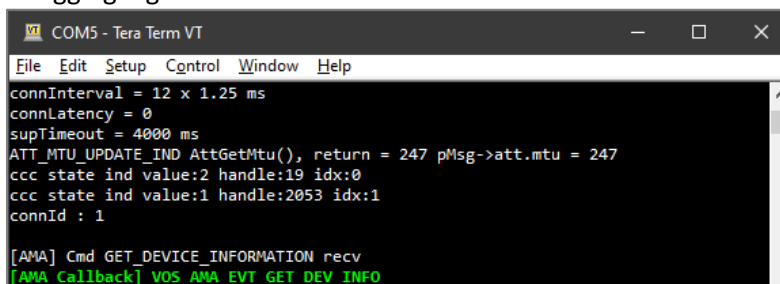


Figure 52. UART print out using Tera Term

Ambiq Micro Voice-on-SPOT with SPP/KWD Software Framework

- **SWO Output (Supported for all EVBs and Maya board)**
 - Define **configUSE_STDIO_PRINTF** and **configUSE_PRINTF_SWO** to 1.

```

//*****
// System level functional module selection
//*****
#define configUSE_SYSVIEWER      0
#define configUSE_SYS_LOG       0
#define configUSE_RTT_RECORDER   0
#define configUSE_STDIO_PRINTF   1

```

Figure 53. configUSE_STDIO_PRINTF definition in am_vos_sys_config.h

```

//*****
// std IO sub module configuration
//*****
#if configUSE_STDIO_PRINTF
#define configUSE_PRINTF_UART0    0
#define configUSE_PRINTF_RTT     0
#define configUSE_PRINTF_SWO     1

```

Figure 54. configUSE_PRINTF_SWO definition in am_vos_sys_config.h

- Run J-Link SWO Viewer -> Select MCU type (e.g., Apollo3 Plus: AMA3B2KK, Apollo3: AMA3B1KK, Apollo2: AMAPH1KK) -> Click measure clock -> Change SWO clock speed to 1 Mhz

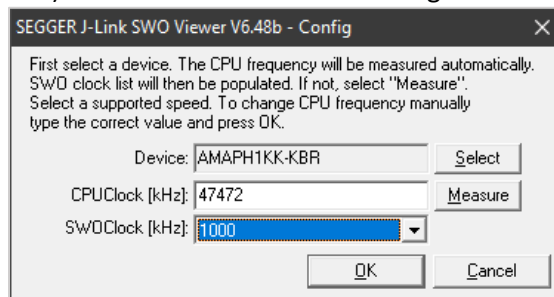


Figure 55. SWO device and clock configuration

- SWO debugging log message is shown as below.

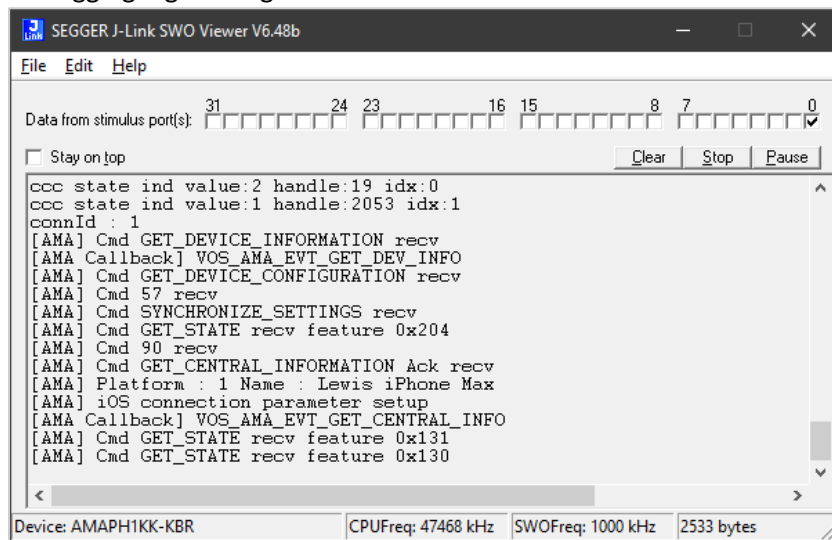


Figure 56. SWO debugging message print out