Voice on Arm

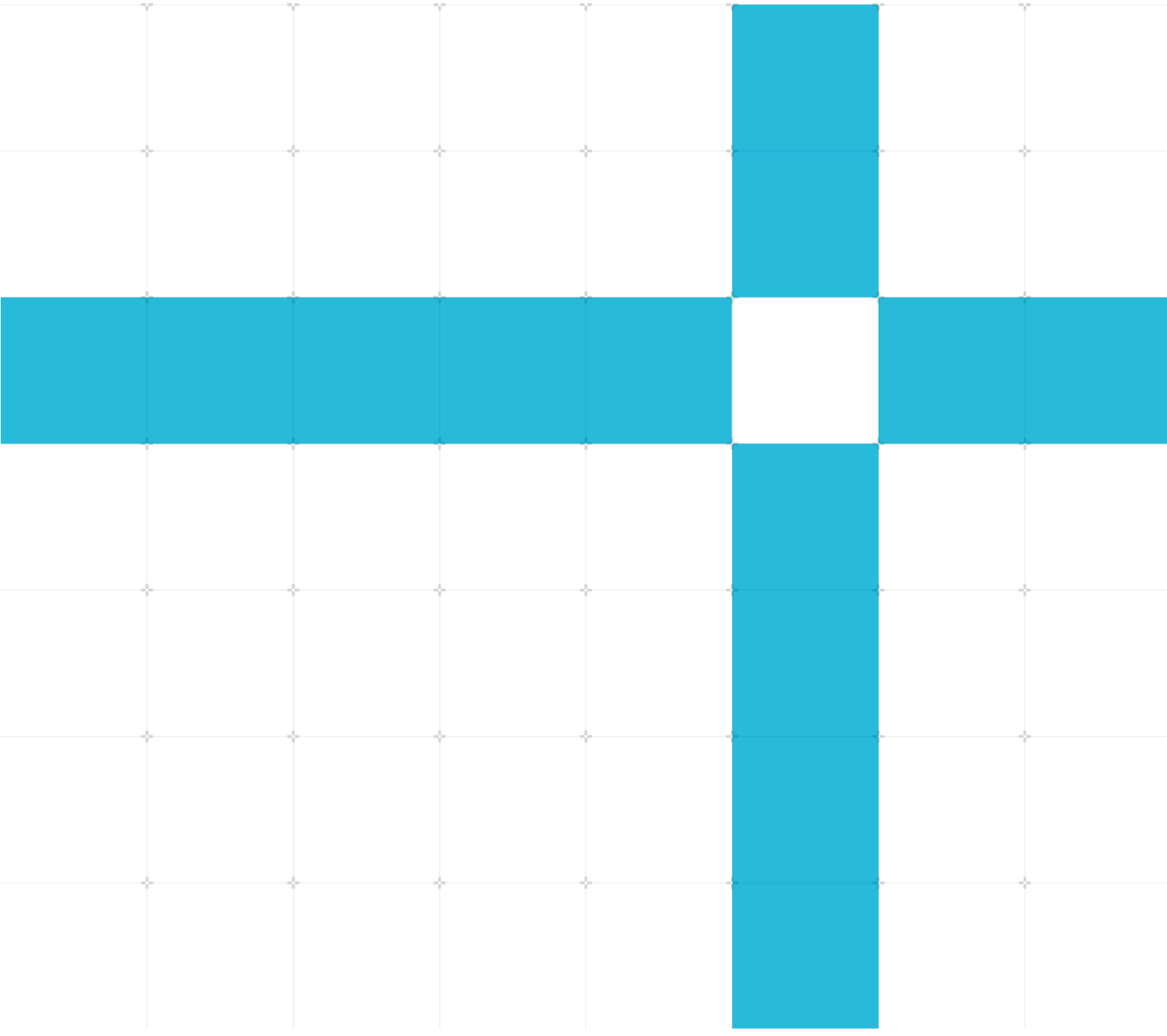
Constrained IoT device demonstration

Non-Confidential

Copyright © 2021 Arm Limited (or its affiliates).
All rights reserved.

Issue 01

Document ID: 102421



Voice on Arm

Constrained IoT device demonstration

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.

Release information

Document history

Issue	Date	Confidentiality	Change
01	18th February 2021	Non-Confidential	First release

Non-Confidential Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word "partner" in reference to Arm's customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its affiliates) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this

document may be the trademarks of their respective owners. Please follow Arm's trademark usage guidelines at http://www.arm.com/company/policies/trademarks.

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Web Address

www.arm.com

Progressive terminology commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used terms that can be offensive. Arm strives to lead the industry and create change.

Contents

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

1 Overview

This guide is for hardware and software architects to learn about setting up voice commands on a constrained IoT device using Alexa Voice Service Integration. The Alexa Voice Service Integration for AWS IoT Core (AIA) offloads memory-intensive and compute-intensive tasks that are needed to integrate with Alexa Voice Services to an Amazon-managed cloud service. Offloading these tasks to the cloud allows constrained devices based on microcontrollers with on-chip memory to support Alexa.

Voice on Arm is a prototyping project that provides a demonstration of the technology and a reference design for voice-activated smart speakers based on Arm Cortex-M series processors. The example presented in this guide provides a starting point to set up a voice-enabled device as a smart speaker and is not intended for production use. In this guide, we will set up voice commands on a constrained IoT device using tap to talk.

This guide shows you how to set up the AIA device with **Amazon FreeRTOS** on a Cypress PSoC 6 Wi-Fi BT Prototyping Kit (CY8CPROTO-062-4343W) based on the **Cypress Amazon FreeRTOS Project**.

1.1 Before you begin

This guide assumes you have:

- A Cypress PSoC 6 Wi-Fi BT Prototyping Kit (CY8CPROTO-062-4343W)
- A USB A to micro USB cable, which is included with the Cypress Prototyping Kit
- An Amazon Web Services (AWS) account
- A host computer with the 64-bit version of Windows, Linux, or MacOS to run:
  - ModusToolbox IDE, which compiles the demo source code and flashes the Cypress board
  - A serial terminal emulator, like Tera Term, which displays output from the Cypress board
- An external audio output module and a speaker. We used a Digilent Pmod I2S2 for this guide.
- Tools to modify the Cypress board:
  - Soldering iron
  - Jumper wire and headers

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Build

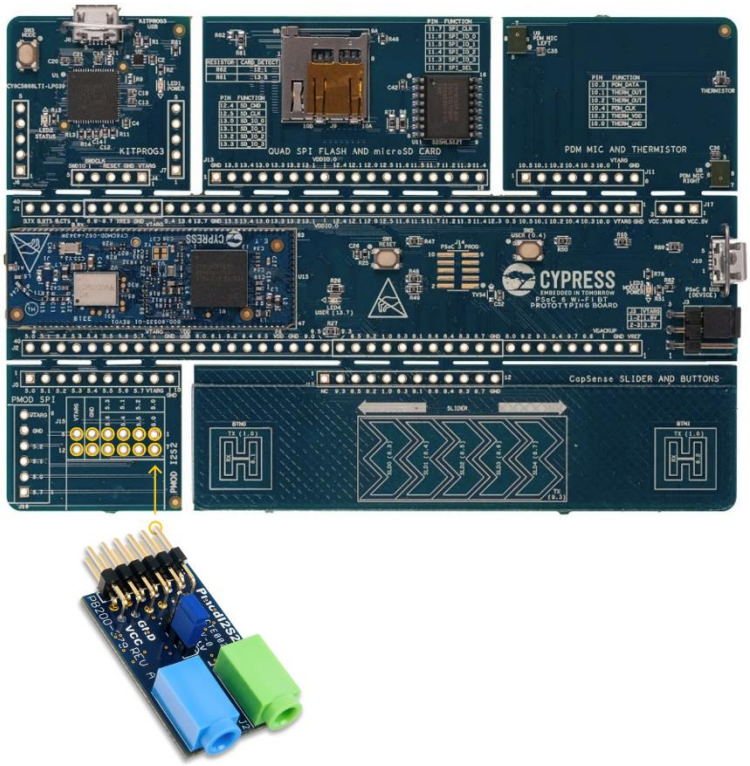
1.2 Prepare the board

Tap to talk uses the CapSense button BTN1 on the PSoC 6 prototyping board to initiate voice commands. The USER LED will be on to indicate that the board is ready to accept voice input and you can tap the button to talk.

In this guide, the audio output uses the Diligent Pmod I2S2 module (CS4344), which requires a Master Clock (MCLK) signal to power up. We will output an MCLK signal on P0.5 generated from the same clock source as the Serial Clock (SCLK) and Left-Right Clock (LRCK) to keep the clocks synchronized.

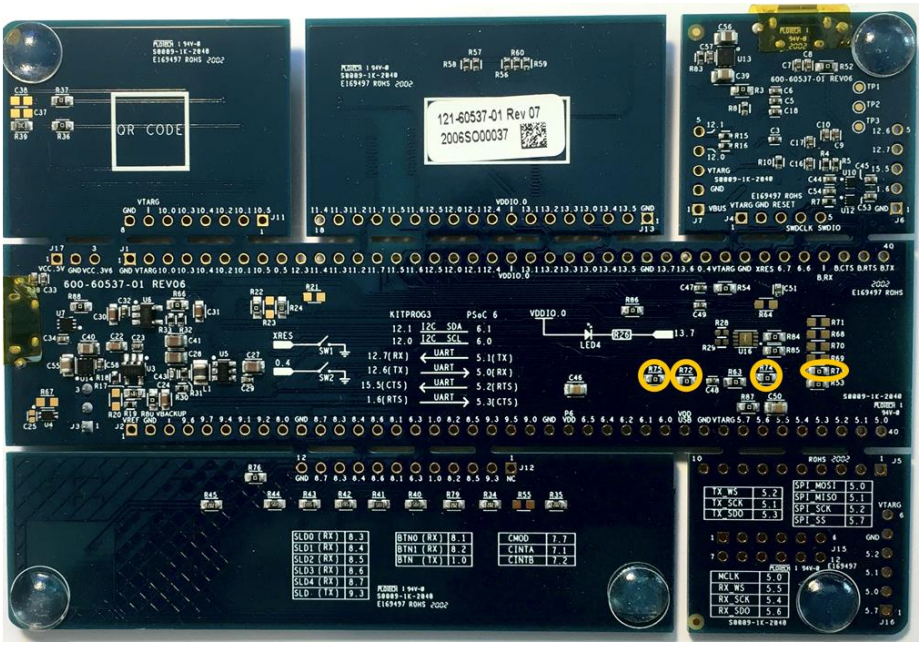
To modify the prototyping board:

1. Connect the Pmod I2S2 to the prototyping board:



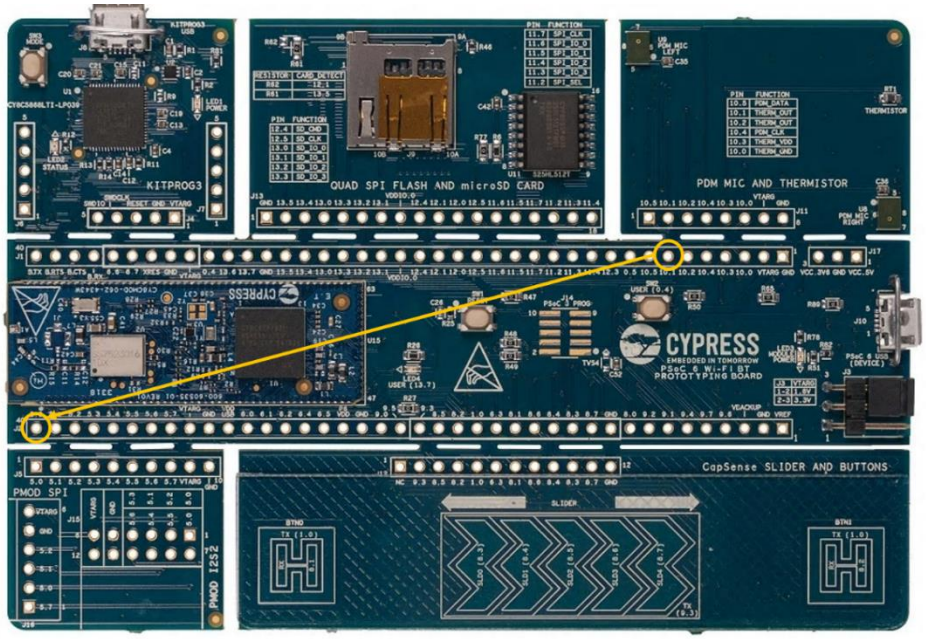
Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

2. Remove resistors R72, R73, R74, and R75 on the back of the board, which are circled in yellow in the diagram below:



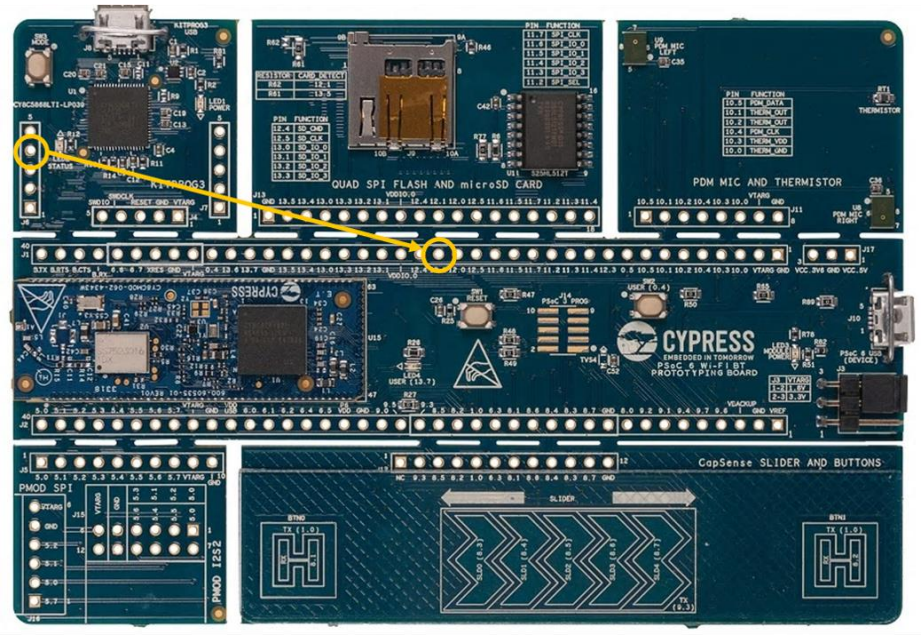
These resistors connect P5.0 - P5.3 to the KitProg3 Universal Asynchronous Receiver/Transmitter (UART) for serial communication. P5.0 to P5.3 must be reserved for the Pmod I2S2.

3. Connect the P0.5 (J1.9) pin to the P5.0 (J2.40 or J5.1) pin with a wire to provide an MCLK for Pmod I2S2:

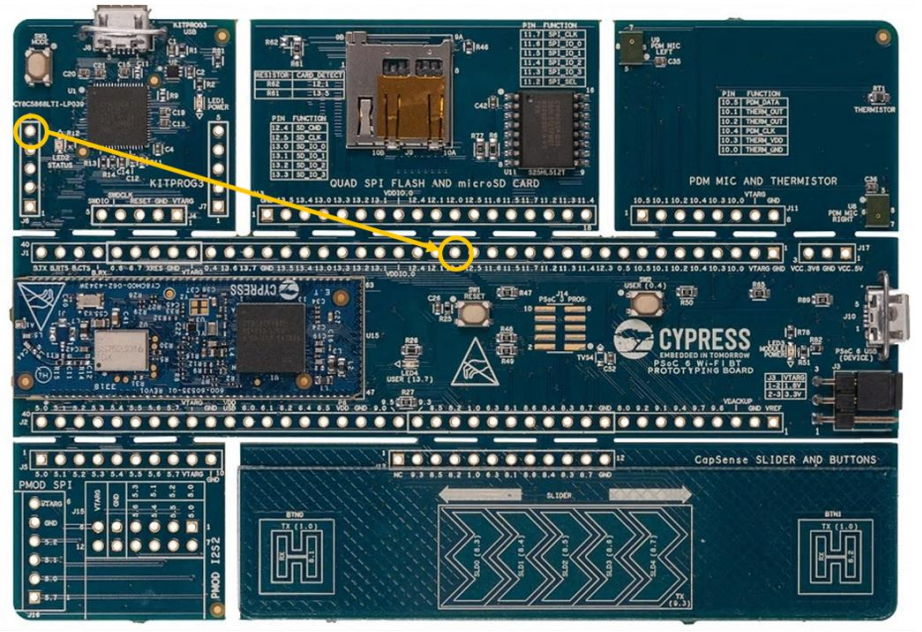


Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

4. Connect the KitProg3 UART_RX (J6.4) pin to P12.1 (J1.19) to set up serial communication:

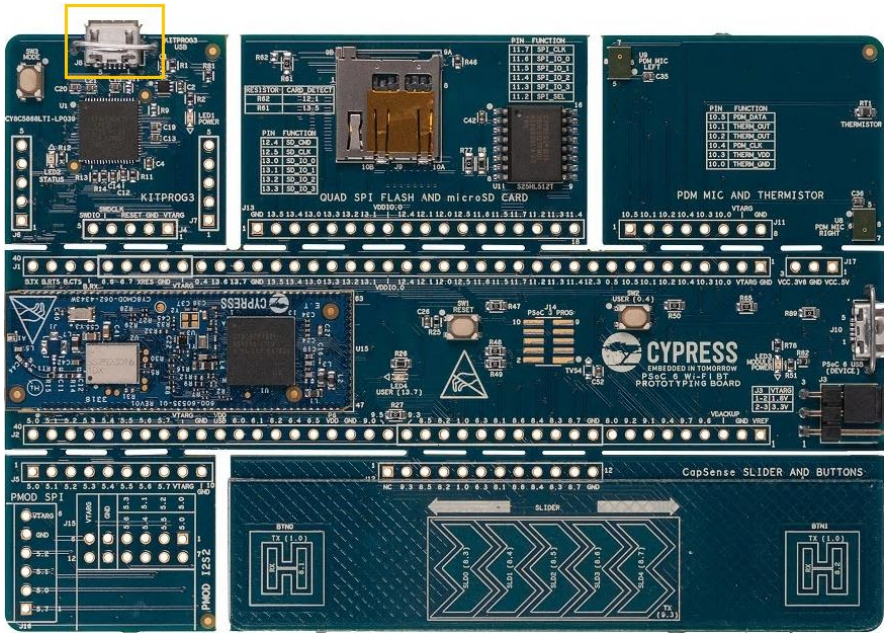


5. Connect the KitProg 3 UART_TX (J6.5) pin J6.5 to P12.0 (J1.18):



Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

6. Connect your computer to the **KitProg3 USB connector (J8)** with the USB A to micro USB cable:



The next step is to download ModusToolbox from Cypress and set up your project files.

1.3 Set up your project

In the following steps, you will download ModusToolbox from Cypress, Opus source code, and files from the Arm GitHub repository for this project.

ModusToolbox is a free development eco-system that includes the ModusToolbox IDE and the PSoC 6 SDK. Opus is an audio codec used for decoding streams from the server.

To create and set up your project:

1. Navigate to the directory in which you want to extract the project files and create a new directory called **`project`**.
2. Download the **`amazon-freertos`** zip or tar file from the Cypress GitHub repository and extract it to your project directory:
   ```
   tar -xvf amazon-freertos-201908-MTBAFR1941.tar.gz
   ```
3. Create a **`demos/aia`** directory in the top-level **`amazon-freertos-xxx`** directory:
   ```
   cd project/amazon-freertos-201908-MTBAFR1941/demos

   mkdir aia
   ```
4. Download the **`MCU-demo-for-AVS-master.zip`** file from the MCU demo for AVS repository and unzip it to **`/tmp`**:
   ```
   cd /tmp
   unzip MCU-demo-for-AVS-master.zip
   ```

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

5. Copy the files from the /**aia** directory and the /**demo** directory into the **demos/aia** directory you just created:

```
cp /tmp/MCU-demo-for-AVS-master/aia/* /<projectdirectory>/project/amazon-freertos-201908-MTBAFR1941/demos/aia/

cp /tmp/MCU-demo-for-AVS-master/demo/* /<projectdirectory>/project/amazon-freertos-201908-MTBAFR1941/demos/aia/
```

6. Download the Opus 1.3.1 source code from the Opus downloads site and extract it to your project directory:

```
tar zxvf opus-1.3.1.tar.gz --one-top-level=libraries/3rdparty/opus --strip-components 1
```

   A new directory **project/amazon-freertos-201908-MTBAFR1941/libraries/3rdparty/opus/** is created.

7. Apply the **CY8CPROTO-062-4343W.patch** file from **MCU-demo-for-AVS-master/patch**. This patch contains necessary changes to the project to run the AIA demo and provides a fix for an issue with the **amazon-freertos-201908.00** release. The Board Support Package (BSP) in the **amazon-freertos-201908.00** release clears the **TOC2_FLAGS** register in SFlash and disables SWJ pins if the MCU is programmed with SFlash area programming permitted. As a result, the Cypress board will be unable to reprogram with KitProg3. This patch sets the **TOC2_FLAGS** to the correct value.

   On Windows, you can use **git apply**.

   If you have Git BASH installed on Windows, you are using Linux, or using MacOS, you can use **patch**.

   Caution: Do not program this release on your board without applying the patch if you have not prohibited SFlash Erase/Program in debugger configurations. If you are unable to reprogram your board, please contact your board provider.

8. If you are using Windows or MacOS, the lwIP library includes a header file **api.h** and the Opus library includes **API.h**. Windows and MacOS are case insensitive for file systems by default. You must specify the path of these two header files in the source files that include them to ensure the correct one is included. To do this, apply the **opus_WINDOWS_MAC.patch** file in the **patch/** directory.

9. Download and install ModusToolbox from the ModusToolbox IDE site.

10. Import the .**project** file from **projects/cypress/CY8CPROTO_062_4343W/mtb/aws_demos/** Go to File > Import and select ModusToolbox > ModusToolbox Application Import from the Import dialog.

11. Click Next and set the project location to **<directory>/project/amazon-freertos-201908-MTBAFR1941/projects/cypress/CY8CPROTO_062_4343W/mtb/aws_demos**.

12. Click Finish.

    A message is displayed in the console to indicate that the process is complete and the **aws_demos** directory appears in Project Explorer.

13. In the Eclipse IDE Quick Panel, launch the Device Configurator. Click File > Save then click File > Exit. This allows the Device Configurator to automatically generate platform configuration source code based on the **design.modus** and **design.cycapsense** files created for this guide.

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

If you do not have the Device Configurator in your Quick Panel, right-click the **aws_demos** project in Project Explorer then click **ModusToolbox** > **Device Configurator 2.20**.

The next step is to set up your credentials to authenticate with the AWS Voice Service.

1.4 Set up credentials

Alexa Voice Service (AVS) uses the Login With Amazon protocol and federated identity scheme to allow smart speaker devices to authenticate with the service.

To begin, you need to register your product with AVS.

To register your product, follow these steps:

1. Follow the **First steps** as described on the AWS site:
   - Setting up your AWS account and permissions. The Quick Connect workflow is not needed for this guide.
   - Registering your MCU board with AWS IoT
2. Log in to the AVS dashboard. If this is the first time registering a product, a welcome screen is displayed. Click **GET STARTED**.
3. Click **PRODUCTS** then **ADD NEW PRODUCT**.
4. Enter a **Product Name**. For example, **AVS Tutorials Project**.
5. Enter a **Product ID**. Do not use any spaces in your product name.
6. In the **Please Select Your Product Type** drop-down menu, select **Device with Alexa built-in**.
7. For **Will your device use a companion app?**, select **No**.
8. In the **Product Category** drop-down menu, select **Smart Home**.
9. Enter a description in the **Brief product description** box.
10. Under **How will users interact with your product?**, select **Touch-initiated**.
11. Skip the **Upload an image step**. This is not required for this guide.
12. Under **Do you intend to distribute this product commercially?**, select **No**.
13. Under **Will your device be used for Alexa for Business?**, select **No**.
14. Under **Is this device associated with one or more AWS IoT Core Accounts?**, select **Yes** then enter your AWS Account ID.
15. Under **Is this a children's product or is it otherwise directed to children younger than 13 years old?**, Select **No**.
16. Click **NEXT**.

AIA and your device use a shared secret for encrypting and decrypting messages. A key exchange is required to retrieve a public key from AIA to generate the shared secret. For more details, see AVS for AWS IoT Registration.

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

This guide uses Code Based Linking for LWA authentication and the registration process is done on a host with HTTP capability. The user private and public key pair and the AIA public key are hardcoded into the source code. A **gen_credentials.sh** bash script is provided to automate this process as much as possible.

Note: In a production scenario, a provisioning application is used instead of this script. For the purposes of this guide, steps are included to run the bash script.

To run the **gen_credentials.sh** script:

1. Enter your information for the following variables at the beginning of the **gen_credentials.sh** script before you run it in a terminal:
   - **AWS_ACCOUNT** using your AWS account ID. To find your ID, log in to AWS. In the top navigation, click your username then select My Security Credentials. The account number appears in Account identifiers if you are the root user, or Account details if you are an IAM user.
   - The **PRODUCT_ID** that you created when you registered your product. The product ID can be found in the Alexa Voice Service Developer Console.
   - The **CLIENT_ID** for your product, which can also be found in the Developer Console. Click on your product and under Product Details, click Security Profile. The Client ID is displayed in the tab Other devices and platforms.

     Note: Do not use the Client ID that is displayed at the top of the page or in the Web tab.
   - The **IOT_THING_NAME** registered with AWS IoT, which can be found in your AWS IoT console. In the console, select Manage >Things from the navigation pane to view a list of your IoT devices. Note that this is the short name, not the fully qualified Amazon Resource Name (ARN).
   - The **IOT_ENDPOINT** your IoT device connects to. This can be found in your AWS IoT console in the Settings menu. The MQTT broker endpoint should point to the Amazon server.
2. Open a bash terminal and enter **./gen_credentials.sh**

   A message is displayed containing the link to log in and your user code. For example:

   ```
   Please navigate to https://amazon.com/us/code, login and enter the user code: BVD3FQ
   After you have finished registration on the above link, hit ENTER to continue...
   ```
3. Navigate to the link displayed and sign in using your AWS account credentials.
4. Enter the code displayed in the bash terminal in the Register Your Device screen.
5. Select your country from the list and click Next.
6. Click Allow to give your project access to Alexa Voice Services and Alexa Account Connection.
7. Press Enter in the terminal to continue running the script. After you run the script, your public key, private key, peer (AIA) public key, and the topic root are created.

To finish entering your credentials, follow these steps:

1. Edit the following in the **aia_client_config.h** file in the MCU-demo-for-AVS/aia/ directory, using the values generated in the **gen_credentials.sh** script:

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

- `aiaconfigAWS_ACCOUNT_ID` To find your AWS account ID, log in to Amazon Web Services. In the top navigation, click your username then select My Security Credentials. The account number appears in Account identifiers (if you are the root user) or Account details (if you are an IAM user).
- `aiaconfigTOPIC_ROOT`
- `aiaconfigCLIENT_PUBLIC_KEY`
- `aiaconfigCLIENT_PRIVATE_KEY`
- `aiaconfigPEER_PUBLIC_KEY`

2. Edit the `aws_clientcredential.h` file in the `project/amazon-freertos-201908-MTBAFR1941/demos/include` directory:
   - The `clientcredentialIOT_THING_NAME` registered with AWS IoT, which can be found in your AWS IoT console. In the console, select Manage >Things from the navigation pane to view a list of your IoT devices. Note that this is the short name, not the fully qualified Amazon Resource Name (ARN).
   - `clientcredentialMQTT_BROKER_ENDPOINT` is the endpoint your IoT device connects to. This can be found in your AWS IoT console in the Settings menu. The MQTT broker endpoint should point to the Amazon server.
   - `clientcredentialWIFI_SSID` is your Wi-Fi network name.
   - `clientcredential_WIFI_PASSWORD` is your Wi-Fi password.
   - `clientcredential_WIFI_SECURITY` is your Wi-Fi security type. Valid values are: `eWiFiSecurityOpen`, `eWiFiSecurityWEP`, `eWiFiSecurityWPA`, or `eWiFiSecurityWPA2`.
3. Edit the `aws_clientcredential_keys.h` file in the `project/amazon-freertos-201908-MTBAFR1941/demos/include` directory:
   - The `keyCLIENT_CERTIFICATE_PEM` value can be found in the AWS console when you create the certificates. Use `tools\certificate_configuration\CertificateConfigurator.html` to get the certificate in the correct format.
   - The `keyCLIENT_PRIVATE_KEY_PEM` value is from the AWS console, when you create the certificates. Use `tools\certificate_configuration\CertificateConfigurator.html` to get the certificate in the correct format.

Your device is now registered and ready to use with AWS. The next step is to compile and flash the device.

1.5 Compile and flash the device

To build your project, follow these steps:

1. Click aws_demos in the Project Explorer in ModusToolbox.
2. Click Build aws_demos Application in the Quick Panel.

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

The default GCC compiler used by the Cypress Amazon FreeRTOS project is **gcc-7.2.1** in the ModusToolbox installation location. If the compiler path does not match the path installed by your ModusToolbox IDE, the build will fail.

If you receive a build error, specify the correct GCC path:

- Set **CY_COMPILER_DEFAULT_DIR** in **vendors/cypress/psoc6/psoc6make/make/core/main.mk**

Or

- Set **CY_COMPILER_DIR and CY_COMPILER_GCC_ARM_DIR** in **projects/cypress/CY8CPROTO_062_4343W/mtb/aws_demos/Makefile**

To flash the device, follow these steps:

1. Click aws_demos in the Project Explorer in ModusToolbox.
2. Click aws_demos Program (KitProg3) in the Quick Panel Launches window.
   The IDE console displays the flashing progress.

The next step is to try out voice commands and check the connection to the board.

1.6 Test your project

Now you are ready to use voice commands with your prototyping board. Tap the BTN1 button in the bottom-right corner on the board to start the conversation.

Connect a serial terminal to the UART, using a serial terminal emulator like Tera Term. Logs are available in the console if you set up the UART as described in Prepare the board. In the console, you can see the device connect to your MQTT endpoint and AIA service, and the device waits in IDLE state for your action. Alternatively, if you have not enabled a UART connection, the device status is indicated on the LED on the board. The LED blinks after the MQTT connection is established and stays on after entering an IDLE state.

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

2 Related information

Here are some resources related to material in this guide:

Webpages

- Endpoint AI webpage
- Voice Recognition page on Developer
- NXP's Voice Control webpage

Case Studies, White Papers and Report

- Sensory's TrullyHandsFree Case Study
- NorthStar Report: Machine Learning has found its voice
- White Paper: The New Voice of the Embedded Intelligent Assistant
- Case Study: Sensory's TrulyHandsfree and Arm's Cortex-M55: Achieving High Wake Word Accuracy with Less Resources for Low-power, Customizable Voice Control

Blogs

- Arm Blueprint - Taking Voice AI to the Next Level
- Arm Blueprint - Partner Q&A: Snips Voice AI Platform
- Fluent.ai efficient multilingual speech recognition and understanding on Arm Cortex MCUs
- Accelerating innovation: transforming voice-based products for the future
- How to implement voice and audio processing on Arm with Alango Technologies

Videos

- Endpoint AI videos
- Speech recognition on Arm Cortex-M by Fluent.ai
- Cost Effective Solution to Bring Alexa to Your Connected Devices
- Implementing Cortex-M55 for Endpoint AI with Cambridge Consultants
- Demystify artificial intelligence on Arm MCUs by Cartesiam

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

3 Next steps

This guide provides an example starting point to set up a voice-enabled device using a smart speaker. After you have familiarized yourself with this guide, you can evaluate the work required for your use case.

The next step is to explore the Voice on Arm resources and decide whether you can use it as the basis for your SoC.

Copyright © 2021 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential