Arm Virtual Hardware Features

Version 1.0

Knowledge Base Article

Non-Confidential

Copyright © 2022 Arm Limited (or its affiliates).
All rights reserved.

Issue 01
107675_0100_01_en

Arm Virtual Hardware Features

Knowledge Base Article

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.

Release information

Document history

Issue	Date	Confidentiality	Change
0100-01	9 November 2022	Non-Confidential	First release

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.

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 https://www.arm.com/company/policies/trademarks.

Copyright © 2022 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|version 21.0)

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.

Product Status

The information in this document is Final, that is for a developed product.

Feedback

Arm® welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on https://support.developer.arm.com

To provide feedback on the document, fill the following survey: https://developer.arm.com/documentation-feedback-survey.

Inclusive language commitment

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

We believe that this document contains no offensive language. To report offensive language in this document, email terms@arm.com.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Contents

1. Generating an AVH API Token....7

2. Running Docker....8

3. Service IP and LAN IP....11
3.1 Services IP....11
3.2 LAN IP....11

4. Power and Action Buttons....12
4.1 Power and Action Buttons....12
4.2 Power On and Off....12
4.3 Pause....13
4.4 Restart....13
4.5 Delete....13

5. Connect to your virtual devices....14
5.1 Connect via VPN....14
5.2 Connect via GDB....14
5.3 Connect via Serial Console....15
5.4 Connect via other methods....15

6. Sensors....16
6.1 Environment....16
6.2 Motion and position....16

7. Console....17

8. CoreTrace....18
8.1 What is CoreTrace?....18
8.2 Setting up CoreTrace....18
8.3 Threads and Processes....19
8.4 Understanding the results....20

9. Connect to Wi-Fi on the Raspberry Pi 4....21

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

10. Communicate Using Bluetooth Low Energy..... 32
10.1 Use BLE Between Two Raspberry Pi Devices..... 32
10.2 Set Up the Raspberry Pi 4 BLE Node..... 32
10.3 Connect to the Node with the Raspberry Pi 4 BLE Master..... 46
10.4 Use BLE Between a Raspberry Pi 4 Board and an Android Device..... 58
10.5 Set Up the Raspberry Pi 4 BLE Node..... 58
10.6 Connect to the Node with the Android BLE Master..... 60

11. Detect Audio through the Microphone with the STM32U5 Board..... 71
11.1 Build the ADF_AudioSoundDetector Firmware File..... 71
11.2 Boot the Custom Firmware and Detect Audio..... 77

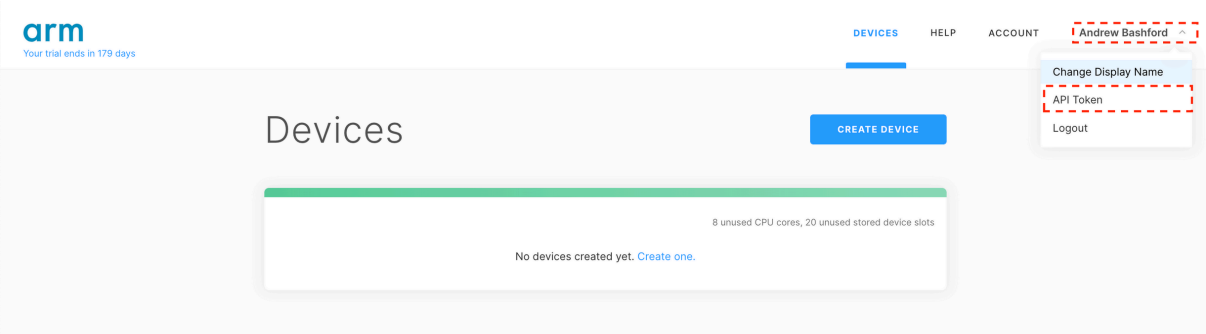
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

1. Generating an AVH API Token

This article demonstrates how to create and retrieve your API Token. This token can be used to authenticate your user for API access. Make sure to keep it secure. You can only have 1 token per user at any one time.

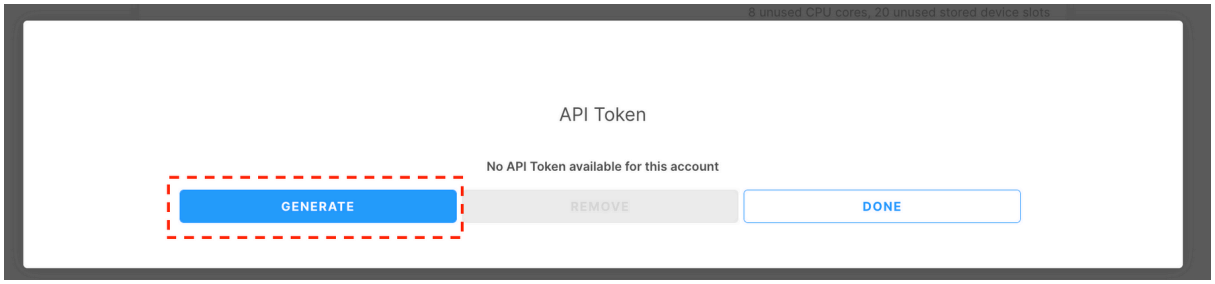
1. Log in to AVH with your Arm user account.
2. To retrieve your API Token, click on your name at the top right corner. Then click “API Token”

Figure 1-1: API token



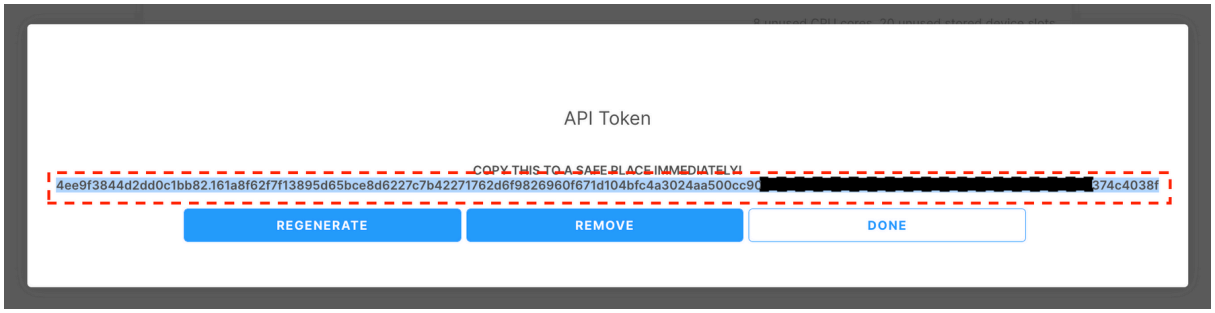
3. Click “GENERATE” this will generate your API Token.

Figure 1-2: Generate API token



4. You can now see your API Token. You will not be shown this token again, copy it and put it somewhere secure to use. Click “DONE”

Figure 1-3: Copy API token



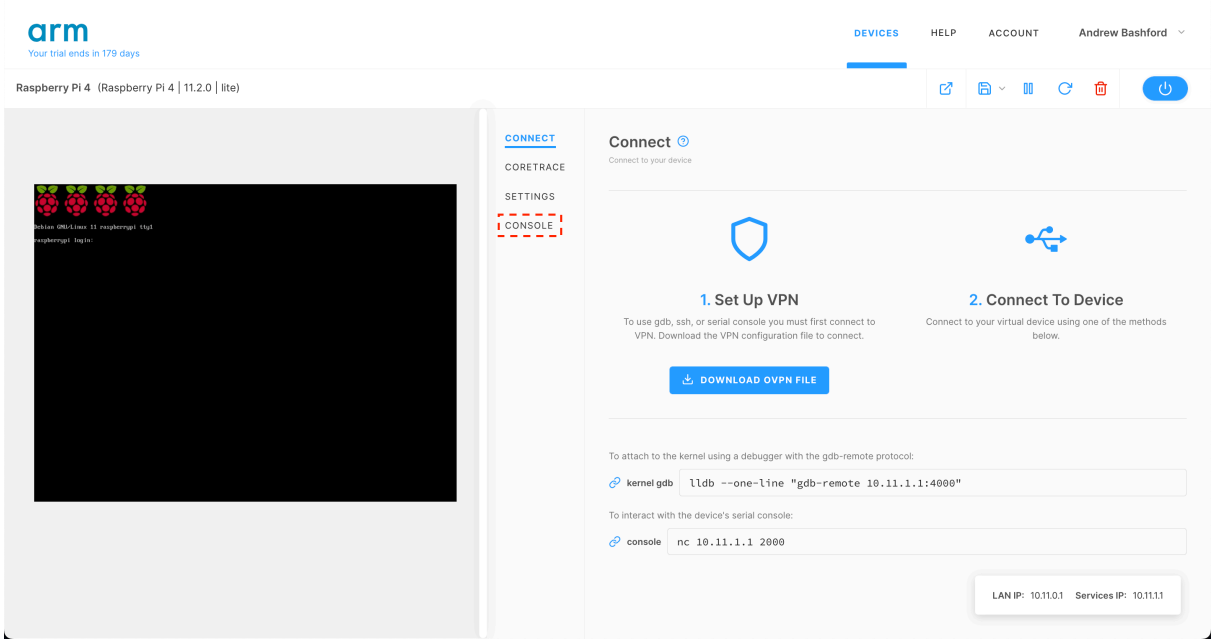
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

2. Running Docker

This tutorial will demonstrate running Docker on a Raspberry Pi 4.

1. Log in to AVH.
2. Select your Raspberry Pi Device. If you haven't got one you can use the Quickstart for AVH (Raspberry Pi4) to create a Raspberry Pi Device.
3. Select CONSOLE tab.

Figure 2-1: Select CONSOLE tab

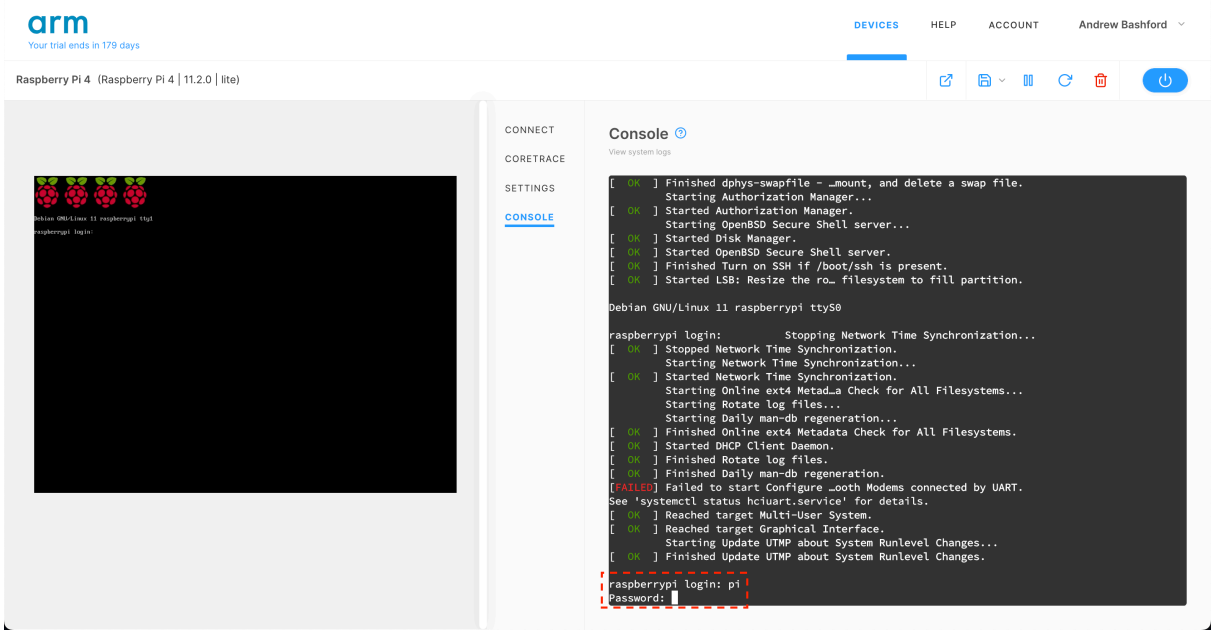


4. Log in with the user: `pi` and password: `raspberry`

   If it is not prompting you for the user, press your `Return` key.

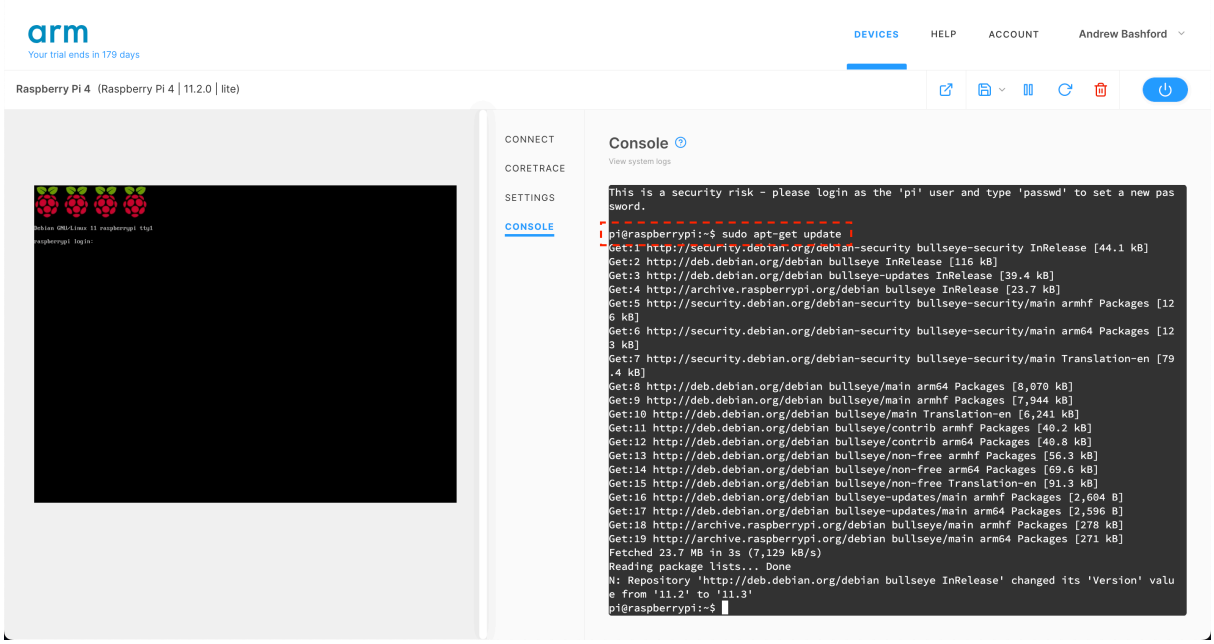
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 2-2: Log in



5. Type `sudo apt-get update` then press `return`.

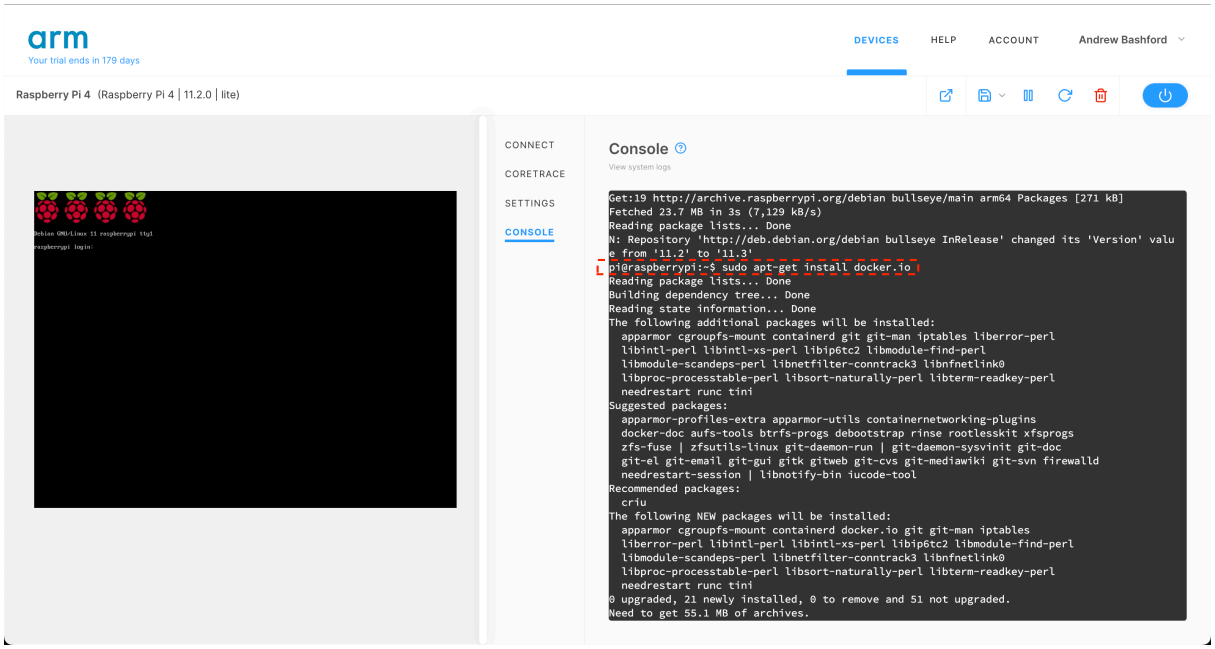
Figure 2-3: sudo apt-get update



6. Now type `sudo apt-get install -y jq` `docker.io` then `return`.

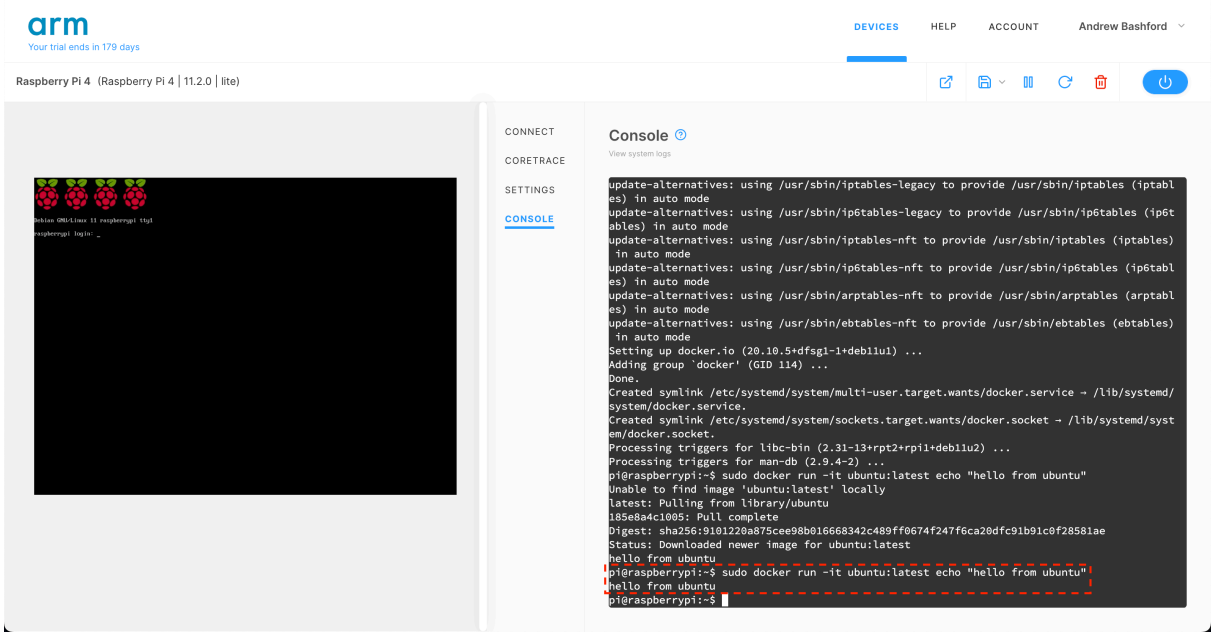
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 2-4: sudo apt-get install -y jq docker.io



7. Try Docker, type `sudo docker run -it ubuntu:latest echo "hello from ubuntu"` then `return`.

Figure 2-5: Try Docker



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

3. Service IP and LAN IP

This article describes the Service IP and LAN IP.

3.1 Services IP

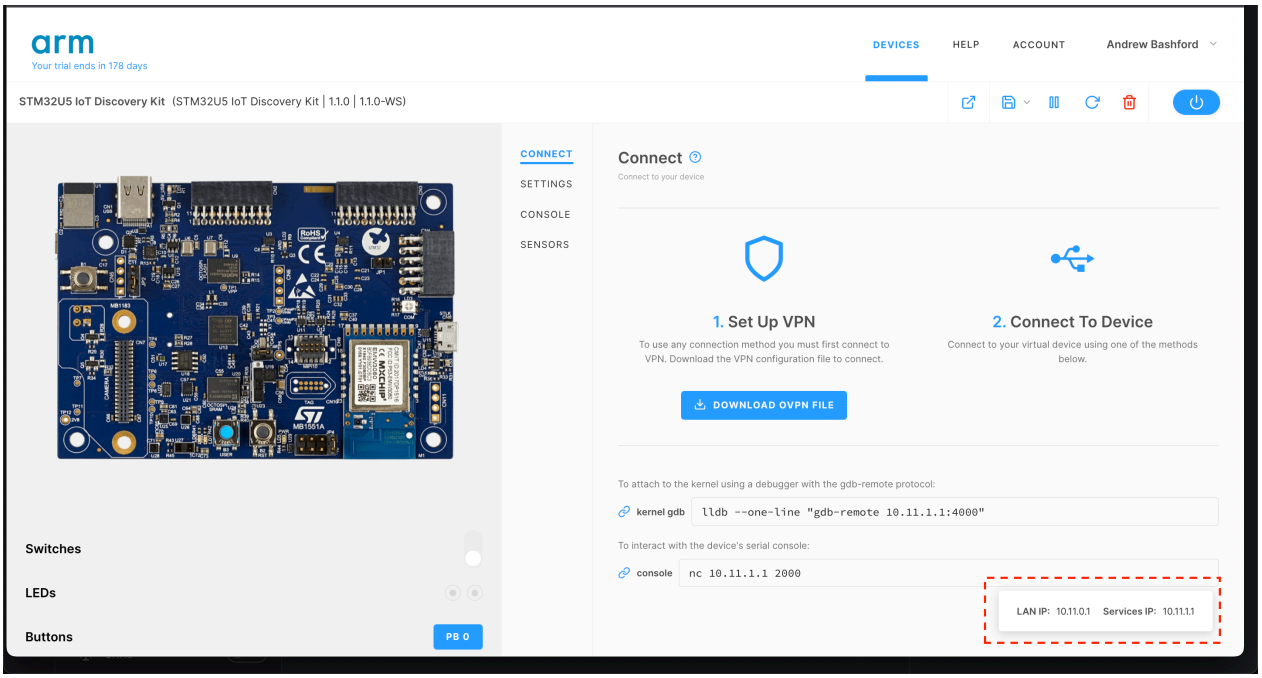
The services IP is a network address used by the Virtual Hardware manager. Virtual services such as serial port (console) pass-through or vMMIO use it. The service IP is used by the Virtual Hardware infrastructure.

3.2 LAN IP

The LAN IP is a network address used by programs running inside the Virtual Hardware. For instance, if there is a web server or SSH server running on the virtual device, it will use the LAN IP.

Both can be found on either the CONNECT or SETTINGS menus of a device.

Figure 3-1: LAN IP and Services IP



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

4. Power and Action Buttons

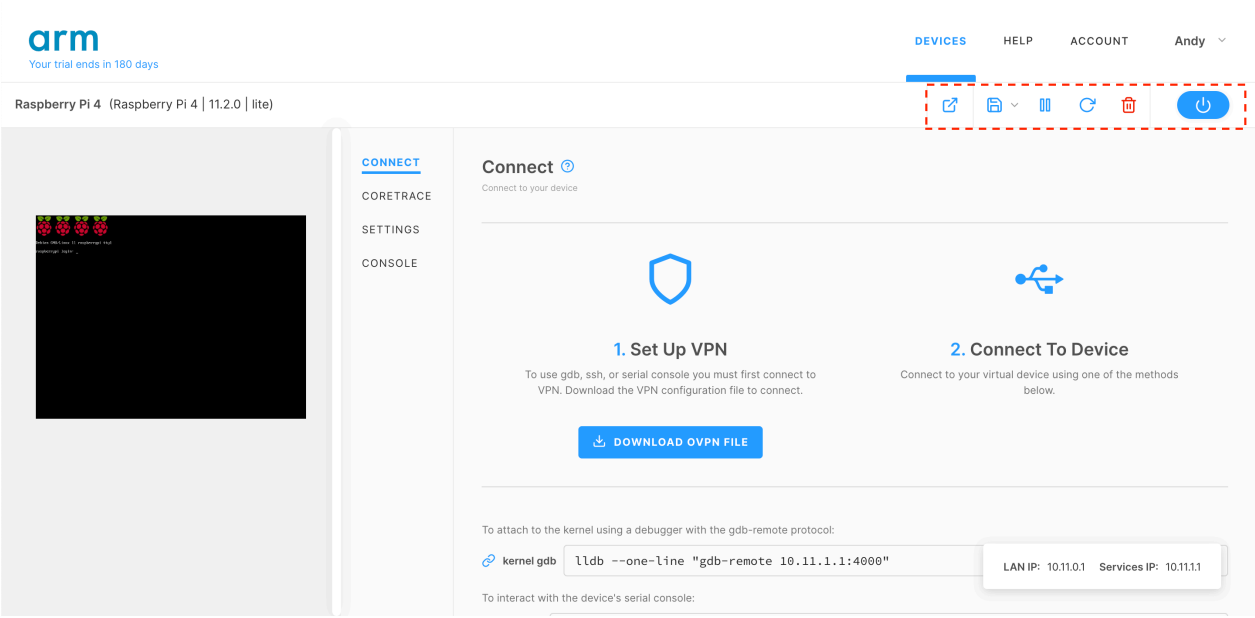
This tutorial will demonstrate how to use your device's action buttons.

- Power and Action Buttons
  - Power On and Off
  - Pause
  - Restart
  - Delete

4.1 Power and Action Buttons

Power and action buttons can be found at the top right when you are in a Device.

Figure 4-1: Power and action buttons



4.2 Power On and Off

Powering the virtual device off is like powering off a physical device. All applications are shut down, and information about the current state of the device is not preserved. When a device is in the Off

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

state, it becomes Stored, and it no longer occupies CPU cores. When you are not using the devices it is best practice to turn them off.

Figure 4-2: Power On / Off button

4.3 Pause

Unlike a physical device, your virtual device can be paused. This action suspends device activity but does not shut down the device. Simply click to pause, and click again to unpause.

Figure 4-3: Pause button

4.4 Restart

The restart button is akin to rebooting a physical device. It's the same as powering the device off and on again.

Figure 4-4: Restart button

4.5 Delete

Deleting a device will permanently delete the device along with all of its snapshots.

Figure 4-5: Delete button

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

5. Connect to your virtual devices

You can connect to your virtual devices via a few methods.

5.1 Connect via VPN

For security, some connections require that you first connect to the virtual device via VPN. Just download the provided configuration file, install it, and you're ready to go.

To connect to VPN, you will need to have a VPN client installed. If you don't already have a VPN client, we recommend an OpenVPN client. Once your VPN client is installed, you can simply download the OpenVPN configuration file from the Connect tab and install it to configure your VPN.

Important Notes

- VPN will only establish connection if you have at least one device in your project that is in the on state. If all your devices are off, or if you have no devices, the VPN will not establish connection.
- Running other VPNs may interfere with your VPN connection. If you are having trouble connecting to the VPN and are already running your own VPN, you may need to disable the other VPN and then reconnect to the VPN.

Troubleshooting Tips

- If you're having trouble connecting, try re-installing your VPN Profile.
- Make sure you're connecting to the right profile for your project, and that at least one device in your project is On.
- Close any other VPNs.
- Check that your network doesn't have a firewall preventing your connection. Learn more about configuring Windows Defender Firewall.

5.2 Connect via GDB

You can connect to the stub via GDB with the address and port for your particular virtual device, located at the bottom of the virtual device page.

Make sure to use the correct version of GDB for your device. 32-bit devices, including Cortex-M based ones, require 32-bit Arm GDB, and 64-bit devices require a 64-bit one.

You can also use other debuggers compatible with the GDB protocol, for instance lldb and IDA debugger.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

5.3 Connect via Serial Console

If you prefer to view the device's serial console input / output in your local console program, such as Terminal, you can copy and paste the Console link from the Connect tab to your local console app. This can be convenient for enabling advanced console features, such as search.

5.4 Connect via other methods

Some virtual devices also support connect methods such as Iris and Cadi.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

6. Sensors

Test devices in real world conditions by configuring a wide variety of environmental sensors and controls.

6.1 Environment

We've added new realism to our virtual devices with an array of new environmental sensors. While you can't hold a cloud device in your hands, you can still make the device behave as if you can.

- Configure the ambient light level (in lux)
- Configure the atmospheric pressure (in hectopascals)
- Configure the temperature (degrees Celsius) and humidity (% RH)
- Simulate presence of objects in front of a proximity sensor (distance in millimeters)

6.2 Motion and position

Simulate your device in motion as it moves around the virtual world.

- Set instantaneous accelerometer readings readings (3-axis, in meters/second^2)
- Set instantaneous gyroscope readings (3-axis, in radians/second)
- Set instantaneous magnetic compass readings (3-axis, in microteslas)

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

7. Console

Console provides convenient access to the virtual device's console.

You can use the Console to quickly see system and kernel logs and to quickly run commands without needing to connect over various methods.

The console can sometimes be flooded with system and kernel logs, making it difficult to use interactively. If you're using the console interactively, you may want to run `dmesg -n 1` to prevent all messages, except emergency (panic) messages. Run `dmesg -n 4` to restore the default behavior, and run `dmesg -n 7` to log all messages to the console.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

8. CoreTrace

CoreTrace is our system call capture tool that offers a quick way to understand a program's behavior.

8.1 What is CoreTrace?

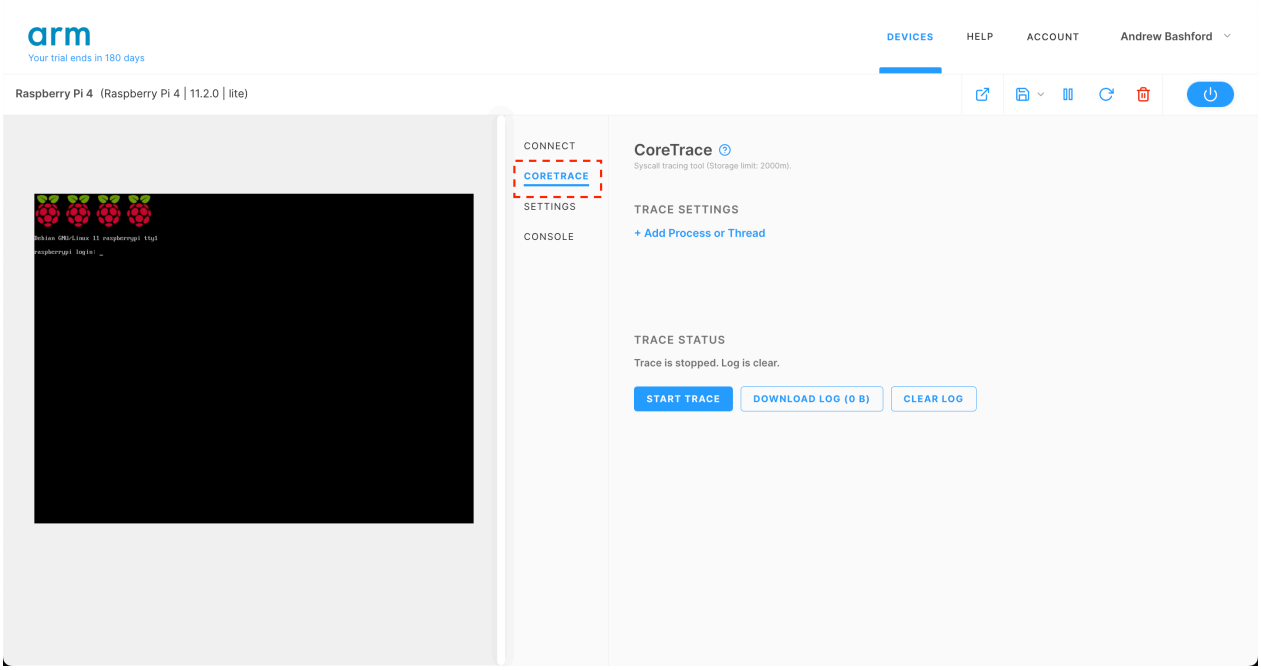
You can trace system calls using either strace, a standard command-line Linux tool, or our proprietary CoreTrace tool. strace is included in our virtual devices, and it is implemented with ptrace. Applications can employ anti-debugging techniques to detect and prevent ptrace-based tracing. However, these techniques cannot prevent, or even easily detect, hypervisor-based tracing. Additionally, you may often be interested in a particular target. CoreTrace makes it easy to filter by specific processes and threads for more targeted analysis.

Tracing system calls is a dynamic analysis reverse engineering technique that can offer a quick way to understand a program's behavior.

8.2 Setting up CoreTrace

To access CoreTrace, open the CoreTrace tab in the device screen. The CoreTrace UI allows you to start and stop a trace, download the log generated by tracing, and clear the log.

Figure 8-1: CoreTrace tab



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

8.3 Threads and Processes

By default, CoreTrace traces all threads in the system. This rapidly produces a huge amount of data. Often you'll be interested in a particular target. To apply a filter to the results, click "Add a process or thread" to display the Processes dialog:

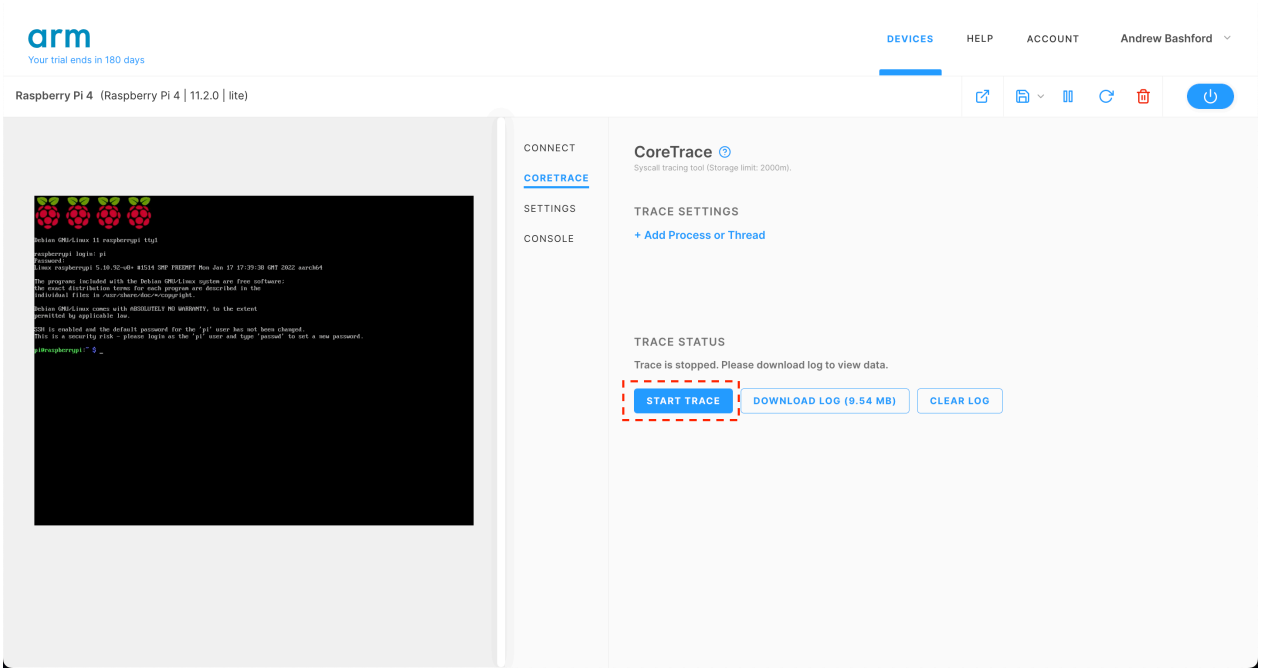
The Processes dialog displays all processes and threads in the system. To examine the threads inside a process, click the "THREADS" button in the process' row.

To add a filter, click the "ADD" button in the process' or thread's row. Alternatively, specify a filter manually. Trace will log traces as long as they match at least one filter.

There are often many processes running. To more easily find the processes and threads you're interesting in, click the magnifying glass in the top-right corner of the dialog and type a phrase. Only rows that contain the phrase will be displayed.

Then, you are ready to click Start Trace.

Figure 8-2: Start Trace



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

8.4 Understanding the results

After you have captured the trace (or during the capture) you can download the log file. Each line of the log will look like this:

```
1<1> [00248.864651618] ffffff806401e040-0/337:sensors@2.1-ser.379/ @00000070efc0b834
 read ( fd: 5, buf: 0x6e5f6ec980, count: 4 ) ... @[ 0000006e5f6f9778
 0000006e5f6f9840 0000006e5f6f948c 0000006e5f6f7f90 0000006e5f6f7b54
 0000006e5f6f7434 00000070efc2088c 00000070efbc0e0c ]
```

or like this:

```
1<1> [00248.864656648] ffffff806401e040-0/337:sensors@2.1-ser.379/
 @00000070efc0b834 ... read ( result: 4, buf: 0x6e5f6ec980 -> [s"001e"] )
```

The fixed line header contains the following information:

```
1<cpu> [time.nsec] threadid-sigid/pid:comm.tid/ @pc
```

Where:

- *cpu* is the processor core the log comes from,
- *time.nsec* is time the entry was captured by the hypervisor,
- *threadid* is the internal kernel thread ID (usually address of a task or thread structure)
- *sigid* is the signal state (if a signal happens, a thread could execute in a different signal state before it's done with the signal, then return to the original signal state),
- *pid* is the process ID (PID of the process on Linux),
- *comm* is the short process name, which may be the original command but may also be set by process itself,
- *tid* is the thread ID (or, PID of the thread),
- *pc* is the PC where the syscall happened in EL0 (userland).

After the header, lines that end with ... are syscall invocations and lines that start with ... are syscall returns. On syscall invocation lines, if the environment permits it, there will be an additional trailer of the form:

```
1@[ lr ret1 ret2 ret3 ... ]
```

This trailer contains the EL0 return stack of the function that invoked the syscall.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

9. Connect to Wi-Fi on the Raspberry Pi 4

The virtual Raspberry Pi 4 board connects by default over the ethernet interface.

To set up a connection over Wi-Fi, perform the following:

1. Create a Raspberry Pi 4 device with the “lite” firmware version and log in using the default credentials `(pi/raspberry)`.

   Figure 9-1: Raspberry Pi login

```
[  OK  ] Finished Update UTMP about System Runlevel Changes.
[  OK  ] Finished Daily man-db regeneration.

raspberrypi login: pi
Password:
Linux raspberrypi 5.10.92-v8+ #1514 SMP PREEMPT Mon Jan 17 17:39:38 GMT 202
2 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been chan
ged.
This is a security risk - please login as the 'pi' user and type 'passwd' t
o set a new password.


Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@raspberrypi:~$
```

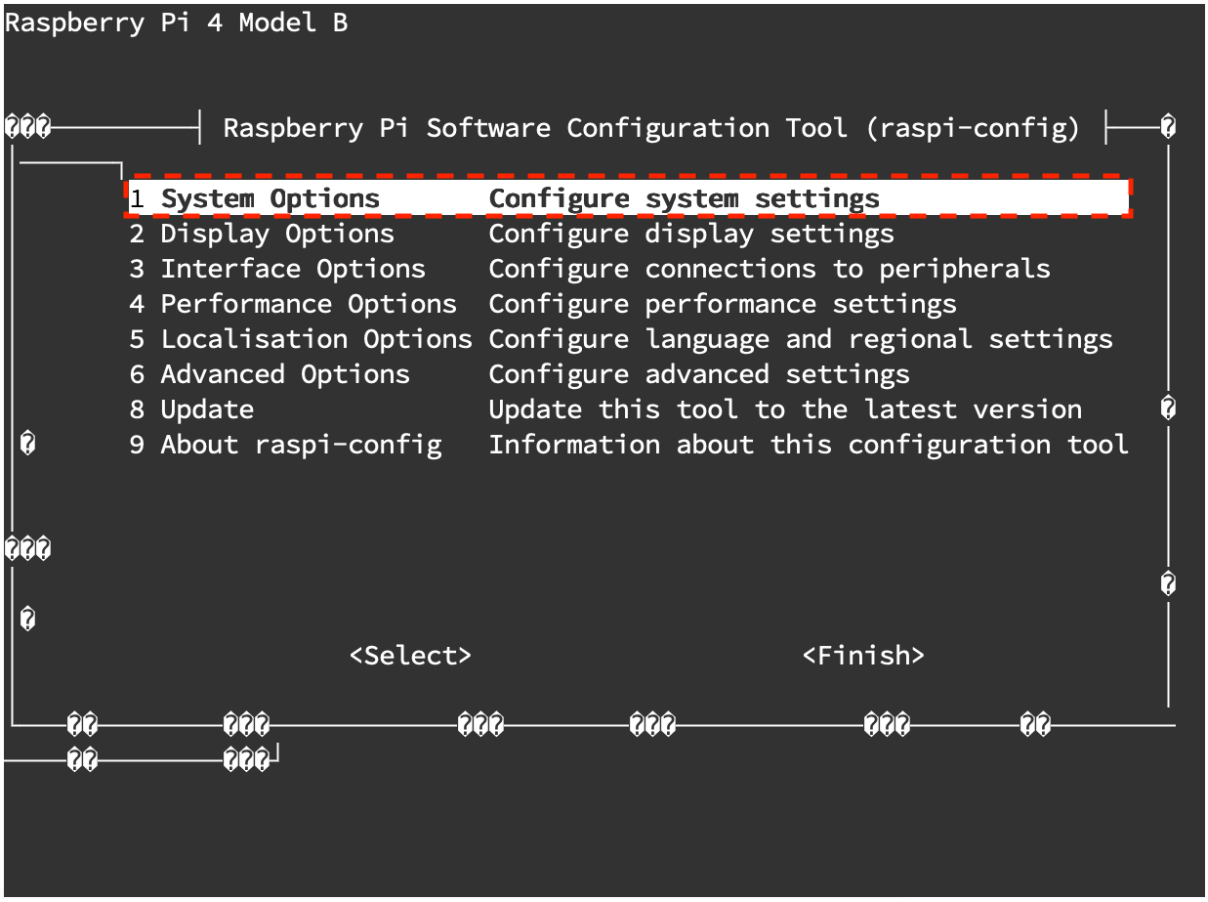
2. Open the configuration tool.

   `sudo raspi-config`

3. Select `System Options` (menu item 1).

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

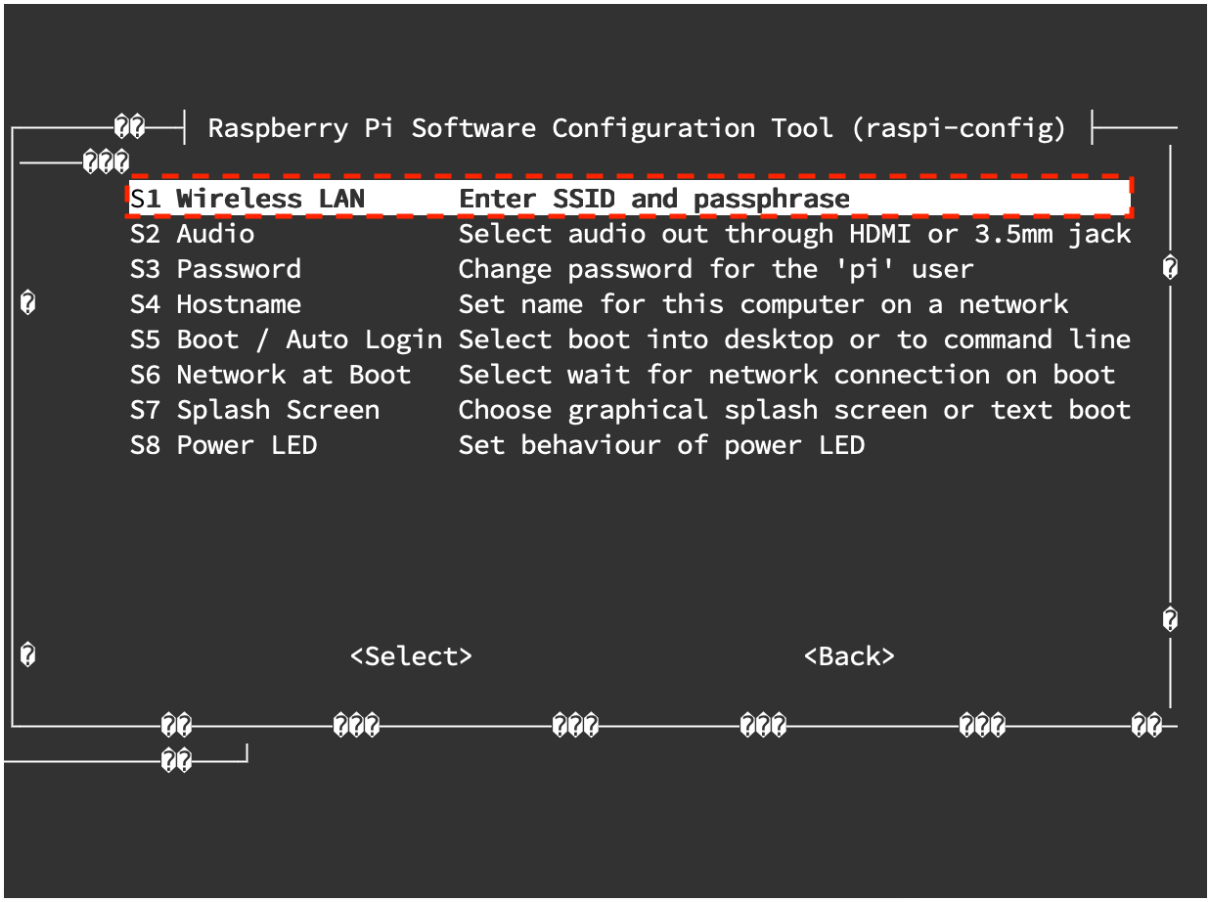
Figure 9-2: Configure main screen



4. Select `Wireless LAN` (menu item S1).

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

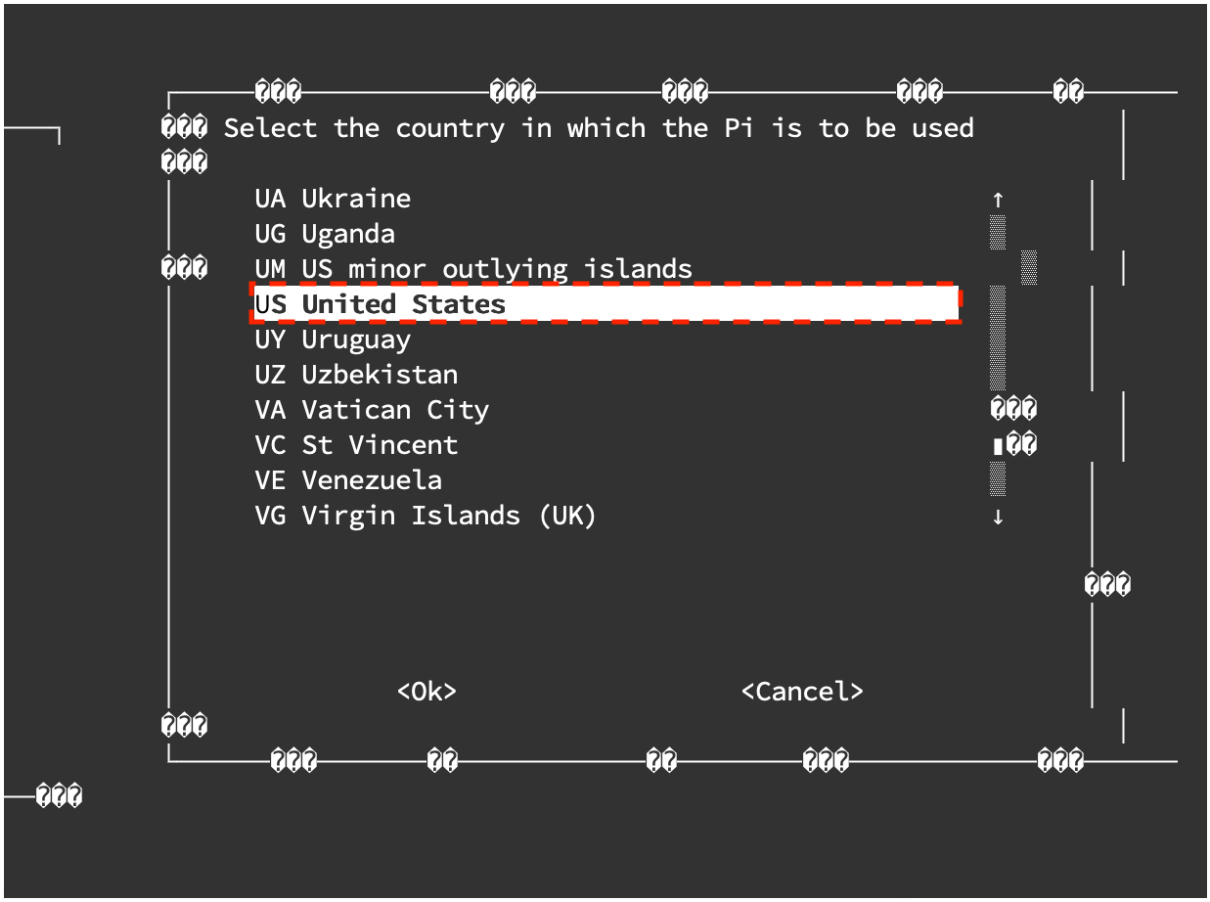
Figure 9-3: Configure system options



5. Set the country. In our example, we will use the United States. (Type “u” to quickly navigate the menu).

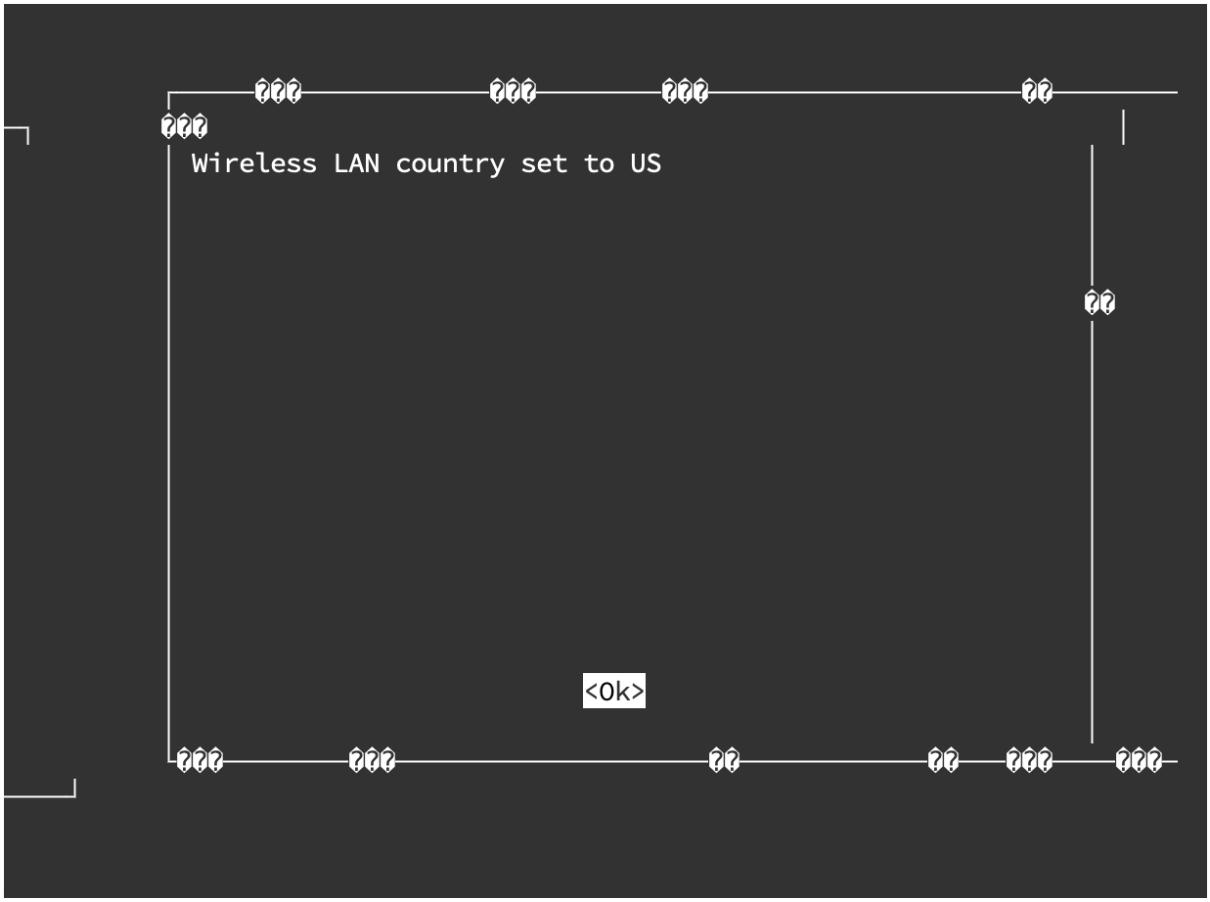
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 9-4: Set coutry



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

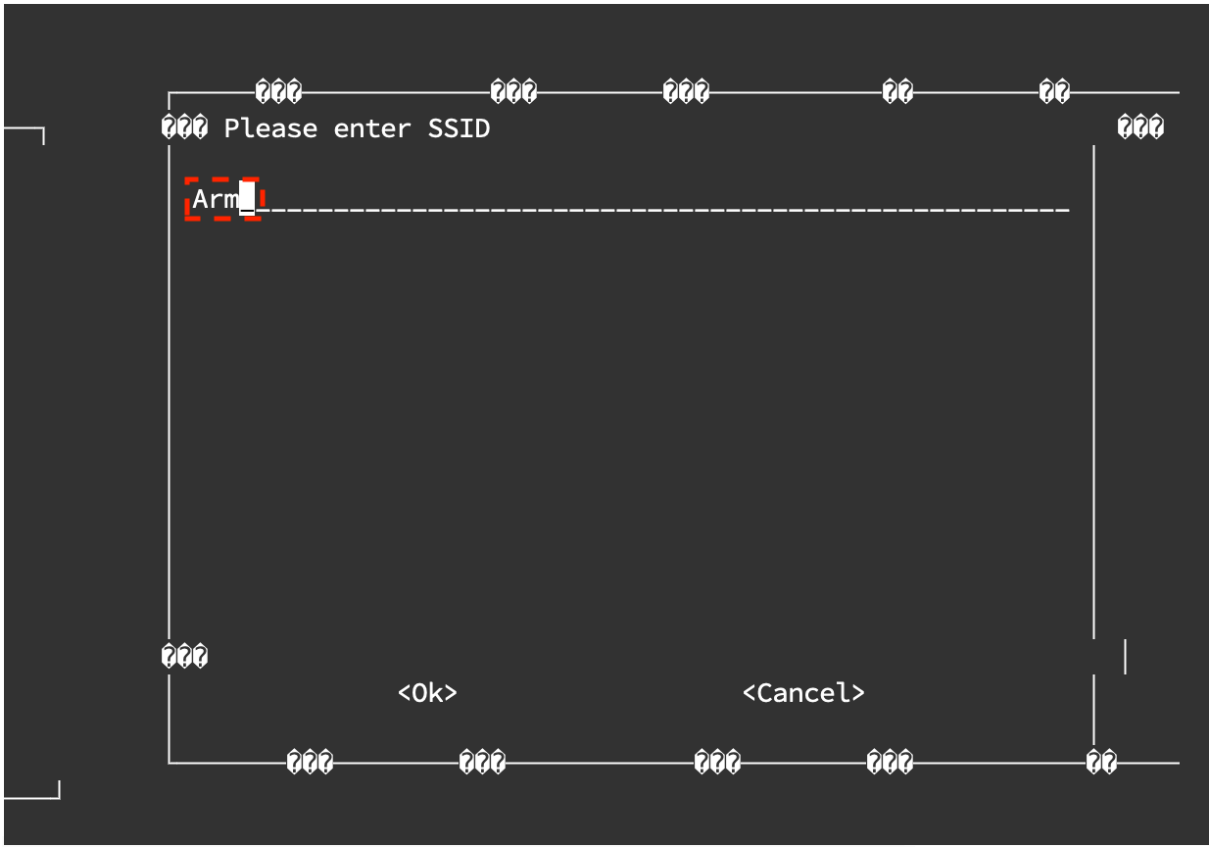
Figure 9-5: Set coutry confirmation



6. Set the SSID to Arm and the passphrase to `password`.

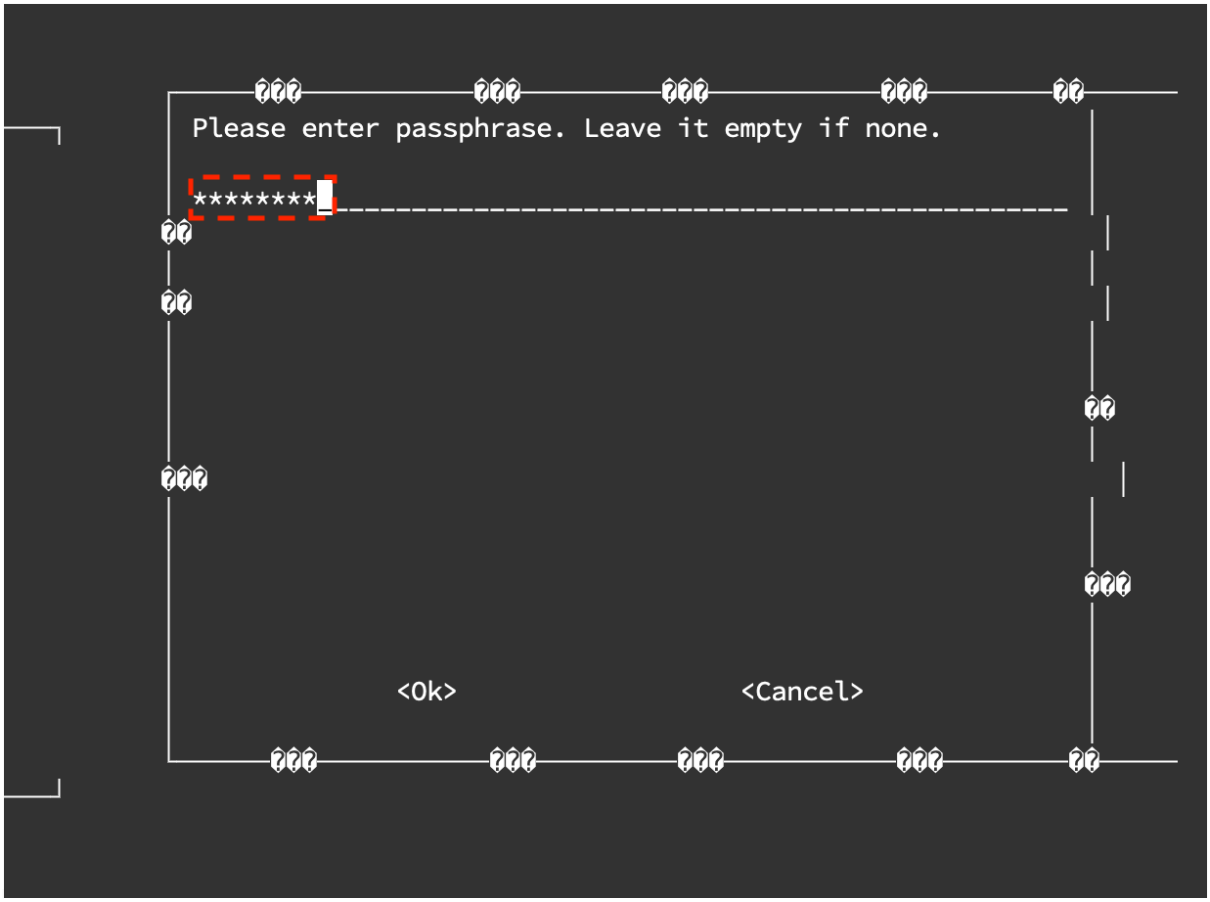
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 9-6: Set SSID



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

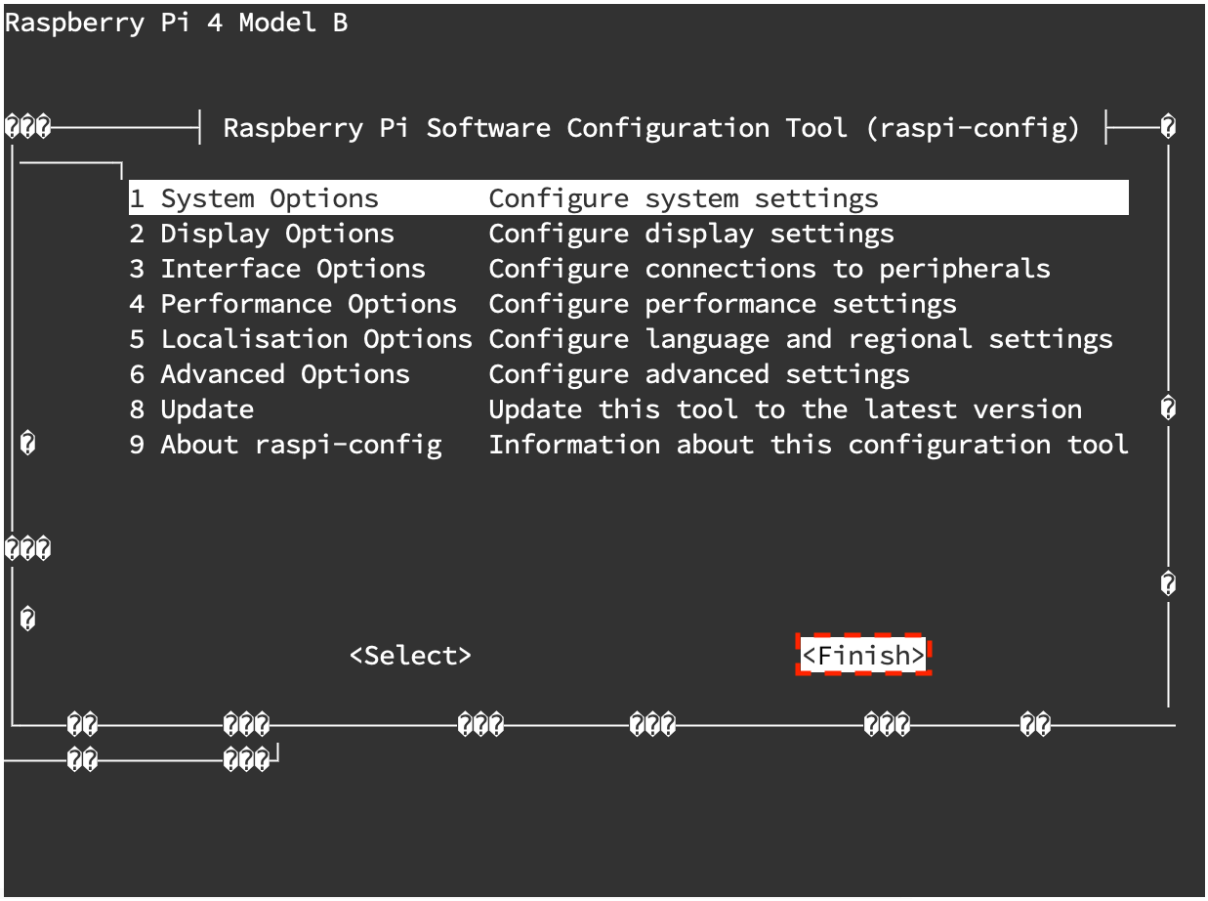
Figure 9-7: Set passphrase



7. At the main configuration screen, choose Finish. (Push the right arrow on your keyboard twice).

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

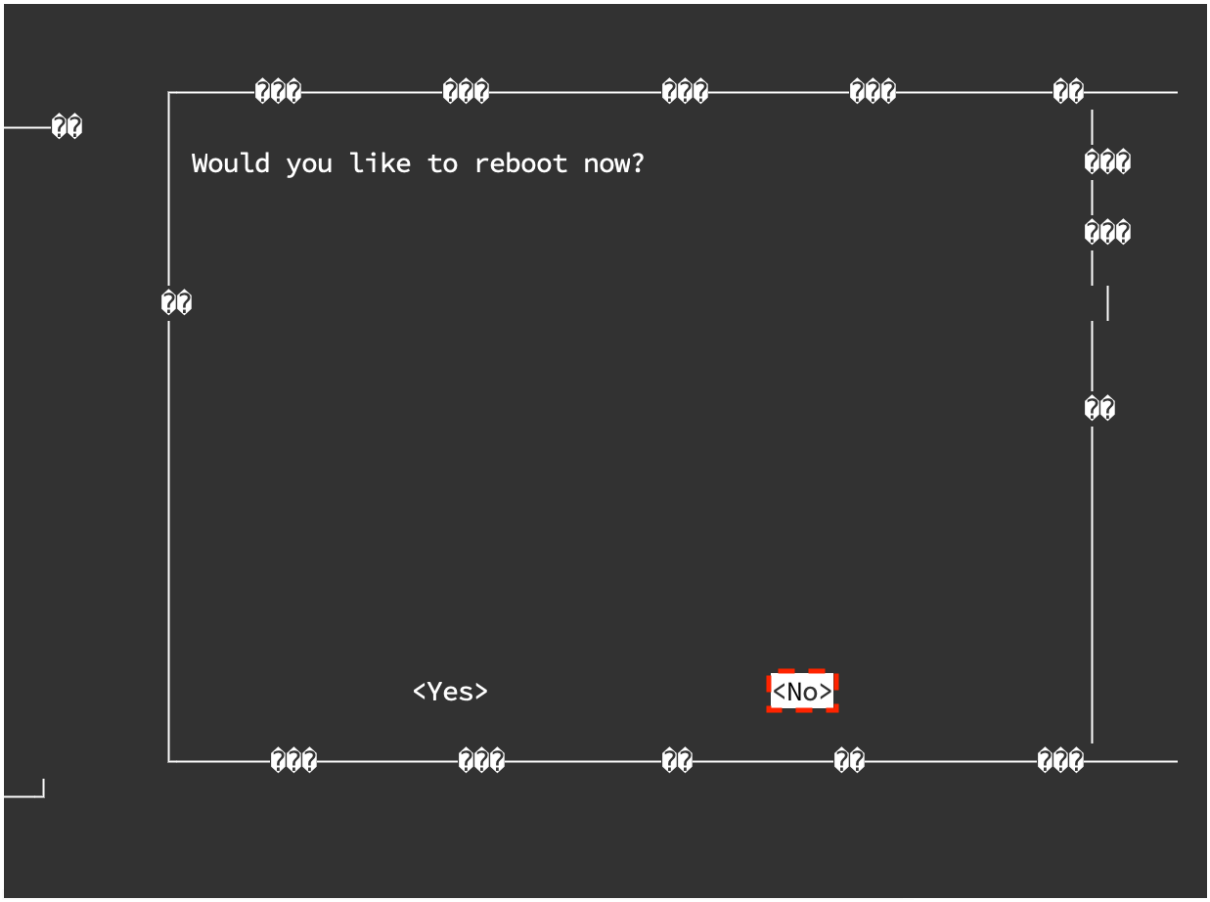
Figure 9-8: Choose finish



8. Do not reboot the machine.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 9-9: Decline to reboot

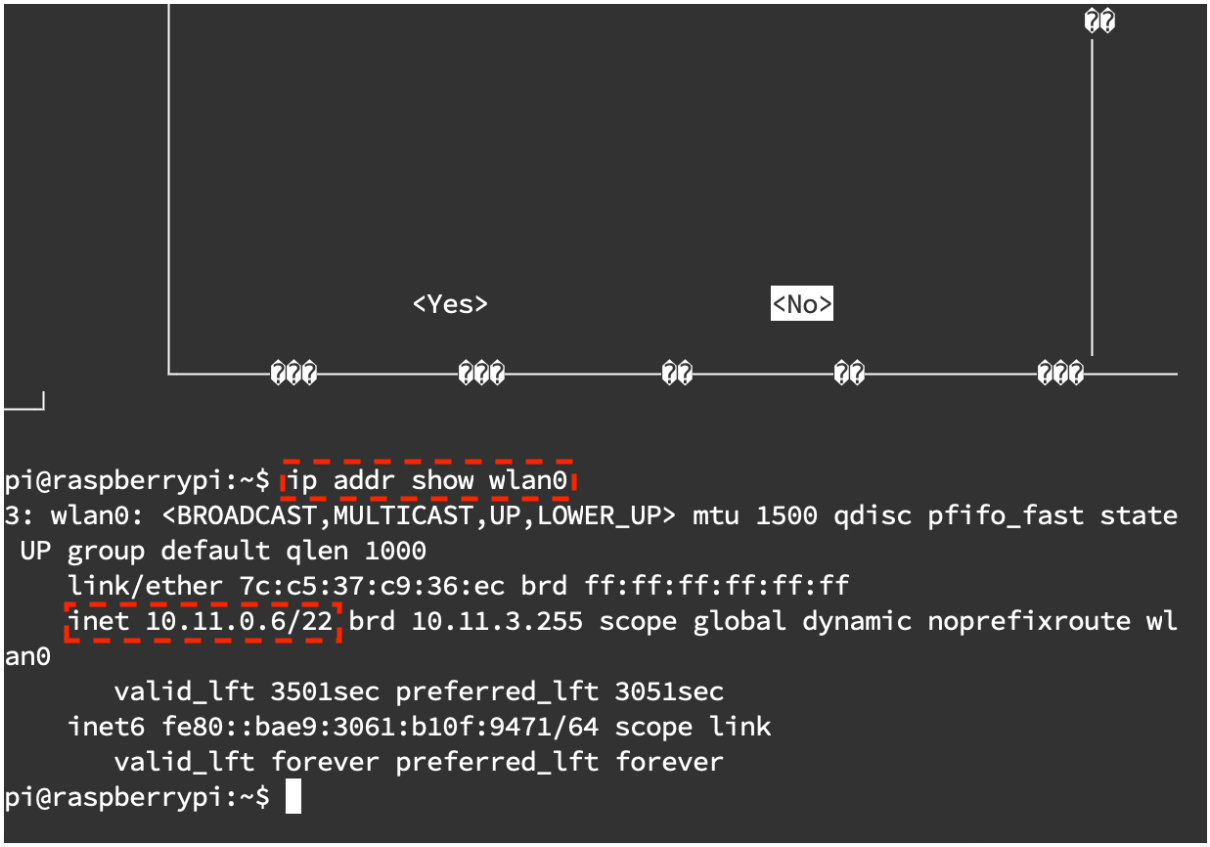


9. Verify that the wlan0 interface shows a valid IPv4 address that does not start with 169. Please note: the IPv6 address for this interface is a local-link auto address and not a real IPv6 address.

```
ip addr show wlan0
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 9-10: Show wlan0



10. Test the Wi-Fi connection to the internet by turning off the ethernet interface and trying to ping a public domain. Look for a result of `0% packet loss`.

```
sudo ifconfig eth0 down
ping -c5 aws.com
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 9-11: Zero packet loss

```
    inet 10.11.0.4/22 brd 10.11.3.255 scope global dynamic noprefixroute wl
an0
       valid_lft 3091sec preferred_lft 2641sec
    inet6 fe80::3ccb:6336:3c38:d42c/64 scope link
       valid_lft forever preferred_lft forever
pi@raspberrypi:~$ sudo ifconfig eth0 down
pi@raspberrypi:~$ ping -c5 aws.com
PING aws.com (18.67.65.31) 56(84) bytes of data.
64 bytes from server-18-67-65-31.iad89.r.cloudfront.net (18.67.65.31): icmp
_seq=1 ttl=238 time=0.949 ms
64 bytes from server-18-67-65-31.iad89.r.cloudfront.net (18.67.65.31): icmp
_seq=2 ttl=238 time=1.02 ms
64 bytes from server-18-67-65-31.iad89.r.cloudfront.net (18.67.65.31): icmp
_seq=3 ttl=238 time=0.979 ms
64 bytes from server-18-67-65-31.iad89.r.cloudfront.net (18.67.65.31): icmp
_seq=4 ttl=238 time=1.02 ms
64 bytes from server-18-67-65-31.iad89.r.cloudfront.net (18.67.65.31): icmp
_seq=5 ttl=238 time=1.01 ms

--- aws.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 0.949/0.994/1.021/0.026 ms
pi@raspberrypi:~$
```

11. After the test, you can bring the ethernet interface back up.

```
sudo ifconfig eth0 up
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

10. Communicate Using Bluetooth Low Energy

Learn how to use BLE on virtual Raspberry Pi 4 boards and virtual Android 12 devices.

- Use BLE Between Two Raspberry Pi Devices
  - Set Up the Raspberry Pi 4 BLE Node
  - Connect to the Node with the Raspberry Pi 4 BLE Master
- Use BLE Between a Raspberry Pi 4 Board and an Android Device
  - Set Up the Raspberry Pi 4 BLE Node
  - Connect to the Node with the Android BLE Master

AVH virtual devices support communicating through virtual Bluetooth Low Energy.

In this guide, we show how to communicate between two virtual Raspberry Pi 4 virtual boards as well as between a virtual Raspberry Pi 4 board and a virtual Android device.

10.1 Use BLE Between Two Raspberry Pi Devices

In this example, we will set up a Raspberry Pi 4 BLE Node and a Raspberry Pi 4 BLE Master.

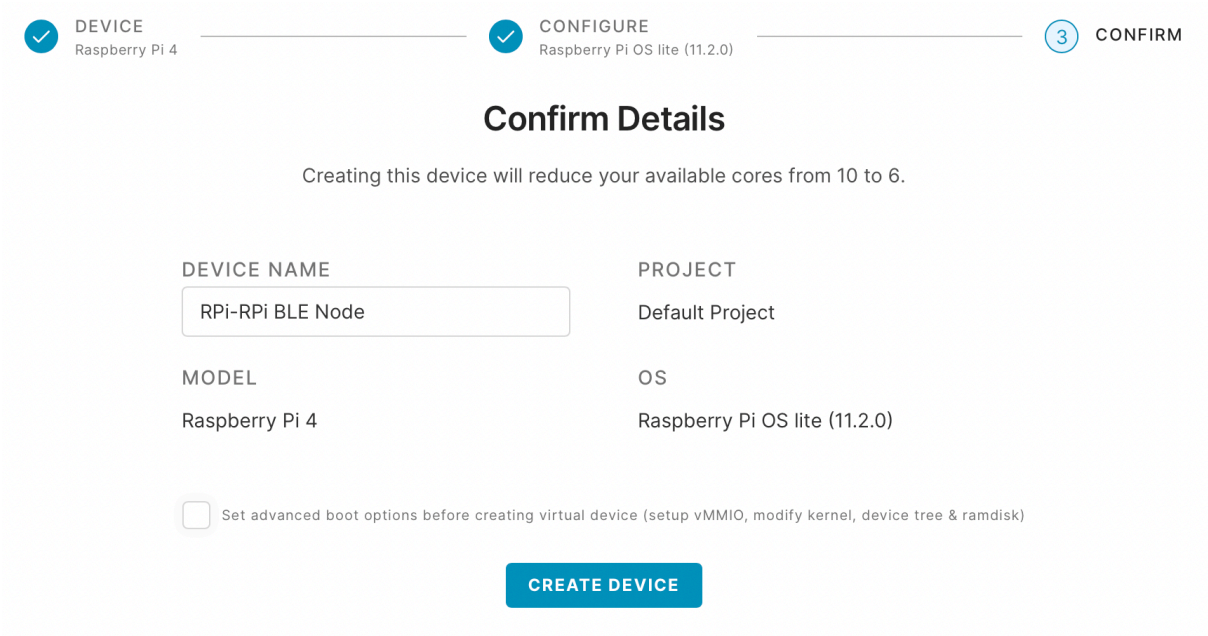
10.2 Set Up the Raspberry Pi 4 BLE Node

To set up the Raspberry Pi 4 BLE node, do the following:

1. Create a new Raspberry Pi 4 instance using the lite firmware. In our example, we will use the name “RPi-RPi BLE Node”.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-1: Create RPi-RPi node



2. Log in using the default credentials (`pi/raspberry`) and open the Bluetooth tool.

```
bluetoothctl1
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-2: Raspberry Pi Login

```
Password:
Linux raspberrypi 5.10.92-v8+ #1514 SMP PREEMPT Mon Jan 17 17:39:38 GMT 2022 aa
rch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to se
t a new password.


Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@raspberrypi:~$ bluetoothctl
Agent registered
[CHG] Controller 3E:A3:EC:CE:B1:61 Pairable: yes
[bluetooth]#
```

3. Turn on Bluetooth.

```
power on
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-3: Power on Bluetooth

```
rch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to se
t a new password.


Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@raspberrypi:~$ bluetoothctl
Agent registered
[CHG] Controller 3E:A3:EC:CE:B1:61 Pairable: yes
[bluetooth]# power on
Changing power on succeeded
[bluetooth]#
```

4. Open the advertise menu.

```
menu advertise
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-4: Menu advertise

```
[bluetooth]# menu advertise
Menu advertise:
Available commands:
-------------------
uuids [uuid1 uuid2 ...]                          Set/Get advertise uuids
service [uuid] [data=xx xx ...]                  Set/Get advertise service dat
a
manufacturer [id] [data=xx xx ...]               Set/Get advertise manufacture
r data
data [type] [data=xx xx ...]                     Set/Get advertise data
discoverable [on/off]                            Set/Get advertise discoverabl
e
discoverable-timeout [seconds]                   Set/Get advertise discoverabl
e timeout
tx-power [on/off]                                Show/Enable/Disable TX power
to be advertised
name [on/off/name]                               Configure local name to be ad
vertised
appearance [on/off/value]                        Configure custom appearance t
o be advertised
duration [seconds]                               Set/Get advertise duration
timeout [seconds]                                Set/Get advertise timeout
secondary [1M/2M/Coded]                          Set/Get advertise secondary c
```

5. Set the manufacturer to `0xffff 0x12 0x34` then read back the values.

```
manufacturer 0xffff 0x12 0x34
manufacturer
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-5: Manufacturer node

```
tx-power [on/off]                                    Show/Enable/Disable TX power
to be advertised
name [on/off/name]                                   Configure local name to be ad
vertised
appearance [on/off/value]                            Configure custom appearance t
o be advertised
duration [seconds]                                   Set/Get advertise duration
timeout [seconds]                                    Set/Get advertise timeout
secondary [1M/2M/Coded]                              Set/Get advertise secondary c
hannel
clear [uuids/service/manufacturer/config-name...]    Clear advertise config
back                                                 Return to main menu
version                                              Display version
quit                                                 Quit program
exit                                                 Quit program
help                                                 Display help about this progr
am
export                                               Print environment variables
[bluetooth]# manufacturer 0xffff 0x12 0x34
[bluetooth]# manufacturer
Manufacturer: 65535
  12 34                                               .4
[bluetooth]#
```

6. Set the name to `avh-demo-1` then read back the value.

```
name avh-demo-1
name
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-6: Set name node

```
vertised
appearance [on/off/value]                          Configure custom appearance t
o be advertised
duration [seconds]                                 Set/Get advertise duration
timeout [seconds]                                  Set/Get advertise timeout
secondary [1M/2M/Coded]                            Set/Get advertise secondary c
hannel
clear [uuids/service/manufacturer/config-name...] Clear advertise config
back                                               Return to main menu
version                                            Display version
quit                                               Quit program
exit                                               Quit program
help                                               Display help about this progr
am
export                                             Print environment variables
[bluetooth]# manufacturer 0xffff 0x12 0x34
[bluetooth]# manufacturer
Manufacturer: 65535
  12 34                                             .4
[bluetooth]# name avh-demo-1
[bluetooth]# name
LocalName: avh-demo-1
[bluetooth]#
```

7. Go back to the main menu.

```
back
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-7: Main menu

```
[bluetooth]# back
Menu main:
Available commands:
-------------------
advertise                                          Advertise Options Submenu
scan                                               Scan Options Submenu
gatt                                               Generic Attribute Submenu
list                                               List available controllers
show [ctrl]                                        Controller information
select <ctrl>                                      Select default controller
devices                                            List available devices
paired-devices                                     List paired devices
system-alias <name>                                Set controller alias
reset-alias                                        Reset controller alias
power <on/off>                                     Set controller power
pairable <on/off>                                  Set controller pairable mode
discoverable <on/off>                              Set controller discoverable m
ode
discoverable-timeout [value]                       Set discoverable timeout
agent <on/off/capability>                          Enable/disable agent with giv
en capability
default-agent                                      Set agent as the default one
advertise <on/off/type>                            Enable/disable advertising wi
```

8. Open the generic attributes menu.

```
menu gatt
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-8: Generic attributes menu

```
[bluetooth]# menu gatt
Menu gatt:
Available commands:
-------------------
list-attributes [dev/local]                    List attributes
select-attribute <attribute/UUID>              Select attribute
attribute-info [attribute/UUID]                Select attribute
read [offset]                                  Read attribute value
write <data=xx xx ...> [offset] [type]         Write attribute value
acquire-write                                  Acquire Write file descriptor
release-write                                  Release Write file descriptor
acquire-notify                                 Acquire Notify file descripto
r
release-notify                                 Release Notify file descripto
r
notify <on/off>                                Notify attribute value
clone [dev/attribute/UUID]                     Clone a device or attribute
register-application [UUID ...]                Register profile to connect
unregister-application                         Unregister profile
register-service <UUID> [handle]               Register application service.
unregister-service <UUID/object>               Unregister application servic
e
register-includes <UUID> [handle]              Register as Included service
```

9. Register a new service. At the prompt, enter `yes`.

```
register-service e2d36f99-8909-4136-9a49-d825508b297b
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-9: Register a new service

```
unregister-includes <Service-UUID><Inc-UUID>      Unregister Included service.
register-characteristic <UUID> <Flags=read,write,notify...> [handle] Register a
pplication characteristic
unregister-characteristic <UUID/object>           Unregister application charac
teristic
register-descriptor <UUID> <Flags=read,write...> [handle] Register application
descriptor
unregister-descriptor <UUID/object>               Unregister application descri
ptor
back                                              Return to main menu
version                                           Display version
quit                                              Quit program
exit                                              Quit program
help                                              Display help about this progr
am
export                                            Print environment variables
[bluetooth]# register-service e2d36f99-8909-4136-9a49-d825508b297b
[NEW] Primary Service (Handle 0x0000)
        /org/bluez/app/service0
        e2d36f99-8909-4136-9a49-d825508b297b
        Vendor specific
[/org/bluez/app/service0] Primary (yes/no): yes
[bluetooth]#
```

10. Register a new read-only characteristic. In our example, we will enter `07` at the prompt.

```
register-characteristic 0x1234 read
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-10: Register a read-only characteristic

```
descriptor
unregister-descriptor <UUID/object>                Unregister application descri
ptor
back                                               Return to main menu
version                                            Display version
quit                                               Quit program
exit                                               Quit program
help                                               Display help about this progr
am
export                                             Print environment variables
[bluetooth]# register-service e2d36f99-8909-4136-9a49-d825508b297b
[NEW] Primary Service (Handle 0x0000)
        /org/bluez/app/service0
        e2d36f99-8909-4136-9a49-d825508b297b
        Vendor specific
[/org/bluez/app/service0] Primary (yes/no): yes
[bluetooth]# register-characteristic 0x1234 read
[NEW] Characteristic (Handle 0x0000)
        /org/bluez/app/service0/chrc0
        0x1234
        Unknown
[/org/bluez/app/service0/chrc0] Enter value: 07
[bluetooth]#
```

11. Register a new read-write characteristic. In our example, we will enter `13` at the prompt.

```
register-characteristic 0x5678 read,write
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-11: Register a read-write characteristic

```
exit                                                Quit program
help                                                Display help about this progr
am
export                                              Print environment variables
[bluetooth]# register-service e2d36f99-8909-4136-9a49-d825508b297b
[NEW] Primary Service (Handle 0x0000)
        /org/bluez/app/service0
        e2d36f99-8909-4136-9a49-d825508b297b
        Vendor specific
[/org/bluez/app/service0] Primary (yes/no): yes
[bluetooth]# register-characteristic 0x1234 read
[NEW] Characteristic (Handle 0x0000)
        /org/bluez/app/service0/chrc0
        0x1234
        Unknown
[/org/bluez/app/service0/chrc0] Enter value: 07
[bluetooth]# register-characteristic 0x5678 read,write
[NEW] Characteristic (Handle 0x0000)
        /org/bluez/app/service0/chrc1
        0x5678
        Unknown
[/org/bluez/app/service0/chrc1] Enter value: 13
[bluetooth]#
```

12. Register the application. You should see a response `Application Registered`.

```
register-application
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-12: Register the application

```
[bluetooth]# register-application
[CHG] Primary Service (Handle 0x0013)
        /org/bluez/app/service0
        e2d36f99-8909-4136-9a49-d825508b297b
        Vendor specific
[CHG] Characteristic (Handle 0x0015)
        /org/bluez/app/service0/chrc1
        0x5678
        Unknown
[CHG] Characteristic (Handle 0x0017)
        /org/bluez/app/service0/chrc0
        0x1234
        Unknown
[CHG] Controller 3E:A3:EC:CE:B1:61 UUIDs: 00001801-0000-1000-8000-00805f9b34fb
[CHG] Controller 3E:A3:EC:CE:B1:61 UUIDs: 00001800-0000-1000-8000-00805f9b34fb
[CHG] Controller 3E:A3:EC:CE:B1:61 UUIDs: e2d36f99-8909-4136-9a49-d825508b297b
[CHG] Controller 3E:A3:EC:CE:B1:61 UUIDs: 00001200-0000-1000-8000-00805f9b34fb
[CHG] Controller 3E:A3:EC:CE:B1:61 UUIDs: 0000110c-0000-1000-8000-00805f9b34fb
[CHG] Controller 3E:A3:EC:CE:B1:61 UUIDs: 0000110e-0000-1000-8000-00805f9b34fb
[CHG] Controller 3E:A3:EC:CE:B1:61 UUIDs: 0000180a-0000-1000-8000-00805f9b34fb
Application registered
[CHG] Controller 3E:A3:EC:CE:B1:61 UUIDs: 00001801-0000-1000-8000-00805f9b34fb
[CHG] Controller 3E:A3:EC:CE:B1:61 UUIDs: 00001800-0000-1000-8000-00805f9b34fb
```

13. Go back to the main menu again.

```
back
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-13: Main menu

```
[bluetooth]# back
Menu main:
Available commands:
-------------------
advertise                                          Advertise Options Submenu
scan                                               Scan Options Submenu
gatt                                               Generic Attribute Submenu
list                                               List available controllers
show [ctrl]                                        Controller information
select <ctrl>                                      Select default controller
devices                                            List available devices
paired-devices                                     List paired devices
system-alias <name>                                Set controller alias
reset-alias                                        Reset controller alias
power <on/off>                                     Set controller power
pairable <on/off>                                  Set controller pairable mode
discoverable <on/off>                              Set controller discoverable m
ode
discoverable-timeout [value]                       Set discoverable timeout
agent <on/off/capability>                          Enable/disable agent with giv
en capability
default-agent                                      Set agent as the default one
advertise <on/off/type>                            Enable/disable advertising wi
```

14. Turn on advertising.

```
advertise on
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-14: Advertise node

```
block [dev]                                          Block device
unblock [dev]                                        Unblock device
remove <dev>                                         Remove device
connect <dev>                                        Connect device
disconnect [dev]                                     Disconnect device
menu <name>                                          Select submenu
version                                              Display version
quit                                                 Quit program
exit                                                 Quit program
help                                                 Display help about this progr
am
export                                               Print environment variables
[bluetooth]# advertise on
[CHG] Controller 3E:A3:EC:CE:B1:61 SupportedInstances: 0x04 (4)
[CHG] Controller 3E:A3:EC:CE:B1:61 ActiveInstances: 0x01 (1)
Advertising object registered
Manufacturer: 65535
  12 34                                               .4
Tx Power: off
LocalName: avh-demo-1
Appearance: off
Discoverable: on
[bluetooth]#
```

15. Your Bluetooth node is now set up. Proceed to the next step to connect to this node using a second Raspberry Pi 4 board.

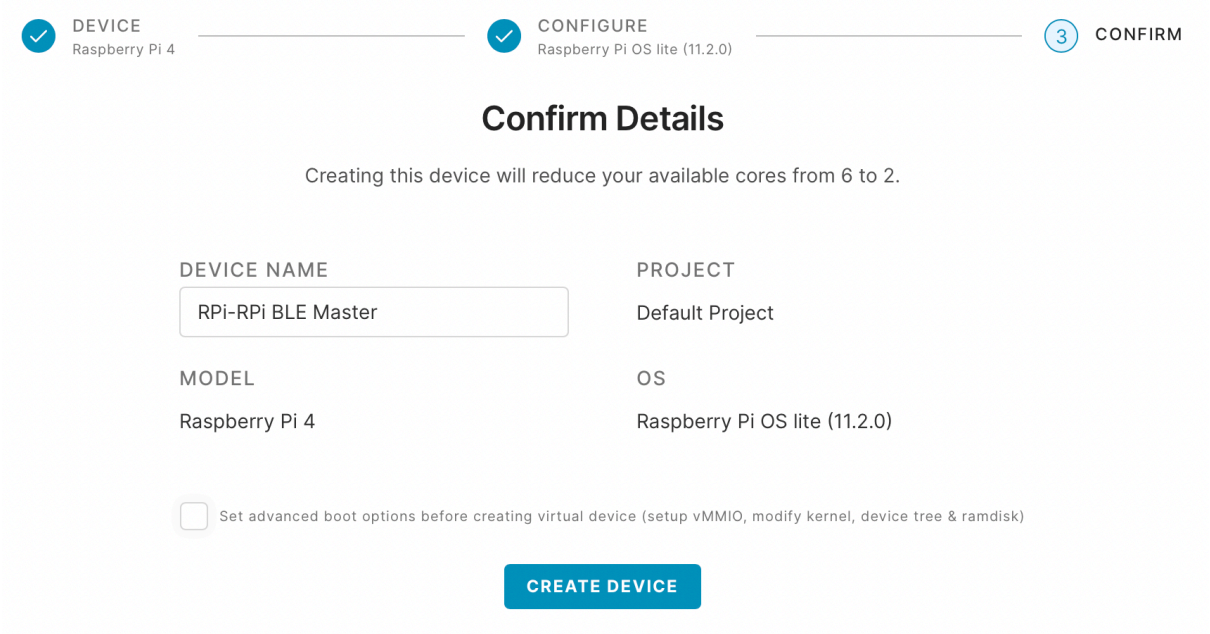
10.3 Connect to the Node with the Raspberry Pi 4 BLE Master

To connect to the node with the Raspberry Pi 4 BLE master, do the following:

1. Create a new Raspberry Pi 4 instance using the lite firmware. In our example, we will use the name “RPi-RPi BLE Master”.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-15: Create Raspberry Pi 4 instance



2. Log in using the default credentials (`pi/raspberry`) and open the Bluetooth tool.

```
bluetoothctl1
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-16: Raspberry Pi Login

```
Password:
Linux raspberrypi 5.10.92-v8+ #1514 SMP PREEMPT Mon Jan 17 17:39:38 GMT 2022 aa
rch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to se
t a new password.


Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@raspberrypi:~$ bluetoothctl
Agent registered
[CHG] Controller 3E:A3:EC:CE:B1:61 Pairable: yes
[bluetooth]#
```

3. Turn on Bluetooth.

```
power on
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-17: Power on Bluetooth

```
rch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to se
t a new password.


Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@raspberrypi:~$ bluetoothctl
Agent registered
[CHG] Controller 3E:A3:EC:CE:B1:61 Pairable: yes
[bluetooth]# power on
Changing power on succeeded
[bluetooth]#
```

4. Scan for Bluetooth devices.

```
scan on
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-18: Scan node

```
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to se
t a new password.


Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.

pi@raspberrypi:~$ bluetoothctl
Agent registered
[CHG] Controller 3E:A3:EC:DF:D6:A4 Pairable: yes
[bluetooth]# power on
Changing power on succeeded
[bluetooth]# scan on
Discovery started
[CHG] Controller 3E:A3:EC:DF:D6:A4 Discovering: yes
[NEW] Device 3E:A3:EC:CE:B1:61 avh-demo-1
[bluetooth]#
```

5. Connect to the `avh-demo-1` device. Make note of the two IDs starting with `00001234` and `00005678`, referring to the custom characteristics that you set up.

```
connect 3E:A3:EC:CE:B1:61
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-19: Connect node

```
[bluetooth]# connect 3E:A3:EC:CE:B1:61
Attempting to connect to 3E:A3:EC:CE:B1:61
[CHG] Device 3E:A3:EC:CE:B1:61 Connected: yes
Connection successful
[NEW] Primary Service (Handle 0xb964)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0006
        00001801-0000-1000-8000-00805f9b34fb
        Generic Attribute Profile
[NEW] Characteristic (Handle 0x0d48)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0006/char0007
        00002a05-0000-1000-8000-00805f9b34fb
        Service Changed
[NEW] Descriptor (Handle 0x0d90)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0006/char0007/desc0009
        00002902-0000-1000-8000-00805f9b34fb
        Client Characteristic Configuration
[NEW] Characteristic (Handle 0x0d48)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0006/char000a
        00002b29-0000-1000-8000-00805f9b34fb
        Client Supported Features
[NEW] Characteristic (Handle 0x0d48)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0006/char000c
        00002b2a-0000-1000-8000-00805f9b34fb
        Database Hash
[NEW] Characteristic (Handle 0x0d48)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0006/char000e
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-20: Connect node

```
        Device Information
[NEW] Characteristic (Handle 0x0d48)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0010/char0011
        00002a50-0000-1000-8000-00805f9b34fb
        PnP ID
[NEW] Primary Service (Handle 0xcd04)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013
        e2d36f99-8909-4136-9a49-d825508b297b
        Vendor specific
[NEW] Characteristic (Handle 0x0d48)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0014
        00005678-0000-1000-8000-00805f9b34fb
        Unknown
[NEW] Characteristic (Handle 0x0d48)
        /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
        00001234-0000-1000-8000-00805f9b34fb
        Unknown
[CHG] Device 3E:A3:EC:CE:B1:61 UUIDs: 00001800-0000-1000-8000-00805f9b34fb
[CHG] Device 3E:A3:EC:CE:B1:61 UUIDs: 00001801-0000-1000-8000-00805f9b34fb
[CHG] Device 3E:A3:EC:CE:B1:61 UUIDs: 0000180a-0000-1000-8000-00805f9b34fb
[CHG] Device 3E:A3:EC:CE:B1:61 UUIDs: e2d36f99-8909-4136-9a49-d825508b297b
[CHG] Device 3E:A3:EC:CE:B1:61 ServicesResolved: yes
[CHG] Device 3E:A3:EC:CE:B1:61 Name: raspberrypi
[CHG] Device 3E:A3:EC:CE:B1:61 Alias: raspberrypi
[CHG] Device 3E:A3:EC:CE:B1:61 Modalias: usb:v1D6Bp0246d0537
[avh-demo-1]#
```

6. Open the generic attributes menu.

```
menu gatt
```

7. Select the read-only attribute with an ID starting with `00001234`. The prompt should change to the appropriate characteristic.

```
select-attribute 00001234-0000-1000-8000-00805f9b34fb
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-21: Select attribute

```
clone [dev/attribute/UUID]                          Clone a device or attribute
register-application [UUID ...]                     Register profile to connect
unregister-application                              Unregister profile
register-service <UUID> [handle]                    Register application service.
unregister-service <UUID/object>                    Unregister application servic
e
register-includes <UUID> [handle]                   Register as Included service
in.
unregister-includes <Service-UUID><Inc-UUID>        Unregister Included service.
register-characteristic <UUID> <Flags=read,write,notify...> [handle] Register a
pplication characteristic
unregister-characteristic <UUID/object>             Unregister application charac
teristic
register-descriptor <UUID> <Flags=read,write...> [handle] Register application
descriptor
unregister-descriptor <UUID/object>                 Unregister application descri
ptor
back                                                Return to main menu
version                                             Display version
quit                                                Quit program
exit                                                Quit program
help                                                Display help about this progr
am
export                                              Print environment variables
[avh-demo-1]# select-attribute 00001234-0000-1000-8000-00805f9b34fb
[raspberrypi:/service0013/char0016]#
```

8. Read the value. You should see the value you set previously, represented in hexidecimal.

```
read
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-22: Read node

```
register-includes <UUID> [handle]                  Register as Included service
in.
unregister-includes <Service-UUID><Inc-UUID>       Unregister Included service.
register-characteristic <UUID> <Flags=read,write,notify...> [handle] Register a
pplication characteristic
unregister-characteristic <UUID/object>            Unregister application charac
teristic
register-descriptor <UUID> <Flags=read,write...> [handle] Register application
descriptor
unregister-descriptor <UUID/object>                Unregister application descri
ptor
back                                               Return to main menu
version                                            Display version
quit                                               Quit program
exit                                               Quit program
help                                               Display help about this progr
am
export                                             Print environment variables
[avh-demo-1]# select-attribute 00001234-0000-1000-8000-00805f9b34fb
[raspberrypi:/service0013/char0016]# read
Attempting to read /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
[CHG] Attribute /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016 Valu
e:
  07                                                .
  07                                                .
[raspberrypi:/service0013/char0016]#
```

9. Attempt to write a new value. You should see an error because the characteristic is read-only.

```
write 08
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-23: Write error

```
pplication characteristic
unregister-characteristic <UUID/object>          Unregister application charac
teristic
register-descriptor <UUID> <Flags=read,write...> [handle] Register application
descriptor
unregister-descriptor <UUID/object>              Unregister application descri
ptor
back                                             Return to main menu
version                                          Display version
quit                                             Quit program
exit                                             Quit program
help                                             Display help about this progr
am
export                                           Print environment variables
[avh-demo-1]# select-attribute 00001234-0000-1000-8000-00805f9b34fb
[raspberrypi:/service0013/char0016]# read
Attempting to read /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
[CHG] Attribute /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016 Valu
e:
  07                                              .
  07                                              .
[raspberrypi:/service0013/char0016]# write 08
Invalid value at index 0
Attempting to write /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
Failed to write: org.bluez.Error.NotSupported
[raspberrypi:/service0013/char0016]#
```

10. Select the read-write characteristic with an ID starting with `00005678`. The prompt should change to the appropriate characteristic.

```
select-attribute 00005678-0000-1000-8000-00805f9b34fb
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-24: Read-write characteristic

```
teristic
register-descriptor <UUID> <Flags=read,write...> [handle] Register application
descriptor
unregister-descriptor <UUID/object>              Unregister application descri
ptor
back                                             Return to main menu
version                                          Display version
quit                                             Quit program
exit                                             Quit program
help                                             Display help about this progr
am
export                                           Print environment variables
[avh-demo-1]# select-attribute 00001234-0000-1000-8000-00805f9b34fb
[raspberrypi:/service0013/char0016]# read
Attempting to read /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
[CHG] Attribute /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016 Valu
e:
  07                                              .
  07                                              .
[raspberrypi:/service0013/char0016]# write 08
Invalid value at index 0
Attempting to write /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
Failed to write: org.bluez.Error.NotSupported
[raspberrypi:/service0013/char0016]# select-attribute 00005678-0000-1000-8000-0
0805f9b34fb
[raspberrypi:/service0013/char0014]#
```

11. Read the value. You should see the value you set previously, represented in hexidecimal.

```
read
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-25: Read node

```
version                                                 Display version
quit                                                    Quit program
exit                                                    Quit program
help                                                    Display help about this progr
am
export                                                  Print environment variables
[avh-demo-1]# select-attribute 00001234-0000-1000-8000-00805f9b34fb
[raspberrypi:/service0013/char0016]# read
Attempting to read /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
[CHG] Attribute /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016 Valu
e:
  07                                                     .
  07                                                     .
[raspberrypi:/service0013/char0016]# write 08
Invalid value at index 0
Attempting to write /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
Failed to write: org.bluez.Error.NotSupported
[raspberrypi:/service0013/char0016]# select-attribute 00005678-0000-1000-8000-0
0805f9b34fb
[raspberrypi:/service0013/char0014]# read
Attempting to read /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0014
[CHG] Attribute /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0014 Valu
e:
  0d                                                     .
  0d                                                     .
[raspberrypi:/service0013/char0014]#
```

12. Write a new value then read it back. In our example, we will use `14`, which is `0d` in hexidecimal.

```
write 14
read
```

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-26: New value node

```
Attempting to read /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
[CHG] Attribute /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016 Valu
e:
  07                                                .
  07                                                .
[raspberrypi:/service0013/char0016]# write 08
Invalid value at index 0
Attempting to write /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0016
Failed to write: org.bluez.Error.NotSupported
[raspberrypi:/service0013/char0016]# select-attribute 00005678-0000-1000-8000-0
0805f9b34fb
[raspberrypi:/service0013/char0014]# read
Attempting to read /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0014
[CHG] Attribute /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0014 Valu
e:
  0d                                                .
  0d                                                .
[raspberrypi:/service0013/char0014]# write 14
Attempting to write /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0014
[raspberrypi:/service0013/char0014]# read
Attempting to read /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0014
[CHG] Attribute /org/bluez/hci0/dev_3E_A3_EC_CE_B1_61/service0013/char0014 Valu
e:
  0e                                                .
  0e                                                .
[raspberrypi:/service0013/char0014]#
```

10.4 Use BLE Between a Raspberry Pi 4 Board and an Android Device

In this example, we will set up a Raspberry Pi 4 BLE Node and an Android BLE Master.

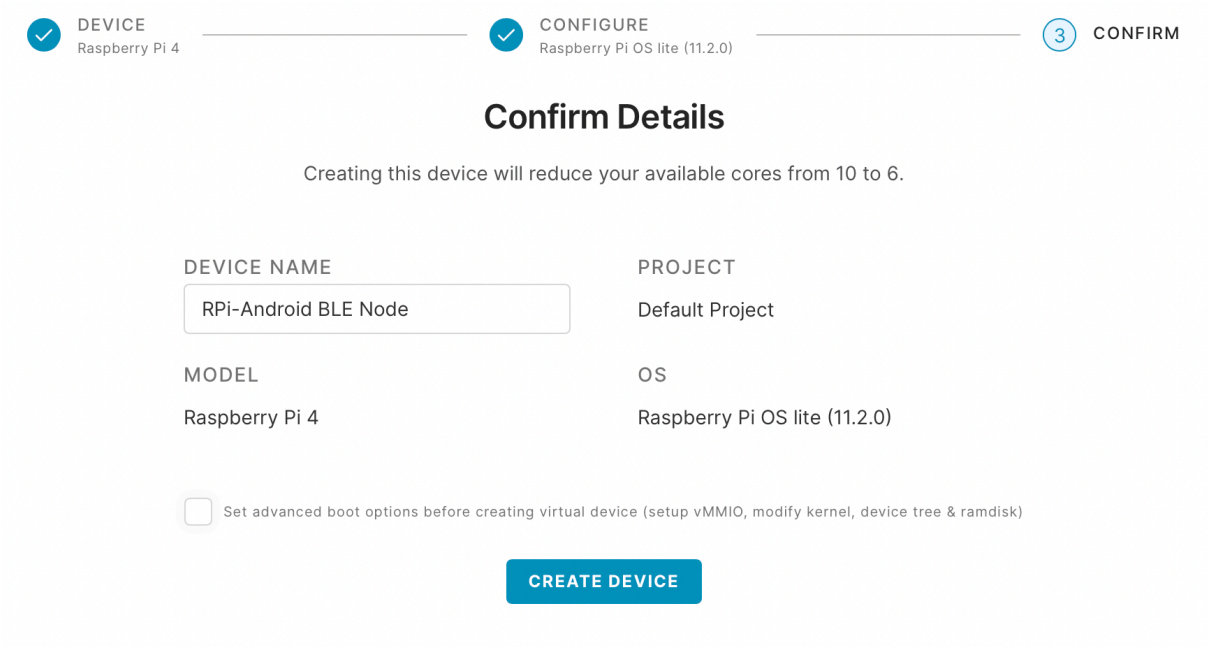
10.5 Set Up the Raspberry Pi 4 BLE Node

To set up the Raspberry Pi 4 BLE node, do the following:

1. Create a new Raspberry Pi 4 board. In our example, we will use the name “RPi-Android BLE Node”.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-27: Create Raspberry Pi 4 instance



2. Follow the steps in the previous section to set up a Raspberry Pi BLE Node. For our example, we will set the name of the Bluetooth node to "avh-demo-2'.

```
bluetoothctl
power on
menu advertise
manufacturer 0xffff 0x12 0x34
name avh-demo-2
back
menu gatt
register-service e2d36f99-8909-4136-9a49-d825508b297b
register-characteristic 0x1234 read
register-characteristic 0x5678 read,write
register-application
back
advertise on
```

3. When the node is advertising, make note of the node's MAC address (`3E:A3:EC:7B:47:66` in our example).

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-28: Node MAC address

```
unblock [dev]                                    Unblock device
remove <dev>                                     Remove device
connect <dev>                                    Connect device
disconnect [dev]                                 Disconnect device
menu <name>                                      Select submenu
version                                          Display version
quit                                             Quit program
exit                                             Quit program
help                                             Display help about this
program
export                                           Print environment variab
les
[bluetooth]# advertise on
[CHG] Controller 3E:A3:EC:7B:47:66 SupportedInstances: 0x04 (4)
[CHG] Controller 3E:A3:EC:7B:47:66 ActiveInstances: 0x01 (1)
Advertising object registered
Manufacturer: 65535
  12 34                                           .4
Tx Power: off
LocalName: avh-demo-2
Appearance: off
Discoverable: on
[bluetooth]#
```

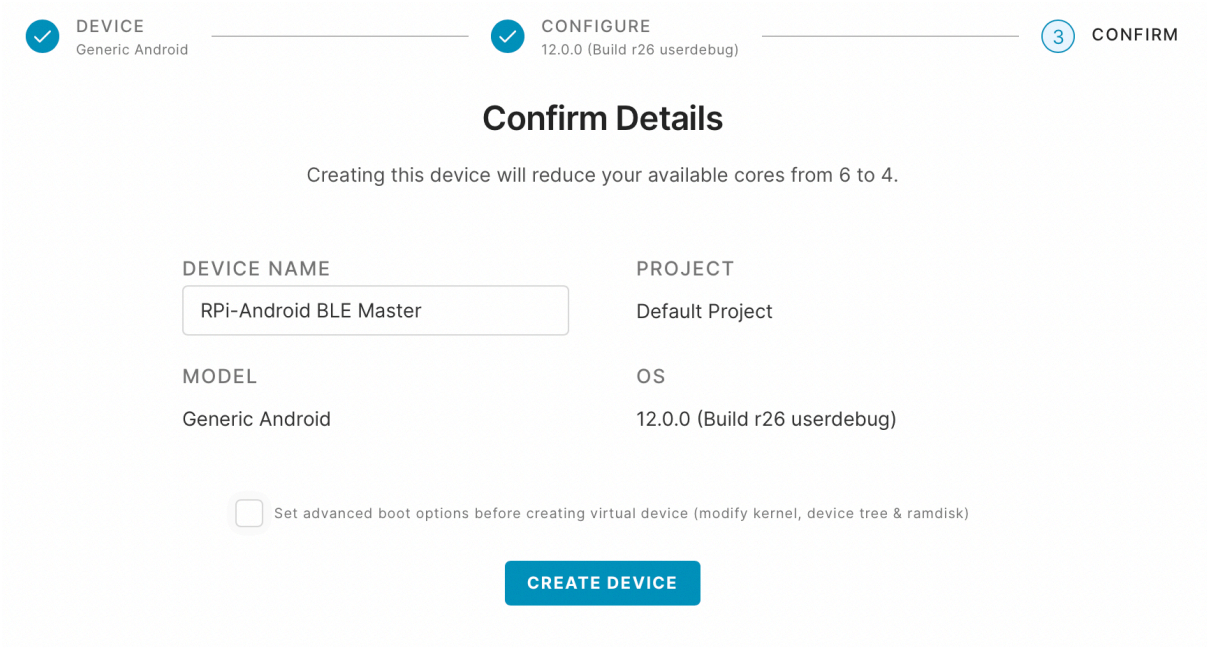
10.6 Connect to the Node with the Android BLE Master

To connect to the node with the Android BLE master, do the following:

1. Create a new Android device running firmware 12.0.0. For our example, we will use the name “RPi-Android BLE Master”.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

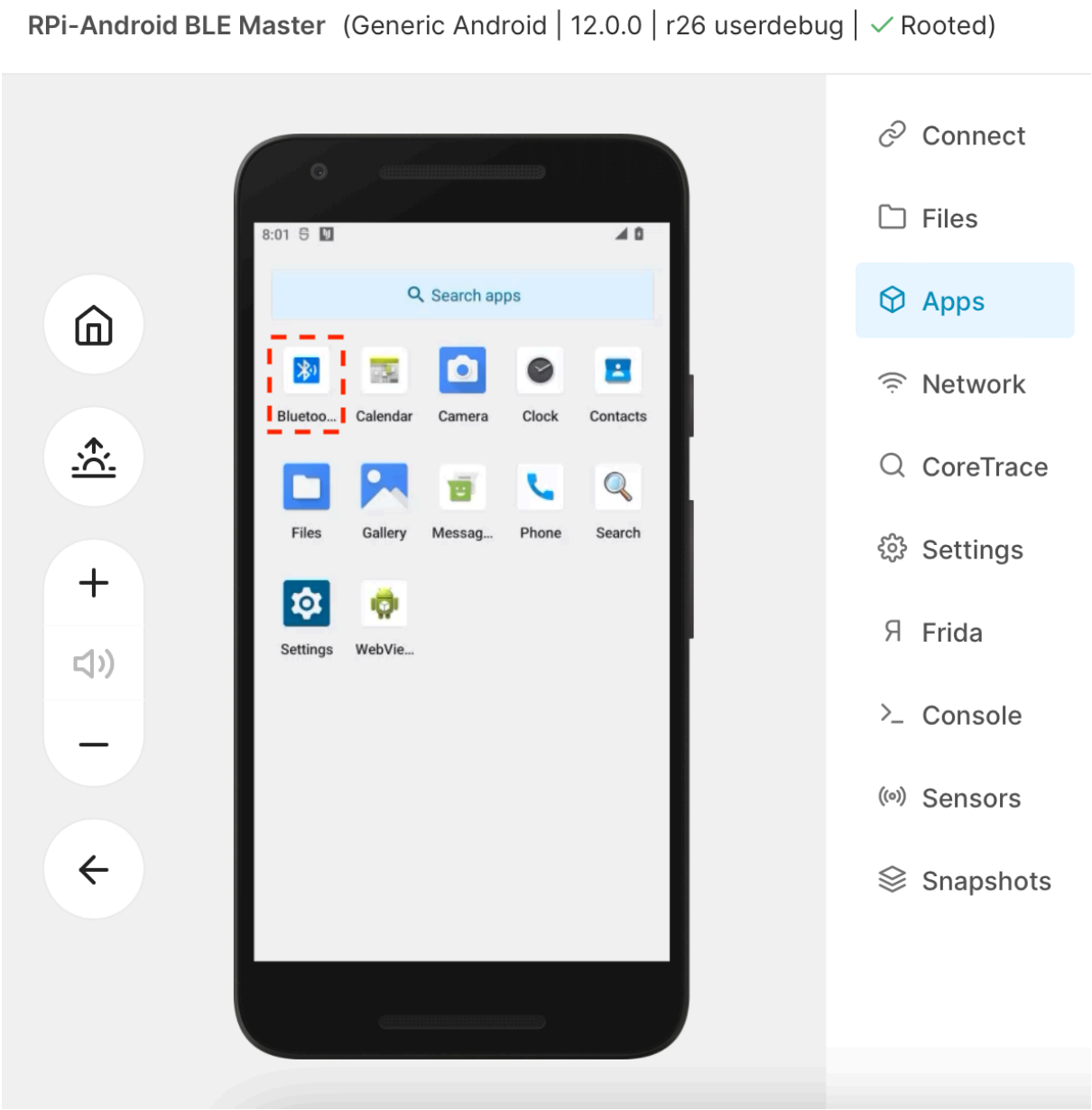
Figure 10-29: Create Raspberry Pi 4 instance



2. Download the Bluetooth LE Scanner APK to your local computer then install the APK through the Apps tab.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

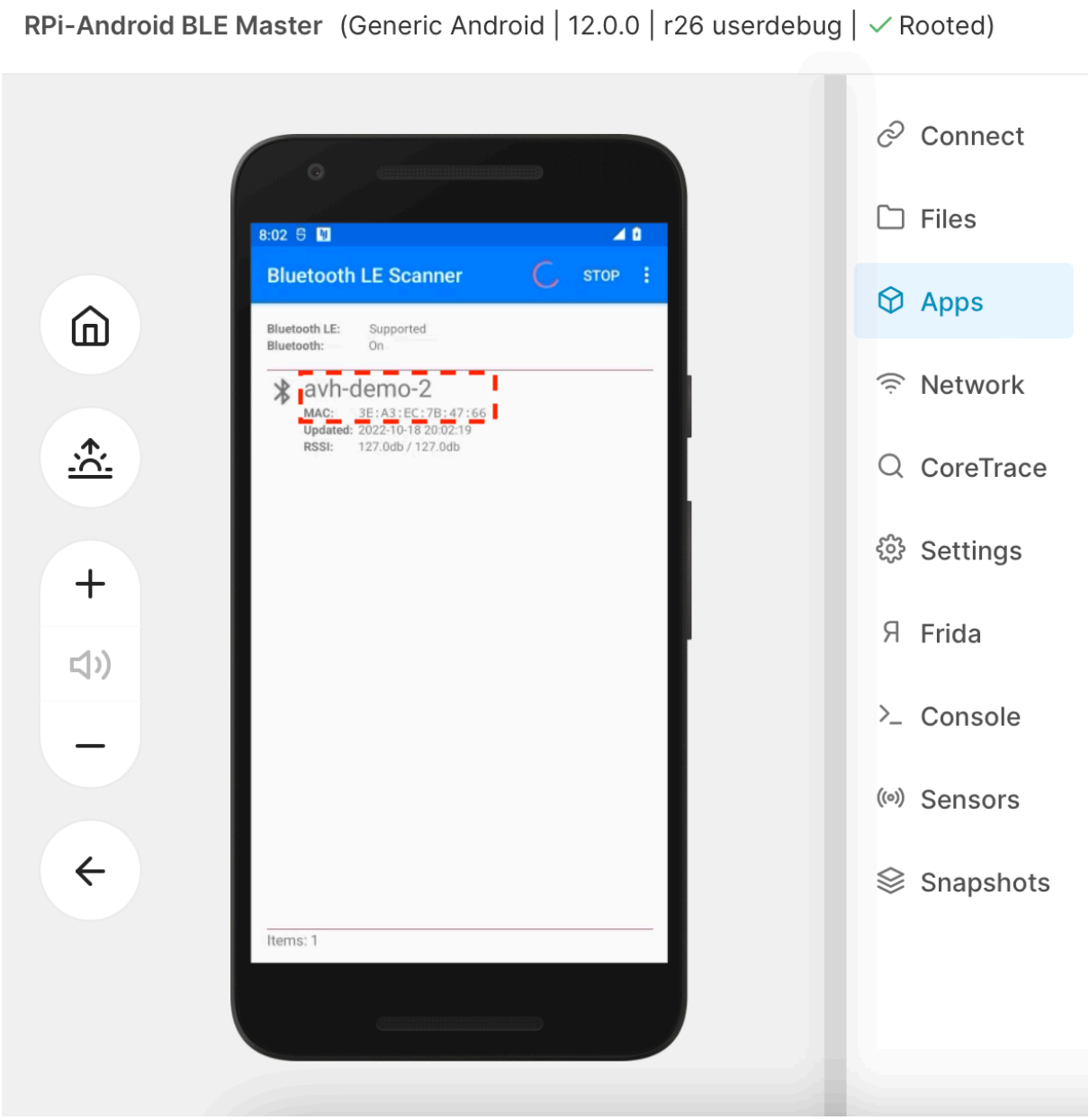
Figure 10-30: Install Bluetooth LE scanner



3. Open the Bluetooth LE Scanner app and click Scan. You should see `avh-demo-2` appear with the matching MAC address.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

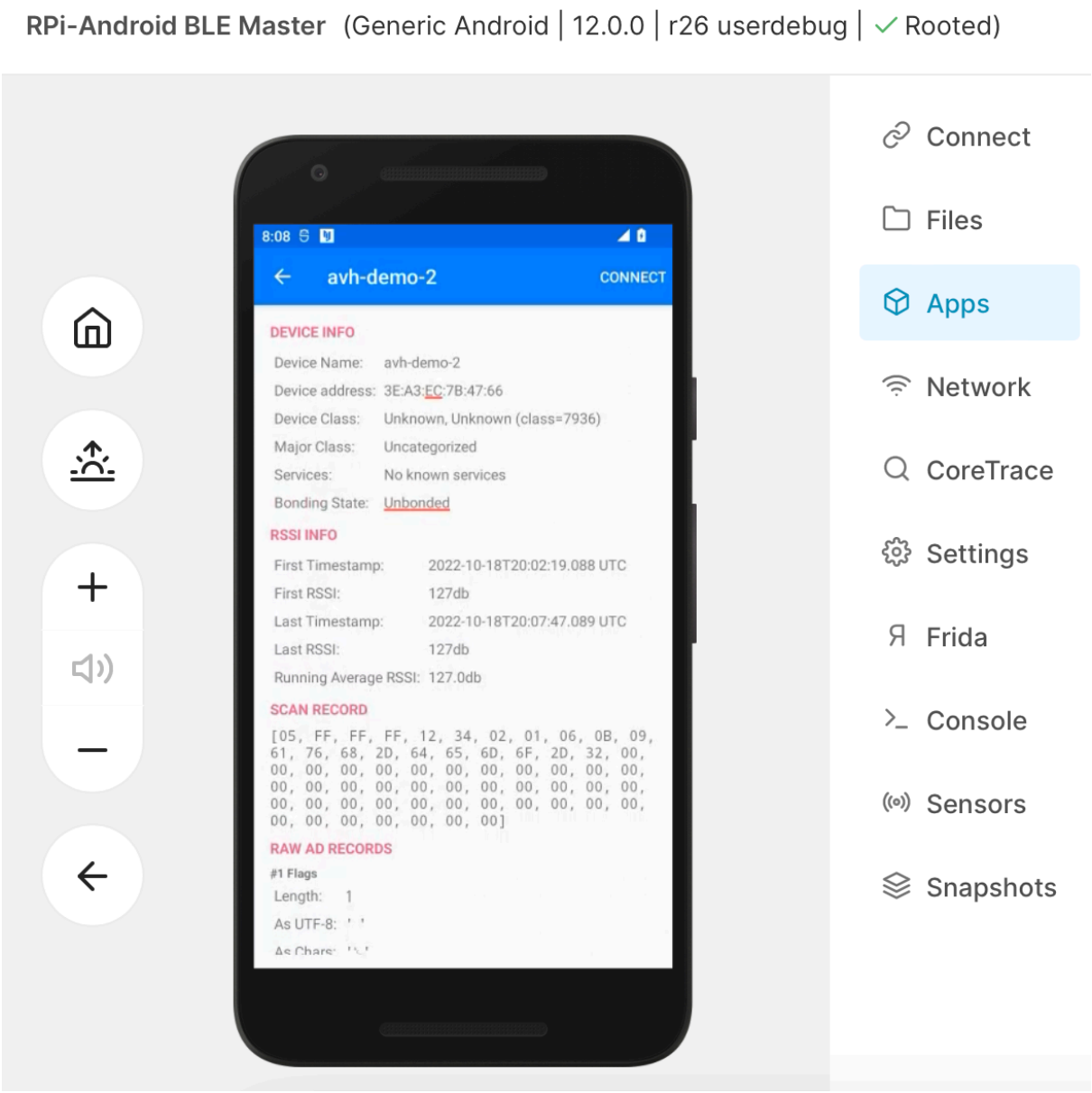
Figure 10-31: Scan for node



4. Click on avh-demo-2 to open up additional details. Scroll to the bottom of the page to see the manufacturer [ff, ff, 12, 34] we set previously.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-32: Additional details view



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-33: Additional details view

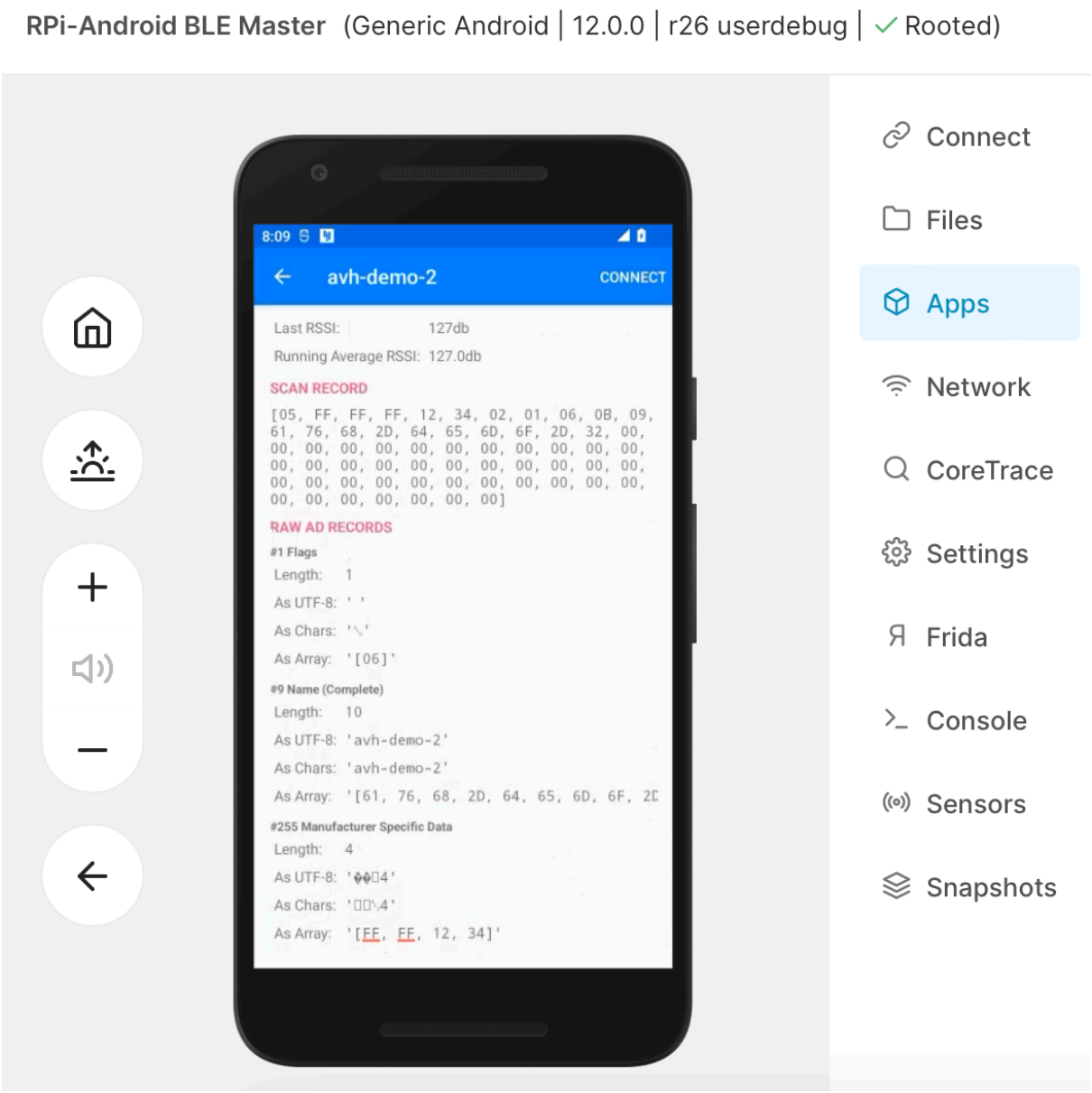
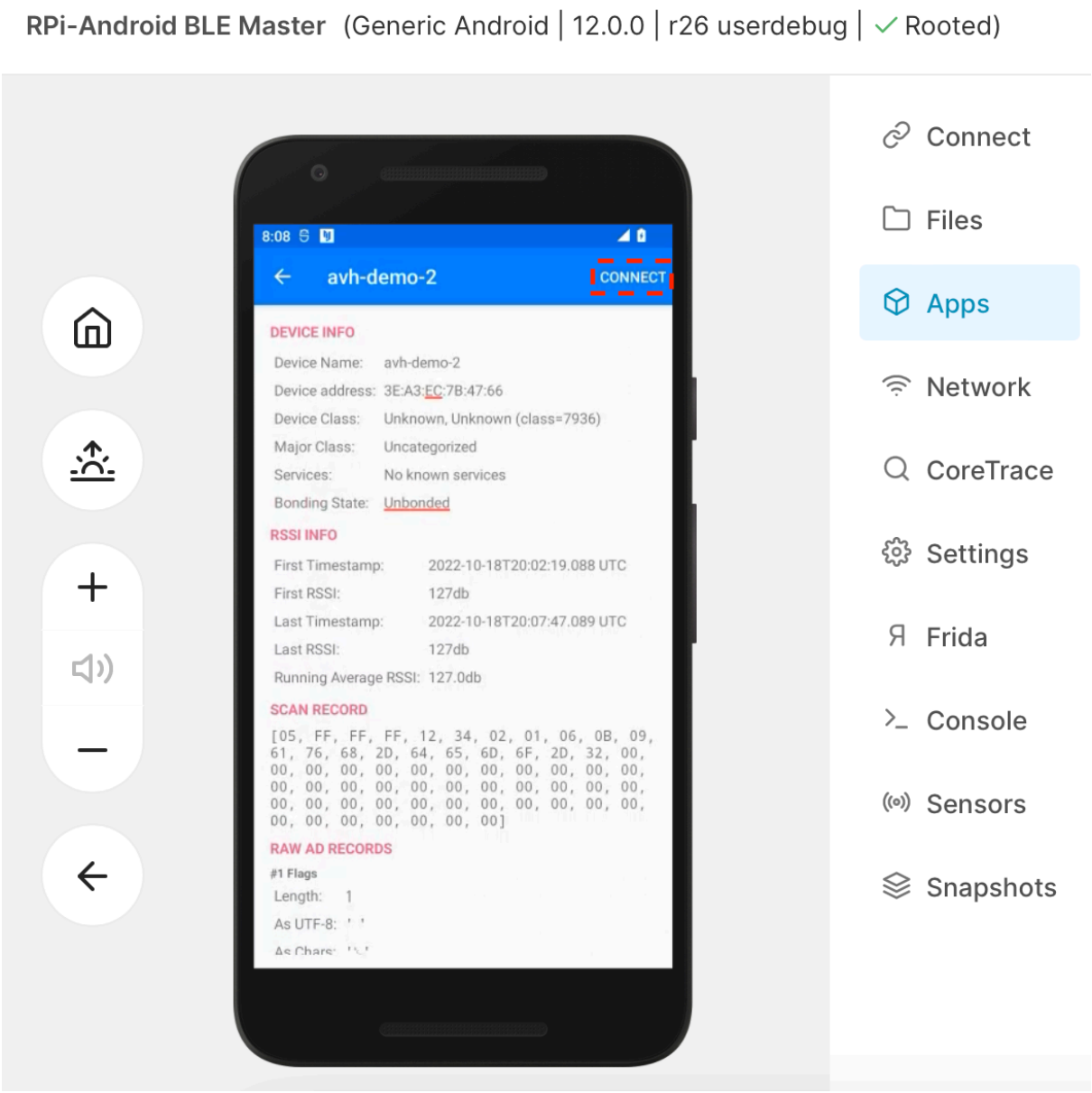

RPi-Android BLE Master (Generic Android | 12.0.0 | r26 userdebug | Rooted)
8:09
avh-demo-2
CONNECT
Last RSSI: 127db
Running Average RSSI: 127.0db
SCAN RECORD
[05, FF, FF, FF, 12, 34, 02, 01, 06, 0B, 09, 61, 76, 68, 2D, 64, 65, 6D, 6F, 2D, 32, 00, 00, 00

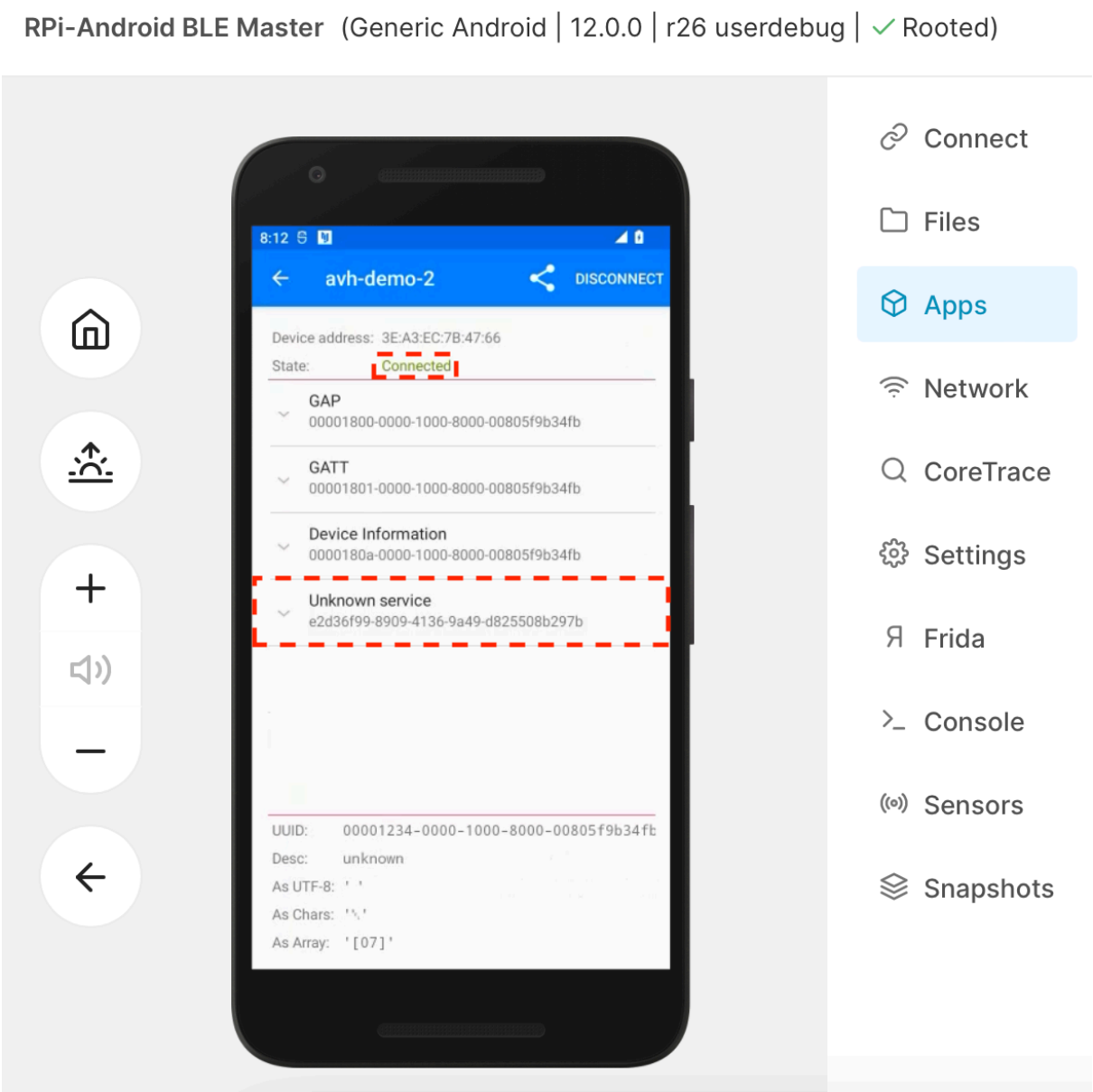
Figure 10-34: Connect to AVH demo



6. After the device connects successfully, you will see an `Unknown Service` with the ID matching what we set up. Expand the service.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

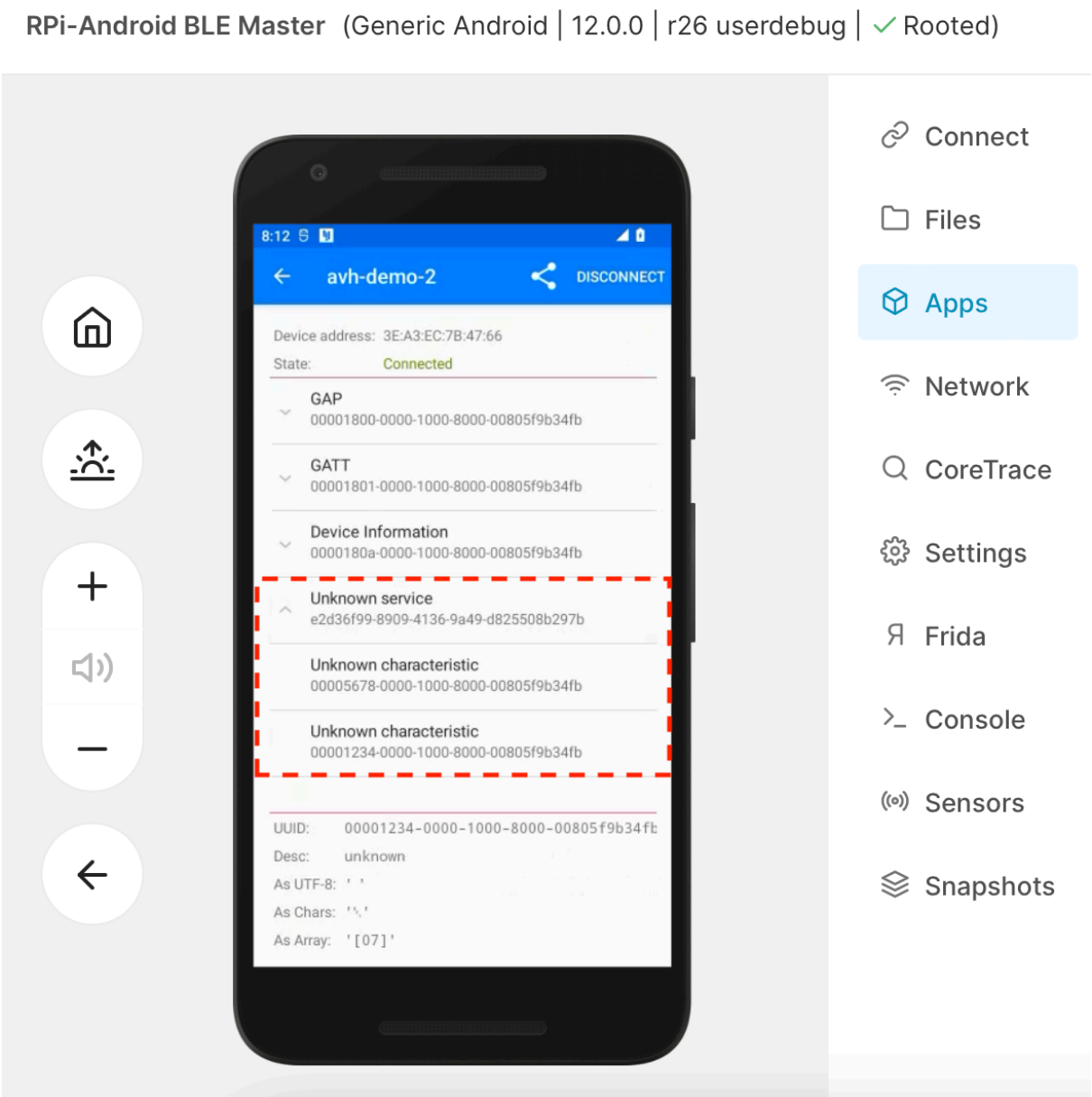
Figure 10-35: Unknown service screen



7. You will see the two custom characteristics `1234` and `5678` that we set up.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

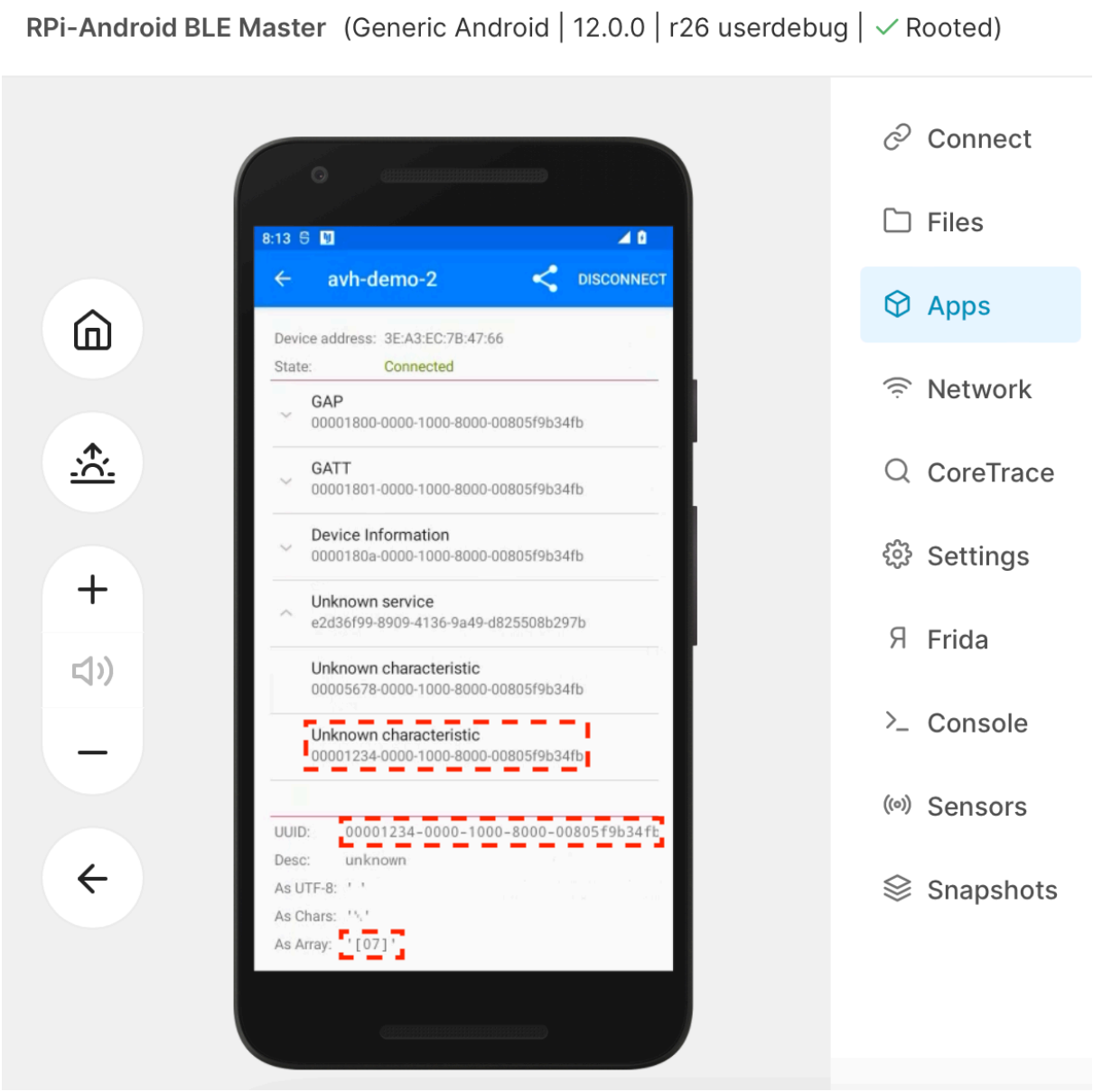
Figure 10-36: Custom characteristics



8. Click on the `1234` and the `5678` services to see the values we had set previously (`07` and `0d`, respectively).

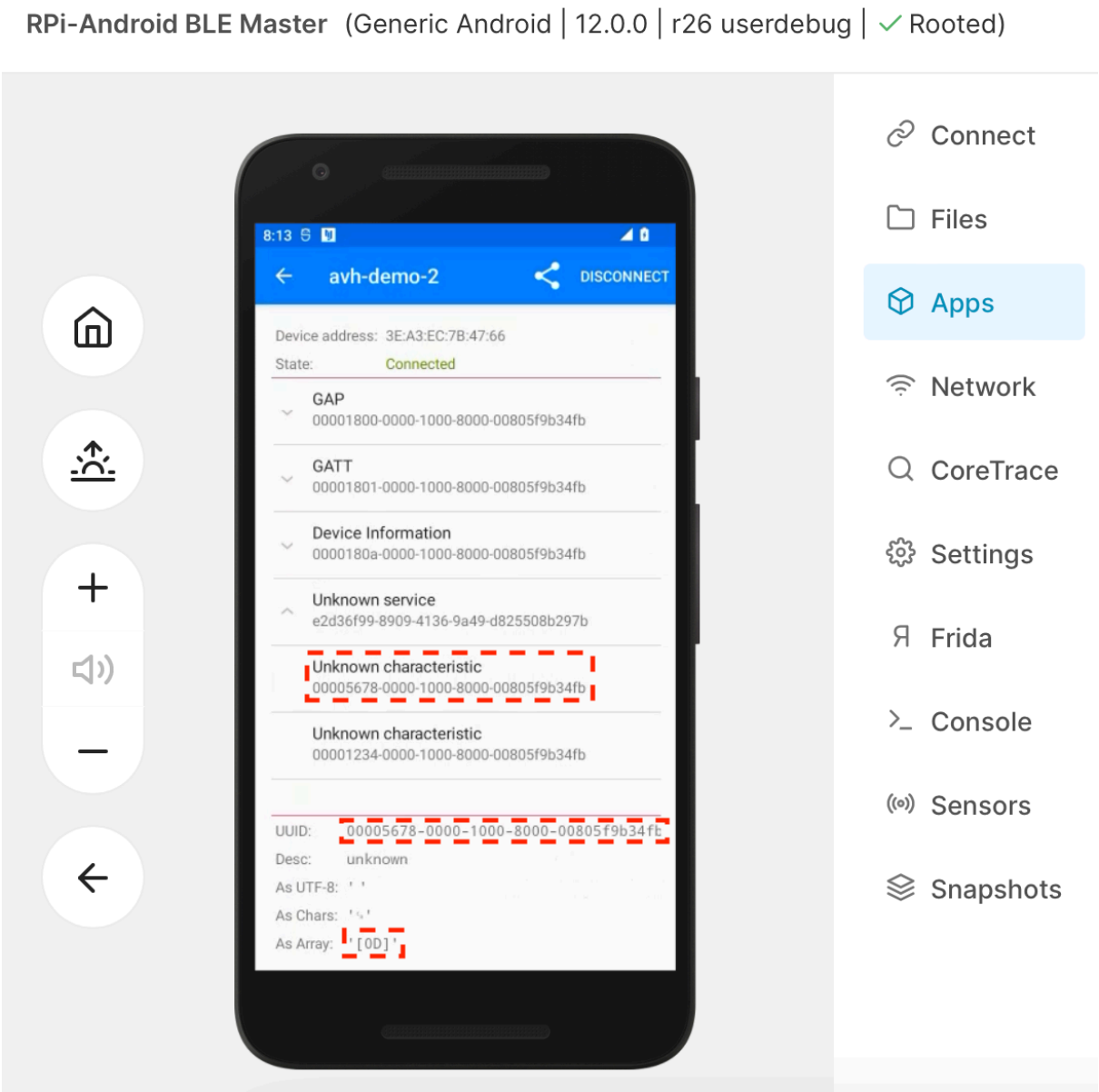
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-37: Custom characteristics values



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 10-38: Custom characteristics values



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

11. Detect Audio through the Microphone with the STM32U5 Board

AVH includes support for connecting your computer's microphone to virtual devices. This article will demonstrate how to detect audio using the virtual STM32U5 board.

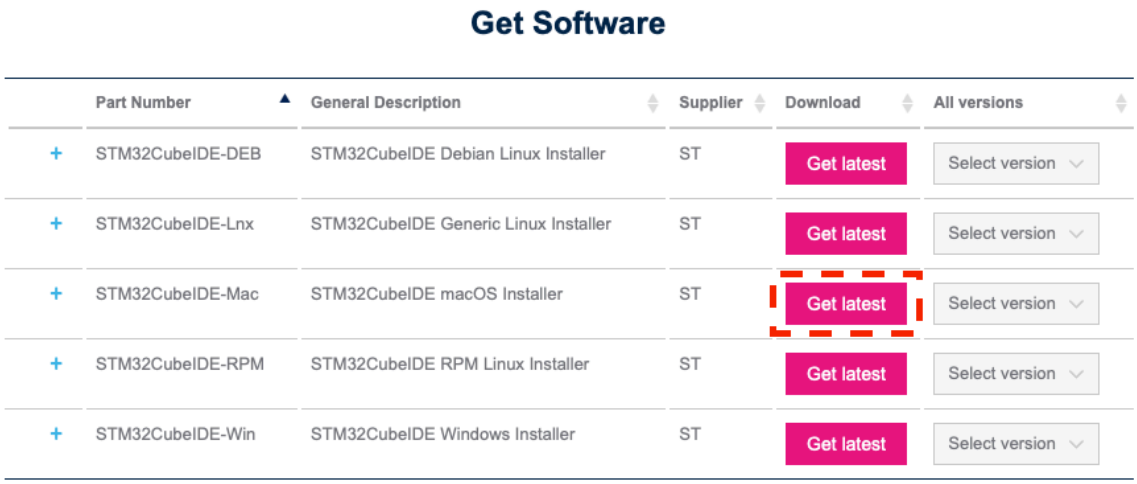
- Build the ADF_AudioSoundDetector Firmware File
- Boot the Custom Firmware and Detect Audio

11.1 Build the ADF_AudioSoundDetector Firmware File

Follow this process to build the required `.elf` custom firmware files using the free STM32CubeIDE tool from STMicroelectronics.

1. Install the STM32CubeIDE tool. In our example, we will use the MacOS version.

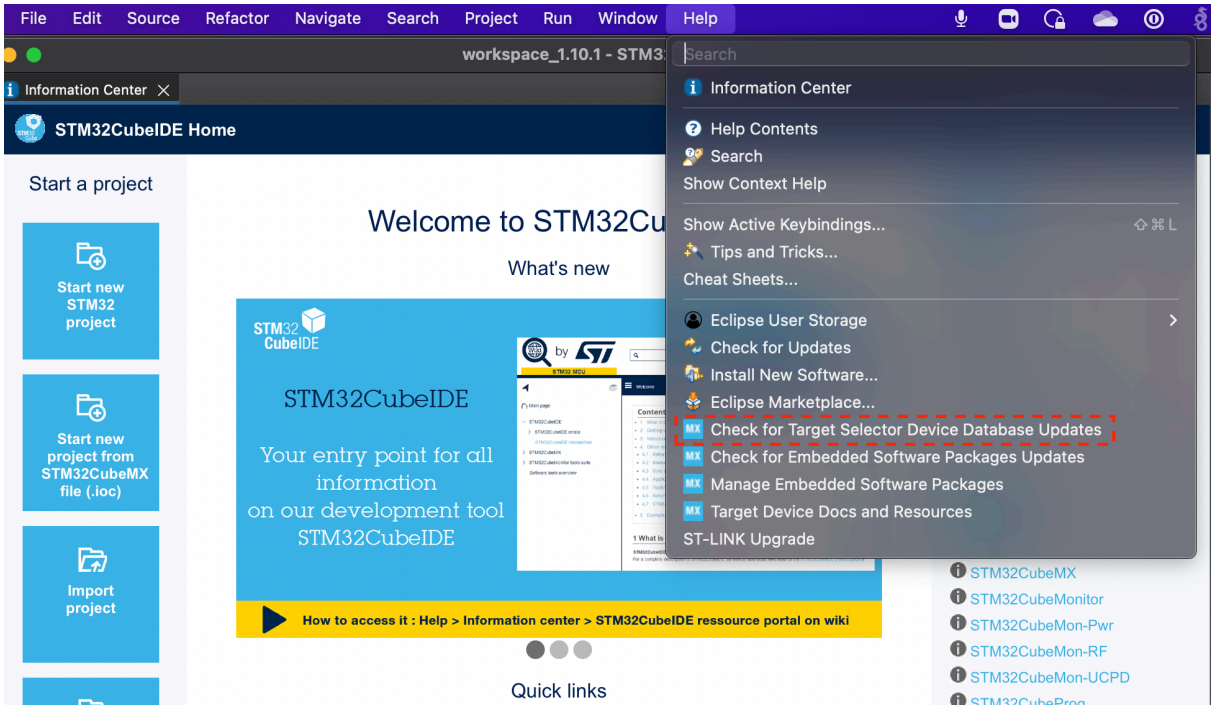
   Figure 11-1: Download CubeIDE



2. On the top menu, click Help then `Check for Target Selector Device Database Updates`.

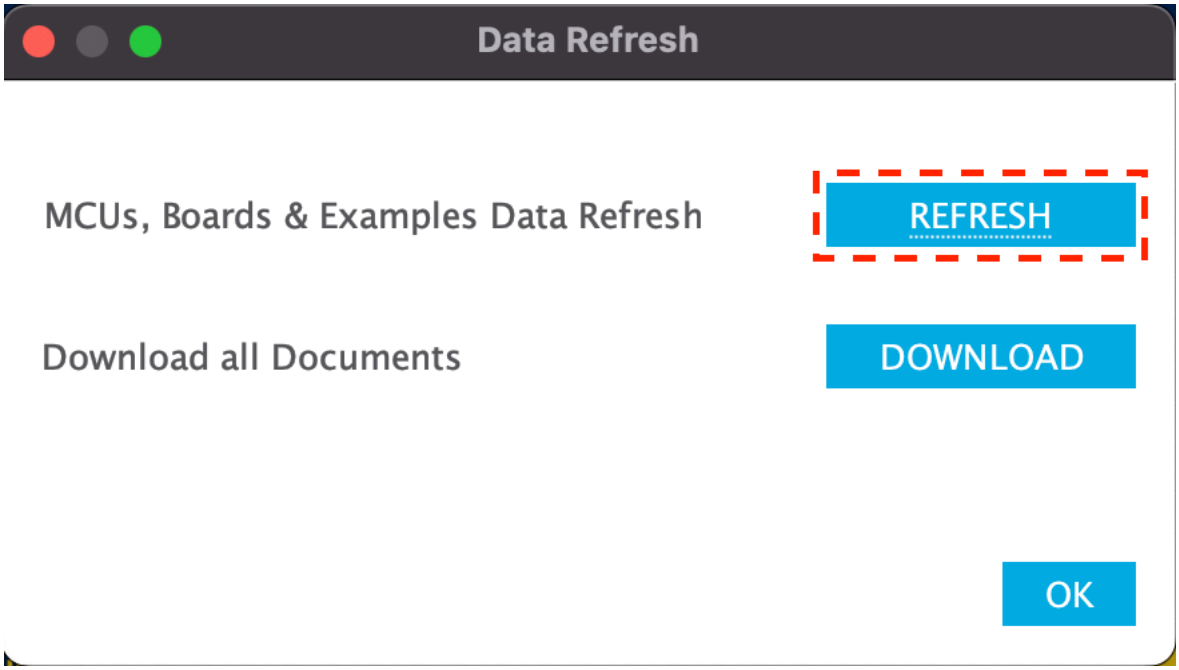
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 11-2: Check for updates



3. On the popup window click REFRESH. Your project files will update.

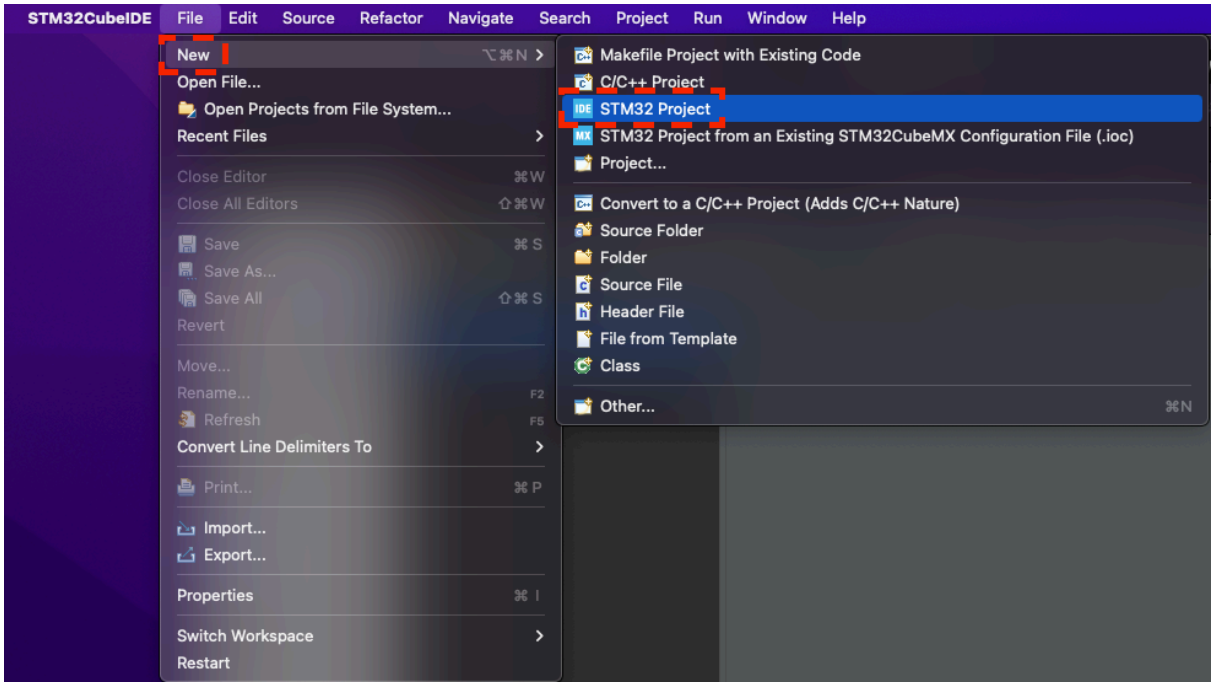
Figure 11-3: Refresh button



4. Create a new project by clicking New then STM32 Project from the File menu.

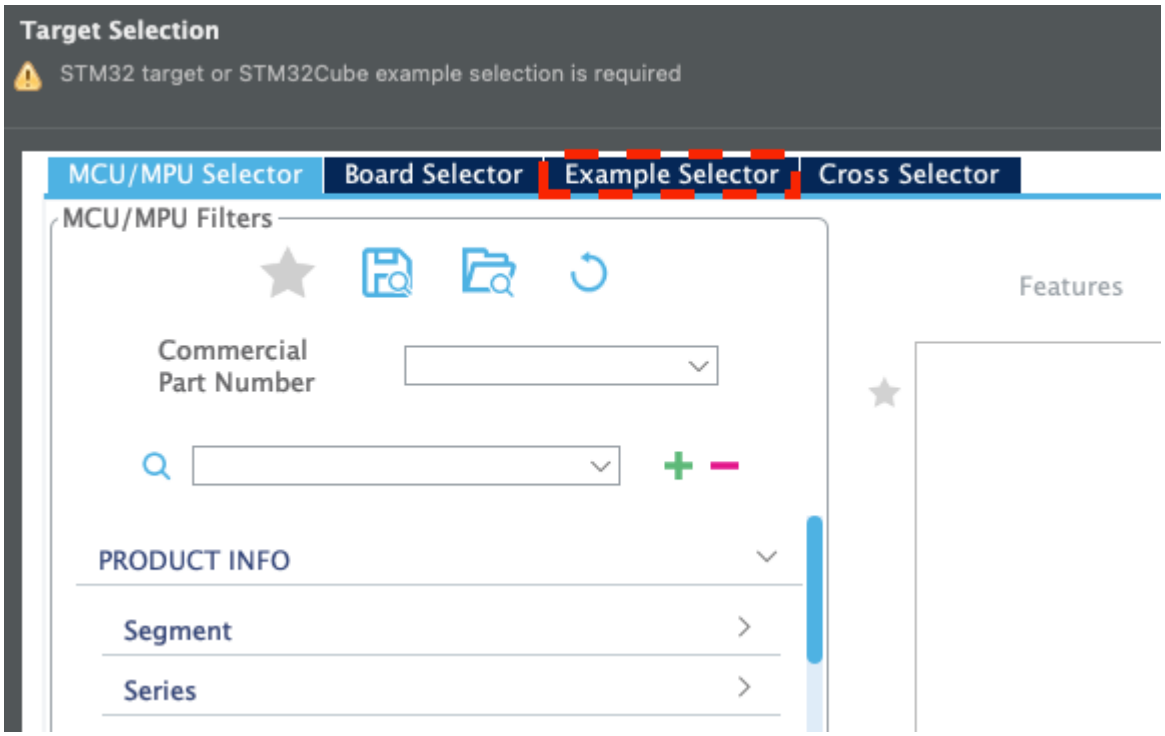
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 11-4: New project screen



5. Click on Example Selector.

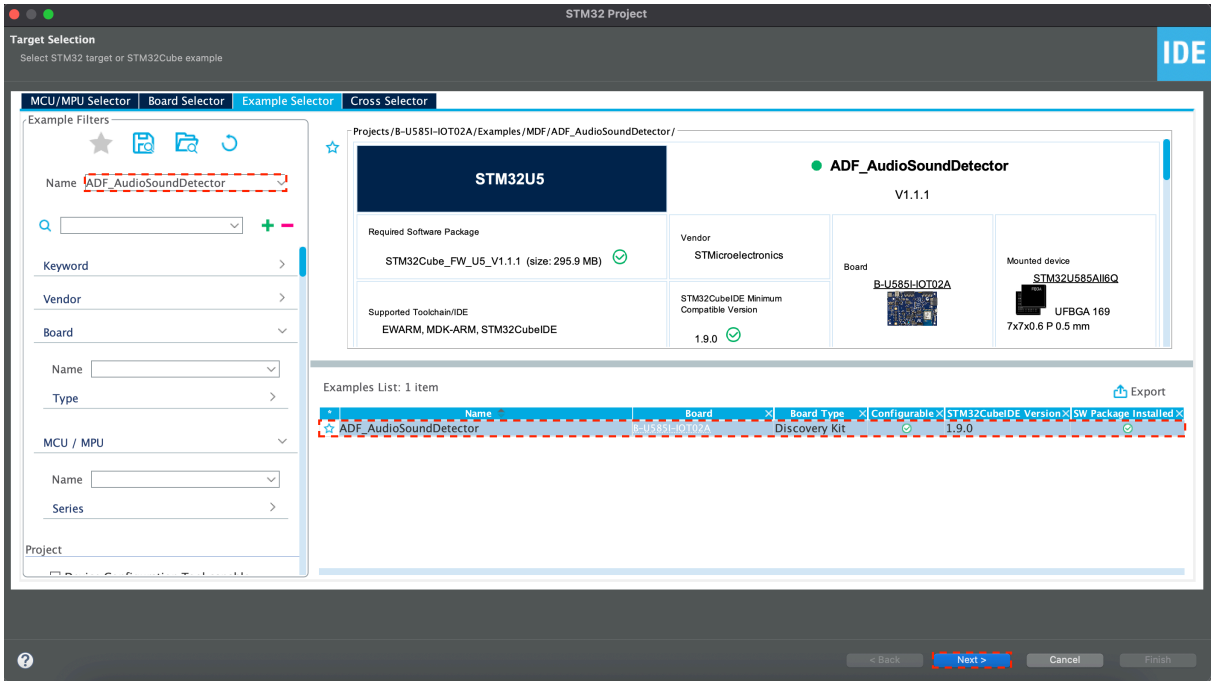
Figure 11-5: Target selection



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

6. Set the Name to ADF_AudioSoundDetector, click on the ADF_AudioSoundDetector example to the right, and click Next.

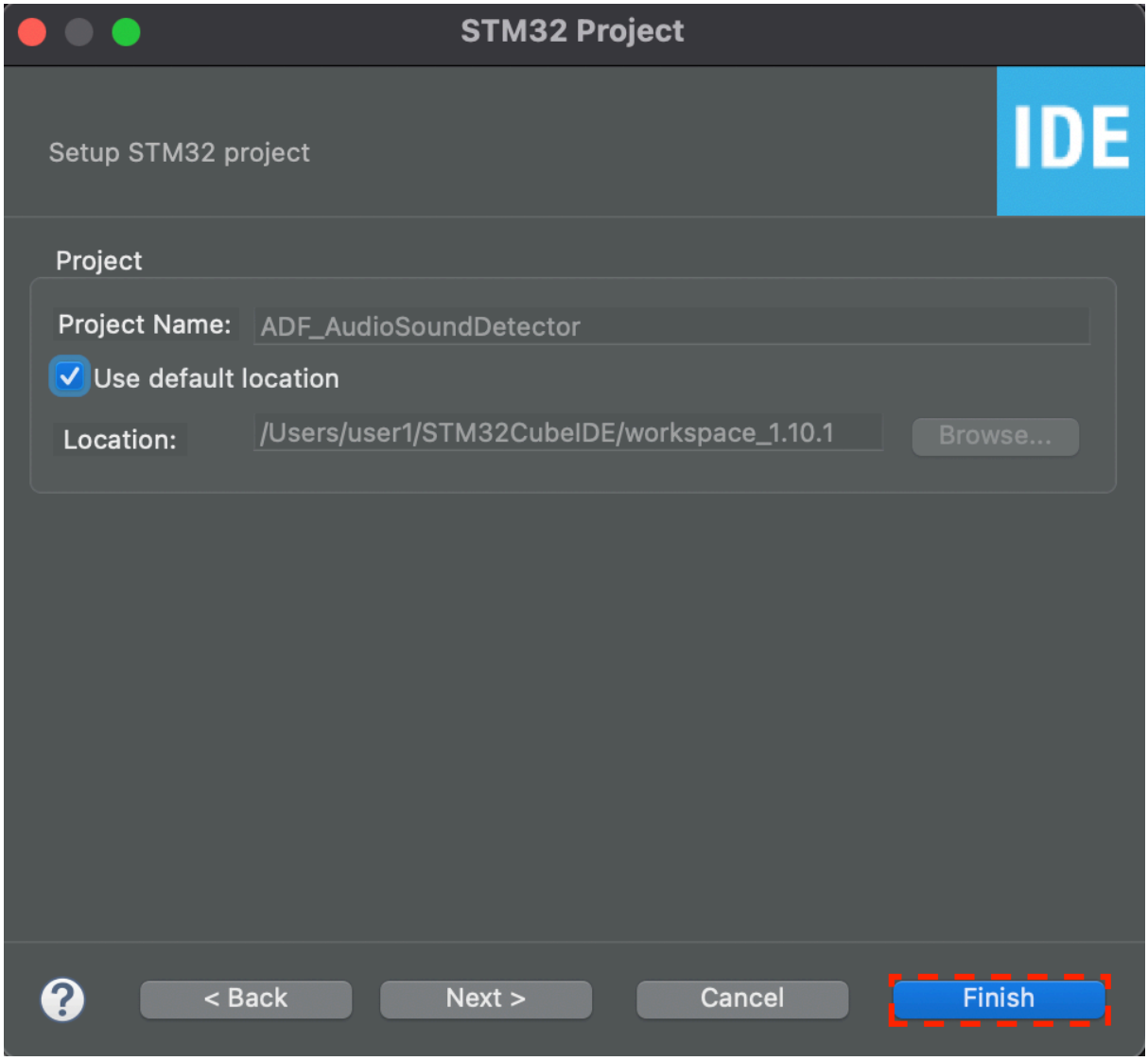
Figure 11-6: ADF_AudioSoundDetector example



7. Click Finish to create the project.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

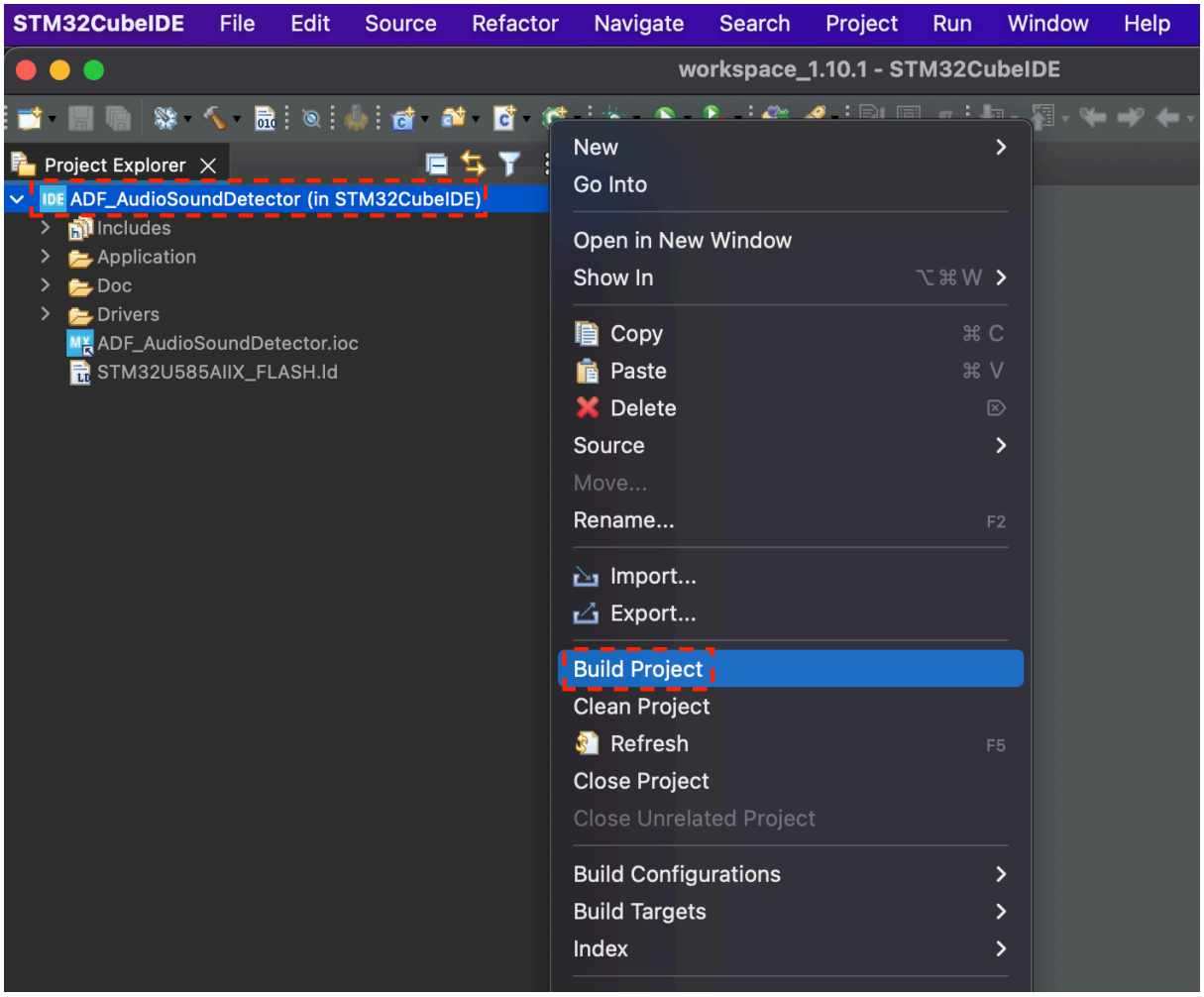
Figure 11-7: Finish project setup



8. In the Project Explorer window, right-click on the ADF_AudioSoundDetector project and choose Build Project.

Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

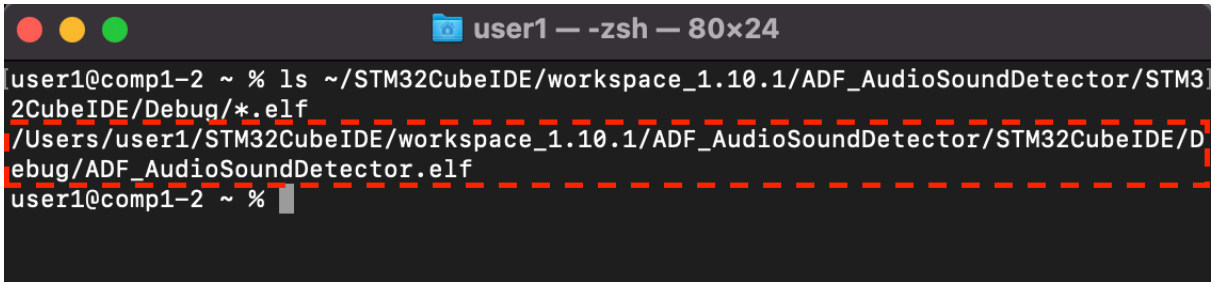
Figure 11-8: Build project



9. The default path for your firmware file is: `~/STM32CubeIDE/workspace_1.0.1/ADF_AudioSoundDetector/STM32CubeIDE/Debug/ADF_AudioSoundDetector.elf`.

```
ls ~/STM32CubeIDE/workspace_1.10.1/ADF_AudioSoundDetector/STM32CubeIDE/Debug/*.el
```

Figure 11-9: .elf File



10. Proceed to the next section to boot the custom firmware file.

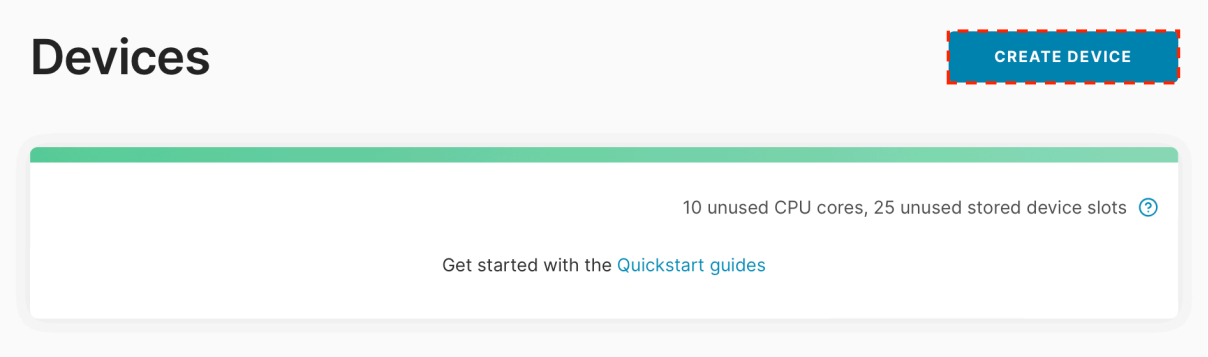
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

11.2 Boot the Custom Firmware and Detect Audio

Follow this process to boot the custom firmware and detect audio:

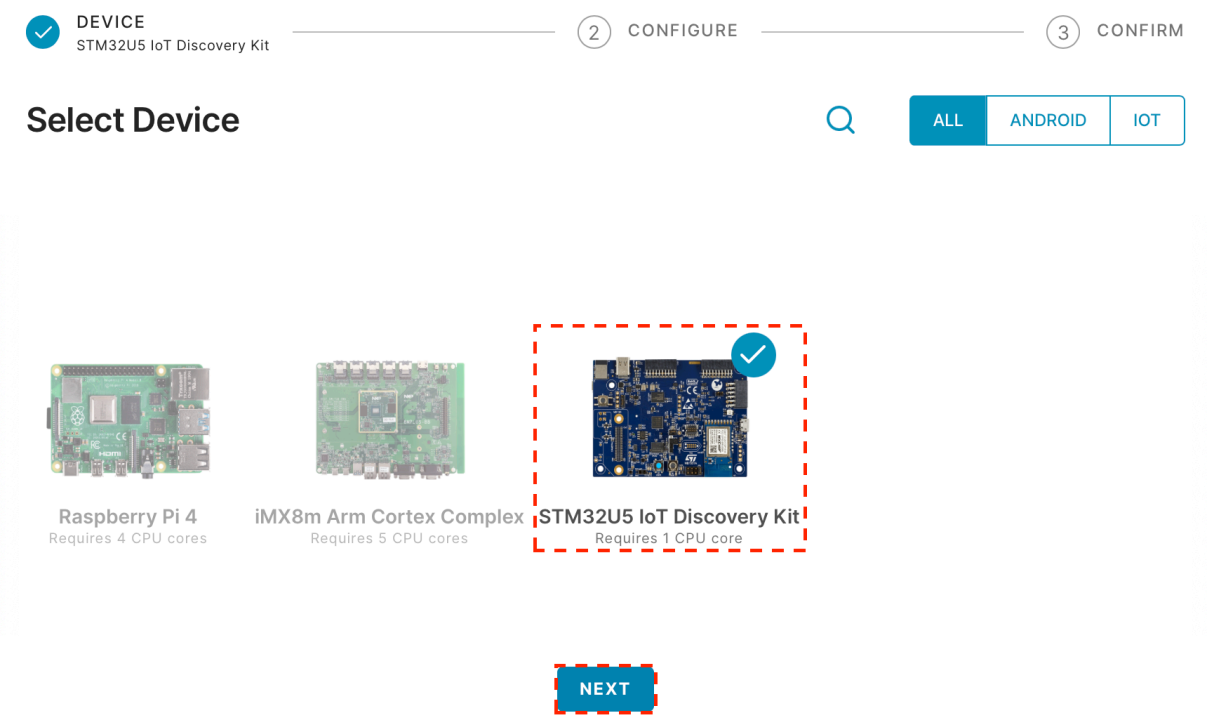
1. Log in to the AVH platform and click CREATE DEVICE.

Figure 11-10: Create device



2. Select the STM32U5 device and click NEXT.

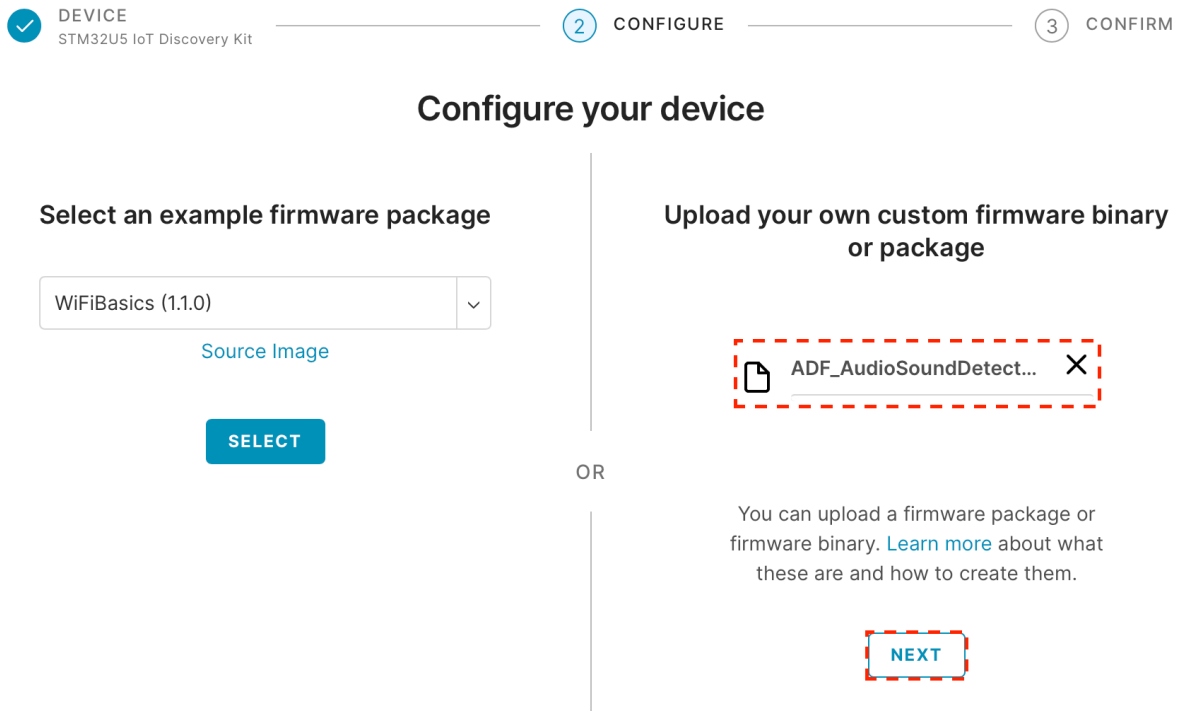
Figure 11-11: Select device



3. Upload your ADF_AudioSoundDetector.elf custom firmware file click NEXT, then choose CREATE DEVICE.

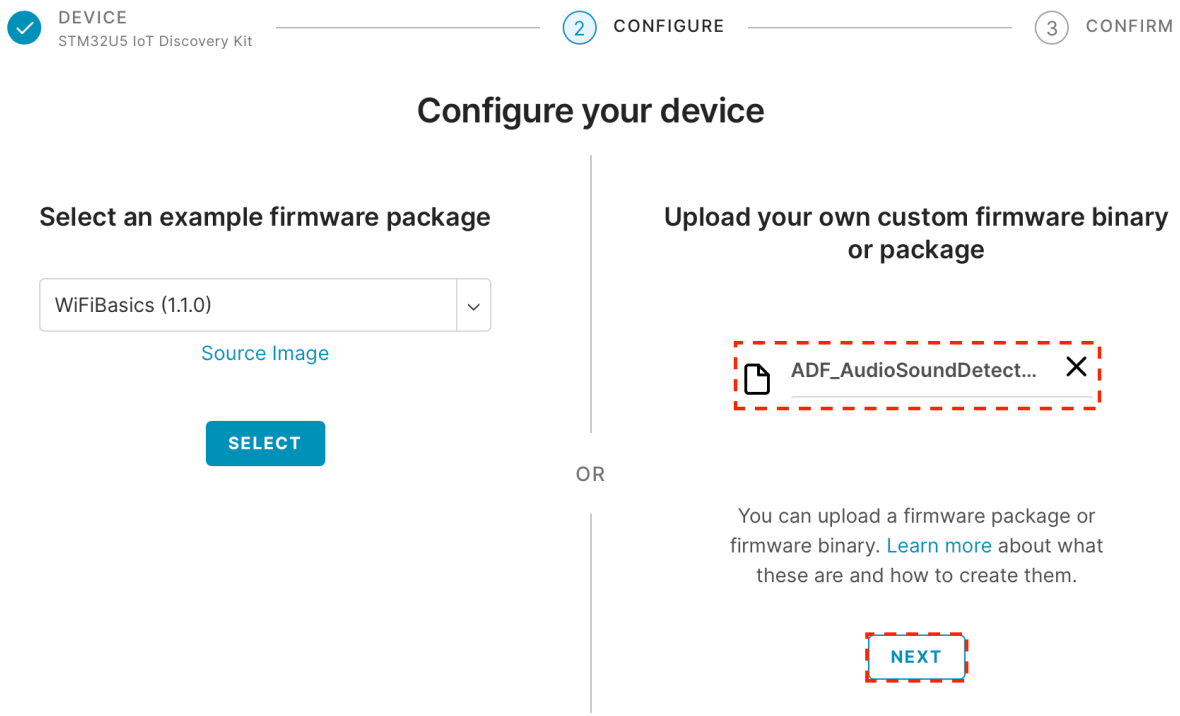
Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

Figure 11-12: Upload firmware



4. After your device boots, open the Sensors tab and enable the microphone.

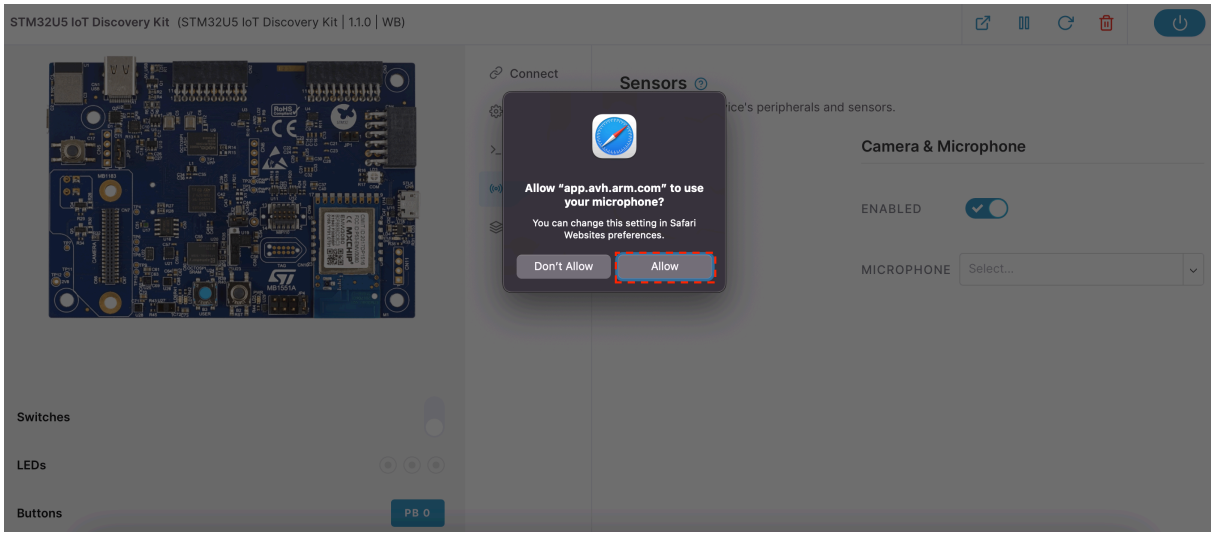
Figure 11-13: Enable microphone



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential

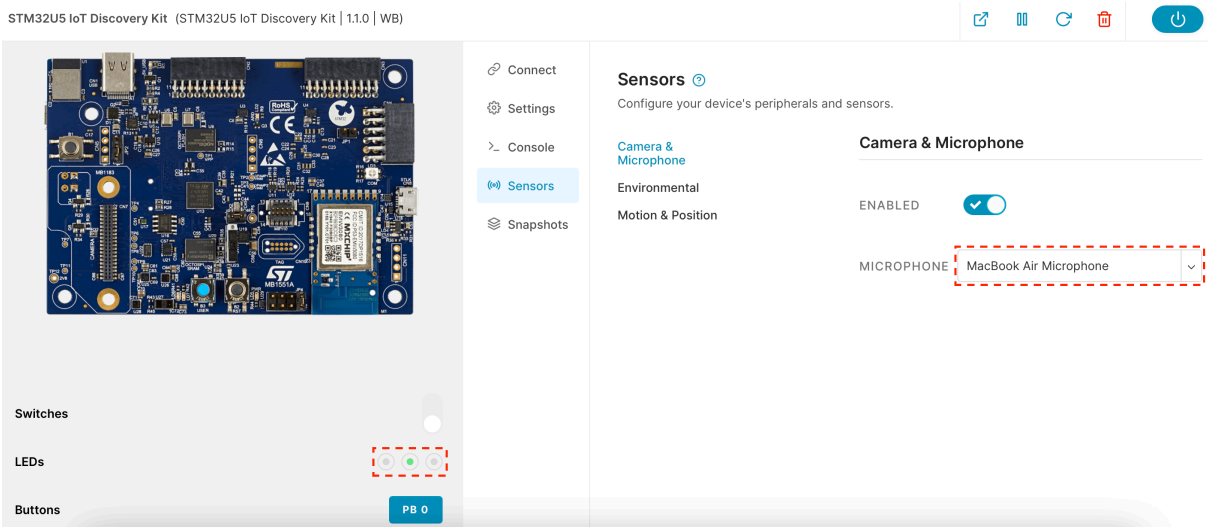
5. Confirm on your browser that you would like to share the microphone.

Figure 11-14: Share microphone with browser



6. The device should begin automatically detecting sound. You will see LED #2 lighting green to indicate audio is detected.

Figure 11-15: Audio detection



Copyright © 2022 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential