GETTING STARTED GUIDE

Apollo4 Family SDK

Ultra-Low Power Apollo SoC Family

A-SOCAP4-GGGA01EN v1.1

Legal Information and Disclaimers

AMBIQ MICRO INTENDS FOR THE CONTENT CONTAINED IN THE DOCUMENT TO BE ACCURATE AND RELIABLE. THIS CONTENT MAY, HOWEVER, CONTAIN TECHNICAL INACCURACIES, TYPOGRAPHICAL ERRORS OR OTHER MISTAKES. AMBIQ MICRO MAY MAKE CORRECTIONS OR OTHER CHANGES TO THIS CONTENT AT ANY TIME. AMBIQ MICRO AND ITS SUPPLIERS RESERVE THE RIGHT TO MAKE CORRECTIONS, MODIFICATIONS, ENHANCEMENTS, IMPROVEMENTS AND OTHER CHANGES TO ITS PRODUCTS, PROGRAMS AND SERVICES AT ANY TIME OR TO DISCONTINUE ANY PRODUCTS, PROGRAMS, OR SERVICES WITHOUT NOTICE.

THE CONTENT IN THIS DOCUMENT IS PROVIDED "AS IS". AMBIQ MICRO AND ITS RESPECTIVE SUPPLIERS MAKE NO REPRESENTATIONS ABOUT THE SUITABILITY OF THIS CONTENT FOR ANY PURPOSE AND DISCLAIM ALL WARRANTIES AND CONDITIONS WITH REGARD TO THIS CONTENT, INCLUDING BUT NOT LIMITED TO, ALL IMPLIED WARRANTIES AND CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHT.

AMBIQ MICRO DOES NOT WARRANT OR REPRESENT THAT ANY LICENSE, EITHER EXPRESS OR IMPLIED, IS GRANTED UNDER ANY PATENT RIGHT, COPYRIGHT, MASK WORK RIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT OF AMBIQ MICRO COVERING OR RELATING TO THIS CONTENT OR ANY COMBINATION, MACHINE, OR PROCESS TO WHICH THIS CONTENT RELATE OR WITH WHICH THIS CONTENT MAY BE USED.

USE OF THE INFORMATION IN THIS DOCUMENT MAY REQUIRE A LICENSE FROM A THIRD PARTY UNDER THE PATENTS OR OTHER INTELLECTUAL PROPERTY OF THAT THIRD PARTY, OR A LICENSE FROM AMBIQ MICRO UNDER THE PATENTS OR OTHER INTELLECTUAL PROPERTY OF AMBIQ MICRO.

INFORMATION IN THIS DOCUMENT IS PROVIDED SOLELY TO ENABLE SYSTEM AND SOFTWARE IMPLEMENTERS TO USE AMBIQ MICRO PRODUCTS. THERE ARE NO EXPRESS OR IMPLIED COPYRIGHT LICENSES GRANTED HEREUNDER TO DESIGN OR FABRICATE ANY INTEGRATED CIRCUITS OR INTEGRATED CIRCUITS BASED ON THE INFORMATION IN THIS DOCUMENT. AMBIQ MICRO RESERVES THE RIGHT TO MAKE CHANGES WITHOUT FURTHER NOTICE TO ANY PRODUCTS HEREIN. AMBIQ MICRO MAKES NO WARRANTY, REPRESENTATION OR GUARANTEE REGARDING THE SUITABILITY OF ITS PRODUCTS FOR ANY PARTICULAR PURPOSE, NOR DOES AMBIQ MICRO ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT OR CIRCUIT, AND SPECIFICALLY DISCLAIMS ANY AND ALL LIABILITY, INCLUDING WITHOUT LIMITATION CONSEQUENTIAL OR INCIDENTAL DAMAGES. “TYPICAL” PARAMETERS WHICH MAY BE PROVIDED IN AMBIQ MICRO DATA SHEETS AND/OR SPECIFICATIONS CAN AND DO VARY IN DIFFERENT APPLICATIONS AND ACTUAL PERFORMANCE MAY VARY OVER TIME. ALL OPERATING PARAMETERS, INCLUDING “TYPICALS” MUST BE VALIDATED FOR EACH CUSTOMER APPLICATION BY CUSTOMER’S TECHNICAL EXPERTS. AMBIQ MICRO DOES NOT CONVEY ANY LICENSE UNDER NEITHER ITS PATENT RIGHTS NOR THE RIGHTS OF OTHERS. AMBIQ MICRO PRODUCTS ARE NOT DESIGNED, INTENDED, OR AUTHORIZED FOR USE AS COMPONENTS IN SYSTEMS INTENDED FOR SURGICAL IMPLANT INTO THE BODY, OR OTHER APPLICATIONS INTENDED TO SUPPORT OR SUSTAIN LIFE, OR FOR ANY OTHER APPLICATION IN WHICH THE FAILURE OF THE AMBIQ MICRO PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR. SHOULD BUYER PURCHASE OR USE AMBIQ MICRO PRODUCTS FOR ANY SUCH UNINTENDED OR UNAUTHORIZED APPLICATION, BUYER SHALL INDEMNIFY AND HOLD AMBIQ MICRO AND ITS OFFICERS, EMPLOYEES, SUBSIDIARIES, AFFILIATES, AND DISTRIBUTORS HARMLESS AGAINST ALL CLAIMS, COSTS, DAMAGES, AND EXPENSES, AND REASONABLE ATTORNEY FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PERSONAL INJURY OR DEATH ASSOCIATED WITH SUCH UNINTENDED OR UNAUTHORIZED USE, EVEN IF SUCH CLAIM ALLEGES THAT AMBIQ MICRO WAS NEGLIGENT REGARDING THE DESIGN OR MANUFACTURE OF THE PART.

Revision History

Revision	Date	Description
1.0	May 25, 2022	Initial release
1.1	May 25, 2023	Correction of a typographical error in Ch. 8

Reference Documents

Document ID	Description
DS-A4-1p0p0	Apollo4 SoC Datasheet
DS-A4P-1p0p1	Apollo4 Plus SoC Datasheet
DS-A4B-1p0p0	Apollo4 Blue SoC Datasheet
DS-A4BP-1p1p0	Apollo4 Blue Plus SoC Datasheet

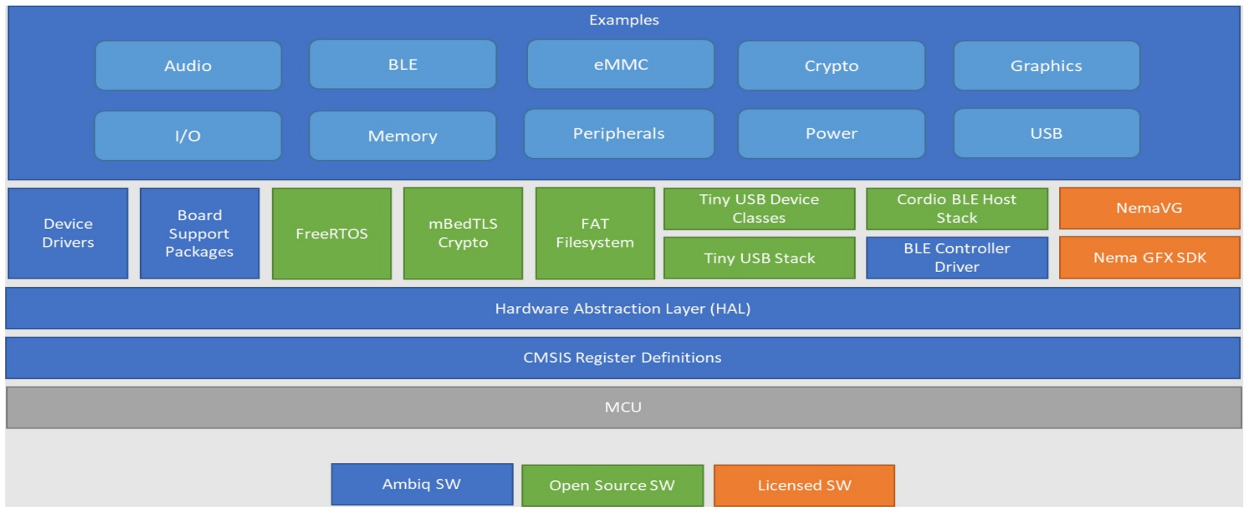
Table of Contents

SECTION 1

Introduction

The Apollo4 Family SDK is composed of a combination of Ambiq software, open-source software, and licensed software (binary libraries and source interface files). Figure 1-1 shows a "Layer Cake" type diagram of the SDK.

Figure 1-1: Apollo4 Family Software Stack



SECTION

2

Installing/Configuring Development Tools

Ambiq's Evaluation Boards (EVBs) all support the standard Segger J-Link debug interface. We recommend customers to install the latest Segger J-Link software, and configure your preferred development IDE to use J-Link debug interface.

Links to development tools that support Apollo4 family devices[1]:

- SEGGER J-Link Software: https://www.segger.com/downloads/jlink
- KEIL uVision 5[2]: https://www.keil.com/demo/eval/arm.htm
  - Latest Keil Pack: http://www.keil.com/dd2/pack/#/third-party-download-dialog
- IAR Version[3]: https://www.iar.com/iar-embedded-workbench/tools-for-arm/arm-cortex-m-edition/
- GCC: https://gcc.gnu.org

[1]The actual tool revisions are subject to change and are specified in the /docs/releasenotes/AmbiqSuite SDK Release Notes.txt included in each SDK. Tool updates specifically to support Apollo4 Plus are underway. In the interim, developers may select AMAP42KK-KBR with the understanding that there are some register differences from Apollo4.
[2]Apollo4 is supported by installing the CMSIS Ambiq Pack v1.2.9 or later and selecting the AMAP42KK-KBR device.
[3]Apollo4 is natively supported with EWARM v9.10.2 and later and selecting the AMAP42KK-KBR device.

SECTION

3

Top Level Directory Layout

The following listing is the top-level directory tree of the AmbiqSuite SDK focused on the most important directories for getting started.

```
.
├── boards
├── CMSIS
├── devices
├── docs
├── mcu
├── third_party
├── tools
├── utils
```

The **boards** directory contains the Board Support Packages (BSP) and supported examples for each customer evaluation boards/kits. The **CMSIS** directory contains the automatically generated register definition files for each supported device. The **devices** directory contains the device drivers for external devices supported on our evaluation boards/kits. The **docs** directory contains the release notes, html register documentation, and license files for the SDK. The **mcu** directory contains the HAL and additional register/instance information. The **third_party** directory contains the bundled licensed and open-source software included with the SDK. The **tools** directory contains (mostly) python scripts and tools supporting BSP generation, security features, over-the-air download, and more. The **utils** directory contains a series of useful C utilities for application development.

SECTION

4

Embedded Documentation and Licenses

Each release contains some related documentation that is specific to the SDK itself. These are located as follows:

```
├── docs
|    ├── licenses
|    ├── registers
|    └── releasenotes
```

The **licenses** directory contains the applicable open-source licenses for the source code that we include in the SDK. This includes the cordio BLE host stack, FreeRTOS, mBedTLS crypto, and tinyUSB code. The **registers** directory contains an HTML document for each supported device. Opening the index.html file will bring up the main menu and allow navigation of registers for each hardware block in the MCU. The **releasenotes** directory contains the main release notes for the release as well as the change history for each device.

SECTION

5

Hardware Abstraction Layer (HAL)

Ambiq defines a Hardware Abstraction Layer (HAL) as a set of functional APIs for controlling the hardware blocks in the Apollo4 SoCs. These functional APIs provide for initialization and operation of a block while abstracting away the register level interfaces of the block. The **apollo4b** directory supports Apollo4/Apollo4 Blue devices, while **apollo4p** supports Apollo4 Plus/Apollo4 Blue Plus devices.

```
├── mcu
|   ├── am_sdk_version.h
|   ├── apollo4b
|   |   ├── am_mcu_apollo.h
|   |   ├── hal
|   |   └── regs
|   ├── apollo4p
|   |   ├── am_mcu_apollo.h
|   |   ├── hal
|   |   └── regs
```

The HAL is built as a stand-alone library as a convenience for generation of examples.

SECTION 6

Board Support Packages (BSP)

The Board Support Packages (BSP) are customized mapping of device pins to a given board hardware implementation. The directory tree below shows the four target evaluation boards (EVBs) or kits provided for a customer to evaluate Apollo4/Apollo4 Blue and Apollo4 Plus/Apollo4 Blue Plus[1].

```
├── boards
|    ├── apollo4b_blue_evb
|    |    ├── bsp
|    ├── apollo4b_evb
|    |    ├── bsp
|    ├── apollo4b_evb_disp_shield
|    |    ├── bsp
|    ├── apollo4p_blue_evb
|    |    ├── bsp
|    ├── apollo4p_evb
|    |    ├── bsp
```

The BSP is built as a stand-alone library as a convenience for generation of examples.

[1] Note that the board targets supplied with each SDK is subject to changes and additions as new devices and target hardware are released. This directory tree is an example of where to find the board specific packages.

SECTION 7

Examples

The examples are created for each board for which they make sense. The directory tree below shows the example for the Apollo4 Blue EVB and the Apollo4 Display Kit. They are classified into directories **audio**, **ble**, **bm_sdmmc_sdio** (eMMC), **crypto**, **graphics**, **interfaces**, **memory**, **peripherals**, **power** and **usb** based on function[1].

```
├── apollo4b_blue_evb
│   ├── examples
│   │   ├── audio
│   │   │   ├── i2s_loopback
│   │   │   └── pdm_to_i2s
│   │   ├── ble
│   │   │   ├── ble_firmware_update
│   │   │   ├── ble_freertos_amota
│   │   │   ├── ble_freertos_fcc_test
│   │   │   └── uart_ble_bridge
│   │   ├── crypto
│   │   │   └── rsa_sign_verify
│   │   ├── memory
│   │   │   ├── info_dump
│   │   ├── peripherals
│   │   │   ├── adc_lpmode0_dma
│   │   │   ├── adc_measure
│   │   │   ├── binary_counter
│   │   │   ├── hello_world
```

[1] Note that the examples released with each SDK is subject to change. This directory tree illustrates where to find different types of examples.

```
|    |    |    ├── rtc_print
|    |    |    ├── stimer
|    |    |    └── watchdog
|    |    ├── power
|    |    |    ├── coremark
|    |    |    ├── deepsleep
|    |    └── usb
|    |         ├── tinyusb_cdc
├── apollo4b_evb_disp_shield
|    ├── examples
|    |    ├── bm_sdmmc_sdio
|    |    |    ├── emmc_bm_fatfs
|    |    |    ├── emmc_raw_block_read_write
|    |    ├── graphics
|    |    |    ├── nemadc_darkening_effect
|    |    |    ├── nemadc_scrolling_effect
|    |    |    ├── nemagfx_balls_bench
|    |    |    ├── nemagfx_benchmarks
|    |    ├── interfaces
|    |    |    ├── ios_fifo
|    |    |    ├── mpu_mspi_ddr_octal_psram_example
|    |    └── usb
|    |         └── tinyusb_cdc_msc_emmc
```

SECTION 8

Tools

8.1 BSP Custom Pin Mapping

Pin mapping is accomplished by the source file called bsp_pins.src in each **bsp** directory. These "canned" BSPs can be modified and rebuilt using the tools below to allow the customer to create a custom BSP for their own development boards. The output is the am_bsp_pins.* files created in the **bsp** directory.

```
├── tools
|   ├── bsp_generator
|   |   ├── apollo4_pinconfig.py
|   |   ├── pinconfig.py
|   |   └── rsonlite.py
```

8.2 INFO0 and Basic Configuration

While an EVB will come with INFO0 programmed, the customer may want to update the parameters or security settings for certain testing. The tools for creation of INFO0 binary is located in **\tools\apollo4b_scripts\create_info0.py**.

The EVB comes pre-programmed as follows:

- Defaults to non-secure boot mode.
- Enables boot override to push button on GPIO18 which is BTN0/SW1 on the EVBs
- Wired updates via UART. UART0 is mapped to JLink debugger. Baud rate is 115200 bps, no parity, 8-bit data length, no flow control. UART mapped to GPIO47 and GPIO60.
- UART timeout is 3 seconds.
- SIMOBuck is not enabled.
- All NVM and Debugger protections are disabled.

The script to accomplish this can be executed in that directly as follows:

```
python3 ./create_info0.py --valid 1 --u0 0x1C200c0 --u1 0xFFFF3c2f --
u2 0x4 --u3 0x4 --u4 0x0 --u5 0x0 --main 0x18000 --version 1 --wTO
3000 --sdcert 0x1ff400 --trim 0x1 info0
```

This will generate info0.bin, which can then be programmed to the device.

AmbiqSuite SDK also provides a sample script **\tools\apollo4b_scripts\jlink-prog-info0.txt** for the direct programming of Info0 using debugger, using the first method. This script can be processed by the JLINK command line utility with an invocation following the following format (for Windows):

```
JLink.exe -CommanderScript jlink-prog-info0.txt
```

8.3 Changing the Memory Map[1]

The memory map for all the projects is determined by a set of configuration files as follows:

```
\tools\config\apollo4b-system-config.yaml
\tools\config\apollo4l-system-config.yaml
\tools\config\apollo4p-system-config.yaml
```

Modifying one of these system-config.yaml files will change the memory map for all corresponding compiled projects. To modify the memory map for just the current project, simply copy the appropriate system-config.yaml file from the tools\config\ directory to the local project directory, for example:

\boards\apollo4p_evb\examples\power\deepsleep\apollo4p-system-config.yaml

[1] This feature was not working in R4.1.0 but will be fixed for R4.2.0.

SECTION 9

Creating a New Project

This section discusses how to start with the "hello_world" project and create a new project.

1. Copy and rename:

   ```
   \boards\apollo4p_evb\examples\peripherals\hello_world
   ```
   as
   ```
   \boards\apollo4p_evb\examples\peripherals\new_project
   ```

2. Go to the **\src** directory and rename **hello_world.c** to **new_project.c** there.

3. The following sections will discuss the remaining steps for each toolchain.

   - **GCC**
     Arm Embedded GCC cross compiler is driven by the Makefile in the **\gcc** subdirectory.
     a. Edit the Makefile and replace **hello_world** with **new_project** there.
     b. You should be able to do a "make clean; make" inside the **\gcc** directory which will result in a **new_project.bin** in the **\bin** directory.

   - **IAR**
     a. In the **\iar** directory, copy the **hello_world.ewp** file and rename it to **new_project.ewp**.
     b. In the Makefile, replace all the **hello_world** with **new_project**.
     c. Open the existing workspace **hello_world.eww** and go to **Project**->**Add Existing** and select the **new_project.ewp** file.
     d. Remove the **hello_world.c** from the project.
     e. Right-click on the **new_project** – **Debug** and **Add**->**Add Files** and select the **new_project.c** file.
     f. Do a **File**->**Save Workspace As** and save the workspace as **new_project.** You can now remove the **hello_world.*** files from the **/iar** directory.

- **Keil**
  a. In the **\keil** directory, rename the **hello_world.uvprojx** to **new_project.uvprojx** and **hello_world.uvoptx** to **new_project.uvoptx**.
  b. In the Makefile, replace all the **hello_world** with **new_project**.
  c. Next, open the **new_project.uvprojx** project file.
  d. Right-click on the **hello_world** top level folder and select **Options for Target 'hello_world'**, select the **Output** tab.
  e. Rename the **Name of Executable** to **new_project**.
  f. Right-click on the **hello_world** folder and rename to **new_project**.
  g. Finally, right-click on the **hello_world.c** file and select **Remove file 'hello_world**', then right click on the **src** folder and select **Add existing files to group 'src'**.
  h. Navigate to the **new_project.c** file created above and add it to the group.

© 2023 Ambiq Micro, Inc. All rights reserved.

6500 River Place Boulevard, Building 7, Suite 200, Austin, TX 78730
www.ambiq.com
sales@ambiq.com
+1 (512) 879-2850

A-SOCAP4-GGGA01EN v1.1
May 2023