QUICK START GUIDE

Apollo SoC Windows

Ultra-Low Power Apollo SoC Family

A-SOCAP1-QSGA02EN v1.0

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	January 12, 2023	Initial release

Reference Documents

Document ID	Description

Table of Contents

SECTION

1 Introduction

This document is intended to help users prepare a Windows system to connect to an Apollo Class SoC. The Ambiq Apollo SoC is currently supported by 3 tool chains:

1. Eclipse(MARS)/gcc
2. Keil MDK
3. IAR EWARM

Before installing or using any of these tool chains, the user should first install the Ambiq Control Center, as discussed in *Section 2 Installing and Using the Ambiq Control Center on page 6*. All of the examples shipped in the Ambiq Control Center come with source code as well as pre-compiled binaries from each of the tool chains. Even without a tool chain installed, one can run all of the example programs from the Ambiq Control Center by selecting the AMFLASH utility.

Once the Ambiq Control Center is installed, the user can then download the desired tool chain as discussed in the appropriate section below.

SECTION

2

Installing and Using the Ambiq Control Center

The Ambiq Control Center installer provides most of the tools needed for connecting to an Apollo Class SoC. This includes USB drivers for the in-circuit debugger interface, and a custom version of OpenOCD, along with a few other utilities. This portion of the document will explain the process of installing the Ambiq Control Center.

2.1 Running the Installer

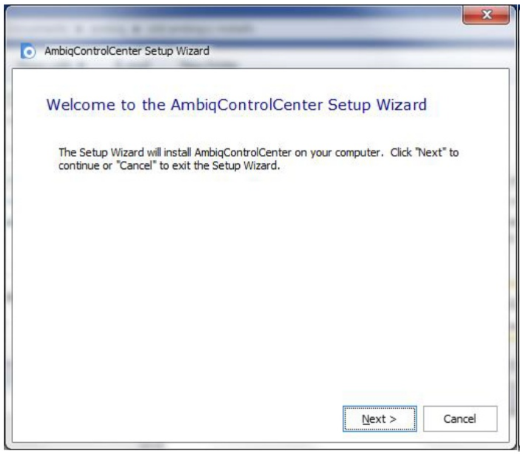
Along with the development hardware, an executable installer for Ambiq Control Center should be available

1. Obtain the installer by going to https://support.ambiq.com.

2. Run the installer.

3. After running the installer, a “Do you want to allow the following program from an unknown publisher to make changes to this computer?” message will popup. Answer **Yes** to this prompt.

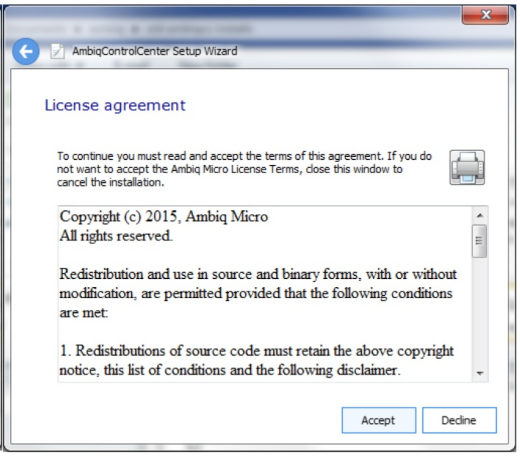
4. The AmbiqControlCenter initialization dialog will first be shown. After a few seconds, the Welcome panel will appear, click **Next** to advance to the License agreement screen, and click **Accept**.



AmbiqControlCenter Setup Wizard

Welcome to the AmbiqControlCenter Setup Wizard

The Setup Wizard will install AmbiqControlCenter on your computer. Click "Next" to continue or "Cancel" to exit the Setup Wizard.



AmbiqControlCenter Setup Wizard

License agreement

To continue you must read and accept the terms of this agreement. If you do not want to accept the Ambiq Micro License Terms, close this window to cancel the installation.

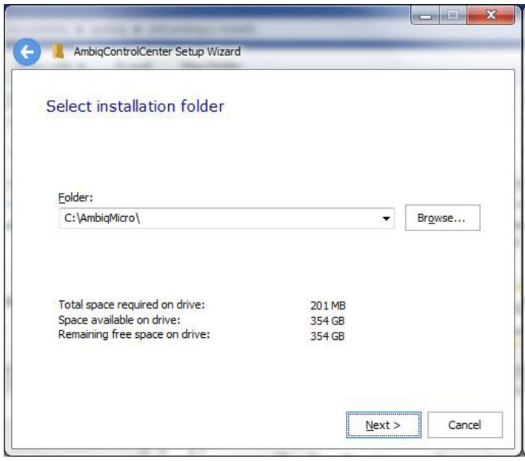
Copyright (c) 2015, Ambiq Micro
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

5. Keep the default **C:\AmbiqMicro\** as the installation folder, click **Next** to accept it. In the next panel, choose a **Typical** installation.

   This will install all of the necessary components for the AmbiqSuite. Note that the Eclipse and gcc based tool chain is installed separately and is covered in *Section 3 Installing and Using the Eclipse/gcc Tool Chain on page 24*

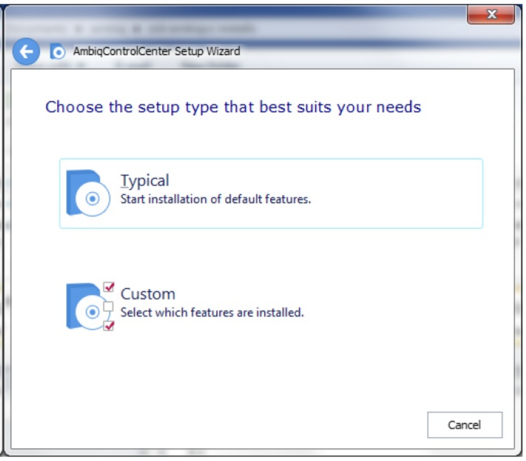


AmbiqControlCenter Setup Wizard

Select installation folder

Folder: C:\AmbiqMicro\

Total space required on drive:	201 MB
Space available on drive:	354 GB
Remaining free space on drive:	354 GB



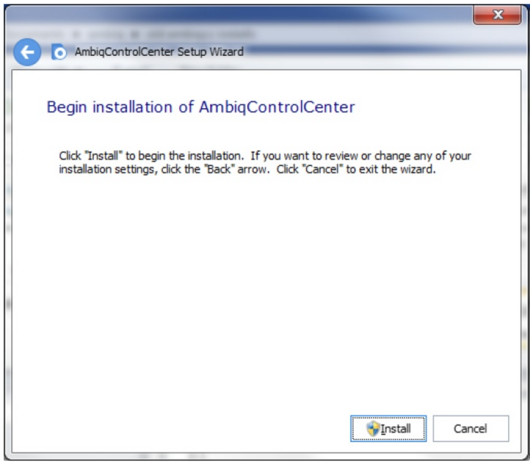
AmbiqControlCenter Setup Wizard

Choose the setup type that best suits your needs

Typical
Start installation of default features.

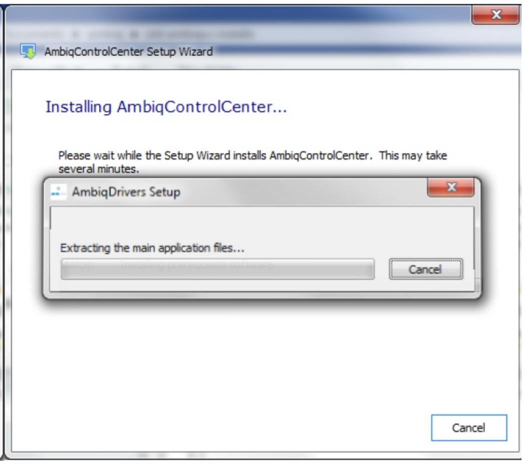
Custom
Select which features are installed.

6. Next, click **Install**, and wait for the installation process to continue.


AmbiqControlCenter Setup Wizard

Begin installation of AmbiqControlCenter

Click "Install" to begin the installation. If you want to review or change any of your installation settings, click the "Back" arrow. Click "Cancel" to exit the wizard.


AmbiqControlCenter Setup Wizard

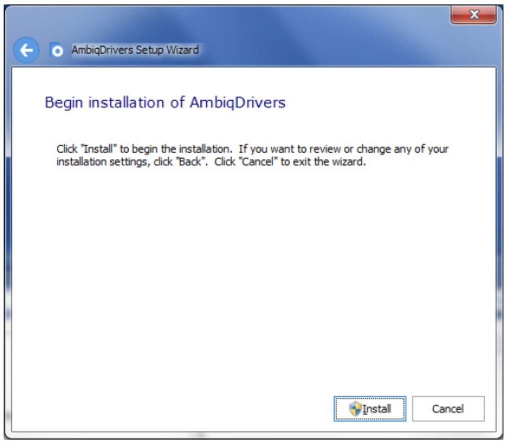
Installing AmbiqControlCenter...

Please wait while the Setup Wizard installs AmbiqControlCenter. This may take several minutes.

AmbiqDrivers Setup

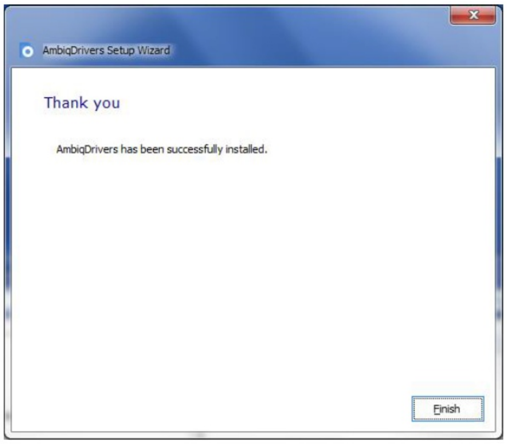
Extracting the main application files...

A sub-installer for the drivers for the FTDI debugger interface on the EVK board will be started from the main installer at this point, displaying the following panels:


AmbiqDrivers Setup Wizard

Begin installation of AmbiqDrivers

Click "Install" to begin the installation. If you want to review or change any of your installation settings, click "Back". Click "Cancel" to exit the wizard.

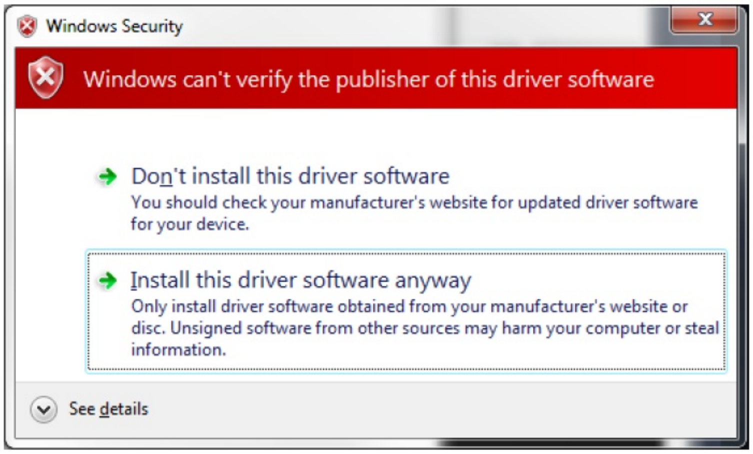

AmbiqDrivers Setup Wizard

Thank you

AmbiqDrivers has been successfully installed.

This action installs two sets of drivers. The first is the standard FTDI drivers for the AmbiqMicro instance of the FT4232H or FT2232H device used as the debugger interface. The second set of drivers is the libusbK driver which is used by the openOCD daemon to control the serial wire debugger (SWD) interface pins on the Apollo SoC.

Note that during the driver installation, particularly if this is the first time that the FTDI drivers have been installed on the target Windows system, a red dialogue may pop up warning that the driver being installed cannot be verified. In order to continue driver installation, click **Install this driver software anyway**.



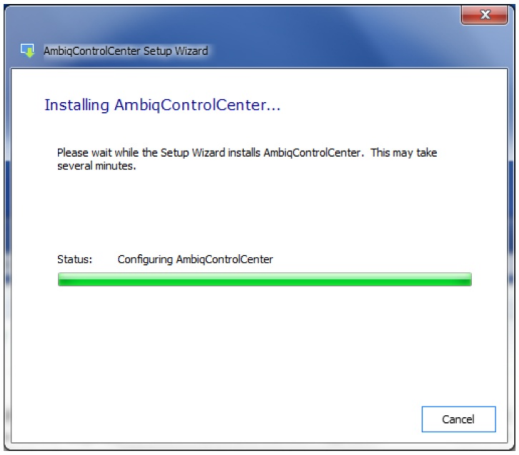
Windows can't verify the publisher of this driver software

Don't install this driver software
You should check your manufacturer's website for updated driver software for your device.

Install this driver software anyway
Only install driver software obtained from your manufacturer's website or disc. Unsigned software from other sources may harm your computer or steal information.

Once the driver installation is finished, the main Ambiq Control Center installer will continue to its completion.

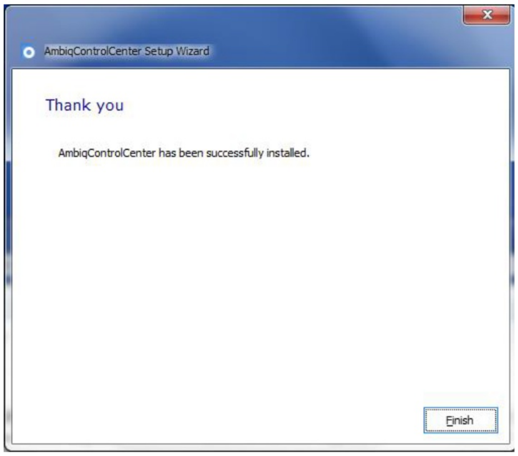
7. Click **Finish**.



Installing AmbiqControlCenter...

Please wait while the Setup Wizard installs AmbiqControlCenter. This may take several minutes.

Status: Configuring AmbiqControlCenter



Thank you

AmbiqControlCenter has been successfully installed.

At this point, the Ambiq Control Center and FTDI drivers should be fully installed at the selected location (C:\AmbiqMicro by default).

2.2 Apollo EVB Board Stack and Integrated Debugger Interface Overview

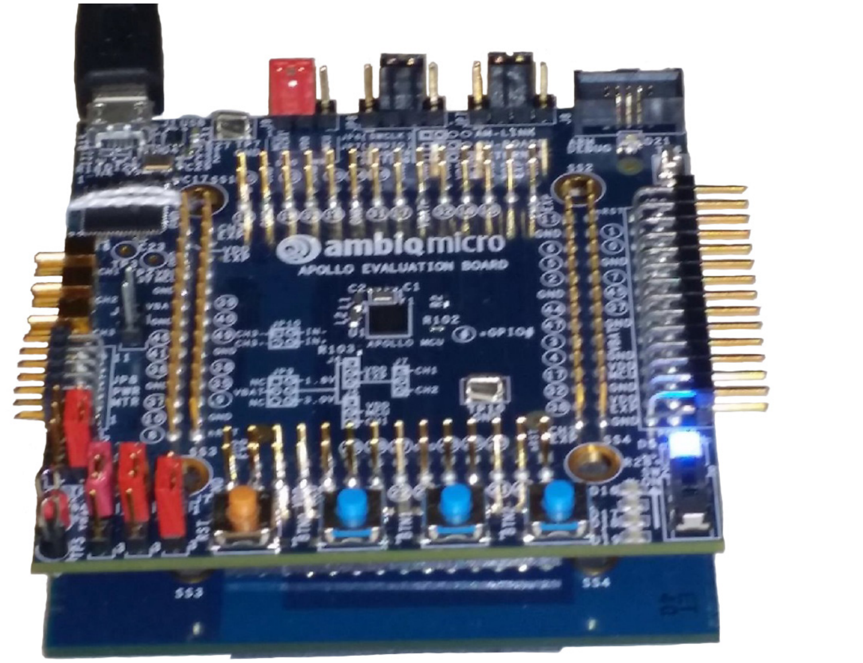
While the tool chains and examples depicted in this document can be used with any board with a suitable debugger connection and peripheral chips, the focused discussions herein are the Apollo EVK board as shown in Figure 2-1. The EVK consists of a two board stack. The top board shown in the figure is the EVK base board which has the Apollo SoC mounted dead center on the board. The lower board, partially visible in the figure, is a sensor board that gives this EVK stack the “personality” of a sensor platform suitable for modeling a wearable device or a sensor hub.

As seen in the figure, the Apollo EVK base board has two methods of connecting to the debugger in one the supported tool chain IDEs to the EVK:

1. Standard 10-pin ARM ULINK2 style connector.
2. USB connection facilitated by an FTDI FT4232H chip on the Apollo EVK base board.

Figure 2-1 shows the USB connection in use and the jumpers set appropriately to support the FTDI based connection. One should refer to the Apollo EVK documentation for more details on jumper settings.

Figure 2-1: Apollo EVK Board



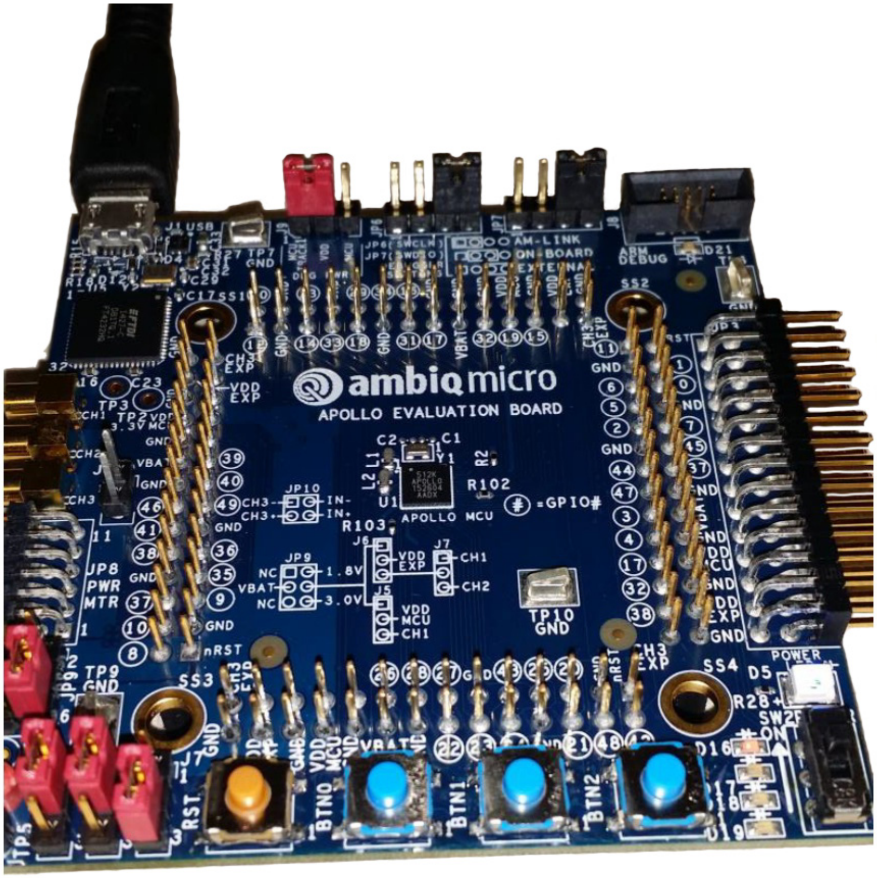
The Apollo EVK base board is supported by the openOCD debug daemon which controls the SWD pins via FT4232H channel A. The openOCD daemon presents a gdb connection to a debugger in the Eclipse IDE and the IAR IDE. Thus Eclipse and IAR eWARM use this mechanism to connect their IDE/debugger applications to the Apollo SoC. Ambiq Micro provides an AGDI driver for the Keil MDK IDE connection to the Apollo SoC's SWD pins. The Apollo SoC SWO pin is connected to FT4232H channel C and presents as a Windows Com Port. These details are discussed further in the sections below.

The AMFLASH Utility, discussed in *Section 2.4 Using the Ambiq Control Center on page 12*, uses the openOCD daemon to control the Apollo EVK base board. It allows one to download and run precompiled binaries of the example programs right out of the box.

2.3 Using the CoreSight 10-Pin Debugger Socket

One can use the ARM 10-pin CMSIS connector to debug embedded firmware on the Apollo SoC by using a Keil ULINK2 USB probe with the Keil MDK IDE or an IAR iJET USB pod with the IAR EWARM IDE. Change the EVK jumpers to the configuration shown in Figure 2-2

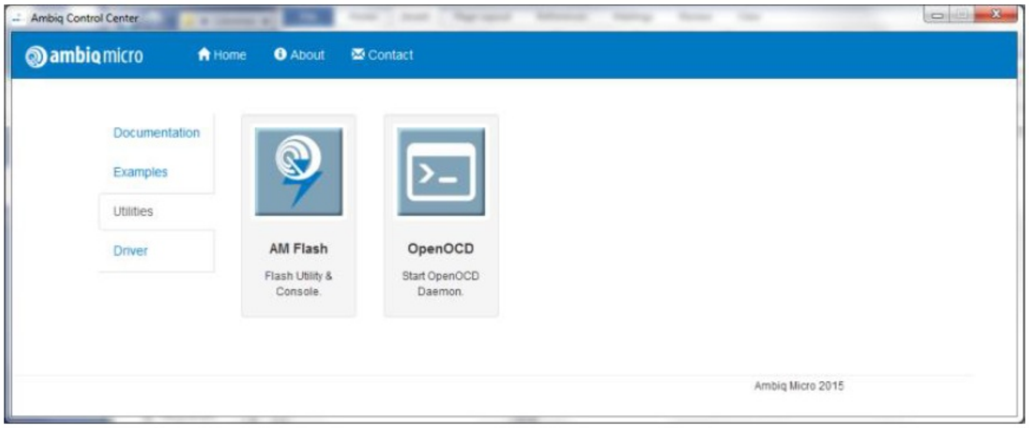
Figure 2-2: Setting EVK Jumpers for Use with ULINK2 or iJET



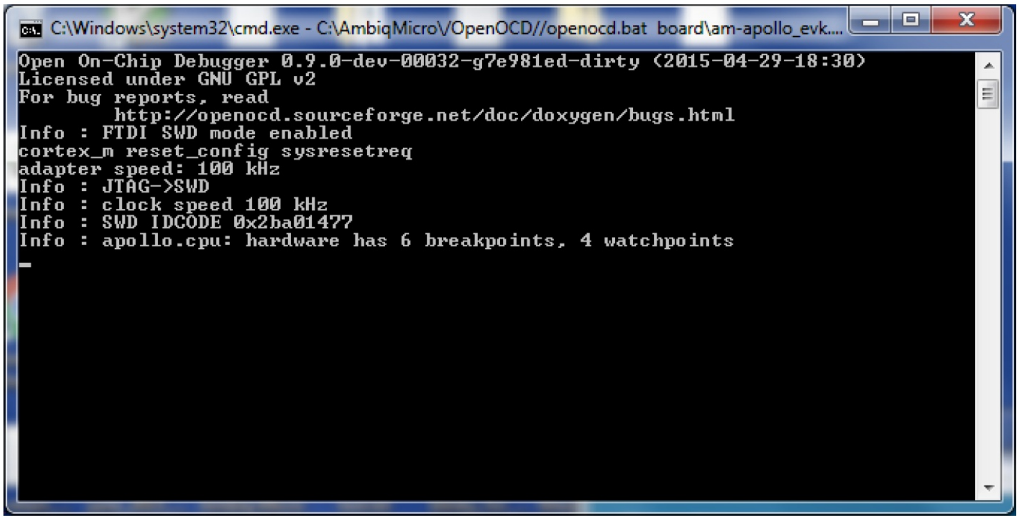
2.4 Using the Ambiq Control Center

Everything that one needs to experience the Apollo SoC and to run the provided example programs can be accomplished with the Ambiq Control Center and the AMFLASH utility without downloading or installing any of the 3 tool chains.

1. Once the Ambiq Control Center is installed, go to the utilities menu and select the **AMFLASH** utility.
2. Check that the drivers installed correctly by launching openOCD from within the Ambiq control Center.



The windows command shell pop up:

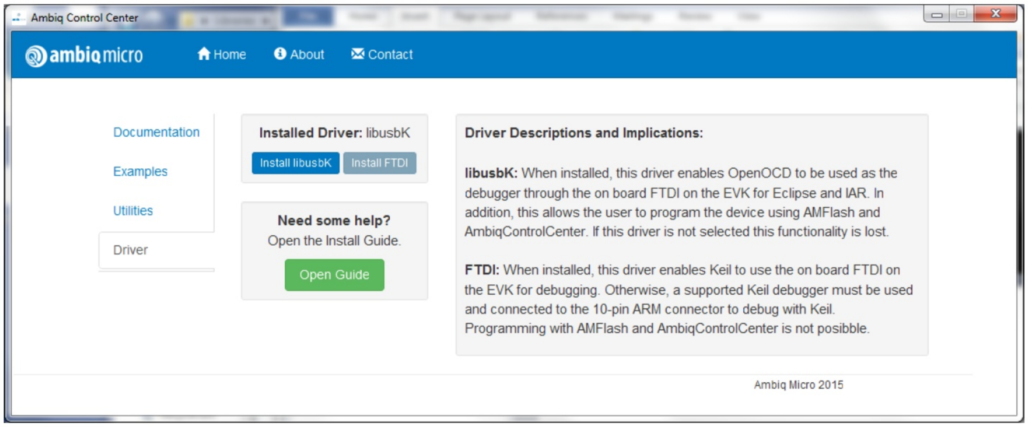

C:\Windows\system32\cmd.exe - C:\AmbiqMicro\/OpenOCD//openocd.bat board\am-apollo_evk....

```
Open On-Chip Debugger 0.9.0-dev-00032-g7e981ed-dirty (2015-04-29-18:30)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.sourceforge.net/doc/doxygen/bugs.html
Info : FTDI SWD mode enabled
cortex_m reset_config sysresetreq
adapter speed: 100 kHz
Info : JTAG->SWD
Info : clock speed 100 kHz
Info : SWD IDCODE 0x2ba01477
Info : apollo.cpu: hardware has 6 breakpoints, 4 watchpoints
_
```

3. Make sure that it says 6 breakpoints and 4 watch points.

   If these numbers are shown, then the drivers are functioning correctly for debugger access to the Apollo SoC over the FTDI chip. If the OpenOCD command shell looks correct, skip the next paragraph to continue with using the AMFlash utility.

If this does not succeed, first try a power cycle of the Apollo EVK and restart OpenOCD. If that does not resolve it, try retargeting the drivers using the driver control panel in the Ambiq Control Center. Try using the **Install libusbk** radio button. This will reinstall the drivers needed by openOCD and AMFlash.



Installed Driver: libusbK

Need some help?
Open the Install Guide.

Driver Descriptions and Implications:

libusbK: When installed, this driver enables OpenOCD to be used as the debugger through the on board FTDI on the EVK for Eclipse and IAR. In addition, this allows the user to program the device using AMFlash and AmbiqControlCenter. If this driver is not selected this functionality is lost.

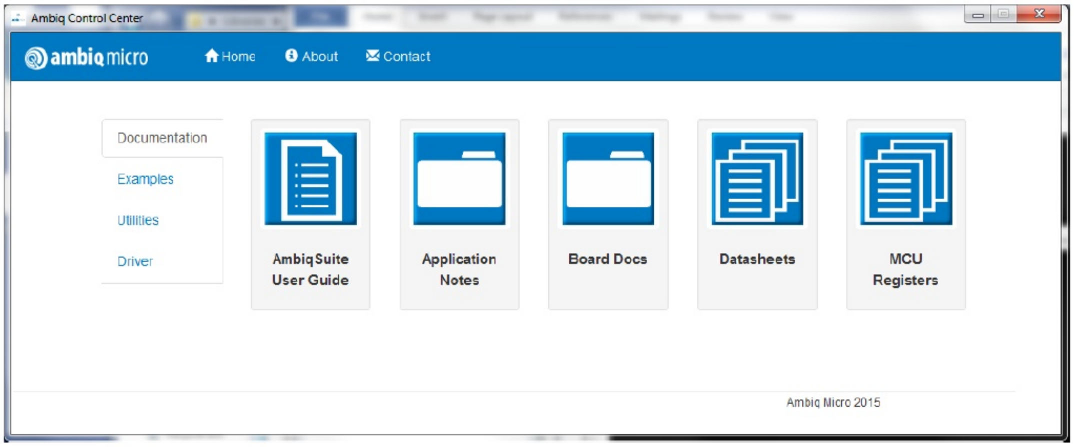
FTDI: When installed, this driver enables Keil to use the on board FTDI on the EVK for debugging. Otherwise, a supported Keil debugger must be used and connected to the 10-pin ARM connector to debug with Keil. Programming with AMFlash and AmbiqControlCenter is not posibble.

Once OpenOCD is working as expected, AMFlash will be able to operate properly. Since AMFlash quietly starts the openOCD daemon as needed, the openOCD command shell will need to be closed, which will kill openOCD. AMFlash may not function properly if openOCD is already running. However, start openOCD in a command shell if working with Eclipse (see also *Section 3.2.2 Starting OpenOCD on page 28*).

2.4.1 Ambiq Control Center Documentation

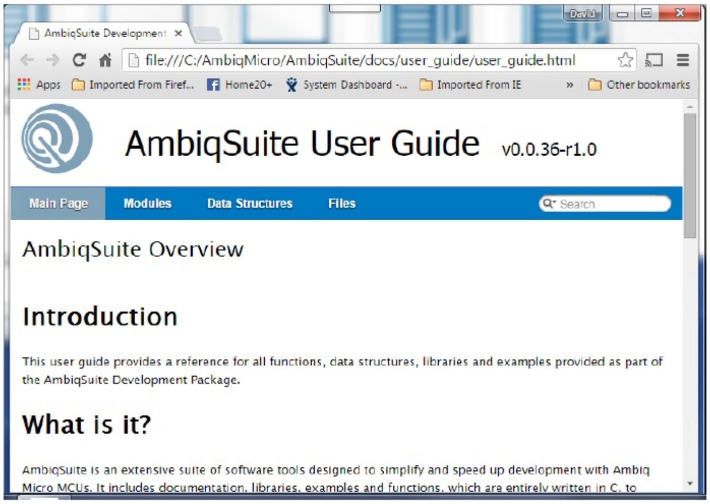
The Ambiq Control Center gives access to lots of installed documentation using the **Documentation** option, as follows:

Figure 2-3: Ambiq Control Center Documentation Options



The AmbiqSuite User Guide contains the doxygen extracted documentation for the APIs.

Figure 2-4: AmbiqSuite User Guide



AmbiqSuite User Guide v0.0.36-r1.0

AmbiqSuite Overview

Introduction

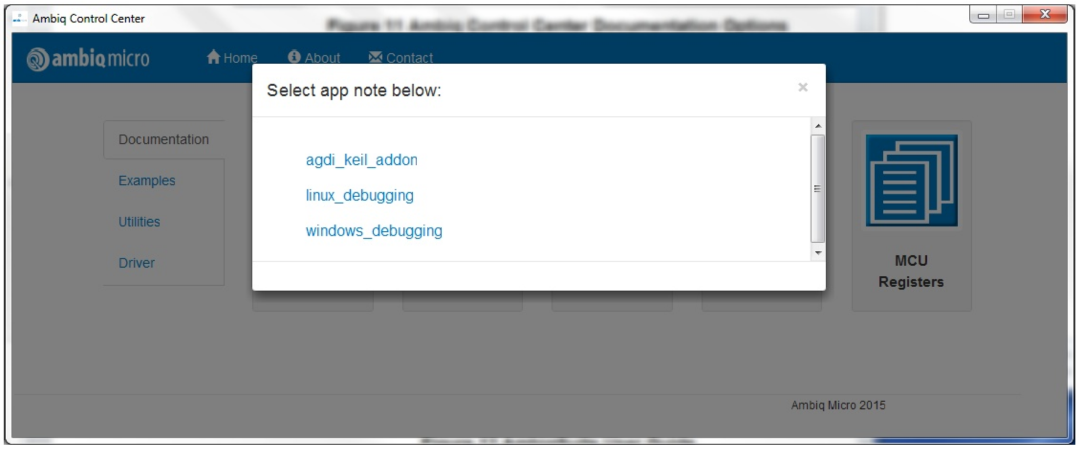
This user guide provides a reference for all functions, data structures, libraries and examples provided as part of the AmbiqSuite Development Package.

What is it?

AmbiqSuite is an extensive suite of software tools designed to simplify and speed up development with Ambiq Micro MCUs. It includes documentation, libraries, examples and functions, which are entirely written in C, to

The Ambiq Control Center also provides quick access to several application notes, including this Windows debugging application note (Quick Start Guide). It contains a similar app note explaining how to setup and then use a Linux platform for debugging the Apollo SoC. It also contains an app note explaining how to activate the AGDI driver for Keil that enables Keil debugging using the FTDI chip on the Apollo EVK Base board with needing a separate ULINK2 USB debug probe.

Figure 2-5: Ambiq Control Center Application Notes



Select app note below:

agdi_keil_addon

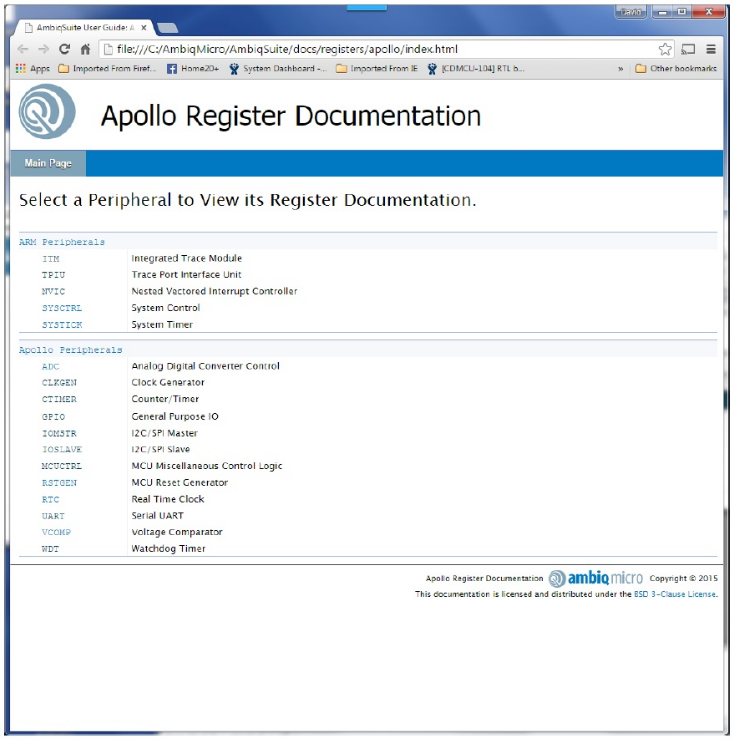
linux_debugging

windows_debugging

The board documentation accessible from within the Ambiq Control Center includes the user's guide for the Apollo EVK Base board as well as schematics for all of the boards in the Apollo EVK stack. The current Data Sheet for the Apollo SoC can also be easily accessed from within the Ambiq Control Center.

Finally, detailed Apollo programmable register documentation can be easily accessed from within the control center by clicking on the SoC documentation button. The browser will be automatically loaded with an index page that lists every peripheral block in the SoC as well Cortex system registers, see Figure 2-6.

Figure 2-6: Register Documentation Block Selection



Apollo Register Documentation

Select a Peripheral to View its Register Documentation.

ARM Peripherals

ITM	Integrated Trace Module
TPIU	Trace Port Interface Unit
NVIC	Nested Vectored Interrupt Controller
SYSCTRL	System Control
SYSTICK	System Timer

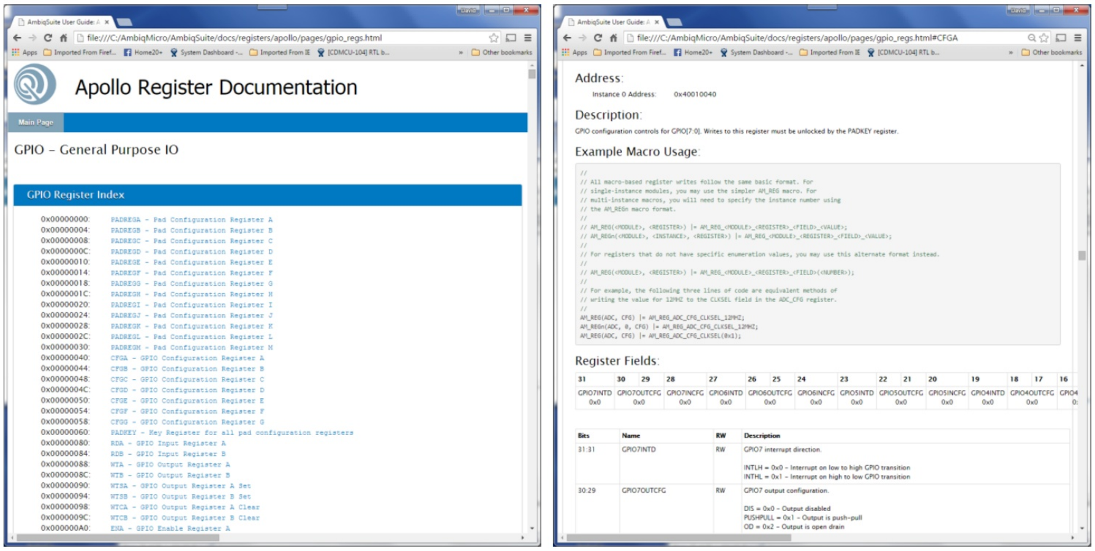
Apollo Peripherals

ADC	Analog Digital Converter Control
CLKGEN	Clock Generator
CTIMER	Counter/Timer
GPIO	General Purpose IO
IOMSTR	I2C/SPI Master
IOSLAVE	I2C/SPI Slave
MCUCTRL	MCU Miscellaneous Control Logic
RSTGEN	MCU Reset Generator
RTC	Real Time Clock
UART	Serial UART
VCOMP	Voltage Comparator
WDT	Watchdog Timer

Apollo Register Documentation ambiqmicro Copyright © 2015
This documentation is licensed and distributed under the BSD 3-Clause License.

Selection of the GPIO block for example will give a page listing all of the registers in the GPIO block. Clicking on one of the register names will open a page detailing the definition of the bit fields in that register. Selecting the GPIO CFG A register for example will provide a page like the one shown in Figure 2-7.

Figure 2-7: GPIO Register Selection


AmbiqSuite User Guide: A ×
file:///C:/AmbiqMicro/AmbiqSuite/docs/registers/apollo/pages/gpio_regs.html
Apps
Imported From Firef...
Home20+
System Dashboard -...
Imported From IE
[CDMCU-104] RTL b...
Other bookmarks
Main Page
AmbiqSuite User Guide: A ×
file:///C:/AmbiqMicro/AmbiqSuite/docs/registers/apollo/pages/gpio_regs.html#CFGA

Apollo Register Documentation

GPIO – General Purpose IO

GPIO Register Index

0x00000000:	PADREGA - Pad Configuration Register A
0x00000004:	PADREGB - Pad Configuration Register B
0x00000008:	PADREGC - Pad Configuration Register C
0x0000000C:	PADREGD - Pad Configuration Register D
0x00000010:	PADREGE - Pad Configuration Register E
0x00000014:	PADREGF - Pad Configuration Register F
0x00000018:	PADREGG - Pad Configuration Register G
0x0000001C:	PADREGH - Pad Configuration Register H
0x00000020:	PADREGI - Pad Configuration Register I
0x00000024:	PADREGJ - Pad Configuration Register J
0x00000028:	PADREGK - Pad Configuration Register K
0x0000002C:	PADREGL - Pad Configuration Register L
0x00000030:	PADREGM - Pad Configuration Register M
0x00000040:	CFGA - GPIO Configuration Register A
0x00000044:	CFGB - GPIO Configuration Register B
0x00000048:	CFGC - GPIO Configuration Register C
0x0000004C:	CFGD - GPIO Configuration Register D
0x00000050:	CFGE - GPIO Configuration Register E
0x00000054:	CFGF - GPIO Configuration Register F
0x00000058:	CFGG - GPIO Configuration Register G
0x00000060:	PADKEY - Key Register for all pad configuration registers
0x00000080:	RDA - GPIO Input Register A
0x00000084:	RDB - GPIO Input Register B
0x00000088:	WTA - GPIO Output Register A
0x0000008C:	WTB - GPIO Output Register B
0x00000090:	WTSA - GPIO Output Register A Set
0x00000094:	WTSB - GPIO Output Register B Set
0x00000098:	WTCA - GPIO Output Register A Clear
0x0000009C:	WTCB - GPIO Output Register B Clear
0x000000A0:	ENA - GPIO Enable Register A

Address:

Instance 0 Address: 0x40010040

Description:

GPIO configuration controls for GPIO[7:0]. Writes to this register must be unlocked by the PADKEY register.

Example Macro Usage:

```
//
// All macro-based register writes follow the same basic format. For
// single-instance modules, you may use the simpler AM_REG macro. For
// multi-instance macros, you will need to specify the instance number using
// the AM_REGn macro format.
//
// AM_REG(<MODULE>, <REGISTER>) |= AM_REG_<MODULE>_<REGISTER>_<FIELD>_<VALUE>;
// AM_REGn(<MODULE>, <INSTANCE>, <REGISTER>) |= AM_REG_<MODULE>_<REGISTER>_<FIELD>_<VALUE>;
//
// For registers that do not have specific enumeration values, you may use this alternate format instead.
//
// AM_REG(<MODULE>, <REGISTER>) |= AM_REG_<MODULE>_<REGISTER>_<FIELD>(<NUMBER>);
//
// For example, the following three lines of code are equivalent methods of
// writing the value for 12MHZ to the CLKSEL field in the ADC_CFG register.
//
AM_REG(ADC, CFG) |= AM_REG_ADC_CFG_CLKSEL_12MHZ;
AM_REGn(ADC, 0, CFG) |= AM_REG_ADC_CFG_CLKSEL_12MHZ;
AM_REG(ADC, CFG) |= AM_REG_ADC_CFG_CLKSEL(0x1);
```

Register Fields:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO7INTD 0x0	GPIO7OUTCFG 0x0		GPIO7INCFG 0x0	GPIO6INTD 0x0	GPIO6OUTCFG 0x0		GPIO6INCFG 0x0	GPIO5INTD 0x0	GPIO5OUTCFG 0x0		GPIO5INCFG 0x0	GPIO4INTD 0x0	GPIO4OUTCFG 0x0		GPIO4 0:

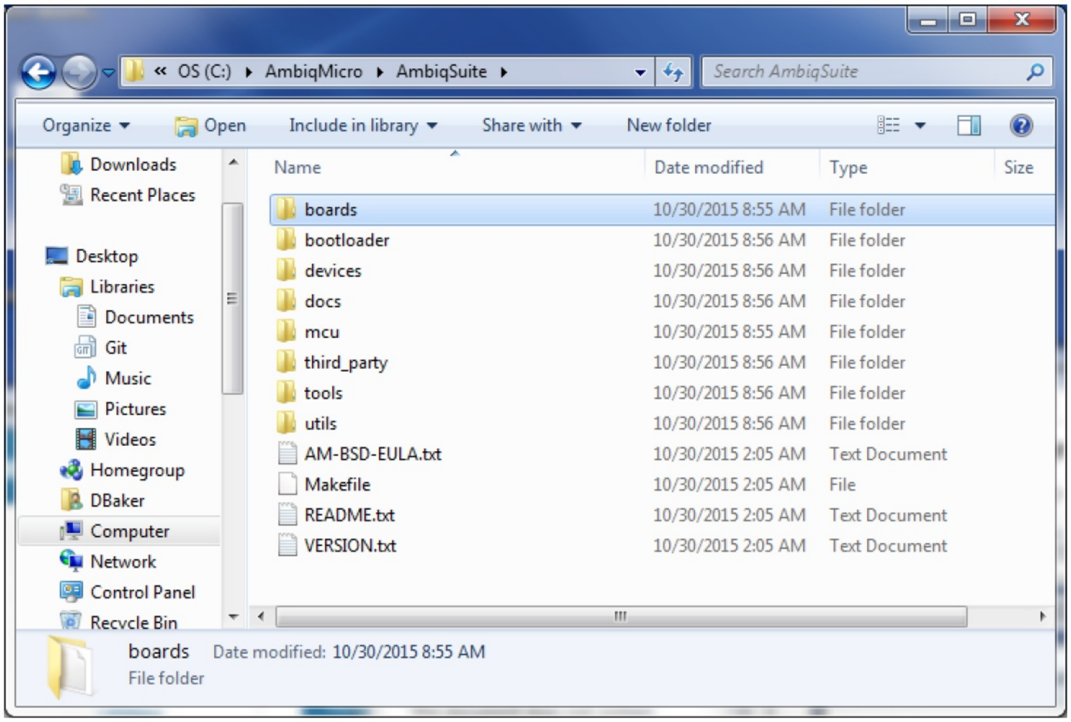
Bits	Name	RW	Description
31:31	GPIO7INTD	RW	GPIO7 interrupt direction. INTLH = 0x0 - Interrupt on low to high GPIO transition INTHL = 0x1 - Interrupt on high to low GPIO transition

30:29	GPIO7OUTCFG	RW	GPIO7 output configuration.
			
			DIS = 0x0 - Output disabled
			PUSHPULL = 0x1 - Output is push-pull
			OD = 0x2 - Output is open drain

2.5 AmbiqSuite Content Structure Overview

The AmbiqSuite ships with 8 directories at its top level as shown in Figure 2-8.

Figure 2-8: Top Level Directory of AmbiqSuite



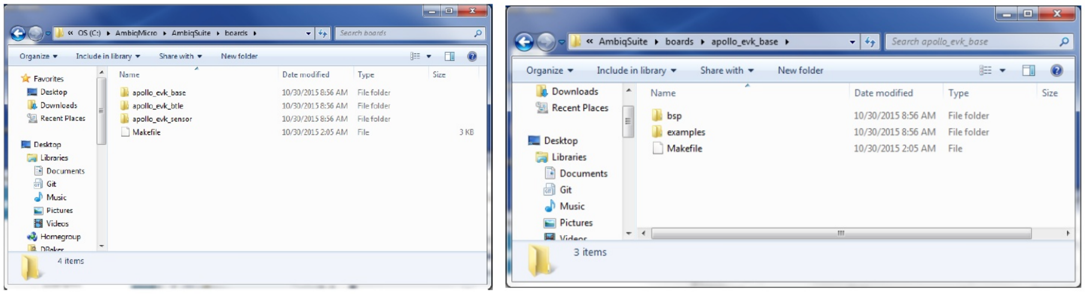
« OS (C:) ▸ AmbiqMicro ▸ AmbiqSuite ▸

Name	Date modified	Type	Size
boards	10/30/2015 8:55 AM	File folder	
bootloader	10/30/2015 8:56 AM	File folder	
devices	10/30/2015 8:56 AM	File folder	
docs	10/30/2015 8:56 AM	File folder	
mcu	10/30/2015 8:55 AM	File folder	
third_party	10/30/2015 8:56 AM	File folder	
tools	10/30/2015 8:56 AM	File folder	
utils	10/30/2015 8:56 AM	File folder	
AM-BSD-EULA.txt	10/30/2015 2:05 AM	Text Document	
Makefile	10/30/2015 2:05 AM	File	
README.txt	10/30/2015 2:05 AM	Text Document	
VERSION.txt	10/30/2015 2:05 AM	Text Document	

boards Date modified: 10/30/2015 8:55 AM
File folder

Selecting the boards directory shows us that users can navigate to directories supporting any of the 3 board configurations: EVK base board stand alone, EVK base board plus EVK sensor board or a 3 board combination consisting of the EVK base, EVK sensor and the EVK Bluetooth Low Energy radio board. For this discussion, see Figure 2-9 for EVK base board directory.

Figure 2-9: Diving into a Board Directory, Such as the one for the Apollo EVB Base



« OS (C:) ▸ AmbiqMicro ▸ AmbiqSuite ▸ boards ▸

Name	Date modified	Type	Size
apollo_evk_base	10/30/2015 8:56 AM	File folder	
apollo_evk_btle	10/30/2015 8:56 AM	File folder	
apollo_evk_sensor	10/30/2015 8:56 AM	File folder	
Makefile	10/30/2015 2:05 AM	File	3 KB

4 items

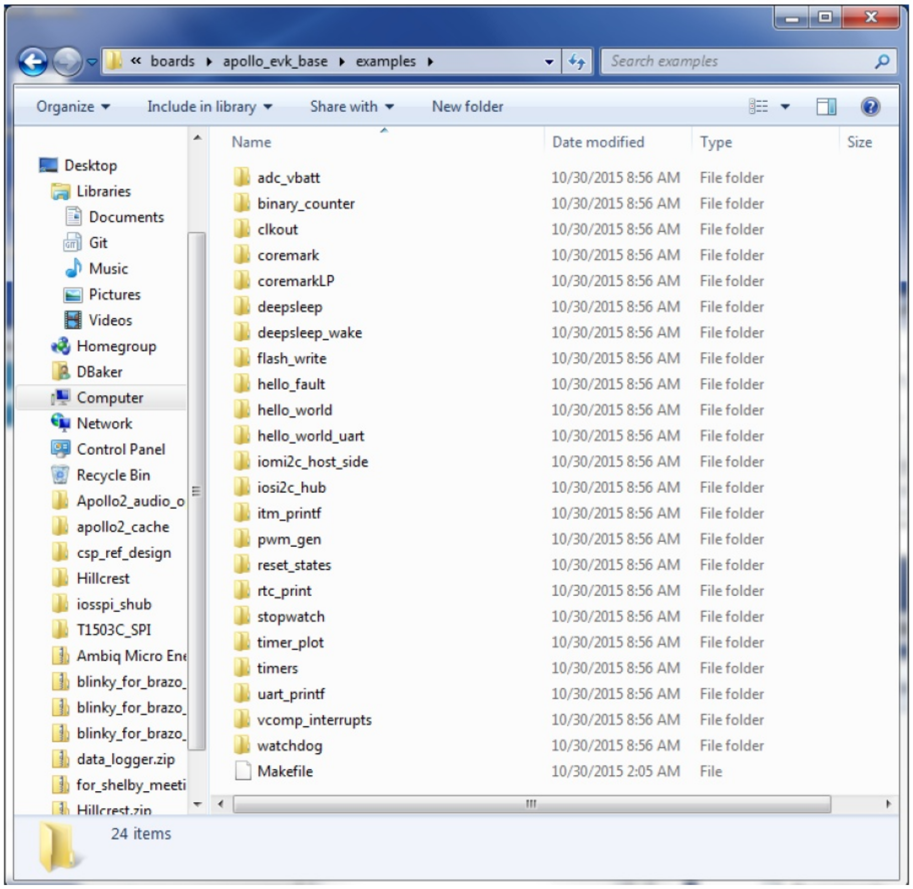
« AmbiqSuite ▸ boards ▸ apollo_evk_base ▸

Name	Date modified	Type	Size
bsp	10/30/2015 8:56 AM	File folder	
examples	10/30/2015 8:56 AM	File folder	
Makefile	10/30/2015 2:05 AM	File	

3 items

The Board Support Package (BSP) directory contains code that is specific to the target board configuration. It contains files that assign uses to pins on the Apollo SoC and assign devices to the appropriate I/O master. The contents of the BSP directory are different for every board configuration supported. Finally, if the user dive into the examples directory for the EVK base board, they will see a long list of examples provided with the AmbiqSuite. None of the examples in this directory requires sensors or Bluetooth.

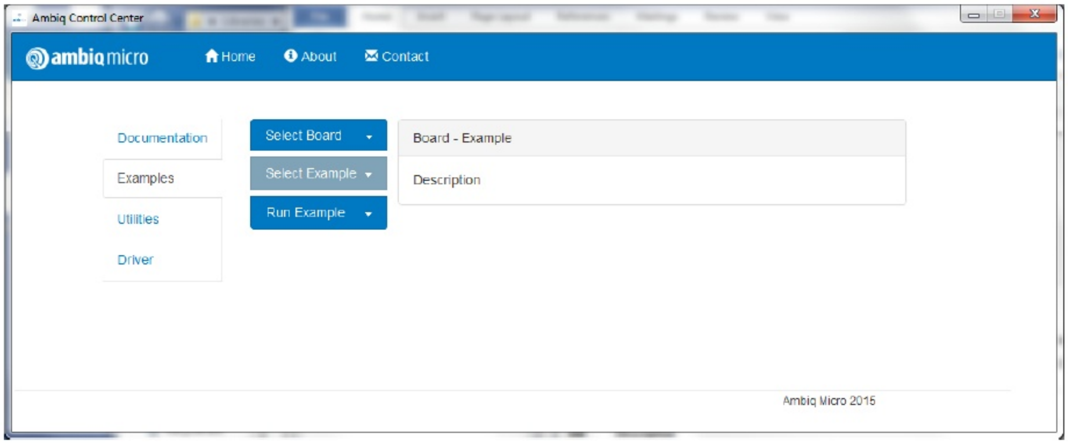
Figure 2-10: Contents of the EVK Base Board Examples Directory



Name	Date modified	Type	Size
adc_vbatt	10/30/2015 8:56 AM	File folder	
binary_counter	10/30/2015 8:56 AM	File folder	
clkout	10/30/2015 8:56 AM	File folder	
coremark	10/30/2015 8:56 AM	File folder	
coremarkLP	10/30/2015 8:56 AM	File folder	
deepsleep	10/30/2015 8:56 AM	File folder	
deepsleep_wake	10/30/2015 8:56 AM	File folder	
flash_write	10/30/2015 8:56 AM	File folder	
hello_fault	10/30/2015 8:56 AM	File folder	
hello_world	10/30/2015 8:56 AM	File folder	
hello_world_uart	10/30/2015 8:56 AM	File folder	
iomi2c_host_side	10/30/2015 8:56 AM	File folder	
iosi2c_hub	10/30/2015 8:56 AM	File folder	
itm_printf	10/30/2015 8:56 AM	File folder	
pwm_gen	10/30/2015 8:56 AM	File folder	
reset_states	10/30/2015 8:56 AM	File folder	
rtc_print	10/30/2015 8:56 AM	File folder	
stopwatch	10/30/2015 8:56 AM	File folder	
timer_plot	10/30/2015 8:56 AM	File folder	
timers	10/30/2015 8:56 AM	File folder	
uart_printf	10/30/2015 8:56 AM	File folder	
vcomp_interrupts	10/30/2015 8:56 AM	File folder	
watchdog	10/30/2015 8:56 AM	File folder	
Makefile	10/30/2015 2:05 AM	File	

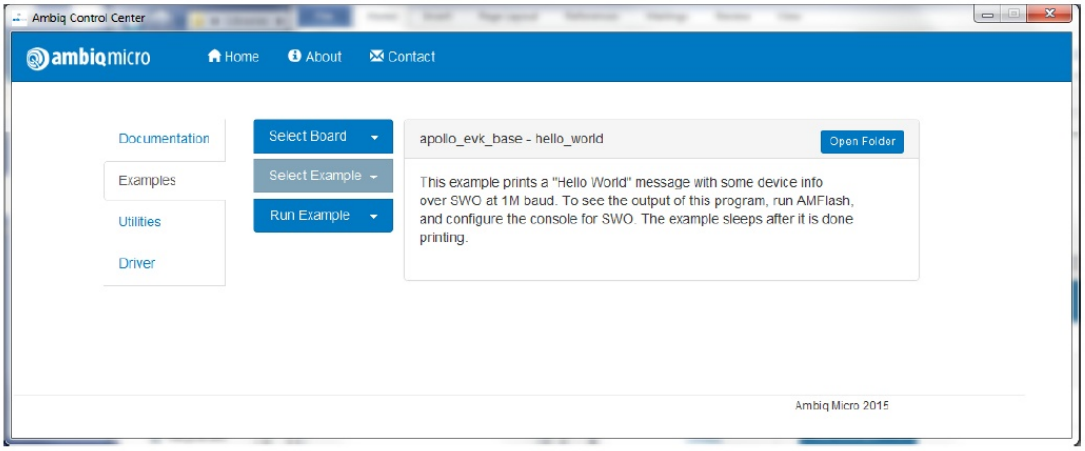
The AmbiqSuite ships with precompiled binaries for all of its examples. All of these binaries can run from directly within the Ambiq Control Center by selecting **Example** button from the main panel see Figure 2-11.

Figure 2-11: Example Selection Dialog



If the apollo_evk_base board and the hello_world example is selected, then the panel in Figure 2-12 is shown

Figure 2-12: Example Selection



apollo_evk_base - hello_world

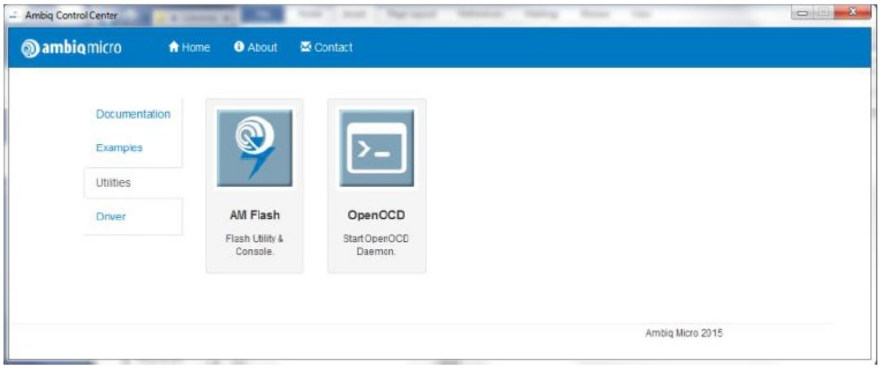
This example prints a "Hello World" message with some device info over SWO at 1M baud. To see the output of this program, run AMFlash, and configure the console for SWO. The example sleeps after it is done printing.

The example description text is derived from the doxygen markup in the example source file. A button will open the directory containing the selected example (e.g., hello_world in this case). Pressing **Run** causes the precompiled binary for the example to be downloaded onto the EVK and then run using the openocd debugger interface. Note that **Run** is a pull down allowing the selection of a specific binary compiled with the desired one of the three tool chains (gcc, Keil, IAR). This selection option is very convenient when experimenting with the ULP Benchmark example.

2.6 Using the AMFLASH Utility From Within the Ambiq Control Center

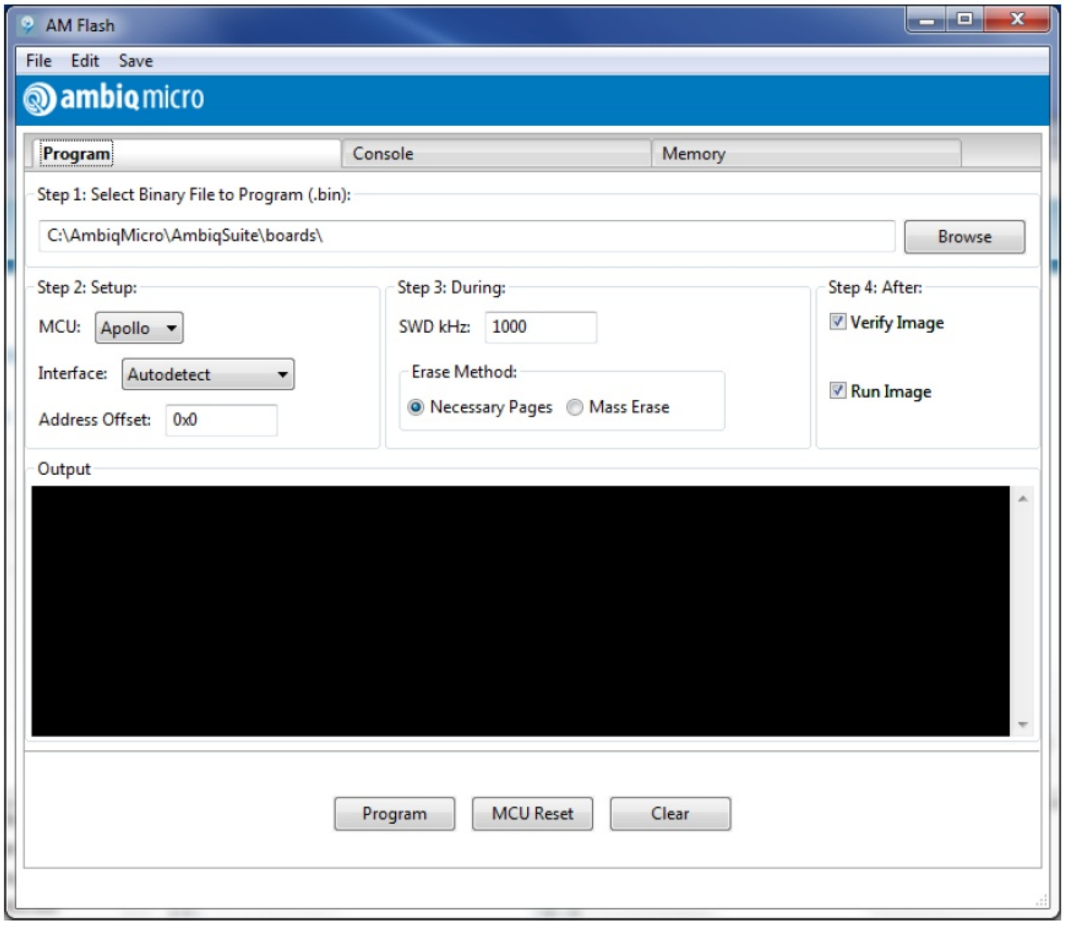
Everything that one needs to experience the Apollo SoC and to run the provided example programs can be accomplished with the Ambiq Control Center and the AMFLASH utility without downloading or installing any of the 3 tool chains. Once the Ambiq Control Center is installed, go to the utilities menu and select the **AMFLASH** utility as shown in Figure 2-13.

Figure 2-13: Launch the AM Flash Utility from within Ambiq Control Center



The AMFlash utility is a standalone program that can also be launched from an icon that was added in the desktop during the installation process. Either way, the startup screen of AMFlash Utility is available.

Figure 2-14: AMFlash Utility



Step 1: Select Binary File to Program (.bin):
C:\AmbiqMicro\AmbiqSuite\boards\

Step 2: Setup:
MCU: Apollo
Interface: Autodetect
Address Offset: 0x0

Step 3: During:
SWD kHz: 1000
Erase Method:
● Necessary Pages ○ Mass Erase

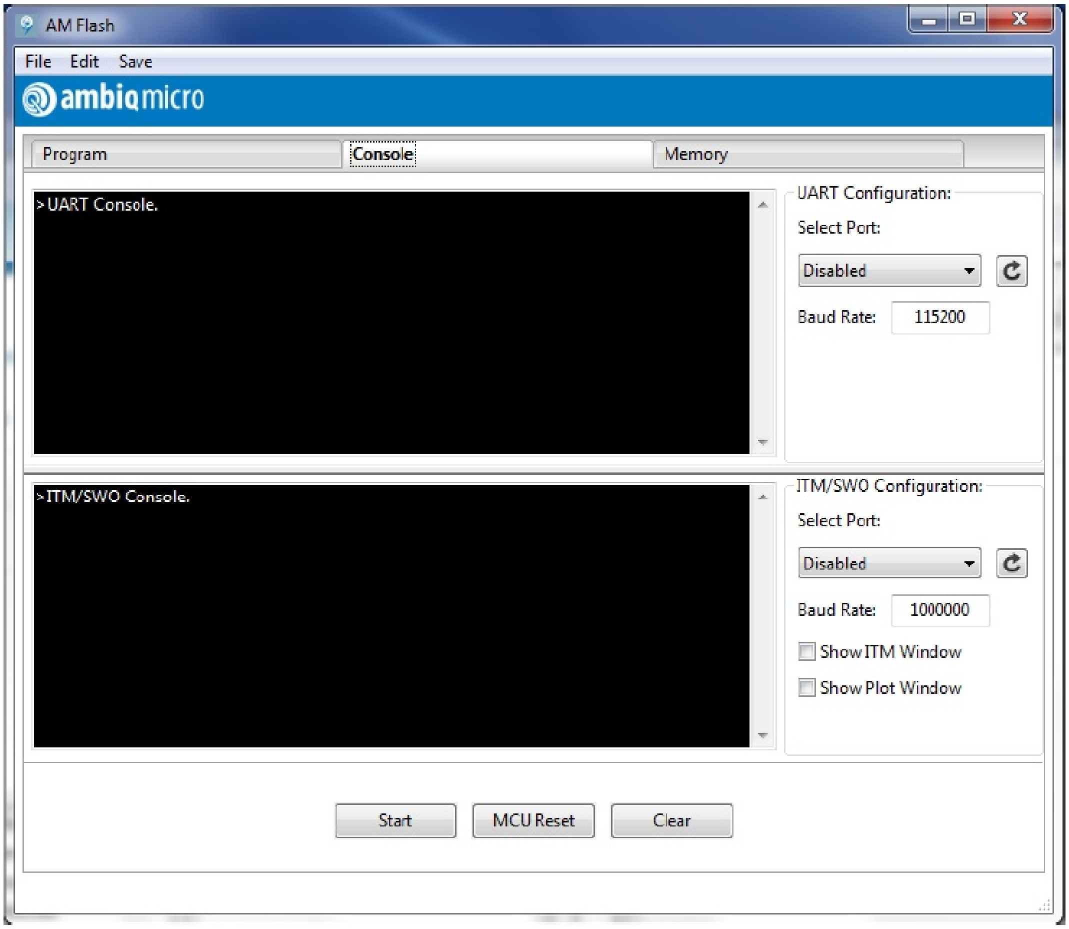
Step 4: After:
☑ Verify Image
☑ Run Image

There are 3 main panels in this utility:

- one for downloading a programming to the Apollo SoC;
- one for displaying console output from things like debug printf statements and
- one for erasing all of the integrated FLASH on the Apollo SoC.

In order to run the hello_world example, the console output must be open, so that must be setup first. Push the button on the Consoletab.

Figure 2-15: AMFlash Utility Console Panel



```
>UART Console.
```

UART Configuration:

Select Port: Disabled

Baud Rate: 115200

```
>ITM/SWO Console.
```

ITM/SWO Configuration:

Select Port: Disabled

Baud Rate: 1000000

□ Show ITM Window

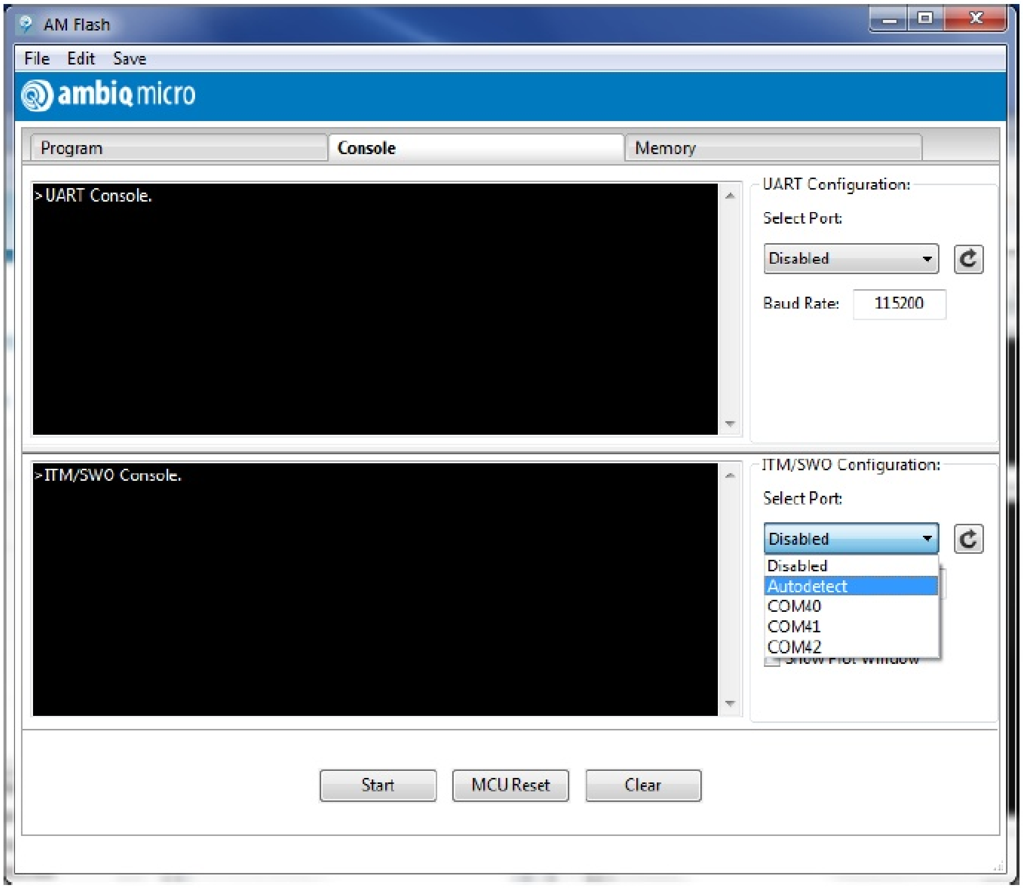
□ Show Plot Window

The console panel gives access to a large number of viewing options, starting with displaying the output from Apollo UART, if it is configured to use GPIO[35] and GPIO[36]. These pins are hard wired to FTDI FT4232 channel D on the Apollo EVK base board. Usage of this UART is infrequent because the UART is usually dedicated to talking to the Bluetooth radio device. The console will be discussed much later. First, activate the ITM/SWO (serial wire output) display using the ITM/SWO console.

2.6.1 Connecting the Com Port for the SWO Output Pin to AMFL

To use this console the user will need to configure it to use the appropriate COM port to collect the SWO packets from the Apollo SoC as shown in Figure 2-16:

Figure 2-16: Autodetecting the SWO COM Port



```
>UART Console.
```

UART Configuration:
Select Port: Disabled
Baud Rate: 115200

```
>ITM/SWO Console.
```

ITM/SWO Configuration:
Select Port: Disabled

Selecting the **autodetect** option and clicking **Start** causes AMFlash to search the available COM ports for the one connected to the SWO pin. When it finds it, it makes it the selected port and starts the ITM/SWO console. The Ambiq print utilities use ITM stimulus register 0 to implement a printf library. Use this console to see the hello_world example's output.

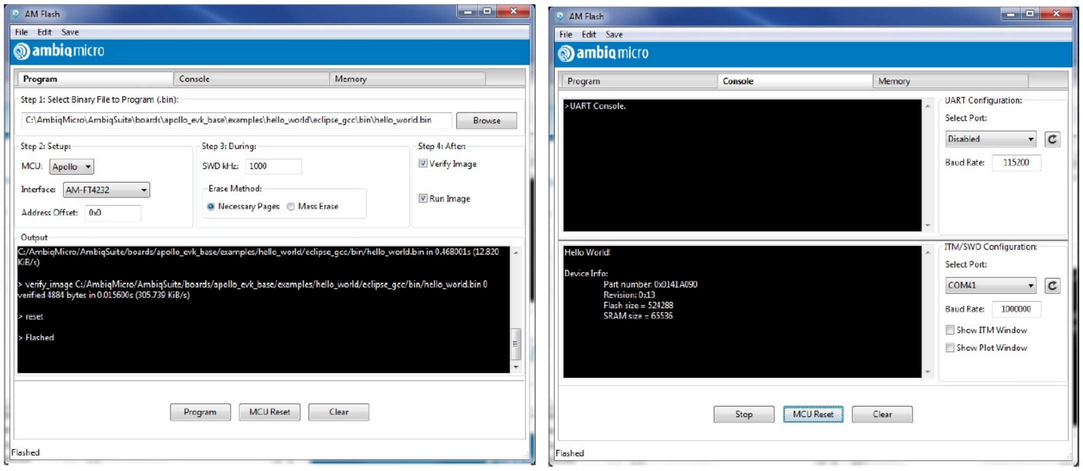
2.6.2 Downloading and Running the Hello World Example

Downloading and running a program like the hello world example in the EVK base examples directory is very easy with AMFlash.

1. Go to the Program panel of AMFlash, and browse to the hello world example directory.
2. Browse into the **eclipse_gcc\bin** directory.
3. Click on **hello_world.bin.**
4. At this point, click **Program**, to download the hello world example into the Apollo flash in preparation for running it.

   A flash progress bar will be shown as it downloads. When download finishes, run it is to issue a reset to the Apollo SoC.
5. Click over to the console panel and see the Hello World output. Easily rerun the example by clicking the **MCU Reset**.

Figure 2-17: Downloading and Running the Hello World Example



Step 1: Select Binary File to Program (.bin): C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\hello_world\eclipse_gcc\bin\hello_world.bin

Step 2: Setup: MCU: Apollo; Interface: AM-FT4232; Address Offset: 0x0

Step 3: During: SWD kHz: 1000; Erase Method: ● Necessary Pages ○ Mass Erase

Step 4: After: ☑ Verify Image ☑ Run Image

Output

```
C:/AmbiqMicro/AmbiqSuite/boards/apollo_evk_base/examples/hello_world/eclipse_gcc/bin/hello_world.bin in 0.468001s (12.820
KiB/s)

> verify_image C:/AmbiqMicro/AmbiqSuite/boards/apollo_evk_base/examples/hello_world/eclipse_gcc/bin/hello_world.bin 0
verified 4884 bytes in 0.015600s (305.739 KiB/s)

> reset

> Flashed
```

Flashed

```
>UART Console.
```

UART Configuration: Select Port: Disabled; Baud Rate: 115200

```
Hello World!

Device Info:
        Part number: 0x0141A090
        Revision: 0x13
        Flash size = 524288
        SRAM size = 65536
```

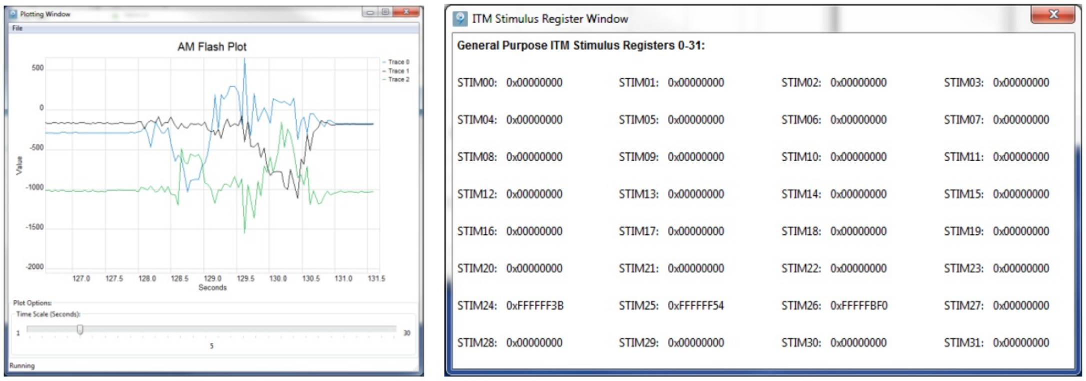
ITM/SWO Configuration: Select Port: COM41; Baud Rate: 1000000; ☐ Show ITM Window ☐ Show Plot Window

Flashed

2.6.3 Show ITM Plot Window and the Show ITM Window Options

AMFlash can extract data samples from values written to ITM registers 24 through 27 and a number of the examples in the am_evk_sensor board package use this facility to plot the output of gyros, accelerometers and magnetometers on that board. Once the ITM port is selected and started for the console tab, one simply clicks on the **Show Plot Window** to get a plot such as this one in Figure 2-18 on page 23. In addition, one can elect to see the most recently written value to each of the ITM stimulus registers as well. If running an example that emits plot samples, the values in stimulus registers 24 through 27 changes as the plot updates.

Figure 2-18: AMFlash Plot Window and ITM Stimulus Register Window



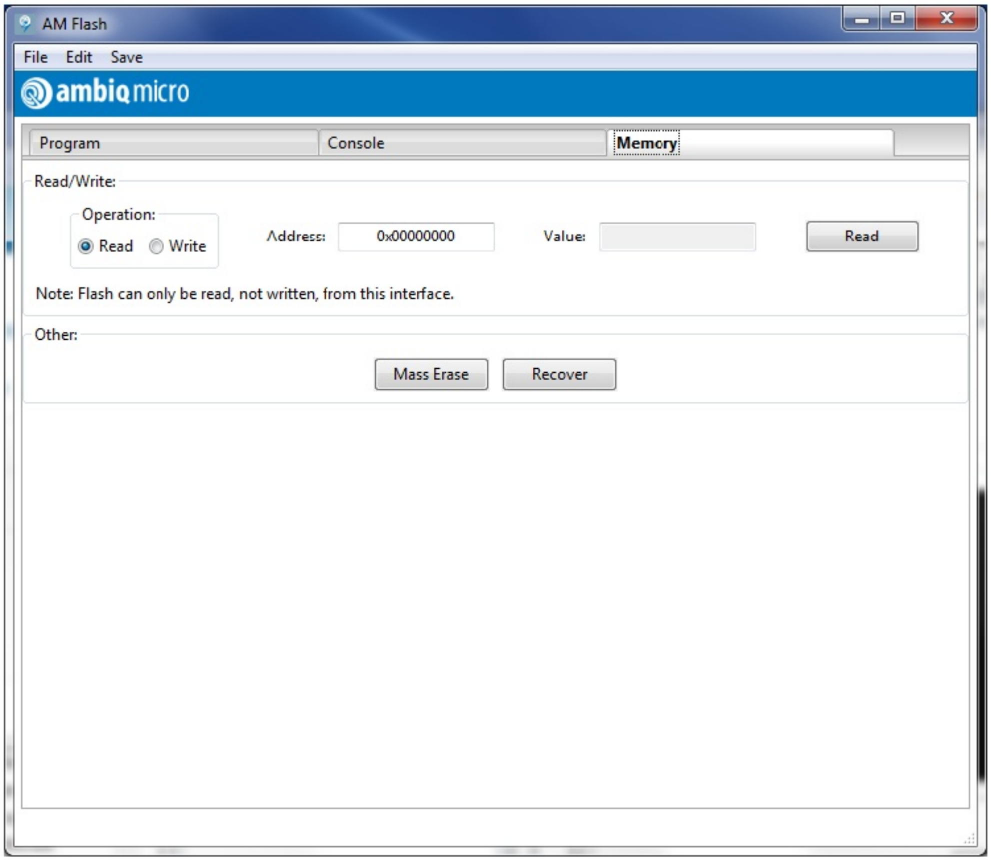
General Purpose ITM Stimulus Registers 0-31:

STIM00: 0x00000000	STIM01: 0x00000000	STIM02: 0x00000000	STIM03: 0x00000000
STIM04: 0x00000000	STIM05: 0x00000000	STIM06: 0x00000000	STIM07: 0x00000000
STIM08: 0x00000000	STIM09: 0x00000000	STIM10: 0x00000000	STIM11: 0x00000000
STIM12: 0x00000000	STIM13: 0x00000000	STIM14: 0x00000000	STIM15: 0x00000000
STIM16: 0x00000000	STIM17: 0x00000000	STIM18: 0x00000000	STIM19: 0x00000000
STIM20: 0x00000000	STIM21: 0x00000000	STIM22: 0x00000000	STIM23: 0x00000000
STIM24: 0xFFFFFF3B	STIM25: 0xFFFFFF54	STIM26: 0xFFFFFBF0	STIM27: 0x00000000
STIM28: 0x00000000	STIM29: 0x00000000	STIM30: 0x00000000	STIM31: 0x00000000

2.6.4 Performing a Mass Erase or Recovery on an Apollo EVK

Sometimes it becomes necessary to completely erase a program to recover convenient debug control of a part. This may be necessary when programs like the watchdog timer example are continuously timing out and resetting the SoC.

Figure 2-19: Mass Erasing the Apollo SoC Flash from AMFlash Utility



Operation: ● Read ○ Write

Address: 0x00000000

Value:

Note: Flash can only be read, not written, from this interface.

SECTION

3

Installing and Using the Eclipse/gcc Tool Chain

The Ambiq Debug Tools installer provides the GNU gcc compiler and Eclipse (Mars version) IDE that can be used for Apollo software development and debug. This portion of the document will explain the process of installing the Ambiq Debug Tools environment.

3.1 Installing the Eclipse/gcc Tool Chain

3.1.1 Prerequisites

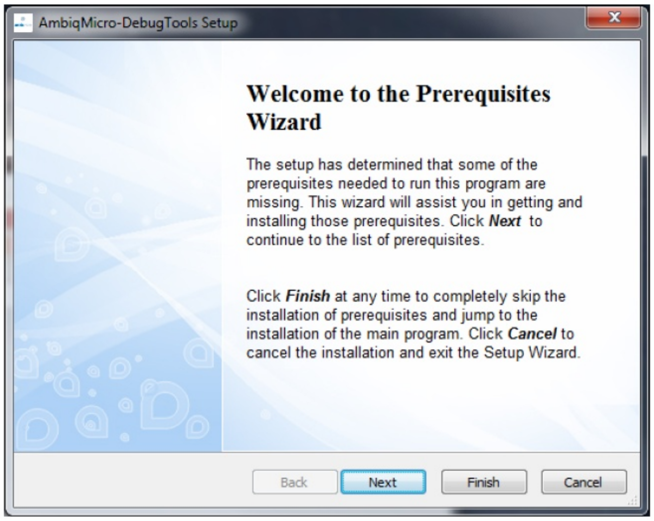
Before installing Ambiq Control Center, make sure the system has a copy of the most recent version of the Java Runtime Environment installed. This is required for the Eclipse IDE, and is available as a free download from the Oracle website.

3.1.2 Installing Eclipse/gcc

After launching the Ambiq Micro DebugTools installer, the following dialogue may be encountered; for example if the correct Java run time environment is not already installed on the target system.

1. Click **Next** to continue with the Prerequisites Wizard, then install whatever recommendations the installer makes.

2. Click **Next** on the Welcome screen.

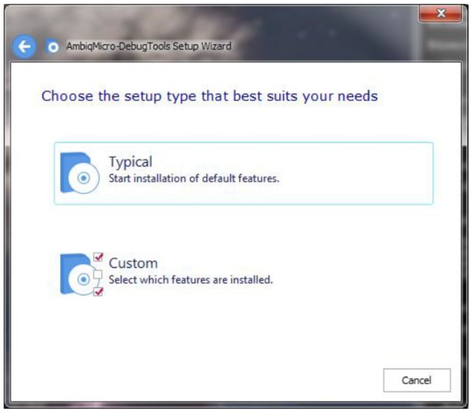


Welcome to the Prerequisites Wizard

The setup has determined that some of the prerequisites needed to run this program are missing. This wizard will assist you in getting and installing those prerequisites. Click *Next* to continue to the list of prerequisites.

Click *Finish* at any time to completely skip the installation of prerequisites and jump to the installation of the main program. Click *Cancel* to cancel the installation and exit the Setup Wizard.

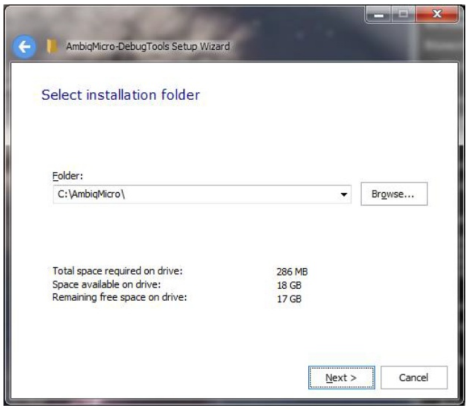
3. Click **Accept** on the License Agreement screen.
4. Select **Typical** to install the Eclipse IDE and GNU tools.
5. Select the default installation folder, and click **Next**.



Choose the setup type that best suits your needs

Typical
Start installation of default features.

Custom
Select which features are installed.

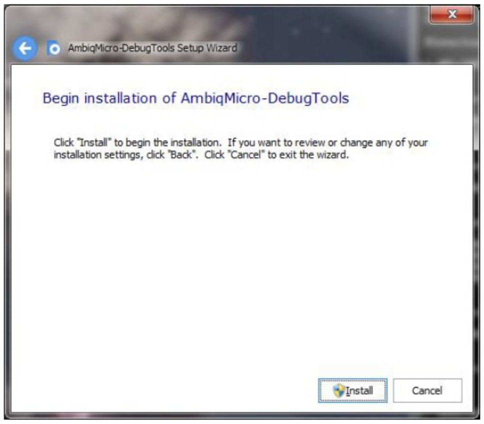


Select installation folder

Folder: C:\AmbiqMicro\

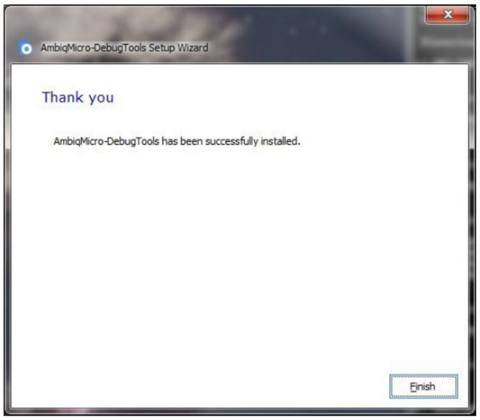
Total space required on drive:	286 MB
Space available on drive:	18 GB
Remaining free space on drive:	17 GB

6. Finally, the Begin Installation panel will appear, click **Install**.
7. Click **Finish** to exit the installer once the installation is complete.



Begin installation of AmbiqMicro-DebugTools

Click "Install" to begin the installation. If you want to review or change any of your installation settings, click "Back". Click "Cancel" to exit the wizard.



Thank you

AmbiqMicro-DebugTools has been successfully installed.

3.1.3 Starting Eclipse and Initial Screens

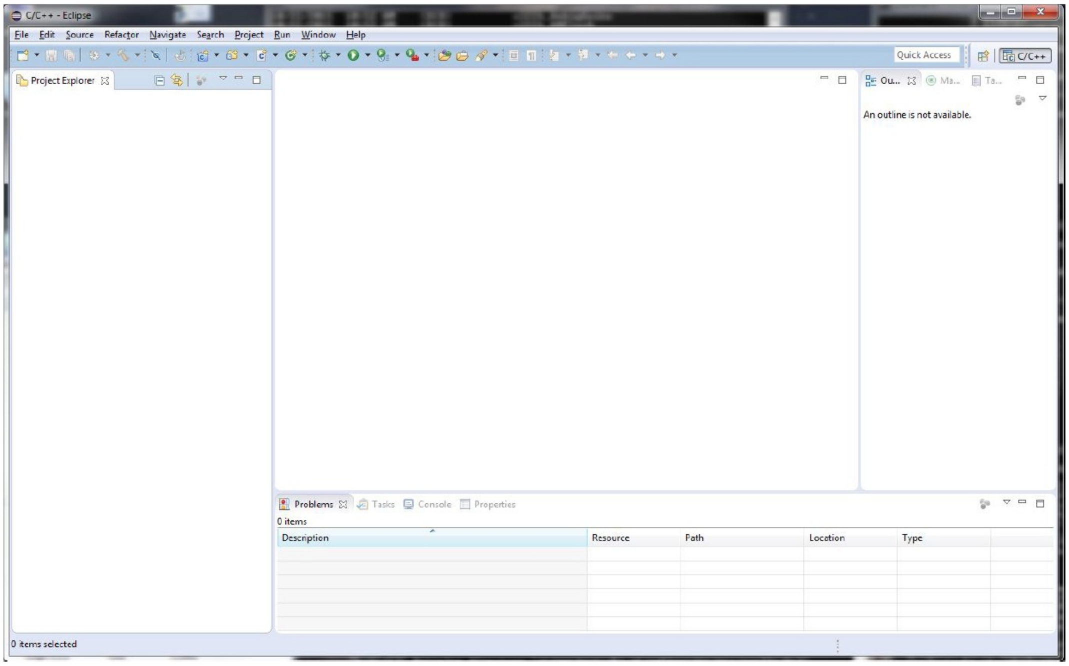
Once installed, an Eclipse shortcut will exist on the desktop.

1. Using the shortcut, start **Eclipse Mars** and resize the window as desired.

   The very first time Eclipse is started, a welcome screen will be displayed which contains various helpful links including Eclipse overview, Tutorials, Samples, What's New, and Go to the workbench.

2. Access the desired links and tutorials, then click **Go to the workbench**.

Figure 3-1: Initial Eclipse Window After Installation



An outline is not available.

0 items

0 items selected

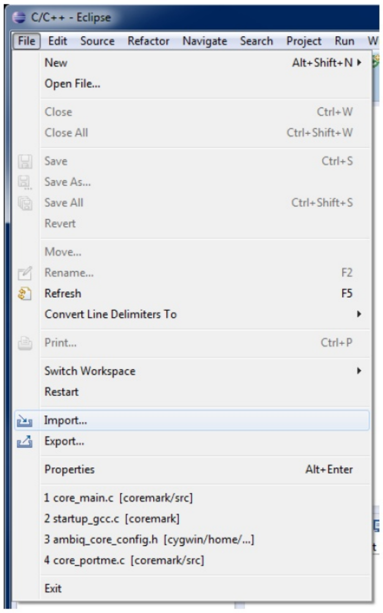
3.2 Importing an Eclipse Project and Debug

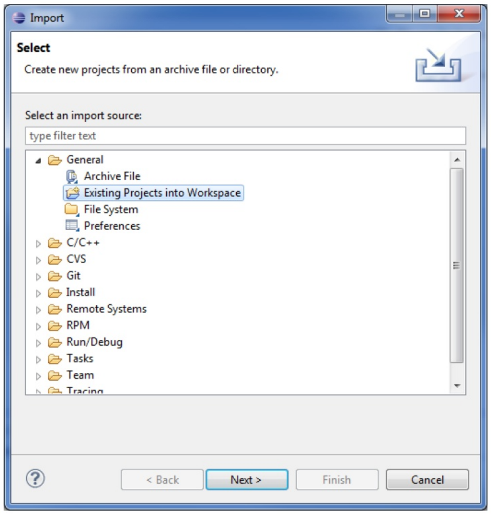
3.2.1 Importing and Building a Project

Importing an existing AmbiqSuite project is straightforward.

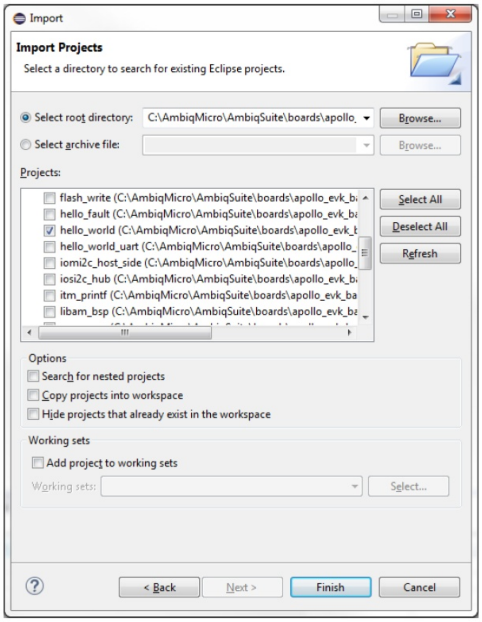
1. Click **File** (or alternatively, right-click in the Project Explorer area) and select **Import**.

2. In the window that pops up, expand **General**, and select **Existing Projects into Workspace**, then click **Next**.





3. On the final screen, browse to or copy the path to the project, click **Deselect All**, deselect **Copy projects into workspace**, then select the desired project(s) (in this case, hello_world) and select **Finish**, at which point the selected project(s) is loaded into the Project Explorer window of Eclipse.


Import

Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: C:\AmbiqMicro\AmbiqSuite\boards\apollo_

○ Select archive file:

Projects:

- □ flash_write (C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_b
- □ hello_fault (C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_b
- ☑ hello_world (C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_
- □ hello_world_uart (C:\AmbiqMicro\AmbiqSuite\boards\apollo_
- □ iomi2c_host_side (C:\AmbiqMicro\AmbiqSuite\boards\apollo_
- □ iosi2c_hub (C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_b
- □ itm_printf (C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_ba
- □ libam_bsp (C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_ba

Options

- □ Search for nested projects
- □ Copy projects into workspace
- □ Hide projects that already exist in the workspace

Working sets

- □ Add project to working sets

Working sets:

4. To build the imported project: select the desired project, click the hammer icon in the toolbar (or alternatively press **Ctrl+B**).

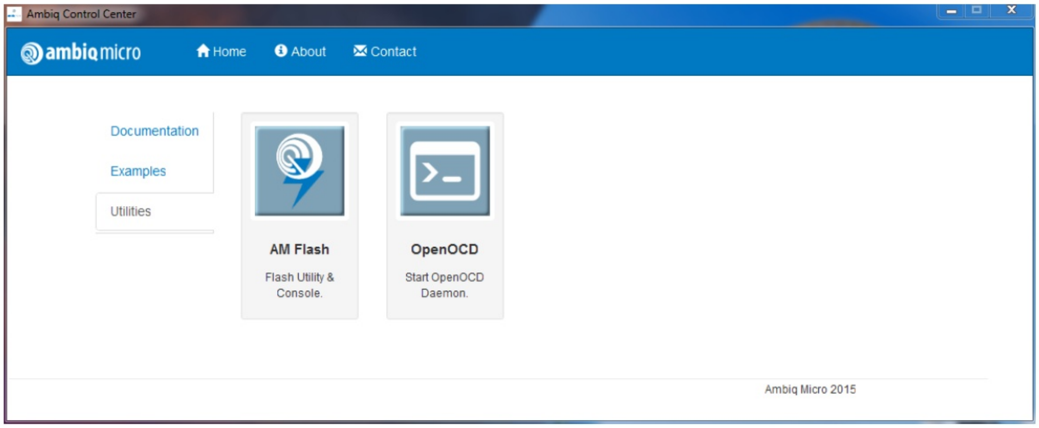
   The console will output some status messages. The project is ready for debugging once it has been successfully built.

3.2.2 Starting OpenOCD

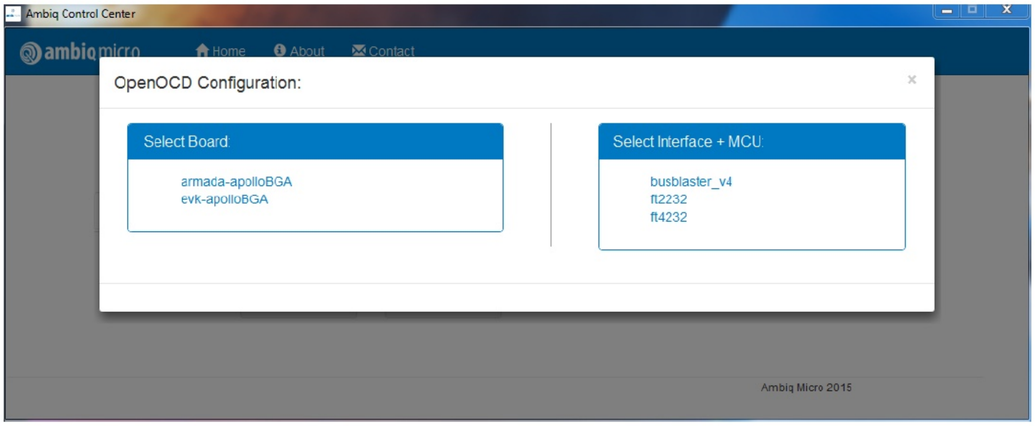
In order to debug within Eclipse, OpenOCD needs to be running. OpenOCD must be manually started to connect Eclipse to the running process using GDB. The easiest way to start OpenOCD is to use Ambiq Control Center.

1. Start Ambiq Control Center.

2. Open the **Utilities** tab, and click **OpenOCD** to start the OpenOCD Daemon.

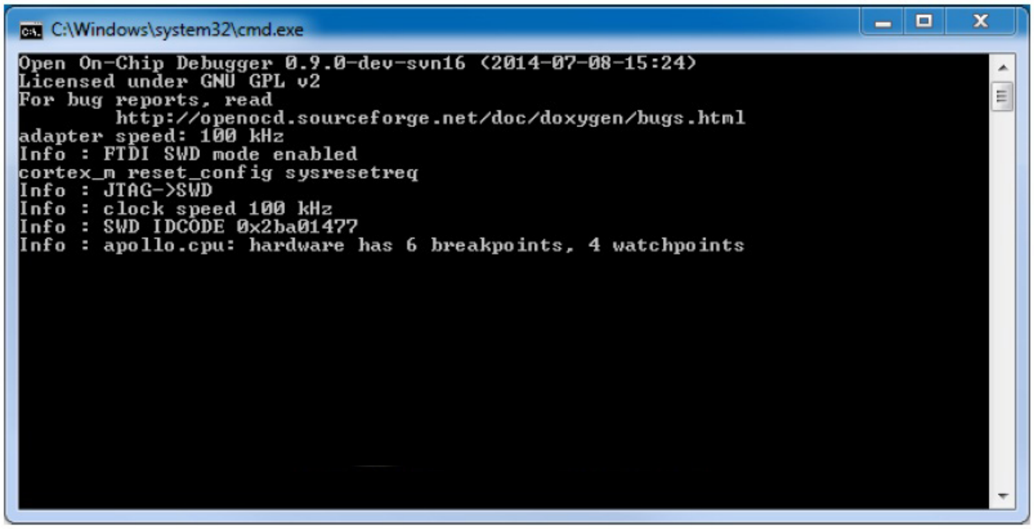


3. Select the board from the list on the left.



This should open a new window with information about the debug link to the Apollo device.

4. Check to make sure the output looks like below.



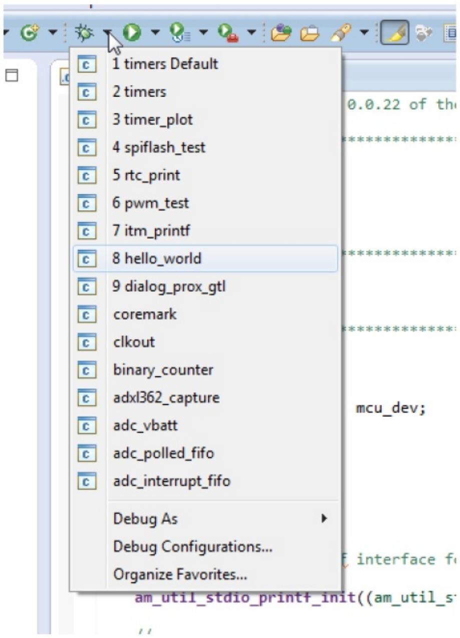
```
Open On-Chip Debugger 0.9.0-dev-svn16 (2014-07-08-15:24)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.sourceforge.net/doc/doxygen/bugs.html
adapter speed: 100 kHz
Info : FTDI SWD mode enabled
cortex_m reset_config sysresetreq
Info : JTAG->SWD
Info : clock speed 100 kHz
Info : SWD IDCODE 0x2ba01477
Info : apollo.cpu: hardware has 6 breakpoints, 4 watchpoints
```

If the OpenOCD output is different, make sure the Apollo device is plugged in and powered on, then close the window, and try again.

3.2.3 Debugging

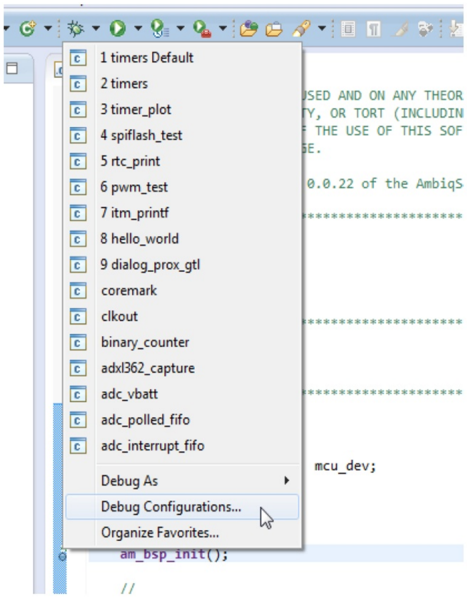
Now that OpenOCD is running and waiting for a GDB connection, it can now be connected.

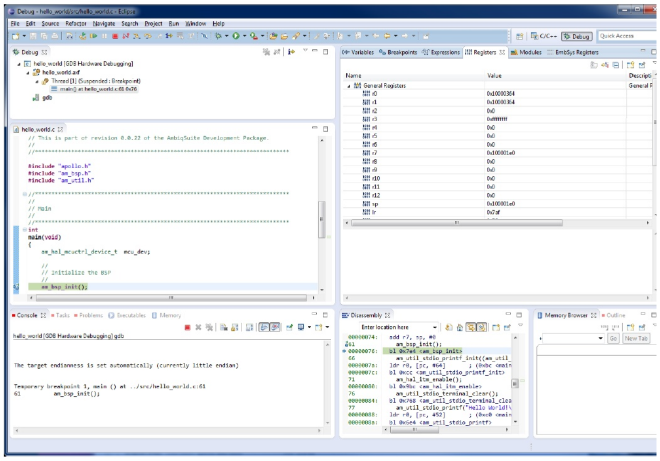
1. Click the down arrow next to the Debug option.
2. Click on **hello_world** (this name will change based on the name of the project).



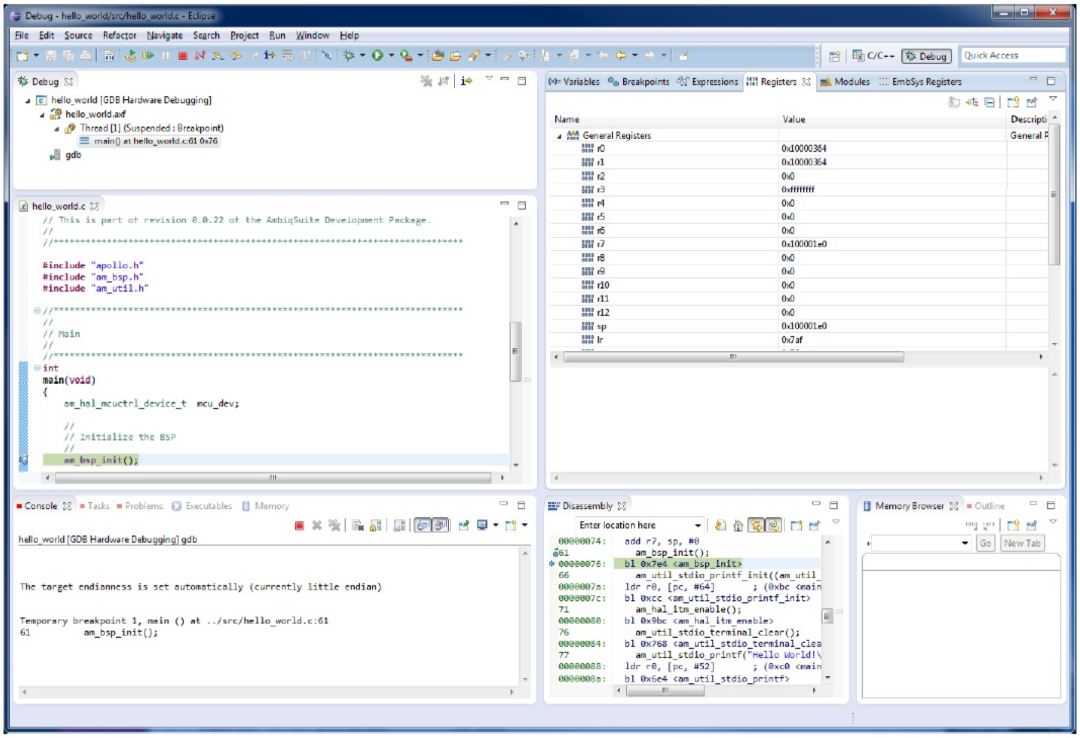
This will start GDB Hardware Debugging and open the Debug window.

Occasionally the drop down menu will fail to populate, especially on the first "import" operation in a new eclipse workspace. If this happens, start the debugger by clicking **Debug Configurations**. Select the project in the **Debug Configurations** dialog, and click **Debug**.





Set breakpoints, view memory and registers, run/halt or perform any debugging functions as needed.



```
// This is part of revision 0.0.22 of the AmbiqSuite Development Package.
//
//*****************************************************************************

#include "apollo.h"
#include "am_bsp.h"
#include "am_util.h"

//*****************************************************************************
//
// Main
//
//*****************************************************************************
int
main(void)
{
    am_hal_mcuctrl_device_t  mcu_dev;

    //
    // Initialize the BSP
    //
    am_bsp_init();
```

Name	Value	Descripti
General Registers		General F
r0	0x10000364	
r1	0x10000364	
r2	0x0	
r3	0xffffffff	
r4	0x0	
r5	0x0	
r6	0x0	
r7	0x100001e0	
r8	0x0	
r9	0x0	
r10	0x0	
r11	0x0	
r12	0x0	
sp	0x100001e0	
lr	0x7af	

```
hello_world [GDB Hardware Debugging] gdb


The target endianness is set automatically (currently little endian)


Temporary breakpoint 1, main () at ../src/hello_world.c:61
61          am_bsp_init();
```

```
00000074:   add r7, sp, #0
61            am_bsp_init();
00000076:   bl 0x7e4 <am_bsp_init>
66            am_util_stdio_printf_init((am_util_
0000007a:   ldr r0, [pc, #64]       ; (0xbc <main
0000007c:   bl 0xcc <am_util_stdio_printf_init>
71            am_hal_itm_enable();
00000080:   bl 0x9bc <am_hal_itm_enable>
76            am_util_stdio_terminal_clear();
00000084:   bl 0x768 <am_util_stdio_terminal_clea
77            am_util_stdio_printf("Hello World!\
00000088:   ldr r0, [pc, #52]       ; (0xc0 <main
0000008a:   bl 0x6e4 <am_util_stdio_printf>
```

SECTION

4

Build and Debug Using Keil

NOTE: Keil (minimum version v5.14) with the Ambiq Micro pack must be installed on the target PC.

If not already done so, go to the Keil MDK download page at https://www.keil.com/demo/eval/arm.htm. Download the MDK and install it as directed by the Keil documentation. Use the Keil MDK pack installer and install the pack for the Ambiq SoCs. For the Apollo EVK, use the APOLLO512BGA device. If using the Keil ULINK2 debug probe, then install it and its drivers now. Connect the ULINK2 to the Apollo EVK Base board and change the jumpers as described in the *Apollo EVK User's Guide*. If using the Ambiq supplied AGDI driver for the EVK instead of a ULINK2, then refer to *Section 2.3 Using the CoreSight 10-Pin Debugger Socket on page 11*.

4.1 Debugger Setup

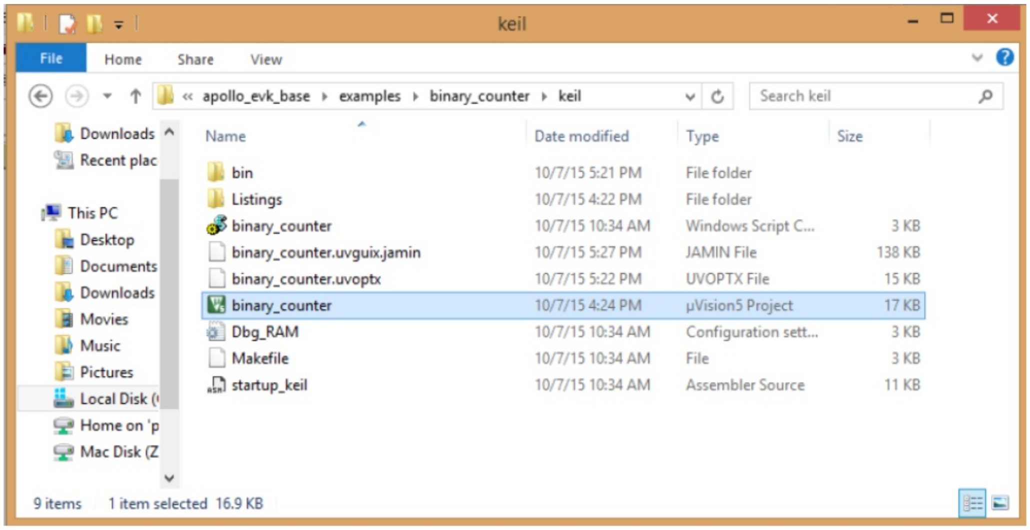
Ambiq EVK boards ship with an on-board debugger that can be used with Keil without a separately purchased debugger probe (e.g., ULINK2). Follow instructions in *AM AGDI Keil Supplemental User's Guide* to setup the debugger and then return to *Section 4.3 Loading Program to Flash on page 32*

4.2 Opening a Project and Build

Use the following to open a project and build:

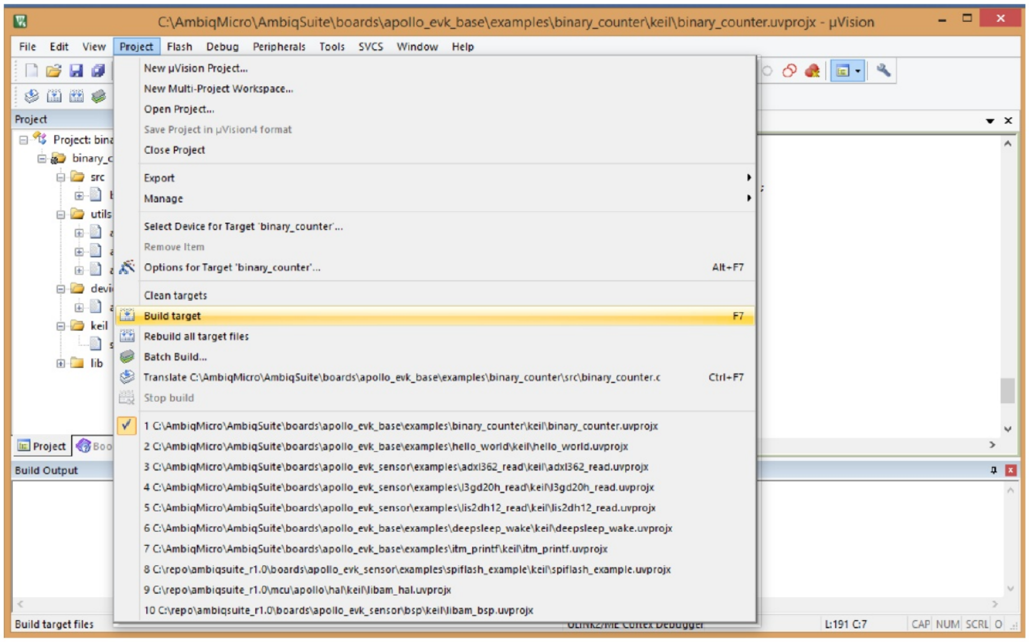
1. Open an existing Keil project **binary_counter** from AmbiqSuite.

   The default location for Ambiqsuite project **binary_counter** is: `C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\binary_counter\keil`



Name	Date modified	Type	Size
bin	10/7/15 5:21 PM	File folder	
Listings	10/7/15 4:22 PM	File folder	
binary_counter	10/7/15 10:34 AM	Windows Script C...	3 KB
binary_counter.uvguix.jamin	10/7/15 5:27 PM	JAMIN File	138 KB
binary_counter.uvoptx	10/7/15 5:22 PM	UVOPTX File	15 KB
binary_counter	10/7/15 4:24 PM	µVision5 Project	17 KB
Dbg_RAM	10/7/15 10:34 AM	Configuration sett...	3 KB
Makefile	10/7/15 10:34 AM	File	3 KB
startup_keil	10/7/15 10:34 AM	Assembler Source	11 KB

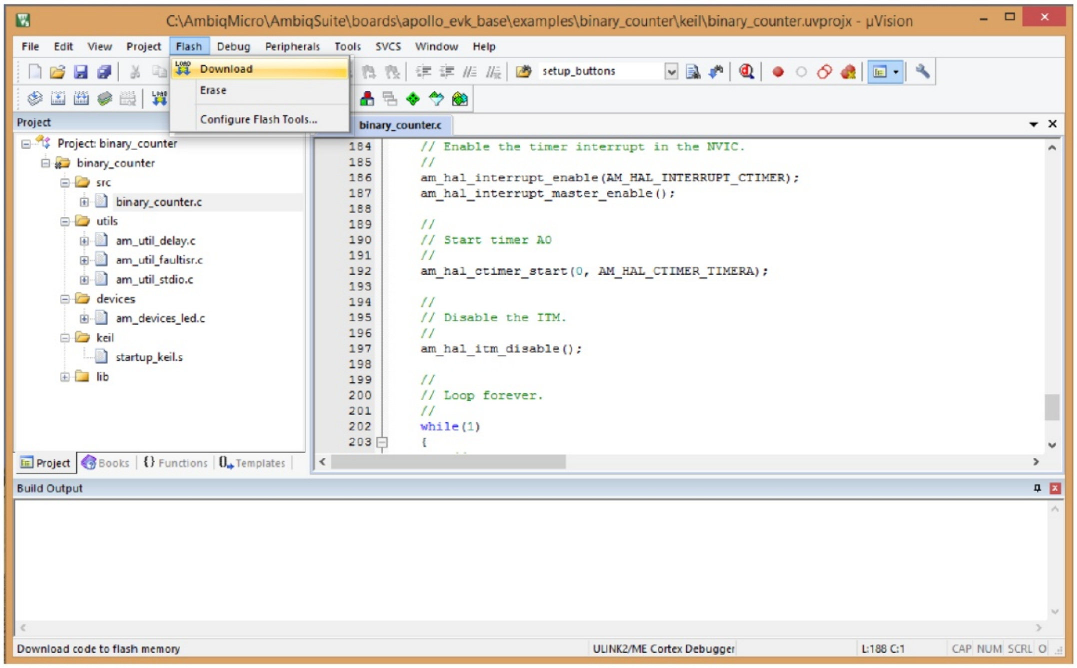
2. Double-click on the **uVision Project file** to open this projects using Keil
3. Build the projects at **Project** > **Build Target**.


C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\binary_counter\keil\binary_counter.uvprojx - µVision
File
Edit
View
Project
Flash
Debug
Peripherals
Tools
SVCS
Window
Help
New µVision Project...
New Multi-Project Workspace...
Open Project...
Save Project in µVision4 format
Close Project
Export
Manage
Select Device for Target 'binary_counter'...
Remove Item
Options for Target 'binary_counter'... Alt+F7
Clean targets
Build target F7
Rebuild all target files
Batch Build...
Translate C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\binary_counter\src\binary_counter.c Ctrl+F7
Stop build
1 C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\binary_counter\keil\binary_counter.uvprojx
2 C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\hello_world\keil\hello_world.uvprojx
3 C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_sensor\examples\adxl362_read\keil\adxl362_read.uvprojx
4 C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_sensor\examples\l3gd20h_read\keil\l3gd20h_read.uvprojx
5 C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_sensor\examples\lis2dh12_read\keil\lis2dh12_read.uvprojx
6 C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\deepsleep_wake\keil\deepsleep_wake.uvprojx
7 C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\itm_printf\keil\itm_printf.uvprojx
8 C:\repo\ambiqsuite_r1.0\boards\apollo_evk_sensor\examples\spiflash_example\keil\spiflash_example.uvprojx
9 C:\repo\ambiqsuite_r1.0\mcu\apollo\hal\keil\libam_hal.uvprojx
10 C:\repo\ambiqsuite_r1.0\boards\apollo_evk_sensor\bsp\keil\libam_bsp.uvprojx
Project
Project: bina
binary_c
src
utils
devi
keil
lib
Project
Boo
Build Output
Build target files
ULINK2/ME Cortex Debugger
L:191 C:7
CAP NUM SCRL O

4.3 Loading Program to Flash

Make sure to have a successfully built project and setup the debugger. Load the image to flash at Flash > Download as shown in Figure 4-1 on page 33.

Figure 4-1: Load Image to Flash Using Keil uVision

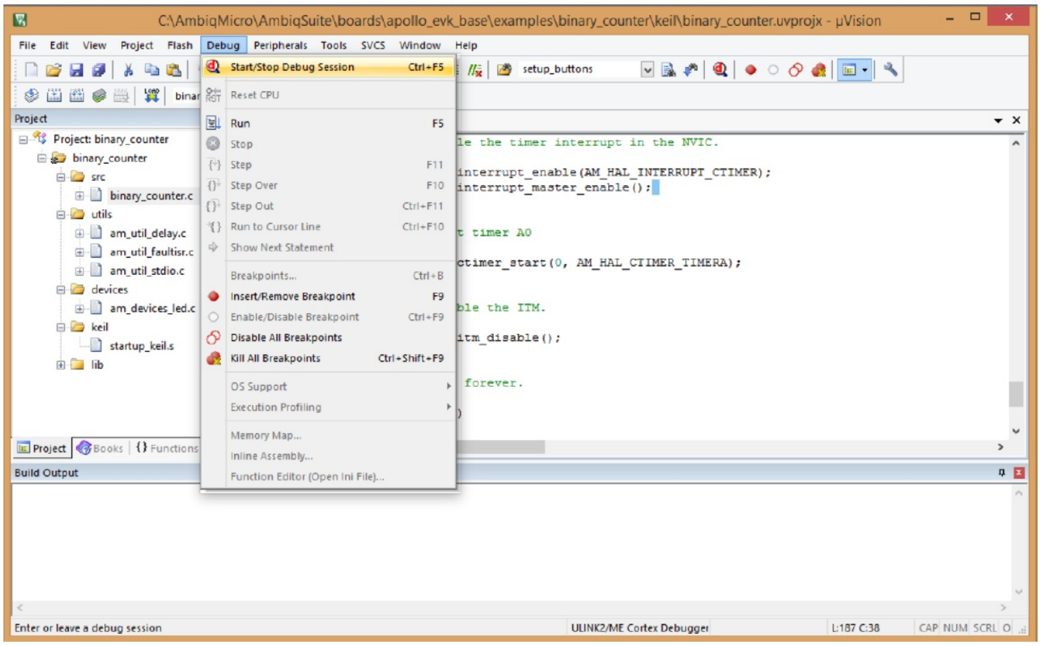


```
// Enable the timer interrupt in the NVIC.
//
am_hal_interrupt_enable(AM_HAL_INTERRUPT_CTIMER);
am_hal_interrupt_master_enable();

//
// Start timer A0
//
am_hal_ctimer_start(0, AM_HAL_CTIMER_TIMERA);

//
// Disable the ITM.
//
am_hal_itm_disable();

//
// Loop forever.
//
while(1)
{
```

4.4 Debugging

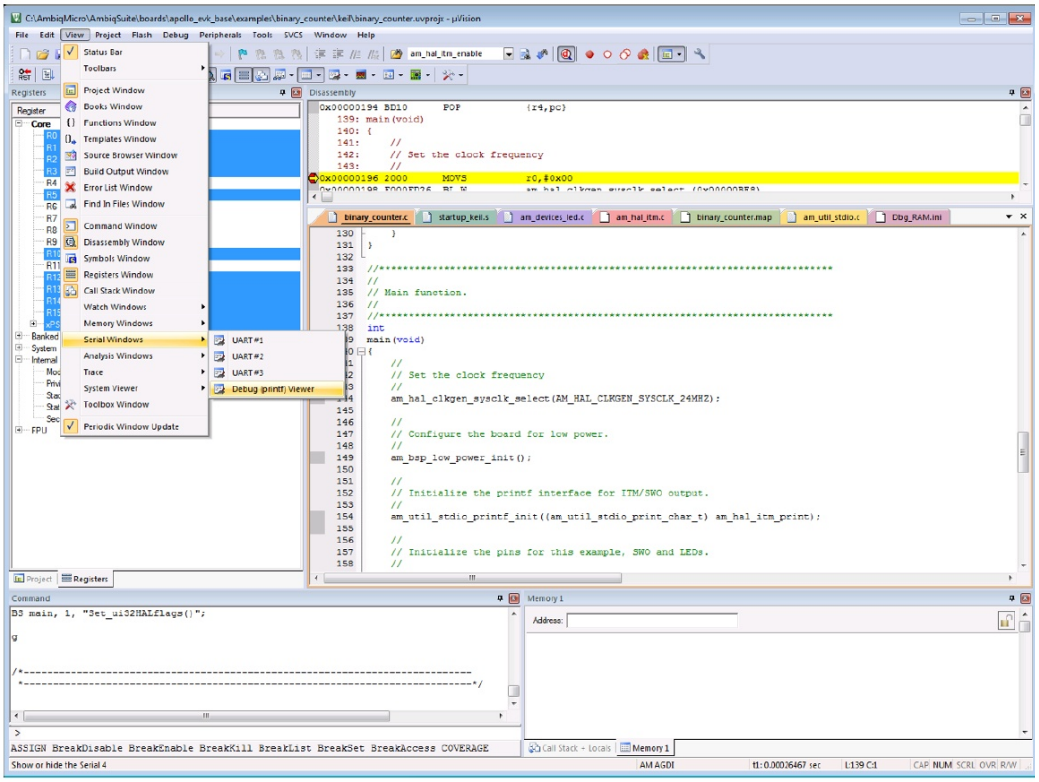
Make sure to setup the debugger.

1. Start debugging by clicking on **Debug** > **Start/Stop Debugging**.


C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\binary_counter\keil\binary_counter.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Start/Stop Debug Session Ctrl+F5
Reset CPU
Run F5
Stop
Step F11
Step Over F10
Step Out Ctrl+F11
Run to Cursor Line Ctrl+F10
Show Next Statement
Breakpoints... Ctrl+B
Insert/Remove Breakpoint F9
Enable/Disable Breakpoint Ctrl+F9
Disable All Breakpoints
Kill All Breakpoints Ctrl+Shift+F9
OS Support
Execution Profiling
Memory Map...
Inline Assembly...
Function Editor (Open Ini File)...
setup_buttons
Project
Project: binary_counter
binary_counter
src
binary_counter.c
utils
am_util_delay.c
am_util_faultisr.c
am_util_stdio.c
devices
am_devices_led.c
keil
startup_keil.s
lib
Project
Books
{} Functions
Build Output
Enter or leave a debug session
ULINK2/ME Cortex Debugger
L:187 C:38
CAP NUM SCRL O

```
le the timer interrupt in the NVIC.

interrupt_enable(AM_HAL_INTERRUPT_CTIMER);
interrupt_master_enable();

t timer A0

ctimer_start(0, AM_HAL_CTIMER_TIMERA);

ble the ITM.

itm_disable();

 forever.
```

2. Once in Debug Mode, open the Serial Debug Viewer Window to see debug printf messages from the Ambiq projects. Open this Window by clicking **View** > **Serial Windows** > **Debug(printf) Viewer**.

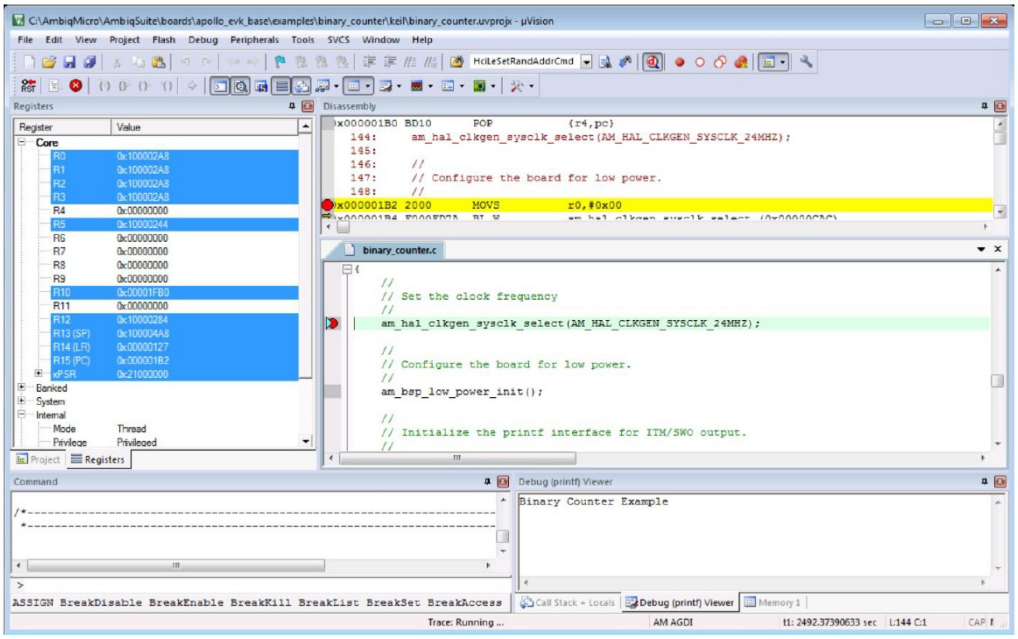


```
0x00000194 BD10      POP       {r4,pc}
   139: main(void)
   140: {
   141:     //
   142:     // Set the clock frequency
   143:     //
0x00000196 2000      MOVS      r0,#0x00
```

```
    }
}

//*****************************************************************************
//
// Main function.
//
//*****************************************************************************
int
main(void)
{
    //
    // Set the clock frequency
    //
    am_hal_clkgen_sysclk_select(AM_HAL_CLKGEN_SYSCLK_24MHZ);

    //
    // Configure the board for low power.
    //
    am_bsp_low_power_init();

    //
    // Initialize the printf interface for ITM/SWO output.
    //
    am_util_stdio_printf_init((am_util_stdio_print_char_t) am_hal_itm_print);

    //
    // Initialize the pins for this example, SWO and LEDs.
    //
```

```
BS main, 1, "Set_ui32HALflags()";
g
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE
```

The debug message from the example on the Debug(printf) Viewer will popup.


C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\binary_counter\keil\binary_counter.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Disassembly
binary_counter.c
Command
Debug (printf) Viewer
Call Stack + Locals
Memory 1
Trace: Running ...

Register	Value
Core	
R0	0x100002A8
R1	0x100002A8
R2	0x100002A8
R3	0x100002A8
R4	0x00000000
R5	0x10000244
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00001FB0
R11	0x00000000
R12	0x10000284
R13 (SP)	0x100004A8
R14 (LR)	0x00000127
R15 (PC)	0x000001B2
xPSR	0x21000000
Banked	
System	
Internal	
Mode	Thread
Privilege	Privileged

```
0x000001B0 BD10      POP       {r4,pc}
   144:     am_hal_clkgen_sysclk_select(AM_HAL_CLKGEN_SYSCLK_24MHZ);
   145:
   146:     //
   147:     // Configure the board for low power.
   148:     //
0x000001B2 2000      MOVS      r0,#0x00
```

```
{
    //
    // Set the clock frequency
    //
    am_hal_clkgen_sysclk_select(AM_HAL_CLKGEN_SYSCLK_24MHZ);

    //
    // Configure the board for low power.
    //
    am_bsp_low_power_init();

    //
    // Initialize the printf interface for ITM/SWO output.
    //
```

```
Binary Counter Example
```

SECTION

5

Build and Debug Using IAR

NOTE: The IAR EWARM (minimum version v7.40.5) must be installed on the PC.

If not already done so, go to the IAR EWARM download page at https://www.iar.com. Download the workbench and install it as directed by the documentation. If using the a third party debug probe such as the IAR I-jet, then install it and its drivers now according to the manufacturers instructions. Connect the third party debugger to the Apollo EVK Base board and change the jumpers as described in the *Apollo EVK User's Guide*. Refer to *Section 5.1.2 Third Party Debugger Setup (I-Jet) on page 38* on debugger setup for third party debuggers.

If using the gdb server with the Ambiq supplied Openocd driver for the EVK instead of third party debugger, then refer to *Section 5.1.1 On-Board Debugger Setup on page 36*.

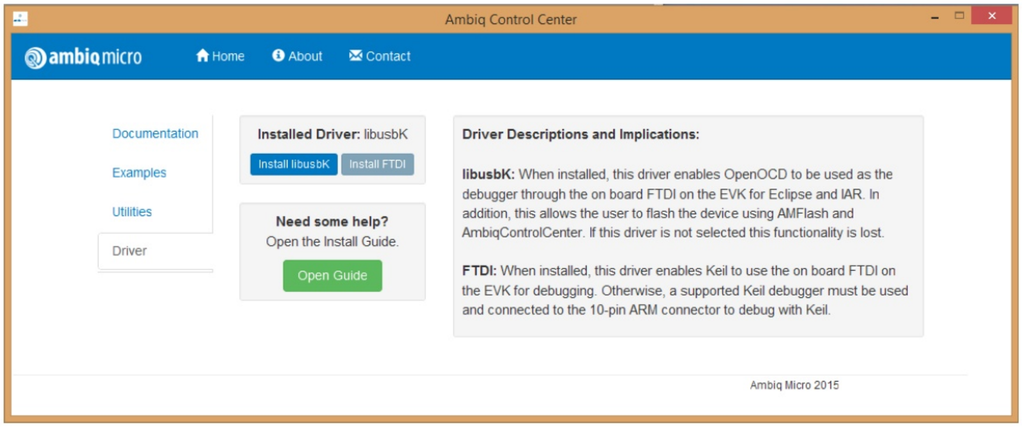
5.1 Debugger Setup

5.1.1 On-Board Debugger Setup

All Ambiq EVK boards ship with an on-board debugger that can be used with IAR. All Ambiq examples are configured by default to use the on-board debugger.

1. Verify that the libusbK drivers are installed on the system PC to use the on-board debugger. To do this open Ambiq Control Center, and check the **Drivers** tab. If libusbK driver is installed, skip to *Section 5.2 Opening a Project and Build on page 39*.

2. If the installed driver is **FTDI**, then click on **Install libusbK** to install the libusbK driver.



Installed Driver: libusbK

Need some help?
Open the Install Guide.

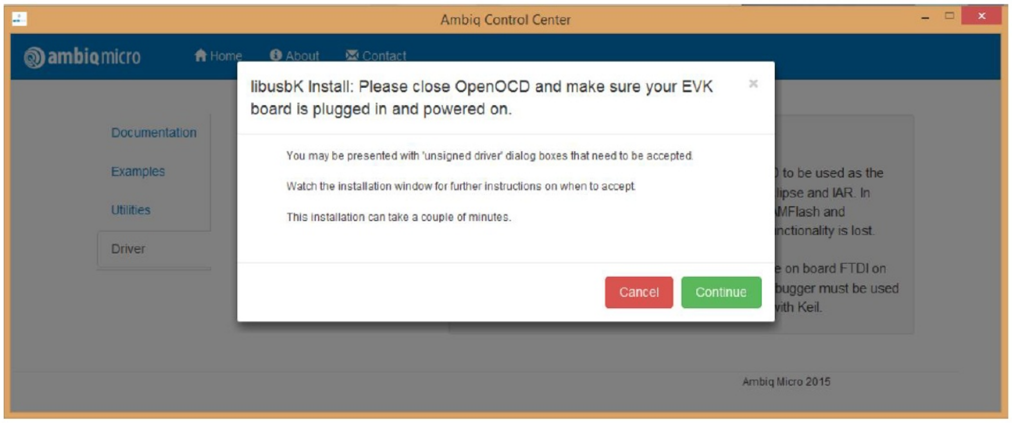
Driver Descriptions and Implications:

libusbK: When installed, this driver enables OpenOCD to be used as the debugger through the on board FTDI on the EVK for Eclipse and IAR. In addition, this allows the user to flash the device using AMFlash and AmbiqControlCenter. If this driver is not selected this functionality is lost.

FTDI: When installed, this driver enables Keil to use the on board FTDI on the EVK for debugging. Otherwise, a supported Keil debugger must be used and connected to the 10-pin ARM connector to debug with Keil.

Ambiq Micro 2015

3. Make sure the EVK is connected and powered on during installation. Verify that the EVK is connected, and powered on – in case missed on the first time, and click **Continue**.



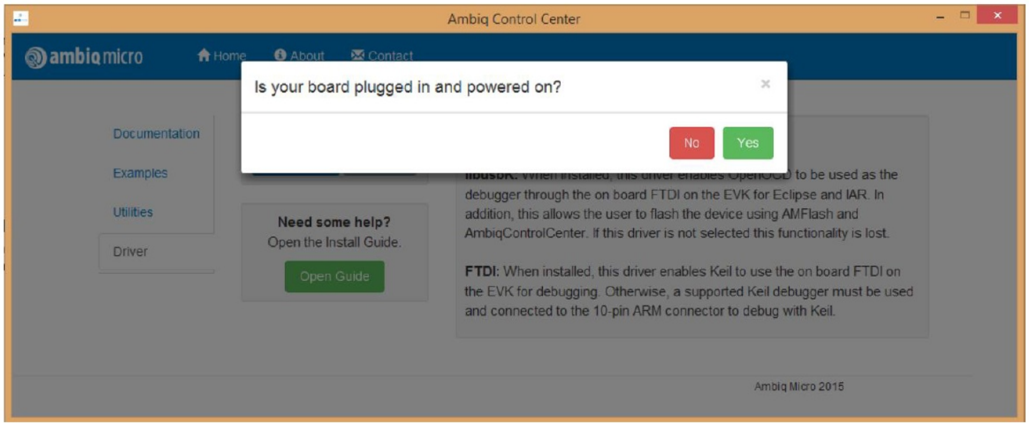
libusbK Install: Please close OpenOCD and make sure your EVK board is plugged in and powered on.

You may be presented with 'unsigned driver' dialog boxes that need to be accepted.

Watch the installation window for further instructions on when to accept.

This installation can take a couple of minutes.

4. Double check that the EVK is connected, and then continue with the driver install by clicking **Yes**.



Is your board plugged in and powered on?

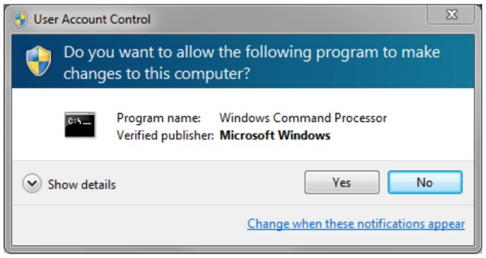
Need some help?
Open the Install Guide.

libusbK: When installed, this driver enables OpenOCD to be used as the debugger through the on board FTDI on the EVK for Eclipse and IAR. In addition, this allows the user to flash the device using AMFlash and AmbiqControlCenter. If this driver is not selected this functionality is lost.

FTDI: When installed, this driver enables Keil to use the on board FTDI on the EVK for debugging. Otherwise, a supported Keil debugger must be used and connected to the 10-pin ARM connector to debug with Keil.

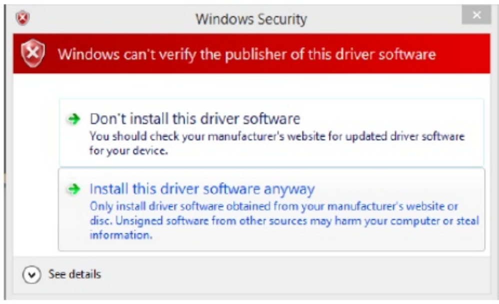
Ambiq Micro 2015

5. A few windows pop up like the User Account Control and the Windows Security Alert windows. Accept these to proceed with the installation.



Do you want to allow the following program to make changes to this computer?

Program name: Windows Command Processor
Verified publisher: **Microsoft Windows**



Windows can't verify the publisher of this driver software

Don't install this driver software
You should check your manufacturer's website for updated driver software for your device.

Install this driver software anyway
Only install driver software obtained from your manufacturer's website or disc. Unsigned software from other sources may harm your computer or steal information.

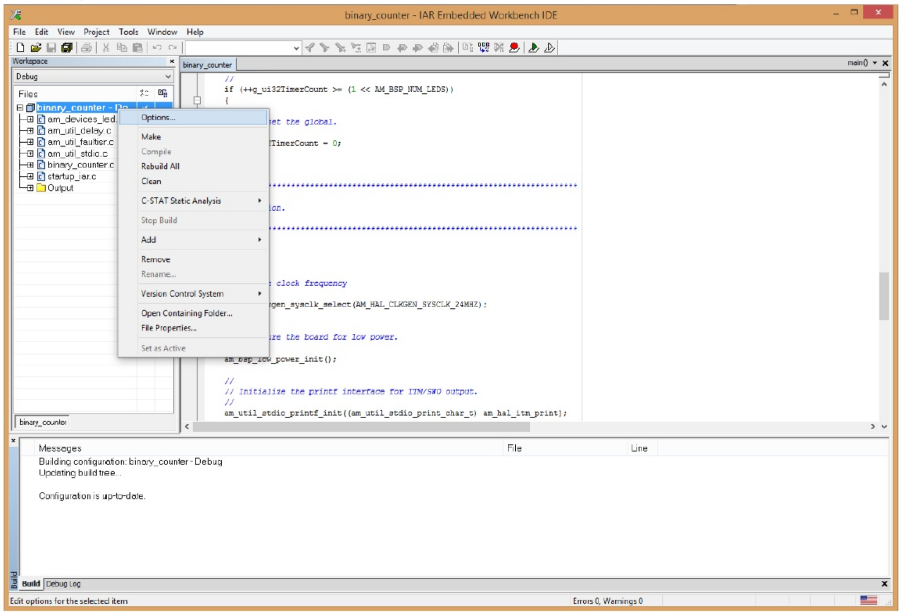
This could take several minutes. At the end of the installation the libusbK driver should be installed.

6. Power cycle the EVK and start using the libusbK driver.

5.1.2 Third Party Debugger Setup (I-Jet)

Use the following procedure to setup third party debugger (I-Jet):

1. Verify the drivers for the debug probe installed.
2. Right-click on the example workspace to open the Options screen.



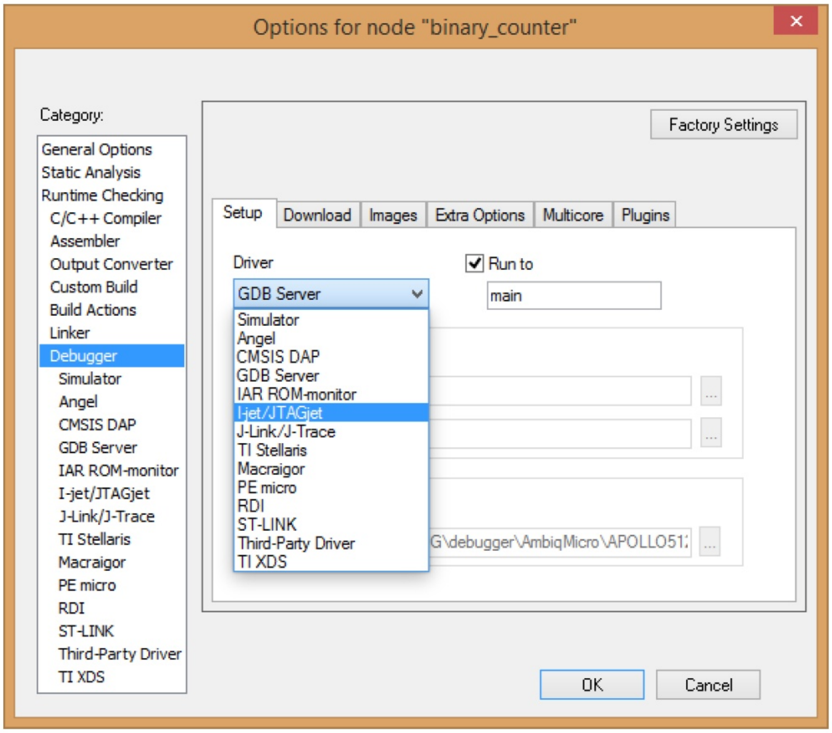
```
        //
        if (++g_ui32TimerCount >= (1 << AM_BSP_NUM_LEDS))
        {
            et the global.
            TimerCount = 0;

    *****************************************************************
    ion.
    *****************************************************************

    clock frequency
    gen_sysclk_select(AM_HAL_CLKGEN_SYSCLK_24MHZ);

    re the board for low power.
    am_bsp_low_power_init();

    //
    // Initialize the printf interface for ITM/SWO output.
    //
    am_util_stdio_printf_init((am_util_stdio_print_char_t) am_hal_itm_print);
```

Messages	File	Line
Building configuration: binary_counter - Debug		
Updating build tree...		
Configuration is up-to-date.		

Edit options for the selected item

Errors 0, Warnings 0

3. In the Options screen, select **Debugger** in **Category**, and then select **I-jet/JTAGjet** (or the third party debugger of preference) from the **Driver**'drop down menu.



Driver: GDB Server

☑ Run to: main

G\debugger\AmbiqMicro\APOLLO51

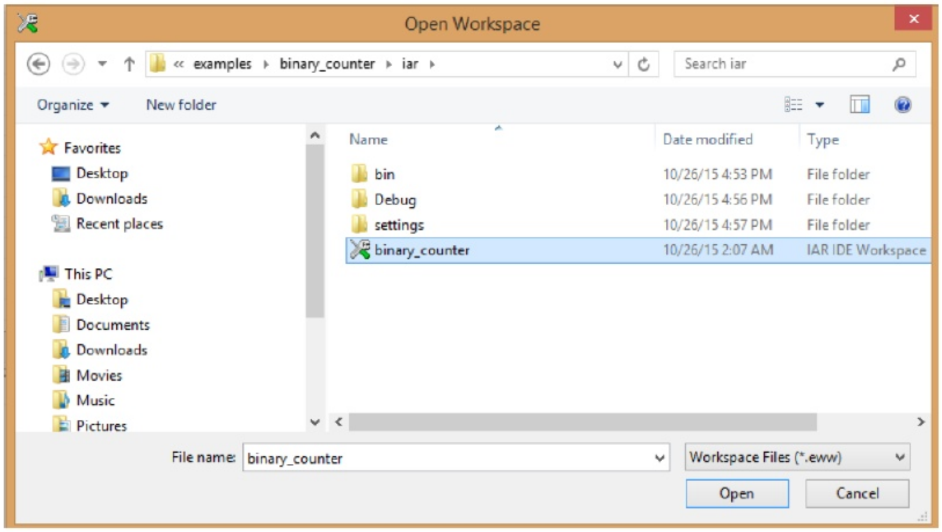
5.2 Opening a Project and Build

Use the following procedure to open a project and build:

1. Open an existing IAR project **binary_counter** from AmbiqSuite.

   The default location for AmbiqSuite project **binary_counter** is: `C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\binary_counter\iar`

2. Double-click on the **IAR IDE Workspace** file to open this projects using IAR.

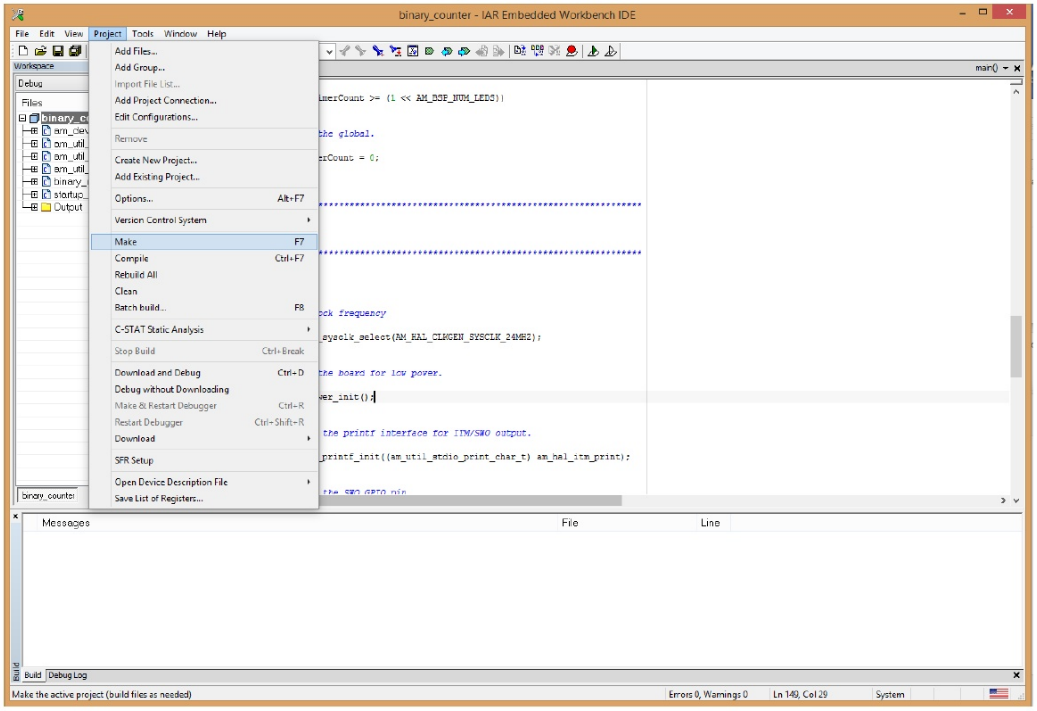


Name	Date modified	Type
bin	10/26/15 4:53 PM	File folder
Debug	10/26/15 4:56 PM	File folder
settings	10/26/15 4:57 PM	File folder
binary_counter	10/26/15 2:07 AM	IAR IDE Workspace

File name: binary_counter

Workspace Files (*.eww)

3. Build the projects at **Project** > **Make**.



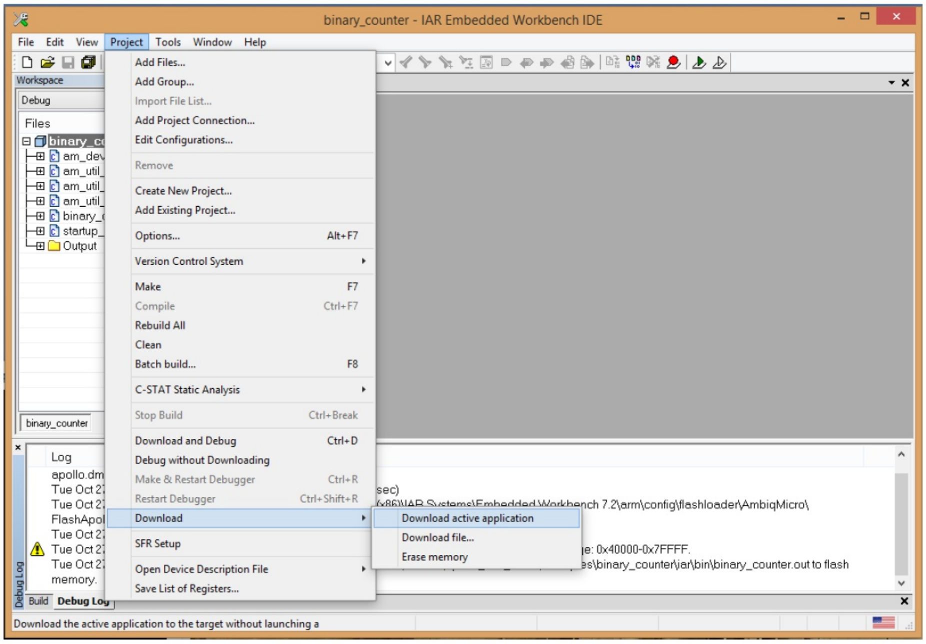
```
imerCount >= (1 << AM_BSP_NUM_LEDS))

the global.

erCount = 0;

****************************************************************

****************************************************************

ock frequency

_sysclk_select(AM_HAL_CLKGEN_SYSCLK_24MHZ);

the board for low power.

wer_init();

 the printf interface for ITM/SWO output.

_printf_init((am_util_stdio_print_char_t) am_hal_itm_print);

 the SWO GPIO pin
```

Make the active project (build files as needed)   Errors 0, Warnings 0   Ln 149, Col 29   System

5.3 Loading Program to Flash

Make sure to successfully built the project and setup the debugger.

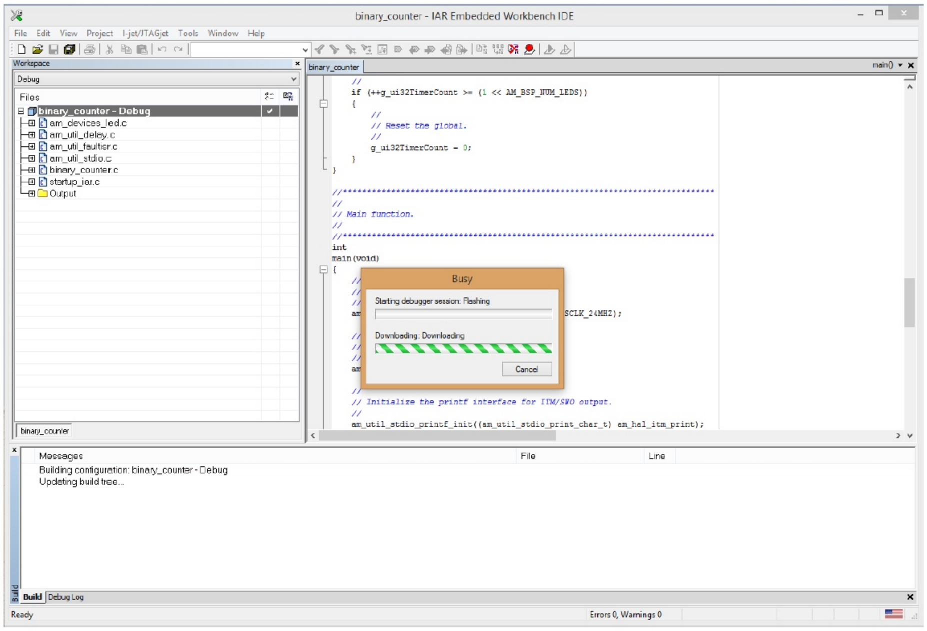
1. Start OpenOCD. Refer to *Section 3.2.2 Starting OpenOCD on page 28* for a refresher.
2. Load the image to flash at **Project** > **Download** > **Download Active Application**.



```
Log
apollo.dm                                sec)
Tue Oct 2                                (x86)\IAR Systems\Embedded Workbench 7.2\arm\config\flashloader\AmbiqMicro\
Tue Oct 2
FlashApol
Tue Oct 2                                                                        e: 0x40000-0x7FFFF.
Tue Oct 2                                                                        es\binary_counter\iar\bin\binary_counter.out to flash
Tue Oct 2
memory.
```

Download the active application to the target without launching a

A progress bar will be shown while the image is written to flash.



```
    //
    if (++g_ui32TimerCount >= (1 << AM_BSP_NUM_LEDS))
    {
        //
        // Reset the global.
        //
        g_ui32TimerCount = 0;
    }
}

//*****************************************************************************
//
// Main function.
//
//*****************************************************************************
int
main(void)
{
                                         SCLK_24MHZ);

    //
    // Initialize the printf interface for ITM/SWO output.
    //
    am_util_stdio_printf_init((am_util_stdio_print_char_t) am_hal_itm_print);
```

Starting debugger session: Flashing

Downloading: Downloading

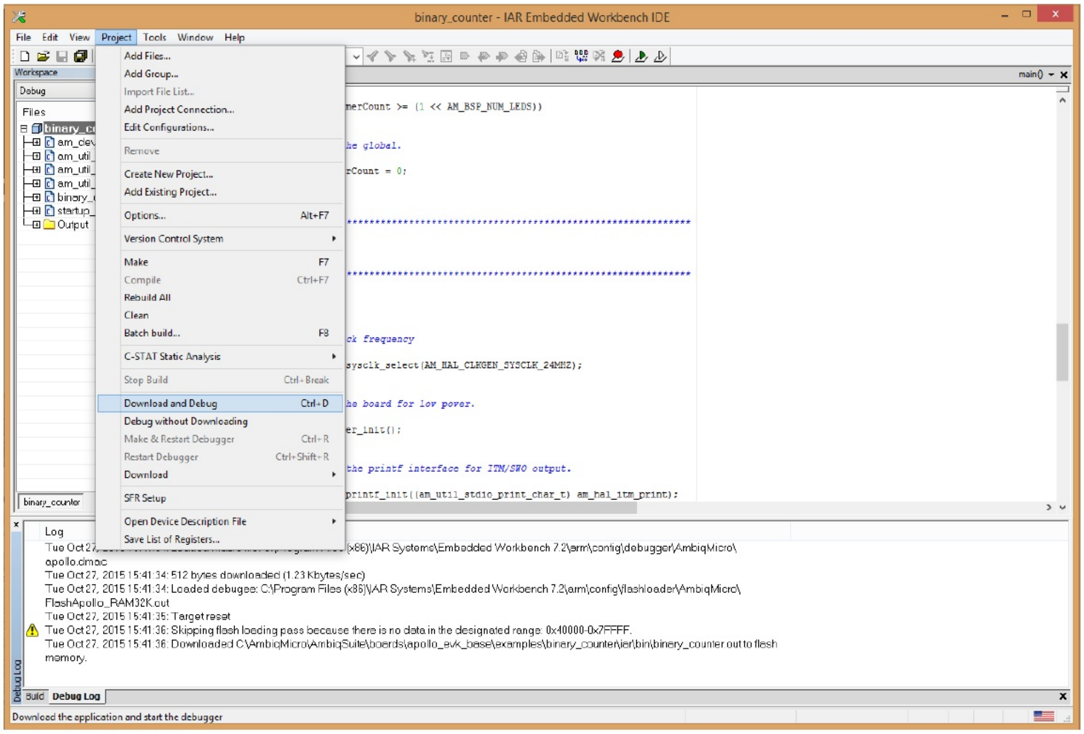
```
Messages                                                File        Line
Building configuration: binary_counter - Debug
Updating build tree...
```

Ready Errors 0, Warnings 0

5.4 Debugging

Make sure to setup the debugger. Start debugging by clicking on **Project** > **Download** and **Debug** as shown in Figure 5-1.

Figure 5-1: Download and Debug



```
merCount >= (1 << AM_BSP_NUM_LEDS))

he global.

rCount = 0;

*****************************************************************

*****************************************************************

ck frequency

sysclk_select(AM_HAL_CLKGEN_SYSCLK_24MHZ);

he board for low power.

er_init();

the printf interface for ITM/SWO output.

printf_init((am_util_stdio_print_char_t) am_hal_itm_print);
```

```
Tue Oct 27, (x86)\IAR Systems\Embedded Workbench 7.2\arm\config\debugger\AmbiqMicro\
apollo.dmac
Tue Oct 27, 2015 15:41:34: 512 bytes downloaded (1.23 Kbytes/sec)
Tue Oct 27, 2015 15:41:34: Loaded debugee: C:\Program Files (x86)\IAR Systems\Embedded Workbench 7.2\arm\config\flashloader\AmbiqMicro\
FlashApollo_RAM32K.out
Tue Oct 27, 2015 15:41:35: Target reset
Tue Oct 27, 2015 15:41:36: Skipping flash loading pass because there is no data in the designated range: 0x40000-0x7FFFF.
Tue Oct 27, 2015 15:41:36: Downloaded C:\AmbiqMicro\AmbiqSuite\boards\apollo_evk_base\examples\binary_counter\iar\bin\binary_counter.out to flash
memory.
```

SECTION

6

Troubleshooting

There are several situations that can cause connectivity issues between the PC and the Apollo device. This section explains some common issues and their solutions.

6.1 Physical Debugger Connection Issues

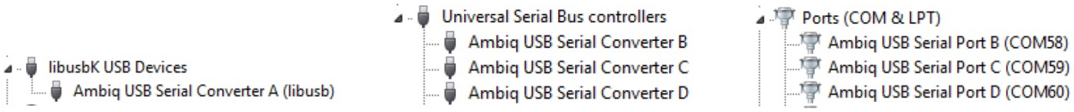
One of the most common reasons for connectivity issues between the PC and an Apollo EVK is related to jumper settings. The Apollo EVK has multiple jumpers related to power and SWD signal routing. If any of these have been placed incorrectly, OpenOCD will either report the absence of a device, or it may report a device with the wrong number of breakpoints and watchpoints. Also, all tools that access the Apollo SoC over SWD (including Eclipse, AM Flash, and Ambiq Control Center) will report errors. Setting the power and SWD jumpers back to their default positions (which can be found in the User's Guide for the EVK) will solve this issue.

6.2 Windows Driver Issues

If an Apollo board is unresponsive, even with all jumpers in their default positions, it's possible that the connection issue is actually related to a Windows USB Driver problem. To check for USB driver issues, open the Windows Device Manager. This can be found under the **Control Panel** > **Hardware and Sound**.

For a correctly enumerated Apollo EVK, the device manager should have entries that look like Figure 6-1:

Figure 6-1: Device Manager Entries



libusbK USB Devices
- Ambiq USB Serial Converter A (libusb)

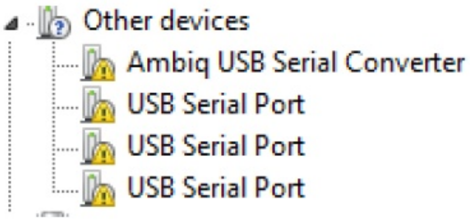
Universal Serial Bus controllers
- Ambiq USB Serial Converter B
- Ambiq USB Serial Converter C
- Ambiq USB Serial Converter D

Ports (COM & LPT)
- Ambiq USB Serial Port B (COM58)
- Ambiq USB Serial Port C (COM59)
- Ambiq USB Serial Port D (COM60)

If any these entries are missing or incorrect, some or all of the SWD or ITM/SWO connections to the board will be unusable. To resolve this issue, re-install the device drivers for the Ambiq debugger hardware. This can either be done automatically by running the Ambiq Driver Installer executable, or, if Ambiq Control Center is already installed it can be performed manually.

To re-install drivers manually, open the Windows Device Manager and locate the devices that have missing or incorrect drivers. Usually, they devices will appear in the device manager in the following form:

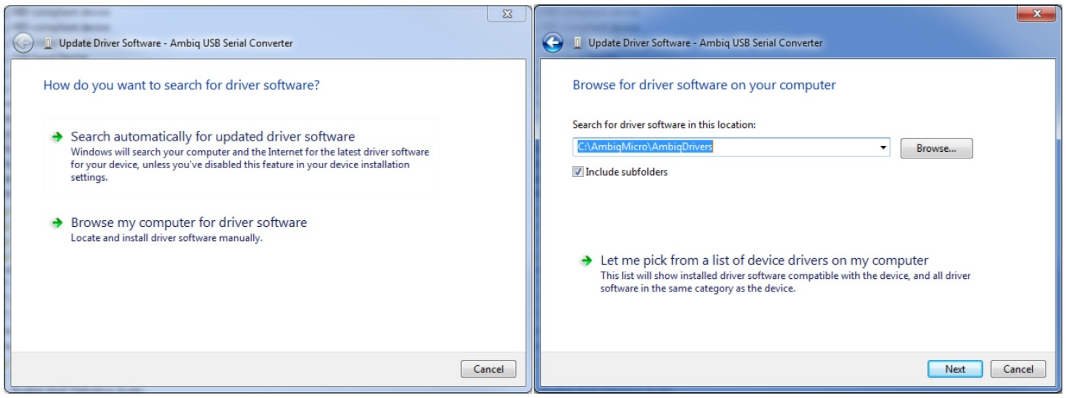
Figure 6-2: Missing Drivers



Other devices
- Ambiq USB Serial Converter
- USB Serial Port
- USB Serial Port
- USB Serial Port

Right-click on one of the devices, and select **Update Driver Software**. This will open the following dialog. Select **Browse my computer for driver software**, and provide the path to the AmbiqDrivers directory that installs along with Ambiq Control Center. The default location for this folder is `C:\AmbiqMicro\AmbiqDrivers`. Make sure that the **Include subfolders** option is selected, and click **Next**.

Figure 6-3: Updating Drivers



Update Driver Software - Ambiq USB Serial Converter

How do you want to search for driver software?

→ Search automatically for updated driver software
Windows will search your computer and the Internet for the latest driver software for your device, unless you've disabled this feature in your device installation settings.

→ Browse my computer for driver software
Locate and install driver software manually.

Update Driver Software - Ambiq USB Serial Converter

Browse for driver software on your computer

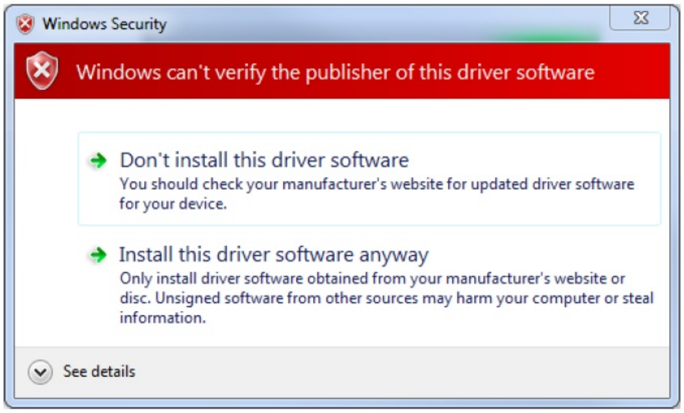
Search for driver software in this location:
C:\AmbiqMicro\AmbiqDrivers

☑ Include subfolders

→ Let me pick from a list of device drivers on my computer
This list will show installed driver software compatible with the device, and all driver software in the same category as the device.

Windows may report that the drivers are unsigned or that the manufacturer cannot be identified. This is normal, especially for early versions of the Ambiq debugger-interface drivers. To continue with driver installation, select **Install this driver software anyway**.

Figure 6-4: Driver Verification



Windows Security

Windows can't verify the publisher of this driver software

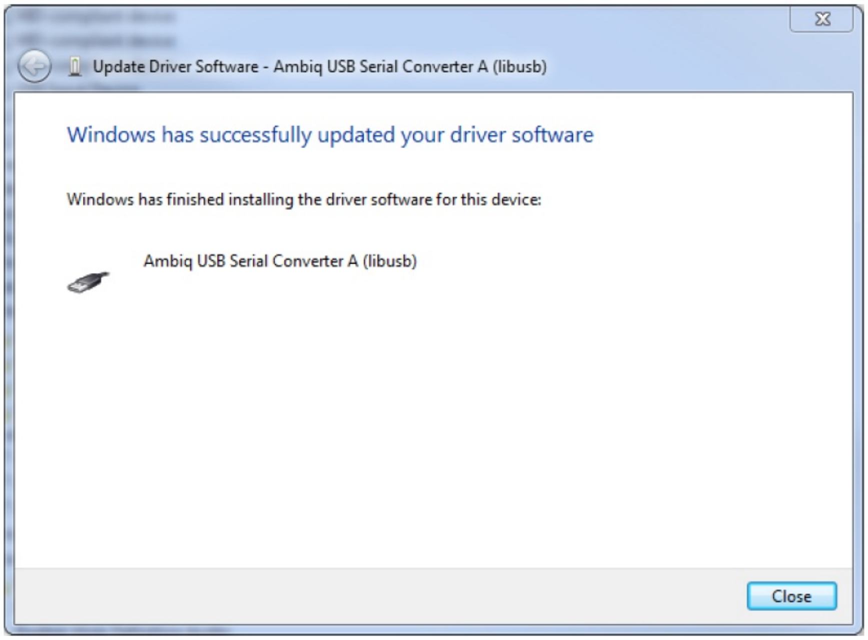
Don't install this driver software
You should check your manufacturer's website for updated driver software for your device.

Install this driver software anyway
Only install driver software obtained from your manufacturer's website or disc. Unsigned software from other sources may harm your computer or steal information.

See details

A window will indicate if the installation is successful. Repeat this process for each of the USB interfaces with missing drivers. After this is complete, the Ambiq debugger tools should be able to connect to the Apollo EVK correctly.

Figure 6-5: Successful Driver Installation



Update Driver Software - Ambiq USB Serial Converter A (libusb)

Windows has successfully updated your driver software

Windows has finished installing the driver software for this device:

Ambiq USB Serial Converter A (libusb)

ambiq

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

A-SOCAP1-QSGA02EN v1.0
January 2023