Getting Started with the Artemis Development Kit

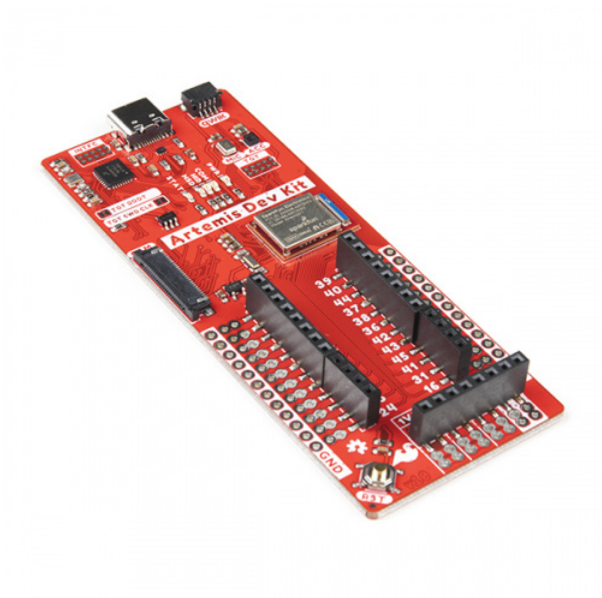
Introduction

OS Requirements: The software utilized to program and use the Artemis Development Kit (ADK) may have limitations on various operating systems (OS):

- **Window 10** (or later) is required for almost all the additional software required for the ADK.
- Currently (7/7/20), there are no limitations for **Mac OS** due to their required OS updates.
- For **Linux** users, we have tested our instructions on Ubuntu 18.04.4 LTS desktop. Users may need to verify the software compatibility for their Linux distribution. In addition, users may need to adapt our instructions to their flavor of Linux.

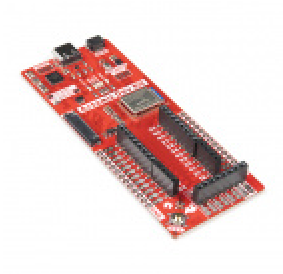
Mbed™ Release: Pending the adoption of the Artemis module into Mbed™ OS, we have temporarily omitted parts of this tutorial. Users looking to get a jump start with Mbed™, can check out this short tutorial on the *beta* version.

The Artemis Development Kit and Artemis Development Kit (with Camera) are the latest development kit from our Artemis family of products. If you are looking to push the edge of your software development skills, the Artemis Development Kit is what you have been waiting for!



SparkFun Artemis Development Kit

SparkFun Artemis Development Kit with Camera
◉ KIT-17071



SparkFun Artemis Development Kit
◉ DEV-16828

Himax CMOS Imaging Camera - HM01B0
◉ SEN-15570

USB 3.1 Cable A to C - 3 Foot
◉ CAB-14743

Jumper Modification

If you would like to modify the jumpers, you will need soldering equipment and/or a hobby knife.

Solder Lead Free - 100-gram Spool
◉ TOL-09325

Chip Quik No-Clean Flux Pen - 10mL
◉ TOL-14579

Weller WLC100 Soldering Station
⊖ TOL-14228

Hobby Knife
◉ TOL-09200

Programming Firmware

Users interested in modifying or updating the firmware on the NXP chip will need an Arm® Programmer and need to solder on a JTAG header. The same programmer and an **additional** header can also be used to program and debug the Atermis module. We recommend these programmers from our catalog:

J-Link BASE Compact Programmer
○ PGM-15347

J-Link EDU Base Programmer
○ PGM-15346

J-Link EDU Mini Programmer
○ PGM-15345

Header - 2x5 Pin (Male, 1.27mm)
◉ PRT-15362

Suggested Reading

As a more professionally oriented product, we will skip over the more fundamental tutorials (i.e. **Ohm's Law** and **What is Electricity?**). However, below are a few tutorials that may help users familiarize themselves with various aspects of the board.

Serial Communication

Asynchronous serial communication concepts: packets, signal levels, baud rates, UARTs and more!

Serial Peripheral Interface (SPI)

SPI is commonly used to connect microcontrollers to peripherals such as sensors, shift registers, and SD cards.

I2C

An introduction to I2C, one of the main embedded communications protocols in use today.

Bluetooth Basics

An overview of the Bluetooth wireless technology.

Logic Levels

Learn the difference between 3.3V and 5V devices and logic levels.

ARM Programming

How to program SAMD21 or SAMD51 boards (or other ARM processors).

Installing an Arduino Library

How do I install a custom Arduino library? It's easy! This tutorial will go over how to install an Arduino library using the Arduino Library Manager. For libraries not linked with the Arduino IDE, we will also go over manually installing an Arduino library.

Using SparkFun Edge Board with Ambiq Apollo3 SDK

We will demonstrate how to get started with your SparkFun Edge Board by setting up the toolchain on your computer, examining an example program, and using the serial uploader tool to flash the chip.

Artemis Development with Arduino

Get our powerful Artemis based boards (Artemis Nano, BlackBoard Artemis, and BlackBoard Artemis ATP) blinking in less than 5 minutes using the SparkFun Artemis Arduino Core!

Designing with the SparkFun Artemis

Let's chat about layout and design considerations when using the Artemis module.

Getting Started with the micro:bit

The BBC micro:bit is a compact, powerful programming tool that requires no software installation. Read on to learn how to use it YOUR way!

Programming the SparkFun Edge with Arduino

Running low-power machine learning examples on the SparkFun Edge can now be done using the familiar Arduino IDE. In this follow-up to the initial Edge tutorial, we'll look at how to get three examples up and running without the need to learn an entirely new SDK.

New!

Artemis Development on Arm® Mbed™ OS (Beta)

New!

Installing Board Definitions in the Arduino IDE

How do I install a custom Arduino board/core? It's easy! This tutorial will go over how to install an Arduino

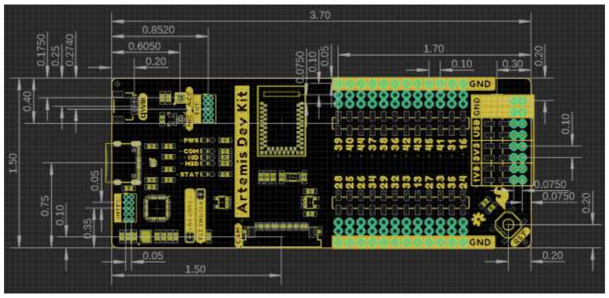
◉ DEV-16828

SparkFun Artemis Development Kit with Camera
◉ KIT-17071

This guide will cover the general design of the development board and the installation of the recommended software used to program the Artemis DK. The primary development programs are the AmbiqSDK, Mbed™ CLI, and the Arduino IDE. In addition, we have provided basic examples to verify the operation of the board. For more advanced functionalities, we have software development guides for each of the recommended software platforms that users can reference.

Product showcase video.

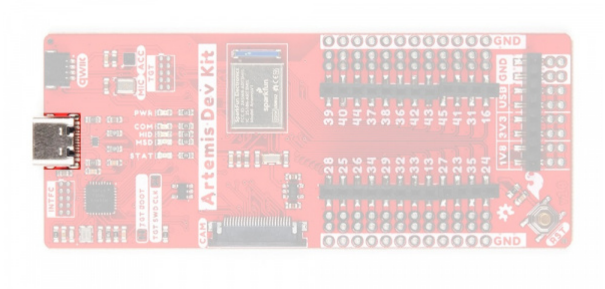
The board dimensions are illustrated in the drawing below. The primary highlight in the layout are the breakout pins, which provide both a .1" pitch spacing for headers and a .08" pitch spacing for IC-hooks, used by most logic analyzers.



Board dimensions (PDF) for the Artemis Development Kit, in inches. (Click to enlarge)

USB-C Connector

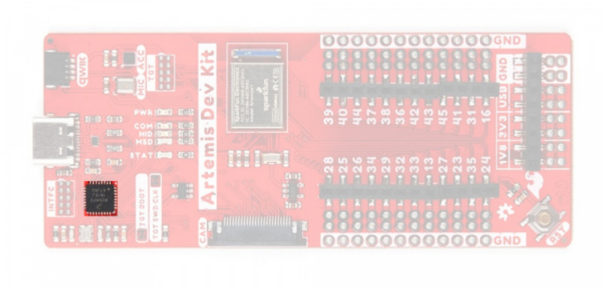
The USB connector is provided to power and program the board. For most users, it will be the primary programing interface for the Artemis module. A more detailed description of that functionality with the interface chip, is documented in the next section.



USB-C connector on the Artemis Development Kit. (Click to enlarge)

Interface Chip

Similar to the CH340, FTDI, or 32U4 chips on the SparkFun RedBoard or Arduino Uno; the MKL26Z128VFM4 Arm® Cortex®-M0+ MCU, from NXP, operates as an interface chip to the Artemis module. (**More advanced developers may recognize this DAPlink and the USB interface from the micro:bit.*)

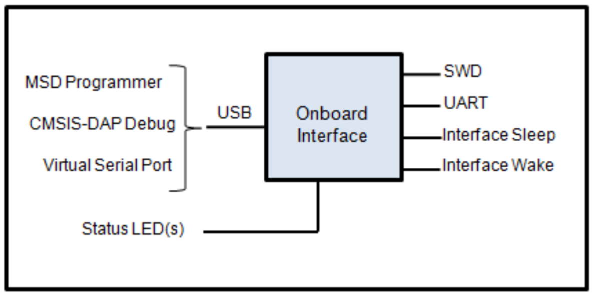


Interface IC for the Artemis Development Kit. (Click to enlarge)

The MKL26Z128VFM4 microcontroller is connected to the software debug (SWD), reset, and UART pins of the Artemis module. To operate as an interface chip, it utilizes firmware from Arm® Mbed™ OS called `Arm Mbed DAPLink` *(link)*, which a bootloader is pre-programmed through the `intfc` JTAG pins *(see details in the **Firmware** section below)*. The firmware enables the board to enumerate as a composite USB device, when connected to a computer; this allows the interface chip to serve multiple functions:

- As a mass storage device (MSD), the interface chip provides drag and drop programming of a target MCU (*i.e. the Aretemis module*) and a USB bootloader for updating the firmware of the interface chip, itself.

- As a communications device class (CDC), the interface chip enumerates as a virtual com port and operates as a pass-through for serial communication between the target MCU to a PC.
- As a WEBUSB human interface device (HID), the interface chip functions as a bridge to the debug access port of the target MCU (*the Aretemis module*) and provides a CMSIS-DAP compliant debug channel.



Interface MCU connections. (Click to enlarge)
(Photo courtesy of Arm® Mbed™.)

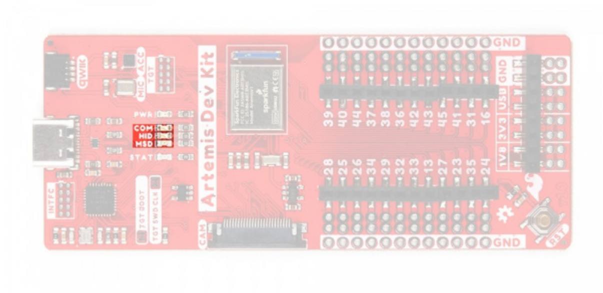
Do I need to install any drivers?

If your computer is running Windows 10, Mac OSX or Linux and using DAPLink firmware version 0240 or later then drivers are not needed. If your computer is running Windows 7 (*only*), you will need to install the serial port driver. The board running DAPLink must be connected to your computer to install this driver.

Further instructions and details on installing the serial port driver can be found in the Mbed™ OS documentation. (**As the Mbed™ OS documentation is continually updated, so are the links. Therefore, users may find it useful to look under the* ***Tutorials*** *>* ***Serial Communication*** *section of the (updated) Mbed™ OS documentation for the serial port driver information.*)

Status LEDs

Attached to the interface chip are three indicator LEDs. These LEDs will flash according to the device class the Artemis DK enumerates as.



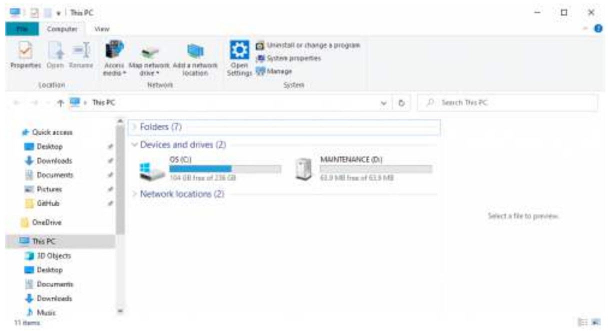
Artemis Development Kit `COM`*,* `HID`*, and* `MSD` *status LED indicators. (Click to enlarge)*

TOGGLE FIRMWARE DETAILS

Click the button *above to toggle the* ***additional information*** *about the Arm® Mbed™ DAPLink interface firmware.*

Maintenance Mode

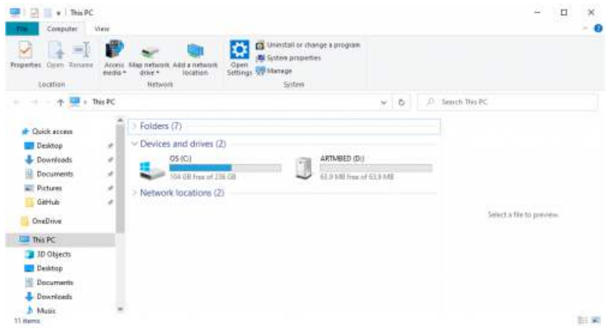
Also known as bootloader mode, this mode accesses the bootloader and is used to update the interface firmware on the MKL26Z128VFM4, without using the JTAG pins. To enter maintenance mode, users need to hold down the `RESET` button, while plugging in the Artemis development kit into the computer. The board will enumerate as a mass storage device named `MAINTENANCE`.



The Artemis DK enumerating as a USB drive, named `MAINTENANCE`. *(Click to enlarge)*

Interface Mode

The interface mode is used to program the target MCU, the Artemis module. In this mode, the board will enumerate as a mass storage device, labeled `ARTEMIS`. Users can then drag and drop **.bin** or **.hex** files, which are written to the flash of the Artemis module.



The Artemis DK enumerating as a USB drive, named `ARTEMIS`. *(Click to enlarge)*

Virtual File System

The file system presented within the storage drive is virtual and not stored in flash memory, as it would on a USB flash disk. When a file is dropped onto the storage drive, the DAPLink firmware streams the data through the debug probe to program the target MCU; which is why the drive ejects itself after new files are written. Below are the virtual files for the Artemis DK:

- `DETAILS.TXT` : This file contains diagnostic information and details on the version of DAPLink installed on the interface chip.
- `ARTEMIS.HTML` : This is a link to SparkFun's Artemis module webpage, to help users get started.
- `FAIL.TXT` : This file may appear, after a flashing error occurs; it will contain details on the cause of failure. Users can reference error.c file from the DAPLink repository to help diagnose the errors.

Why DAPLink? The reason for using this programming interface method was to provide full support for the Arduino BLE library, on the Artemis module.

Specifically, Mbed™ was needed to implement the Bluetooth functionality; and although DAPLink wasn't required for Mbed™, it was required to verify the pull request of our implementation of the Bluetooth functionality into Mbed™ OS. For more information about the reason Mbed™ was required to implement the Bluetooth functionality in the Arduino IDE, check out this article from Arduino, "Why we chose to build the Arduino Nano 33 BLE core on Mbed OS."

Reset Button

This button serves two functions: to reset the program running on the Artemis module and to enter into maintenance mode, for updating the firmware on the interface MCU.

Livestream of the product demonstration with Mbed™.
Users may need to watch the video on YouTube.

Note: We do **NOT** recommend novices begin with this board. There are a lot of details and fundamental knowledge involved with the use of this board. The content of this hardware guide alone can be daunting; not to mention, the three software development guides associated with this board, which can also be overwhelming for a first-time user.

Those who have become familiar with programming and electronic hardware will have an easier time developing with the Artemis DK as various components of this tutorial are geared towards the *professional* software developer.

Required Materials

To get started, users will need a few items. Now some users may have a few of these items, feel free to modify your cart accordingly.

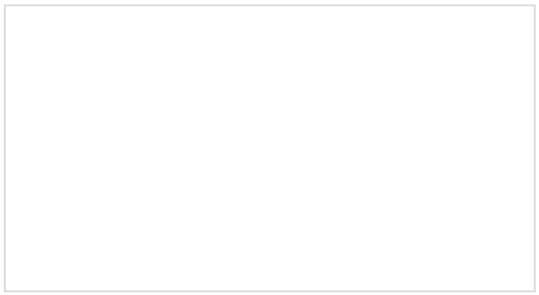
- SparkFun Artemis Development Kit or SparkFun Artemis Development Kit (with Camera)
- USB 3.1 Cable A to C - 3 Foot - The USB interface serves two purposes: it powers the board and allows you to upload programs to it. (**If your computer doesn't provide a USB-A slot, then you will need to choose an appropriate cable or purchase an adapter as well.*)
- Himax CMOS Imaging Camera - HM01B0 - The camera is optional, but is recommended for users interested in visual recognition applications. (** The camera is included in the SparkFun Artemis Development Kit (with Camera).*)
- Computer with the an operating system (OS) that is compatible with all the software installation requirements.

With the latest Artemis DK, board, we now offer full Bluetooth support within the Arduino IDE and development with Mbed™ OS. While we have worked tirelessly to get the Artemis module supported in the next Mbed™ OS release, the next release isn't slated until after the Artemis DK becomes available to the public. Therefore, this post will provide users with a jump start for developing with Mbed™ Studio, prior to the next release (in a beta of sorts), by utilizing our fork of Mbed™ OS.

board definition using the Arduino Board Manager. We will also go over manually installing third-party cores, such as the board definitions required for many of the SparkFun development boards.

Artemis Development with the Arduino IDE

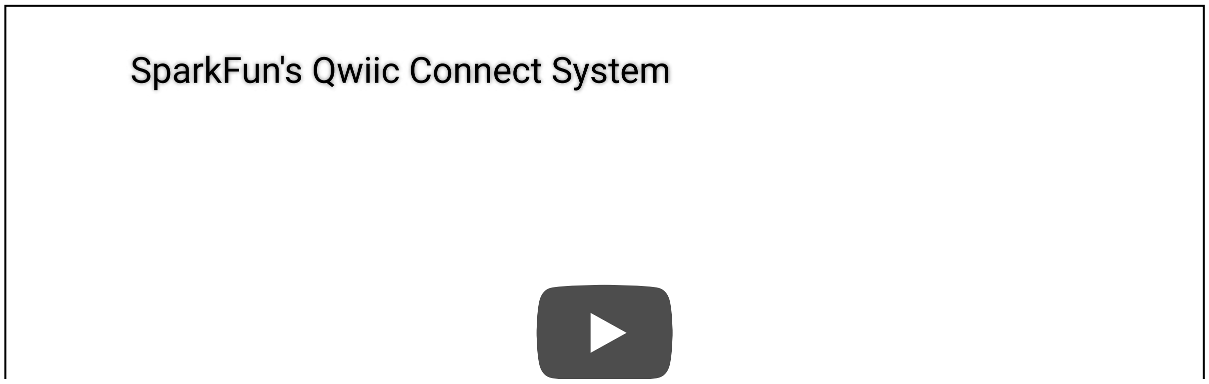
This is an in-depth guide on developing in the Arduino IDE for the Artemis module and any Artemis microcontroller development board. Inside, users will find setup instructions and simple examples from blinking an LED and taking ADC measurements; to more complex features like BLE and I2C. All the information within will provide a foundation for users to "Start Something" with the the SparkFun Artemis module!



Installing Arduino IDE

A step-by-step guide to installing and testing the Arduino software on Windows, Mac, and Linux.

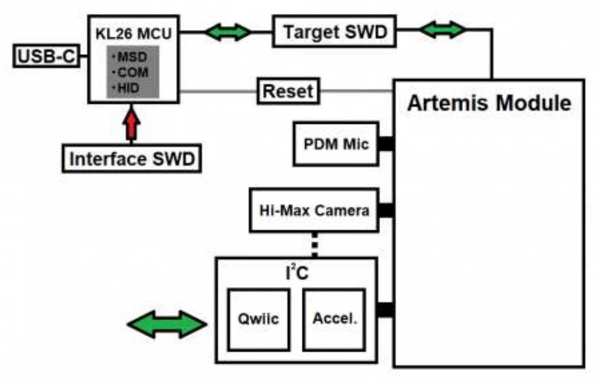
One of the new, advanced features of the board is that it takes advantage of the Qwiic connect system. We recommend familiarizing yourself with the **Logic Levels** and **I^{2}C** tutorials. Click on the banner above to learn more about Qwiic products.



Hardware Overview

Board Operation /Functionality

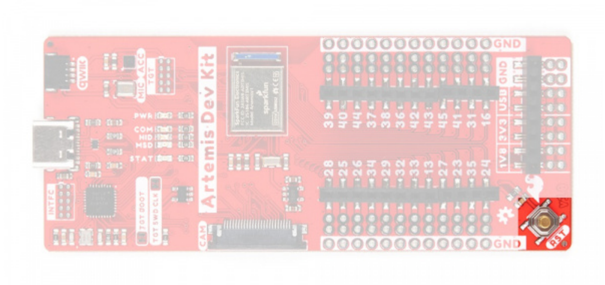
The functionality of this board can be broken up into 3 different systems: the USB interface, microcontroller (MCU), and the sensors and input/output (I/O) pins.



A functional block diagram for the Artemis Development Kit. (Click to enlarge)

1. **MCU:** The Artemis module serves as the brains of the board.
    - All the pins on the Artemis module are utilized either by the sensors, camera connector, or I/O pins.
2. **Sensors and I/O Pins:** There are two built-in sensors, with the option to add a camera, a Qwiic connector, and the breakout pins are relatively straight forward in their operation.
    - The first sensor is the SPH0641LM4H-1 PDM microphone, that can be used for speach recognition.
    - The second sensor is the LIS2DH12TR MEMS accelerometer, attached to the *"Qwiic"* I^{2}C bus.
    - The third *psuedo*-sensor is the camera connector, where an optional HM01B0 Himax camera can be attached.
        - The I^{2}C pin connections for the camera utilize the same bus as the *"Qwiic"* I^{2}C bus.
    - Any other Qwiic sensors attach to the same *"Qwiic"* I^{2}C bus.
    - The breakout pins all have their own dedicated connections to pins on the Artemis module. (i.e. The I/O pins do not share any connections to the sensors listed above.)
3. **USB Interface:** The USB interface IC is the heart of the board's operation, which provides a link between the USB port and the Artemis module.
    - Utilizing the provided DAPLink firmware, the IC enumerates as a composite device, with the following device classes:
        - Mass Storage Device (MSD): Used to provide drag and drop programming.
        - Human Interface Device (HID): Used for the debugging interface.
        - Communication Port (COM): Used to provide a serial communication UART between the Artemis and the USB connection.

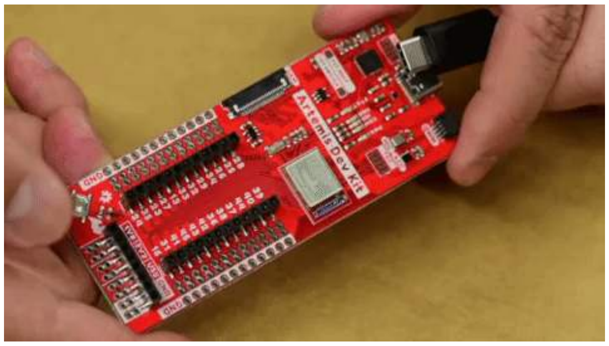
Board Dimensions



Reset button on the Artemis Development Kit. (Click to enlarge)

Maintenance Mode

As mentioned earlier, maintenance mode is used to update the firmware on the MKL26Z128VFM4, without using the JTAG pins. To enter maintenance mode, users need to hold down the `RESET` button, while plugging in the Artemis development kit into the computer. The board will enumerate as a mass storage device named `MAINTENANCE`. Users can then drag and drop in the updated firmware **.hex** file.

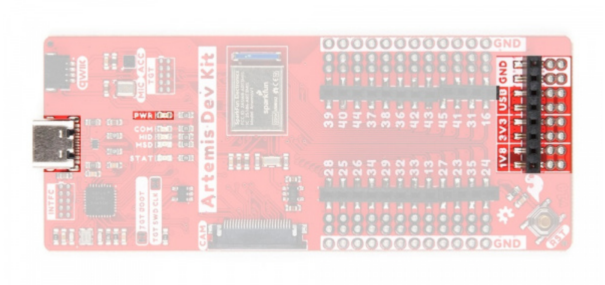


Entering `MAINTENANCE` mode. (Click to enlarge)

Power

The Artemis Development Kit only requires 3.3V to power the board. However, the simplest method to power the board is with the USB-C connector. The power header consists of voltage supply pins, which are traditionally **used as power sources** for other peripheral hardware (like LEDs, potentiometers, and other circuits).

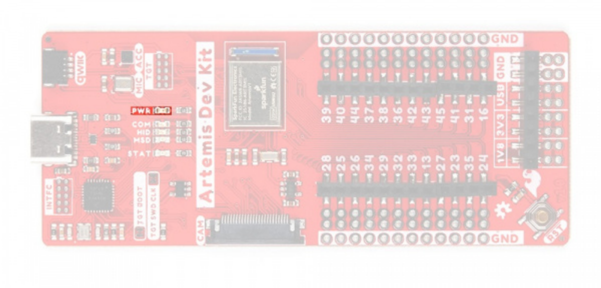
- **1.8V** - A regulated 1.8V voltage source.
    - Regulated from the 3.3V power.
    - Used to power the camera connector.
- **3.3V** - A regulated 3.3V voltage source.
    - Regulated from the USB 5V power.
    - Used to power the interface chip, Artemis module, Qwiic I^{2}C bus (*including the accelerometer and camera connector*), and PDM microphone.
- **USB** - The input voltage from the USB-C connector, usually 5V.
- **GND** - The common ground or the 0V reference for the voltage supplies.



Artemis Development Kit power connections. (Click to enlarge)

Power Status LED

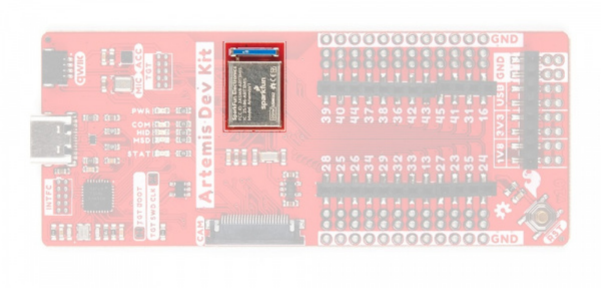
The `POWER` LED will light up once **3.3V** is supplied to the board; however, for most users, it will light up when **5V** is supplied through the USB connection.



Artemis Development Kit `POWER` status LED indicator. (Click to enlarge)

Artemis Module

The Artemis module is the primary microcontoller on the Artemis Development Kit.



Artemis module on the Artemis Development Kit. (Click to enlarge)

For details on the Artemis module, users should refer to the Designing with the SparkFun Artemis hookup guide. Additionally, users can reference the following resources for more technical information:

Designing with the SparkFun Artemis

JUNE 20, 2019

Let's chat about layout and design considerations when using the Artemis module.

Hardware Information:

- Artemis Integration Guide (PDF)
- Ambiq MCU Product Page
    - Apollo3 DataSheet (PDF)
    - Pin Function Map (PDF)
- Artemis GitHub Repository
    - Artemis Bootloader GitHub

Development Platforms:

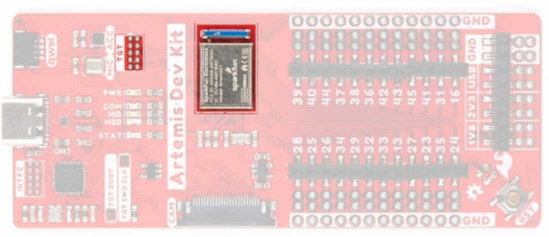
- SparkFun Ambiq Apollo3 Arduino Core
- Mbed™ OS
- AmbiqSDK
- pyOCD

Note: While most users will utilize the interface chip for drag-and-drop programming, the Artemis module can also be programmed through its JTAG or SWD pins. This might is useful for individuals developing and testing firmware that would be flashed directly onto the Artemis module, such as in production for commercial applications. For more details on programming, please check out our ARM Programming tutorial

ARM Programming

MAY 23, 2019

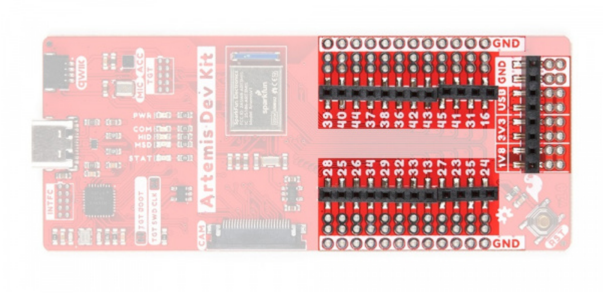
How to program SAMD21 or SAMD51 boards (or other ARM processors).



Artemis module and its JTAG pins on the Artemis Development Kit. (Click to enlarge)

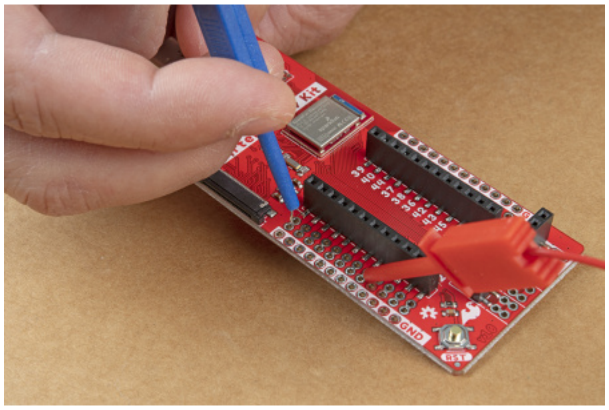
Breakout Pin Connections

The pins from the Artemis module are broken out into logical connections and available power connections are also broken out.



Artemis Development Kit breakout pins and headers. (Click to enlarge)

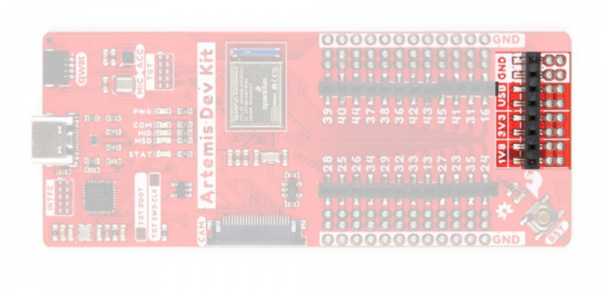
Note: The layout of the PTH connections for the breakout pins include both a .1" pitch spacing for headers and a .08" pitch spacing for IC-hooks, used by most logic analyzers.



Connecting IC hooks to the Artemis Development Kit. (Click to enlarge)

Power Connections

The power header and pins aren't really **I/O** (Input/Output) connections for the microcontroller; however, they are pertinent to the board. (**For details on the pin connections to the peripherals, check out the next section.*)



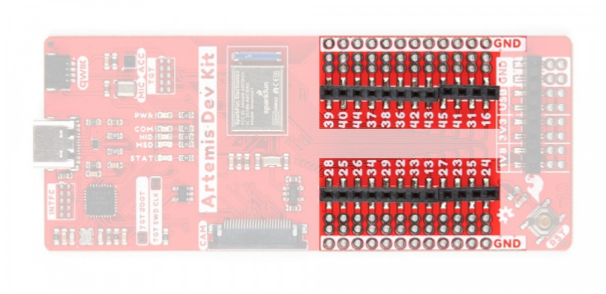
Artemis Development Kit power pins. (Click to enlarge)

The power I/O mostly consists of voltage supply pins. These pins are traditionally **used as power sources** for other pieces of hardware (like LEDs, potentiometers, and other circuits).

- **1.8V** - A regulated 1.8V voltage source.
- **3.3V** - A regulated 3.3V voltage source.
- **USB** - The input voltage from the USB-C connector, usually 5V.
- **GND** - The common ground or the 0V reference for the voltage supplies.

I/O Pins

There are **24 I/O pins** broken out on this board, which can be used as digital **inputs** to or **outputs** from the Artemis module.



I/O breakout pins and headers. (Click to enlarge)

All of the Artemis Development Kit pins are broken out to 0.1" spaced female headers (i.e. connectors). There are also two rows of breakout pins with .1" pitch spacing for headers; and a .08" pitch spacing to clip on IC-hooks, used by most logic analyzers. It is best practice to define the **`pinMode()`** *(link)* in the setup of each **sketch** (programs written in the Arduino IDE) for the pins used.

Input

When configured properly, an **input** pin will be looking for a **HIGH** or **LOW** state. **Input** pins are **High Impedance** and takes very little current to move the input pin from one state to another.

Output

When configured as an **output** the pin will be at a **HIGH** or **LOW** voltage. **Output** pins are **Low Impedance**: This means that they can provide a relatively substantial amount of current to other circuits.

Note: It should be noted that there are electrical limitations to the amount of current that the Artemis module can sink or source. For more details, check out the **Absolute Maximum Ratings** section of the **Electrical Characteristics** on page 774 of the Apollo3 datasheet.

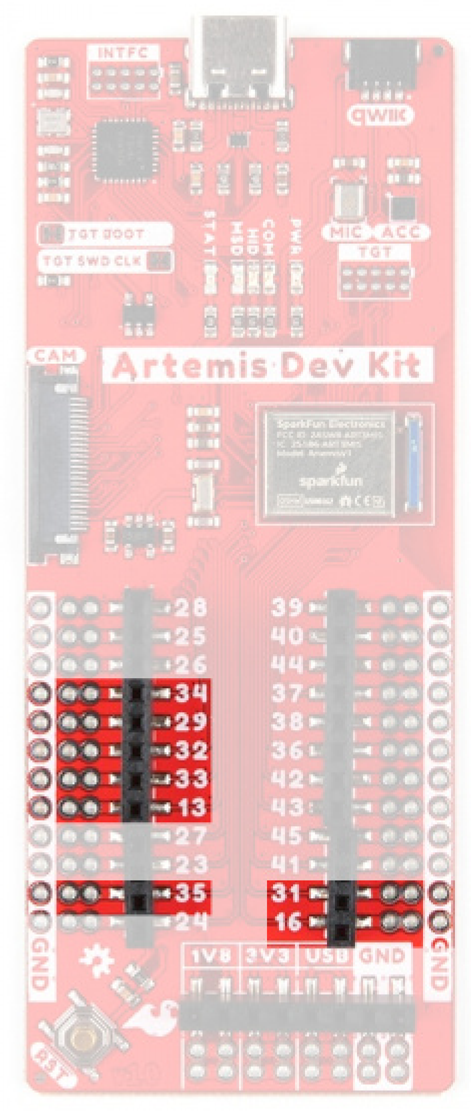
Additional Functions

There are several pins that have special functionality in addition to general **digital I/O**. These pins and their additional functions are listed in the tabs below. For more technical specifications on the **I/O** pins, you can refer to the Apollo 3 datasheet and Artemis Integration Guide.

ANALOG INPUT | PWM OUTPUT | SERIAL COMM. | SPI | I^{2}C

Analog Input Pins

The Artemis module offers a **14-bit ADC** input for eight of the breakout I/O pins. This functionality is accessed in the Arduino IDE using the `analogRead(pin)` function. *(*There is also a quick reference table for the available pin functionalities, labeled on the back of the board.)*



Analog input pins on the Artemis Development Kit. (Click to enlarge)

Note: By default, in the Arduino IDE, `analogRead()` returns a 10-bit value. To change the resolution of the value returned by the `analogRead()` function, use the `analogReadResolution(bits)` function.

Note: To learn more about analog vs. digital signals, check out this great tutorial.

Analog vs. Digital

JULY 18, 2013

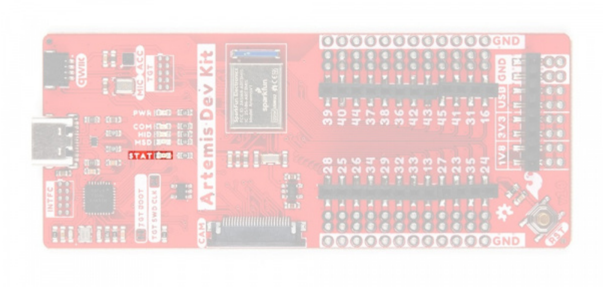
This tutorial covers the concept of analog and digital signals, as they relate to electronics.

Pin Connections for Peripheral Devices

The following peripherals have dedicated pin connections.

STAT LED

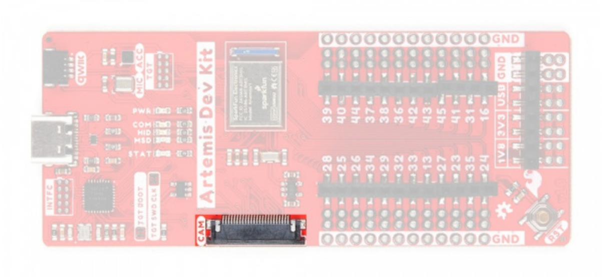
The `STAT` LED is typically used as a test LED to make sure that a board is working or for basic debugging. This indicator is connected to pin `23`, which can also be referenced as `D23` or `LED1` in the Arduino IDE (**use `LED1` in Mbed™ OS*).



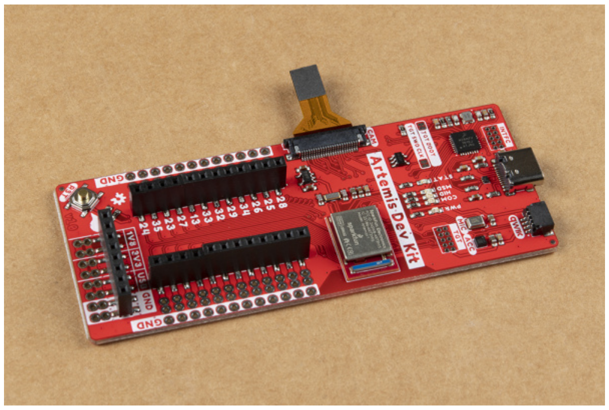
`STAT` LED indicator for the Artemis Development Kit. (Click to enlarge)

Camera Connector

There is a 24-pin (FPC) camera connector intended for a Himax (HM01B0) camera, which can be used for visual recognition applications. Technical specifications for the camera can be found in the datasheet. The pin connections for the camera connector are detailed in the table below. (**Make sure to attach the camera in the correct orientation, as shown below.*)



Camera connector on the Artemis Development Kit. (Click to enlarge)



Camera attached to the Artemis Development Kit. (Click to enlarge)

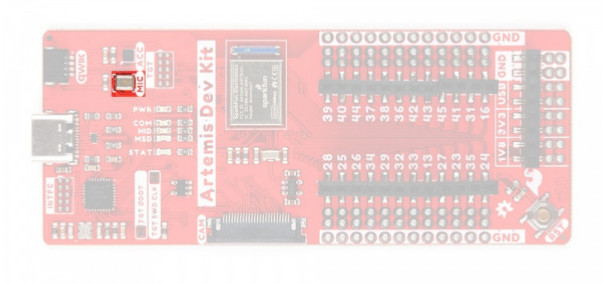
Pin Number *(Camera Connector)*	**Connection Name**	**Pad Number** *(Artemis Module)*
4	AVDD	**3.3V**
10	IOVDD	**1.8V**
2	GND	**GND**
6	FVLD VSYNC	D15
16	LVLD VSYNC	D17
8	MCLK	D18
17	PCLK	D19
7	TRIG	D14
18	INT	D10
21	SCL	D8
22	SDA	D9
14	D0	D0
3	D1	D1
9	D2	D2
5	D3	D3
12	D4	D4
20	D5	D5

19	D6	D6
13	D7	D7

Note: The camera image capture operation is only supported within the AmbiqSDK. For more details, please check out our Using SparkFun Edge Board with Ambiq Apollo3 SDK tutorial

PDM Mic

There is a SPH0641LM4H-1 PDM microphone, which can be used for voice recognition applications. Technical specifications for the microphone can be found in the datasheet. The pin connections for the microphone are detailed in the table below.



PDM microphone on the Artemis Development Kit. (Click to enlarge)

Connection	CLK	DATA
Pad Number ***(Artemis Module)***	AD12	AD11

Note: In the Arduino IDE, a library is not required for users to utilize the PDM microphone. Additionally, there are exanples built-in with the installation of the board definitions. Users can access a list of examples from the **File** > **Examples** > **PDM** *(under **Examples for the Artemis Dev Kit**)* drop down menu.

Note: Currently, with the latest revision to the Arduino core, the examples will not compile due problems with the `math` library. For more details, users can track the issue in the GitHub repository.

Primary I^{2}C Bus

An accelerometer and a Qwiic connector are attached to the primary I^{2}C bus. The primary I^{2}C bus for this board utilizes the pin connections, detailed in the table below:

Connection	VDD	GND	SCL	SDA
Pad Number ***(Artemis Module)***	**3.3V**	**GND**	D8	D9

The Artemis DK was designed to be utilized with three primary development environments. The Apollo3 MCU in the Artemis module is now fully supported in the Arduino IDE, including it's BLE functionality. In order to implement compatibility with the Arduino's built-in BLE library, we needed to

New!

Artemis Development with the Arduino IDE

SEPTEMBER 10, 2020

This is an in-depth guide on developing in the Arduino IDE for the Artemis module and any Artemis microcontroller development board. Inside, users will find setup instructions and simple examples from blinking an LED and taking ADC measurements; to more complex features like BLE and I2C. All the information within will provide a foundation for users to "Start Something" with the the SparkFun Artemis module!

Note: The tutorial for Mbed™ is in progress, pending the next Mbed™ release. In the meantime, users can check out the tutorial below to *beta* test our fork of Mbed™ OS.

New!

Artemis Development on Arm® Mbed™ OS (Beta)

SEPTEMBER 10, 2020

With the latest Artemis DK, board, we now offer full Bluetooth support within the Arduino IDE and development with Mbed™ OS. While we have worked tirelessly to get the Artemis module supported in the next Mbed™ OS release, the next release isn't slated until after the Artemis DK becomes available to the public. Therefore, this post will provide users with a jump start for developing with Mbed™ Studio, prior to the next release (in a beta of sorts), by utilizing our fork of Mbed™ OS.

Using SparkFun Edge Board with Ambiq Apollo3 SDK

MARCH 28, 2019

We will demonstrate how to get started with your SparkFun Edge Board by setting up the toolchain on your computer, examining an example program, and using the serial uploader tool to flash the chip.

Installation Requirements

Depending on the application (i.e. development environment) users want to work with, there may be additional software required. Below, is a list of additional software that we recommend installing for each development environment:

- Arduino IDE
  - Install Arduino IDE
    - Install the **SparkFun Apollo3 Boards** Board Definitions
    - Install the **ArduinoBLE** Arduino Library
    - Install the **SparkFun LIS2DH12** Arduino Library
- Ambiq SDK
  - Python 3 (*required for install*)
  - GCC Arm® Toolchain (*version: **8-2018-q4-major***)
    - (compiling source code)
  - Make (building projects)

Bash (Windows Only)

Although a Bash isn't required, it is a user friendly interface to pull GitHub repositories and navigate through a file system. For those that are unfamiliar, Bash is a Unix Shell (i.e. a command-line interpreter or shell that provides a command line user interface for Unix-like operating systems).

On most Mac and Linux computers, the Terminal program uses bash by default; otherwise, if you are accessing the shell, it is bash. For Windows computers, users will need to install a Bash shell emulator that has access to your file system.

We highly recommend Git Bash. For more information, please check out the following resources to get started with installing and utilizing Git Bash.

- Gitfor Windows
- GitHub Repository
- FAQ

Adding File Paths: To utilize `bash` , users will need to add it to the operating system's file path.

Make

Make is an automated build tool - you give it information about source locations and dependencies and it will automatically call a sequence of commands to compile or re-compile your project based only on the changed files. This can speed up compilation significantly in large projects. Make is a time-tested tool.

Adding File Paths: To utilize `make` , users will need to add it to the operating system's file path.

Python 3

Users will need to install Python 3 to utilize the `pyOCD` debugging package; as well as, to install Mbed™ OS and AmbiqSDK development environments. For more information, please check out the following resources from the Python Software Foundation to get started with installing and utilizing Python 3.

- Python Software Foundation: Home Page
- Software Download Page
- Documentation Page
- Beginner's Guide

Python Package Installer

`pip3` is the package installer for Python 3. It should be natively installed with Python 3. For more information Python Package Installer, check out the online documentation.

GNU Arm® Embedded Toolchain

The GNU Arm® Embedded Toolchain is an open-source suite of tools for C, C++ and assembly programming. Designed to program target Arm® Cortex®-A, Arm® Cortex®-M, and Arm® Cortex®-R processors, the toolchain includes the GNU Compiler (GCC) and and other tools necessary for bare-metal software development. The GNU Arm® Embedded Toolchain is provided by Arm® for embedded software development on Windows, Linux, and Mac OS X operating systems.

Note: We recommend installing version **8-2018-q4-major** of the GNU Arm® Embedded Toolchain.

Adding File Paths: To utilize `gcc` , users will need to add it to the operating system's file path.

Arduino IDE

Note: For first-time users, who have never programmed before and are looking to use the Arduino IDE, we recommend beginning with the SparkFun Inventor's Kit (SIK), which includes a simpler board like the Arduino Uno or SparkFun RedBoard and is designed to help users get started programming with the Arduino IDE.

Most users will be familiar with the Arduino IDE and it's use. As a point of reference for professional developers who aren't aware, the Arduino IDE is an open-source development environment, written in Java, that makes it easy to write code and upload it to a supported board. For more details, feel free to check out the Arduino website.

To get started with the Arduino IDE, check out the following tutorials:

Installing an Arduino Library

How do I install a custom Arduino library? It's easy! This tutorial will go over how to install an Arduino library using the Arduino Library Manager. For libraries not linked with the Arduino IDE, we will also go over manually installing an Arduino library.

What is an Arduino?

What is this 'Arduino' thing anyway? This tutorials dives into what an Arduino is and along with Arduino projects and widgets.

Installing Arduino IDE

A step-by-step guide to installing and testing the Arduino software on Windows, Mac, and Linux.

New!

Installing Board Definitions in the Arduino IDE

How do I install a custom Arduino board/core? It's easy! This tutorial will go over how to install an Arduino board definition using the Arduino Board Manager. We will also go over manually installing third-party cores, such as the board definitions required for many of the SparkFun development boards.

Software Dependencies

- Install the **SparkFun Apollo3 Boards** Board Definitions in the Arduino IDE

New!

Installing Board Definitions in the Arduino IDE

SEPTEMBER 9, 2020

How do I install a custom Arduino board/core? It's easy! This tutorial will go over how to install an Arduino board definition using the Arduino Board Manager. We will also go over manually installing third-party cores, such as the board definitions required for many of the SparkFun development boards.

Note: Users will need to install the *board definitions* for our development boards with an Artemis module.

For more instructions, users can follow this tutorial on Installing Additional Cores provided by Arduino. Users will also the `.json` file for the SparkFun Ambiq Apollo3 Arduino Core:

```
https://raw.githubusercontent.com/sparkfun/Arduino_Apollo3/master/package_sparkfun_apollo3_index.json
```

- Install the associated Arduino libraries for the board's features:
  - The **ArduinoBLE** Arduino Library
  - The **SparkFun LIS2DH12** Arduino Library

Installing an Arduino Library

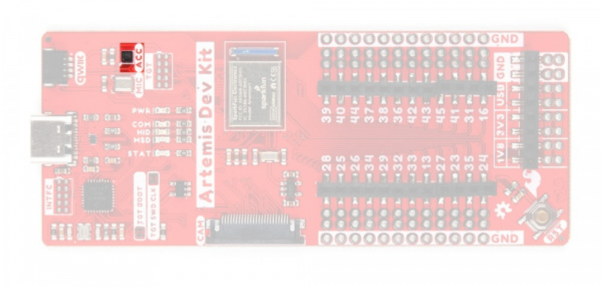
JANUARY 11, 2013

How do I install a custom Arduino library? It's easy! This tutorial will go over how to install an Arduino library using the Arduino Library Manager. For libraries not linked with the Arduino IDE, we will also go over manually installing an Arduino library.

AmbiqSuite SDK

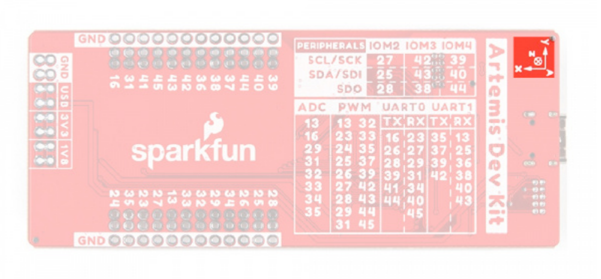
Accelerometer

There is an LIS2DH12 3-axis accelerometer, which can be used for gesture recognition. Technical specifications for the accelerometer can be found in the datasheet.



Accelerometer on the Artemis Development Kit. (Click to enlarge)

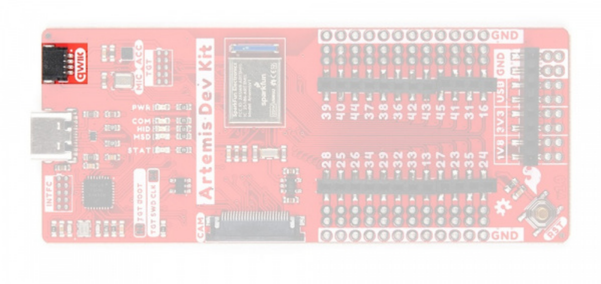
Note: In the Arduino IDE, users can utilize the SparkFun LIS2DH12 Arduino Library. Also, the orientation of the sensor's axes is labeled on the back of the board.



Silk marking the orientation of the accelerometer's axes, in reference to the Artemis Development Kit. (Click to enlarge)

Qwiic Connector

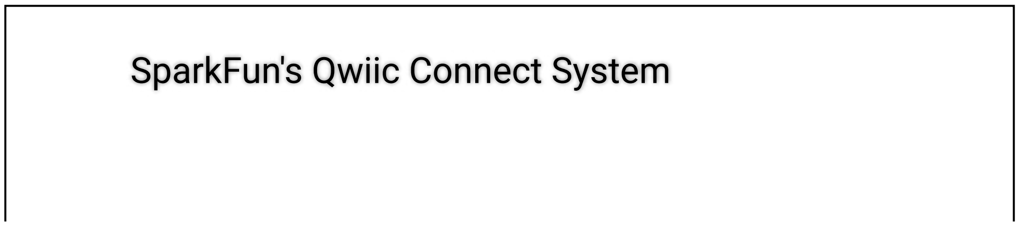
A Qwiic connector is provided for users to seamlessly integrate with SparkFun's Qwiic Ecosystem.



Qwiic connector on the Artemis Development Kit. (Click to enlarge)

What is Qwiic?

The Qwiic system is intended a quick, hassle-free cabling/connector system for I^{2}C devices. Qwiic is actually a play on words between "quick" and I^{2}C or "iic".

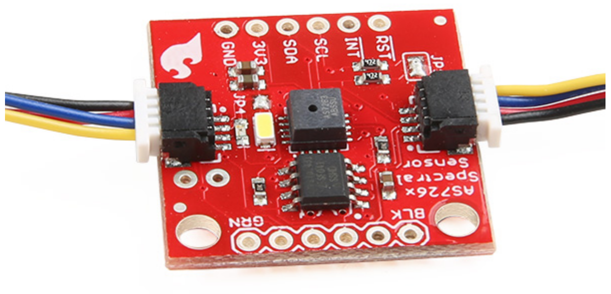


Features of the Qwiic System

NO SOLDERING | POLARIZED CONNECTOR | DAISY CHAIN

Keep your soldering iron at bay.

Cables plug easily between boards making quick work of setting up a new prototype. We currently offer three different lengths of Qwiic cables as well as a breadboard friendly cable to connect any Qwiic enabled board to anything else. Initially you may need to solder headers onto the shield to connect your platform to the Qwiic system but once that's done it's plug and go!



Qwiic cables connected to Spectral Sensor Breakout

Software Overview

Drivers

As mentioned in the hardware section, a driver is not required for any of the recommended operating systems (Mac OSX, Linux, Windows 10). However, there is a serial port driver available for Windows 7 (only), which is provided by Mbed™. We will not be providing support for Windows 7; a forum is available from Mbed™, for users that run into issues with their driver.

Integrated Development Environments (IDEs)

Note: These instructions are subject to change in the near future; pending the adoption of the Artemis module into Mbed™ OS.

The Ambiq Suite SDK is a collection of software enablement for the Apollo, Apollo2, Apollo2-Blue, and Apollo3-Blue MCUs. The SDK includes a hardware abstraction layer (HAL), device drivers, and example applications to speed the understanding of the operation of the MCUs. Third party software including Arm®'s Cordio BLE Host stack and FreeRTOS v10.1.1 and v9.0 are distributed along with debugging tools and other support. This is the most 'raw' option - you can write C or C++ code that runs pretty close to 'bare metal' in a 'professional' software development kit.

For more information on the Apollo3 MCU and the AmbiqSDK, check out the resources below:

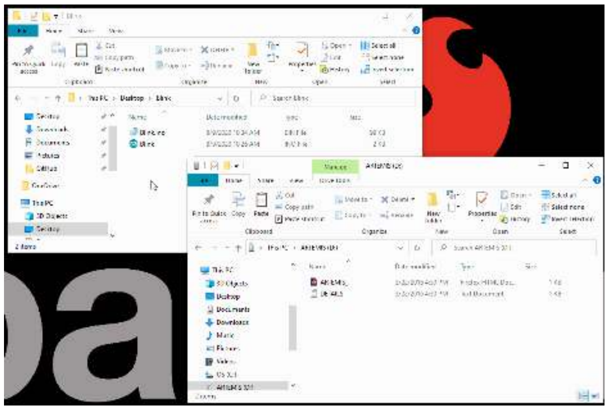
- Apollo MCU Product Page
    - Apollo 3 Blue Product Page *(used in Artemis module)*
- Ambiq Documentation
- AmbiqSuite SDK Getting Started Guide (v1.1 June 2017)
- AmbiqSDK maintained by SparkFun

Software Dependencies

- Python (uploading code to board)
- Make (building projects)
- GCC Arm® Toolchain (compiling source code)
- Git (sort of - you could just download a ZIP from github)

Programming the Board

The programming process for the Artemis DK is straight forward. Once users build or compile a binary file for their code with the Artemis DK as the target, the file can be dragged-and-dropped (or copied) into the `ARTEMIS` mass storage drive on the computer.



Programming the Artemis DK. (Click to enlarge)

Hardware Assembly

USB Programming

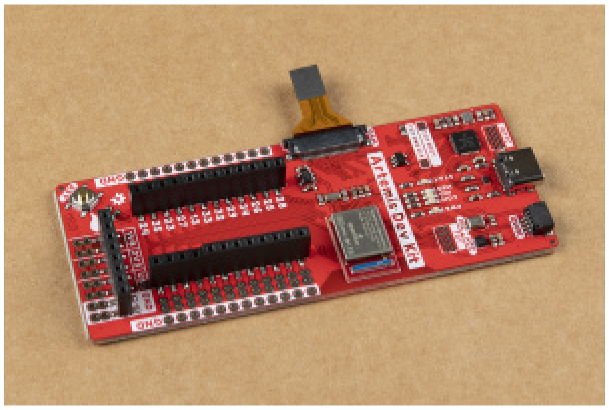
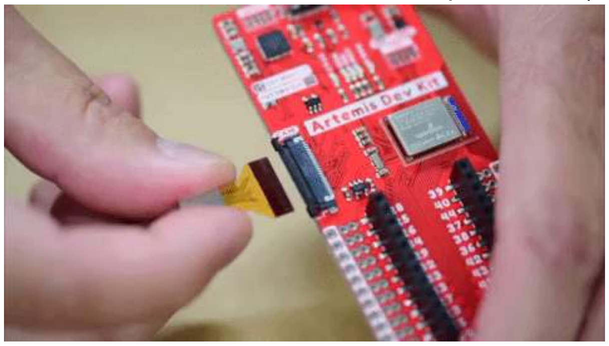
Most users will utilize the USB connection for the drag-and-drop programming interface, serial communication, and debugging. Users only need to plug their Artemis DK into a computer using a USB-C cable.

The Artemis Development Kit attached to a computer. The board is appears as a mass storage device, named `ARTEMIS` *, in the displayed window. (Click to enlarge)*

Camera

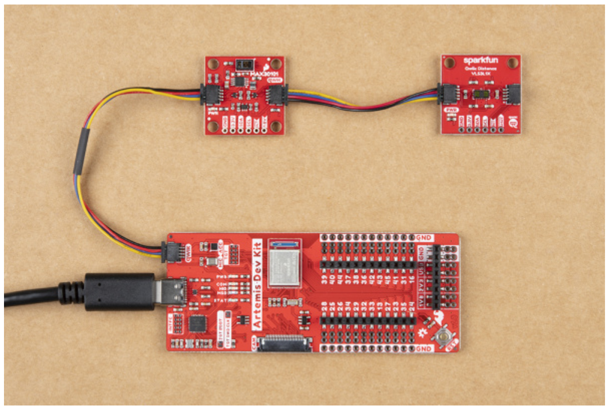
Attaching the Himax camera should be straight forward. However, steps and pictures are provided below as a reference for users. (**It should be noted that the proper technique for closing the camera connector, should be with two hands and with each finger on either side of the snap-in tab. Unfortunately, the process proved difficult to get a clean shot of and we had to improvise, as shown.*)

1. To attach the camera, carefully pop open the snap-in tab on the camera connector.
2. Seat the camera's ribbon cable, fully into the camera connector, with the lens pointing towards the bottom of thet board.
3. Close the snap-in tab from the sides, with both fingers.


The Himax camera attached to the Artemis Development Kit. (Click to enlarge)

Inserting the Himax camera into the camera connector. (Click to enlarge)

Qwiic Devices

The Qwiic system allows users to effortlessly prototype with a Qwiic compatible I^{2}C device without soldering. Users can attach any Qwiic compatible sensor or board, with just a Qwiic cable. (**The example below, is for demonstration purposes and is not pertinent to the board functionality or this tutorial.*)



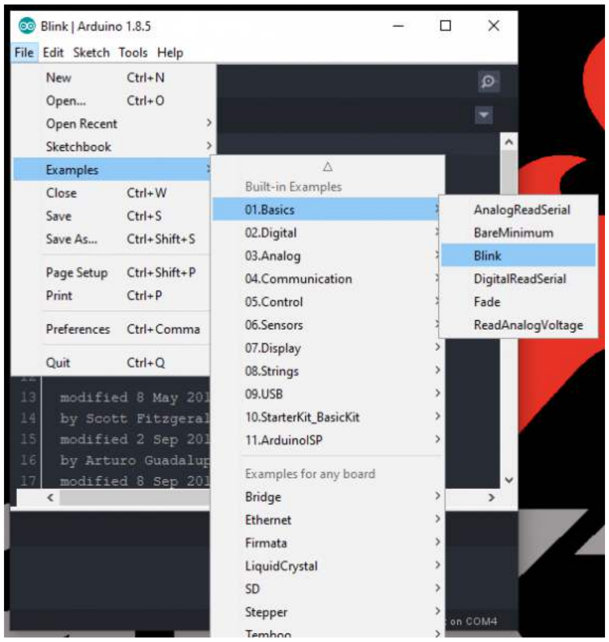
The Qwiic photodetector and distance sensors connected to the Artemis Development Kit.

Arduino Examples

Blink

With full Arduino support, users will be able to implement the built-in examples of the Arduino IDE on the Artemis DK. To begin, let's start with the most basic example: **Blink**.

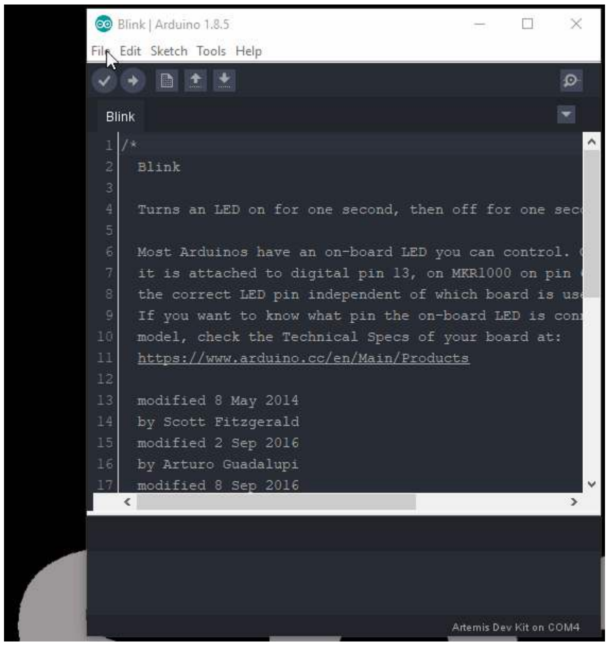
1. From the drop down file menu, pull up the **Blink** sketch:



```
  modified 8 May 201
  by Scott Fitzgeral
  modified 2 Sep 201
  by Arturo Guadalup
  modified 8 Sep 201
```

*Screen shot of the **Blink** example in the Arduino IDE. (Click to enlarge)*

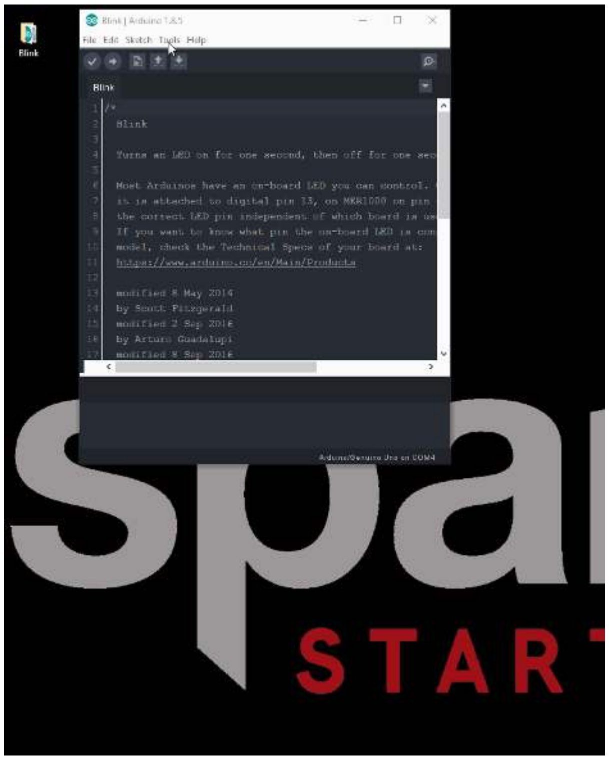
2. Since users will need the compiled binary code in order to program the Artemis DK, save this example somewhere that is easily accessible; the desktop is a good option. (**Remember to use **Save As**, so that you retain the original example within the Arduino IDE.*)



```
/*
  Blink

  Turns an LED on for one second, then off for one sec

  Most Arduinos have an on-board LED you can control. 
  it is attached to digital pin 13, on MKR1000 on pin 
  the correct LED pin independent of which board is us
  If you want to know what pin the on-board LED is con
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
```

***Blink** example being saved to the Desktop. (Click to enlarge)*

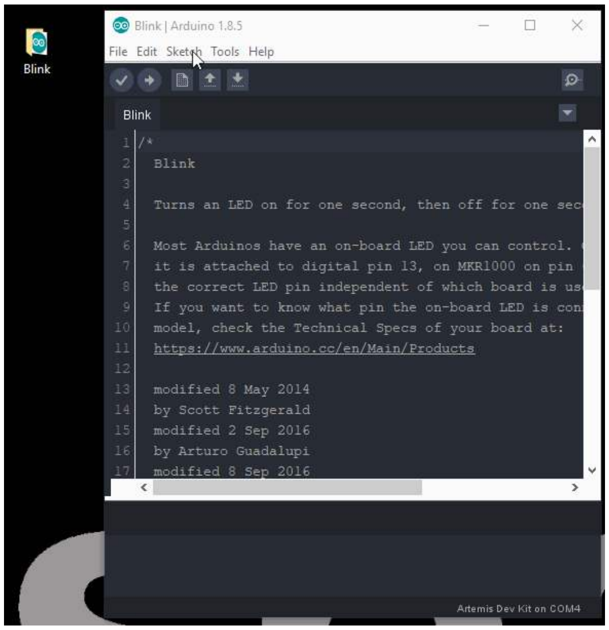
3. Next, you will want to select the **Artemis Dev Kit** board from the tools drop down menu: **Tools > Board: > Artemis Dev Kit**



```
/*
  Blink

  Turns an LED on for one second, then off for one sec

  Most Arduinos have an on-board LED you can control. 
  it is attached to digital pin 13, on MKR1000 on pin 
  the correct LED pin independent of which board is us
  If you want to know what pin the on-board LED is con
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
```

Selecting Artemis DK board definition in the Arduino IDE. (Click to enlarge)

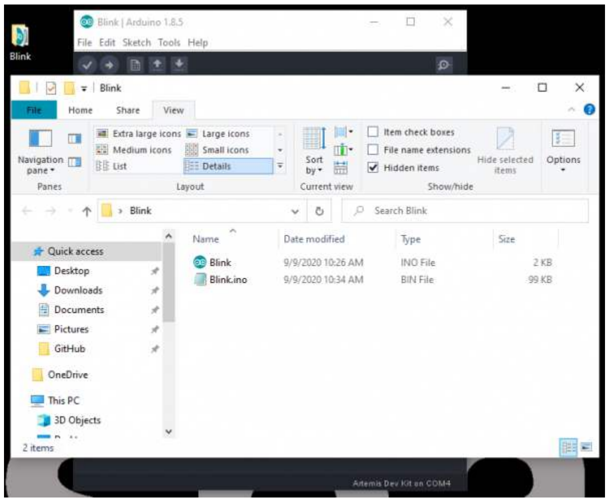
4. Finally, you will need to acquire the compiled binary code using the sketch drop down menu: **Sketch > Export compiled Binary** (`Ctrl + Alt + S`)



```
/*
  Blink

  Turns an LED on for one second, then off for one sec

  Most Arduinos have an on-board LED you can control.
  it is attached to digital pin 13, on MKR1000 on pin
  the correct LED pin independent of which board is us
  If you want to know what pin the on-board LED is con
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
```

Process for exporting the compiled binary from the Arduino IDE. (Click to enlarge)

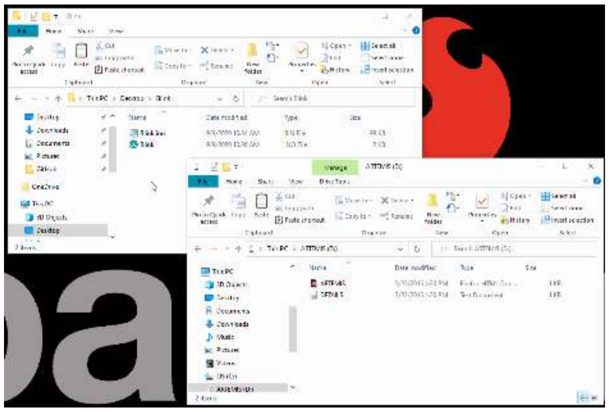
5. Now, the compiled binary will be located in the same location that the example was saved, from the previous step.



Name	Date modified	Type	Size
Blink	9/9/2020 10:26 AM	INO File	2 KB
Blink.ino	9/9/2020 10:34 AM	BIN File	99 KB

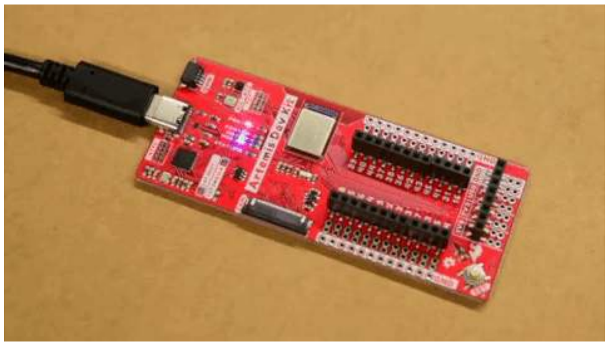
Screen shot of compiled binary in the example folder. (Click to enlarge)

6. All that is left is to drag and drop (or copy) the compiled binary file into the mass storage device for the Artemis DK. The interface chip will take care of the rest.



Programming the Artemis DK. (Click to enlarge)

7. The storage drive will automatically eject once the programming is complete. To verify that the board is working and that the programming was successful, users should see the `STAT` LED blinking in a ~1 sec. interval.


A demonstration of the `STAT` LED blinking. (Click to enlarge)

Hello World

In this example, users will *create* their own sketch, compile and export the binary file, program the Artemis DK, and view the serial communication in a terminal emulator. (**Most of the steps below are illustrated the procedure of the previous example.*)

1. Open a new sketch in the Arduino IDE from the drop down file menu: **File > New**.
2. Copy the example code below into the sketch:

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // print out "Hello World"
  Serial.println("Hello World");
  delay(500);
}
```

3. Save the sketch (**File > Save As...**) using the file drop down menu.
4. Select the then, export the compiled binary code (**Sketch > Export compiled Binary**) using the sketch drop down menu.
5. Drag and drop (or copy) the binary `.bin` file into the `ARTEMIS` mass storage drive. The storage drive will automatically eject once the programming is complete.
6. To verify that the board is working and that the programming was successful, open a terminal emulator, such as TeraTerm or the Serial Monitor in the Arduino IDE. Connect to the serial port for the Artemis DK; don't forget to set the baud rate to **9600 bps**.

Artemis Development with Arduino

JUNE 20, 2019

Get our powerful Artemis based boards (Artemis Nano, BlackBoard Artemis, and BlackBoard Artemis ATP) blinking in less than 5 minutes using the SparkFun Artemis Arduino Core!

Additionally, since the Arduino core for the Artemis module is built upon the Mbed™ OS, the API can also be utilized within the Arduino IDE. For more information, check out the Mbed™ software development tutorial.

New!

Artemis Development on Arm® Mbed™ OS (Beta)

SEPTEMBER 10, 2020

With the latest Artemis DK, board, we now offer full Bluetooth support within the Arduino IDE and development with Mbed™ OS. While we have worked tirelessly to get the Artemis module supported in the next Mbed™ OS release, the next release isn't slated until after the Artemis DK becomes available to the public. Therefore, this post will provide users with a jump start for developing with Mbed™ Studio, prior to the next release (in a beta of sorts), by utilizing our fork of Mbed™ OS.

Note: The tutorial for Mbed™ is in progress, pending the next Mbed™ release. In the meantime, users can check out the tutorial above for *beta* testing with our fork of Mbed™ OS.

Troubleshooting

Below, we have also included some additional troubleshooting tips for issues that you may come across with the Artemis Development Kit.

1. One of our employees compiled a great list of troubleshooting tips based on the most common customer issues. This is the perfect place to start.
2. For any Arduino IDE specific issues, we recommend starting with their troubleshooting guide.

If neither of the troubleshooting guides above were able to help, here are some tips you might have missed. (Most of this material is summarized from the tutorial.):

Are You Using a Recommended Computer OS?

This board is not compatible with the Arduino Web IDE. We do **NOT** recommend using a Chromebook, Netbook, tablet, phone, or the Arduino Web IDE in general. If you are here, try a **RECOMMENDED** operating system (refer to the notice in the Introduction).

My Board Isn't Working:

Every Artemis DK gets tested before getting packaged up. That being said, let's try a basic test to see if just the board is working. Disconnect everything that you have attached to the board; we just want to test the board.

1. **Inspect the board:**
   Check the board to make sure everything looks about right. Use the pictures on the product page to verify component placement or alignment, and bad solder joints, or damage.
2. **Power and check the status LEDs:**
   Using a known good USB-C cable, plug your board in to the computer. Do any of the status LEDs turn on (see Hardware Overview)?
    - New boards, out of the bag, will come programmed with a test sketch that blinks that status LED at a rate of approximately 2 blinks per second.
3. **Test a *Blink* sketch:**
   Try to program the blink sketch. Why blink? It is simple, known to work (from the example files), and you have an indicator LED.
    - Double check that you have the proper **Board** selected prior to exporting the compiled binary.
    - For boards that are already running the blink example, I recommend changing the timing parameters to check for a change in the board's response.

   Verify that you see the status LED blinking properly. If you are having trouble compiling or exporting the binary file, try using this pre-compiled binary file.

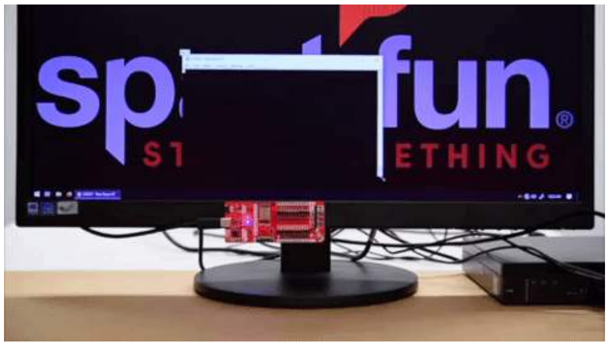
I Don't See My Board on a Serial/COM Port:

If you don't see your board as an available COM port on the Arduino IDE:

- Try to re-open the Arduino IDE.
- Check the **Device Manager** to verify that your computer recognizes the board.
- The issue might be related to your USB cable. Check that you are using a USB cable capable of data transfers. **Some cables only have the power pins connected for charging**. A good way to test this is to plug in a device to your USB cable (like a phone). If it doesn't show up as a device or drive, then try a new USB-C cable.
- This rarely happens, but it is easy to check. If you are using a USB 3.0 port (you will see a blue "*tongue*" in the USB jack or bad USB port, try a different USB port. You can also try to test the board on a different computer to double check for a hardware incompatibility (usually with expansion boards).

Additional Tips:

- If an input pin is read and that is floating (with nothing connected to it), you may see random data/states. In practice, it is useful to tie an input pin to a known state with a pull-up resistor (to VCC), or a pull-down resistor (to GND).
- The maximum current an I/O pin can source (provide positive current) or sink (provide negative current) is 50 mA (milliamps). You can power small sections of LED strips or small motors, but will run into issue with



A demonstration of the board's serial communication. (Click to enlarge)

Bluetooth: LED

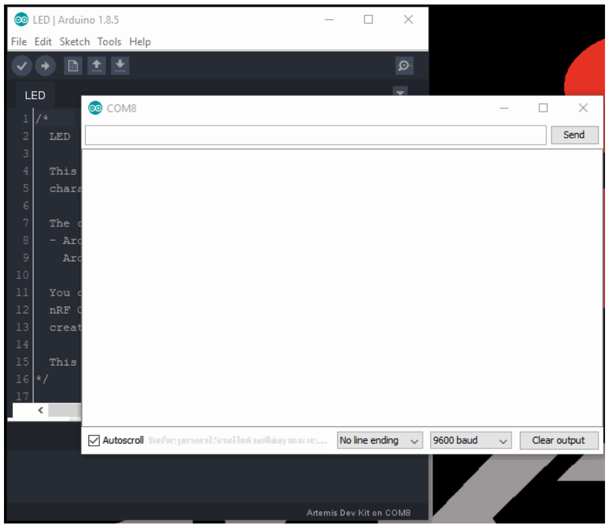
Note: Users will need to download the NRF Connect app onto their phone. We highly **recommend utilizing an Android phone** as we have had difficulties with device connections in the app for Apple phones.

As in the prior examples, these steps will follow a similar procedure.

1. From the drop down file menu, open the **LED** sketch: **File > Examples > ArduinoBLE > Peripheral > LED**.
2. Save the sketch (**File > Save As...**) using the file drop down menu.
3. Select the then, export the compiled binary code (**Sketch > Export compiled Binary**) using the sketch drop down menu.
4. Drag and drop (or copy) the binary `.bin` file into the `ARTEMIS` mass storage drive. The storage drive will automatically eject once the programming is complete.
5. To verify that the board is working and that the programming was successful, open a terminal emulator, such as TeraTerm or the Serial Monitor in the Arduino IDE. Connect to the serial port for the Artemis DK; don't forget to set the baud rate to **9600 bps**.
6. Click the reset button to begin the program. Users should see a `BLE LED Peripheral` message in the terminal emulator.
7. Follow the steps and video below to control the `STAT` LED on the Artemis DK.
   1. Unlock the phone and enable the Bluetooth.
   2. Open the NRF app.
   3. Click on the `Scan` button on the upper right corner.
   4. In the results, locate the device labeled `LED`. Click on the `CONNECT` button to the right of the device name.
      - In the terminal emulator, users should see a `Connected to central:` message followed by the mac address of the phone.
   5. Once connected, locate the `Unknown Service`. It will have the following service UUID: `19B10000-E8F2-537E-4F6C-D104768A1214`.
   6. Click on the service. Then, locate and click on the upload button (up arrow).
   7. A dialog box will open. We want to write a value, so locate the `New Value` text field, under the **Write value** drop-down menu option. Based on the control inputs to the LED, an `LED on` or `LED off` message will appear in the terminal emulator.
      - Enter `01` to turn the LED on. Then click the `SEND` button.
      - Enter `00` to turn the LED off. Then click the `SEND` button.

Artemis Dev Kit 3

Utilizing the example with the NRF phone app.



```
/*
  LED

  This
  chara

  The
  - Ar
    Ar

  You
  nRF
  creat

  This
*/
```

Serial output of the `LED` Bluetooth example. (Click to enlarge)

More Advanced Examples and Features

For more advanced examples and features of the Artemis DK with the Arduino IDE, check out the Arduino IDE software development tutorial.

high power devices.
- The pins on the Artemis DK use **3.3V** logic and aren't directly compatible with **5V** devices.

Resources and Going Further

For more on the Artemis Development Kit, check out the links below:

- Schematic (PDF)
- Eagle Files (ZIP)
- Board Dimensions (PDF)
- Software Guides:
  - Artemis Development with the Arduino IDE
  - Artemis Development with Arm® Mbed™ OS (Beta)
  - Artemis Development with Arm® Mbed™ OS (Pending release)
  - Artemis Development with the AmbiqSDK
- Hardware Component Information:
  - Artemis Module:
    - Apollo3 Datasheet (PDF)
    - Apollo3 Pad Mapping (PDF)
    - Artemis Integration Guide (PDF)
    - Ambiq
  - Interface Chip:
    - MKL26Z128VFM4 Datasheet (PDF)
    - MKL26P64M48SF5 Reference Manual (PDF)
  - Himax Camera (HM01B0) Datasheet (PDF)
  - PDM Microphone (SPH0641LM4H-1) Datasheet (PDF)
  - Accelerometer (LIS2DH12) Datasheet (PDF)
  - SparkFun Qwiic Connect System
- GitHub Hardware Repository
- GitHub Repository for Apollo3 Board Support Packages (BSP)
  - DAPLink Bootloader: `kl26z_artemis_dk_if.bin` (BIN)
  - DAPLink Interface Firmware: `kl26z_bl.bin` (BIN)
- Development Platforms for Artemis Module:
  - SparkFun Ambiq Apollo3 Arduino Core
    - `.json` file needed for the SparkFun Ambiq Apollo3 Arduino Core: `https://raw.githubusercontent.com/sparkfun/Arduino_Apollo3/master/package_sparkfun_apol`
  - Mbed™ OS (Ported)
  - AmbiqSDK (Ported)
  - pyOCD (Ported)
- SFE Product Showcase
- Product Live Stream with Mbed™

Need some inspiration for your next project? Check out some of these related tutorials:

ARDUINO IDE | BOARDS & SHIELDS | BOARD FUNCTIONALITY | SOFTWARE DEVELOPMENT GUIDES

SparkFun Tutorials

Installing an Arduino Library
How do I install a custom Arduino library? It's easy! This tutorial will go over how to install an Arduino library using the Arduino Library Manager. For libraries not linked with the Arduino IDE, we will also go over manually installing an Arduino library.

What is an Arduino?
What is this 'Arduino' thing anyway? This tutorials dives into what an Arduino is and along with Arduino projects and widgets.

Installing Arduino IDE
A step-by-step guide to installing and testing the Arduino software on Windows, Mac, and Linux.

New!

Installing Board Definitions in the Arduino IDE
How do I install a custom Arduino board/core? It's easy! This tutorial will go over how to install an Arduino board definition using the Arduino Board Manager. We will also go over manually installing third-party cores, such as the board definitions required for many of the SparkFun development boards.

Arduino Tutorials

- Getting Started > Introduction: What is Arduino and what I can use it for?
- Getting Started with Arduino and Genuino products
- Arduino Software (IDE)
- Arduino Troubleshooting
- Arduino: Contact Us