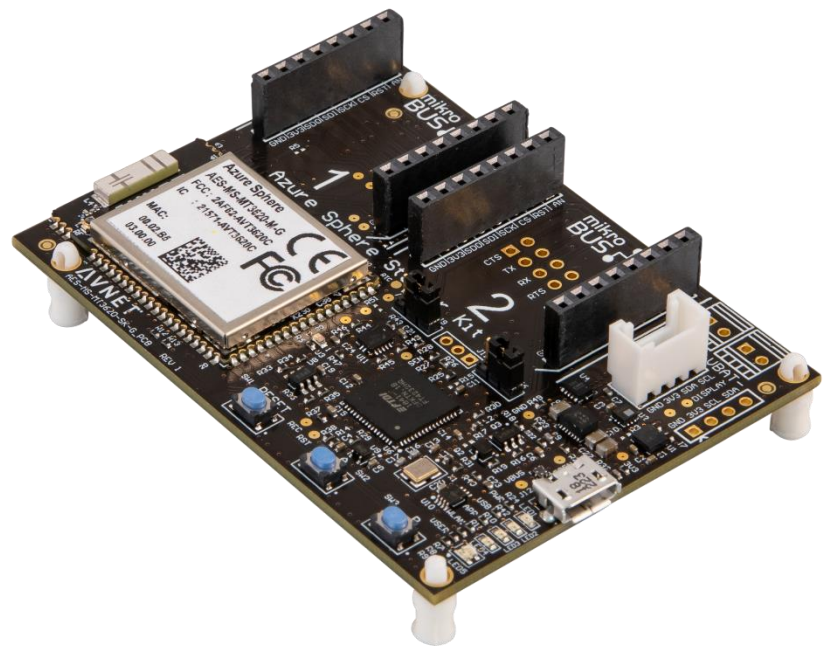


Avnet Technical Training Course

Azure Sphere: Getting Started Lab 1



Azure Sphere SDK:	19.05
Training Version:	v1
Date:	24 June 2019

© 2019 Avnet. All rights reserved. All trademarks and registered trademarks are the property of their respective owners. All specifications are subject to change without notice.

NOTICE OF DISCLAIMER: Avnet is providing this design, code, or information "as is." By providing the design, code, or information as one possible implementation of this feature, application, or standard, Avnet makes no representation that this implementation is free from any claims of infringement. You are responsible for obtaining any rights you may require for your implementation. Avnet expressly disclaims any warranty whatsoever with respect to the adequacy of the implementation, including but not limited to any warranties or representations that this implementation is free from claims of infringement and any implied warranties of merchantability or fitness for a particular purpose.

Introduction

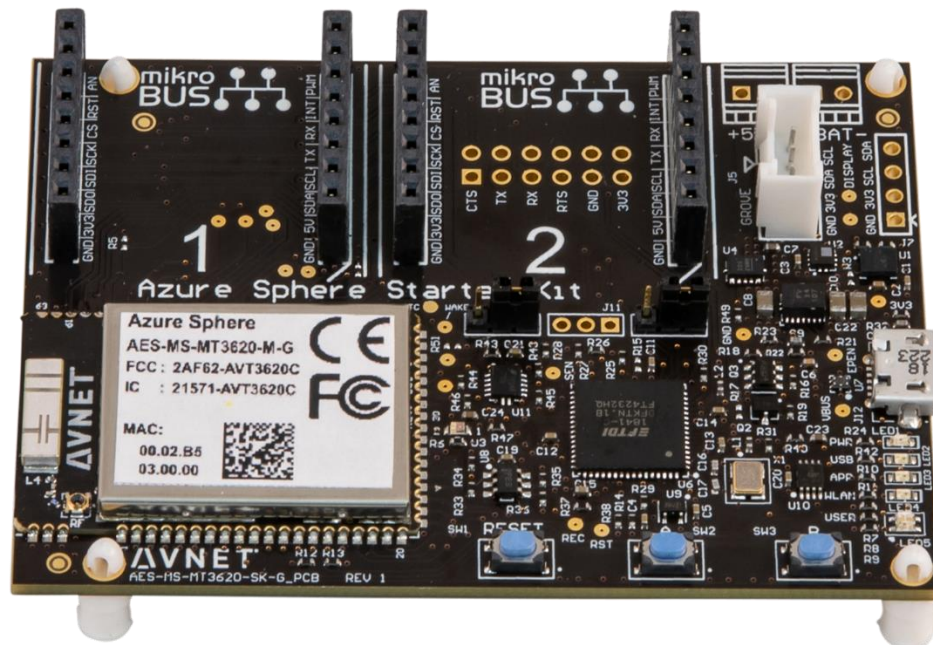
This Lab will walk the students through validating the development tools functionality, setting up an Azure Sphere Tenant, and preparing the Starter Kit for development.

Avnet Azure Sphere Starter Kit Overview

The Avnet Azure Sphere Starter Kit from Avnet Electronics Marketing provides engineers with a complete system for prototyping and evaluating systems based on the MT3620 Azure Sphere device.

The Avnet Azure Sphere MT3620 Starter Kit supports rapid prototyping of highly secure, end-to-end IoT implementations using Microsoft's Azure Sphere. The small form-factor carrier board includes a production-ready MT3620 Sphere module with Wi-Fi connectivity, along with multiple expansion interfaces for easy integration of off-the-shelf sensors, displays, motors, relays, and more.

The Starter Kit includes Avnet's MT3620 Module. Having the module on the Starter Kit means that you can do all your development work for your IoT project on the Starter Kit and then easily migrate your Azure Sphere Application to your custom hardware design using Avnet's MT3620 Module.



Avnet Azure Sphere Starter Kit

Lab 1: Objectives

The objectives of lab 1 are to perform all the checks and tasks to begin Azure Sphere Development

- Verify that the com port drivers are correctly installed
- Determine if the student has access to a Azure Sphere Tenant, and if not setup a new Azure Sphere Tenant
- Update the OS on the Azure Sphere device
- Claim the Azure Sphere device to your Azure Sphere Tenant
- Configure the Azure Sphere device Wi-Fi to connect to your local Wi-Fi access point or hotspot
- Unlock our Starter Kit for development

Requirements

Hardware

- A PC running Windows 10 Anniversary Update or later (Version 1607 or greater)
- An unused USB port on the PC
- An Avnet Azure Sphere Starter Kit
- A micro USB cable to connect the Starter Kit to your PC

Software

- Visual Studio 2019 Enterprise, Professional, or Community version 16.04 or later; or Visual Studio 2017 version 15.9 or later **installed**
- Azure Sphere SDK 19.05 or the current SDK release **installed**

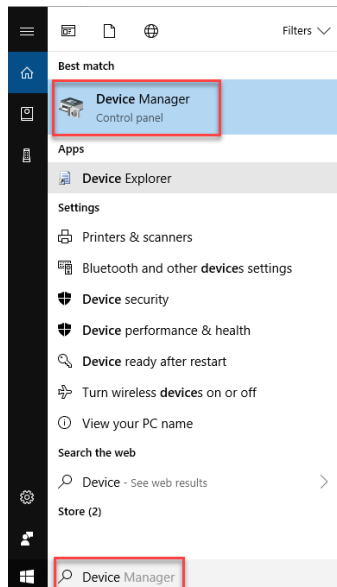
Other

- Administrator rights on your PC

Com Ports

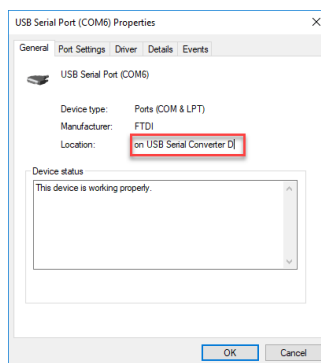
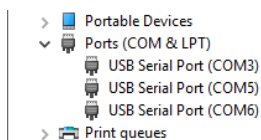
The first thing we need to do is to verify that the com port drivers are installed correctly.

1. Connect your Azure Sphere Starter Kit to your PC using the micro USB cable provided with the kit
2. Open your Windows Device Manager



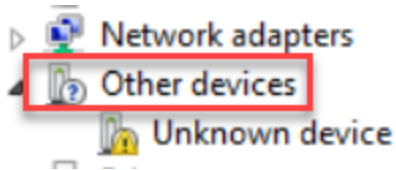
a. Type “Device Manager” in the Windows Search bar at the bottom of your desktop. When you see a selection for Device Manager, select it.

3. Once Device Manager is open, find the folder for “Ports (COM & LPT).” You should see three USB Serial Ports enumerated. Your COM numbers will likely be different than mine. These are the three COM ports for the Azure Sphere Starter Kit.



a. If you right click on each COM port and select “Properties” you should see three different “Locations,” USB Serial Converter A, USB Serial Converter B, and USB Serial Converter D. If you see these “Locations” for your three COM ports then they are all configured correctly! If not, you’ll need to update the FTDI drivers. See step #4 below.

4. If you don’t see the ports listed under “Ports (COM & LPT), then they may be under a folder called “Other Devices.” If this is the case you need to update the drivers.



5. To update the drivers, first download the FTDI drivers from the FTDI website [here](#). Then right-click on the “Unknown device” and click “Update Driver.” This will open a dialog box where you can browse to the folder where you put your FTDI drivers. You will need to repeat this process for each COM port.
6. Note, I’ve seen this process require two different driver updates. You’ll know the drivers are installed correctly if the COM port properties are as described in step 3.a above.

Azure Sphere Tenant

Next we need to gain access to an Azure Sphere Tenant. It's important to make sure you get setup with a Tenant that you can use for the long term. If you have a Tenant available through your work, I recommend using that tenant. If you need to setup your own tenant, that's fine too, but you should plan to keep the Azure account and the tenant for the long term. In a few minutes we're going to claim our device to our tenant. This is a one-time process, once a device has been claimed to a tenant, **it can't be moved to another tenant!** So be sure you have the tenant that you want to use for all your sphere development.

Microsoft has awesome documentation on the process to determine if you have access to an existing tenant, or if you need to create a new tenant. Please navigate to the Microsoft documentation [here](#). The tenant is used to manage Sphere devices.

Update the Sphere OS

Okay, we're making good progress! We've verified that our COM ports are correctly configured and we have access to a tenant. Next I want to show you how to update the Sphere OS. We want to update the OS in case your Starter Kit shipped with an older OS.

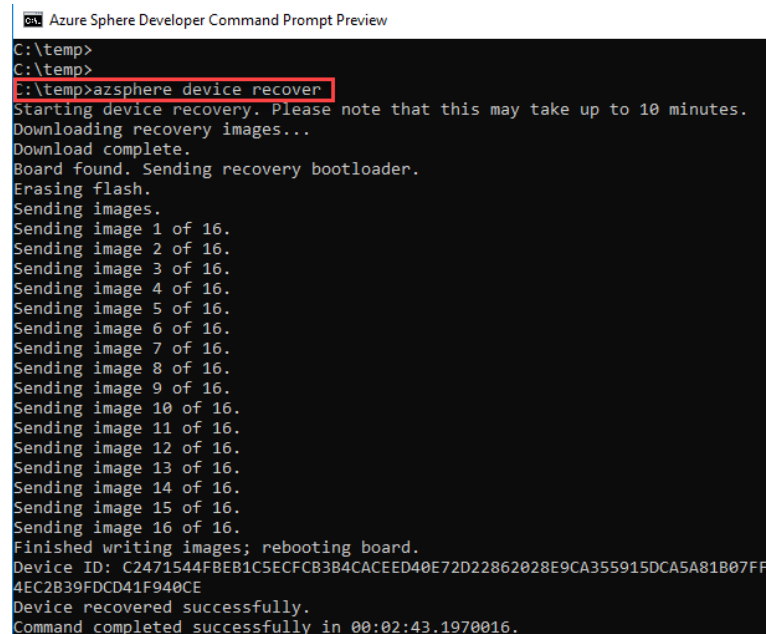
The on-line [Azure Sphere documentation](#) includes release notes for all the OS releases. Once you're on the webpage, navigate to the Resources → Release Notes section in the left menu. I recommend you review the information on the site so you can see what kind of changes you can expect when new Azure Sphere operating systems are released.

This is an easy process and just takes a few minutes. This is a manual process and once devices that are deployed they will automatically get OS updates Over-The-Air (OTA) without having to perform this step.

1. Open the Azure Sphere CLI from your start menu. Start → Azure Sphere → Azure Sphere Developer Command Prompt Preview
2. A terminal window will open with a prompt. When you installed the SDK this application was installed and furthermore a utility, azsphere.exe, was also installed. The azsphere CLI is used to configure things on your tenant and your Sphere device. You can learn more about this tool from the on-line documentation [here](#), or you can just run the command and see what the available options are. Feel free to explore!

From the command line type the following command: `azsphere device recover` and hit the enter key. The recover command will start the process of downloading the current Sphere OS files and flashing each OS component to your Sphere device. Your output should look similar to the screenshot below.

When you perform the `azsphere device recover` command you put the device back into its initial state. That is any applications on the device will be erased from Flash and any Wi-Fi networks that were previously configured will be removed.



```

C:\temp>
C:\temp>
C:\temp>azsphere device recover
Starting device recovery. Please note that this may take up to 10 minutes.
Downloading recovery images...
Download complete.
Board found. Sending recovery bootloader.
Erasing flash.
Sending images.
Sending image 1 of 16.
Sending image 2 of 16.
Sending image 3 of 16.
Sending image 4 of 16.
Sending image 5 of 16.
Sending image 6 of 16.
Sending image 7 of 16.
Sending image 8 of 16.
Sending image 9 of 16.
Sending image 10 of 16.
Sending image 11 of 16.
Sending image 12 of 16.
Sending image 13 of 16.
Sending image 14 of 16.
Sending image 15 of 16.
Sending image 16 of 16.
Finished writing images; rebooting board.
Device ID: C2471544FBE81C5ECFCB3B4CACEED40E72D22862028E9CA355915DCA5A81B07FF
4EC2B39FDCD41F940CE
Device recovered successfully.
Command completed successfully in 00:02:43.1970016.
```

If you encounter a timeout after the device reboots then the TAP driver installed by the SDK installer is not working correctly. You'll need to get the TAP driver working before moving on. You can quickly determine if the TAP driver is working correctly by executing the command: `azsphere device show-attached`. You'll either see a response, or you'll get another timeout.

- Some things to try . . .
 - Confirm that the COM ports are all correctly enumerated in the Device Manager
 - If you're using a VPN, confirm that the split tunneling feature is enabled. If not, then disconnect from your VPN and try again.
 - There are many other things to try all loosely documented on the MSDN Azure Sphere Forum [here](#).

Claim your Azure Sphere Device

The next step is to claim your device to your Azure Sphere Tenant. This process adds an entry in a database in Azure somewhere to associate your device (using the device ID) to your tenant. This allows you to manage your devices and can tie your device to other Azure services such as a Device Provisioning Service (DPS).

Having the device associated to your tenant is also a security feature. For example, if someone was able to get hold of one of your applications and they tried to load it onto a device that was NOT in your tenant, then that device would not be authorized to use the DPS associated with your tenant, and thus would not be able to connect to your Azure IoT Hub.

Claiming a device is a simple process.

1. Login to your tenant.
 - a. From the command line enter the command: `azsphere login` and hit the enter key. A window will open and prompt you for your tenant credentials. These are the credentials you setup in the “Azure Sphere Tenant” section of this document. This step ensures that you are authorized to manage the tenant.
 - b. The command will output the ID of your tenant. Note that once you login to your tenant the login is sticky. You will remain logged in until you log out.

```
Azure Sphere Developer Command Prompt Preview
C:\temp>azsphere login
The selected Azure Sphere tenant 'sdeAvnetInc' (8d34f65c-532e-4dcf-a1d6-3e811c1e5c68) will be retained.
Successfully logged in with the selected AAD user. This authentication will be used for subsequent commands.
Command completed successfully in 00:00:12.9877681.
```

2. Make sure your device is connected to your PC, then execute the command `azsphere device claim` and hit the enter key.

```
Azure Sphere Developer Command Prompt Preview
Successfully logged in with the selected AAD user. This authentication will be used for subsequent commands.
Command completed successfully in 00:00:12.9877681.
C:\temp>azsphere device claim
Claiming device.
Successfully claimed device ID 'C2471544FBEB1C5ECFCB384CACEED40E72D22862028E9CA355915DCA5A81B07FF0A171953152A40FE22A30
0C1ECBA9AB7FF332277EF4EC2B39FDCD41F940CE' into tenant 'sdeAvnetInc' with ID '8d34f65c-532e-4dcf-a1d6-3e811c1e5c68'.
Command completed successfully in 00:00:03.0047724.
```

- a. When you execute the claim command the tool will read the device ID from your device, send the device to the Azure Sphere Tenant where the ID will be stored in a database.
- b. Note that this command can also be executed without the device connected using the `-i <device Id>` arguments. So if you needed to claim 100,000 devices and you had all the device IDs in a text file you could script the process.

Configure the Wi-Fi

We've almost completed this lab. The last thing we need to do is to configure the Wi-Fi on our device to connect to a Wi-Fi access point or hotspot.

1. Make sure your device is still connected to your PC
2. Execute the command `azsphere device wifi add -s <your ssid> -k <your ssid password>` where <your ssid> and < your ssid password> are your SSID and you SSID password!

```
cmd Azure Sphere Developer Command Prompt Preview
C:\temp>
C:\temp>azsphere device wifi add -k new-network -s new-password
Add network succeeded:
ID           : 0
SSID         : new-password
Configuration state : enabled
Connection state : unknown
Security state  : psk
Command completed successfully in 00:00:01.9878154.
```

3. You can check the status of your Wi-Fi connection by executing the `azsphere device wifi show-status` command.

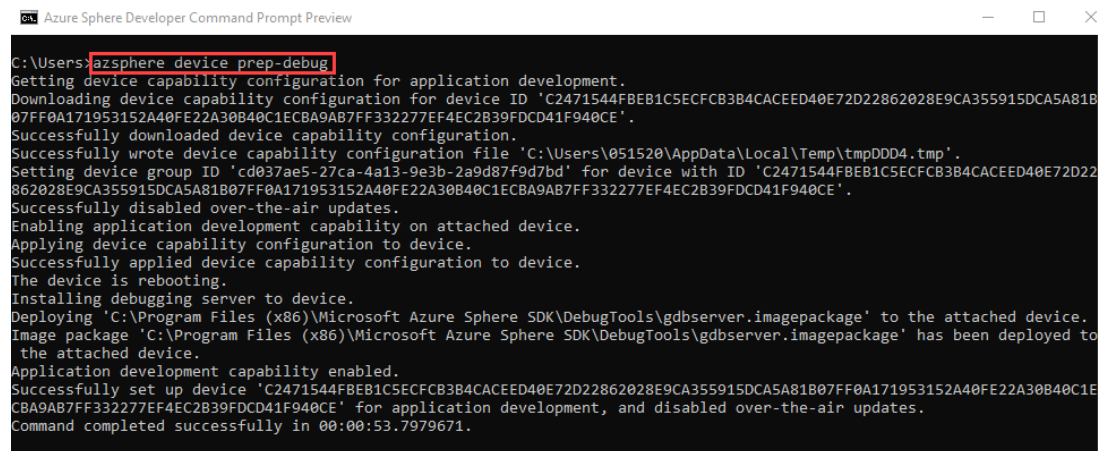
```
cmd Azure Sphere Developer Command Prompt Preview
Command completed successfully in 00:00:02.2539561.
C:\temp>azsphere device wifi show-status
SSID           : wirelessnetwork-5G
Configuration state : enabled
Connection state  : connected
Security state    : psk
Frequency        : 5765
Mode             : station
Key management    : WPA2-PSK
WPA State        : COMPLETED
IP Address       : 192.168.1.40
MAC Address      : 00:02:b5:03:28:b4
Command completed successfully in 00:00:01.6603153.
```

Unlock the Starter Kit for Development

By default Azure Sphere is a locked down device. You can't load any applications onto the device even with a physical connection, if you could, bad actors who had physical access to the device could load whatever application they wanted to onto the device. That would be bad! So when Sphere devices are deployed into the wild, they are put into field-prep mode that adds the device into a device group and assigns a product SKU to the device so that the Azure Sphere Security Service (in the cloud) knows what OS SKU and user Application to send to the device over-the-air. Once in field-prep mode the ONLY way to change software on the device is OTA via the AS3. You can read more about OTA software deployments [here](#).

Devices ship in a locked state, so we need to “unlock” it. This is accomplished using, you guessed it, the `azsphere` command. We should still be logged into our tenant. Since we want to change the accessibility of our device, the AS3 will first authenticate us to the tenant, then when we execute the command to change the device from field-prep to prep-debug, AS3 will confirm that our device belongs (or has been claimed) to our tenant. If not, then we won't be able to change the status of the device. So if someone was to steal a deployed device, they could not do anything with the device outside its main purpose in life.

1. Confirm that your device is connected to your PC using the supplied USB cable
2. Next, put the device into prep-debug mode by entering the command `azsphere device prep-debug`. You should see output similar to the graphic below



```

C:\Users>azsphere device prep-debug
Getting device capability configuration for application development.
Downloading device capability configuration for device ID 'C2471544FBEB1C5ECFCB3B4CACEED40E72D22862028E9CA355915DCA5A81B07FF0A171953152A40FE22A30B40C1ECBA9AB7FF332277EF4EC2B39FDCD41F940CE'.
Successfully downloaded device capability configuration.
Successfully wrote device capability configuration file 'C:\Users\051520\AppData\Local\Temp\tmpDDD4.tmp'.
Setting device group ID 'cd037ae5-27ca-4a13-9e3b-2a9d87f9d7bd' for device with ID 'C2471544FBEB1C5ECFCB3B4CACEED40E72D22862028E9CA355915DCA5A81B07FF0A171953152A40FE22A30B40C1ECBA9AB7FF332277EF4EC2B39FDCD41F940CE'.
Successfully disabled over-the-air updates.
Enabling application development capability on attached device.
Applying device capability configuration to device.
Successfully applied device capability configuration to device.
The device is rebooting.
Installing debugging server to device.
Deploying 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' to the attached device.
Image package 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' has been deployed to the attached device.
Application development capability enabled.
Successfully set up device 'C2471544FBEB1C5ECFCB3B4CACEED40E72D22862028E9CA355915DCA5A81B07FF0A171953152A40FE22A30B40C1ECBA9AB7FF332277EF4EC2B39FDCD41F940CE' for application development, and disabled over-the-air updates.
Command completed successfully in 00:00:53.7979671.
```

Wrap Up

In this Lab we learned a little more about the Azure Sphere ecosystem. Specifically we learned . . .

- How to validate that our Azure Sphere COM ports are correctly configured and what to do if they are not
- How to setup an Azure Sphere Tenant
- How to access our Azure Sphere Tenant
- How to update the OS on our Azure Sphere device
- How to claim our Azure Sphere device to our tenant
- How to configure the Wi-Fi on our device and how to verify that it's connected to a Wi-Fi network
- How to put our device into prep-debug mode

The next lab will get us started with some Azure Sphere development! That's the fun part.

Revision History

Date	Version	Revision
25 June 19	01	Preliminary release
9 July 19	02	Minor updates based on document reviews