

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

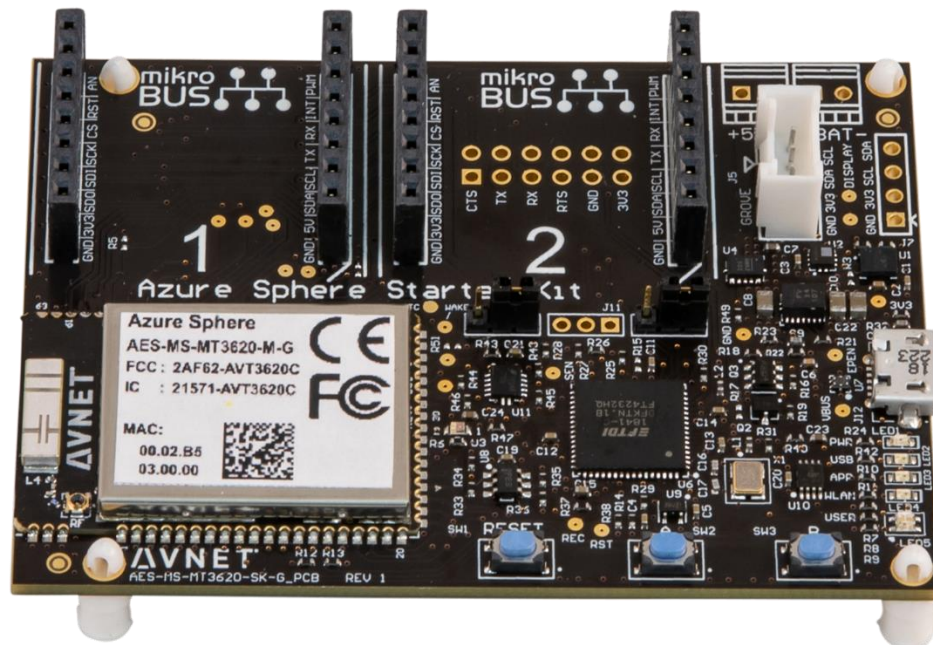
This Lab will walk the students through validating the development tools functionality, setting up an Azure Sphere Tenant, and preparing the Starter Kit for development.

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 tasks required to begin Azure Sphere Development

- Determine if you have access to an Azure Sphere Tenant. If not, then setup a new Tenant
- Update the OS on the Azure Sphere device
- Claim the Azure Sphere device to your Azure Sphere Tenant
- Configure the device Wi-Fi settings 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 version 16.04 or later (Enterprise, Professional, or Community version); **or** Visual Studio 2017 version 15.9 or later **installed**
- Azure Sphere SDK 19.11 or the current SDK release **installed**

Other

- Administrator rights on your PC

Update the Sphere OS

It is important that you start by updating your Starter Kit to the latest Azure Sphere OS.

Manually updating the OS will not be required in a deployment situation. When devices are deployed, all software, including the OS will be performed Over-the-Air (OTA). Even in a development environment, we don't have to manually update the OS. Anytime a device is connected to the internet, it checks the OS version after reset and *will* receive OTA the latest OS update. (ie. One could just configure the Wi-Fi on your device and wait for the OS update to happen, but we are going to manually force this update)

Note that when we manually update the OS, this is called a “recover” process, all applications and any Wi-Fi configurations on the device will be removed/erased. Not a big deal, but something to be aware of.

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 takes just a few minutes. This is a manual process and as mentioned, once a device is connected to the internet, it will automatically get OS updates OTA without need for 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 (This **azsphere.exe** command line utility is part of your SDK installation). 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 enter the **azsphere** 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 this **recover** command, any applications on the device will be erased from Flash memory and all Wi-Fi networks that were previously configured will be removed)

```
CS: Azure Sphere Developer Command Prompt Preview
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: C2471544FBEB1C5ECFCB3B4CACEED40E72D22862028E9CA355915DCA5A81B07FF
4EC2B39FDCD41F940CE
Device recovered successfully.
Command completed successfully in 00:02:43.1970016.
```

If you encounter a timeout after the device reboots then your serial ports are not correctly configured, or the TAP driver installed by the SDK installer is not working correctly. You'll need to resolve this issue before moving on. This usually just a matter of updating the serial port drivers, see the appendix at the end of this document for details. If updating the serial port drivers does not resolve your issue, the see the troubleshooting guide by following the link below.

Microsoft has been documenting potential issues and how to work around them. This page is continually being updated.

- [Troubleshooting installation and setup on Windows](#)

Azure Sphere Tenant

To manage your Azure Sphere devices, you need access to an Azure Sphere "Tenant", this is the Microsoft provided method to isolate- and manage Azure Sphere devices (only used for Azure Sphere). It's important to get setup with a tenant that can be used for the long term. If a tenant is available through your work, it is recommended to use that. If your own tenant needs to be set up, that's fine too, but plan on keeping long term your Azure account and tenant. You will claim your Starter Kit device to this tenant, which is a one-time process, as once claimed, **it can't be moved to another tenant** (i.e. Be sure this the tenant you want to use for all your sphere development)

Microsoft has good Azure Sphere Tenant documentation [here](#).

Login and/or Create an Azure Sphere Tenant

- Sign in to Azure Sphere, using your Microsoft Windows account credentials:
 - **azsphere login**

If you've never logged-in to Azure Sphere or have just installed 19.10 or later SDK, then add the **--newuser** parameter along with your Microsoft account email address:

- **azsphere login --newuser <email-address>**

After you're logged in you'll fall into one of three situations:

1. You are already assigned to one and only one tenant. That tenant will become the "current" tenant
2. You have access to more than one tenant, so you need to select one:
 - **azsphere tenant list**
 - **azsphere tenant select -i <tenant GUID>**
3. You don't have access to any tenants and none exist for you to use, so you need to create one:
 - **azsphere tenant create <tenant name> -f**
 - **azsphere tenant select -i <tenant guidID>**

You can always show the currently selected tenant using the **azsphere tenant show-selected** command

4. You can also add users to your tenant
 - **azsphere register-user -u <email_address>**
 - **azsphere role add -r Contributor | Administrator -u <email_address>**

Claim your Azure Sphere Device

Now you will claim your Starter Kit device to your Azure Sphere Tenant. This process adds an entry in an Azure Sphere Security Service database, to associate your device (using the device ID) with 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 (as completed in previous section).
 - 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**

```
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 'C2471544FBEB1C5ECFCB3B4CACEED40E72D22862028E9CA355915DCA5A81B07FF0A171953152A40FE22A300C1ECBA9AB7FF332277EF4EC2B39FDCD41F940CE' 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 <ssid> -p <password> --targeted-scan** where <ssid> and <password> are your SSID and your SSID password
 - a. The **--targeted-scan (-t)** option was added in the 19.09 release and is major improvement to Azure Sphere Wi-Fi. **You should always add Wi-Fi networks with this -t option.**

```
C:\temp>azsphere device wifi add -s new-network -p new-password --targeted-scan
Add network succeeded:
ID                : 3
SSID              : new-network
Configuration state : enabled
Connection state  : unknown
Security state    : psk
Targeted scan     : True

Command completed successfully in 00:00:01.9714795.
```

3. You can check the status of your Wi-Fi connection by executing **azsphere device wifi show-status**
4. You can check the status of all Wi-Fi networks configured on the device by executing **azsphere device wifi list**

```
Azure Sphere Developer Command Prompt Preview
Command completed successfully in 00:00:02.2539561.

C:\temp>azsphere device wifi show-status
SSID                : willessnetwork-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.
```

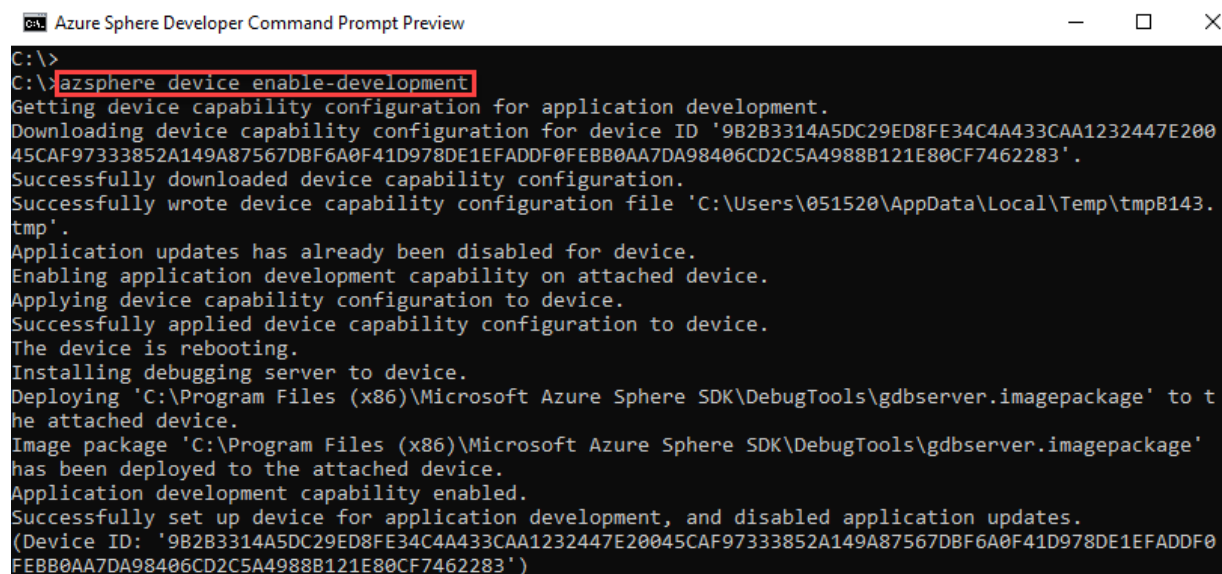

Unlock the Starter Kit for Development

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. So when Sphere devices are deployed into the wild, they are put into enable-cloud-test mode that adds the device into a device group and assigns a product ID 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 enable-cloud-test 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 accomplished using an **azsphere** command. We should still be logged into to 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 enable-cloud-test to enable-development mode, 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 accessibility 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 enable-development mode by entering the command **azsphere device enable-development** or **azsphere device edv**

You should see output similar to the graphic below



```
C:\>
C:\>azsphere device enable-development
Getting device capability configuration for application development.
Downloading device capability configuration for device ID '9B2B3314A5DC29ED8FE34C4A433CAA1232447E20045CAF97333852A149A87567DBF6A0F41D978DE1EFADDF0FEBB0AA7DA98406CD2C5A4988B121E80CF7462283'.
Successfully downloaded device capability configuration.
Successfully wrote device capability configuration file 'C:\Users\051520\AppData\Local\Temp\tmpB143.tmp'.
Application updates has already been disabled for device.
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 for application development, and disabled application updates.
(Device ID: '9B2B3314A5DC29ED8FE34C4A433CAA1232447E20045CAF97333852A149A87567DBF6A0F41D978DE1EFADDF0FEBB0AA7DA98406CD2C5A4988B121E80CF7462283')
```


Wrap Up

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

- 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 enable-development mode

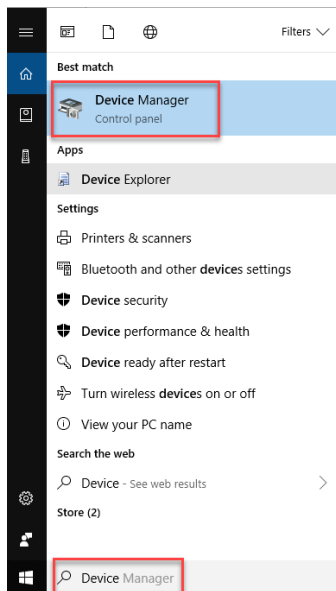
The next lab will get us started with some Azure Sphere development! That's the fun part.

Appendix

Com Ports

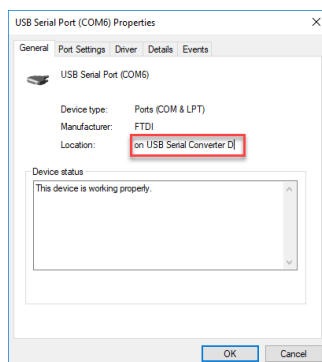
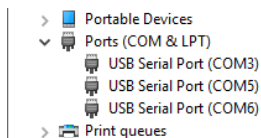
If your Azure Sphere board does not communicate with the azsphere.exe utility, you may just need to install/update USB Serial port drivers. See this section for details.

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.

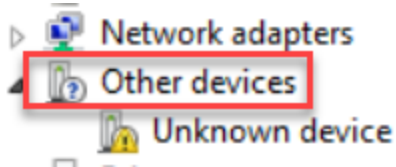


If you right click each COM port and select “**Properties**”, their “**Location**” will list one of the following:

- USB Serial Converter A
- USB Serial Converter B
- USB Serial Converter D

If this seen for the COM ports, then they are configured correctly!
If not, you will 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.

Revision History

Date	Version	Revision
25 Jun 2019	01	Preliminary release
09 Jul 2019	02	Minor updates based on document reviews
19 Oct 2019	03	Updated wifi add details for 19.09 Azure Sphere release
25 Oct 2019	04	Updated wifi add details to use targeted scan feature
08 Nov 2019	05	Updated for changes in 19.10 SDK Fixed broken links
14 Nov 2019	06	Minor updates
23 Dec 2019	07	Changes for the 19.11 release. Created an appendix and moved the section on validating com ports there.
18 Jan 2020	08	Clean-up and readability edits
20 Jan 2020	09	Added details for logging into the Azure Sphere Tenant
24 Jan 2020	10	Add details for adding users to Azure Sphere Tenants