IoT Development Lab 1

Windows 10 IoT Core

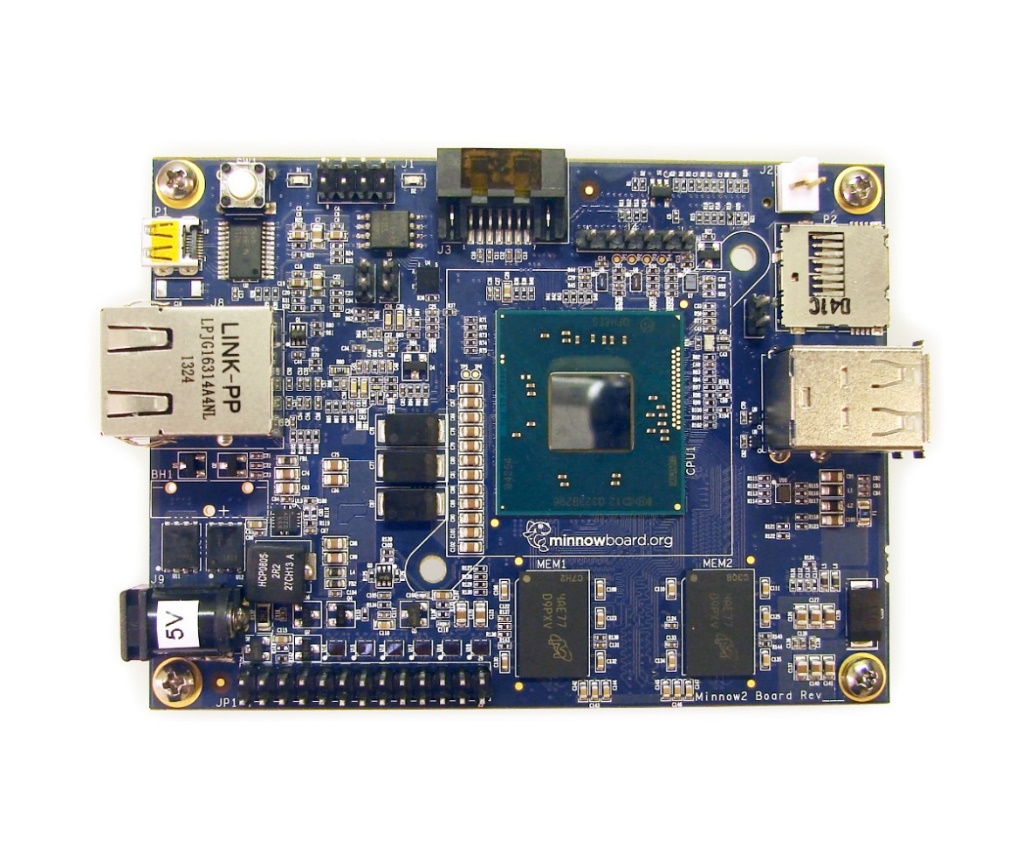
Image Creation & Customization

***Abstract*:** This is a fictional lab, intended to show the use of the template. The various styles in the WinHEC HOL template are used throughout. The lab does not imply HOL best practices nor appropriate length. The copyright page should be used intact in your lab, however. Also, do not move, resize, write text over or otherwise edit the logos on the cover page.

***Prerequisites*:** This lab is a hands-on look at the creation of the Universe. The lab assumes a working knowledge of Alchemy or general Windows magic. No prior knowledge of Windows 10 IoT Core is required.

***Install and configure*:** You will need:

* A desktop PC running on Windows 10 RTM 2014
* A MinnowBoard Max development board running on Windows 10 IoT Core
* A Micro-SD card to flash the IoT Image on.
* A keyboard, a mouse and a monitor for the MinnowBoard Max.



5V Power Adapter 🡪

Ethernet Cable 🡪

🡨 USB port x2

Micro-HDMI Cable 🡪

🡨 Micro-SD card

Power Button

↓

* An Ethernet cable to connect the desktop PC and MinnowBoard Max

The following table shows configuration information for these computers

|  |  |
| --- | --- |
| Development Computer Admin Account | Admin |
| Development Computer Admin Password |  |

|  |  |
| --- | --- |
| MinnowBoard Max Computer Name | minwinpc |
| MinnowBoard Max Admin Account | Administrator |
| MinnowBoard Max Admin Password | p@ssw0rd |

Copyright (c) 2015 Microsoft. All rights reserved.

This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice.

Contents

[Lab objective 4](#_Toc433743975)

[Exercise 1: Verify Hardware Setup 4](#_Toc433743976)

[Section 1: MinnowBoard Max Setting 4](#_Toc433743977)

[Section 2: Development PC and Software Settings 4](#_Toc433743978)

[Exercise 2: Create a Driver Package using Command Line 5](#_Toc433743979)

[Section 1: Examine gpio.pkg.xml 5](#_Toc433743980)

[Section 2: Install OEM Test Certificates 5](#_Toc433743981)

[Section 3: Create a Driver Package using PkgGen.exe 6](#_Toc433743982)

[Exercise 3: Add the Driver Package into IoT Core Package Files 6](#_Toc433743983)

[Section 1: Add to OS Component and Layout Packages. 6](#_Toc433743984)

[Section 2: Add the Feature into the Feature Manifest. 7](#_Toc433743985)

[Section 3: Add a Feature into ProductionOEMInput.xml 8](#_Toc433743986)

[Exercise 4: Build an Image using Command Line 9](#_Toc433743987)

[Section 1: Create FFU using imggen.exe 9](#_Toc433743988)

[Exercise 5: Deploy the Image to MinnowBoard Max 9](#_Toc433743989)

[Section 1: Flash the image to SD card. 9](#_Toc433743990)

[Section 2: Power up the MinnowBoard Max. 10](#_Toc433743991)

[Section 3: Use Web Browser to Check Device Information 11](#_Toc433743992)

[[Optional]Exercise 6: Configure Default App for Windows IoT Core 12](#_Toc433743993)

[Section 1: Deploy a App using Web Browser. 12](#_Toc433743994)

[Section 2: Set the Default App 12](#_Toc433743995)

**Lab objective**

Participants will learn about how to customize an FFU image for Windows 10 IoT Core by adding a sample GPIO controller driver. The tutorials for this lab will be presented as hands-on tasks that have been grouped into the following sections:

* Verify the hardware setup
* Create a driver package for the sample driver.
* Modify the necessary file to include the driver package.
* Build and deploy the FFU image to MinnowBoard Max.

**Exercise 1: Verify Hardware Setup**

In this exercise, you will make sure that your Desktop computer and the development board (MinnowBoard Max) have been set up with the files and software tools that you need for this hands-on lab.

Section 1: MinnowBoard Max Setting

In this section you will verify that your development board (MinnowBoard Max) is set up correctly for this lab.

**On MinnowBoard Max**

1. Make sure that the MinnowBoard Max is powered up and running, and you can see the Windows 10 IoT Core Default IoT App displayed on the monitor.
2. Record the IP Address on the Default IoT App.



MinnowBoard Max IP Address = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

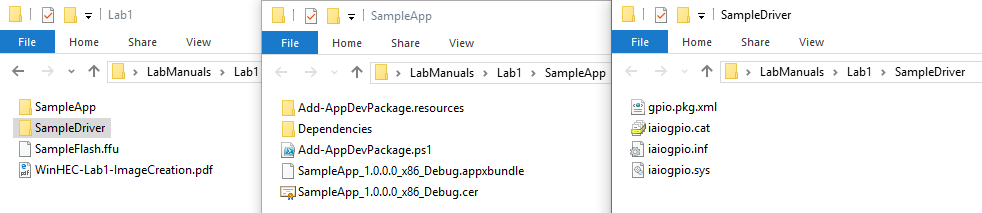
1. Check that there is an Ethernet cable connecting your MinnowBoard Max to your Dev PC

Section 2: Development PC and Software Settings

In this section you will check if the development PC is ready to create AllJoyn Device System Bridge app.

**On Development Computer**

1. Make sure that your Dev PC is booted up and running on Window 10.
2. Launch **WindowsIoTCoreWatcher**, verify that your IoT device’s IP Address in the watcher matches the IP Address displayed on the Default IoT App.
3. On the Desktop, double-click to open the **LabManuals\Lab1** folder, then verify that it contains files as shown in the following image.



1. Close the folder.

From now, the rest of exercises will be performed **on the Development PC** unless stated otherwise.

**Exercise 2: Create a Driver Package using Command Line**

To include a driver into a FFU image, we need to create a driver package and add the driver package information into the files that are used to create the FFU image. In this lab, we will walk you through the process using a sample GPIO driver.

## Section 1: Examine gpio.pkg.xml

In this section, you will examine the gpio.pkg.xml, which is the templated used to create a driver package for our Sample GPIO Driver.

1. Navigate to the **LabManuals\Lab1** folder on Desktop.
2. Under the **SampleDriver,** open **gpio.pkg.xml** with Notepad.
3. Check that the names of the driver INF file and SYS file match those in the **SampleDriver** folder.
4. Close **gpio.pkg.xml**.

## Section 2: Install OEM Test Certificates

In this section, we will install OEM test certificates so we can sign the driver package. You only need to install the OEM certificates once (meaning you only need to do this section once on a machine).

1. Launch **Deployment and Imaging Tools Environment** as ADMIN
2. Set environment variables. Please make sure that the commands are copied as they are shown. **Be careful NOT to add extra spaces at the end of each command**.

SET W10\_KITROOT=%ProgramFiles(x86)%\Windows Kits\10

SET WPDKCONTENTROOT=%W10\_KITROOT%

SET W10\_TOOL=%W10\_KITROOT%\bin\x86;%W10\_KITROOT%\Tools\bin\i386

SET PATH=%PATH%;%W10\_TOOL%;

1. Install OEM Certificates:

installoemcerts.cmd

Section 3: Create a Driver Package using PkgGen.exe

In this section, you will create CBS cab filed based on the gpio.pkg.xml using PkgGen.exe.

1. Launch **Deployment and Imaging Tools Environment** as ADMIN
2. Set the environment variables by inputting the following into the cmd:

SET PATH=%KITSROOT%tools\bin\i386;%PATH%

1. Specify that we are using the OEM test certificates

set SIGN\_OEM=1

set SIGN\_WITH\_TIMESTAMP=0

1. Navigate to the **SampleDriver** folder, where the driver files are located.

Cd C:\Users\Admin\Desktop\LabManuals\Lab1\SampleDriver

1. Build the image with the following command:

PkgGen.exe /output:. /version:10.0.12649.0 /build:fre /cpu:x86 /variables:"HIVE\_ROOT=C:\Program Files (x86)\Windows Kits\10\CoreSystem\10.0.10240.0\x86" /config:"c:\Program Files (x86)\Windows Kits\10\tools\bin\i386\pkggen.cfg.xml" gpio.pkg.xml

1. When the command completes, you should see **WinHEC.MBM.GPIO.cab** and **WinHEC.MBM.GPIO.spkg** under the **SampleDriver** folder.

# Exercise 3: Add the Driver Package into IoT Core Package Files

In this exercise, you will add the necessary components into the files used in the image creation.

## Section 1: Add to OS Component and Layout Packages.

In this section you will copy the CBS cab file we just created for the sample GPIO driver under the MSPackages folder.

1. Navigate to **C:\Program Files (x86)\Windows Kits\10\MSPackages\Retail\X86\fre**
2. Copy the **WinHEC.MBM.GPIO.cab** we created into this location.

[Continue on next page…]

## Section 2: Add the Feature into the Feature Manifest.

Drivers are included in an image as “features”. In this section you will learn how to declare our sample GPIO driver as a feature in the feature manifest file.

1. Navigate to **C:\Program Files (x86)\Windows Kits\10\FMFiles\OEM**.
2. Open **OEMFM.xml** with NotePad.
3. Find the **<OEM>** node under the **<Features>** node. You will a list of “features” in there already.

<FeatureManifest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns="http://schemas.microsoft.com/embedded/2004/10/ImageUpdate">

<Features>

...

<OEM>

<PackageFile Path="$(mspackageroot)\Retail\X86\fre" Name="Intel.MBM.GRFX.cab" ID="Intel.MBM.GRFX">

<FeatureIDs>

<FeatureID>MBM\_DRIVERS</FeatureID>

</FeatureIDs>

</PackageFile>

<PackageFile Path="$(mspackageroot)\Retail\X86\fre" Name="Intel.MBM.I2C.cab" ID="Intel.MBM.I2C">

<FeatureIDs>

<FeatureID>MBM\_DRIVERS</FeatureID>

</FeatureIDs>

</PackageFile>

...

</OEM>

<OEMFeatureGroups />

...

</Features>

...

</FeatureManifest>

1. Before closing **</OEM>** node, add the highlighted lines below.

<OEM>

...

<PackageFile Path="$(mspackageroot)\Retail\X86\fre" Name="WinHEC.MBM.GPIO.cab" FeatureIdentifierPackage="true" ID="WinHEC.MBM.GPIO">

<FeatureIDs>

<FeatureID>WinHEC\_GPIO</FeatureID>

</FeatureIDs>

</PackageFile>

</OEM>

1. **Save** OEMFM.xml. If you receive a “Access denied” pop-up when trying to save OEMFM.xml at C:\Program Files (x86)\Windows Kits\10\FMFiles\OEM. You may save it at Desktop first, then use “cut & paste” to replace the OEMFM.xml under C:\Program Files (x86)\Windows Kits\10\FMFiles\OEM.

Notice that package path (“$(mspackageroot)\Retail\X86\fre”) matches the location to where we copied our WinHEC.MBM.GPIO.cab in Section 1. The name also matches the name of our driver package (WinHEC.MBM.GPIO.cab). Lastly, take a note on the feature ID, WinHEC\_GPIO. We are going to use this feature ID in the next section.

## Section 3: Add a Feature into ProductionOEMInput.xml

In this section, we are going to add the feature we declared in the OEM feature manifest file into the ProductionOEMInput.xml, which is what our FFU image will be based on.

1. Navigate to **C:\Program Files (x86)\Windows Kits\10\OEMInputSamples\OEM**
2. Open **ProductionOEMInput.xml** with NotePad.
3. Find the **<AdditionalFMs>** node. Make sure the OEMFM.xml you modified is included and the path is correct.

<AdditionalFMs>

<AdditionalFM>%AKROOT%\FMFiles\x86\IoTUAPNonProductionPartnerShareFM.xml</AdditionalFM>

<AdditionalFM>%AKROOT%\FMFiles\OEM\OEMFM.xml</AdditionalFM>

</AdditionalFMs>

1. Find the **<OEM>** node under **<Features>** node.
2. Add the feature ID **WinHEC\_GPIO** into the OEM feature list, as highlighted below.

<Features>

<Microsoft>

<Feature>IOT\_EFIESP</Feature>

<Feature>IOT\_EFIESP\_BCD</Feature>

<Feature>IOT\_NETCMD</Feature>

<Feature>IOT\_DELETE\_FW\_VARS</Feature>

<Feature>PRODUCTION\_CORE</Feature>

<Feature>PRODUCTION</Feature>

<Feature>IOT\_TOOLKIT</Feature>

<Feature>IOT\_FTP</Feature>

<Feature>IOT\_WEBB\_EXTN</Feature>

<Feature>IOT\_KDSERIAL\_SETTINGS</Feature>

<Feature>IOT\_ENABLE\_TESTSIGNING</Feature>

<Feature>IOT\_DISABLE\_UMCI</Feature>

<Feature>IOT\_DEFAULTAPP</Feature>

<Feature>IOT\_TOOLS</Feature>

<Feature>IOT\_DRIVERS\_WDTFINFRA</Feature>

<Feature>IOT\_VS\_TOOLS</Feature>

<Feature>IOT\_POWERSHELL</Feature>

<Feature>IOT\_ALLJOYN</Feature>

<Feature>IOT\_SSH</Feature>

<Feature>IOT\_SIREP</Feature>

<Feature>IOT\_VERIFIER</Feature>

</Microsoft>

<OEM>

<Feature>MBM\_DRIVERS</Feature>

<Feature>PRODUCTION</Feature>

<Feature>WinHEC\_GPIO</Feature>

</OEM>

</Features>

1. **Save** ProductionOEMInput.xml.

# Exercise 4: Build an Image using Command Line

In this exercise, you will build an IoT Core image using a command-line tool called imggen.exe.

## Section 1: Create FFU using imggen.exe

1. Launch **Deployment and Imaging Tools Environment** as ADMIN
2. Set the environment variables by inputting the following (in order) into the cmd:

SET PATH=%KITSROOT%tools\bin\i386;%PATH%

SET AKROOT=%KITSROOT%

1. Create a folder named **MyFFU** under the **Lab1** folder. Navigate to the new folder, this is where we will place our FFU image.

CD C:\Users\Admin\Desktop\LabManuals\Lab1\MyFFU

1. Build the image with the following command:

imggen.cmd MyFlash.ffu "%KITSROOT%OEMInputSamples\OEM\ProductionOEMInput.xml" "%KITSROOT%MSPackages" x86

The process will take about 25 minutes to complete. To save time, we will continue the lab using a pre-created image (**SampleFlash.ffu**) that contains the sample GPIO controller driver. You may leave the process running while we continue with Exercise 5.

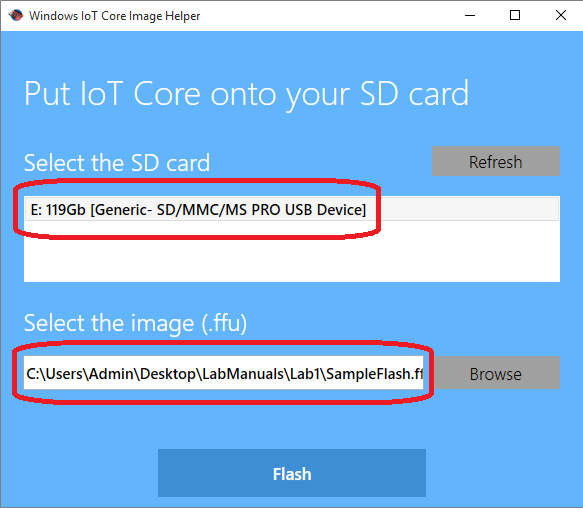
# Exercise 5: Deploy the Image to MinnowBoard Max

In this exercise, we will deploy the image we just created onto MinnowBoard Max and use web browser to check that the sample GPIO device we packed in the FFU is up and running.

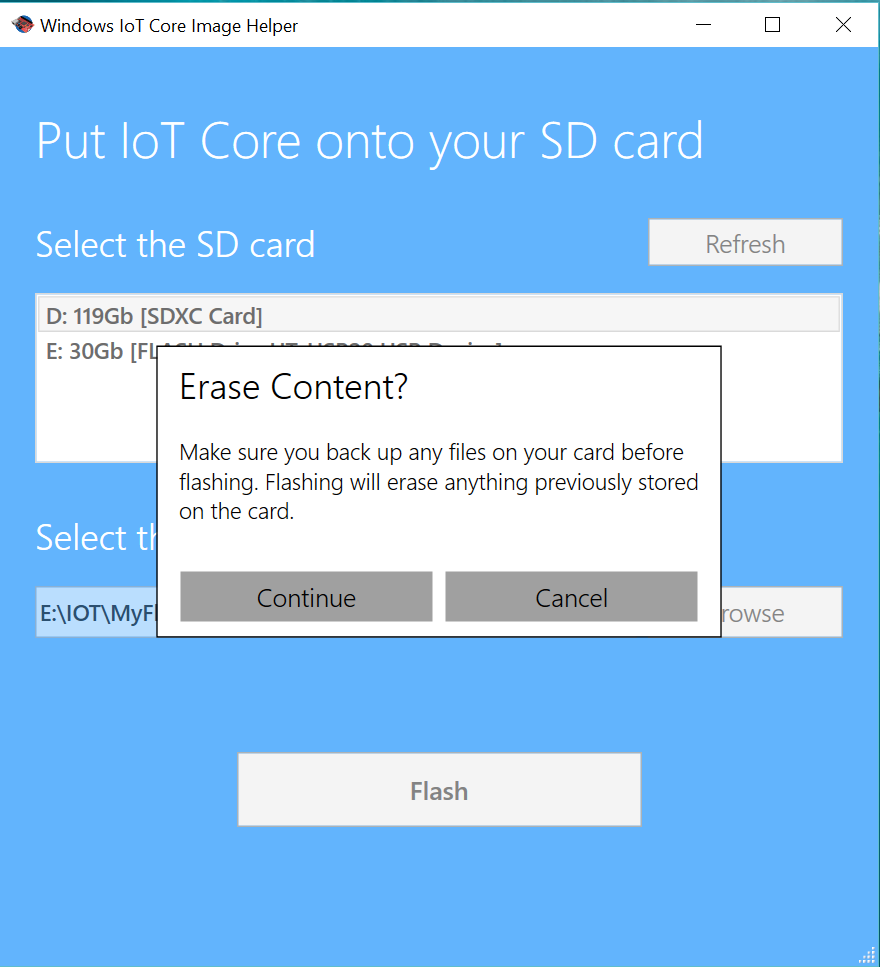
## Section 1: Flash the image to SD card.

In this section you will add support for signals to the adapter by adding an event handler.

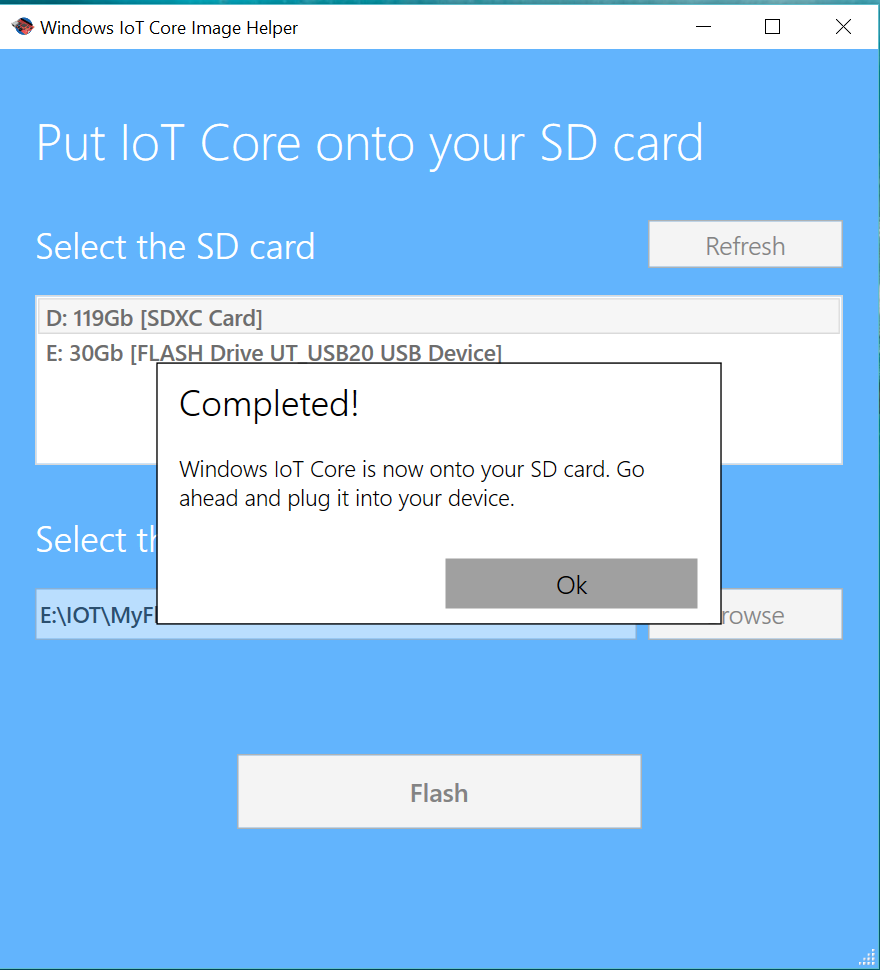
1. Power off the MinnowBoard Max if you have not done so. Eject the micro-SD card.
2. Insert the SD card into your Development PC.
3. Launch **WindowsIoTImageHelper**.
4. Select the SD card. If your SD card is not on the list, use “Refresh” button to refresh the list.
5. Select the image (.ffu). We will use **SampleFlash.ffu** located under **Lab1** folder.



1. Click “Flash” to flash the image to the SD card. Select **Continue** when prompt to erase content.



1. Deployment is done when the pop-up appear. Click **OK**.



1. Eject the SD card from the development PC.

## Section 2: Power up the MinnowBoard Max.

In this section you will add support for signals to the adapter by adding an event handler.

1. Insert the SD card into MinnowBoard Max.
2. Press the power button to power on MinnowBoard Max. Initial boot may take a few minutes to complete. Afterwards, the boot time is usually about 30 seconds.

[Continue on next page…]

## Section 3: Use Web Browser to Check Device Information

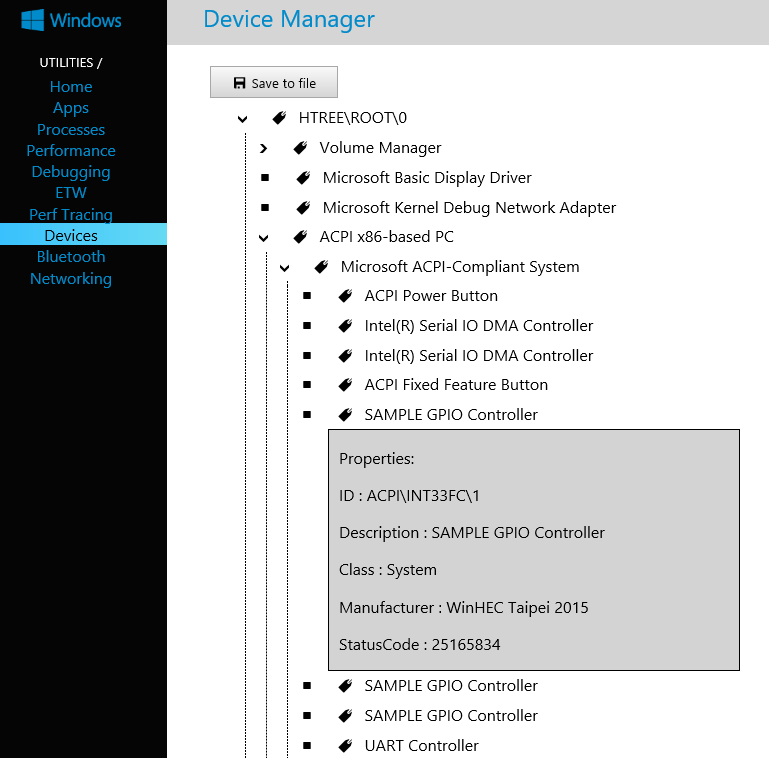
In this section you will use web browser to verify that our sample GPIO controller is running in Windows 10 IoT Core.

1. Launch **WindowsIoTCoreWatcher**. Find your IoT device in the list and make sure the IP Address displayed in the Watcher is correct.
2. Right-click on your device and select **Web Browser Here**. Use the following information to log in:

Username: Administrator

Password: p@ssw0rd

1. Select **Devices** on the left menu. Expand **ACPI x86-based PC** > **Microsoft ACPI-Compliant System**. Device node **SAMPLE GPIO Controller** should appear in the list. Click on the TA icon to see device properties. There is no Error Code listed in the device properties, so we good to go.



# [Optional]Exercise 6: Configure Default App for Windows IoT Core

In this exercise, we will use the web browser to deploy an UWP and set the UWP to default so it will automatically go to the UWP when device reboots.

## Section 1: Deploy a App using Web Browser.

In this section you will add support for signals to the adapter by adding an event handler.

1. Open a Web Browser, enter “**<your MBM IP Addresss>:8080**” in the address bar. Use the following information to log-in:

Username: Administrator

Password: p@ssw0rd

1. Select **Apps** on the left menu to go to **AppX Manager**.
2. Under the **Install App** Section, select the specified filed under “C:\Users\Admin\Desktop\Lab1\SampleApp”:

For AppX, select **SampleApp\_1.0.0.0\_x86\_Debug.appxbundle**.

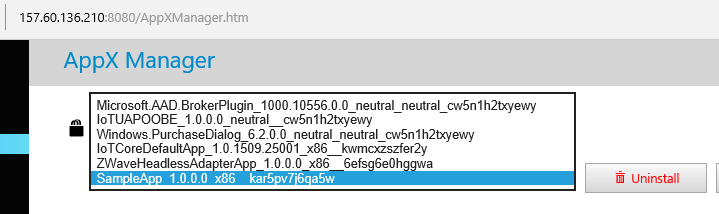
For Certificate, **select SampleApp\_1.0.0.0\_x86\_Debug.cer**.

1. For Dependencies, click **+ Add dependency** to add the two dependency files under “C:\Users\Admin\Desktop\Lab1\SampleApp\Dependencies\x86”: **Microsoft.NET.CoreRuntime.1.0.appx** and **Microsoft.VCLibs.x86.Debug.14.00.appx**.
2. Click **Install** button under **Actions** to install the app.
3. If the installation fails, click **Reset** button under **Actions**.
4. Repeat Step 3 and 5 (exclude step 4).

## Section 2: Set the Default App

In this section you will set the sample app as the default startup app.

1. Refresh the web browser.
2. Under **Installed Apps,** select the item that start with “**SampleApp…**”.



1. Click **Set Default** button. The sample UWP should start running.
2. Reboot MinnowBoard Max.
3. The sample UWP should start automatically when device comes back from reboot.