Connect Microchip Technology’s “PIC-IoT” Development Board to Azure IoT Services

Introduction

**About this Document**

This document describes how to connect the Microchip Technology “PIC-IoT” Development Board to Azure IoT services (e.g. IoT Explorer & IoT Central) using Azure IoT Plug-and-Play Device models.

IoT Plug and Play certified device simplifies the process of building devices without custom device code. Using Solution builders can be integrated quickly using the certified IoT Plug and Play enabled device based on Azure IoT Central as well as third-party solutions.

This “Getting Started Guide” provides step by step instructions on getting the device provisioned to Azure IoT Hub using Device Provisioning Service (DPS) and then using both IoT Explorer and IoT Central to interact with the device's Plug and Play capabilities.

Diagram

Description automatically generated

### **Features of the “PIC-IoT” Development Board**

Microchip’s “PIC-IoT” Development Board combines a powerful PIC24FJ128GA705 MCU, an ATECC608A CryptoAuthentication™ secure element IC, and the fully-certified ATWINC1510 Wi-Fi® network controller module - which provides the most simple and effective way to connect your embedded application to the Cloud. The board also includes an on-board debugger and requires no external hardware to program and debug the MCU.

The fully programmable flash-based PIC24F MCU enables you to quickly connect and send data to the Azure Cloud using the on-board temperature and light sensors. Two mechanical buttons and 4 LEDs are all used to simulate telemetry events and to demonstrate read-only and writable device properties. Once you are ready to build your own custom design, you can easily generate code using the free software libraries from Microchip’s MPLAB Code Configurator (MCC).

A picture containing shape

Description automatically generated

The PIC-IoT W<x> (where <x> = ‘A’ or ‘G’) development board is distributed/marketed using 2 variant part numbers, both which work for connecting to Azure IoT services when reprovisioned with Microchip software development tools & Microsoft’s Embedded C SDK for Azure IoT:

1. PIC-IoT WA Development Board (Microchip Part Number EV54Y39A)
2. PIC-IoT WG Development Board (Microchip Part Number AC164164)

A picture containing diagram

Description automatically generated

Prepare the Device for IoT Explorer

**Development Environment Setup**

IoT Plug and Play Certification is certifying specific device code implementation against a specific certified device model. Device builders should either pre-install device code or make the binary downloadable.

1. Follow the online procedure ending with successfully adding an enrollment group (i.e. up until the beginning of Section 4) : [Connecting the PIC-IoT Development Kit to Azure IoT](https://github.com/Azure-Samples/Microchip-PIC-IoT-Wx)
2. Clone the MPLAB X demo project by issuing the following commands in a Git Bash window (e.g. in Windows, open the **Start** menu by clicking on the Windows icon and typing “Git Bash” into the search bar)

git clone https://github.com/Azure-Samples/Microchip-PIC-IoT-Wx.git

cd Microchip-PIC-IoT-Wx

git submodule update –init

1. Launch the MPLAB X IDE and navigate to the main toolbar’s **File** 🡪 **Open Project** operation to load the demo project (\*.X) located at:

[path]\Microchip-PIC-IoT-Wx\AzureIotPnpDps.X

1. Set the AzureIotPnpDps.X project as the main (currently focused/active) project by right-clicking on it and selecting Set as Main Project

Graphical user interface, text, application

Description automatically generated

1. Modify header file AzureIotPnpDps\Header Files\platform\config\conf\_winc.h with your wireless router’s SSID and password

A picture containing text

Description automatically generated

1. Build the AzureIotPnpDps.X project and program the target device (perform both actions in a single operation by clicking on the Toolbar icon Make and Program Target). If there are any errors while building the project, they must be addressed before the board can be successfully programmed. Be sure that the MPLAB X “Output” window states that programming has been completed before proceeding to the next step

Graphical user interface, application

Description automatically generated

1. Launch a terminal emulator window and connect to the COM port corresponding to the PIC-IoT board at 9600 bps. In the terminal window, hit [RETURN] to get the list of available commands for the Command Line Interface (CLI). The Command Line Interface allows you to send simple ASCII-string commands to set or get the user-configurable operating parameters of the application while it is running. The CLI prompt is simply the . character

Graphical user interface, text, application

Description automatically generated

1. Look up the ID Scope for your DPS (either from the IoT\_Sensor\_Node\_config.h header file from the other AzureIotDps.X demo project or through your account on the Microsoft Azure Portal). The ID Scope is programmed/saved into the PIC-IoT board in the next step

Graphical user interface, text, application, email

Description automatically generated

1. In the terminal emulator window, confirm that “local echo” is enabled in the terminal settings. At the CLI prompt, type in the idscope {ID-scope} command to set the ID Scope (which gets saved in the ATECC608A secure element on the PIC-IoT board) and then hit [ENTER]. The ID Scope can be read out from the board by issuing the idscope command (without specifying any parameter)

Graphical user interface, text

Description automatically generated

Get ID Scope

Set ID Scope

1. Using the CLI prompt, type in the command reset and hit [ENTER].
2. Wait for the PIC-IoT board to connect to your DPS (allow up to 2 minutes); eventually the Blue and Green LEDs should both stay constantly on. If the Red LED comes on, then something was incorrectly programmed (e.g. firmware and/or specific device configurations).
3. To enable the “full” debug messaging output to the terminal emulator window, execute the command debug 4 on the Command Line Interface (CLI). To disable the debug messages at any time, execute the command debug 0 (debug levels range from 0 to 4). The CLI is always active, even while debug messages are being continuously displayed on the terminal window.

Integration with Azure IoT Explorer

1. In the Azure IoT Explorer window, click on the Home link near the top of the window

A picture containing graphical user interface, text, application

Description automatically generated

1. On the left-hand side of the IoT Explorer window, click on IoT Plug and Play Settings

Graphical user interface, text, application

Description automatically generated

1. Click on Pick a folder and select the model sub-folder which contains the model definition JSON-LD file (e.g. “PIC\_IoT\_WM\_v1.json”)

Graphical user interface, text, application

Description automatically generated

1. Click on Save

Graphical user interface, text, application

Description automatically generated

1. On the left-hand side of the IoT Explorer window, click on IoT hubs

Graphical user interface, application

Description automatically generated

1. Verify that the name of your IoT hub is displayed, then click on View devices in this hub

Graphical user interface, application

Description automatically generated

1. Verify that your device ID is displayed (and status is enabled), then click on it

Graphical user interface, text, application, email

Description automatically generated

1. On the left-hand side of the IoT Explorer window, click on IoT Plug and Play components

Graphical user interface, text, application, email

Description automatically generated

1. Click on Default component near the bottom of the IoT Explorer window

Graphical user interface, text, application, email

Description automatically generated

1. Click on Properties (read-only) near the top of the IoT Explorer window

Graphical user interface, text, application, email

Description automatically generated

1. Confirm that the “Enum” value of each LED matches the state observed on the PIC-IoT board (1 = On, 2 = Off, 3 = Blinking) and note the maximum temperature reading since device reset

Graphical user interface, text, application, email

Description automatically generated

1. Click on Properties (writable) near the top of the IoT Explorer window
2. Click on the input field labeled led\_yellow and select Blink

Graphical user interface, text, application

Description automatically generated

1. Click on Update desired value

Graphical user interface, text, application

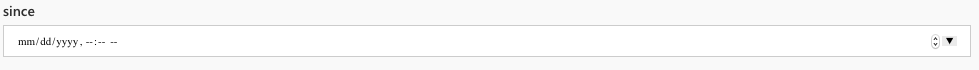
Description automatically generated

1. Observe the notification that the request to write the property was accepted by your device, and that the Yellow LED on the PIC-IoT board is blinking/toggling/flashing

Text

Description automatically generated

1. Click on Commands near the top of the IoT Explorer window
2. To set up the getMaxMinReport command, hover the mouse near the right corners of the since input field until the little triangle shows up and then click on it



1. When the calendar window pops up, select today’s date

A picture containing calendar

Description automatically generated

1. Type in the current time

Graphical user interface

Description automatically generated

1. Click on Send command

Graphical user interface, text, application, chat or text message

Description automatically generated

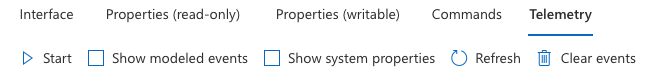
1. Confirm that the command was successfully invoked and that both the max & min temperatures are reported in the message that appears in the pop-up window (once the message window disappears, it can be read again by clicking on Notifications)



Text

Description automatically generated

1. Click on Telemetry near the top of the IoT Explorer window and then click on Start



1. Observe the telemetry data (for the temperature and light sensors) is updating every few seconds

Text

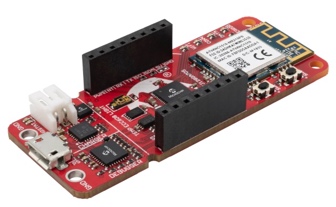
Description automatically generated

1. Increase the ambient light shining on top of the board and observe that the value of the light sensor increases within a few seconds

Text, letter

Description automatically generated

1. On the PIC-IoT Wx development board, press and release user button SW0 and/or SW1



2 Mechanical User Buttons

1. Observe the button event message that is generated each time a user button has been pressed/released

Text, letter

Description automatically generated

1. Click on Commands near the top of the IoT Explorer window



1. Click on the input field for delay and type PT10S, then click Send command. Confirm that the command was successfully invoked via a notification message, and then the board resets itself in approximately 10 seconds (the LEDs on the board will cycle and re-initialize)

Graphical user interface, text, application, chat or text message

Description automatically generated

1. Experiment with the CLI commands to control the LED’s from the terminal emulator window by issuing the led command (usage: led <led>,<state>)

|  |  |
| --- | --- |
| **<led>** | **Translation** |
| 1 | Blue |
| 2 | Green |
| 3 | Yellow |
| 4 | Red |
| **<state>** | **Translation** |
| 1 | On |
| 2 | Off |
| 3 | Start Blinking |
| 4 | Stop Blinking |
| **Command** | **Result** |
| led 1,1 | Blue LED = On |
| led 1,2 | Blue LED = Off |
| led 1,3 | Blue LED 🡪 Start Blinking |
| led 1,4 | Blue LED 🡪 Stop Blinking |
| led 2,1 | Green LED = On |
| led 2,2 | Green LED = Off |
| led 2,3 | Green LED 🡪 Start Blinking |
| led 2,4 | Green LED 🡪 Stop Blinking |
| led 3,1 | Yellow LED = On |
| led 3,2 | Yellow LED = Off |
| led 3,3 | Yellow LED 🡪 Start Blinking |
| led 3,4 | Yellow LED 🡪 Stop Blinking |
| led 4,1 | Red LED = On |
| led 4,2 | Red LED = Off |
| led 4,3 | Red LED 🡪 Start Blinking |
| led 4,4 | Red LED 🡪 Stop Blinking |

Connect to Azure IoT Central

Follow the below steps to create an X.509 enrollment group in an Azure IoT Central application and then see your device send its telemetry data to the Dashboard.

1. Create a custom IoT Central application from scratch [Azure IoT Central - My apps](https://apps.azureiotcentral.com/myapps) (or even better – get a jump start by using the existing application template provided for the PIC-IoT Development Board [PIC-IoT Application Template](https://apps.azureiotcentral.com/build/new/1f71e467-74da-4554-8444-c650de86e4df) and skip forward to Step 4)
2. Click on the button **Build an app** (or **+ New application**)
3. Click on the image titled **Custom apps**

Graphical user interface, application

Description automatically generated

1. Enter the desired settings for your IOT Central application and click on the **Create** button. Refer to the online tutorial for additional details: [Create an Azure IoT Central application](https://docs.microsoft.com/en-us/azure/iot-central/core/quick-deploy-iot-central)
2. Create an X.509 enrollment group for your IoT Central application. Open your IoT Central application and navigate to **Administration** in the left pane and select **Device connection**
3. Select **+ Create enrollment group**, and create a new enrollment group using any name (Group type = **IoT devices**, attestation type = **Certificates (X.509)**). Hit **Save** when finished:

Graphical user interface, text, application, email

Description automatically generated

1. Now that the new enrollment group has been created, select **+ Manage Primary**.
2. Select the file/folder icon associated with the **Primary** field and upload the root certificate file root-ca.crt. The message **(!) Needs verification** should appear. The **Subject** and **Thumbprint** fields will automatically populate themselves.
3. Click **Generate verification code** (this code will be copied to the clipboard which will be needed in a future step)
4. Open a Git Bash window: Start menu 🡪 type Git Bash, a window like this will pop up:

Text

Description automatically generated

1. Change to your generated certification folder:

cd drive\<your\_path>\.microchip-iot

For example:

cd /C/Users/john5/Azure/.microchip-iot

1. Execute the below command in the Git Bash window (copy and paste for best results):

**Note**: Once you enter the below command, you will then be asked to enter information for various fields that will be incorporated into your certificate request. Enter the verification code (which was just generated previously) when prompted for the Common Name. It's recommended to just copy the Verification code to the clipboard and paste it when it's time to enter the Common Name. For the rest of the fields, you can enter anything you want (or just hit [ENTER] to keep them blank which is fine for basic demonstration purposes). If you accidentally hit [ENTER] when asked for the Common Name, you will need to run the command all over again...

openssl req -new -key root-ca.key -out azure\_root\_ca\_verification.csr

1. Generate the verification certificate by executing the following command (copy and paste the entire command for best results):

openssl x509 -req -in azure\_root\_ca\_verification.csr -CA root-ca.crt -CAkey root-ca.key -CAcreateserial -out azure\_signer\_verification.cer -days 365 -sha256

1. Click **Verify** and select the azure\_signer\_verification.cer file to upload. Confirm that the **Primary** certificate has been verified and that a **Thumbprint** has been generated for your certificate. The X.509 enrollment group should be ready to go!

Graphical user interface, text, application

Description automatically generated

Prepare the Device for IoT Central

Before proceeding with this section, confirm that your device already connects to your existing Azure IoT Hub successfully and is communicating the correct telemetry data to the IoT Explorer GUI tool.

1. Launch a terminal emulator window and connect to the COM port corresponding to the PIC-IoT board at 9600 bps. In the terminal window, hit [RETURN] to get the list of available commands for the Command Line Interface (CLI).

Graphical user interface, text, application

Description automatically generated

1. Look up the ID Scope for the DPS created/used by your IoT Central application (using the left-hand navigation pane, select **Administration** 🡪 **Device connection**). The ID Scope will be programmed/saved into the PIC-IoT board in the next step

Graphical user interface, application, Teams

Description automatically generated

1. In the terminal emulator window, confirm that “local echo” is enabled in the terminal settings. At the CLI prompt, type in the idscope <ID-scope> command to set it (which gets saved in the ATECC608A secure element on the PIC-IoT board) and then hit [ENTER]. The ID Scope can be read out from the board by issuing the idscope command without any parameter

Graphical user interface, text

Description automatically generated

Get ID Scope

Set ID Scope

1. Using the CLI prompt, type in the command reset and hit [ENTER].
2. Wait for the PIC-IoT board to connect to your IoT Central’s DPS (allow up to 2 minutes); eventually the Blue and Green LEDs should both stay constantly on. If the Red LED comes on, then something was incorrectly programmed (e.g. ID scope was entered incorrectly).
3. Go back to your web browser to access the Azure IoT Central application. Use the left-hand side pane and select **Devices** 🡪 **All Devices**. Confirm that your device is listed – the device name & ID is the Common Name of the device certificate (sn<XXXXXXXXXXXXXXXXX>):

Graphical user interface, application

Description automatically generated

1. Change the Device name by clicking on it and then editing the box at the top of the page

Graphical user interface, text, application

Description automatically generated

1. Confirm that the **Blue** & **Green LED states** are both set to “**Turn On**”

Graphical user interface, text, application

Description automatically generated

1. Click on **Overview** under your device name to see the telemetry displays being updated every few seconds:

Graphical user interface, application

Description automatically generated

1. Click on **Commands** under your device name. Go to the Get Max-Min report box and click on the calendar icon



1. Select today’s date, then click **Run**

Graphical user interface, text, application, email

Description automatically generated

1. Click on **command history**, then **View** **Payload**

Diagram

Description automatically generated

1. Confirm that valid data shows up in the Response payload:

Graphical user interface, text, application

Description automatically generated

1. Click on **Raw data** under your device name to see the raw telemetry messages being received every few seconds:

Graphical user interface, application

Description automatically generated

1. Follow the procedure [Configure the IoT Central application dashboard](https://docs.microsoft.com/en-us/azure/iot-central/core/howto-add-tiles-to-your-dashboard) to create a customized dashboard for your IoT Central application. The below screen captures show examples of dashboards that highlight the telemetry data facilitated by the PIC-IoT Wx Development Board based on its Plug and Play interface:

Graphical user interface, application, website

Description automatically generated

Graphical user interface

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, application

Description automatically generated

Additional Links

Please refer to the below links for additional information for Plug and Play and IoT Central

* [Manage cloud device messaging with Azure-IoT-Explorer](https://github.com/Azure/azure-iot-explorer/releases)
* [Import the Plug and Play model](https://docs.microsoft.com/en-us/azure/iot-pnp/concepts-model-repository)
* [Configure to connect to IoT Hub](https://docs.microsoft.com/en-us/azure/iot-pnp/quickstart-connect-device-c)
* [How to use IoT Explorer to interact with the device](https://docs.microsoft.com/en-us/azure/iot-pnp/howto-use-iot-explorer#install-azure-iot-explorer)
* [Create an Azure IoT Central application](https://docs.microsoft.com/en-us/azure/iot-central/core/quick-deploy-iot-central)
* [Manage devices in your Azure IoT Central application](https://docs.microsoft.com/en-us/azure/iot-central/core/howto-manage-devices)
* [How to connect devices with X.509 certificates for IoT Central](https://docs.microsoft.com/en-us/azure/iot-central/core/how-to-connect-devices-x509)
* [Configure the IoT Central application dashboard](https://docs.microsoft.com/en-us/azure/iot-central/core/howto-add-tiles-to-your-dashboard)
* [Customize the IoT Central UI](https://docs.microsoft.com/en-us/azure/iot-central/core/howto-customize-ui)

A close up of a logo

Description automatically generated

Microchip logo

A picture containing drawing

Description automatically generated