

Device Update Sample

Microchip SAM E54 Xplained Pro using MPLAB X IDE

User Guide

Published: November 2022

For the latest information, please see   
azure.com/rtos

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.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

Azure RTOS provides OEMs with components to secure communication and to create code and data isolation using underlying MCU/MPU hardware protection mechanisms. It is ultimately the responsibility of the device builder to ensure the device fully meets the evolving security requirements associated with its specific use case.

 © 2022 Microsoft. All rights reserved.

Microsoft Azure RTOS, Azure RTOS FileX, Azure RTOS GUIX, Azure RTOS GUIX Studio, Azure RTOS NetX, Azure RTOS NetX Duo, Azure RTOS ThreadX, Azure RTOS TraceX, Azure RTOS Trace, event-chaining, picokernel, and preemption-threshold are trademarks of the Microsoft group of companies. All other trademarks are property of their respective owners.

Revision 6.2

Table of Contents

[Overview 4](#_Toc119937233)

[Prepare Azure Resources 7](#_Toc119937234)

[Create an IoT Hub and register a device 7](#_Toc119937235)

[Create Device Update account 7](#_Toc119937236)

[Prepare the new firmware 8](#_Toc119937237)

[Build new firmware 8](#_Toc119937238)

[Prepare new firmware for the simulated leaf device 10](#_Toc119937239)

[Generate import manifest 10](#_Toc119937240)

[Upload firmware and manifest 11](#_Toc119937241)

[Prepare the device 12](#_Toc119937242)

[Modify the configuration 12](#_Toc119937243)

[Download and run the project 12](#_Toc119937244)

[Deploy new firmware 14](#_Toc119937245)

[Add a tag to your device 14](#_Toc119937246)

[Deploy update 14](#_Toc119937247)

[Clean up resources 16](#_Toc119937248)

[Next steps 17](#_Toc119937249)

# Overview

The following steps detail how to configure, build and execute the [Device Update for IoT Hub](https://docs.microsoft.com/azure/iot-hub-device-update/) example on the Microchip SAM E54 Xplained Pro. For this sample, we will learn:

* Create Azure IoT Hub and Device Update for IoT Hub resources.
* Prepare the firmware for the dev board and the manifest file required for it.
* Prepare the firmware for the simulated leaf device for [proxy update](https://docs.microsoft.com/azure/iot-hub-device-update/device-update-proxy-updates) and manifest file required for it.
* Upload the files and create the group for the device in the Device Update for IoT Hub.
* Deploy the firmware to the device and observe the result of it.

The tool uses in the sample is MPLAB X IDE 6.0.0 development tools. It can be downloaded from this page: <https://www.microchip.com/mplab/mplab-x-ide>.

You will also need MPLAB XC32/32++ Compiler 4.1.0 or later to build the sample projects. It can be downloaded from this page: <https://www.microchip.com/mplab/compilers>.

A circuit board

Description automatically generated

Figure Microchip SAM E54 Xplained Pro

The sample distribution zip file contains the following sub-folders:

|  |  |
| --- | --- |
| Folder | Contents |
| *common\_hardware\_code* | Common code for SAM E54 board |
| *docs* | User guides |
| *netxduo* | NetX Duo source code |
| *sample\_azure\_iot\_embedded\_sdk\_pnp* | Sample project to connect to Azure IoT Hub using Azure IoT Middleware for Azure RTOS |
| *same54\_lib* | SAM E54 drivers |
| *threadx* | ThreadX source code |
| *filex* | FileX source code |

# Prepare Azure Resources

## Create an IoT Hub and register a device

Follow the same steps in ***Azure\_RTOS\_ATSAME54-XPRO\_Azure\_IoT\_Embedded\_SDK\_For\_MPLAB.pdf***.

## Create Device Update account

Follow this guide to create a device update account using Azure portal: <https://learn.microsoft.com/azure/iot-hub-device-update/create-device-update-account>

# Prepare the new firmware

To connect the device to Azure, you'll modify a configuration file for Azure IoT settings, build and flash the image to the device.

## Build new firmware

1. Open MPLab and select ***File > Open Project*** and select the following projects from the extracted zip file.
   * threadx
   * netxduo
   * same54\_lib
   * sample\_azure\_iot\_embedded\_sdk\_pnp
2. Select the sample\_azure\_iot\_embedded\_sdk\_pnp sample, right click on it on in the left ***Projects*** pane and select ***Set as main project***.
3. Expand the Header files folder to open sample\_config.h to set the Azure IoT device information constants to the values that you saved after you created Azure resources.

|  |  |
| --- | --- |
| Constant name | Value |
| HOST\_NAME | {Your IoT hub hostName value} |
| DEVICE\_ID | {Your deviceID value} |
| DEVICE\_SYMMETRIC\_KEY | {Your primaryKey value} |

1. In the same folder, open ***sample\_azure\_iot\_embedded\_sdk\_adu.c,*** modify the firmware version to mimic it is a new firmware that will be deployed from Device Update.

#define SAMPLE\_DEVICE\_INSTALLED\_CRITERIA "1.1.0"

#define SAMPLE\_LEAF\_DEVICE\_INSTALLED\_CRITERIA "1.1.0"

1. Make sure all sample project’s dependent libraries (threadx, netxduo, same54\_lib) are built by selecting the project in the ***Projects*** pane, right click on it and select ***Build***.

Graphical user interface, text, application

Description automatically generated

1. MPLAB X [does not create .bin file by default](https://microchipsupport.force.com/s/article/MPLAB-X-IDE---Generating-binary-file---xc32-Compiler#:~:text=Answer.%20MPLAB%20X%20does%20not%20create%20.bin%20file,build%20%E2%80%99%20option%20and%20insert%20the%20following%20command.). Right click on sample\_azure\_iot\_embedded\_sdk\_pnp project, select Properties > Building, check Execute this line after build and insert the following command:

${MP\_CC\_DIR}\xc32-objcopy -I ihex -O binary "${DISTDIR}/${PROJECTNAME}.${IMAGE\_TYPE}.hex" "${DISTDIR}/${PROJECTNAME}.${IMAGE\_TYPE}.bin"

Graphical user interface, text, application, email

Description automatically generated

1. The binary firmware will be generated in ***mplab\sample\_azure\_iot\_embedded\_sdk\_pnp\*** ***dist\default\production.*** Copy ***.bin*** file to ***mplab\tools\AzureDeviceUpdateScripts*** folder where the scripts for generating manifest files are within. It is optional but recommended also change the file name that matches with your firmware version such as ***firmware\_1.1.0.bin****.*

Now you have prepared the new firmware for the dev board that will be uploaded to Device Update for IoT Hub.

## Prepare new firmware for the simulated leaf device

Even though we do not physically connect any leaf devices to the dev board we are using. But it is a common scenario for your MCU device that you might have one or more leaf devices connect to it via UART or USB as peripherals or sensors. For this sample, we just use a simulated leaf device and push an empty firmware to demonstrate the experience. To do that:

Create a file and name it as ***leaf\_firmware\_1.1.0.bin*** and save in the same ***mplab/tools/AzureDeviceUpdateScripts*** folder with this PowerShell command:

New-Item "leaf\_firmware\_1.1.0.bin" -ItemType File -Value "Empty leaf firmware"

## Generate import manifest

An [import manifest](https://docs.microsoft.com/azure/iot-hub-device-update/import-concepts) is a JSON file that defines important information about the update that you are importing that is required by the Device Update for IoT Hub. You can learn the detailed steps about importing new update from [here](https://docs.microsoft.com/azure/iot-hub-device-update/import-update). For this sample:

1. Ensure you have installed [PowerShell v7.0](https://github.com/PowerShell/PowerShell/releases/tag/v7.0.3) or above.
2. In PowerShell, navigate to ***mplab\tools\******AzureDeviceUpdateScripts*** directory and run:

Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process

1. Run the following script and follow the prompt to generate the manifest files for the dev board and the simulated leaf device:

.\CreateSAME54Update.ps1

Supply values for the following parameters:

(Type !? for Help.)

Version: 1.1.0

HostPath: ./firmware\_1.1.0.bin

LeafPath: ./leaf\_firmware\_1.1.0.bin

Replace the values with the actual path and version you have for the firmware. You can also type !? for the help.

1. All files required to upload will be generated in ***MICROCHIP.SAME54.1.1.0*** folder.

## Upload firmware and manifest

1. Open the IoT Hub you created before with Device Update enabled from Azure portal.
2. Select ***Device management > Updates***, and in ***Updates*** tab select ***Import a new update***.
3. Choose ***Select from storage container****.* For the storage container, you can create a new or use existing storage container to host the firmware file.
4. Select ***Upload*** and choose all firmware and manifest files you will use for the deployment in ***MICROCHIP.SAME54.1.1.0*** folder
5. You can go to ***Import history*** to see the progress of publishing the files. And once done, it will show in the ***Available updates*** tab.

Now you have the new firmware and manifest files uploaded for deployment. It’s time to make the device be up and running to receive the new firmware.

# Prepare the device

We will build and run the same device application but on an older version, so later we can observe the new firmware deployed to it.

## Modify the configuration

1. In MPLab, open ***sample\_azure\_iot\_embedded\_sdk\_adu.c***, modify version to an older one to mimic current firmware:

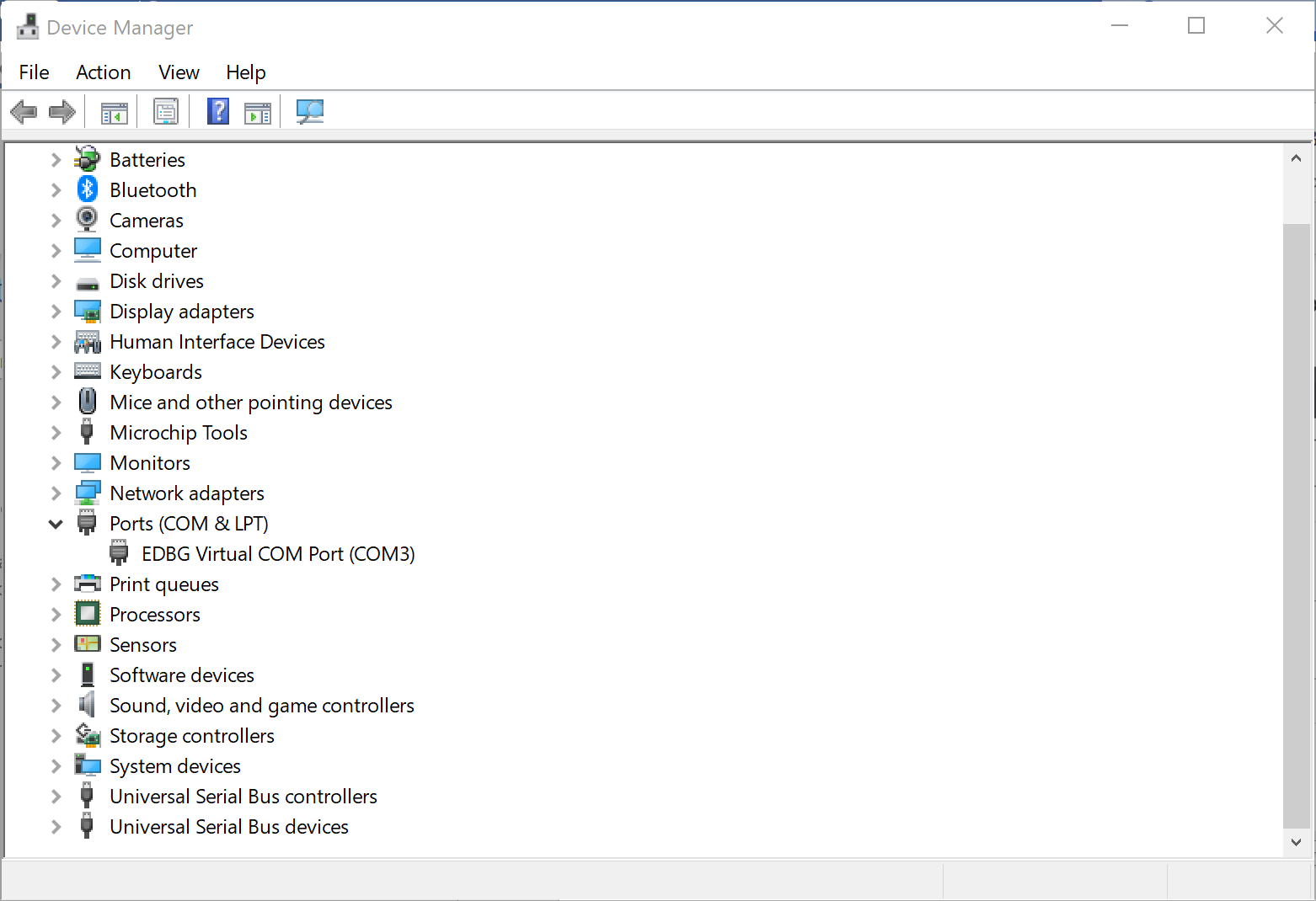
#define SAMPLE\_DEVICE\_INSTALLED\_CRITERIA "1.0.0"

#define SAMPLE\_LEAF\_DEVICE\_INSTALLED\_CRITERIA "1.0.0"

1. Now rebuild the ***sample\_azure\_iot\_embedded\_sdk\_pnp*** project.

## Download and run the project

1. Use the Micro USB cable to connect the Debug USB port on the Microchip SAM E54, and then connect it to your computer.
2. Use the Ethernet cable to connect the Microchip SAM E54 to an Ethernet port.
3. In MPLab, select ***Debug > Debug Main Project***.
4. Verify the serial port in your OS’s device manager. It should show up as a COM port.



1. Open your favorite serial terminal program such as [Putty](https://www.chiark.greenend.org.uk/~sgtatham/putty/) or [Tera Term](https://ttssh2.osdn.jp/index.html.en) and connect to the COM port discovered above. Configure the following values for the serial ports:

Baud rate: ***115200***

Data bits: ***8***

Stop bits: ***1***

1. As the project runs, the demo prints out status information to the terminal output window. Check the terminal output to verify that messages have been successfully sent to the Azure IoT hub and the version number of the firmware is running.

NOTE: The terminal output content varies depending on which sample you choose to build and run.

Text

Description automatically generated

Keep terminal window open to monitor device output in subsequent steps.

# Deploy new firmware

## Add a tag to your device

1. Keep the device application running from the previous step.
2. Log into Azure portal and navigate to the IoT Hub.
3. From ***IoT Devices***, select the IoT device you use and navigate to ***Device Twin*** tab.
4. Delete any existing Device Update tag value by setting them to null.

"tags": {

"ADUGroup": null

}

1. Add a new Device Update tag value:

"tags": {

"ADUGroup": "<CustomTagValue>"

}

Graphical user interface, text, application

Description automatically generated

## Deploy update

1. In the same ***Groups and Deployments*** tab, next to the group you just created, choose ***Deploy***.
2. Leave the default option as ***Start immediately*** and choose ***Create*** to start a deployment.
3. Go back to terminal output window, you can see the update firmware is pushed from ADU to the device and after downloading it, the device will reboot with the new firmware:

Text

Description automatically generated

Text

Description automatically generated

Since the simulated leaf device just does nothing. The terminal output will only show that the empty firmware is deployed to the device and device will echo back to it.

# Clean up resources

If you no longer need the Azure resources created in this tutorial, you can use the Azure CLI to delete the resource group and all the resources you created for this tutorial. Optionally, you can use Azure IoT Explorer to delete individual resources including devices and IoT hubs.

If you continue to another tutorial in this getting started guide, you can keep the resources you've already created and reuse them.

Important: Deleting a resource group is irreversible. The resource group and all the resources contained in it are permanently deleted. Make sure that you do not accidentally delete the wrong resource group or resources.

To delete a resource group by name:

1. Run the [az group delete](https://docs.microsoft.com/cli/azure/group?view=azure-cli-latest#az-group-delete) command. This removes the resource group, the IoT Hub, and the device registration you created.

az group delete --name MyResourceGroup

1. Run the [az group list](https://docs.microsoft.com/cli/azure/group?view=azure-cli-latest#az-group-list) command to confirm the resource group is deleted.

az group list

# Next steps

In this tutorial you created an IoT Hub and added the Device Update resource to it. Then you prepared and deployed the new firmware for the dev board and the simulated leaf device.

To learn more about the APIs of the Device Update agent for Azure RTOS, or the Device Update for IoT Hub service, view <https://aka.ms/azrtos/adu>

To learn more about Azure RTOS and how it works with Azure IoT, view <https://azure.com/rtos>.