



# Azure RTOS Samples for Microchip SAM E54 Xplained Pro using IAR User Guide

Published: November 2022

For the latest information, please see  
[azure.com/rtos](https://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.

FileX supports the Microsoft exFAT file system format. Your use of exFAT technology in your products requires a separate license from Microsoft. Please see the following link for further details on exFAT licensing:

<https://www.microsoft.com/en-us/legal/intellectualproperty/mtl/exfat-licensing.aspx>

© 2022 Microsoft. All rights reserved.

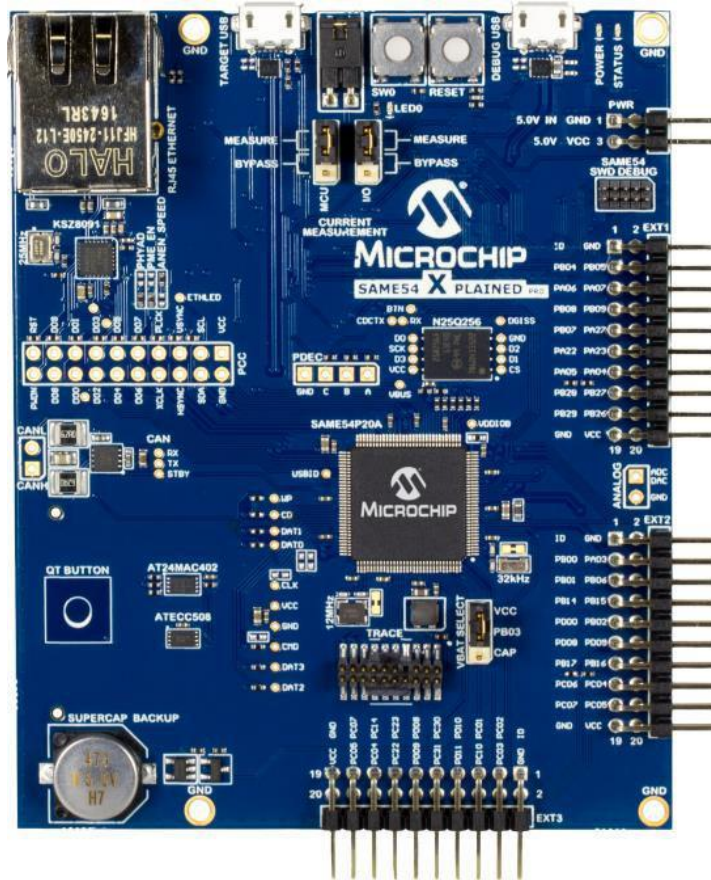
Microsoft Azure RTOS, Azure RTOS FileX, Azure RTOS GUIX, Azure RTOS GUIX Studio, Azure RTOS NetX, Azure RTOS NetX Duo, Azure RTOS ThreadX, Azure RTOS TraceX, Azure RTOS Trace, event-chaining, picokernel, and preemption-threshold are trademarks of the Microsoft group of companies. All other trademarks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

Revision 6.2

# Table of Contents

- Overview ..... 1
- Getting Started ..... 4
- Sample Descriptions ..... 5
  - Azure RTOS IoT Embedded SDK Samples ..... 5
  - ThreadX Sample ..... 6
  - FileX RAM Disk Sample ..... 7
  - NetX Duo Simple Ping Sample ..... 9
  - NetX Duo Iperf Throughput Sample ..... 12

# Overview



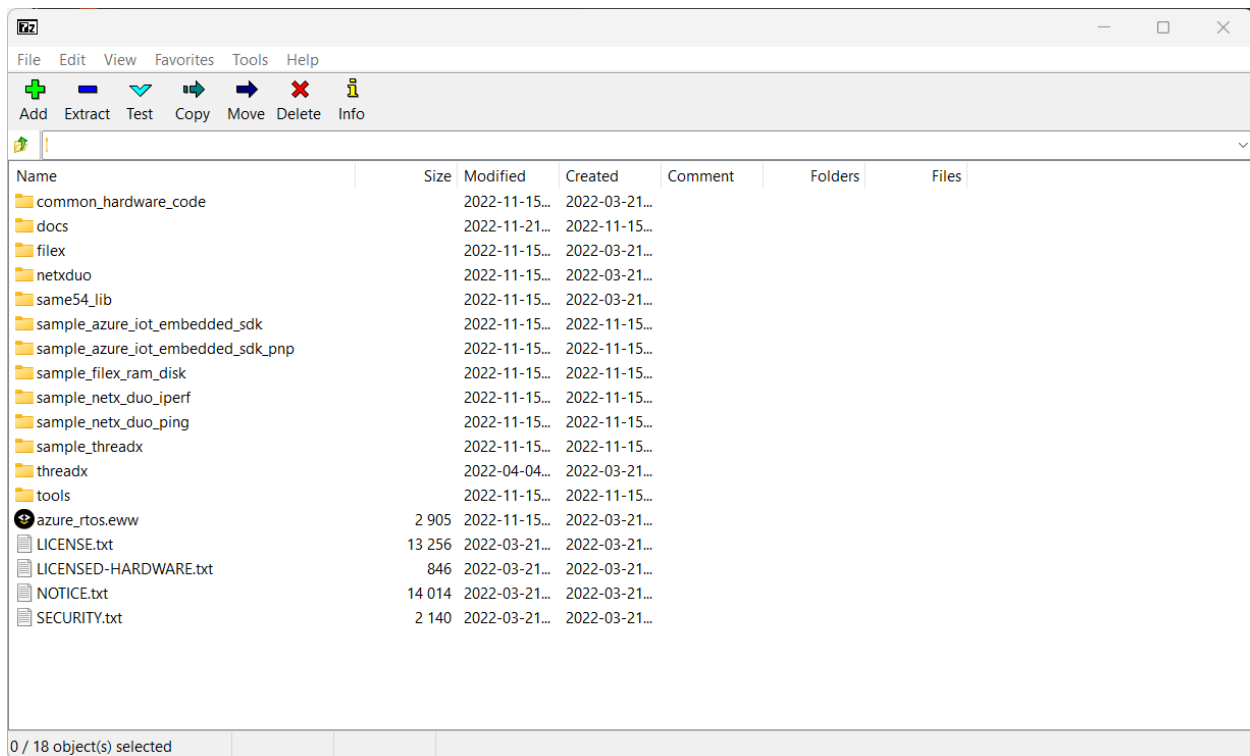
Microchip SAM E54 Xplained Pro board

Azure RTOS Samples for each component (Azure connectivity, ThreadX, FileX, GUIX, NetX Duo, and USBX) are designed to run on the ATSAME54-XPRO “out-of-the-box.” Each sample project is described later in this document along with links to further information as necessary.

All samples are designed to run using the IAR Embedded Workbench for ARM (EWARM) 9.20.0 or later, with the on-board Atmel's Embedded Debugger (EDBG) (Debug USB port). The default factory jumper selections are assumed. A 30-day free trial of IAR's EWARM development tools can be downloaded from this page:

<https://www.iar.com/iar-embedded-workbench/#!?architecture=Arm>

The sample distribution zip file has following organization:



| Name                              | Size   | Modified      | Created       | Comment | Folders | Files |
|-----------------------------------|--------|---------------|---------------|---------|---------|-------|
| common_hardware_code              |        | 2022-11-15... | 2022-03-21... |         |         |       |
| docs                              |        | 2022-11-21... | 2022-11-15... |         |         |       |
| filex                             |        | 2022-11-15... | 2022-03-21... |         |         |       |
| netxduo                           |        | 2022-11-15... | 2022-03-21... |         |         |       |
| same54_iib                        |        | 2022-11-15... | 2022-03-21... |         |         |       |
| sample_azure_iot_embedded_sdk     |        | 2022-11-15... | 2022-11-15... |         |         |       |
| sample_azure_iot_embedded_sdk_pnp |        | 2022-11-15... | 2022-11-15... |         |         |       |
| sample_filex_ram_disk             |        | 2022-11-15... | 2022-11-15... |         |         |       |
| sample_netx_duo_iperf             |        | 2022-11-15... | 2022-11-15... |         |         |       |
| sample_netx_duo_ping              |        | 2022-11-15... | 2022-11-15... |         |         |       |
| sample_threadx                    |        | 2022-11-15... | 2022-11-15... |         |         |       |
| threadx                           |        | 2022-04-04... | 2022-03-21... |         |         |       |
| tools                             |        | 2022-11-15... | 2022-11-15... |         |         |       |
| azure_rtos.eww                    | 2 905  | 2022-11-15... | 2022-03-21... |         |         |       |
| LICENSE.txt                       | 13 256 | 2022-03-21... | 2022-03-21... |         |         |       |
| LICENSED-HARDWARE.txt             | 846    | 2022-03-21... | 2022-03-21... |         |         |       |
| NOTICE.txt                        | 14 014 | 2022-03-21... | 2022-03-21... |         |         |       |
| SECURITY.txt                      | 2 140  | 2022-03-21... | 2022-03-21... |         |         |       |

0 / 18 object(s) selected

The root directory contains the **azure\_rtos.eww** workspace as well as following sub-folders:

| Folder                                   | Contents                                                                                                   |
|------------------------------------------|------------------------------------------------------------------------------------------------------------|
| <i>iar</i>                               | Contains the following sub-folders as well as the <b>azure_rtos.eww</b> workspace                          |
| <i>common_hardware_code</i>              | Contains common code for ATSAME54-XPRO board                                                               |
| <i>docs</i>                              | Contains user guides and supporting documentation                                                          |
| <i>filex</i>                             | Contains FileX source code                                                                                 |
| <i>netxduo</i>                           | Contains NetX Duo and NetX Secure source code                                                              |
| <i>sample_azure_iot_embedded_sdk</i>     | Sample project to connect to Azure IoT Hub using Azure IoT Middleware for Azure RTOS                       |
| <i>sample_azure_iot_embedded_sdk_pnp</i> | Sample project to connect to Azure IoT Hub using Azure IoT Middleware for Azure RTOS via IoT Plug and Play |
| <i>sample_netx_duo_iperf</i>             | Contains NetX Duo iPerf sample project                                                                     |
| <i>sample_netx_duo_ping</i>              | Contains NetX Duo ping sample project                                                                      |
| <i>sample_threadx</i>                    | Contains ThreadX sample project                                                                            |
| <i>sample_filex_ram_disk</i>             | Contains sample for FileX RAM disk                                                                         |
| <i>same54_library</i>                    | Contains ATSAME54 drivers                                                                                  |
| <i>threadx</i>                           | Contains ThreadX source code                                                                               |

# Getting Started

- 1) Unpack the sample zip file into a folder of your choice, we recommend:

***C:\azure\_rtos\same54Xpro\iar***

NOTE: If you unzip the IAR project in a very long path, it might not open properly.

- 2) Open the ***azure\_rtos.eww*** EWARM Workspace, using IAR EWARM.
- 3) Select the desired sample project in the IAR workspace. There is a menu of available reference project tabs in the Project Pane of the EWARM IDE (as shown below). Click on the desired sample project and make the project active by selecting ***Active Project*** via a right-click.

The ***sample\_threadx*** project is shown above as the currently active project.

- 4) Select ***Project > Batch Build*** and choose ***build\_all*** and select ***Make*** to build all projects. You will observe compilation and linking of all sample projects.
- 5) Select ***Download and Debug*** to download and start execution of the project. By default, execution stops at a breakpoint set at ***main***.
- 6) Select ***Go*** to start execution of the demonstration. Please review the sample descriptions later in this guide for additional setup and expected behavior.

# Sample Descriptions

## Azure RTOS IoT Embedded SDK Samples

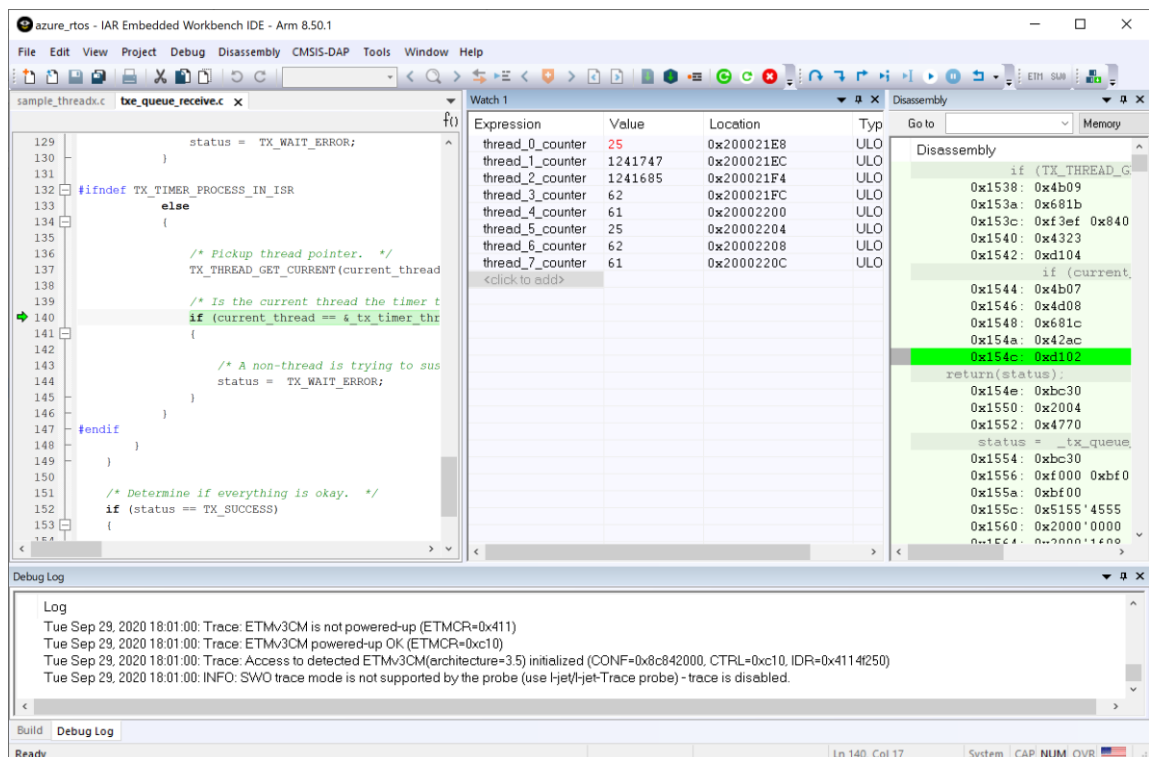
Please see the *Azure\_RTOS\_ATSAME54-XPRO\_Azure\_IoT\_Embedded\_SDK\_For\_IAR.pdf* for a detailed description of the sample as well as a step-by-step set of instructions on how to connect to Azure IoT via the Azure IoT Embedded C SDK and Azure IoT Middleware for Azure RTOS.

# ThreadX Sample

This sample is the standard 8-thread ThreadX example, that illustrates the use of the main ThreadX services, including threads, message queues, timers, semaphores, byte memory pools, block memory pools, event flag groups, and mutexes. This demonstration is fully described, including a source code listing, in Chapter 6 of the **Azure\_RTOS\_ThreadX\_User\_Guide.pdf** (also provided in this distribution).

To run the ThreadX Sample project, simply follow these steps (assuming the **azure\_rtos.ewww** workspace is already open):

1. Click on the **sample\_threadx** project and make the project active by selecting **Active Project** via a right-click.
2. Select **Make** button to build the **Active Project** selected. You will observe compilation and linking of the selected sample project.
3. Select **Download and Debug** to download and start execution of the demonstration. The sample will initially stop at **main**. Select another **Go** to execute the sample.



After hitting **Break** the IAR debugger screen shot above shows various counters incremented by the ThreadX sample as each of the main components of the ThreadX are exercised.

To learn more about Azure RTOS ThreadX, view <https://docs.microsoft.com/azure/rtos/threadx/>.

# FileX RAM Disk Sample

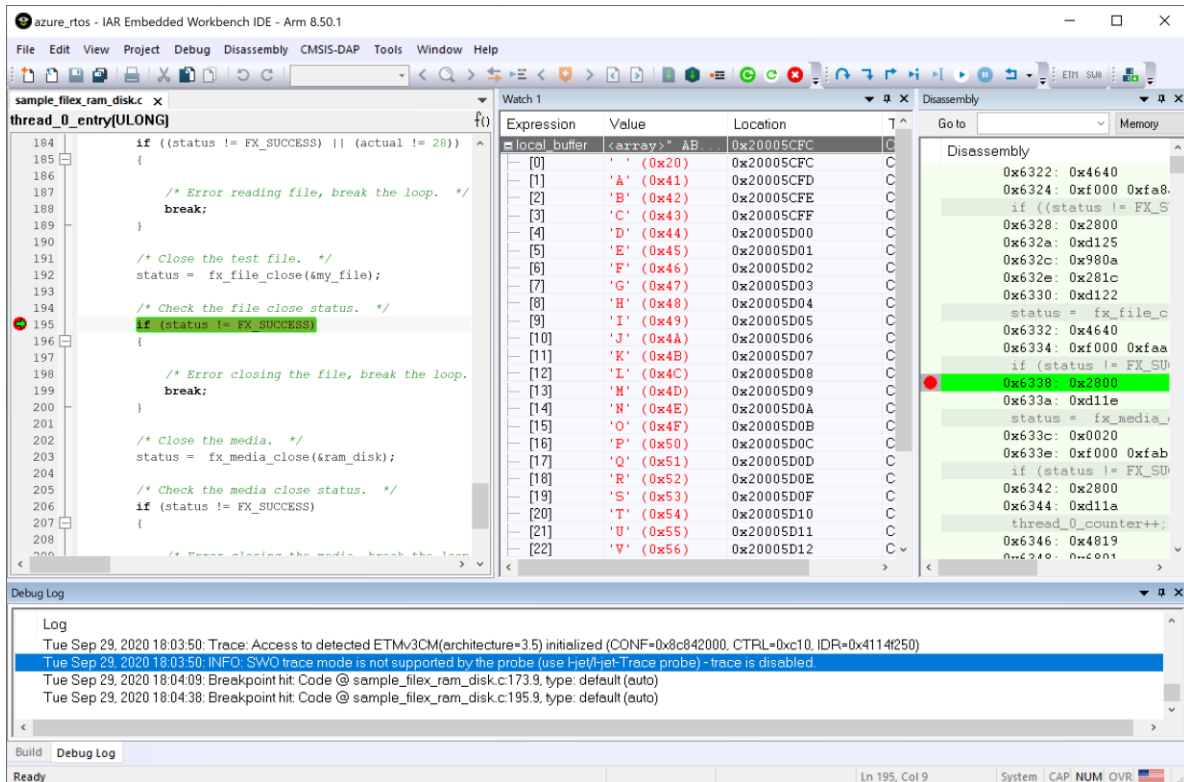
This sample illustrates the use of the FileX embedded FAT file system. The example creates a small RAM-disk with a sample file and data, and reads the file data back into memory. The debugger is able to show the data being read.

To run the FileX RAM Disk Sample project, simply follow these steps (assuming the **azure\_rtos.eww** workspace is already open):

1. Click on the **sample\_filex\_ram\_disk** project and make the project active by selecting **Active Project** via a right-click.
2. Open **sample\_filex\_ram\_disk.c** and set a breakpoint around Line 194 at *if (status != FX\_SUCCESS)*

You can turn on the line number by going to **Tools > Options > Editor > Show Line Numbers**.

3. Select **Make** button to build the **Active Project** selected. You will observe compilation and linking of the selected sample project.
4. Select **Download and Debug** to download and start execution of the demonstration. The sample will initially stop at **main**. Select another **Go** to execute the RAM disk sample to the point after the file read is complete.
5. In the Watch window, ensure you watch the **local\_buffer** variable as expression.



The IAR debugger screen shot above shows the file data read back in the RAM disk sample.

To learn more about Azure RTOS FileX, view <https://docs.microsoft.com/azure/rtos/filex/>.

# NetX Duo Simple Ping Sample

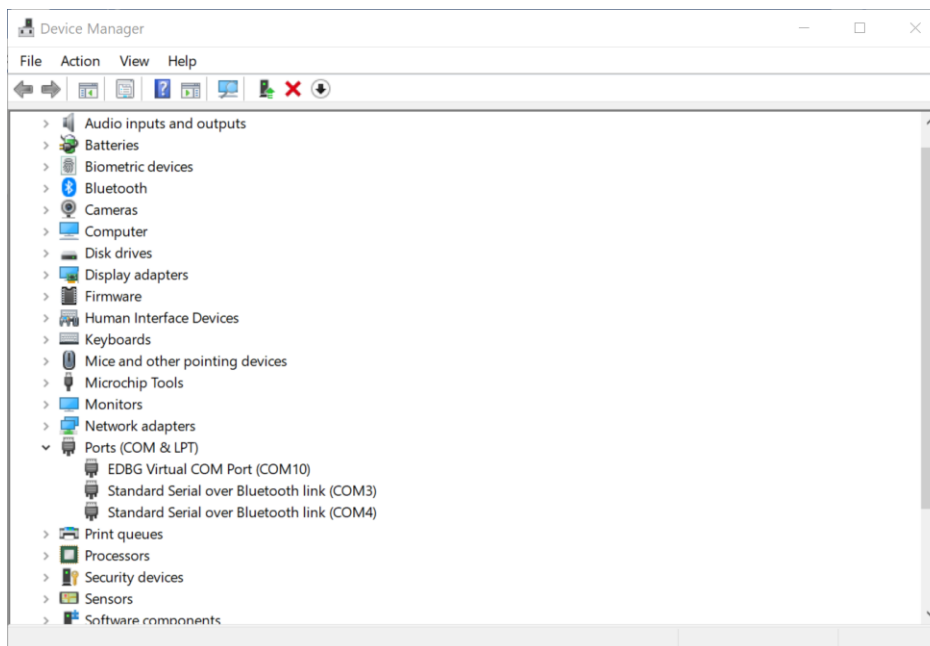
This sample project illustrates the setup and use of NetX Duo IPv4/IPv6 TCP/IP stack via ping from another node on the local network. By default, this demonstration requests an IP Address via DHCP, and displays the status and assigned IP Address via Terminal I/O in the IAR debugger. In addition, the retrieved IP address is displayed in the watch window of the IAR debugger.

To run the NetX Duo Ping Sample project, simply follow these steps (assuming the **azure\_rtos.eww** workspace is already open):

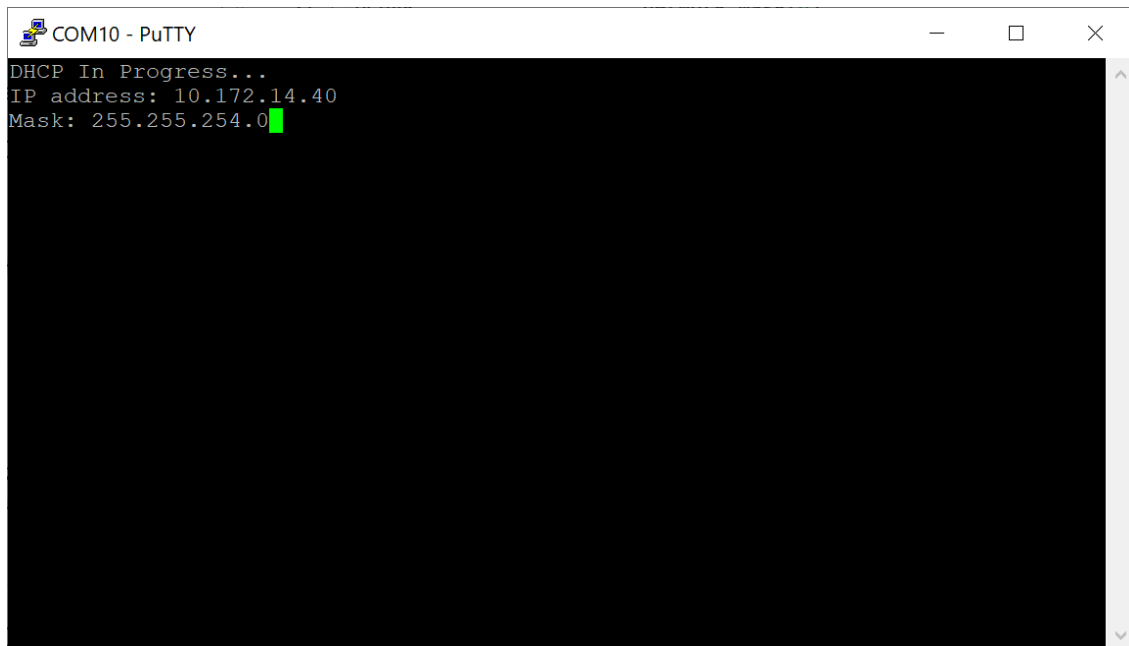
1. Click on the **sample\_netx\_duo\_ping** project and make the project active by selecting **Active Project** via a right-click.
2. Select **Make** button to build the **Active Project** selected. You will observe compilation and linking of the selected sample project.

*Note: This sample is Ethernet based and therefore assumes an Ethernet cable is connected to the Ethernet connector on the board.*

3. Select **Download and Debug** to download and start execution of the demonstration. The sample will initially stop at **main**. Select another **Go** to execute the sample.
4. Verify the serial port in your OS's device manager. It should show up as a COM port.

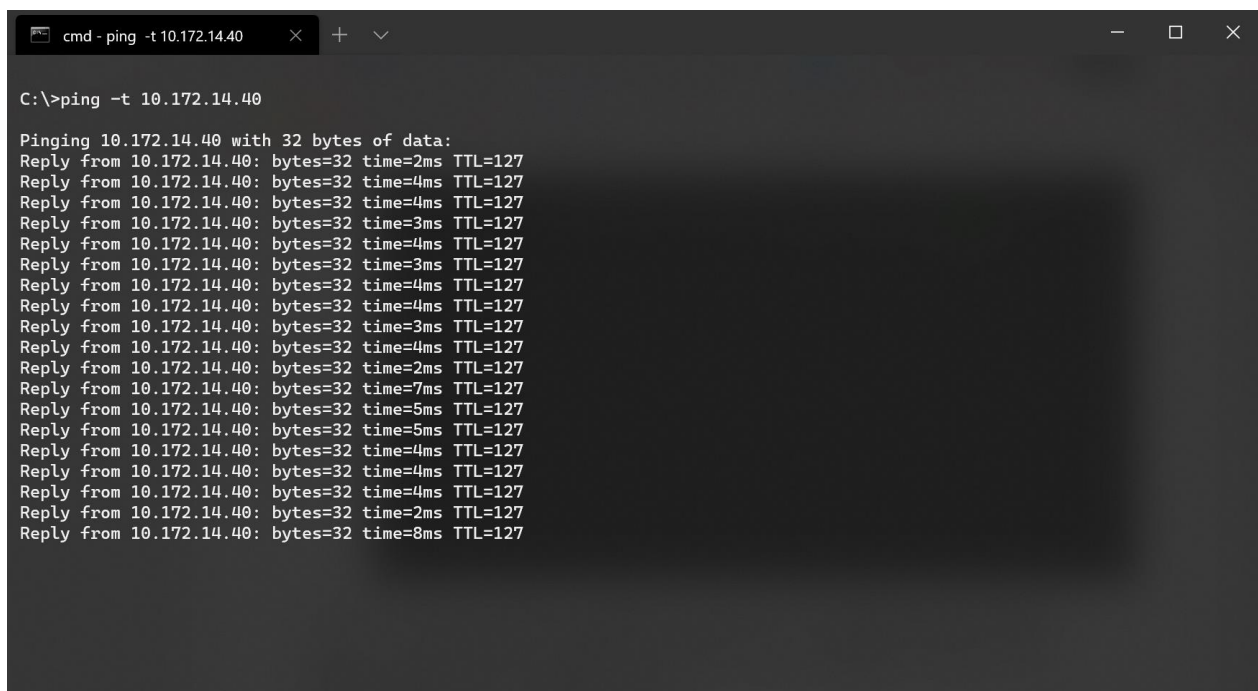


5. Open your favorite serial terminal program such as Putty and connect to the COM port discovered above. As the project runs you should observe the IP address assigned via DHCP in the output window.



```
COM10 - PuTTY
DHCP In Progress...
IP address: 10.172.14.40
Mask: 255.255.254.0
```

The example above shows that the assigned IP address of the ATSAME54-XPRO board is 10.172.14.40. When the demonstration is running it can be pinged by any machine on the network. The following is an example of a ping from a Windows machine on the same local network (using the DOS command window):



```
cmd - ping -t 10.172.14.40
C:\>ping -t 10.172.14.40

Pinging 10.172.14.40 with 32 bytes of data:
Reply from 10.172.14.40: bytes=32 time=2ms TTL=127
Reply from 10.172.14.40: bytes=32 time=4ms TTL=127
Reply from 10.172.14.40: bytes=32 time=4ms TTL=127
Reply from 10.172.14.40: bytes=32 time=3ms TTL=127
Reply from 10.172.14.40: bytes=32 time=4ms TTL=127
Reply from 10.172.14.40: bytes=32 time=3ms TTL=127
Reply from 10.172.14.40: bytes=32 time=4ms TTL=127
Reply from 10.172.14.40: bytes=32 time=4ms TTL=127
Reply from 10.172.14.40: bytes=32 time=3ms TTL=127
Reply from 10.172.14.40: bytes=32 time=4ms TTL=127
Reply from 10.172.14.40: bytes=32 time=2ms TTL=127
Reply from 10.172.14.40: bytes=32 time=7ms TTL=127
Reply from 10.172.14.40: bytes=32 time=5ms TTL=127
Reply from 10.172.14.40: bytes=32 time=5ms TTL=127
Reply from 10.172.14.40: bytes=32 time=4ms TTL=127
Reply from 10.172.14.40: bytes=32 time=4ms TTL=127
Reply from 10.172.14.40: bytes=32 time=4ms TTL=127
Reply from 10.172.14.40: bytes=32 time=2ms TTL=127
Reply from 10.172.14.40: bytes=32 time=8ms TTL=127
```

*Note: Static IP address assignment is also possible by disabling **`NX_ENABLE_DHCP`** in the project settings and modifying the default static IP address of **`192.2.2.149`** in the source file **`demo_netx_duo_ping.c`**.*

To learn more about Azure RTOS NetX Duo, view <https://docs.microsoft.com/azure/rtos/netx/>.

# NetX Duo Iperf Throughput Sample

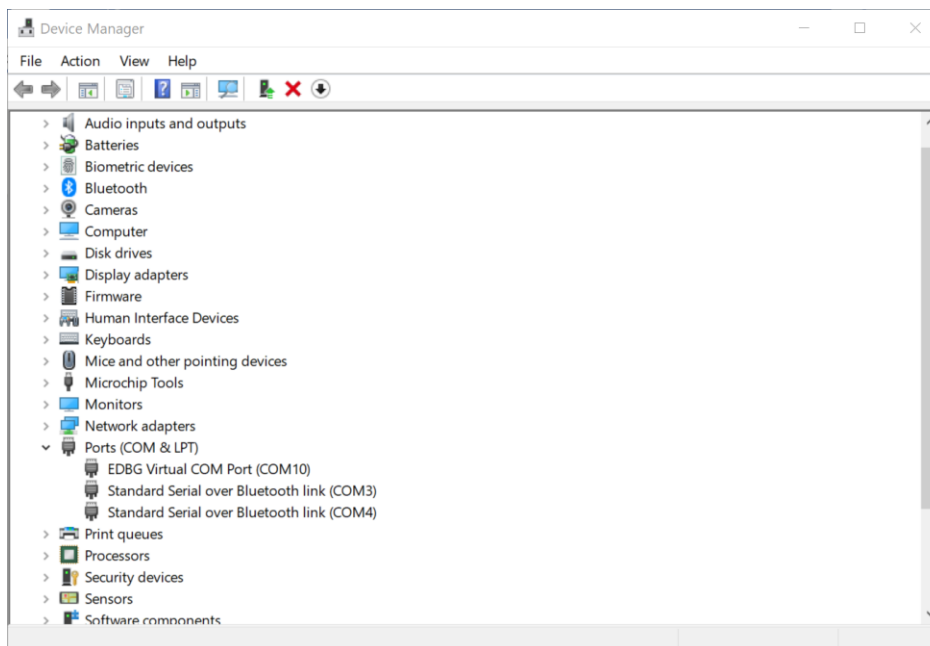
This demonstration illustrates TCP and UDP network throughput, using Express Logic's NetX Duo IPv4/IPv6 TCP/IP stack, and the industry-standard Iperf network throughput benchmark, with Jperf GUI. By default, this demonstration requests an IP Address via DHCP, and displays the status and assigned IP Address via Terminal I/O in the IAR debugger. In addition, the retrieved IP address is displayed in the watch window of the IAR debugger.

To run the NetX Duo Iperf Sample project, simply follow these steps (assuming the **azure\_rtos.eww** workspace is already open):

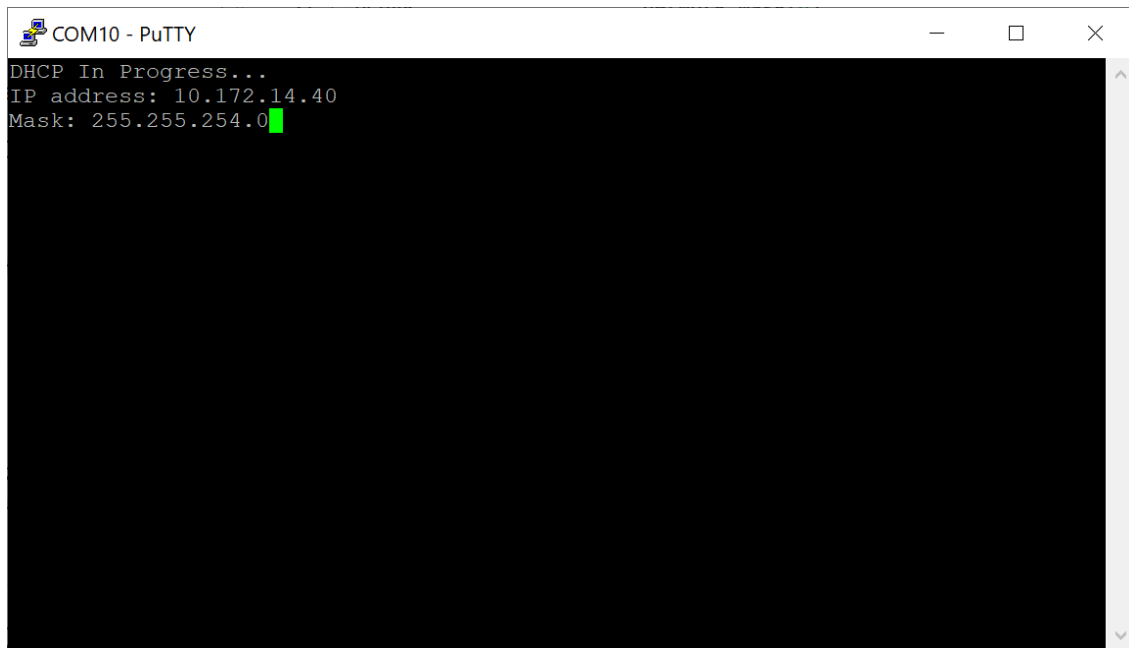
1. Click on the **sample\_netx\_duo\_iperf** project and make the project active by selecting **Active Project** via a right-click.
2. Select **Make** button to build the **Active Project** selected. You will observe compilation and linking of the selected sample project.

*Note: This sample is Ethernet based and therefore assumes an Ethernet cable is connected to the Ethernet connector on the board.*

3. Select **Download and Debug** to download and start execution of the demonstration. The sample will initially stop at **main**. Select another **Go** to execute the sample.
4. Verify the serial port in your OS's device manager. It should show up as a COM port.

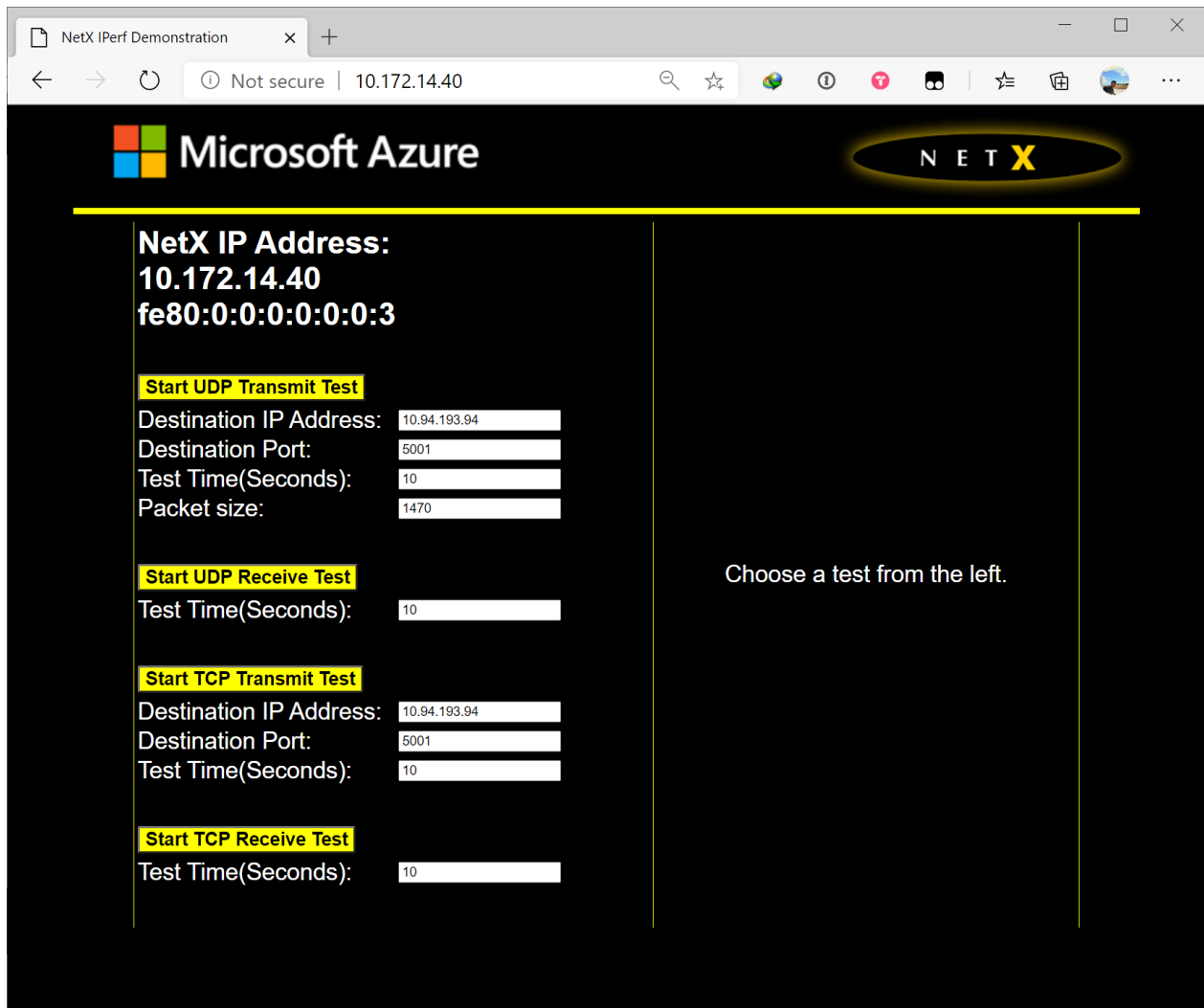


5. Open your favorite serial terminal program such as Putty and connect to the COM port discovered above. As the project runs you should observe the IP address assigned via DHCP in the output window.



```
COM10 - PuTTY
DHCP In Progress...
IP address: 10.172.14.40
Mask: 255.255.254.0
```

Once running, simply browse to target IP address (in the screen shot above it is 10.172.14.40) to view the NetX Duo Iperf server page, which provides options for running each Iperf test as well as displays the results of each test. Here is as sample view after browsing 10.172.14.40:



You will now need to setup and run Jperf on a Windows host on the same local network. To learn how to use the Jperf with Iperf sample, view ***Azure\_RTOS\_NetX\_Duo\_Iperf\_User\_Guide.pdf***.

*Note: Static IP address assignment is also possible by disabling **`NX_ENABLE_DHCP`** in the project settings and modifying the default static IP address of **`192.2.2.149`** in the source file **`sample_netx_duo_iperf.c`**.*

To learn more about Azure RTOS NetX Duo, view <https://docs.microsoft.com/azure/rtos/netx/>.