NXP Semiconductor, Inc.

i.MXRT With Panasonic 9026 WiFi module - Hands on Lab Guide

Rev 1.0, 07/2020

Author(s): Damian Wells, Massimo Incerti, Ralf Lehmann

Introduction

NXP Semiconductors **i.MX RT1060** Crossover Processor is based on the Arm® Cortex-M7 MPCore™ Platform, operating at up to 600MHz to provide high CPU performance and excellent real-time response. The i.MX RT1060 processor has 1MB of on-chip RAM. 512KB can be flexibly configured as TCM or general purpose on-chip RAM, while the other 512KB is general-purpose on-chip RAM. The i.MX RT1060 integrates advanced power management module with DC-DC and LDO that reduces complexity of external power supply and simplifies power sequencing.

The i.MX RT1060 Crossover Processor provides various memory interfaces, including SDRAM, RAW NAND FLASH, NOR FLASH, SD/eMMC, Quad SPI, and a wide range of other interfaces for connecting peripherals, such as WLAN, Bluetooth™, GPS, displays, and camera sensors. The i.MX RT1060 offers a rich audio and video feature set, including LCD display, basic 2D graphics, camera interface, SPDIF, and I^2S audio interface. The i.MX RT1060 also has analogue interfaces, including ADC, ACMP, and TSC.

This device is fully supported by NXP's MCUXpresso Software and Tools, a comprehensive and cohesive set of free software development tools for Kinetis, LPC, and i.MX RT microcontrollers. MCUXpresso SDK also includes project files for Keil MDK and IAR Embedded Workbench for Arm

The Panasonic Industrial **PAN9026** is a 2.4 GHz and 5 GHz ISM band Wi-Fi and Bluetooth radio module based on the NXP 88W977 SOC, which includes a wireless radio for easy integration of Wi-Fi and Bluetooth connectivity into various electronic devices.

In this hands-on lab guide we will walk through the steps required to get the PAN9026WiFi module up and running with the MIMXRT1060-EVK.
NOTE: These instructions can be easily adapted to cover the IMXRT1050-EVKB as well through the 2.8 version of the MCUXPresso SDK.

Prerequisites

A WiFi Access Point with public SSID and known password.
1 spare USB port on PC
1 x MIMXRT1060-EVK
1 x PAN9026 WiFi Module
1 x SD to uSD Card adaptor
1 x USB Key with Lab Software (Optional)

To complete this entire Lab series you will need to download and install the software listed overleaf:

Alternately we recommend that you take a copy from the class provided USB stick now for use during the class.

Required Software

1) MCUXpresso V11.2 Download link (nxp.com account needed)
2) MCUXpresso SDK for IMXRT1060 V2.8 accessed from the SDK Builder web page (See Lab 1, Section 1.1.1 for how to configure SDK builder if you are not familiar with its use)
3) Iperf Tool, download page: https://iperf.fr/iperf-download.php

Table of Contents

1.0 Software Installation Instructions

This lab will cover the initial software installation required to run further labs.

Required Equipment

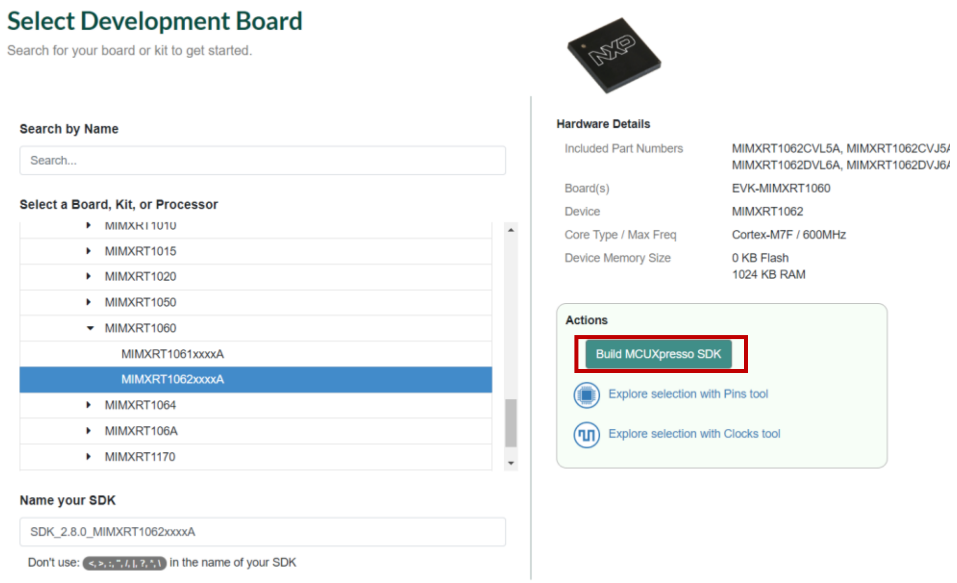
MCUXpresso IDE software download saved to your PC (see prerequisites)
MCUXpresso SDK software download saves to your PC (see prerequisites)

1.1 Using SDK Builder to download the MCUXpresso SDK (Optional)

This section describes how to manually download the most recent MCUXpresso SDK for the i.MXRT1060 using the SDK Builder from NXP's website.

1. Open your favorite web browser. Browse to https://mcuxpresso.nxp.com and sign-in with your NXP account (Create an account if you do not have one yet). Click on "Select Development Board"

2. From under "**Select a Device, Board or Kit**", select **Processors->i.MX**->**RT->MIMXRT1060** Then browse the list of available devices and select the **MIMXRT1062xxxxA** device. Then click on **Build MCUXpresso SDK**.



Select Development Board
Search for your board or kit to get started.

Search by Name
Search...

Select a Board, Kit, or Processor
- MIMXRT1010
- MIMXRT1015
- MIMXRT1020
- MIMXRT1050
- MIMXRT1060
  - MIMXRT1061xxxxA
  - MIMXRT1062xxxxA
- MIMXRT1064
- MIMXRT106A
- MIMXRT1170

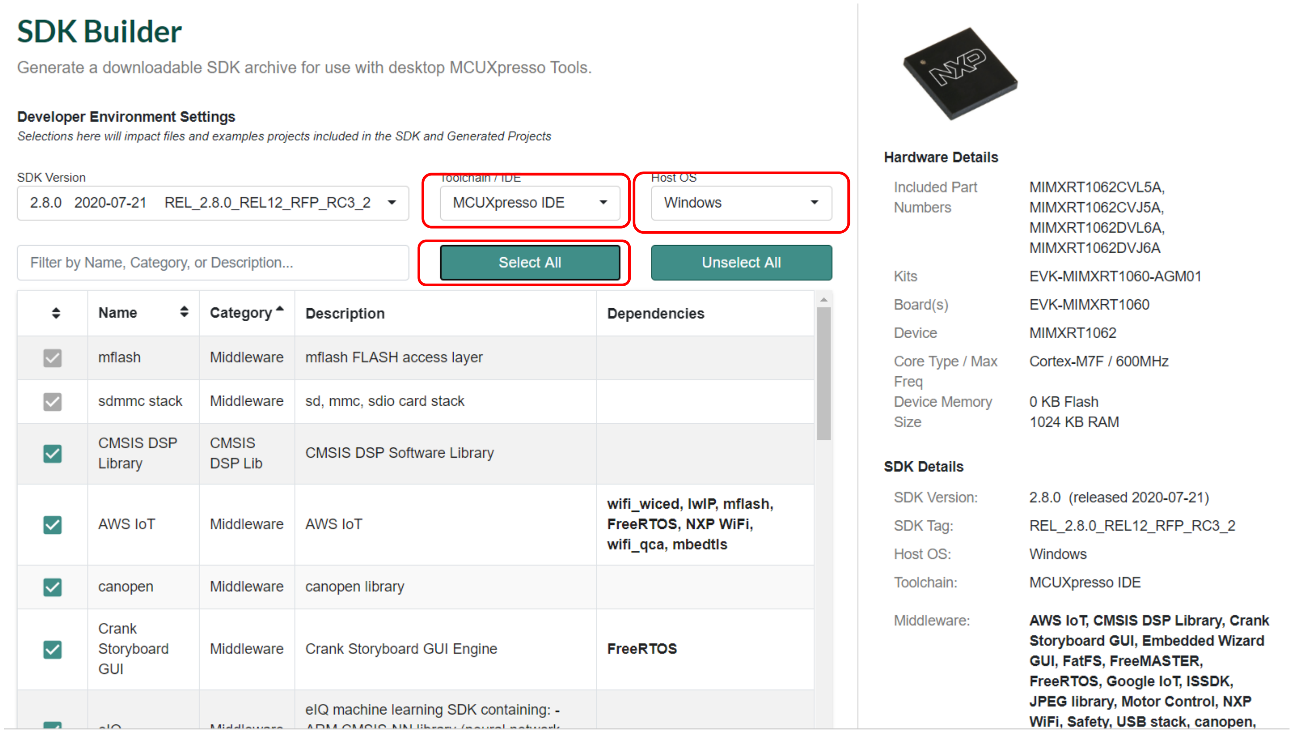
Name your SDK
SDK_2.8.0_MIMXRT1062xxxxA
Don't use: <, >, :, ", /, |, ?, *, \ in the name of your SDK

Hardware Details

Included Part Numbers	MIMXRT1062CVL5A, MIMXRT1062CVJ5A MIMXRT1062DVL6A, MIMXRT1062DVJ6A
Board(s)	EVK-MIMXRT1060
Device	MIMXRT1062
Core Type / Max Freq	Cortex-M7F / 600MHz
Device Memory Size	0 KB Flash 1024 KB RAM

Actions
Build MCUXpresso SDK
Explore selection with Pins tool
Explore selection with Clocks tool

3. From the **SDK Builder** menu, make sure that **Windows** is selected as a Host OS, **MCUXpresso IDE** is selected as Toolchain/IDE and **Select All** has been highlighted to select all SDK components. Then click the **Download SDK** to download the MCUXpresso SDK for the IMXRT1060 EVK.



SDK Builder

Generate a downloadable SDK archive for use with desktop MCUXpresso Tools.

Developer Environment Settings

Selections here will impact files and examples projects included in the SDK and Generated Projects

SDK Version: 2.8.0 2020-07-21 REL_2.8.0_REL12_RFP_RC3_2

Toolchain / IDE: MCUXpresso IDE

Host OS: Windows

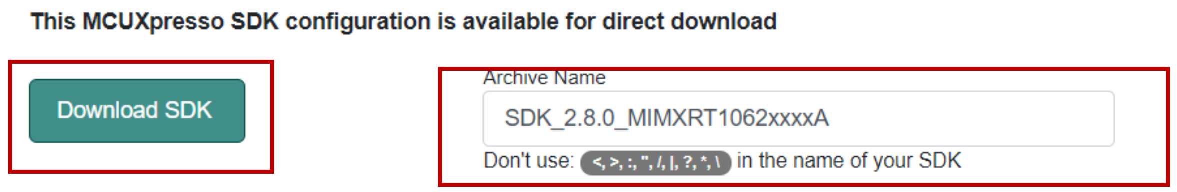
	Name	Category	Description	Dependencies
✓	mflash	Middleware	mflash FLASH access layer	
✓	sdmmc stack	Middleware	sd, mmc, sdio card stack	
✓	CMSIS DSP Library	CMSIS DSP Lib	CMSIS DSP Software Library	
✓	AWS IoT	Middleware	AWS IoT	wifi_wiced, lwIP, mflash, FreeRTOS, NXP WiFi, wifi_qca, mbedtls
✓	canopen	Middleware	canopen library	
✓	Crank Storyboard GUI	Middleware	Crank Storyboard GUI Engine	FreeRTOS
			eIQ machine learning SDK containing: -	

Hardware Details

Included Part Numbers	MIMXRT1062CVL5A, MIMXRT1062CVJ5A, MIMXRT1062DVL6A, MIMXRT1062DVJ6A
Kits	EVK-MIMXRT1060-AGM01
Board(s)	EVK-MIMXRT1060
Device	MIMXRT1062
Core Type / Max Freq	Cortex-M7F / 600MHz
Device Memory Size	0 KB Flash 1024 KB RAM

SDK Details

SDK Version:	2.8.0 (released 2020-07-21)
SDK Tag:	REL_2.8.0_REL12_RFP_RC3_2
Host OS:	Windows
Toolchain:	MCUXpresso IDE
Middleware:	AWS IoT, CMSIS DSP Library, Crank Storyboard GUI, Embedded Wizard GUI, FatFS, FreeMASTER, FreeRTOS, Google IoT, ISSDK, JPEG library, Motor Control, NXP WiFi, Safety, USB stack, canopen,



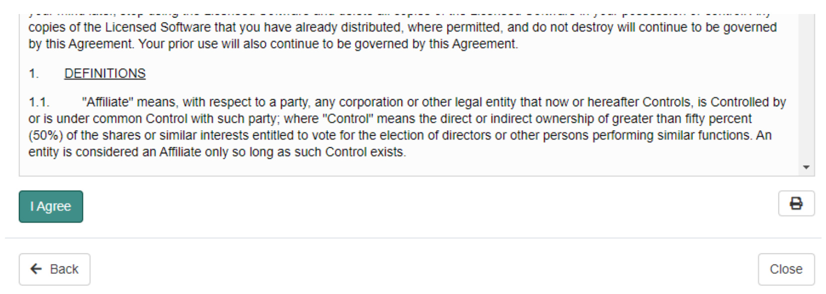
This MCUXpresso SDK configuration is available for direct download

Archive Name

SDK_2.8.0_MIMXRT1062xxxxA

Don't use: <, >, :, ", /, |, ?, *, \ in the name of your SDK

4. Accept the Software Terms and Conditions



copies of the Licensed Software that you have already distributed, where permitted, and do not destroy will continue to be governed by this Agreement. Your prior use will also continue to be governed by this Agreement.

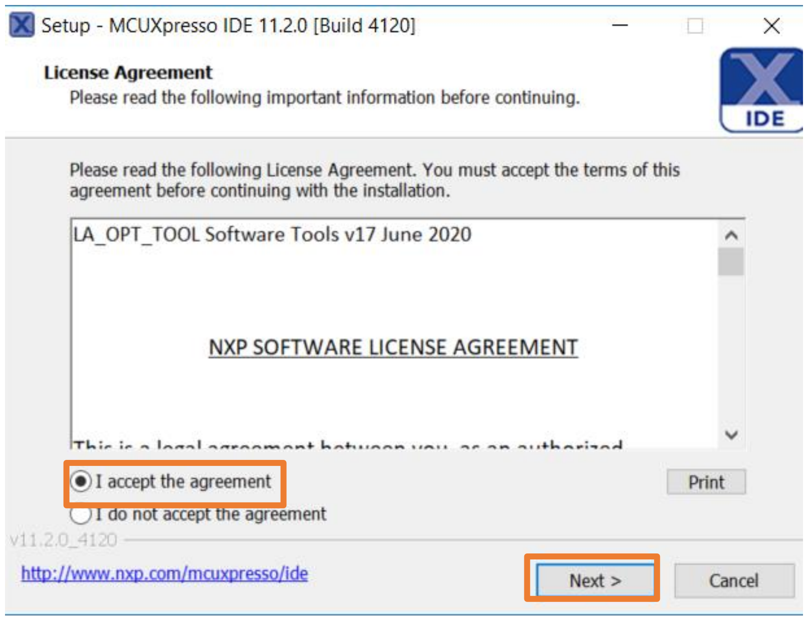
1. DEFINITIONS

1.1. "Affiliate" means, with respect to a party, any corporation or other legal entity that now or hereafter Controls, is Controlled by or is under common Control with such party; where "Control" means the direct or indirect ownership of greater than fifty percent (50%) of the shares or similar interests entitled to vote for the election of directors or other persons performing similar functions. An entity is considered an Affiliate only so long as such Control exists.

5. Select a location on your local PC to save your SDK file and click on **Save**

1.2 Installing MCUXpresso IDE

1. Locate and open the folder where you saved the software downloaded above.
2. Double click on ***MCUXpressoIDE_11.2.0_4120.exe***
3. Accept the license agreement and click [**Next**]


Setup - MCUXpresso IDE 11.2.0 [Build 4120]

License Agreement
Please read the following important information before continuing.

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

LA_OPT_TOOL Software Tools v17 June 2020

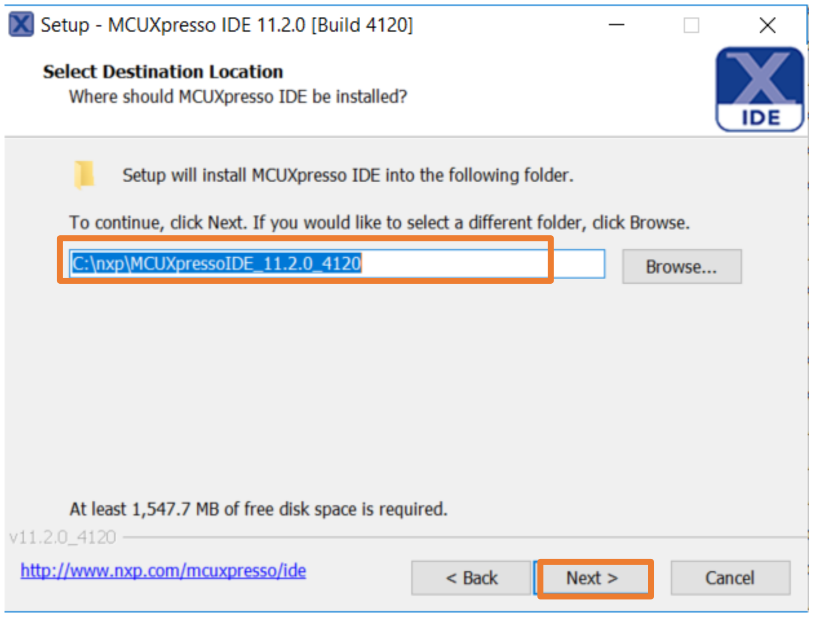
NXP SOFTWARE LICENSE AGREEMENT

This is a legal agreement between you, as an authorized

[✓ I accept the agreement]
I do not accept the agreement

4. On the **Information** dialog click [**Next**]

5. At the ***Set Destination*** dialog choose a destination for installing the IDE. We highly recommend you to keep the installation path simple. Here we choose the destination folder as ***NXP***. You can install the IDE on any of your hard-drives. Click [**Next**] to continue.


Setup - MCUXpresso IDE 11.2.0 [Build 4120]

Select Destination Location
Where should MCUXpresso IDE be installed?

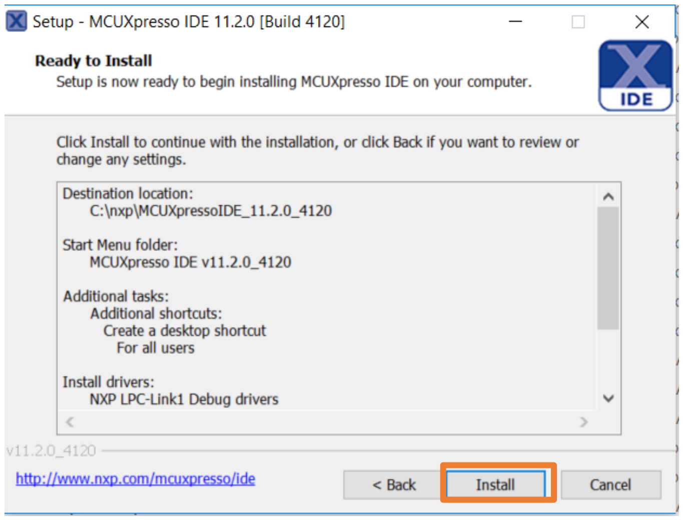
Setup will install MCUXpresso IDE into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

C:\nxp\MCUXpressoIDE_11.2.0_4120

At least 1,547.7 MB of free disk space is required.

6. Proceed by clicking the [**Next**] button on all dialogs until the **Ready to Install** dialog is displayed. Then, click [**Install**] to start the installation. Accept the installation of all drivers and parts.



Setup - MCUXpresso IDE 11.2.0 [Build 4120]

Ready to Install

Setup is now ready to begin installing MCUXpresso IDE on your computer.

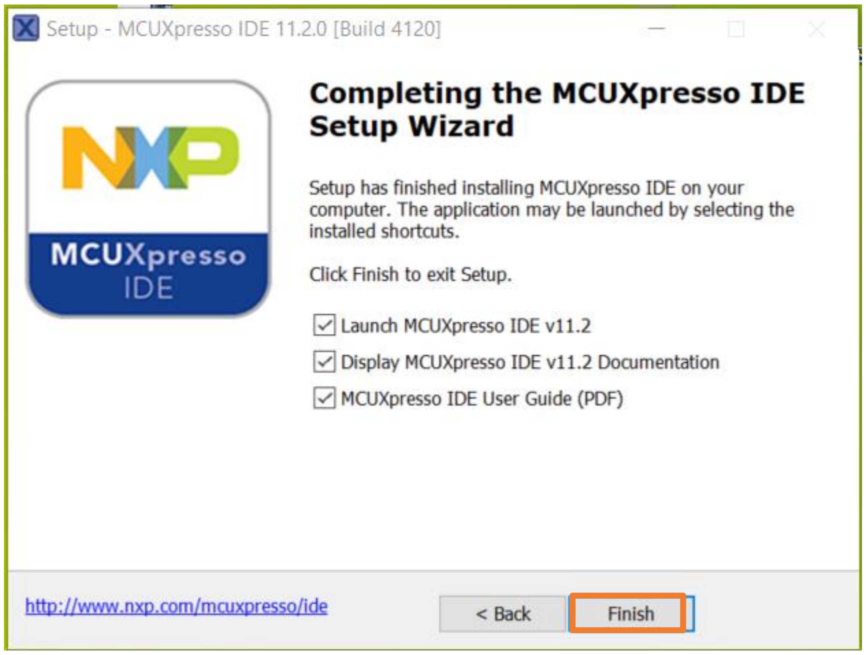
Click Install to continue with the installation, or click Back if you want to review or change any settings.

```
Destination location:
    C:\nxp\MCUXpressoIDE_11.2.0_4120

Start Menu folder:
    MCUXpresso IDE v11.2.0_4120

Additional tasks:
    Additional shortcuts:
      Create a desktop shortcut
        For all users

Install drivers:
    NXP LPC-Link1 Debug drivers
```

v11.2.0_4120

http://www.nxp.com/mcuxpresso/ide

7. Continue to click [**Next**] until the installation completes and the final dialog appears. Click [**Finish**] to conclude the installation.



Setup - MCUXpresso IDE 11.2.0 [Build 4120]

Completing the MCUXpresso IDE Setup Wizard

Setup has finished installing MCUXpresso IDE on your computer. The application may be launched by selecting the installed shortcuts.

Click Finish to exit Setup.

☑ Launch MCUXpresso IDE v11.2

☑ Display MCUXpresso IDE v11.2 Documentation

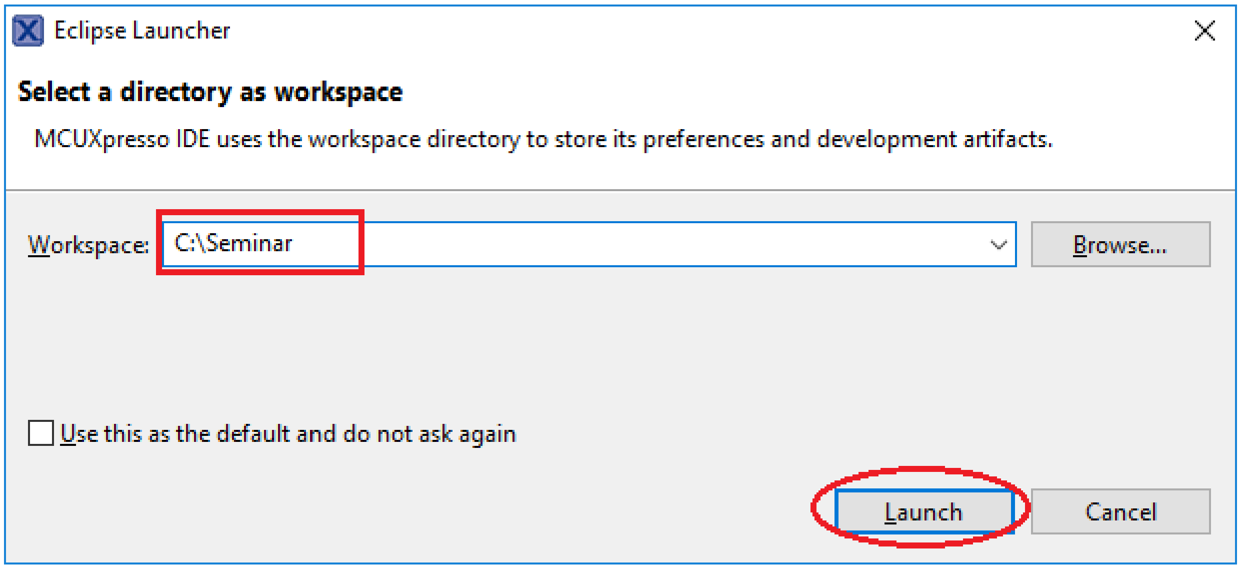
☑ MCUXpresso IDE User Guide (PDF)

http://www.nxp.com/mcuxpresso/ide

1.3 Installing MCUXpresso SDK for i.MXRT1060

1. Start the IDE from the desktop shortcut icon shortcut icon, toolbar icons, or by running the program from the install location, for example: ***C:\NXP\MCUXpressoIDE_11.2.0_4120\ide\mcuxpressoide.exe***.

2. The IDE will first prompt you for a workspace to store preferences, source code and development artifacts. It does not really matter where this workspace is located, nor the name that you provide. For simplicity, give the workspace the name ***Seminar***, leave it in the default location and then click [**Launch**]. Ignore these instructions if you know what you are doing and would prefer a more convenient location.



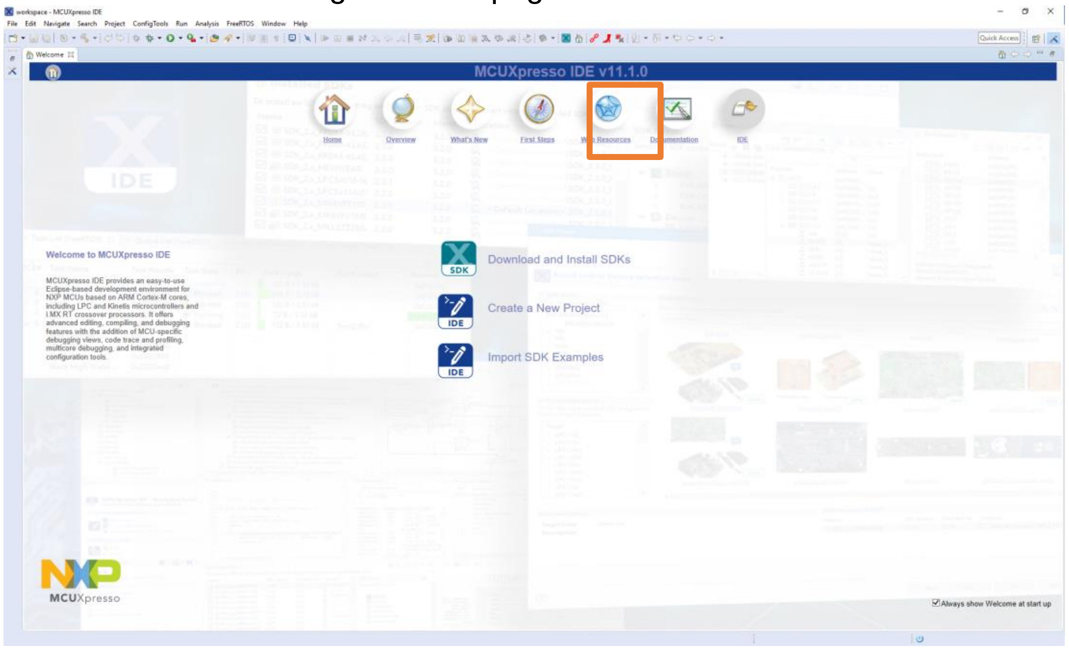
Eclipse Launcher

Select a directory as workspace

MCUXpresso IDE uses the workspace directory to store its preferences and development artifacts.

Workspace: C:\Seminar

3. The IDE will launch the following welcome page:



workspace - MCUXpresso IDE

MCUXpresso IDE v11.1.0

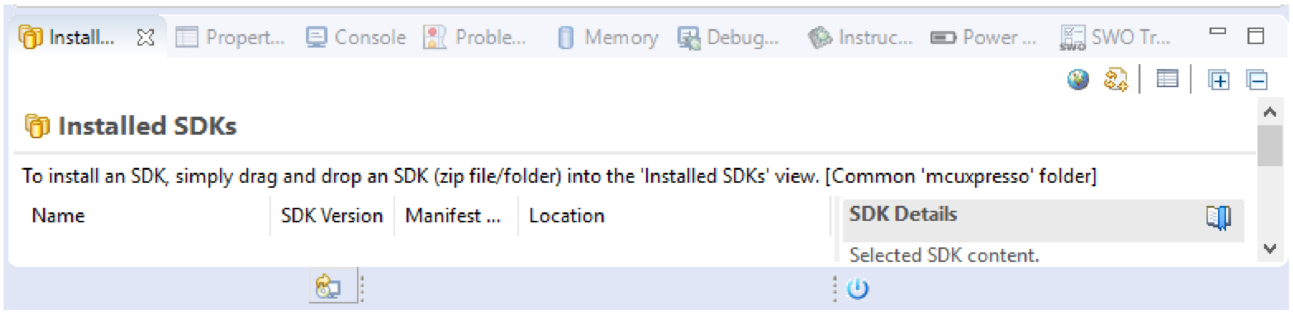
Welcome to MCUXpresso IDE

MCUXpresso IDE provides an easy-to-use Eclipse-based development environment for NXP MCUs based on ARM Cortex-M cores, including LPC and Kinetis microcontrollers and i.MX RT crossover processors. It offers advanced editing, compiling, and debugging features with the addition of MCU-specific debugging views, code trace and profiling, multicore debugging, and integrated configuration tools.

Click on the IDE icon to switch to main IDE perspective.

4. MCUXpresso opens with the **Develop** perspective, and observe the **Quickstart Panel** view (bottom left) and the **Installed SDKs** view at the lower edge of the perspective.

5. Next, we import the SDK into MCUXpresso IDE with a 'drag and drop' operation. Locate the **SDK_2.8.0_MIMXRT1062xxxxA.zip** for IMXRT1060-EVK that you as an archive using the SDK Builder as described in section 1.1.1 Then drag-and-drop it into the **Installed SDKs** view in MCUXpresso IDE.



To install an SDK, simply drag and drop an SDK (zip file/folder) into the 'Installed SDKs' view. [Common 'mcuxpresso' folder]

Name	SDK Version	Manifest ...	Location

SDK Details

Selected SDK content.

6. Depending upon the host permission on your account, this process may, or may not work. If the drag and drop fails, you can manually copy the ZIP file into the following folder. Note that it is in your User account on your host PC. ***C:\Users\<Your User Name>\mcuxpresso\SDKPackages***

7. You should end up with the SDK installed into MCUXpresso IDE (you may need to restart MCUXpresso IDE). The SDK will be visible in the Installed SDKs view and we can now start to use the example software.

End of Lab

Lab 1: Using the Getting Started Guide to set up the board.

Quickstart Video on www.nxp.com

Getting started video available HERE.

2.1. Getting familiar with the hardware

The 88W8977 System-on-Chip (SoC) is a highly integrated single-chip solution that incorporates both WLAN (2.4/5 GHz) and Bluetooth® technology, and the SoC is specifically designed to support the speed, reliability, and quality requirements of next-generation products. An IEEE 802.11n compliant dual-band system-on-chip offering Wi-Fi® data rates up to MCS 7 (150 Mbit/s), the 88W8977 is designed to offer the smallest footprint and lowest bill of materials (BOM) for devices such as Wearables, Internet of Things (IoT) and

Smart Home markets. It also provides for 3-way coexistence for WLAN, Bluetooth, and ZigBee operation with ZigBee Solutions.

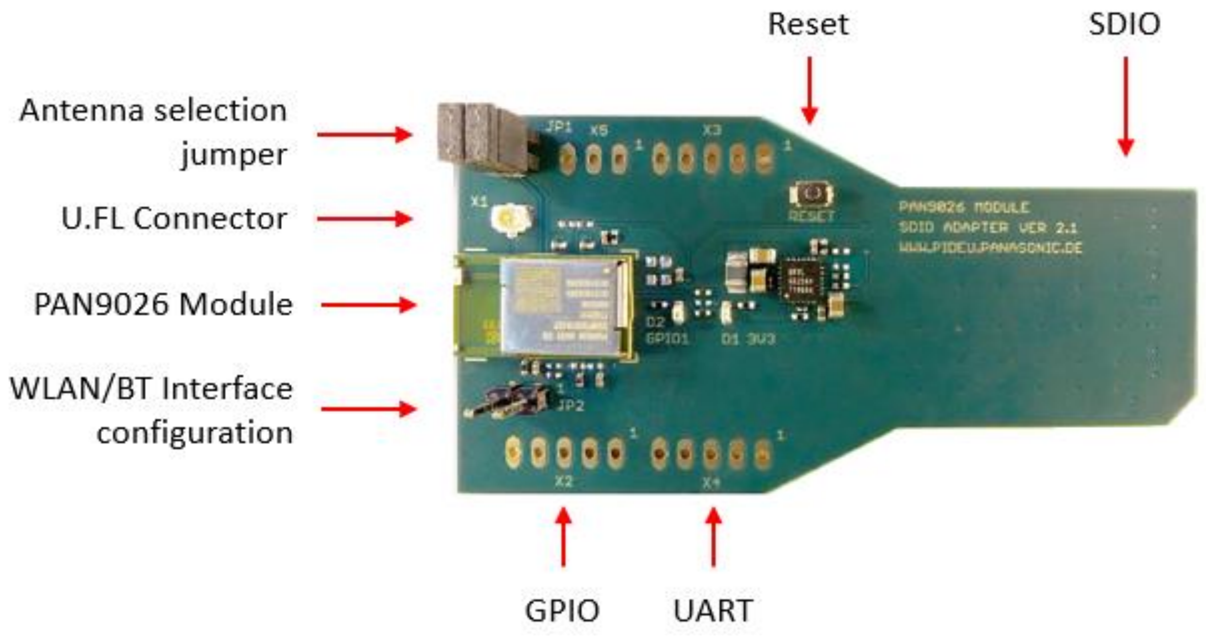
Panasonic PAN9026 platform

The **PAN9026** is a 2.4 GHz and 5 GHz ISM band Wi-Fi and Bluetooth radio module, specifically designed for highly integrated and cost-effective applications. This module is based on the chipset 88W8977 supporting 802.11a/b/g/n simultaneous station, access point. Integrated power management, a fast dual-core CPU, 802.11i security standard support, and high-speed data interfaces deliver the performance for the speed, reliability, and quality requirements of products.

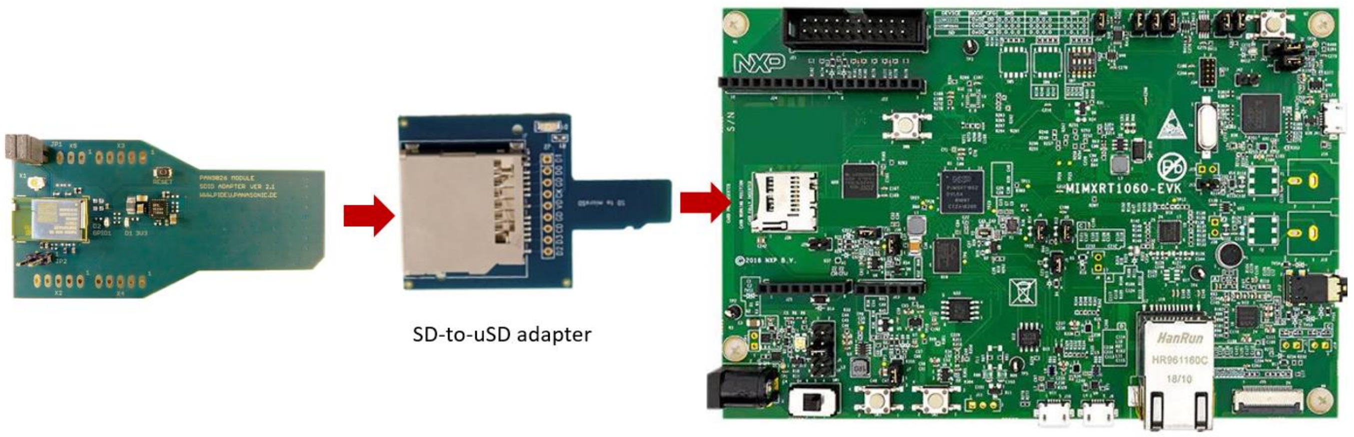
Wi-Fi chip	Panasonic Module
NXP 88W8977	PAN9026

Visit module manufacturer for more information about this Wi-Fi module.

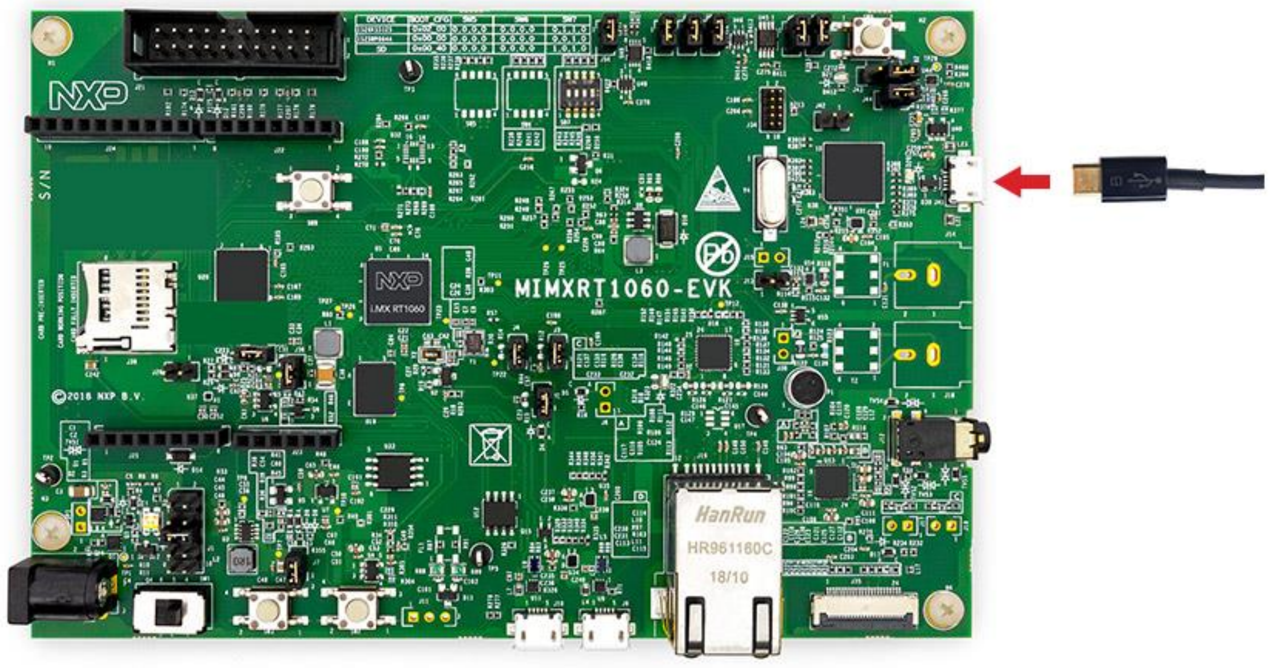
The layout of the module is as shown below



2.2 Connect the WiFi module to the board via an SD to uSD adaptor.



2.3 Plug in the board and connect the other end of the USB to your PC.



End of Lab

Lab 2: Build & Run your first WiFi Example

Required equipment

1 x MIMXRT1060-EVK
1 x PAN9026 WiFi Module
1 x SD to uSD Card adaptor

3.1 Introduction

The MCUXpresso SDK comes with a long list of example application code. To see what's available, browse to the SDK boards folder of your SDK installation and select your board, the MIMXRT1060-EVK (*/boards/evkmimxrt1060*).

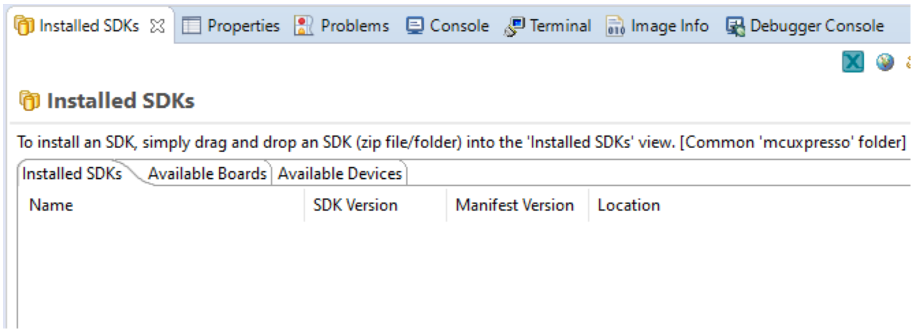
To learn more about specific example code, open the readme.txt file in an example's directory.

3.2 Build, Run and Debug MCUXpresso SDK Examples

The MCUXpresso SDK provides a collection of WI-FI example applications.
Follow these steps to import, configure, build, debug and run a WI-FI demo example through MCUXpresso IDE. This guide will use i.MX RT 1060 as reference, but similar steps apply to other EVKs.

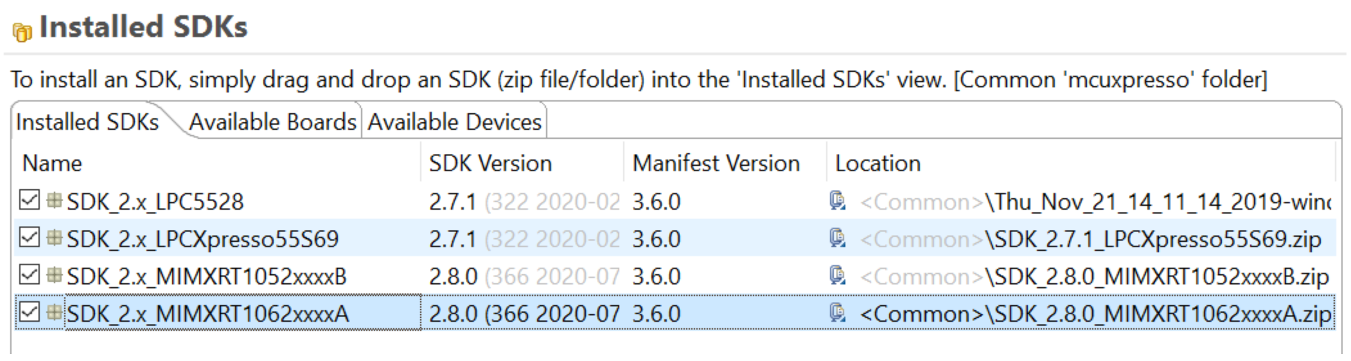
3.3 Import SDK to your MCUXpresso workspace (if not yet done in software installs)

1. Open the MCUXpresso IDE.
2. Switch to the 'Installed SDKs' view within the MCUXpresso IDE window.



To install an SDK, simply drag and drop an SDK (zip file/folder) into the 'Installed SDKs' view. [Common 'mcuxpresso' folder]

3. If you haven't imported the SDK into MCUXpresso IDE, drag and drop the SDK for your development board (in zip format) into the "Installed SDKs" section.You will get the following pop-up. Click on **OK** to continue the import.
4. The SDK will now appear in the Installed SDKs view as shown below:



Installed SDKs

To install an SDK, simply drag and drop an SDK (zip file/folder) into the 'Installed SDKs' view. [Common 'mcuxpresso' folder]

Installed SDKs | Available Boards | Available Devices

Name	SDK Version	Manifest Version	Location
☑ SDK_2.x_LPC5528	2.7.1 (322 2020-02	3.6.0	<Common>\Thu_Nov_21_14_11_14_2019-winc
☑ SDK_2.x_LPCXpresso55S69	2.7.1 (322 2020-02	3.6.0	<Common>\SDK_2.7.1_LPCXpresso55S69.zip
☑ SDK_2.x_MIMXRT1052xxxxB	2.8.0 (366 2020-07	3.6.0	<Common>\SDK_2.8.0_MIMXRT1052xxxxB.zip
☑ SDK_2.x_MIMXRT1062xxxxA	2.8.0 (366 2020-07	3.6.0	<Common>\SDK_2.8.0_MIMXRT1062xxxxA.zip

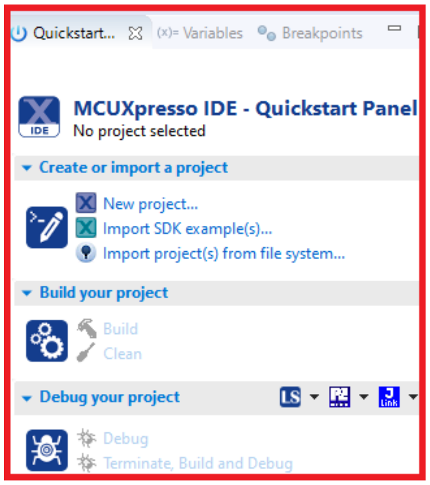
3.4 Open & Build an Example Application

The following steps will guide you to build and run the **`wifi_iperf`** application using MCUXpresso IDE.

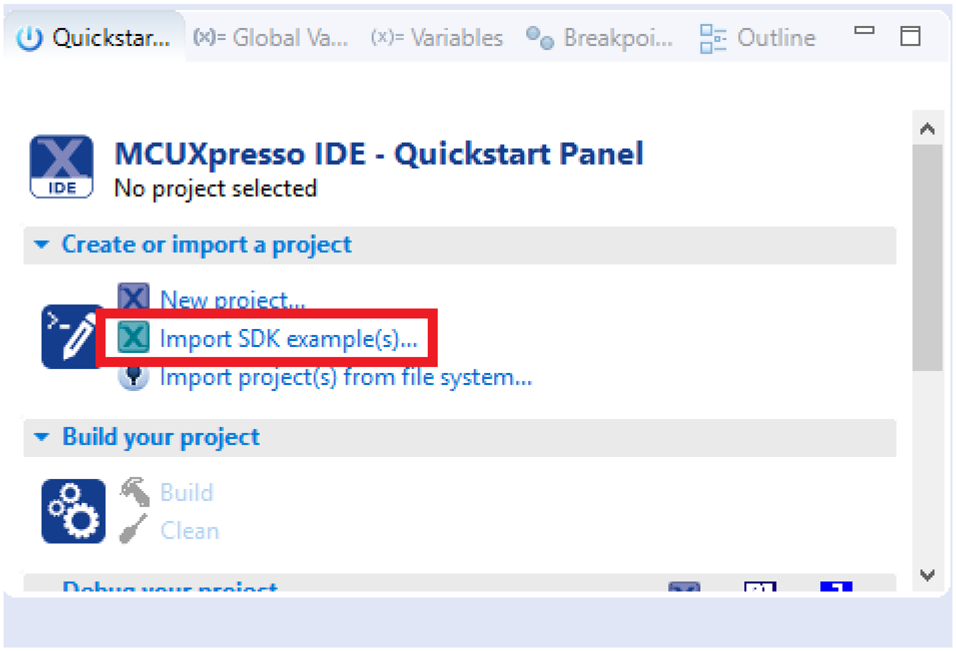
The **`wifi_iperf`** application demonstrates how to implement different features

- Perform a Network scan
- Connect to an Access Point
- Start your own Access Point
- Enable Deep Sleep operation
- TCP and UDP throughput measurements acting as a server or as a client
- Print network information

1. Locate the Quickstart Panel in the lower left-hand corner of the IDE.



2. Click on ‘Import SDK example(s)…



MCUXpresso IDE - Quickstart Panel

No project selected

Create or import a project

New project...

Import SDK example(s)...

Import project(s) from file system...

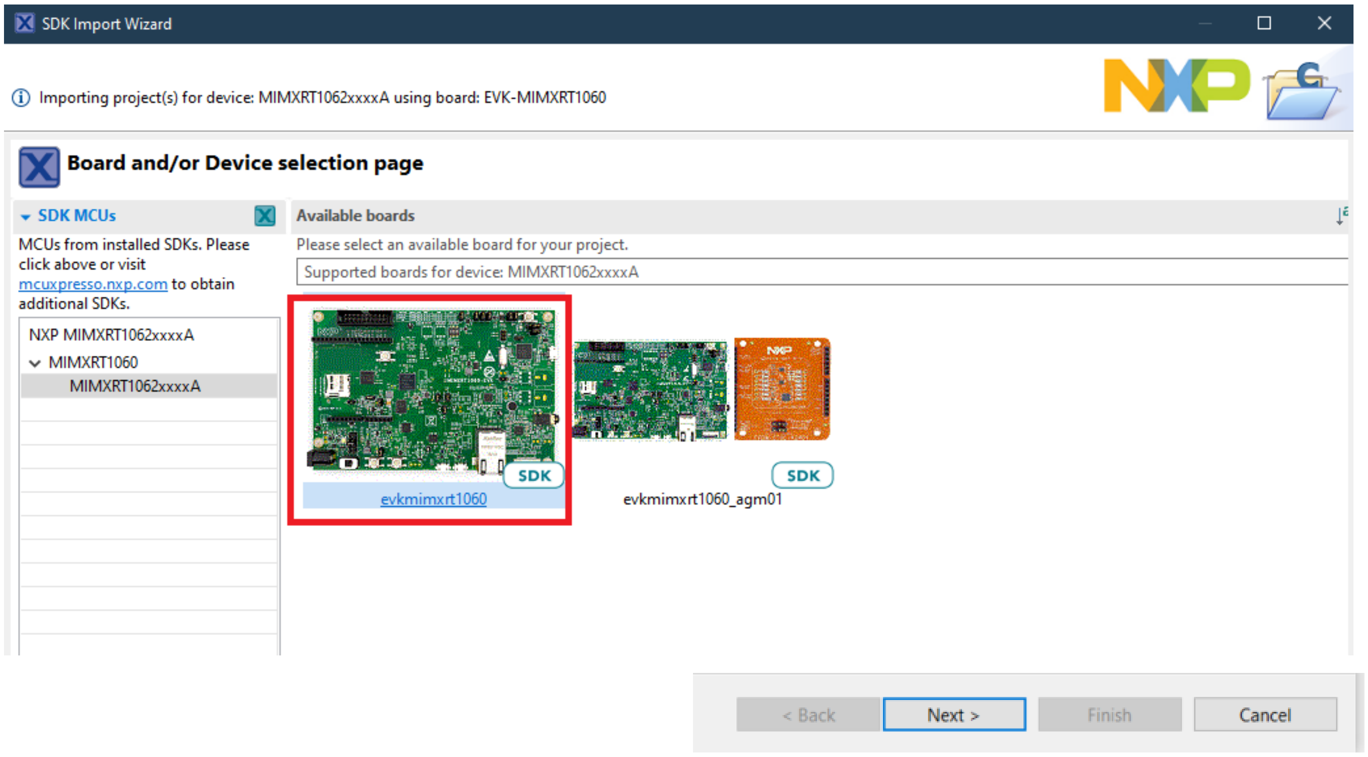
Build your project

Build

Clean

Debug your project

3. Select your evaluation board (i.MXRT1060), and click on Next.



Importing project(s) for device: MIMXRT1062xxxxA using board: EVK-MIMXRT1060

Board and/or Device selection page

SDK MCUs

MCUs from installed SDKs. Please click above or visit mcuxpresso.nxp.com to obtain additional SDKs.

NXP MIMXRT1062xxxxA

MIMXRT1060

MIMXRT1062xxxxA

Available boards

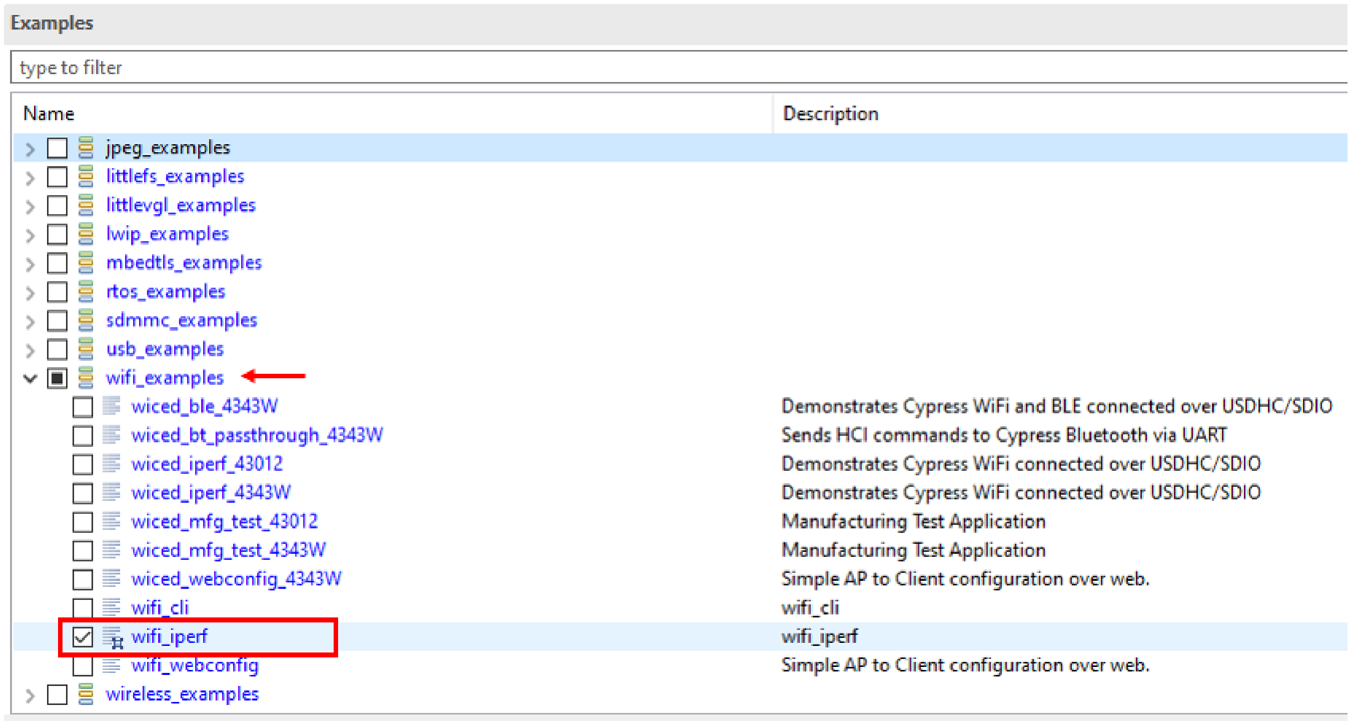
Please select an available board for your project.

Supported boards for device: MIMXRT1062xxxxA

evkmimxrt1060

evkmimxrt1060_agm01

4. Use the arrow button to expand the wifi_examples category and click the checkbox next to wifi_iperf to select that project. Then, click on Finish.



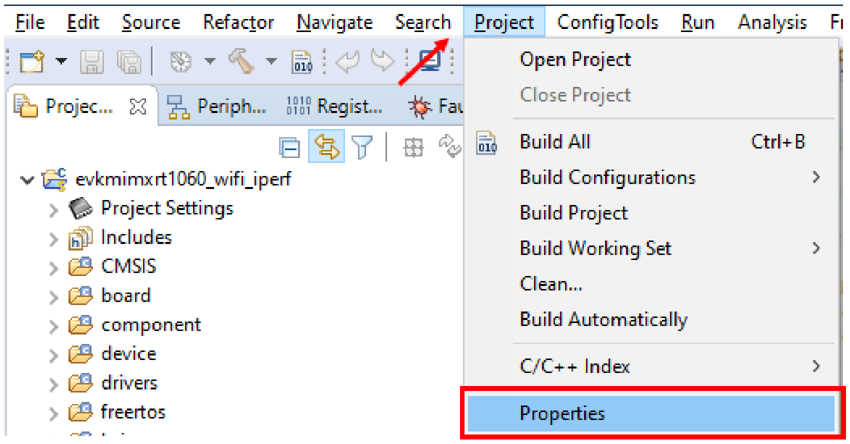
Examples

type to filter

Name	Description
jpeg_examples	
littlefs_examples	
littlevgl_examples	
lwip_examples	
mbedtls_examples	
rtos_examples	
sdmmc_examples	
usb_examples	
wifi_examples	
wiced_ble_4343W	Demonstrates Cypress WiFi and BLE connected over USDHC/SDIO
wiced_bt_passthrough_4343W	Sends HCI commands to Cypress Bluetooth via UART
wiced_iperf_43012	Demonstrates Cypress WiFi connected over USDHC/SDIO
wiced_iperf_4343W	Demonstrates Cypress WiFi connected over USDHC/SDIO
wiced_mfg_test_43012	Manufacturing Test Application
wiced_mfg_test_4343W	Manufacturing Test Application
wiced_webconfig_4343W	Simple AP to Client configuration over web.
wifi_cli	wifi_cli
✓ wifi_iperf	wifi_iperf
wifi_webconfig	Simple AP to Client configuration over web.
wireless_examples	

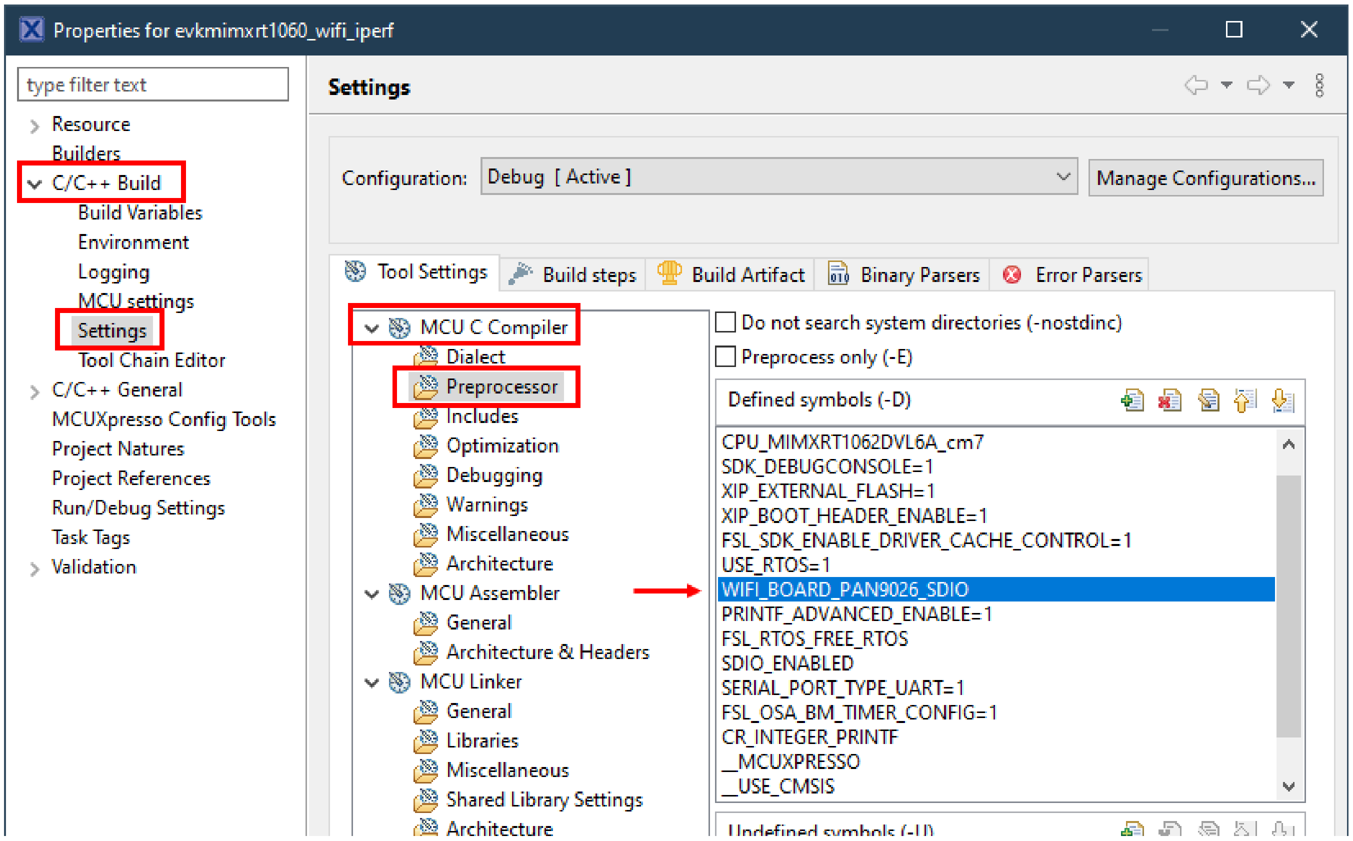
5. By default, the project is configured to use the PAN9026 Wi-Fi module (SD8977). If you are using another module, e.g. AzureWave AW-NM191MA (SD8801), follow these steps to modify the project configuration:

   a. Open project's Properties.



b. Go to C/C++ Build → Settings → MCU C Compiler → Preprocessor

c. Replace the **`WIFI_BOARD_PAN9026_SDIO`** symbol with **`WIFI_BOARD_AW_NM191MA`**



Properties for evkmimxrt1060_wifi_iperf

Settings

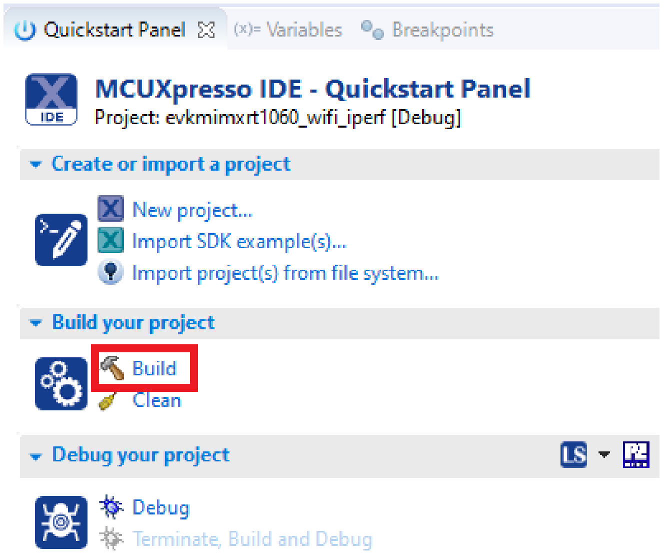
Configuration: Debug [Active]

Defined symbols (-D):

```
CPU_MIMXRT1062DVL6A_cm7
SDK_DEBUGCONSOLE=1
XIP_EXTERNAL_FLASH=1
XIP_BOOT_HEADER_ENABLE=1
FSL_SDK_ENABLE_DRIVER_CACHE_CONTROL=1
USE_RTOS=1
WIFI_BOARD_PAN9026_SDIO
PRINTF_ADVANCED_ENABLE=1
FSL_RTOS_FREE_RTOS
SDIO_ENABLED
SERIAL_PORT_TYPE_UART=1
FSL_OSA_BM_TIMER_CONFIG=1
CR_INTEGER_PRINTF
__MCUXPRESSO
__USE_CMSIS
```

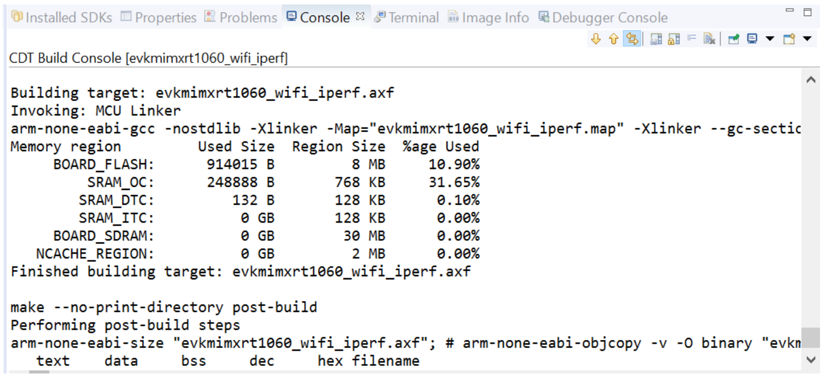
d. Apply and Close.

6. Select the project and build it.



Quickstart Panel
(x)= Variables
Breakpoints
MCUXpresso IDE - Quickstart Panel
Project: evkmimxrt1060_wifi_iperf [Debug]
Create or import a project
New project...
Import SDK example(s)...
Import project(s) from file system...
Build your project
Build
Clean
Debug your project
Debug
Terminate, Build and Debug

7. The project should build without errors problems in about 5 minutes. The building results are shown in the following picture



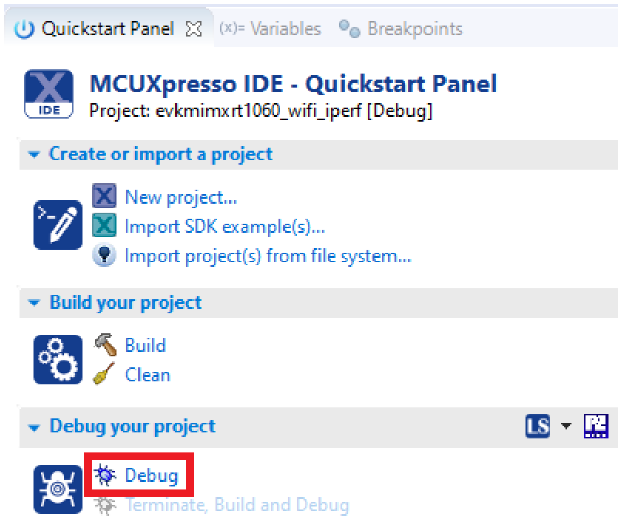
CDT Build Console [evkmimxrt1060_wifi_iperf]

```
Building target: evkmimxrt1060_wifi_iperf.axf
Invoking: MCU Linker
arm-none-eabi-gcc -nostdlib -Xlinker -Map="evkmimxrt1060_wifi_iperf.map" -Xlinker --gc-sectic
Memory region         Used Size  Region Size  %age Used
     BOARD_FLASH:      914015 B         8 MB     10.90%
         SRAM_OC:      248888 B       768 KB     31.65%
        SRAM_DTC:         132 B       128 KB      0.10%
        SRAM_ITC:          0 GB       128 KB      0.00%
     BOARD_SDRAM:          0 GB        30 MB      0.00%
   NCACHE_REGION:          0 GB         2 MB      0.00%
Finished building target: evkmimxrt1060_wifi_iperf.axf

make --no-print-directory post-build
Performing post-build steps
arm-none-eabi-size "evkmimxrt1060_wifi_iperf.axf"; # arm-none-eabi-objcopy -v -O binary "evkm
   text    data     bss     dec     hex filename
```

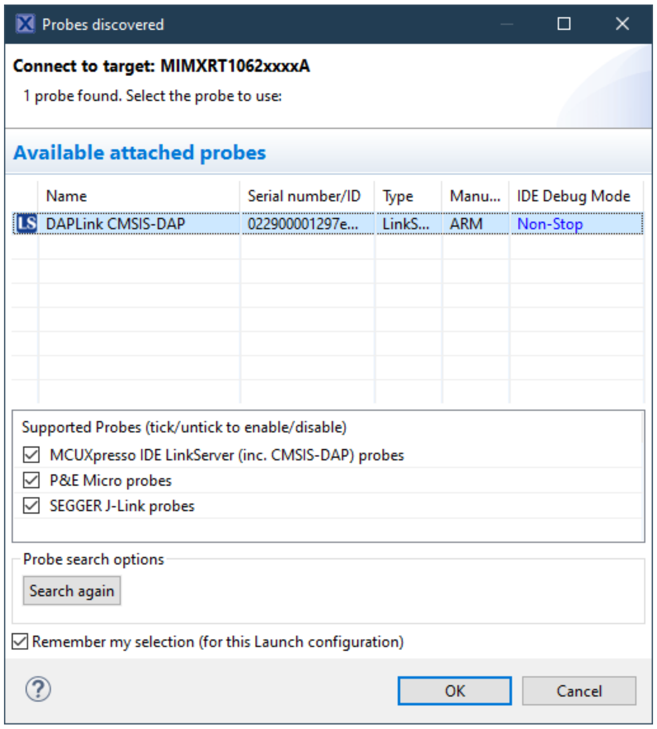
3.4 Debug the Example Application

1) Make sure the Wi-Fi module is attached to the EVK and connect your board to your computer.

2) Download the application to your i.MX RT-EVK.


Quickstart Panel
(x)= Variables
Breakpoints
MCUXpresso IDE - Quickstart Panel
Project: evkmimxrt1060_wifi_iperf [Debug]
Create or import a project
New project...
Import SDK example(s)...
Import project(s) from file system...
Build your project
Build
Clean
Debug your project
Debug
Terminate, Build and Debug

3) Select the debug probe of the board **DAPLink CMSIS-DAP** connected to your PC (note: you will see a debugger serial number different than the one in the picture)



Probes discovered

Connect to target: MIMXRT1062xxxxA

1 probe found. Select the probe to use:

Available attached probes

Name	Serial number/ID	Type	Manu...	IDE Debug Mode
DAPLink CMSIS-DAP	022900001297e...	LinkS...	ARM	Non-Stop

Supported Probes (tick/untick to enable/disable)

- ☑ MCUXpresso IDE LinkServer (inc. CMSIS-DAP) probes
- ☑ P&E Micro probes
- ☑ SEGGER J-Link probes

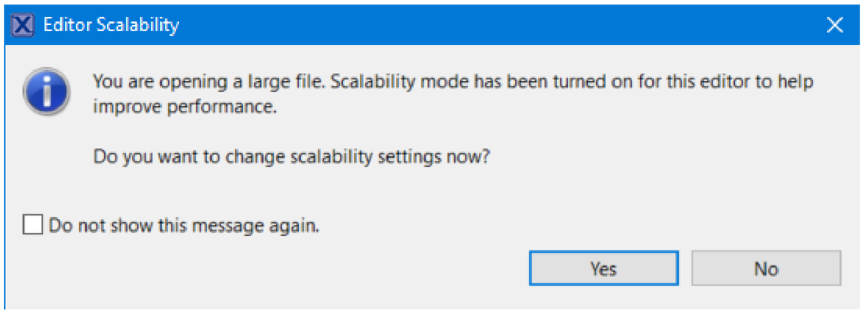
Probe search options

Search again

☑ Remember my selection (for this Launch configuration)

OK Cancel

NOTE: at the end of the download you will see the following message. Scalability mode is coming from the Eclipse IDE on top of which MCUXpresso is built, the warning message is embedded into the settings of MCUXpresso. Enabling scalability mode affects the performance of the IDE itself (it will be faster for large file access) and the IDE will turn it on when it needs to. The recommendation is **NOT** to change the default setting at this stage by clicking on **No** in the dialog)



Editor Scalability

You are opening a large file. Scalability mode has been turned on for this editor to help improve performance.

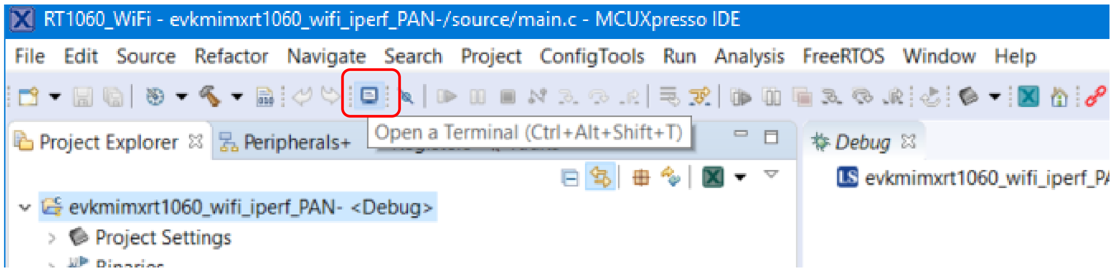
Do you want to change scalability settings now?

☐ Do not show this message again.

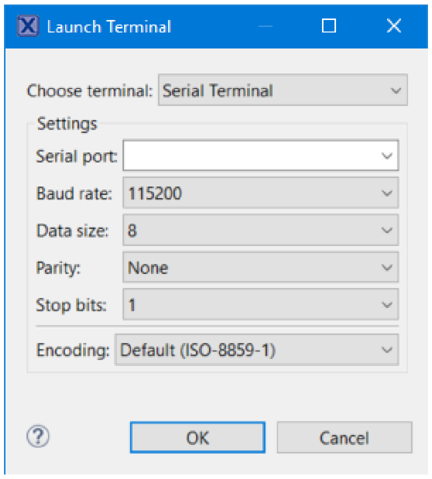
Yes No

4) Before we start the application we will start a serial terminal. This could be done using a 3rd party application such as **TeraTerm** or **PUTTY** however in this example we will use the Terminal Emulator built into the IDE.

5) Click on the Terminal icon on the IDE toolbar


RT1060_WiFi - evkmimxrt1060_wifi_iperf_PAN-/source/main.c - MCUXpresso IDE
File Edit Source Refactor Navigate Search Project ConfigTools Run Analysis FreeRTOS Window Help
Open a Terminal (Ctrl+Alt+Shift+T)
Project Explorer
Peripherals+
Debug
evkmimxrt1060_wifi_iperf_PAN- <Debug>
Project Settings
evkmimxrt1060_wifi_iperf_P

6) The Launch Terminal window appears



Choose terminal: Serial Terminal
Serial port:
Baud rate: 115200
Data size: 8
Parity: None
Stop bits: 1
Encoding: Default (ISO-8859-1)

Ensure the **Choose terminal** is set to **Serial Terminal**.

Select your Serial port: from the Drop Down

Baud Rate: 115200

Data size: 8

Parity: None

Stop bits: 1

Encoding: Default (ISO-8859-1)

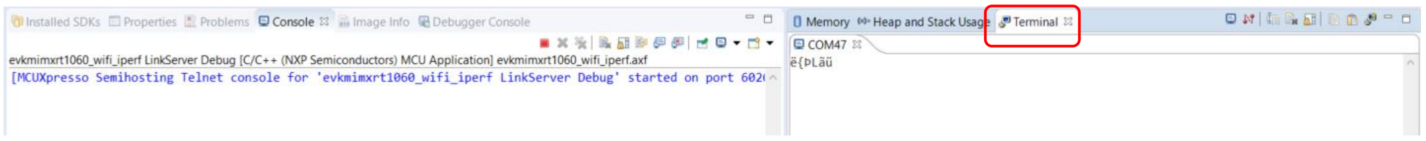
Click **OK**

7) Reposition the Terminal

The terminal window is placed by default in the bottom centre group.

Left mouse click on the terminal tab and drag and drop it in the bottom left group.

This will allow you to see the console window and the terminal window at the same time.

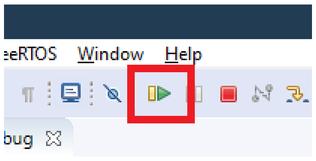


```
evkmimxrt1060_wifi_iperf LinkServer Debug [C/C++ (NXP Semiconductors) MCU Application] evkmimxrt1060_wifi_iperf.axf
[MCUXpresso Semihosting Telnet console for 'evkmimxrt1060_wifi_iperf LinkServer Debug' started on port 602
```

```
ë{ÞLäü
```

8) Running the Code

Click on the Resume button to start the application.



9) Check the output of the Terminal Window looks as follows:

```
========================================
wifi iperf demo
========================================
MCU FW Version: NXPSDK_v1.3.r14.p1
========================================
Initialize WLAN Driver
MAC Address: 00:13:43:75:91:7F
========================================
For Soft AP demonstration
Start a Soft AP using option "A" in WPA2 security mode from menu
This also starts DHCP Server with IP 192.168.10.1, NETMASK 255.255.255.0
========================================
For Station demonstration
Start an External AP with SSID as "nxp_wifi_demo" in Open mode
Start DHCP Server on External AP
Station network is configured with Dynamic address assignment
Application provides IPerf support
Set IPERF_SERVER_ADDRESS while using as IPerf Client
========================================
  A  Start Soft AP
  S  Stop Soft AP
  s  Start Scan for external APs
  c  Connect to External AP (SSID='nxp_wifi_demo')
  D  Disconnect from External AP
  I  Enable IEEE PS on Station
  i  Disable IEEE PS on Station
  d  Enable Deep sleep on Station
  e  Disable Deep sleep on Station
  p  Print All Network info
  1  TCP server mode (RX only test)
  2  TCP client mode (TX only test)
  3  TCP client dual mode (TX and RX in parallel)
  4  TCP client tradeoff mode (TX and RX sequentially)
  5  UDP server mode (RX only test)
  6  UDP client mode (TX only test)
  7  UDP client dual mode (TX and RX in parallel)
  8  UDP client tradeoff mode (TX and RX sequentially)
  h  Help (print this menu)
  H  Print extended help
[net] Initialized TCP/IP networking stack
TSF: 0.1205214
========================================
WLAN is Initialized
========================================
WLAN FW Version: w8977o-V2, RF87XX, FP91, 16.91.10.p81, WPA2_CVE_FIX 1, PVE_FIX 1
```

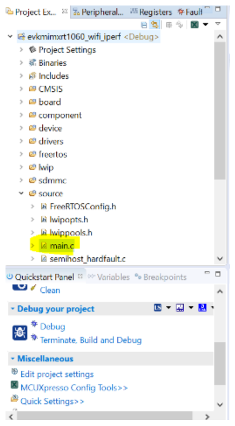
=======================================

10) You have now successfully built and debugged your first WiFi project. At this stage, press Terminate on the menu bar to end the Debug Session, in the next lab we will configure the code to connect to an AP and run several tests.

End of Lab

3.5 Configure the application to match your network settings.

1) Navigate the emkmimxrt1060_wifi_iperf project, locate the Source folder and double click on `main.c` file to edit it.

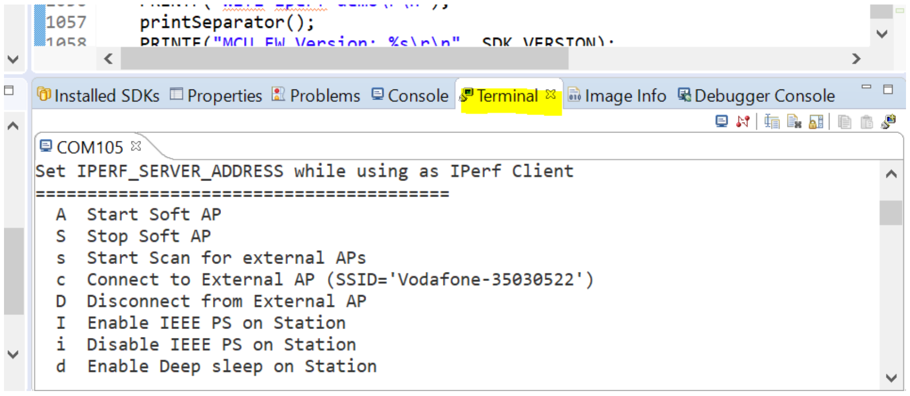


2) At lines 94-95 replace the two macros `EXT_AP_SSID` and `EXT_AP_PASSPHRASE` with your network name and password in this way

```
#define EXT_AP_SSID       "your network name here"
#define EXT_AP_PASSPHRASE "your network password here"
```

3) We are now ready to run some basic tests, click on Debug in the Quickstart Panel, your project will be rebuilt, the firmware download and subsequently the debugger will start and the application execution will hit the breakpoint at the first instruction in `main().`

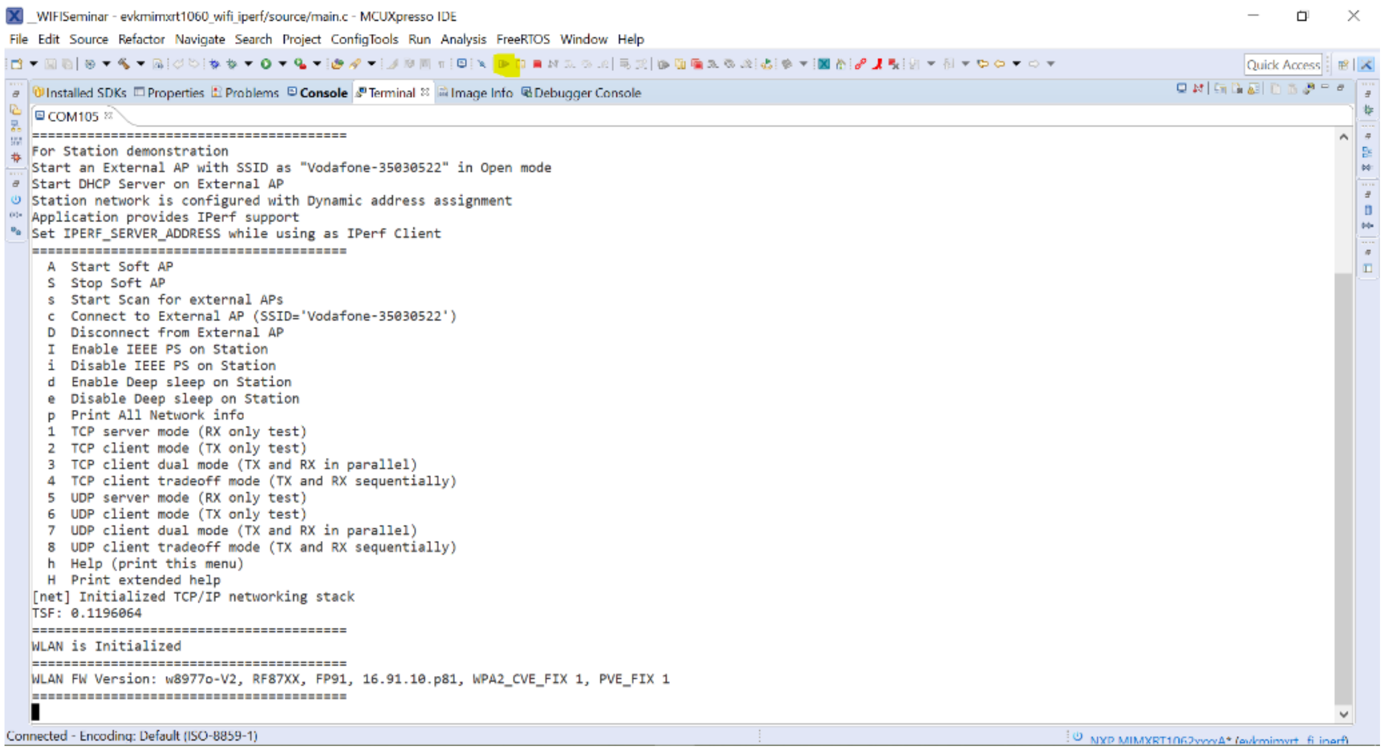
4) We now need to run a terminal application, you can run your favourite or simply enable the terminal built in into MCUXPresso as described at Section 3.4. steps 1) to 7). We recommend that you double click on Terminal tab to enlarge the view and have an almost full screen view of the Terminal output. Right click at the center of the Terminal window and select Clear Terminal to clear any previous output.



```
printSeparator();
```

```
Set IPERF_SERVER_ADDRESS while using as IPerf Client
========================================
  A   Start Soft AP
  S   Stop Soft AP
  s   Start Scan for external APs
  c   Connect to External AP (SSID='Vodafone-35030522')
  D   Disconnect from External AP
  I   Enable IEEE PS on Station
  i   Disable IEEE PS on Station
  d   Enable Deep sleep on Station
```

5) After Clicking on the Resume Button (green arrow) and waiting few seconds for the application to initialize you will see the main menu printed onto the Terminal window.


_WIFISeminar - evkmimxrt1060_wifi_iperf/source/main.c - MCUXpresso IDE
File Edit Source Refactor Navigate Search Project ConfigTools Run Analysis FreeRTOS Window Help
Quick Access
Installed SDKs
Properties
Problems
Console
Terminal
Image Info
Debugger Console
COM105
Connected - Encoding: Default (ISO-8859-1)

```
========================================
For Station demonstration
Start an External AP with SSID as "Vodafone-35030522" in Open mode
Start DHCP Server on External AP
Station network is configured with Dynamic address assignment
Application provides IPerf support
Set IPERF_SERVER_ADDRESS while using as IPerf Client
========================================
  A   Start Soft AP
  S   Stop Soft AP
  s   Start Scan for external APs
  c   Connect to External AP (SSID='Vodafone-35030522')
  D   Disconnect from External AP
  I   Enable IEEE PS on Station
  i   Disable IEEE PS on Station
  d   Enable Deep sleep on Station
  e   Disable Deep sleep on Station
  p   Print All Network info
  1   TCP server mode (RX only test)
  2   TCP client mode (TX only test)
  3   TCP client dual mode (TX and RX in parallel)
  4   TCP client tradeoff mode (TX and RX sequentially)
  5   UDP server mode (RX only test)
  6   UDP client mode (TX only test)
  7   UDP client dual mode (TX and RX in parallel)
  8   UDP client tradeoff mode (TX and RX sequentially)
  h   Help (print this menu)
  H   Print extended help
[net] Initialized TCP/IP networking stack
TSF: 0.1196064
========================================
WLAN is Initialized
========================================
WLAN FW Version: w8977o-V2, RF87XX, FP91, 16.91.10.p81, WPA2_CVE_FIX 1, PVE_FIX 1
========================================
```

Check whether you successfully modified the original project main.c file with your network settings, by having a look at the following line shown in the terminal window:

```
c  Connect to External AP (SSID='your network name here')
```

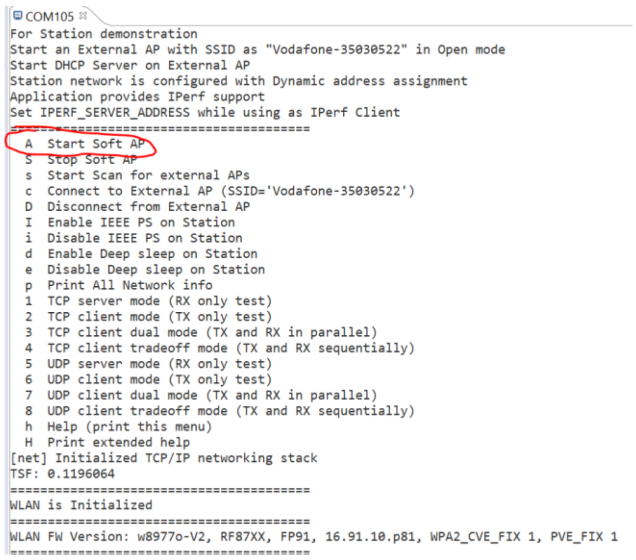
and make sure the `'your network name here' = the` SSID name you entered in main.c We are now almost ready to execute some operations interacting with the iperf application.

End of the lab

Lab 3: RT board as Soft Access Point

1) As a prerequisite you must have completed Lab 2 and started the WiFi_iPerf application in debug mode.

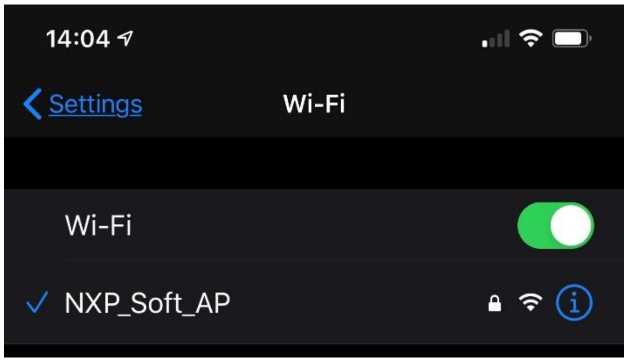
2) In the Terminal window press `A` to start the Soft Access Point



```
COM105
For Station demonstration
Start an External AP with SSID as "Vodafone-35030522" in Open mode
Start DHCP Server on External AP
Station network is configured with Dynamic address assignment
Application provides IPerf support
Set IPERF_SERVER_ADDRESS while using as IPerf Client
========================================
  A  Start Soft AP
  S  Stop Soft AP
  s  Start Scan for external APs
  c  Connect to External AP (SSID='Vodafone-35030522')
  D  Disconnect from External AP
  I  Enable IEEE PS on Station
  i  Disable IEEE PS on Station
  d  Enable Deep sleep on Station
  e  Disable Deep sleep on Station
  p  Print All Network info
  1  TCP server mode (RX only test)
  2  TCP client mode (TX only test)
  3  TCP client dual mode (TX and RX in parallel)
  4  TCP client tradeoff mode (TX and RX sequentially)
  5  UDP server mode (RX only test)
  6  UDP client mode (TX only test)
  7  UDP client dual mode (TX and RX in parallel)
  8  UDP client tradeoff mode (TX and RX sequentially)
  h  Help (print this menu)
  H  Print extended help
[net] Initialized TCP/IP networking stack
TSF: 0.1196064
========================================
WLAN is Initialized
========================================
WLAN FW Version: w8977o-V2, RF87XX, FP91, 16.91.10.p81, WPA2_CVE_FIX 1, PVE_FIX 1
========================================
```

This will start a DHCP server on the WiFi module, the SSID will be `NXP_Soft_AP` and the Password will be `12345678`.

3) Try now to connect to NXP_Soft_AP with your smartphone or your PC (below is an iPhone screen shot)



4) Once the connection has been established you can stop the Soft AP simply entering `S` into the Terminal window.

End of the lab

Lab 4: WiFi Landscape

1) As a prerequisite you must have completed Lab 2 and started the WiFi_iPerf application in debug mode.
2) Now press s to scan the access points around you; after few seconds you will get a list of the WiFi network visible around you together with some information such has MAC address, SSID, RSSI, Security type, Channel and if WMM is supported by that network.
3) Check if in the list your favourite network appears (SSID needs to match the SSID you entered at step Section 3.5 step 2)
4) Enter c into the terminal window and after few seconds a success message will be prompted on the terminal to show you are connected to your favourite network.

```
Key 'c': Connect to External AP (SSID='your network name here')

Connecting to your network name here .....Connected to following BSS:SSID = [your network name here], IP = [192.168.1.3]
```

5) Entering now p on the Terminal, you'll get a full set of information regarding your connection, including channel, IP Address, Security and many more similar to the following:

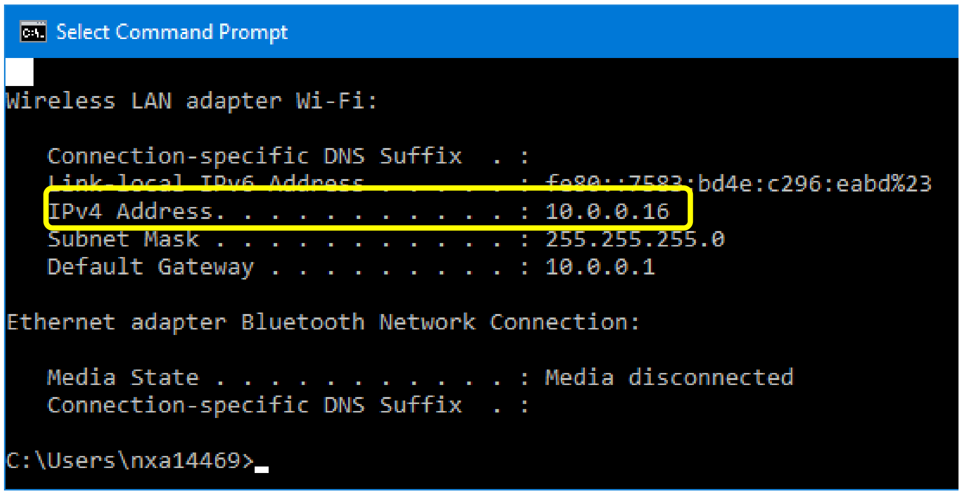
```
Station connected to:

"sta_network"

        SSID: your network name here
        BSSID: BC:15:AC:C4:46:5A
        channel: 3
        role: Infra
        security [Wildcard]: WPA2

        IPv4 Address

        address: DHCP

                IP:          192.168.1.3
                gateway:          192.168.1.1
                netmask:          255.255.255.0
                dns1:          192.168.1.1
                dns2:          0.0.0.0

uAP not started
```

It is now time to perform some tests using the iperf tool.

End of the lab

Lab 5: Setup `Iperf` Testing

1) Download the iperf tool 2.0.4 (please refer to the required SW section at the beginning of this guide) on your PC and unzip the files into a folder of your choice. First thing we need to do is to know the IP address of your machine running the iperf tool server.
2) To do this simply open a command prompt and type ipconfig followed by enter. Make a note the IP address of your machine.



```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::7583:bd4e:c296:eabd%23
   IPv4 Address. . . . . . . . . . . : 10.0.0.16
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.0.0.1

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

C:\Users\nxa14469>_
```

3) You need to modify the source code of the project to test the wifi module performance. To do this go back to your MCUXpresso project, in the main.c file and replace the macro located at line 98

```
#define IPERF_SERVER_ADDRESS "insert your PC IP address here"
```

Entering the IP address of your machine.

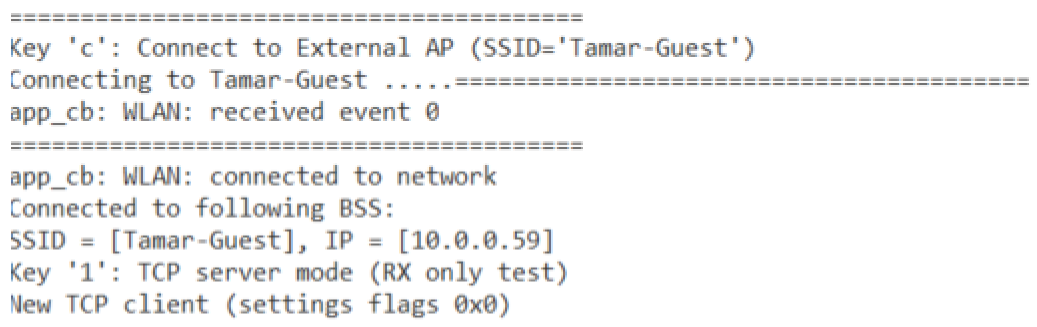
4) Build and debug the application using the dedicated buttons in the Quickstart Panel, after firmware download you'll enter Debug and the debugger will hit the breakpoint at first instruction in main().
5) Click on Resume (F8) to start program's execution, with the new configuration, at the Terminal press c to connect to your network and wait few seconds to get the message your board is connected to the selected network.

```
Key 'c': Connect to External AP (SSID='your network name here')
Connecting to your network name here
.....=======================================
app_cb: WLAN: received event 0
========================================
app_cb: WLAN: connected to      network
Connected to following BSS:
SSID = [your network name here], IP = [10.0.0.59]
```

End of the lab

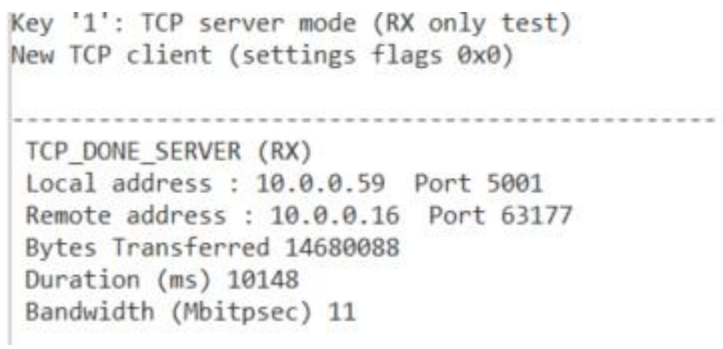
Lab 6: TCP testing with RT as server and PC as client

1) As a prerequisite you must have completed Lab 2 and started the WiFi_iPerf application in debug mode and then Lab 5 preparing the iPerf environment and pressing c to connect to `your network`. e.g.



```
========================================
Key 'c': Connect to External AP (SSID='Tamar-Guest')
Connecting to Tamar-Guest .....========================================
app_cb: WLAN: received event 0
========================================
app_cb: WLAN: connected to network
Connected to following BSS:
SSID = [Tamar-Guest], IP = [10.0.0.59]
Key '1': TCP server mode (RX only test)
New TCP client (settings flags 0x0)
```

2) Once you get the message you are connected to `your network` press 1 to start the iperf test in server mode using TCP protocol, by default it will use the port 5001. e.g.



```
Key '1': TCP server mode (RX only test)
New TCP client (settings flags 0x0)

------------------------------------------------
 TCP_DONE_SERVER (RX)
 Local address : 10.0.0.59  Port 5001
 Remote address : 10.0.0.16  Port 63177
 Bytes Transferred 14680088
 Duration (ms) 10148
 Bandwidth (Mbitpsec) 11
```

3) Go now to the iperf folder on your PC and open a command prompt window. You will need the IP address of the module which was prompted to the terminal at the end of Lab 5 when the connection was established as per below shown in red:

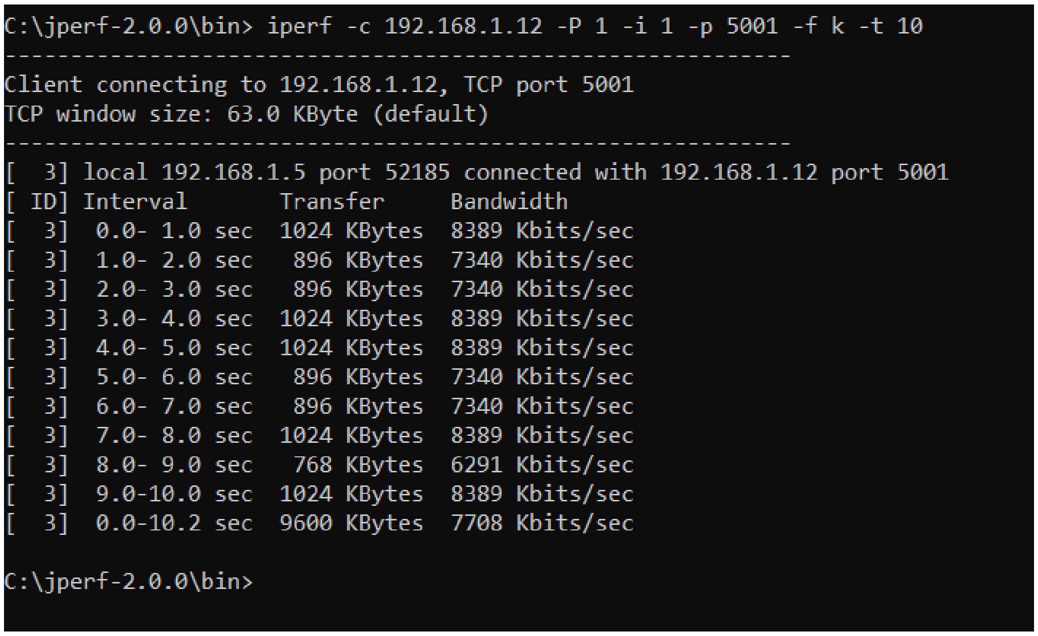
```
========================================
WLAN is Initialized
========================================
WLAN FW Version: w8977o-V2, RF87XX, FP91, 16.91.10.p81, WPA2_CVE_FIX 1, PVE_FIX 1
========================================
```

```
Key 'c': Connect to External AP (SSID=' your network name here’)
Connecting to your network name here.....Connected to following BSS:SSID = [your network name here], IP = [192.168.1.3]
```

4) Take a note of the IP address assigned to the module and at the PC command prompt enter the following command to start the iperf client:

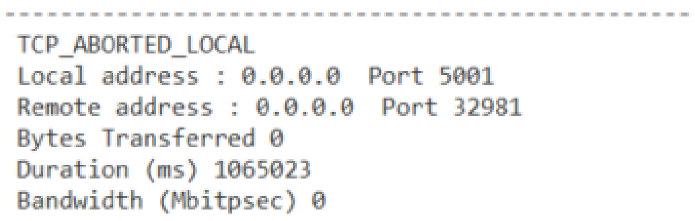
```
iperf -c <assigned IP address> -P 1 -i 1 -p 5001 -f k -t 10
```

5) Press Enter and wait for the test to be completed, when the test will be completed you will see something similar to this on command window in Windows.



```
C:\jperf-2.0.0\bin> iperf -c 192.168.1.12 -P 1 -i 1 -p 5001 -f k -t 10
------------------------------------------------------------
Client connecting to 192.168.1.12, TCP port 5001
TCP window size: 63.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.5 port 52185 connected with 192.168.1.12 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  1024 KBytes  8389 Kbits/sec
[  3]  1.0- 2.0 sec   896 KBytes  7340 Kbits/sec
[  3]  2.0- 3.0 sec   896 KBytes  7340 Kbits/sec
[  3]  3.0- 4.0 sec  1024 KBytes  8389 Kbits/sec
[  3]  4.0- 5.0 sec  1024 KBytes  8389 Kbits/sec
[  3]  5.0- 6.0 sec   896 KBytes  7340 Kbits/sec
[  3]  6.0- 7.0 sec   896 KBytes  7340 Kbits/sec
[  3]  7.0- 8.0 sec  1024 KBytes  8389 Kbits/sec
[  3]  8.0- 9.0 sec   768 KBytes  6291 Kbits/sec
[  3]  9.0-10.0 sec  1024 KBytes  8389 Kbits/sec
[  3]  0.0-10.2 sec  9600 KBytes  7708 Kbits/sec

C:\jperf-2.0.0\bin>
```

6) At the end of the test, you will see a report of the tests being printed onto the terminal in MCUXPresso. Press <space> to stop the server on the terminal. E.g.

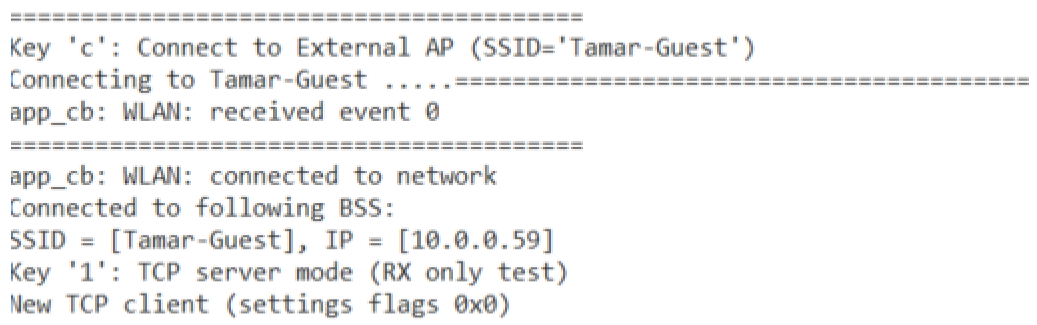


```
-----------------------------------------------
 TCP_ABORTED_LOCAL
 Local address : 0.0.0.0  Port 5001
 Remote address : 0.0.0.0  Port 32981
 Bytes Transferred 0
 Duration (ms) 1065023
 Bandwidth (Mbitpsec) 0
```

End of the lab

Lab 7: UDP testing with RT as server and PC as client

1) As a prerequisite you must have completed Lab 2 and started the WiFi_iPerf application in debug mode and then Lab 5 preparing the iPerf environment and pressing c to connect to `your network`. e.g.



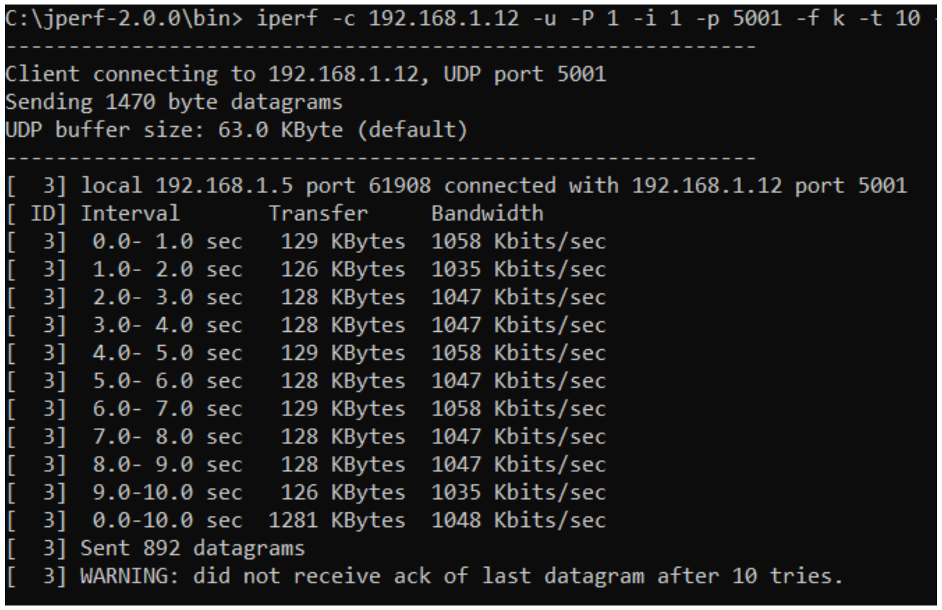
```
========================================
Key 'c': Connect to External AP (SSID='Tamar-Guest')
Connecting to Tamar-Guest .....========================================
app_cb: WLAN: received event 0
========================================
app_cb: WLAN: connected to network
Connected to following BSS:
SSID = [Tamar-Guest], IP = [10.0.0.59]
Key '1': TCP server mode (RX only test)
New TCP client (settings flags 0x0)
```

Once you get the message you are connected to `your network` then start the Iperf test in server mode using the UDP protocol by pressing 5 at the terminal prompt in MCUXPresso.

2) As per lab 6, go to the iperf folder on your PC and enter the following command

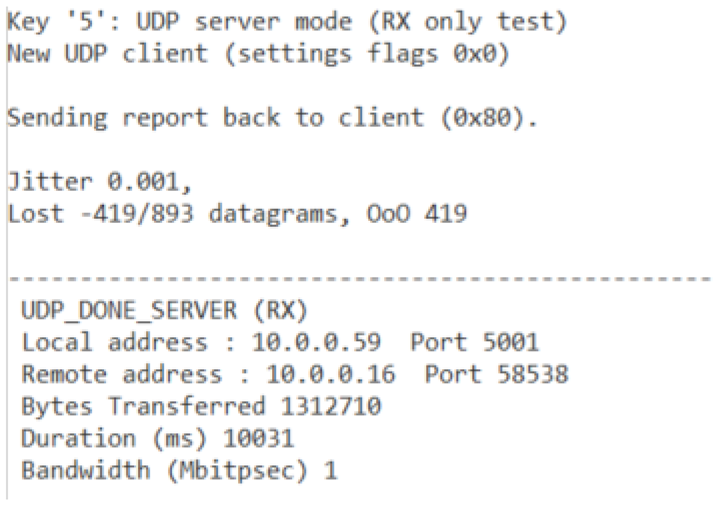
```
iperf -c <assigned IP address> -u -P 1 -i 1 -p 5001 -f k -t 10 -T 1
```

3) Press Enter and wait for the test to be completed, when the test will be completed you will see something similar to this on command window in Windows.



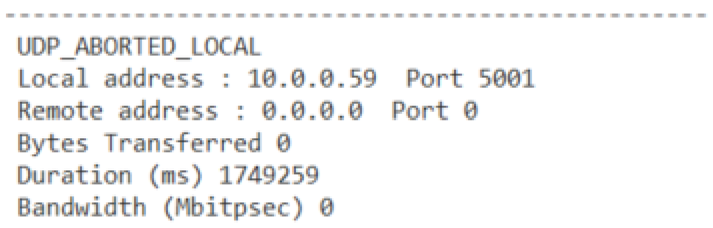
```
C:\jperf-2.0.0\bin> iperf -c 192.168.1.12 -u -P 1 -i 1 -p 5001 -f k -t 10
------------------------------------------------------------
Client connecting to 192.168.1.12, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 63.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.5 port 61908 connected with 192.168.1.12 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   129 KBytes  1058 Kbits/sec
[  3]  1.0- 2.0 sec   126 KBytes  1035 Kbits/sec
[  3]  2.0- 3.0 sec   128 KBytes  1047 Kbits/sec
[  3]  3.0- 4.0 sec   128 KBytes  1047 Kbits/sec
[  3]  4.0- 5.0 sec   129 KBytes  1058 Kbits/sec
[  3]  5.0- 6.0 sec   128 KBytes  1047 Kbits/sec
[  3]  6.0- 7.0 sec   129 KBytes  1058 Kbits/sec
[  3]  7.0- 8.0 sec   128 KBytes  1047 Kbits/sec
[  3]  8.0- 9.0 sec   128 KBytes  1047 Kbits/sec
[  3]  9.0-10.0 sec   126 KBytes  1035 Kbits/sec
[  3]  0.0-10.0 sec  1281 KBytes  1048 Kbits/sec
[  3] Sent 892 datagrams
[  3] WARNING: did not receive ack of last datagram after 10 tries.
```

4) At the end of the test, you will see a report of the tests being printed onto the terminal in MCUXPresso. Please note the difference in performance from UDP and TCP protocol tests.



```
Key '5': UDP server mode (RX only test)
New UDP client (settings flags 0x0)

Sending report back to client (0x80).

Jitter 0.001,
Lost -419/893 datagrams, OoO 419

-----------------------------------------------
 UDP_DONE_SERVER (RX)
 Local address : 10.0.0.59  Port 5001
 Remote address : 10.0.0.16  Port 58538
 Bytes Transferred 1312710
 Duration (ms) 10031
 Bandwidth (Mbitpsec) 1
```

5) Press <space> to stop the server on the terminal.



```
-----------------------------------------------
 UDP_ABORTED_LOCAL
 Local address : 10.0.0.59  Port 5001
 Remote address : 0.0.0.0  Port 0
 Bytes Transferred 0
 Duration (ms) 1749259
 Bandwidth (Mbitpsec) 0
```

End of the lab

Lab 8: TCP testing with PC as server and RT as client

1) As a prerequisite you must have completed Lab 2 and started the WiFi_iPerf application in debug mode and then Lab 5 preparing the iPerf environment.

2) In this lab we are executing the iperf TCP ***client*** on the board, therefore the first thing we must do is to start the iperf TCP ***server*** on the PC application.

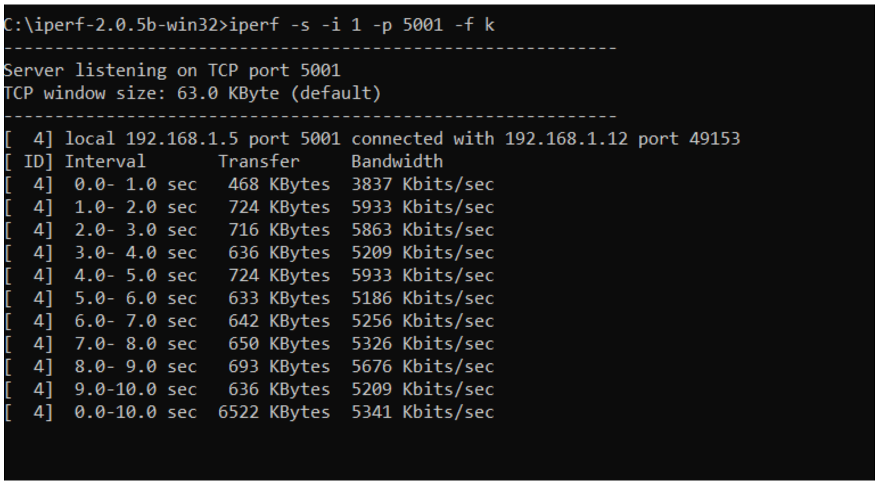
3) As per lab 6, go to the iperf folder on your PC and enter the following command

```
iperf -s -I 1 -p 5001 -f k
```

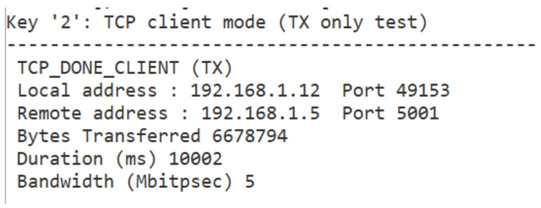
4) Now that the TCP server started, go to the MCUXpresso console to start the TCP client test entering the command 2 (Note: Please make sure the IP address of your PC is the

same as the one hardcoded into the project and defined at Lab 5 Step 3) and wait for the test to complete.

5) You should see a summary when the test is finished something similar to the following screenshot.



```
C:\iperf-2.0.5b-win32>iperf -s -i 1 -p 5001 -f k
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 63.0 KByte (default)
------------------------------------------------------------
[  4] local 192.168.1.5 port 5001 connected with 192.168.1.12 port 49153
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec   468 KBytes  3837 Kbits/sec
[  4]  1.0- 2.0 sec   724 KBytes  5933 Kbits/sec
[  4]  2.0- 3.0 sec   716 KBytes  5863 Kbits/sec
[  4]  3.0- 4.0 sec   636 KBytes  5209 Kbits/sec
[  4]  4.0- 5.0 sec   724 KBytes  5933 Kbits/sec
[  4]  5.0- 6.0 sec   633 KBytes  5186 Kbits/sec
[  4]  6.0- 7.0 sec   642 KBytes  5256 Kbits/sec
[  4]  7.0- 8.0 sec   650 KBytes  5326 Kbits/sec
[  4]  8.0- 9.0 sec   693 KBytes  5676 Kbits/sec
[  4]  9.0-10.0 sec   636 KBytes  5209 Kbits/sec
[  4]  0.0-10.0 sec  6522 KBytes  5341 Kbits/sec
```



```
Key '2': TCP client mode (TX only test)
--------------------------------------------------
 TCP_DONE_CLIENT (TX)
 Local address : 192.168.1.12  Port 49153
 Remote address : 192.168.1.5  Port 5001
 Bytes Transferred 6678794
 Duration (ms) 10002
 Bandwidth (Mbitpsec) 5
```

6) Press Ctrl-C to close the console or simply close the window to exit iperf. Press the space bar on the MCUXPresso terminal input to get back to the application main menu.

End of the lab

Lab 9: UDP testing with RT as server and PC as client

1) As a prerequisite you must have completed Lab 2 and started the WiFi_iPerf application in debug mode and then Lab 5 preparing the iPerf environment.

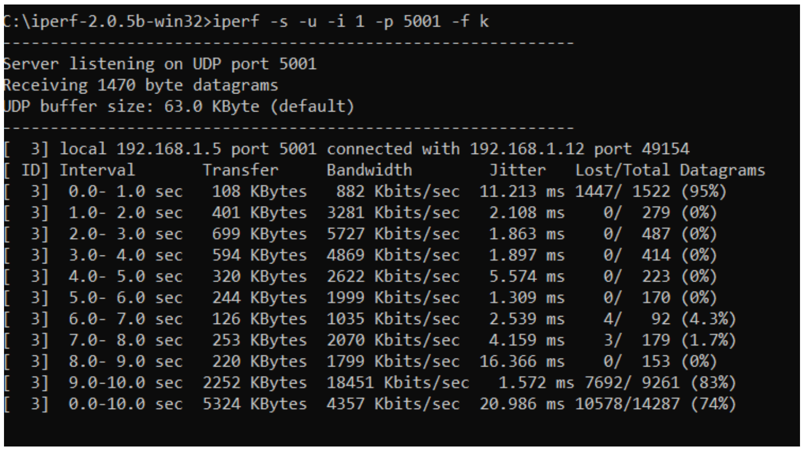
2) In this lab we are executing the iperf TCP ***client*** on the board, therefore the first thing we must do is to start the iperf TCP ***server*** on the PC application.

3) As per lab 6, go to the iperf folder on your PC and enter the following command

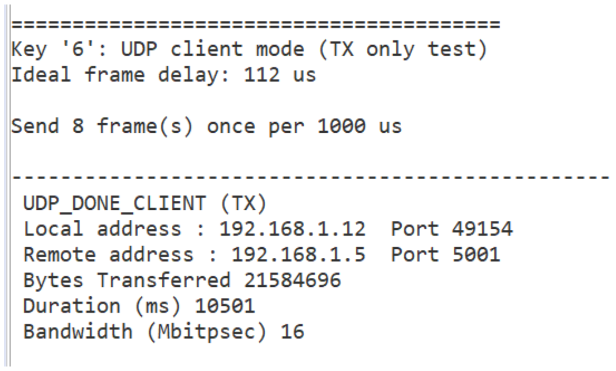
4) Start the UDP server on the PC using the following syntax

```
iperf -s -u -i 1 -p 5001 -f k
```

5) Now the UDP server is running on the PC, enter 6 on the MCUXPresso Terminal and wait for the test to be completed. At the end you will see a summary like this when the test is finished.



```
C:\iperf-2.0.5b-win32>iperf -s -u -i 1 -p 5001 -f k
------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 63.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.5 port 5001 connected with 192.168.1.12 port 49154
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec   108 KBytes   882 Kbits/sec  11.213 ms 1447/ 1522 (95%)
[  3]  1.0- 2.0 sec   401 KBytes  3281 Kbits/sec   2.108 ms    0/  279 (0%)
[  3]  2.0- 3.0 sec   699 KBytes  5727 Kbits/sec   1.863 ms    0/  487 (0%)
[  3]  3.0- 4.0 sec   594 KBytes  4869 Kbits/sec   1.897 ms    0/  414 (0%)
[  3]  4.0- 5.0 sec   320 KBytes  2622 Kbits/sec   5.574 ms    0/  223 (0%)
[  3]  5.0- 6.0 sec   244 KBytes  1999 Kbits/sec   1.309 ms    0/  170 (0%)
[  3]  6.0- 7.0 sec   126 KBytes  1035 Kbits/sec   2.539 ms    4/   92 (4.3%)
[  3]  7.0- 8.0 sec   253 KBytes  2070 Kbits/sec   4.159 ms    3/  179 (1.7%)
[  3]  8.0- 9.0 sec   220 KBytes  1799 Kbits/sec  16.366 ms    0/  153 (0%)
[  3]  9.0-10.0 sec  2252 KBytes  18451 Kbits/sec   1.572 ms 7692/ 9261 (83%)
[  3]  0.0-10.0 sec  5324 KBytes  4357 Kbits/sec  20.986 ms 10578/14287 (74%)
```



```
========================================
Key '6': UDP client mode (TX only test)
Ideal frame delay: 112 us

Send 8 frame(s) once per 1000 us

------------------------------------------------
 UDP_DONE_CLIENT (TX)
 Local address : 192.168.1.12  Port 49154
 Remote address : 192.168.1.5  Port 5001
 Bytes Transferred 21584696
 Duration (ms) 10501
 Bandwidth (Mbitpsec) 16
```

6) It is important to mention that results might vary depending on multiple factors like number of access points around, network traffic, topology, Security as well as the proximity to the access point.

End of Lab