AWR16xx nonOS OOB Demo

NOTE: ES2.0 devices only

This version of the mmWave Demo lab will work only with xWR1642BOOST ES2.0 EVMs, which require **mmWave SDK version 2.0 or above**.

Contents

- Lab Overview
- Requirements
- Software setup
  - Pre-requisites
  - Downloading the Lab Project
  - Building the project
- Hardware setup
  - Preparing the EVM
  - Connecting the EVM
- Running the Demo

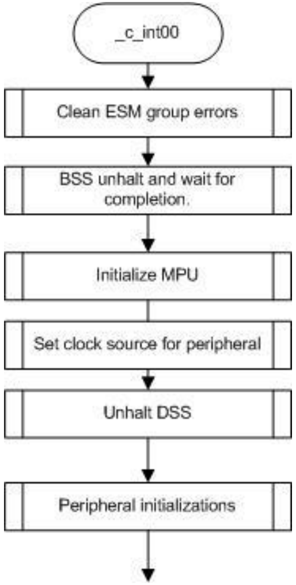
Lab Overview

- This lab exercise demonstrates the non-OS version of mmw demo running on AWR1642 mmWave Sensor.
- In this application TI-RTOS is removed from MSS & DSS driver and application code.
- The application showcase SDK driver OSAL implementation with Non OS routines and how mmw application can run in single threaded environment.
- Application’s features
  - mmWave Layer is removed from MSS and DSS application. It directly uses mmWaveLink to Program Radar Front end.
  - DSS runs mmWaveLink instance on it, which does RadarSS configuration. It receives all the CLI commands from MSS over mailbox.
  - No changes are required in drivers and libraries of mmWave SDK to work in non-OS environment. OSAL source & library for Non-OS are provided in the same package.
  - Interrupt(VIM) and Timer(RTI) driver were part of TI RTOS and hence to remove TI RTOS, implementation of these drivers are included in the application code.
- Application’s limitation compare to mmw demo of mmWave SDK v2.0
  - Dynamic sensor-stop over CLI is not possible as there is no multi-thread in this application to listen any further CLI command while MSS is busy pumping object data over UART. So to stop the frame or to change the configuration, user needs to restart the device and send updated CLI commands.
  - This demo provides a reference to implement datapath configuration and processing chain without any OS. It doesn’t include LVDS streaming, CQ & Analog monitoring features.

Lab Overview
Startup & OSAL

Startup Sequence

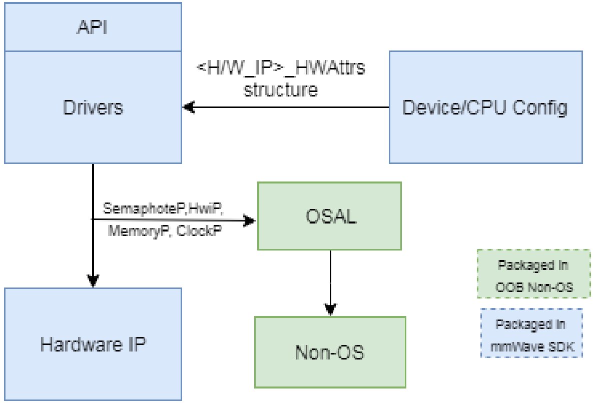
In case of default mmw demo, OS takes care of startup sequence but in Non-OS demo it needs to implement its own startup sequence.



_c_int00
Clean ESM group errors
BSS unhalt and wait for completion.
Initialize MPU
Set clock source for peripheral
Unhalt DSS
Peripheral initializations

OSAL

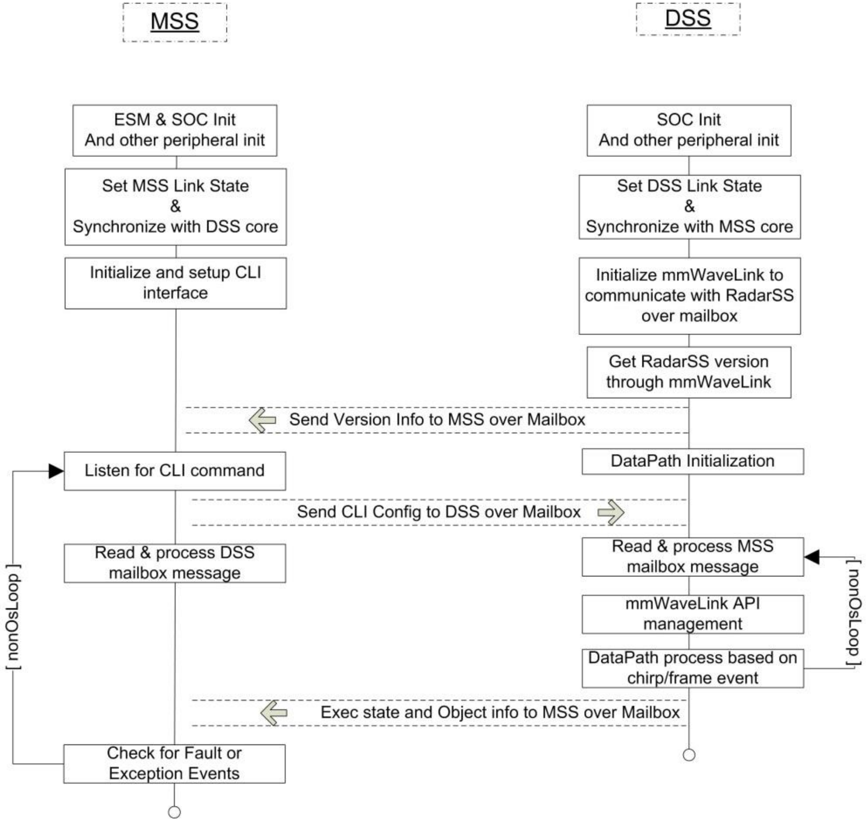
To avoid modification of drivers, this package provides OSAL implementation for Non-OS environment which has equivalent OS calls like Semaphore, Event, Memory etc.



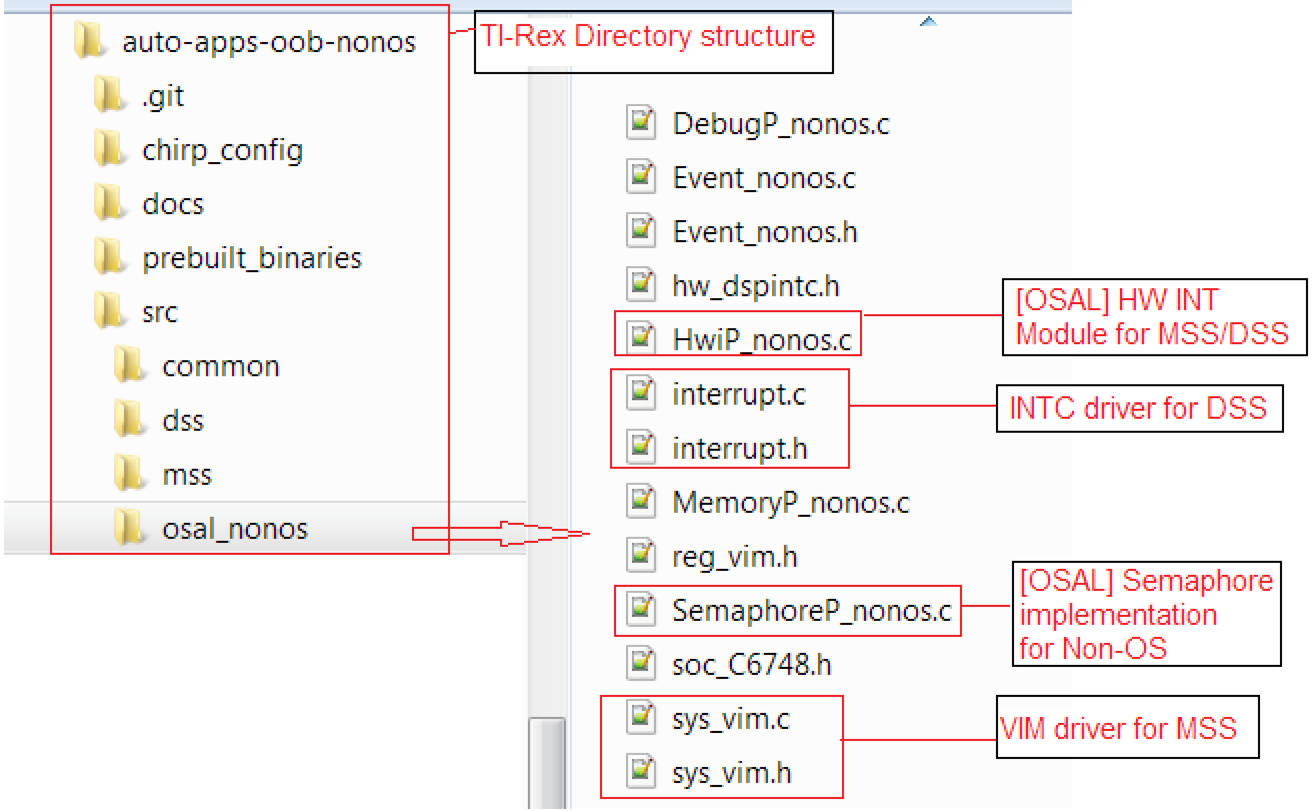
Lab Overview

Application Control Flow

- Control flow of OOB Non-OS at MSS and DSS cores are shown here
- DataPath processing chain is same as default mmw demo, please refer doxygen document available in mmWave-SDK.
- The following plots are available:
  - Scatter Plot
  - Range Profile
  - Noise Profile
  - Range Azimuth Heat Map
  - Range Doppler Heat Map
  - Statistics


MSS
DSS
ESM & SOC Init And other peripheral init
SOC Init And other peripheral init
Set MSS Link State & Synchronize with DSS core
Set DSS Link State & Synchronize with MSS core
Initialize and setup CLI interface
Initialize mmWaveLink to communicate with RadarSS over mailbox
Get RadarSS version through mmWaveLink
Send Version Info to MSS over Mailbox
Listen for CLI command
DataPath Initialization
Send CLI Config to DSS over Mailbox
Read & process DSS mailbox message
Read & process MSS mailbox message
mmWaveLink API management
DataPath process based on chirp/frame event
Exec state and Object info to MSS over Mailbox
Check for Fault or Exception Events
[nonOsLoop]
[nonOsLoop]

Lab Overview
OSAL Implementation with Non-OS Env.



- auto-apps-oob-nonos
  - .git
  - chirp_config
  - docs
  - prebuilt_binaries
  - src
    - common
    - dss
    - mss
    - osal_nonos

TI-Rex Directory structure

- DebugP_nonos.c
- Event_nonos.c
- Event_nonos.h
- hw_dspintc.h
- HwiP_nonos.c
- interrupt.c
- interrupt.h
- MemoryP_nonos.c
- reg_vim.h
- SemaphoreP_nonos.c
- soc_C6748.h
- sys_vim.c
- sys_vim.h

[OSAL] HW INT Module for MSS/DSS

INTC driver for DSS

[OSAL] Semaphore implementation for Non-OS

VIM driver for MSS

Lab Overview

Memory Comparison with OS App

Free Memory	Non-OS	OS
DSS		
L2-UMAP1 [128KB]	14.8 KB	1.28 KB
L2-UMAP0 [128KB]	60 KB	35.5 KB
L1P-SRAM	7.7 KB	1.1 KB
L3-SRAM	389.6 KB	381.8 KB
MSS		
PROG_RAM	209 KB	172 KB
DATA_RAM	178 KB	133 KB
MetaImage Size	196 KB	264 KB

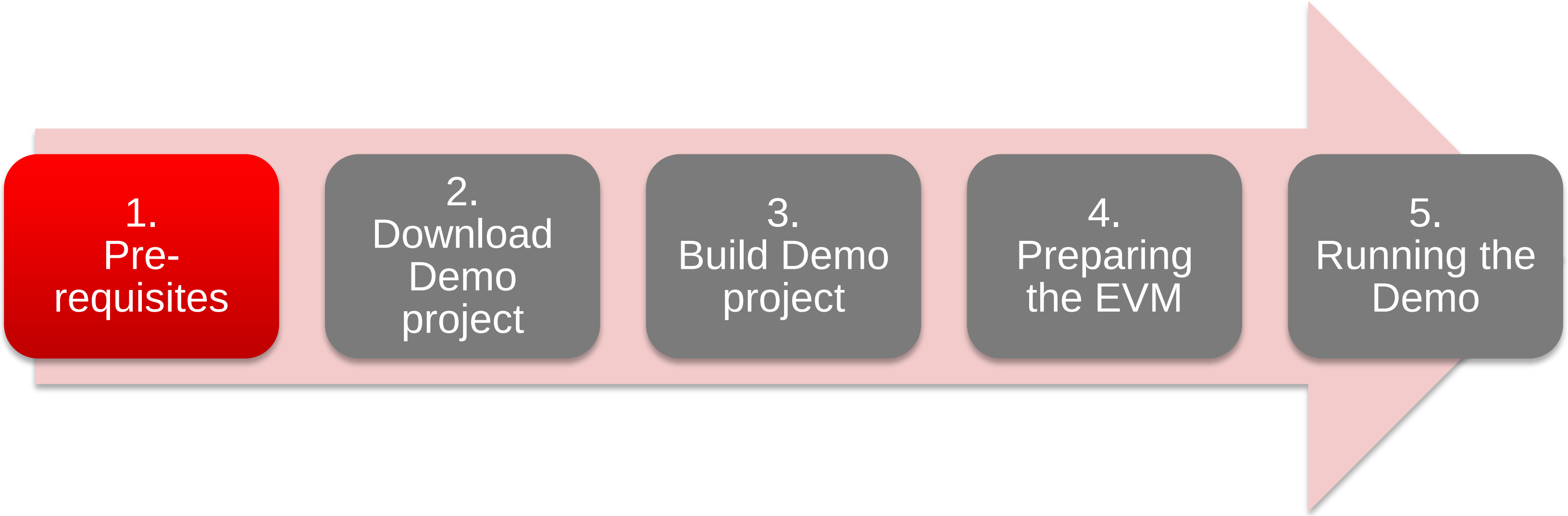
Note- In this comparison LVDS and Monitoring features are removed at DSS app.

1. Requirements

- Software
  - **Pre-requisites**
    - TI mmWave SDK specified in Lab release notes, and all related dependencies installed as mentioned in the mmWave SDK release notes.
  - Google Chrome with TI Cloud Agent Extension
    - For running the mmWave Demo Visualizer
    - Download from TI Cloud Agent or install when accessing the demo
  - mmWave SDK Demo
    - Download from TI Resource Explorer
  - UniFlash
    - For flashing firmware images onto
    - Download from TI.com/tool/uniflash
  - XDS110 Drivers
    - For EVM XDS device support
    - Included with CCS Installation, or standalone through TI XDS Emulation Software

- Hardware
  - xWR16xx ES2.0 EVM
  - Micro USB cable (included in the EVM package)
  - 5V/2.5A Power Supply
    - Purchase from Digikey

Steps



1. Pre-requisites

2. Download Demo project

3. Build Demo project

4. Preparing the EVM

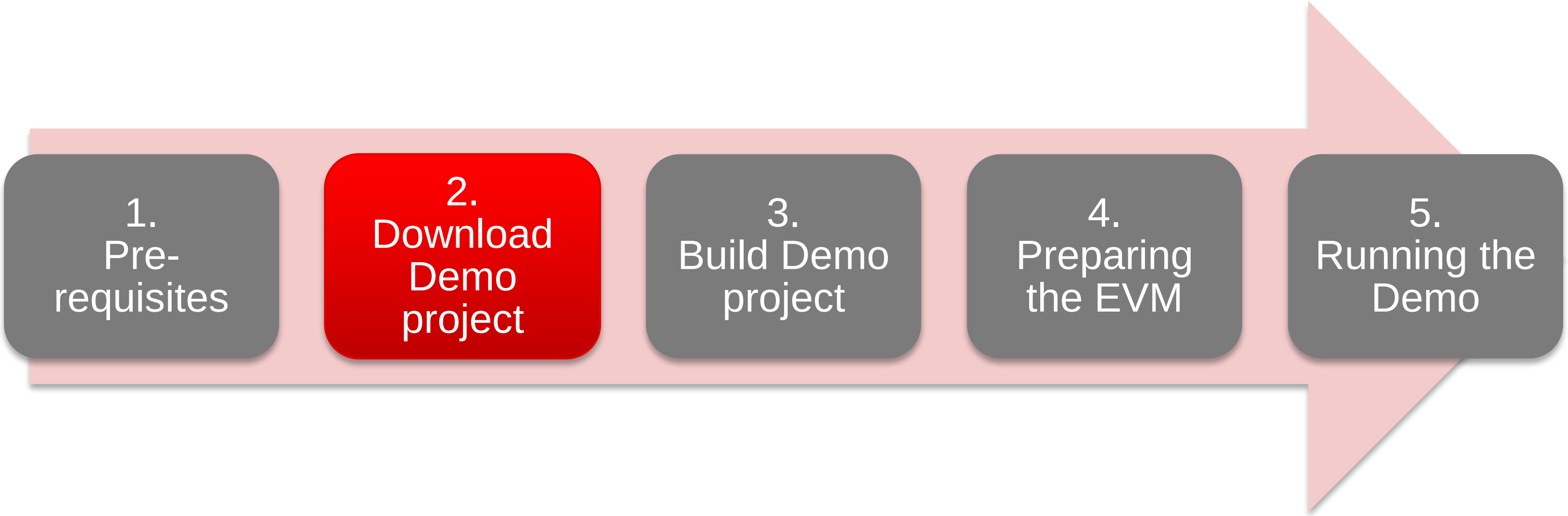
5. Running the Demo

1. Pre-requisites

- It is assumed that you have the mmWave SDK specified in Lab release notes and all related dependencies installed as mentioned in the mmWave SDK release notes.
  - The mmWave SDK release notes include the links for downloading the required tools.
  - Helpful Tips
    - Beginning with SDK 1.1.0.2, the mmwave SDK installer automatically installs the correct versions of the required tools (except CCS)
    - Beginning with SDK 1.1.0.2, PERL and crc.pm are no longer required
- If you have already installed the mmWave SDK and all the required tools, you can move on to the next step i.e. downloading the lab on to your machine.

Tool	Version	Download Link
mmWave SDK	2.0.0.4	download link
CCS	7.4 or later	download link
TI ARM Compiler	16.9.6.LTS	Included in mmwave sdk installer
TI CGT Compiler	8.1.3	Included in mmwave sdk installer
XDC	3.50.04.43	Included in mmwave sdk installer
C64x+ DSPLIB	3.4.0.0	Included in mmwave sdk installer
C674x DSPLIB	3.4.0.0	Included in mmwave sdk installer
C674x MATHLIB (little-endian, elf/coff format)	3.1.2.1	Included in mmwave sdk installer
mmwave Radar device support packages	1.5.9 or later	Upgrade to the latest using CCS update process (see SDK user guide for more details)
TI Emulators package	7.0.188.0 or later	Upgrade to the latest using CCS update process (see SDK user guide for more details)
Uniflash	latest	Uniflash tool is used for flashing xWR1xxx devices Cloud version (Recommended): https://dev.ti.com/uniflash Offline version: http://www.ti.com/tool/uniflash
mmWave Demo Visualizer	latest	TI Gallery APP for configuring mmWave sensors and visualizing the point cloud objects generated by the mmWave SDK demo https://dev.ti.com/mmWaveDemoVisualizer

Steps



1. Pre-requisites

2. Download Demo project

3. Build Demo project

4. Preparing the EVM

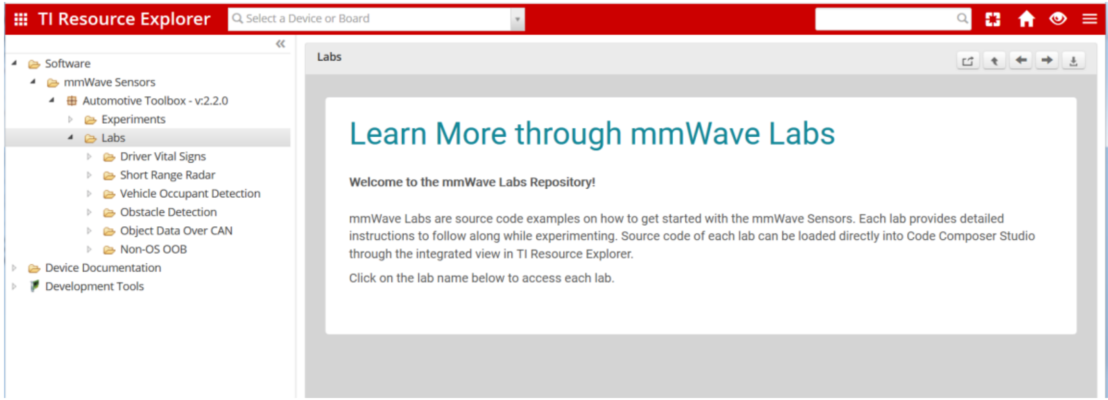
5. Running the Demo

2. Download the Lab project

- The nonOS-oob projects are available under **mmWave Sensors ► Automotive Toolbox** in CCS Resource Explorer.

- To download the nonOS-oob demo, start CCS v7.4 (or later) and select **View ► Resource Explorer** to open the Resource Explorer.

- In the Resource Explorer Window, select **Software ► mmWave Sensors ► Automotive toolbox ► Labs**.



Learn More through mmWave Labs

Welcome to the mmWave Labs Repository!

mmWave Labs are source code examples on how to get started with the mmWave Sensors. Each lab provides detailed instructions to follow along while experimenting. Source code of each lab can be loaded directly into Code Composer Studio through the integrated view in TI Resource Explorer.

Click on the lab name below to access each lab.

2. Download - continued

1

2
Download Demo project

3

4

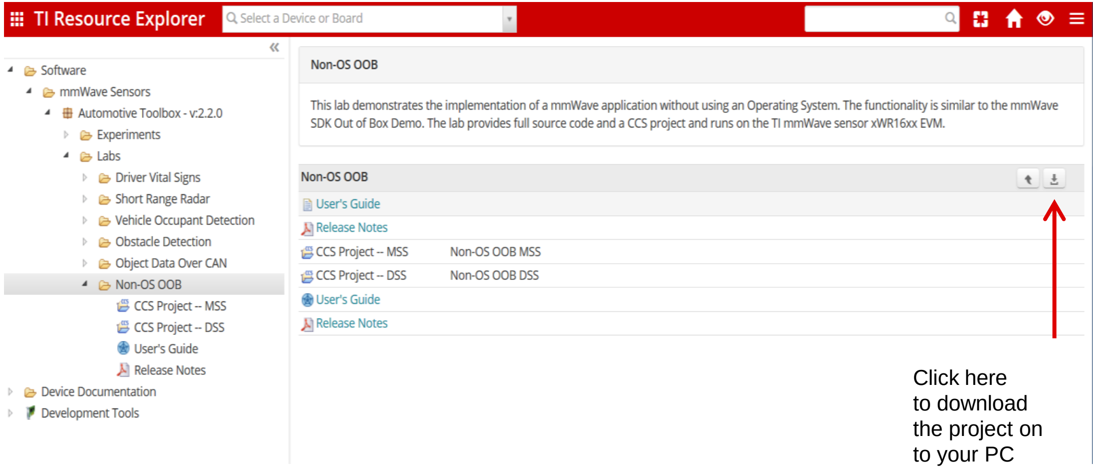
5

- Select the **16xx nonOS OOB** demo in the left view.

- The right view shows the contents of the Lab which contains the CCS Project and the PC GUI.

- Click on the **Download and Install** button ⤓ in the top right corner as shown.

- Select the **Make Available Offline** option from the drop down to start downloading the Lab.



- Software
  - mmWave Sensors
    - Automotive Toolbox - v:2.2.0
      - Experiments
      - Labs
        - Driver Vital Signs
        - Short Range Radar
        - Vehicle Occupant Detection
        - Obstacle Detection
        - Object Data Over CAN
        - Non-OS OOB
          - CCS Project -- MSS
          - CCS Project -- DSS
          - User's Guide
          - Release Notes
- Device Documentation
- Development Tools

Non-OS OOB

This lab demonstrates the implementation of a mmWave application without using an Operating System. The functionality is similar to the mmWave SDK Out of Box Demo. The lab provides full source code and a CCS project and runs on the TI mmWave sensor xWR16xx EVM.

Non-OS OOB

User's Guide	
Release Notes	
CCS Project -- MSS	Non-OS OOB MSS
CCS Project -- DSS	Non-OS OOB DSS
User's Guide	
Release Notes	

Click here to download the project on to your PC

2. Download - continued

1

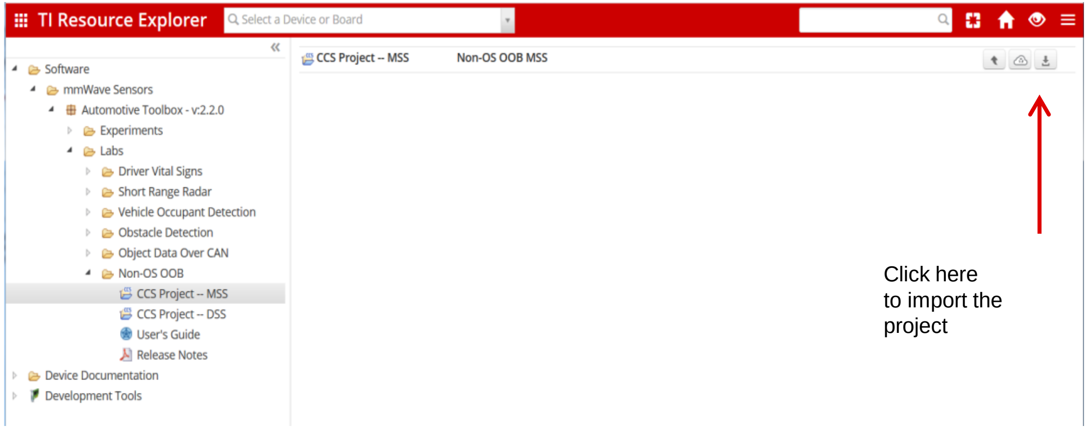
2
Download Demo project

3

4

5

- The AWR16xx nonOS OOB Demo consists of two CCS projects, one for the R4F core and one for the C674x DSP core
- The project will be downloaded in C:\ti\mmwave_automotive_toolbox
- Select the CCS Project MSS file in the left view
- Click on the **Import to IDE** button which should be visible in the right side view after a successful download.
- This copies the project in the user's workspace and imports it into the CCS project explorer.
  - It is important to note that the copy created in the workspace is the one that gets imported in CCS. The original project downloaded in mmwave_industrial_toolbox is not modified.
- Repeat with the CCS Project DSS file


TI Resource Explorer

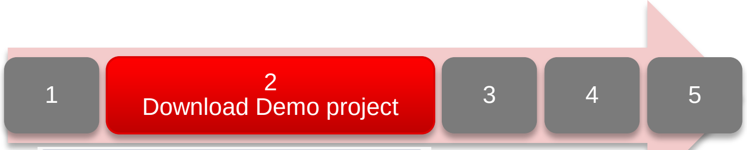
Select a Device or Board

- Software
  - mmWave Sensors
    - Automotive Toolbox - v:2.2.0
      - Experiments
      - Labs
        - Driver Vital Signs
        - Short Range Radar
        - Vehicle Occupant Detection
        - Obstacle Detection
        - Object Data Over CAN
        - Non-OS OOB
          - CCS Project -- MSS
          - CCS Project -- DSS
          - User's Guide
          - Release Notes
- Device Documentation
- Development Tools

CCS Project -- MSS Non-OS OOB MSS

Click here to import the project

2. Download - continued


1

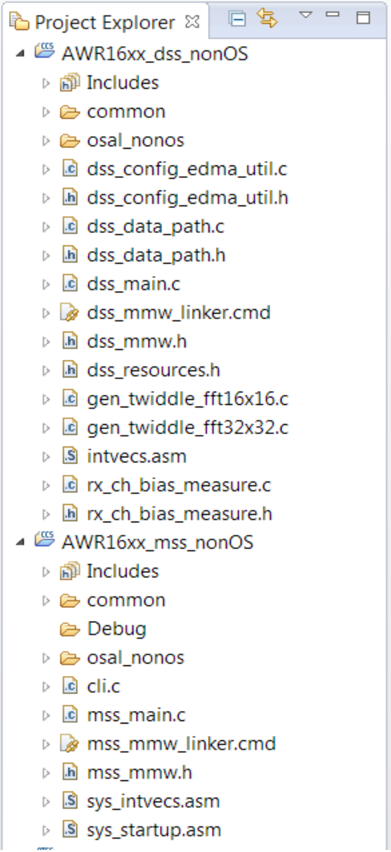
2
Download Demo project

3

4

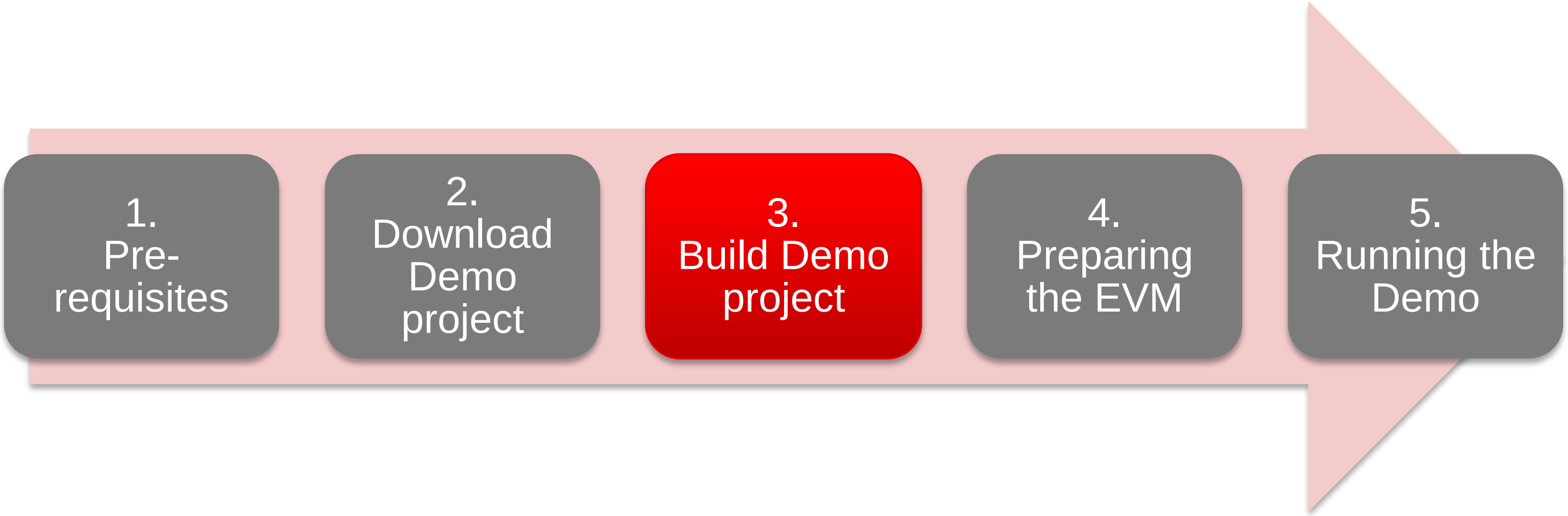
5

- After successfully completing the **Import to IDE** operation, the both projects should be visible in CCS Project Explorer as shown here.
- At this point, we have successfully downloaded the nonOS OOB demo and imported it in CCS.
- We are ready to move on to the next step i.e. Building the projects.


Project Explorer

```
AWR16xx_dss_nonOS
    Includes
    common
    osal_nonos
    dss_config_edma_util.c
    dss_config_edma_util.h
    dss_data_path.c
    dss_data_path.h
    dss_main.c
    dss_mmw_linker.cmd
    dss_mmw.h
    dss_resources.h
    gen_twiddle_fft16x16.c
    gen_twiddle_fft32x32.c
    intvecs.asm
    rx_ch_bias_measure.c
    rx_ch_bias_measure.h
AWR16xx_mss_nonOS
    Includes
    common
    Debug
    osal_nonos
    cli.c
    mss_main.c
    mss_mmw_linker.cmd
    mss_mmw.h
    sys_intvecs.asm
    sys_startup.asm
```

Steps



1. Pre-requisites

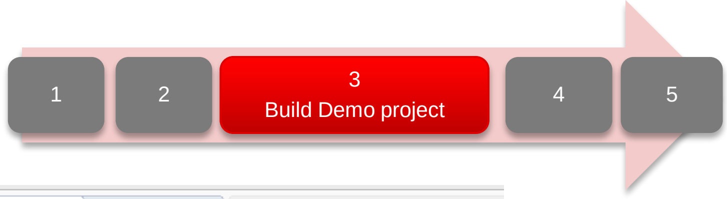
2. Download Demo project

3. Build Demo project

4. Preparing the EVM

5. Running the Demo

3. Build the Lab



1

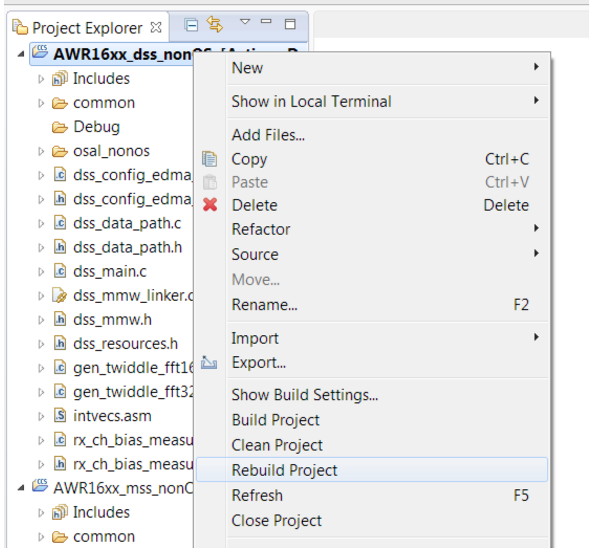
2

3
Build Demo project

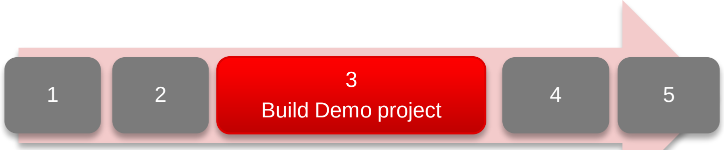
4

5

- With the AWR16xx_dss_nonOS project selected in Project Explorer, right click on the project and select **Rebuild Project**.
  - Selecting **Rebuild** instead of **Build** ensures that the project is always re-compiled. This is especially important in case the previous build failed with errors.
- On successful completion of the build, you should see the output in CCS console as shown here and the following two files should be produced in the project debug directory
  - xwr16xx_mmw_dss_nonOS.xe674
  - xwr16xx_mmw_dss_nonOS.bin (note, this image is not flashed directly. It is merged into a combined meta image when building the MSS; shown on the next page).
- If the build fails with errors, please ensure that all the pre-requisites are installed as mentioned in the mmWave SDK release notes.
  - Please note that pre-built binary files are provided with the demo under mmwave SDK.
  - Look under <mmwave_sdk_install_dir>\packages\ti\demo\xwr16xx\mmw



3. Build the Lab

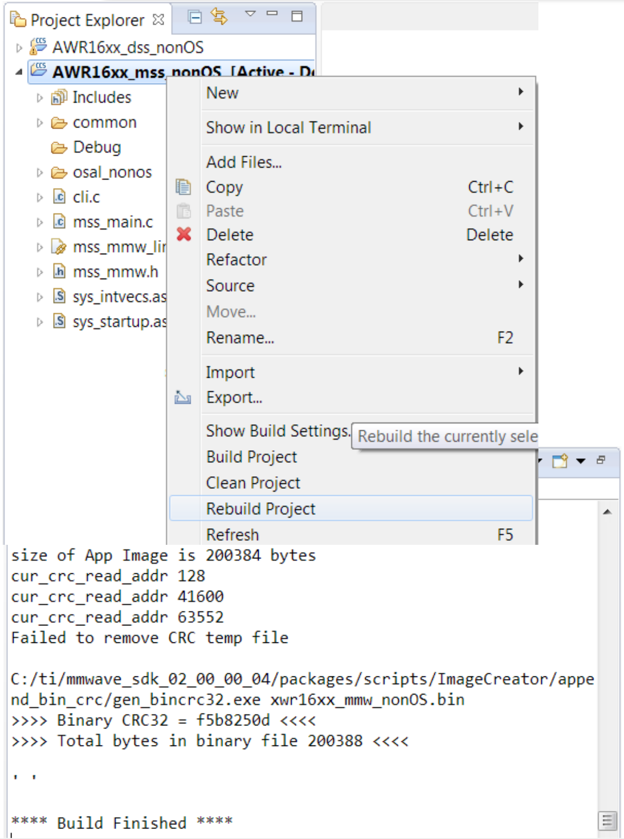


- The AWR16xx_dss_nonOS project must be built BEFORE the AWR16xx_mmw_nonOS project is built.

- With the AWR16xx_mmw_nonOS project selected in Project Explorer, right click on the project and select **Rebuild Project**.

- On successful completion of the MSS build, you should see the output in CCS console as shown here and the following three files should be produced in the project debug directory
  - xwr16xx_mmw_mss_nonOS.xer4f
  - xwr16xx_mmw_nonOS.bin
  - xwr16xx_mmw_nonOS.bin (this is the meta image to be flashed)

- If the build fails with errors, please ensure that all the pre-requisites are installed as mentioned in the mmWave SDK release notes.
  - Please note that pre-built binary files are provided with the demo under mmwave SDK.
  - Look under <mmwave_sdk_install_dir>\packages\ti\demo\xwr16xx\mmw



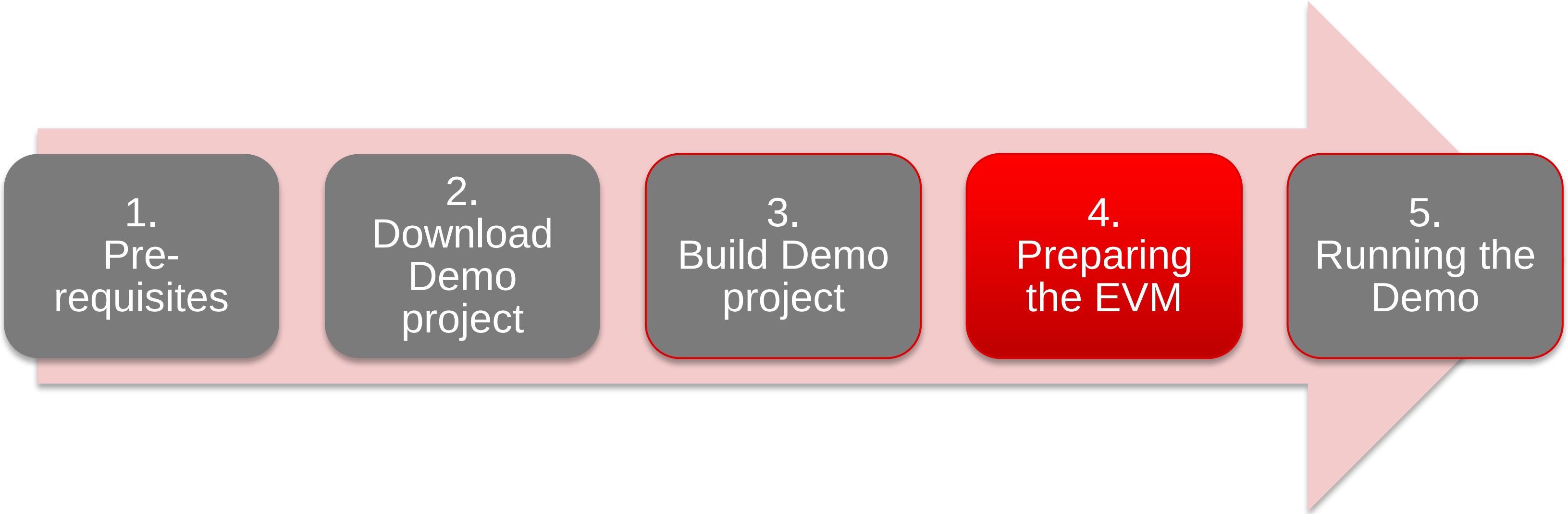
```
size of App Image is 200384 bytes
cur_crc_read_addr 128
cur_crc_read_addr 41600
cur_crc_read_addr 63552
Failed to remove CRC temp file

C:/ti/mmwave_sdk_02_00_00_04/packages/scripts/ImageCreator/appe
nd_bin_crc/gen_bincrc32.exe xwr16xx_mmw_nonOS.bin
>>>> Binary CRC32 = f5b8250d <<<<
>>>> Total bytes in binary file 200388 <<<<

' '

**** Build Finished ****
```

Steps



1. Pre-requisites

2. Download Demo project

3. Build Demo project

4. Preparing the EVM

5. Running the Demo

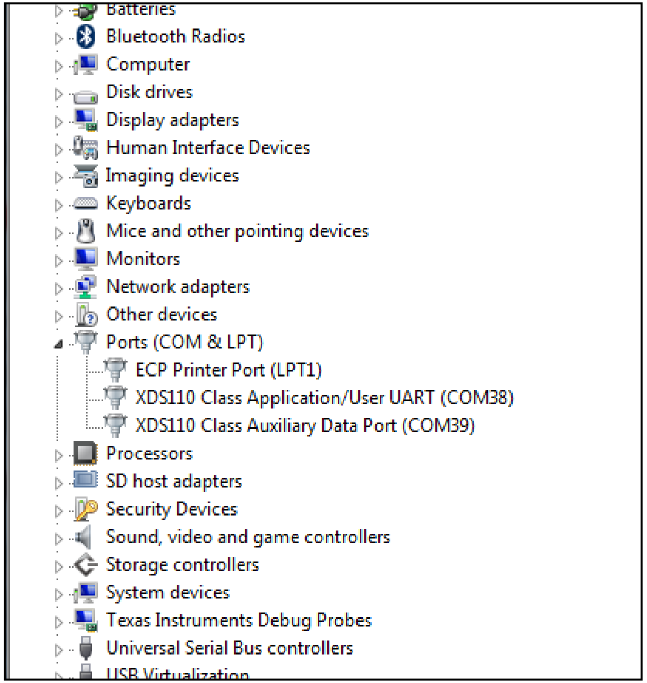
4.1 Preparing the EVM

- There are two ways to execute the compiled code on the EVM:
  - Deployment mode: Flashing the binary (.bin image) on to the EVM serial flash
    - In this mode, the EVM boots autonomously from flash and starts running the bin image.
  - Debug mode: Downloading and running the executable (.xer4f image and .xe674) from CCS.
    - You will need to flash a small CCS debug firmware on the EVM (one time) to allow connecting with CCS. This debug firmware image is provided with the mmWave SDK.
  - As a recap, the build process in Step 3 produces the .bin .xer4f and .xe674 images.

- This presentation explains the second method i.e. Debug mode (CCS).
  - To prepare the EVM for debug mode, we start with flashing the CCS debug firmware image.
  - Please note that the same flashing process can be used to flash the Lab binary to run it in deployment mode.

4.2 Connecting to the EVM

- Power on the EVM using a 5V/2.5A power supply.
- Connect the EVM to your PC and check the COM ports in Windows Device Manager
- The EVM exports two virtual COM ports as shown below:
  - XDS110 Class Application/User UART (COM_{UART}):
    - Used for passing configuration data and firmware to the EVM
  - XDS110 Class Auxiliary Data Port (COM_{AUX})
    - Used to send processed radar data output
- Note the COM_{UART} and COM_{AUX} port numbers, as they will be used later for flashing and running the Lab.



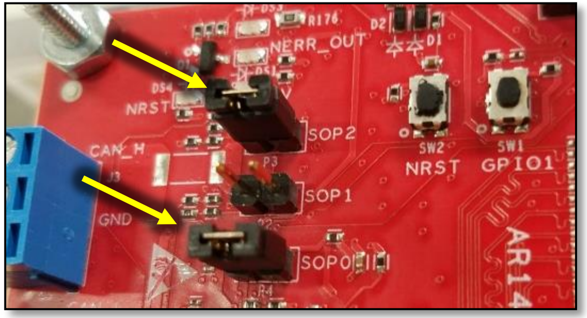
- Batteries
- Bluetooth Radios
- Computer
- Disk drives
- Display adapters
- Human Interface Devices
- Imaging devices
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Other devices
- Ports (COM & LPT)
  - ECP Printer Port (LPT1)
  - XDS110 Class Application/User UART (COM38)
  - XDS110 Class Auxiliary Data Port (COM39)
- Processors
- SD host adapters
- Security Devices
- Sound, video and game controllers
- Storage controllers
- System devices
- Texas Instruments Debug Probes
- Universal Serial Bus controllers
- USB Virtualization

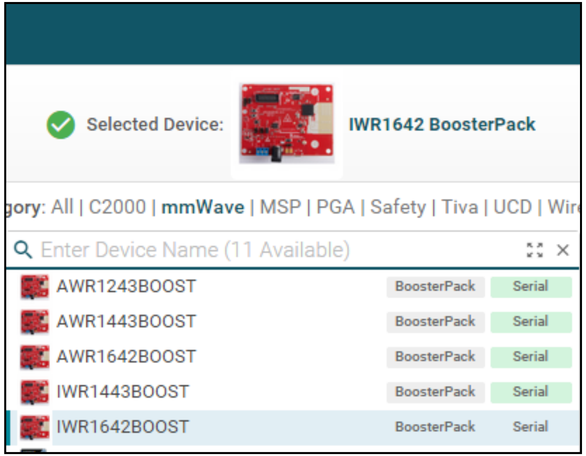
COM_{UART}: COM38 **COM_{AUX}**: COM39

- The actual port numbers on your machine may be different

4.3 Flashing CCS debug firmware

1. Put the EVM in flashing mode by connecting jumpers on SOP0 and SOP2 as shown in the image.
2. Open the **UniFlash** tool
3. In the **New Configuration** section, locate and select AWR1642BOOST
4. Click **Start** to proceed





Selected Device: IWR1642 BoosterPack

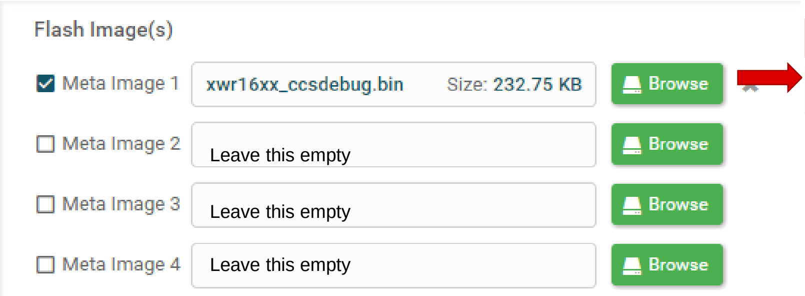
gory: All | C2000 | **mmWave** | MSP | PGA | Safety | Tiva | UCD | Wir

Enter Device Name (11 Available)

AWR1243BOOST	BoosterPack	Serial
AWR1443BOOST	BoosterPack	Serial
AWR1642BOOST	BoosterPack	Serial
IWR1443BOOST	BoosterPack	Serial
IWR1642BOOST	BoosterPack	Serial

4.3 Flashing CCS debug firmware

5. In the **Program** tab, browse and locate the Radar SS and MSS images shown below:



Flash Image(s)

☑ Meta Image 1 xwr16xx_ccsdebug.bin Size: 232.75 KB Browse

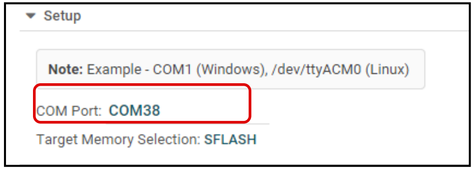
☐ Meta Image 2 Leave this empty Browse

☐ Meta Image 3 Leave this empty Browse

☐ Meta Image 4 Leave this empty Browse

Image	Location
Meta Image 1	C:\ti\mmwave_sdk_<ver>\packages\ti\utils\ccsdebug

6. In the **Settings & Utilities** tab, fill the **COM Port** text box with the Application/User UART COM port number (**COM$_{\text{UART}}$**) noted earlier



Setup

Note: Example - COM1 (Windows), /dev/ttyACM0 (Linux)

COM Port: COM38

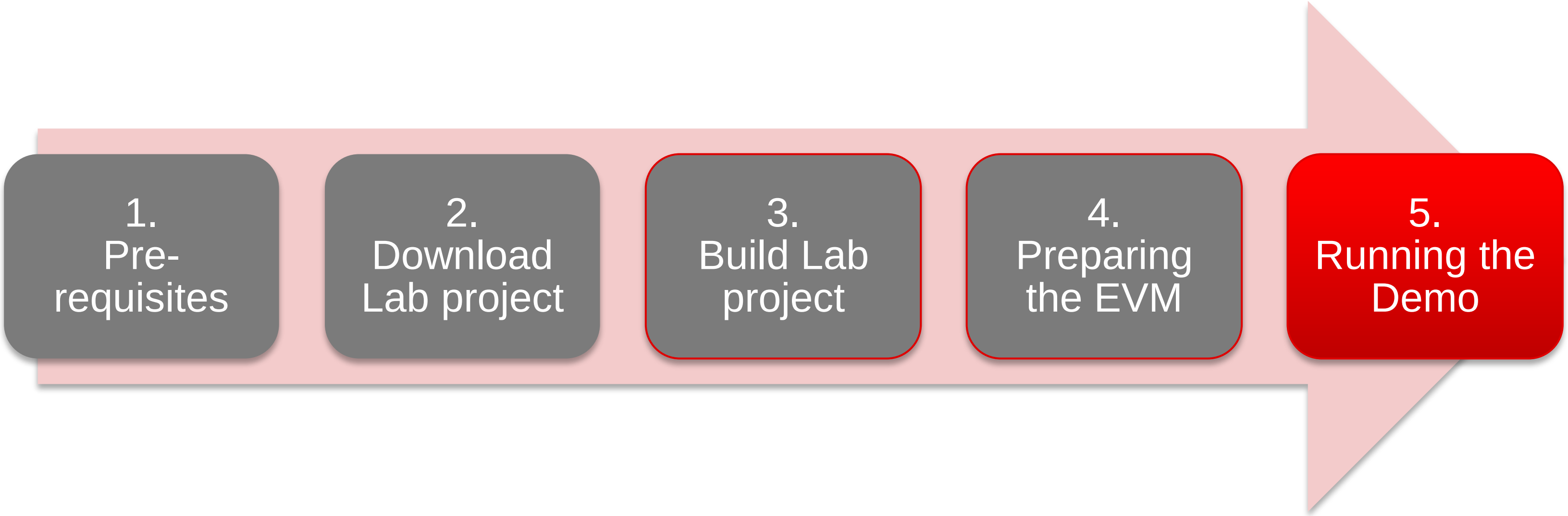
Target Memory Selection: SFLASH

7. Return to the **Program** tab, power cycle the device and click on **Load Images**

8. When the flash procedure completes, UniFlash's console should indicate: [SUCCESS] Program Load completed successfully

9. Power off the board and remove the jumper from only header **SOP2.** Power the board back on (this puts the board back in functional mode)

Steps



1. Pre-requisites

2. Download Lab project

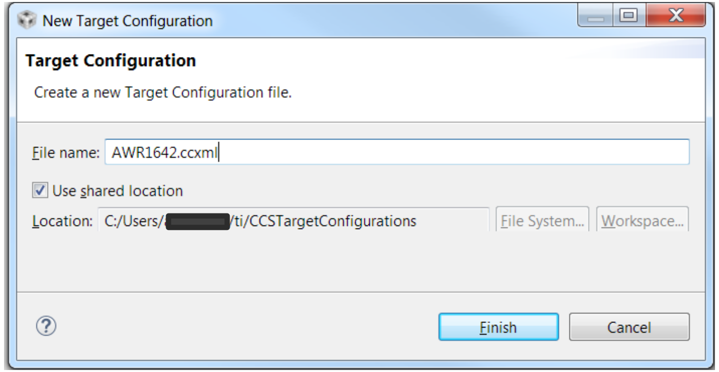
3. Build Lab project

4. Preparing the EVM

5. Running the Demo

5.1 Connecting EVM to CCS

- It is assumed that you were able to download and build the Lab in CCS (completed steps 1, 2 and 3)
- To connect the Radar EVM to CCS, we need to create a target configuration
  - Go to File ► New ► New Target Configuration File
  - Name the target configuration accordingly and check the “Use shared location” checkbox. Press Finish
  - In the configuration editor window:
    - Select “Texas Instruments XDS110 USB Debug Probe” for **Connection**
    - Select **IWR1642** or **AWR1642** in the **Board or Device** list
    - Press the **Save** button to save the target configuration.
    - You can press the **Test Connection** button to check the connection with the board.


New Target Configuration

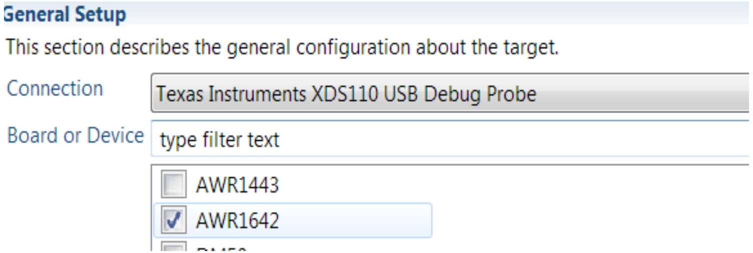
Target Configuration

Create a new Target Configuration file.

File name: AWR1642.ccxml

☑ Use shared location

Location: C:/Users/[redacted]/ti/CCSTargetConfigurations


General Setup

This section describes the general configuration about the target.

Connection: Texas Instruments XDS110 USB Debug Probe

Board or Device: type filter text

☐ AWR1443

☑ AWR1642

5.1 Connecting - continued

- Go to **View ► Target Configurations** to open the target configuration window.

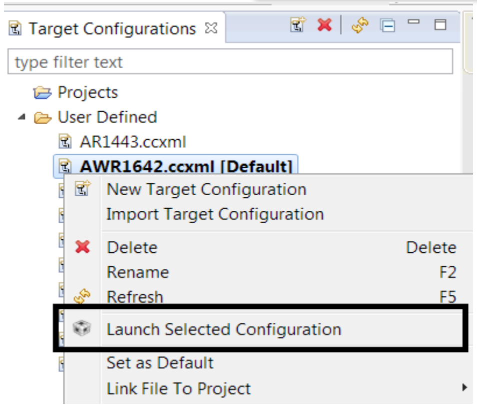
- You should see your target configuration under **User Defined** configurations.

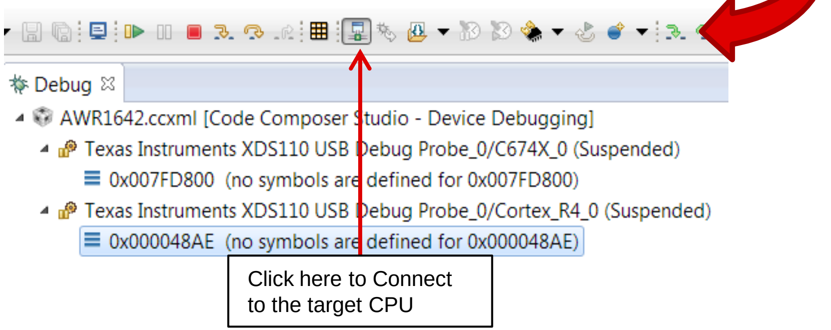
- With the board powered on, right click on the target configuration and select **Launch Select Configuration**.

- This will launch the target configuration in the debug window.

- Select the Texas Instruments XDS110 USB Debug probe/C674X_0 and press the **Connect Target** button

- Select the Texas Instruments XDS110 USB Debug probe/Cortex_R4_0and press the **Connect Target** button



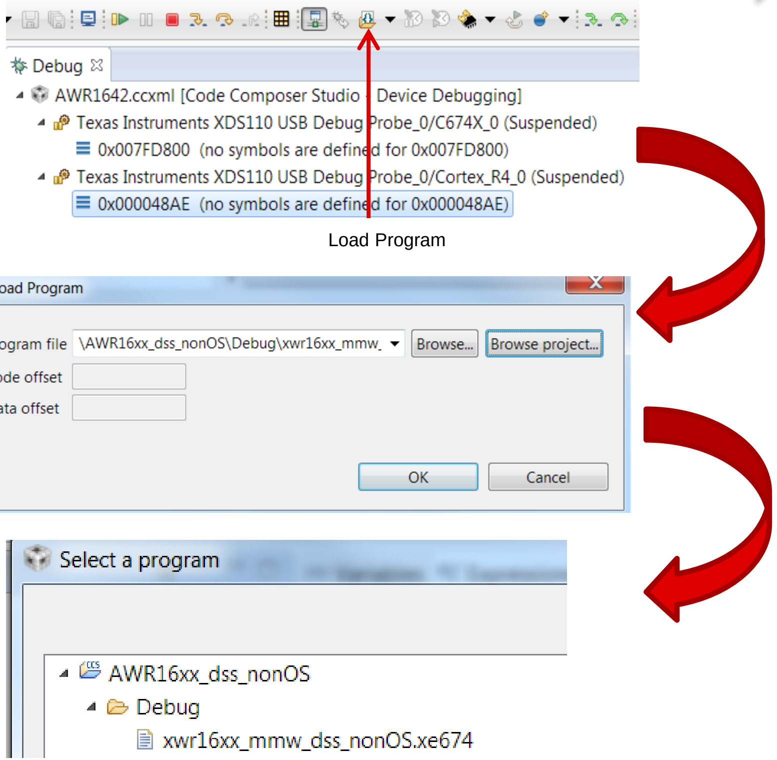


AWR1642.ccxml [Code Composer Studio - Device Debugging]
Texas Instruments XDS110 USB Debug Probe_0/C674X_0 (Suspended)
0x007FD800 (no symbols are defined for 0x007FD800)
Texas Instruments XDS110 USB Debug Probe_0/Cortex_R4_0 (Suspended)
0x000048AE (no symbols are defined for 0x000048AE)

Click here to Connect to the target CPU

5.2 Loading the binary

- Once both targets are connected, select the C674X_0 target, and click on the **Load** button in the toolbar.
- In the **Load Program** dialog, press the **Browse Project** button .
- Select the lab executable (.xe674) found in the AWR16xx_dss_nonOS project as shown, and press OK.
- Press OK again in the **Load Program** dialog.



Debug

AWR1642.ccxml [Code Composer Studio - Device Debugging]

Texas Instruments XDS110 USB Debug Probe_0/C674X_0 (Suspended)

0x007FD800 (no symbols are defined for 0x007FD800)

Texas Instruments XDS110 USB Debug Probe_0/Cortex_R4_0 (Suspended)

0x000048AE (no symbols are defined for 0x000048AE)

Load Program

Load Program

Program file: \AWR16xx_dss_nonOS\Debug\xwr16xx_mmw_

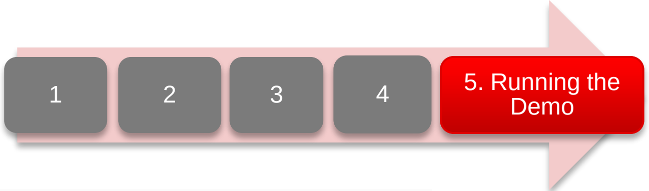
Code offset:

Data offset:

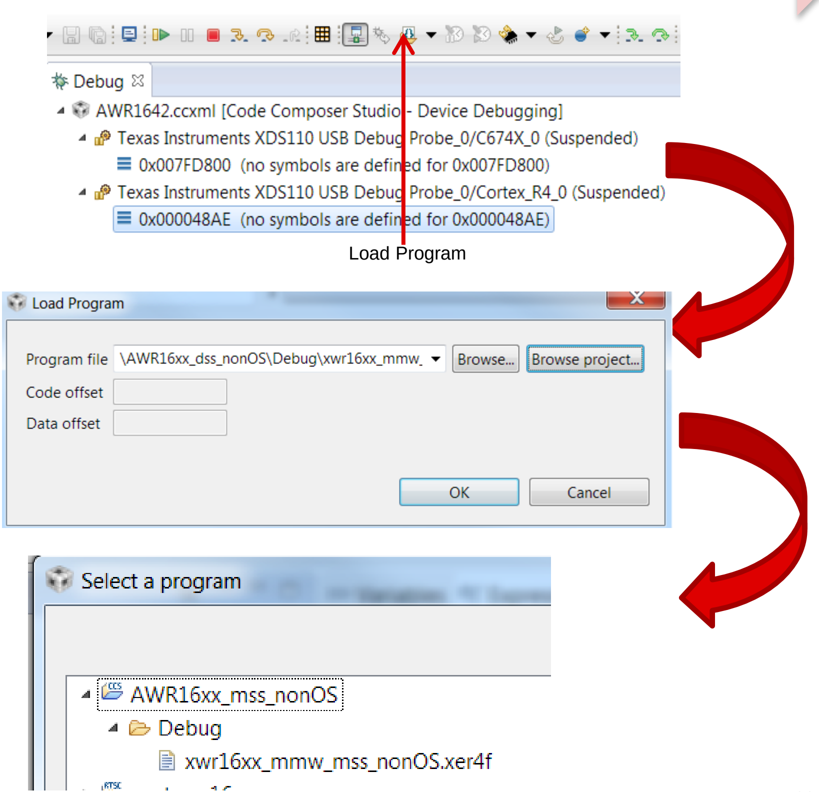
Select a program

AWR16xx_dss_nonOS
Debug
xwr16xx_mmw_dss_nonOS.xe674

5.2 Loading the binary



- Now select the Cortex_R4_0 target, and click on the **Load** button in the toolbar.
- In the **Load Program** dialog, press the **Browse Project** button .
- Select the lab executable (.xer4f) found in the AWR16xx_mss_nonOS project as shown, and press OK.
- Press OK again in the **Load Program** dialog.



Debug

AWR1642.ccxml [Code Composer Studio - Device Debugging]

Texas Instruments XDS110 USB Debug Probe_0/C674X_0 (Suspended)

0x007FD800 (no symbols are defined for 0x007FD800)

Texas Instruments XDS110 USB Debug Probe_0/Cortex_R4_0 (Suspended)

0x000048AE (no symbols are defined for 0x000048AE)

Load Program

Load Program

Program file: \AWR16xx_dss_nonOS\Debug\xwr16xx_mmw_

Code offset:

Data offset:

Select a program

AWR16xx_mss_nonOS

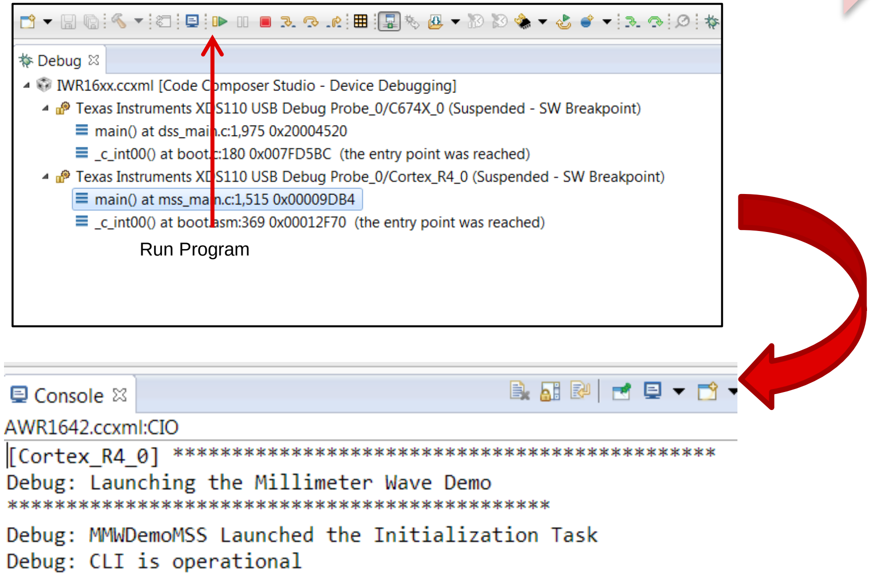
Debug

xwr16xx_mmw_mss_nonOS.xer4f

5.3 Running the binary

- With both executables loaded, select mss_main.c, as shown, and press the Run/Resume button

- The program should start executing and generate console output as shown.



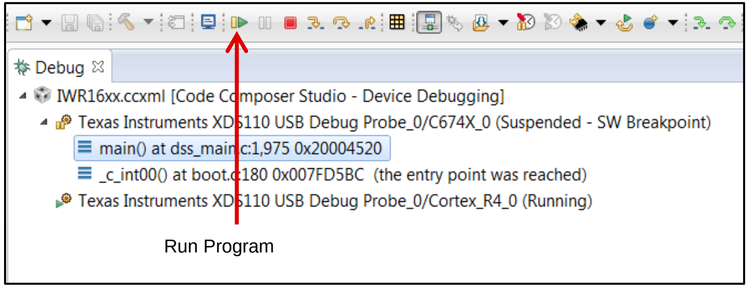
```
Debug
IWR16xx.ccxml [Code Composer Studio - Device Debugging]
  Texas Instruments XDS110 USB Debug Probe_0/C674X_0 (Suspended - SW Breakpoint)
    main() at dss_main.c:1,975 0x20004520
    _c_int00() at boot.c:180 0x007FD5BC  (the entry point was reached)
  Texas Instruments XDS110 USB Debug Probe_0/Cortex_R4_0 (Suspended - SW Breakpoint)
    main() at mss_main.c:1,515 0x00009DB4
    _c_int00() at boot.asm:369 0x00012F70  (the entry point was reached)
```

Run Program

```
Console
AWR1642.ccxml:CIO
[Cortex_R4_0] **********************************************
Debug: Launching the Millimeter Wave Demo
**********************************************
Debug: MMWDemoMSS Launched the Initialization Task
Debug: CLI is operational
```

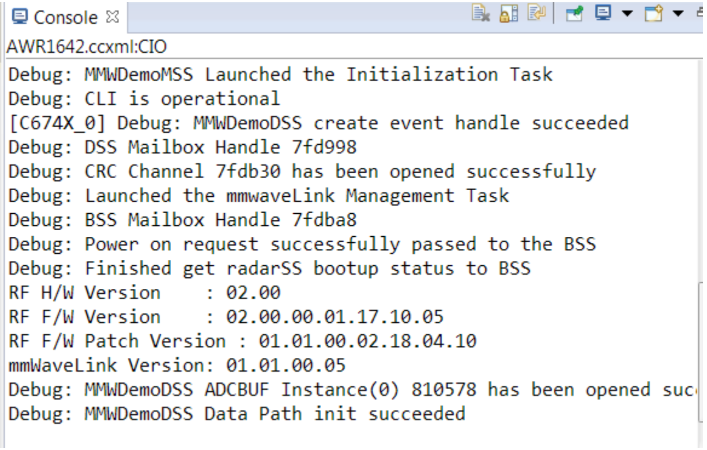
5.3 Running the binary

- Select dss_main.c, as shown, and press the Run/Resume button ▶
- Further console output should be generated as shown.
- You should see the “CLI is operational” message which indicates that the program is ready and waiting for the sensor configuration
- The sensor configuration is sent using the web GUI



```
IWR16xx.ccxml [Code Composer Studio - Device Debugging]
  Texas Instruments XDS110 USB Debug Probe_0/C674X_0 (Suspended - SW Breakpoint)
    main() at dss_main.c:1,975 0x20004520
    _c_int00() at boot.c:180 0x007FD5BC  (the entry point was reached)
  Texas Instruments XDS110 USB Debug Probe_0/Cortex_R4_0 (Running)
```

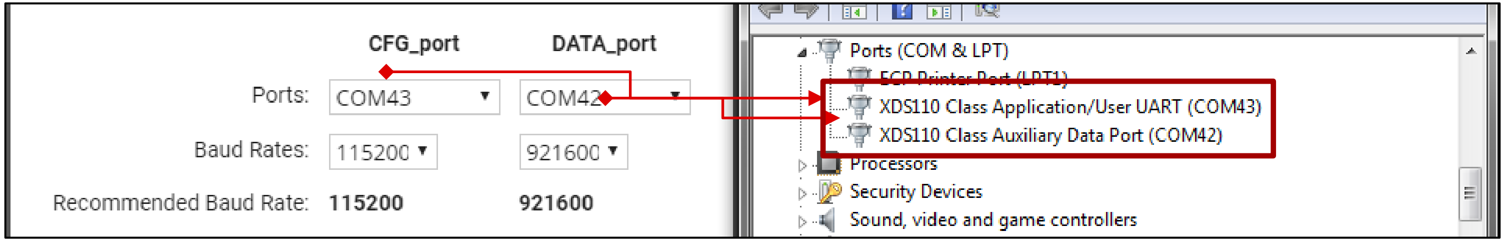
Run Program



```
AWR1642.ccxml:CIO
Debug: MMWDemoMSS Launched the Initialization Task
Debug: CLI is operational
[C674X_0] Debug: MMWDemoDSS create event handle succeeded
Debug: DSS Mailbox Handle 7fd998
Debug: CRC Channel 7fdb30 has been opened successfully
Debug: Launched the mmwaveLink Management Task
Debug: BSS Mailbox Handle 7fdba8
Debug: Power on request successfully passed to the BSS
Debug: Finished get radarSS bootup status to BSS
RF H/W Version    : 02.00
RF F/W Version    : 02.00.00.01.17.10.05
RF F/W Patch Version : 01.01.00.02.18.04.10
mmWaveLink Version: 01.01.00.05
Debug: MMWDemoDSS ADCBUF Instance(0) 810578 has been opened suc
Debug: MMWDemoDSS Data Path init succeeded
```

5.4 Running the Lab GUI

1. Using Google Chrome, navigate to the following URL: https://dev.ti.com/mmWaveDemoVisualizer
   – Alternatively, go to https://dev.ti.com/gallery and search for “mmWave Demo Visualizer”
2. If prompted, follow the on-screen instructions for installing TI Cloud Agent
3. Once the demo is loaded, go to **Options → Serial Port**
4. In the serial port window, enter the appropriate port in each of the drop down menus based on your port numbers from Step 2



	CFG_port	DATA_port
Ports:	COM43	COM42
Baud Rates:	115200	921600
Recommended Baud Rate:	115200	921600

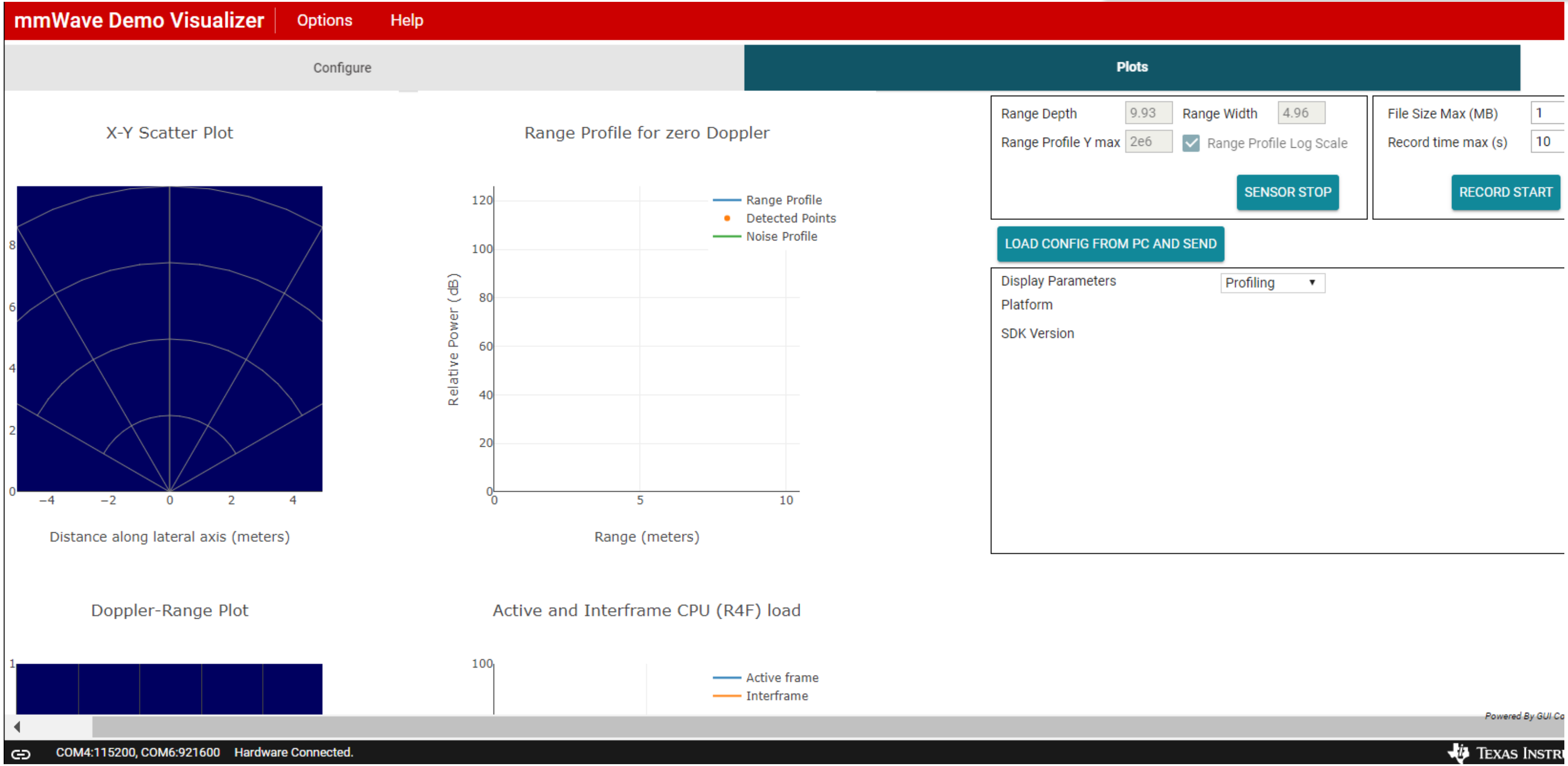
Ports (COM & LPT)
ECP Printer Port (LPT1)
XDS110 Class Application/User UART (COM43)
XDS110 Class Auxiliary Data Port (COM42)
Processors
Security Devices
Sound, video and game controllers

5.4 Running GUI - continued

5. Click on **Configure** and the demo will automatically connect to the EVM
   - Not connected: ⊂⊃ Connected: 🔗
   - If the connection fails, try clicking on the connection icon in the bottom left corner
6. Select the appropriate mmWave device from the **Platform** dropdown menu
7. Use the available options to create the desired configuration
   - Additional details about the configuration parameters can be found in the **mmWave Demo Visualizer User Guide**
8. When ready to send the configuration, click on **Send Config To mmWave Device**
9. Click on the **Plots** tab to view the plots that were selected to be shown
10. Move a highly reflective object in front of the EVM and see how the demo responds

5.4 Running GUI - continued

1 | 2 | 3 | 4 | 5. Running the Demo



Range Depth 9.93

Range Width 4.96

Range Profile Y max 2e6

☑ Range Profile Log Scale

File Size Max (MB) 1

Record time max (s) 10

Display Parameters Profiling ▾

Platform

SDK Version

COM4:115200, COM6:921600 Hardware Connected.

Learn more about TI mmWave Sensors

- Learn more about xWR1x devices, please visit the product pages
  - AWR1443: http://www.ti.com/product/AWR1443
  - AWR1642: http://www.ti.com/product/AWR1642
  - IWR1443: http://www.ti.com/product/IWR1443
  - IWR1642: http://www.ti.com/product/IWR1642
- Get started evaluating the platform with xWR1x EVMs, purchase EVM at
  - AWR1443 EVM: http://www.ti.com/tool/AWR1443BOOST
  - AWR1642 EVM: http://www.ti.com/tool/AWR1642BOOST
  - IWR1443 EVM: http://www.ti.com/tool/IWR1443BOOST
  - IWR1642 EVM: http://www.ti.com/tool/IWR1642BOOST
- Download mmWave SDK @ http://www.ti.com/tool/MMWAVE-SDK
- Ask question on TI's E2E forum mmWave Sensors forum @ https://e2e.ti.com/support/sensor/mmwave_sensors/

Texas Instruments