

AWR16xx Multi Gesture Demo

This version of the Multi-Gesture Demo lab will work only with **AWR1642ODS ES2.0** EVMs, which require **mmWave SDK version 3.1 or above**. These EVMs are marked with a sticker which says "ES2.0".

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 demonstrates the use of TI single-chip millimeter-wave (mmwave) technology for detection and classification of natural gestures. The example provided in this demo can recognize six different hand gestures: Left/Right swipe, Up/Down swipe and Swirl/Anit-swirl.
- Such a system can be used to implement gesture based Human Machine Interfaces (HMI)
- In addition, the lab also demonstrates motion detection at ranges up to about 1.5 feet and power save features.
- Power save mode is nothing but using low intensity configuration with very loose chirp width. Based on motion detection in near field, application switches to intense configuration where it can detect the valid Gesture.
- The lab provides full source code with CCS projects and runs on the TI [AWR1642ODS](#) 77GHz mmWave sensor Evaluation Platform.
- Directly run the demo, flash the pre-build binary and refer section 5.4

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.

- Multi-Gesture Demo

- Download from [TI Resource Explorer](#)

- UniFlash

- For flashing application MetaImage 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

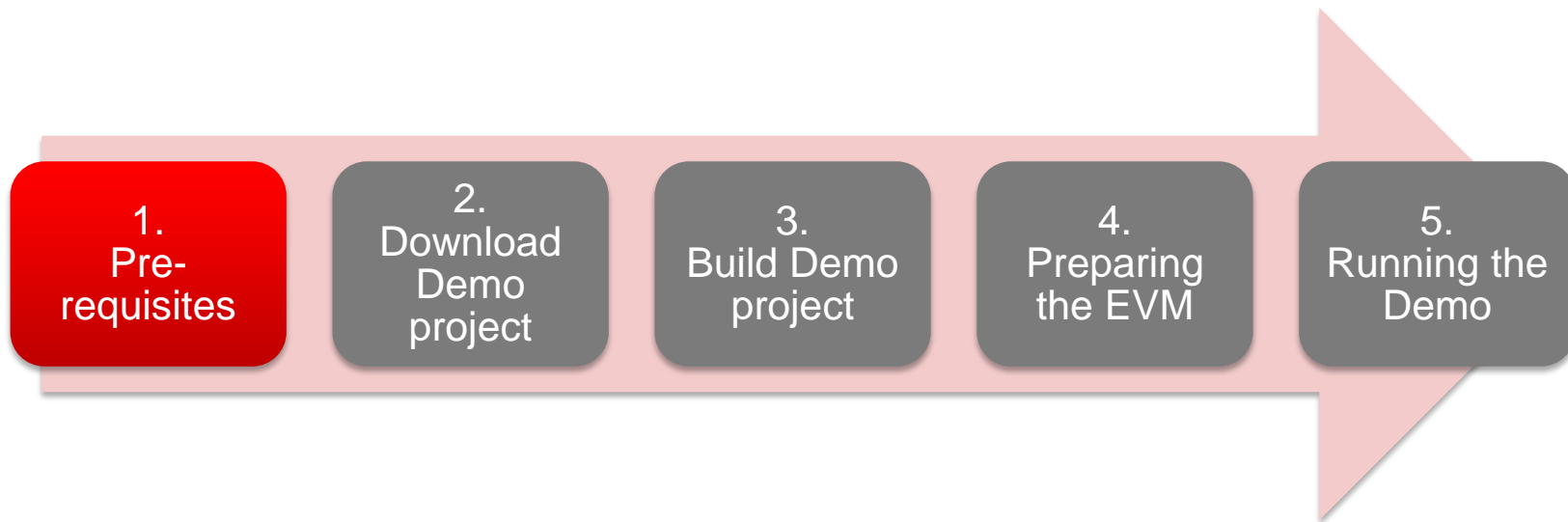
- [AWR1642ODS](#) ES2.0 EVM

- Micro USB cable (included in the EVM package)

- 5V/2.5A Power Supply

- [Purchase from Digikey](#)

Steps



1. Pre-requisites

1. Install Pre-requisites

2

3

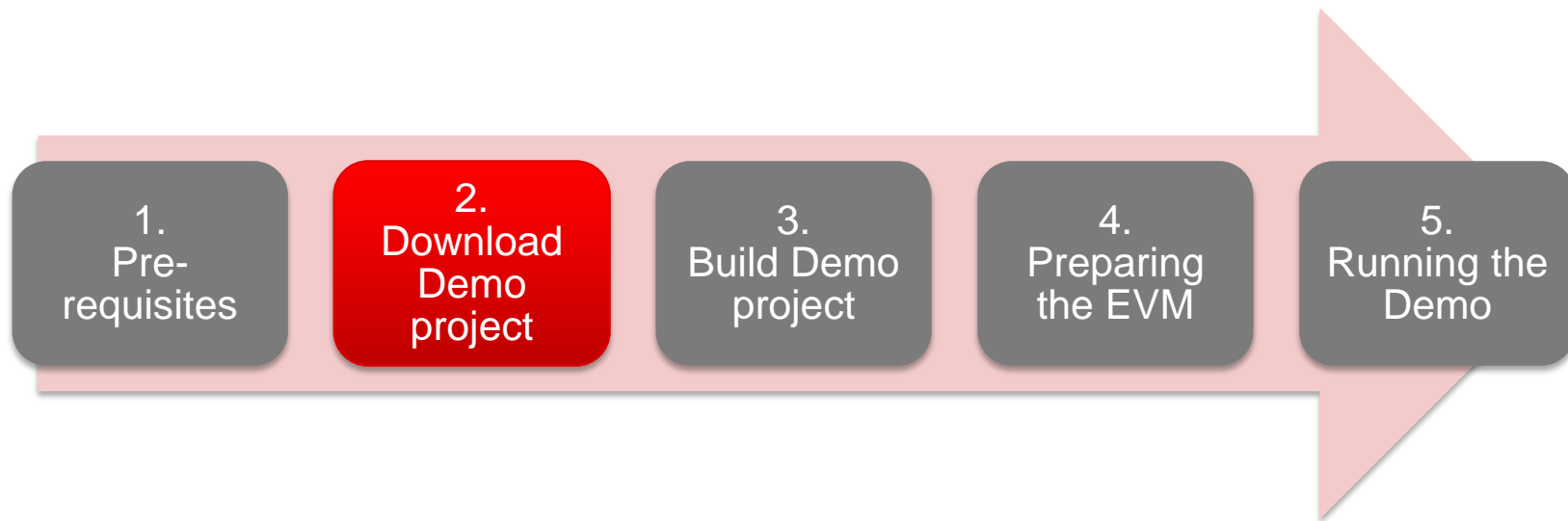
4

5

- 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 required version of 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.

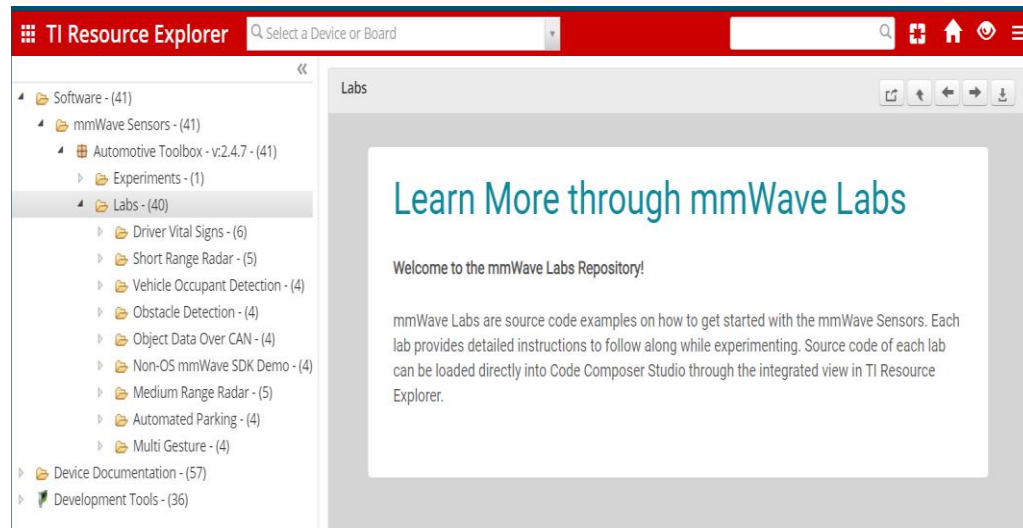
Tool	Version	Download Link
mmWave SDK	3.2.0.4	download link
CCS	8.0 or later	download link
TI ARM Compiler	16.9.6.LTS	Included in mmwave sdk installer
TI CGT Compiler	8.1.3	Included in mmwave sdk installer
XDC	3.50.08.24	Included in mmwave sdk installer
C64x+ DSPLIB	3.4.0.0	Included in mmwave sdk installer
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

Steps




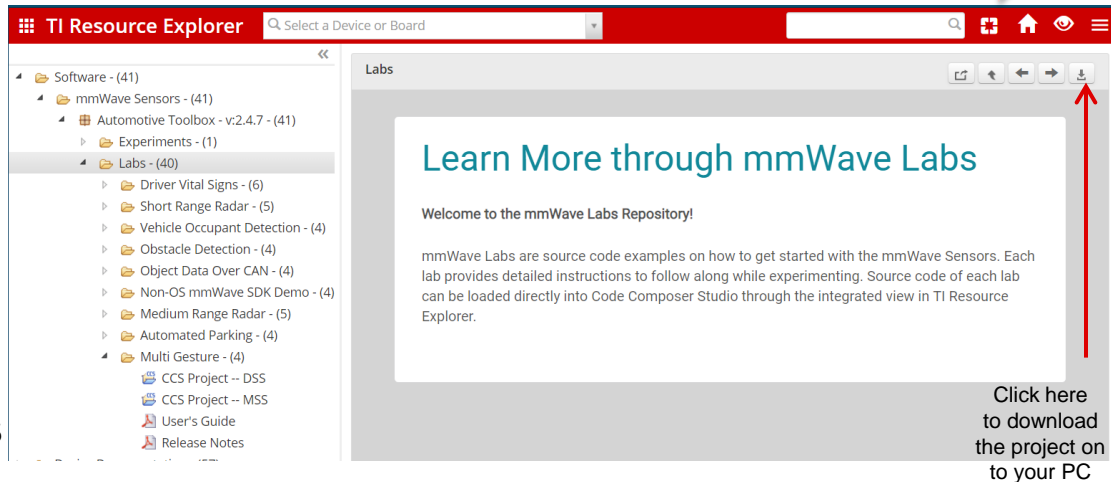
2. Download the Lab project

- The Mutli-Gesture projects are available under **mmWave Sensors** ► **Automotive Toolbox** in CCS Resource Explorer.
- To download the Multi-Gesture demo, start CCS v8.0 (or later) and select **View** ► **Resource Explorer** to open the Resource Explorer.
- In the Resource Explorer Window, select **Software** ► **mmWave Sensors** ► **Automotive toolbox** ► **Labs**.




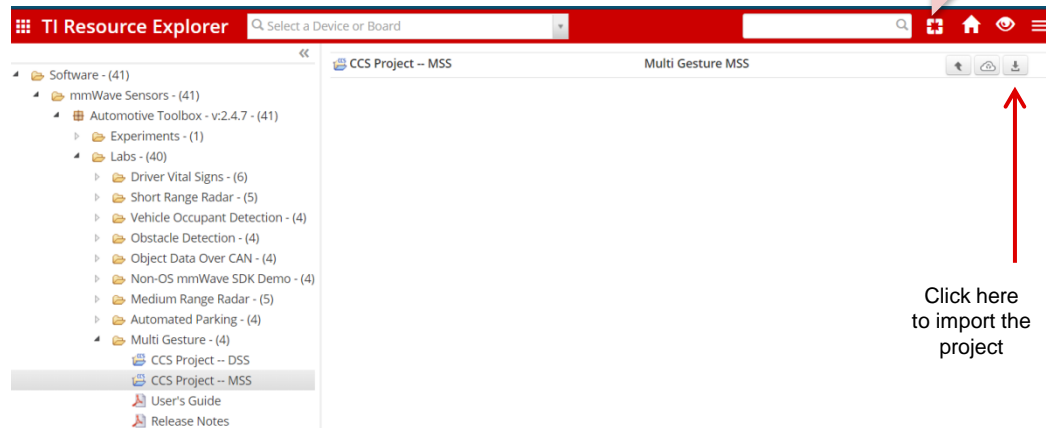
2. Download - continued

- Select the **Multi-Gesture** 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.



2. Download - continued

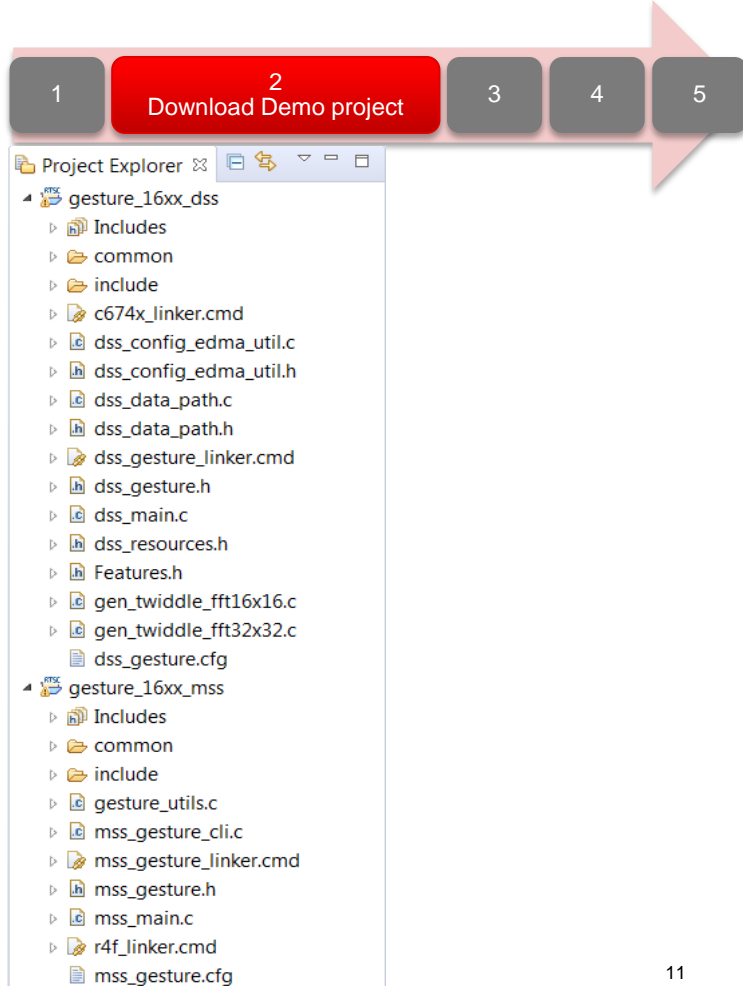
- The AWR16xx Multi-Gesture Demo consists of two CCS projects, one for the R4F core and one for the C674x DSP core
- The project will be downloaded in C:\ti\mmwave_automotive_toolbox
- Select the CCS Project MSS file in the left view
- Click on the **Import to IDE**  button which should be visible in 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



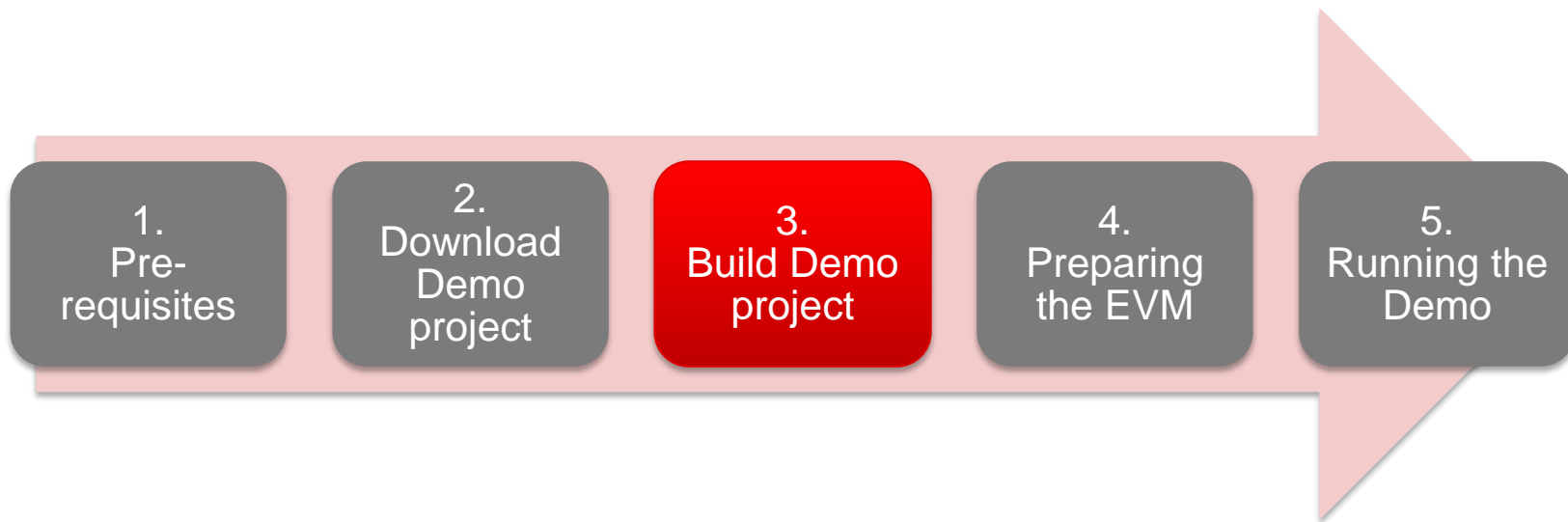
Click here
to import the
project

2. Download - continued

- 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 Multi-Gesture demo and imported it in CCS.
- We are ready to move on to the next step i.e. Building the projects.

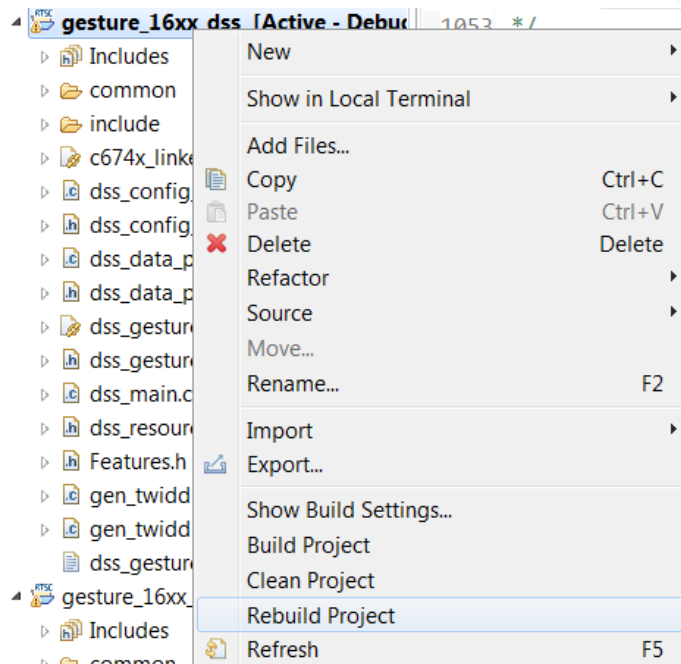
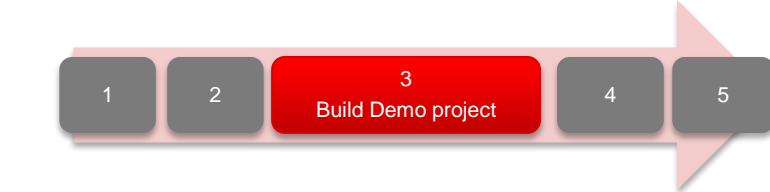


Steps



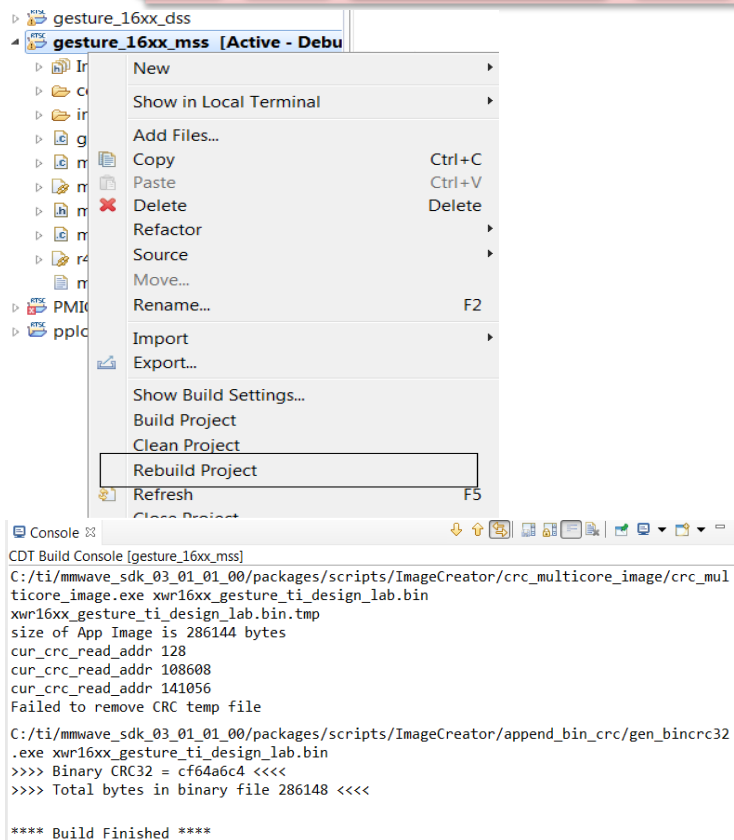
3. Build the Lab

- With the `Gesture_16xx_dss` 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_gesture_ti_design_dss.xe674`
 - `xwr16xx_gesture_ti_design_dss.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 file and debug images are provided with the demo under 'pre-build binary'.

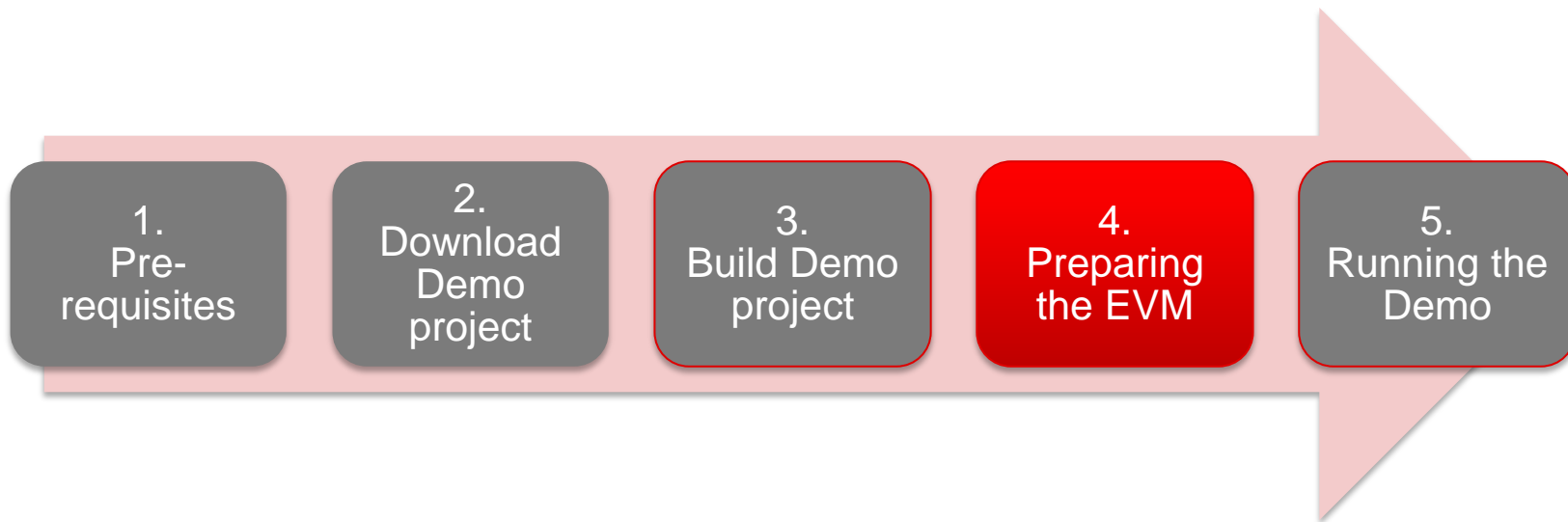


3. Build the Lab

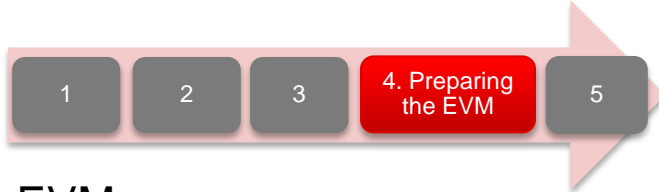
- The Gesture_16xx_dss project must be built BEFORE the Gesture_16xx_mss project is built.
- With the Gesture_16xx_mss project selected in Project Explorer, right click on the project and select **Rebuild Project**.
- On successful completion of the MSS build, you should see the output in CCS console as shown here and the following files should be produced in the project debug directory
 - xwr16xx_gesture_ti_design_mss.xer4f
 - xwr16xx_gesture_ti_design_lab.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 file and debug images are provided with the demo under 'prebuilt_binaries' directory.



Steps



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.
 - Pre-built image is provides under 'prebuilt_binaries' directory.
 - 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 (ccsdebug.bin) 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 both of methods i.e. Debug mode (CCS) and application mode.
 - To prepare the EVM for debug mode, we start with flashing the CCS debug firmware image.
 - To run the application with pre-built binary, flash the image directly and follow section 5.4

4.2 Connecting to the EVM

1

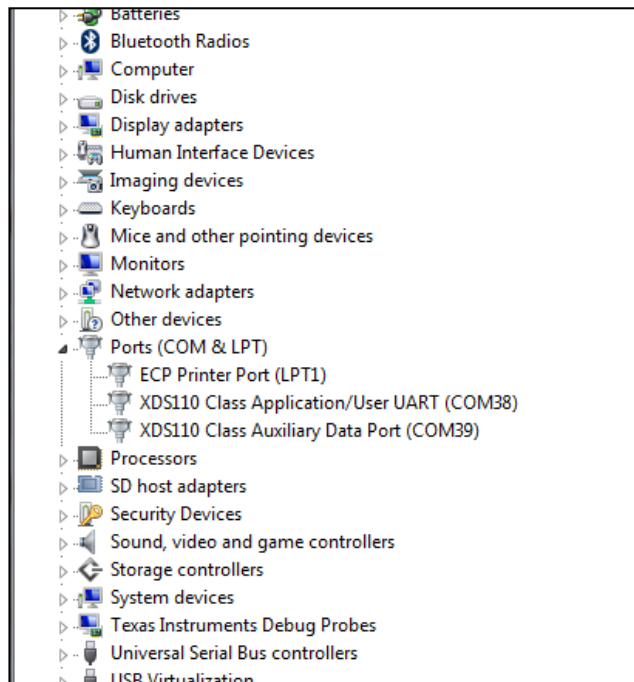
2

3

4. Preparing
the EVM

5

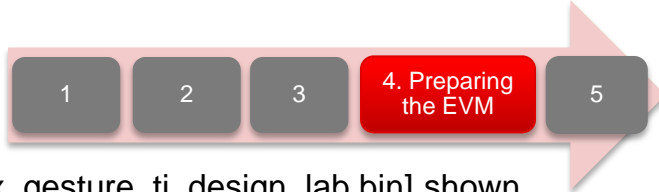
- Power on the EVM using a 5V/2.5A power supply.
- Connect the EVM to your PC and check the COM ports in Windows Device Manager
- The EVM exports two virtual COM ports as shown below:
 - XDS110 Class Application/User UART (COM_{UART}):
 - Used for passing configuration data and firmware to the EVM
 - 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.



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

- The actual port numbers on your machine may be different

4.3 Flashing Pre-built Binary



5. In the **Program** tab, browse and locate the Multi-Gesture image [xwr16xx_gesture_ti_design_lab.bin] shown below:

Flash Image(s)





<input checked="" type="checkbox"/> Meta Image 1	xwr16xx_gesture_ti_design_lab.bin Size: 279.44 KB	 Browse
<input type="checkbox"/> Meta Image 2	Leave this empty	 Browse
<input type="checkbox"/> Meta Image 3	Leave this empty	 Browse
<input type="checkbox"/> Meta Image 4	Leave this empty	 Browse

Image	Location
Meta Image 1	C:\ti\automotive_toolbox_<ver>\labs\ti\multi-gesture\prebuilt_binaries\

6. In the **Settings & Utilities** tab, fill the **COM Port** text box with the Application/User UART COM port number

▼ 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)

4.3 Flashing CCS debug firmware

1

2

3

4. Preparing the EVM

5

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

Flash Image(s)

☒ Meta Image 1 Size: 232.75 KB

☐ Meta Image 2

☐ Meta Image 3

☐ Meta Image 4

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_{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)

4.3 Flashing CCS debug firmware

1

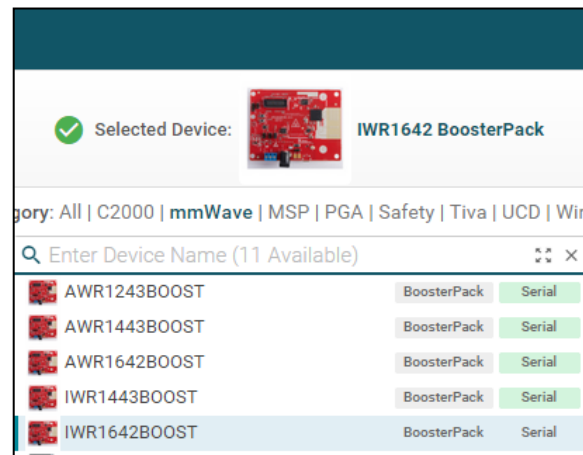
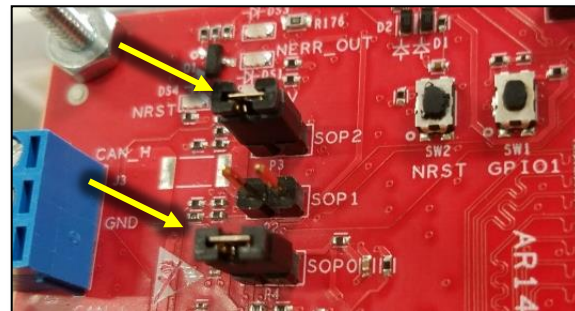
2

3

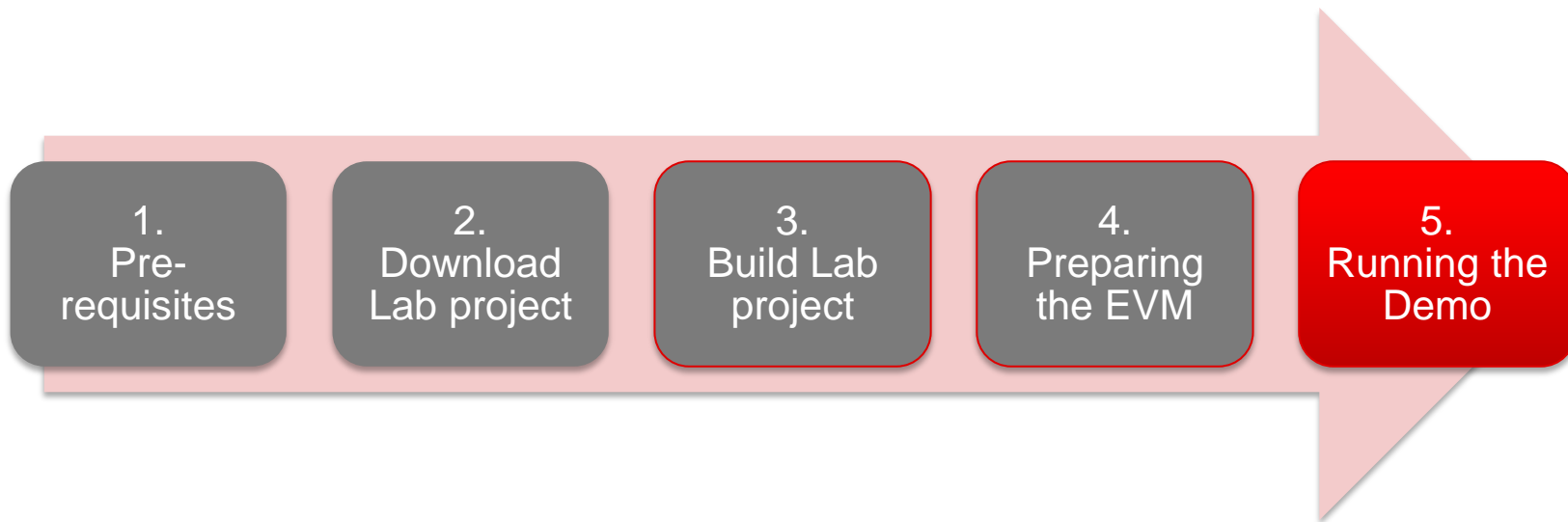
4. Preparing the EVM

5

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



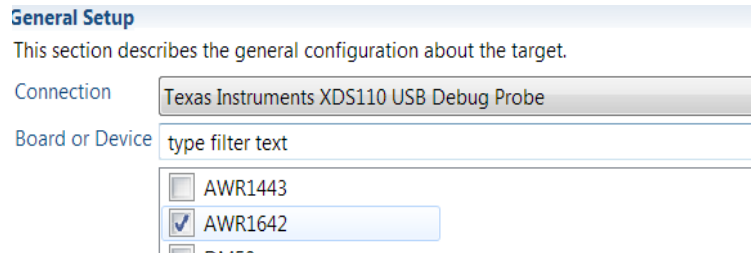
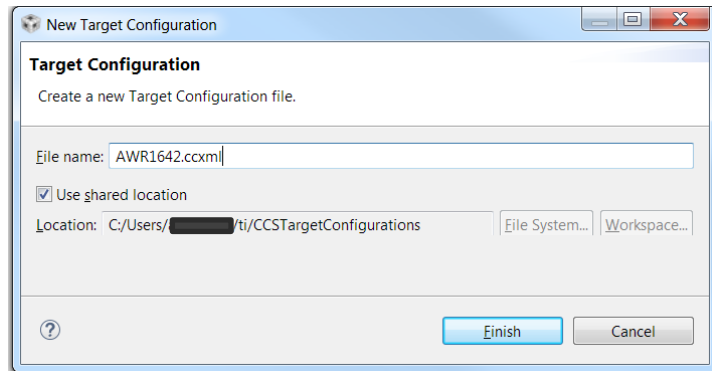
Steps





5.1 Connecting EVM to CCS

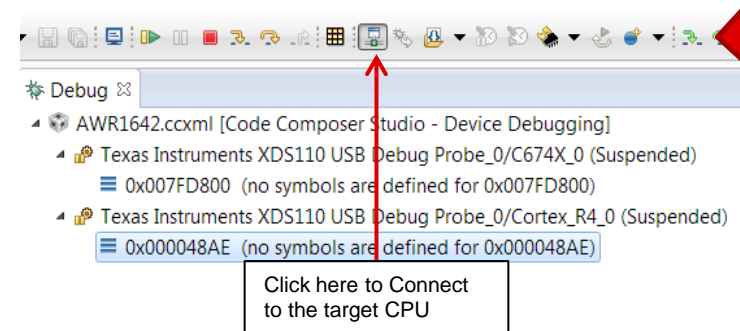
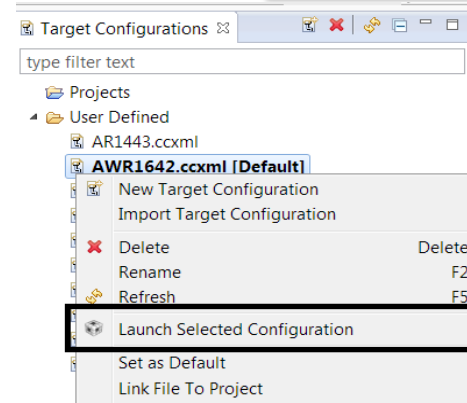


- It is assumed that you were able to download and build the Lab in CCS (completed steps 1, 2 and 3)
- To connect the Radar EVM to CCS, we need to create a target configuration
 - Go to File ► New ► New Target Configuration File
 - Name the target configuration accordingly and check the “Use shared location” 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.



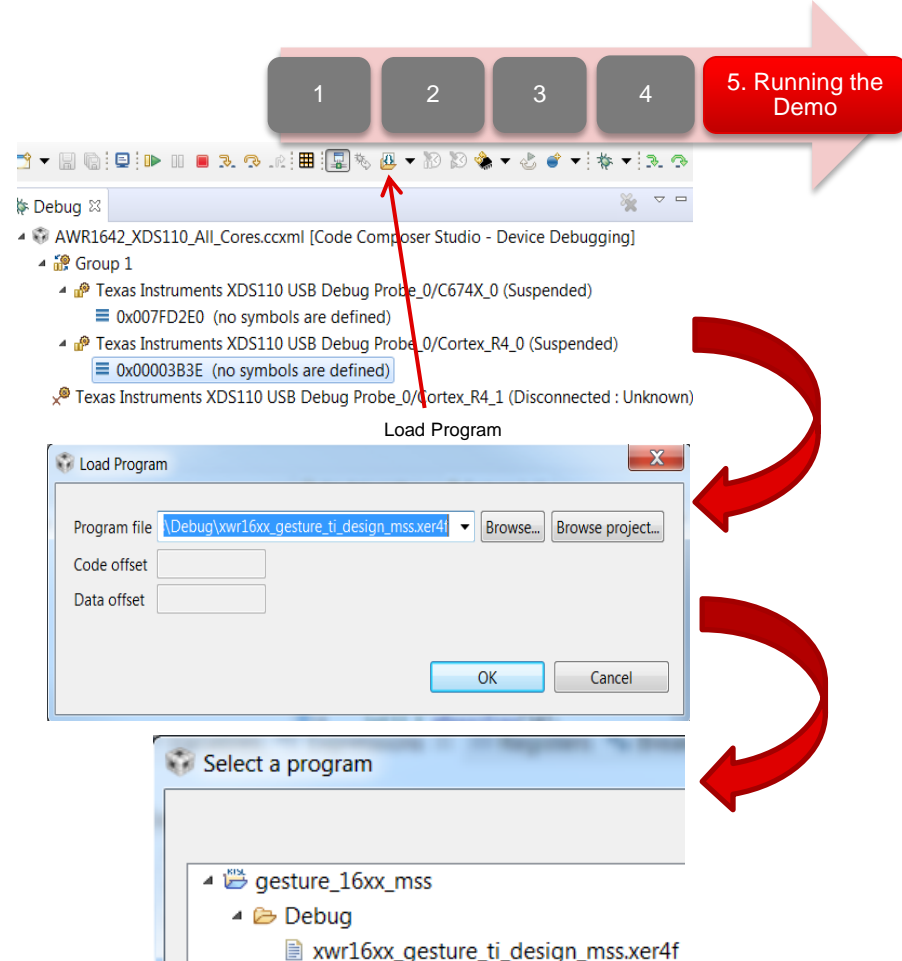
5.1 Connecting - continued

- 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_0 and press the **Connect Target** button 




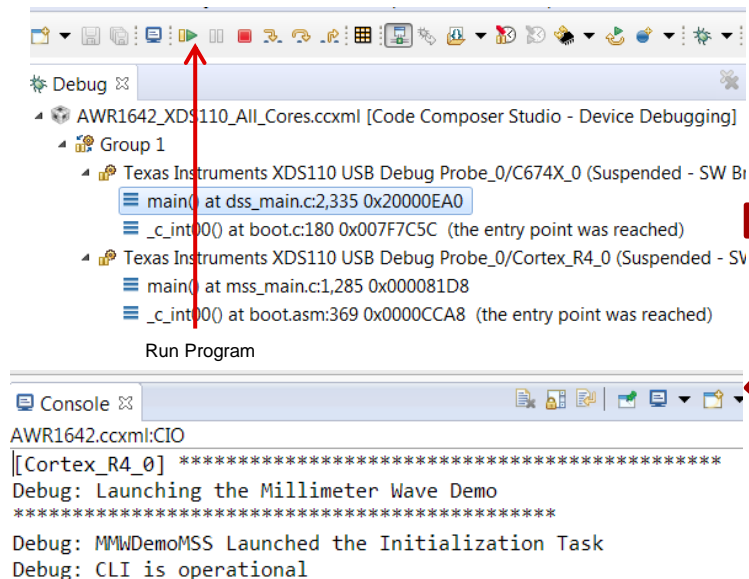
5.2 Loading the binary

- 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 gesture_16xx_dss project as shown, and press OK.
- Press OK again in the **Load Program** dialog.
- Follow same steps for R4F_0 (MSS) core with .xer4f debug image file.

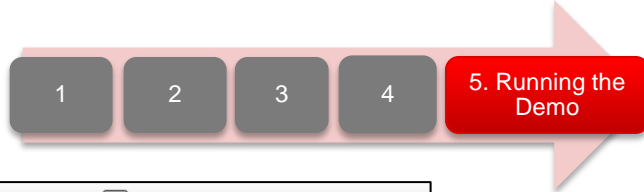



5.3 Running the binary

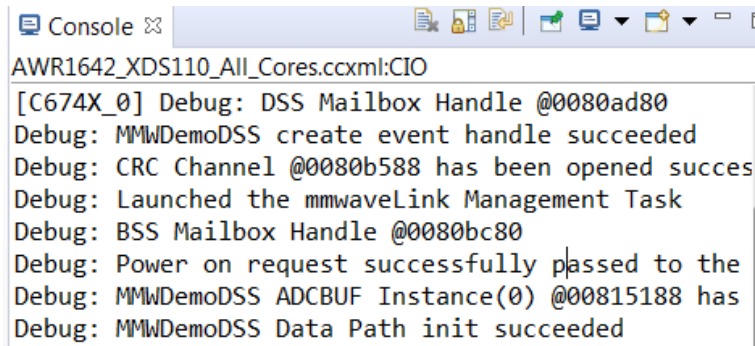
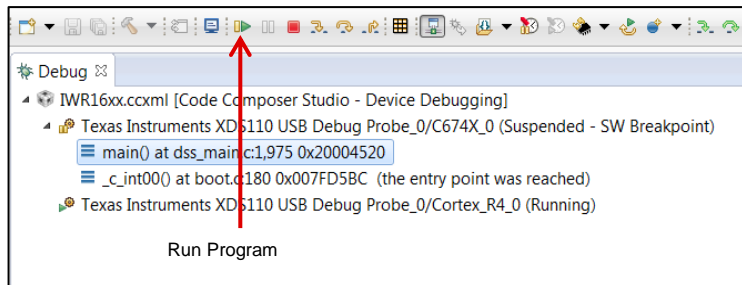
- With both executables loaded, select `mss_main.c`, as shown, and press the Run/Resume button 
- The program should start executing and generate console output as shown.



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
- In the default source code all the configuration happens automatically, user doesn't need to send any command over UART.



5.4 Running the Lab

1

2

3

4

5. Running the Demo

Here are the default application properties which user can change if needed

- CLI option is bypassed i.e. at the bootup application will configure the device automatically without waiting for any CLI commands.

- This option can be reverted with a change in 'common\cli.c' file

```
#define USE_HARD_CODED_CONFIG 0
```

- Application will send the Gesture decision in the string format (background, left, right etc.)

- This can be changed to send all the feature vectors which further can be plot using 'gui\Gesture_Feature_Plot.exe' tool. Follow the readme.txt available at the same path.

```
#define STRING_OUTPUT_ONLY 0
```

- By default power saving mode is on in the application, i.e. application will go to idle-mode if it doesn't find any movement around the sensor and switch to active-mode at any movement where it can detect a valid gesture.

- Power saving mode can be disabled with a change in dss_main.c file

```
#define DISABLE_POWER_SAVING_MODE 1
```

- By default application sends all the gesture result (feature vectors) over UART (auxiliary COM port @921600) but if user needs to send same data over CAN then

- Define '*TRANSMIT_OVER_CAN*' in MSS CCS project and recompile. This macro definition is applicable for AWR1642BOOST CAN interface.
- If user has designed a board based TIDA-02001 design then along with above definition, define '*CAN_OVER_UART_LINE*' also in MSS CCS project (Properties->Build->ARM Compiler->Predefined Symbols) and rebuild.

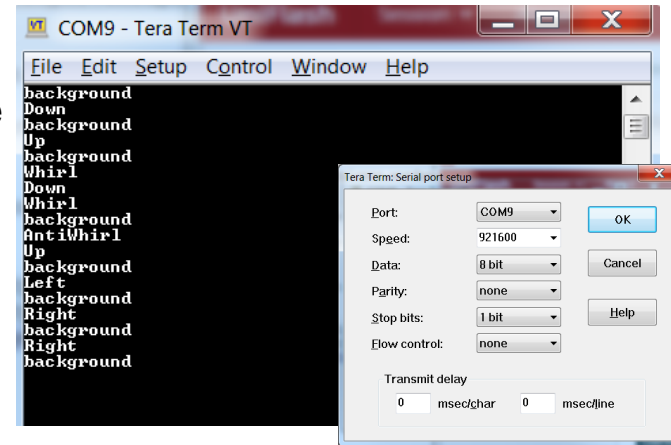
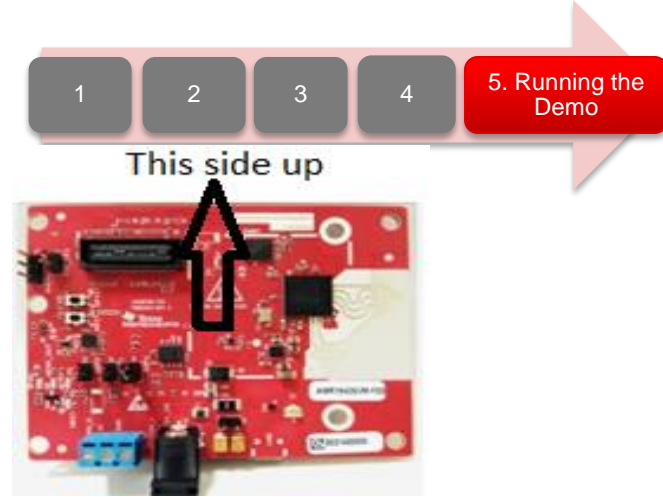
5.4 Running the Lab

– Placement of AWR1642ODS EVM

- Gesture demo application is written for a fixed pattern of antenna (virtual).
- So to get the correct result, EVM must need to be placed as shown in the picture.

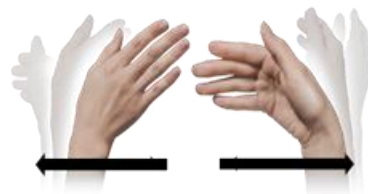
– Get the Gesture

- User needs to connect a serial terminal to one of the UART COM port (Auxiliary Data Port) at 921600 baud rate to get the Gesture result printed on the screen.
- Here we demonstrate this demo with TeraTerm.
- This demo supports total six gesture so you will see a string for each type of gestures.
 - Background : if no valid gesture.



5.5 How to do the Gesture

- Check the EVM alignment
- Swipe your hand/palm in front of EVM within a feet range.
- Speed of movement should not be too fast or too slow.
- Below is the reference snapshot for all six gestures which this demo supports.



1. Right2Left
Swipe

2. Left2Right
Swipe



3. Up2Down
Swipe



4. Down2Up
Swipe



5. Clockwise
Finger Rotate



6. Anti-Clockwise
Finger Rotate

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
 - AWR1642ODS <http://www.ti.com/tool/AWR1642BOOST-ODS>
 - TIDA-02001 <http://www.ti.com/tool/TIDA-020011>
 - 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/



© Copyright 2017 Texas Instruments Incorporated. All rights reserved.

This material is provided strictly “as-is,” for informational purposes only, and without any warranty.
Use of this material is subject to TI’s **Terms of Use**, viewable at TI.com