

mmWave Studio Cascade

Contents

1.	Introduction.....	3
2.	Hardware Pre-requisites	3
3.	Software Pre-requisites.....	3
4.	Setting Up TDA Capture Card	4
4.1	Hardware Setup	4
4.2	Flashing the TDA Firmware.....	5
5.	Booting the EVM.....	6
6.	Formatting the SSD (required for the first-time).....	7
7.	Device Configuration (mmWaveStudio).....	10
7.1	Cascade selection and setup.....	10
7.2	Running LUA Script.....	12
7.3	Using GUI.....	15
7.3.1	TDA Configuration	15
7.3.2	Device Configuration	16
7.3.3	Capturing Data	17
8.	Transferring Files and Post-Processing	22
8.1	Transferring the Data	22
8.1.1	Using mmWaveStudio	22
8.1.2	Using WinSCP	24
8.2	Post Processing the ADC Data.....	26
8.2.1	Default processing (done in a same session as configuration and capture).....	26
8.2.2	Standalone processing (done in a different session from configuration and capture)	27
8.3	Cascade Use Cases – Configuration and Post-processing	30
8.3.1	Calibration.....	30
8.3.2	TDM MIMO	30
8.3.3	TX Beamforming.....	33
9.	Deleting data on SSD.....	35

10.	Miscellaneous Features.....	36
10.1	Read/Write GPIO in AWR.....	36
10.2	Read Width and Height from TDA	36
10.3	Display CPU Statistics from TDA	37
10.4	Running apps.out application manually from teraterm.....	37
11.	Configuring IP Address	41
11.1	Static IP Address (default)	41
11.2	DHCP	42
12.	Known Issues and Limitations.....	43
13.	Troubleshooting.....	44

1. Introduction

The mmWaveStudio GUI is designed to characterize and evaluate the TI Radar devices. The mmWave device is configured and controlled from the mmWaveStudio by sending commands to the device over SPI. ADC data is captured using DCA1000 EVM or the TSW1400 EVM board for the single chip system and TDA2XX for the cascade system. The data is processed in MATLAB and the results are displayed in the GUI.

NOTE: This document focuses on the usage of mmWaveStudio for cascade chip setup (TDA2XX). For Single Chip EVMs refer to the *mmwave_studio_user_guide.pdf*

mmWaveStudio GUI utilizes C DLLs and a set of API's to communicate from the GUI to the Cascaded device through TDA2XX host over Wired LAN (Ethernet). The TDA2XX host interfaces with each of the device in the cascade system over different SPI lines.

Key features of the mmWaveStudio GUI are:

- Firmware Download over SPI interface to all devices (master and slave devices)
- Configuring all devices using the Radar API commands
- Configuration of the TDA2XX Host device to capture raw ADC data on SSD over CSI interface
- Basic Post-processing of the ADC data and visualization of the processed data

2. Hardware Pre-requisites

- RF Cascade EVM (Revision C) - <http://www.ti.com/tool/MMWCAS-RF-EVM>
- TDA2 Cascade Radar Host Board (Revision E4 or later) - <http://www.ti.com/tool/MMWCAS-DSP-EVM>
- PCIe NVME M.2 SSD
- Micro-SD card (> 2GB)
- Card Reader
- 12V 5A Power supply
- Mini-USB Cable
- Ethernet Cable
- Host running Windows (7 or 10)

3. Software Pre-requisites

For mmWaveStudio installation refer to the Section 2.1 of mmWave_studio_user_guide.pdf. Specifically for the cascade system the following are required

- TDA2XX Firmware – *mmWaveStudio\PlatformBinaries\TDA2Firmware*

- Meta image from mmWave DFP 1.2.5.1 – <http://www.ti.com/tool/MMWAVE-DFP>
- MmwaveStudio 2.1.0.0
- Serial term application (Tera Term, Putty etc.)
- WinSCP (optional)
- Balena etcher (for flashing the TDA firmware) – Version 1.5.39 - <https://github.com/balena-io/etcher/releases/tag/v1.5.39>
Download *balenaEtcher-Portable-1.5.39-x64.exe* from the above link.

4. Setting Up TDA Capture Card

The kit comes with a user-replaceable SSD. The following steps show how to mount an SSD to the board.

Some of these steps might already be done on the SSD that is included in the kit and are listed here for reference when setting up a new SSD.

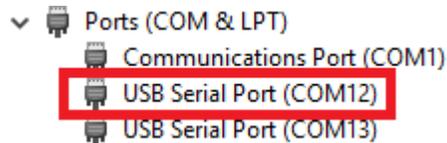
An SD card factory-flashed with a version of the TDA2XX firmware is also included in the kit. It is recommended to use an updated version from Studio on a different card to ensure that you always have the latest released version.

4.1 Hardware Setup

- Mount the SSD to the TDA2XX Host board via the J14 M.2 connector. The SSD slides into the connector at an angle. Then screw it down lightly to the board -Do not screw it down tightly

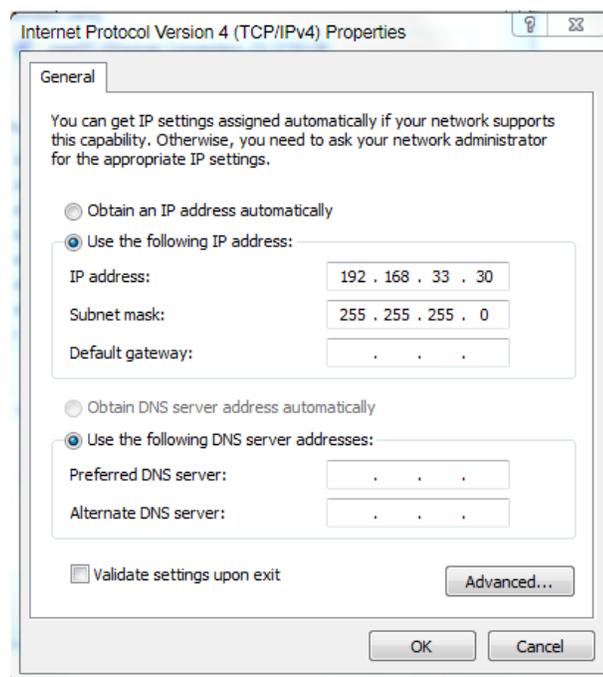


- Connect the RF EVM to the TDA2XX Host Board using the board-to-board connectors (J1 and J18 on TDA2XX). Use the mechanical spacers and screws to connect the two boards mechanically
- Connect the mini-USB cable to the J16 connector on the TDA2 Host Board. Connect the cable to the host PC. This connector exposes the UART terminal on the host PC.



- Connect the Ethernet cable to the Ethernet connector on the TDA2XX Host Board. Connect the other end to the Ethernet port of your PC. TDA2XX Host Board is configured by default to acquire a static IP address (192.168.33.180). The board can be configured to acquire the IP address over DHCP and be connected to a LAN. Refer to [Section 11](#) for instruction on how to configure the board to use DHCP or to a different static IP.
- The PC also has to be configured to a static IP address (in the case if TDA2XX Host Board is configured to static IP). In the case where DHCP is used for TDA2XX, make sure the PC and the TDA2XX are in the same network.
Set the PC IP address to *192.168.33.30*
Set Sub Net Mask as *255.255.255.0*

NOTE: Make sure that the PC and the TDA2XX Host board are in the same network connected through Ethernet.

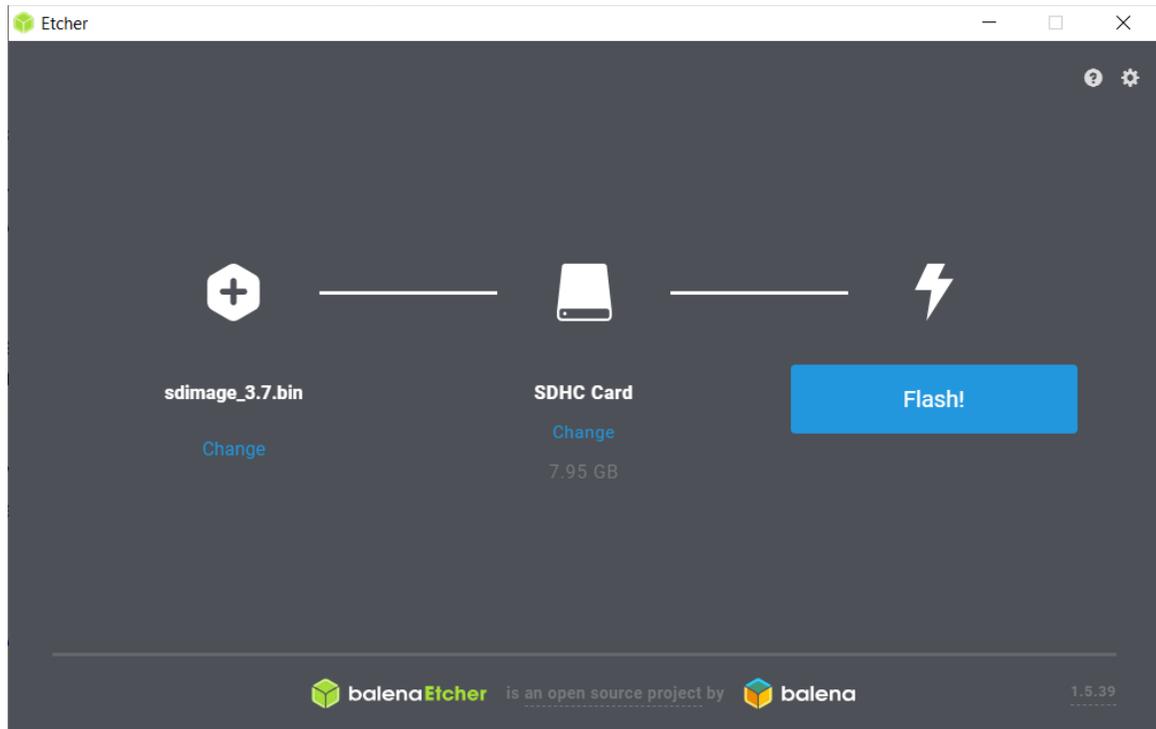


- Connect a 12V 5A power supply to the EVM using the J10 barrel jack connector or the J11 screw-terminal connector

4.2 Flashing the TDA Firmware

- Put the SD card in the Card reader and insert it in card reader slot of your PC
- Run the Balena Etcher application

- Select the TDA Firmware image
- Select the drive (corresponding to SD card) and Flash.



- Wait for the Flashing to complete (this step takes time)

5. Booting the EVM

- Insert this micro-SD card into the J15 card cage on the TDA2XX Host board.
- Power up the 12V supply
- Connect the Mini-USB to a PC running a serial terminal application. Two serial ports should be enumerated. If not, install the latest FTDI drivers on your PC and reconnect the cable. (FTDI drivers are present as a part of the mmWaveStudio package)
- Use Serial terminal application to open the first of the two serial ports enumerate with following setting:
 - Baud rate : 115200
 - Data : 8 bit
 - Parity : None
 - Stop : 1 bit
 - Flow Control : None

NOTE: In order to enumerate the settings in the case of Teraterm, click on Setup->Serial Port

- The running logs from device should be visible. If not, switch S1 on TDA2XX board should be pressed to restart the device.
- Enter the username as 'root' in terminal when prompt.
- Obtain the IP address of the TDA2XX Host board using "ifconfig" command. By default, it should be "192.168.33.180". If DHCP is used instead of static IP (default mode), this might be different.

```

eth0      Link encap:Ethernet  HWaddr F0:B5:D1:12:2C:B8
          inet addr:192.168.33.180  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::f2d5:d1ff:fe12:2cb8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:116774 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85900 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8791743 (8.3 MiB)  TX bytes:5620318 (5.3 MiB)
          Interrupt:251

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:332 errors:0 dropped:0 overruns:0 frame:0
          TX packets:332 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:27628 (26.9 KiB)  TX bytes:27628 (26.9 KiB)

```

6. Formatting the SSD (required for the first-time)

Formatting is required only during the first boot after attaching the SSD. For the subsequent iteration, this step can be skipped. The following steps explain the procedure to do the same.

- List all the devices present through the following command

```
ls /dev/nvme*
```

NOTE: Ensure that there is an entry named 'nvme0n1' from the list of devices displayed. This indicates that an SSD is present and is detected by the TDA2 Host board.

If 'nvme0n1' doesn't show up in the list, make sure the SSD is fitted properly as mentioned in the [Section 4.1](#) and try rebooting the TDA2 Host board.

- Use the following command to start the formatting process. This will ask the user to enter relevant inputs regarding the formatting.

```
fdisk /dev/nvme0n1
```

- Enter 'm' to understand the all the options (as shown below)

```

root@dra7xx-evm:~# fdisk /dev/nvme0n1

The number of cylinders for this disk is set to 488386.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
 1) software that runs at boot time (e.g., old versions of LILO)
 2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): m
Command Action
a toggle a bootable flag
b edit bsd disklabel
c toggle the dos compatibility flag
d delete a partition
l list known partition types
n add a new partition
o create a new empty DOS partition table
p print the partition table
q quit without saving changes
s create a new empty Sun disklabel
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)

```

- Once the help option for the command appears as in the screenshot above, provide the following input in the same sequence (refer to the below screenshot).

```

n
p
1
<press enter for default>
<press enter for default>
p
w
Command (m for help): n
Partition type
  p primary partition (1-4)
  e extended
p
Partition number (1-4): 1
First sector (32-1000215215, default 32):
Using default value 32
Last sector or +size(<K,M,G,T>) (32-1000215215, default 1000215215):
Using default value 1000215215

Command (m for help): p
Disk /dev/nvme0n1: 477 GB, 512110190592 bytes, 1000215216 sectors
488386 cylinders, 64 heads, 32 sectors/track
Units: sectors of 1 * 512 = 512 bytes

Device            Boot StartCHS      EndCHS           StartLBA        EndLBA        Sectors  Size Id Type
/dev/nvme0n1p1    0,1,1          1023,63,32      32 1000215215 1000215184 476G 83 Linux

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table
[ 84.877014] nvme0n1: p1
root@dra7xx-evm:~# [ 84.907031] nvme0n1: p1
[ 85.293810] EXT4-fs (nvme0n1p1): recovery complete
[ 85.316862] EXT4-fs (nvme0n1p1): mounted filesystem with ordered data mode. Opts: (null)
[ 86.001963] EXT4-fs (nvme0n1p1): mounted filesystem with ordered data mode. Opts: (null)

```

- Use the following command to “unmount” the mounted file system, informing the system to safely detach it.

```
umount /dev/nvme0n1p1
```

- Use the following command to create the filesystem. It will ask for confirmation, enter “y” to proceed (refer to the below screenshot)

```
mkfs.ext4 /dev/nvme0n1p1
```

```
root@dora7xx-evm:~# umount /dev/nvme0n1p1
root@dora7xx-evm:~# mkfs.ext4 /dev/nvme0n1p1
mke2fs 1.43-WIP (18-May-2015)
/dev/nvme0n1p1 contains a ext4 file system labelled 'ssd'
      last mounted on /mnt/ssd on Mon May 13 19:54:55 2019
Proceed anyway? (y,n) y
Discarding device blocks: done
Creating filesystem with 125026898 4k blocks and 31260672 inodes
Filesystem UUID: 907ad9ee-b5d1-4b7e-b6ef-d8a6493a3187
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424, 20480000, 23887872, 71663616, 78675968,
    102400000
Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

- Reboot the TDA2 Host board by pressing the S1 switch.
- Wait for the login console to come and login as “root”.
- Run the following command to confirm if the formatting operations is successful or not. It should show an entry named “/dev/nvme0n1p1” (as shown below) in case of successful formatting.

```
ls /dev/nvme*
```

```
root@dora7xx-evm:~# ls /dev/nvme*
/dev/nvme0  /dev/nvme0n1  /dev/nvme0n1p1
```

7. Device Configuration (mmWaveStudio)

7.1 Cascade selection and setup

- On Startup of mmWaveStudio, the default Radar System selected is the Single Chip as shown below.



- In order to enable cascade chip configuration, choose “Cascade” from the drop down option. The cascade related configuration will be displayed as below. By default, the **4 chip** cascade configuration is chosen. With the current hardware setup, RadarDevice1 maps to the Master chip and other device selections to the Slave chips.



Radar System

Cascade

DCA1000

TSW1400

TDA

Set Up TDAxx
Capture Card

Cascade

Four Device

Device Selection

RadarDevice1

RadarDevice2

RadarDevice3

RadarDevice4

- Configuration and Capture of the devices can be done in two ways
 1. Using LUA scripts ([Section 7.2](#))
 2. Using GUI ([Section 7.3](#))

Both the methods have been explained in the upcoming sections.

7.2 Running LUA Script

Some sample LUA scripts for cascade systems are already provided in '*mmWaveStudio\Scripts\Cascade*' directory for ease of use. Please refer to the following use-cases to get started.

For all the use-cases mentioned below, update the following in the configuration LUA scripts before running it.

1. *Meta image path* – Use the DFP 1.2.5.1 official release Meta image.
<http://www.ti.com/tool/MMWAVE-DFP>
2. *TDA2XX Host IP Address* – Update the IP address of the TDA2XX Host obtained through the serial terminal. ([Section 5](#))

The capture script "*Cascade_Capture.lua*" is common across all use-cases. Update the following in the above script if required before running.

1. Number of files to pre-allocate (*n_files_allocation*)
2. Data Packing format (*data_packaging*)
3. Filename of capture directory (*capture_directory*)
4. Number of frames to capture (*num_frames_to_capture*)
5. Type of framing (*framing_type*)

Detailed instructions about the parameters are provided in the script for reference.

- **Use case 1 : Test Source**

Scripts used:

1. *Cascade_Configuration_TestSource.lua* for configuration of devices
2. *Cascade_Capture.lua* for capturing data

Once the TDA2XX Host IP address and Meta image path are updated appropriately, run the "*Cascade_Configuration_TestSource.lua*" script for configuring all devices till frame configuration.

Once the execution of the configuration script is completed, run the "*Cascade_Capture.lua*" script for capturing data.

NOTE: Update the capture directory name ("*capture_directory*") in the "*Cascade_Capture.lua*" script before running it. If the "*Cascade_Capture.lua*" script is called multiple times without changing the directory name, then all captured files will be in the same directory. Hence, it is strongly recommended to change the directory name between captures.

User can change the object position/velocity for test source in the "*Cascade_Configuration_TestSource.lua*" script by tweaking some of the parameters in the

ar1.SetTestSource_mult API. Use “help ar1.SetTestSource_mult” in the LUA shell for more information on the parameters.

On completion of the capture script, press “Transfer Files” button in Sensor Config tab to transfer the captured data. For basic post processing of the captured data, press “Post Proc” button.

Refer [Section 8](#) on more information regarding Transfer of Captured files and Post processing.

NOTE: Once the test source script is ran as a trial, make sure to disable the “test source enable” checkbox present in the Sensor Configuration Tab before running any other script.

- **Use case 2 : MIMO Configuration**

Scripts used:

1. *Cascade_Configuration_MIMO.lua* for configuration of devices
2. *Cascade_Capture.lua* for capturing data

Once the TDA2XX Host IP address and Meta image path are updated appropriately, run the “*Cascade_Configuration_MIMO.lua*” script for configuring all devices till frame configuration.

Once the execution of the configuration script is completed, run the “*Cascade_Capture.lua*” script for capturing data.

NOTE: Update the capture directory name (“*capture_directory*”) in the “*Cascade_Capture.lua*” script before running it. If the “*Cascade_Capture.lua*” script is called multiple times without changing the directory name, then all captured files will be in the same directory. Hence, it is strongly recommended to change the directory name between captures.

On completion of the capture script, press “Transfer Files” button in Sensor Config tab to transfer the captured data. For basic post processing of the captured data, press “Post Proc” button.

Refer [Section 8](#) on more information regarding Transfer of Captured files and Post processing.

- **Use case 3 : TX Beam Forming Configuration**

NOTE: TX Beam Forming is done by programming the TX phase shifters to specific angles for each TX. The specific phase shifter setting to use for each TX depends on the phase shifter calibrations for each TX. Currently the code to calculate the angle taking into account the phase shifter calibrations is available only in the Matlab scripts used in section 8.3.3.

The LUA scripts described in this section are for reference only, and were generated for a single configuration with a specific phase-shifter calibration configuration.

Scripts used:

1. *Cascade_Configuration_Basic.lua* for configuration of devices, except for Profile, Chirp, Phase Shifter and Frame configuration
2. *Cascade_Configuration_TXBF.lua* for configuring the devices with a TX beam forming configuration
3. *Cascade_Capture.lua* for capturing data

Once the TDA2XX Host IP address and Meta image path are updated appropriately, run the “*Cascade_Configuration_Basic.lua*” script to do the first part of the configuration. This script does not configure the Profiles, Chirps, Frames, or Phase shifters. Then run the “*Cascade_Configuration_TXBF.lua*” script to complete the remaining configuration of all devices till frame configuration.

Once the execution of the configuration script is completed, run the “*Cascade_Capture.lua*” script for capturing data.

NOTE: Update the capture directory name (“*capture_directory*”) in the “*Cascade_Capture.lua*” script before running it. If the “*Cascade_Capture.lua*” script is called multiple times without changing the directory name, then all captured files will be in the same directory. Hence, it is strongly recommended to change the directory name between captures.

On completion of the capture script, press “Transfer Files” button in Sensor Config tab to transfer the captured data. For basic post processing of the captured data, press “Post Proc” button.

Refer [Section 8](#) on more information regarding Transfer of Captured files and Post processing.

- **Use case 4 : Monitoring and Calibration**

Scripts used:

1. *Cascade_Configuration_MIMO.lua* for configuration of devices
2. *Cascade_Monitoring_Example.lua* for enabling the monitors, calibrations and framing

Once the TDA2XX Host IP address and Meta image path are updated appropriately, run the *Cascade_Configuration_MIMO.lua* script for configuring all devices till frame configuration.

Once the execution of the configuration script is completed, review the following parameters in the “*Cascade_Monitoring_Example.lua*” script before running.

1. Device map (*device_map*)
2. Calibration Monitoring Time Unit (*cal_mon_unit*)
3. Number of devices (*num_devices*)
4. Calibration Periodicity Multiplication factor (*calib_mult_factor*)

Detailed instructions about the parameters are provided in the script for reference.

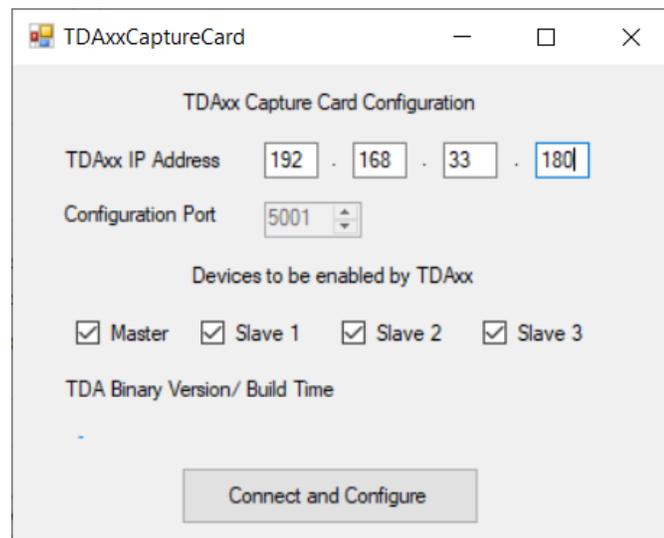
7.3 Using GUI

7.3.1 TDA Configuration

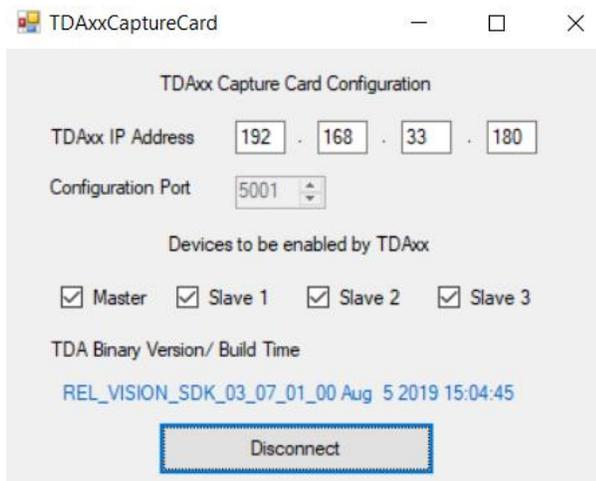
- Click on 'Setup TDAxx Capture Card' button at the left pan of GUI. The GUI pop-up will be displayed as shown below



- Enter the IP Address of the TDA as "192.168.33.180" if static IP is configured or provide TDA DHCP IP address here. Click Check-boxes to enable the device. Click on "Connect and Configure".



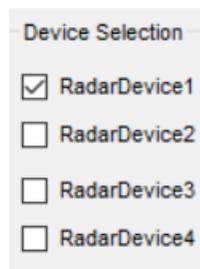
- If the connection establishment with TDA was successful, then the button changes to "Disconnect". Also, the corresponding TDA Binary version and build time would be displayed. Please refer Output log for more information.



```
[17:00:23] [RadarAPI]: ar1.SelectCaptureDevice("TDA2XX")
[17:00:23] [RadarAPI]: Status: Passed
[17:00:42] [RadarAPI]: TDAStatusCallback Event Registered
[17:00:42] [RadarAPI]: registerTDAStatusCallback Status: Passed
[17:00:42] [RadarAPI]: ar1.ConnectTDA("192.168.33.180", 5001, 15)
[17:00:42] [RadarAPI]: Device 32 : CAPTURE_RESPONSE_ACK Async event recieved() with status 0
[17:00:42] [RadarAPI]: Device 32 : CAPTURE_RESPONSE_ACK Async event recieved() with status 0
[17:00:42] [RadarAPI]: Device 32 : CAPTURE_RESPONSE_VERSION_INFO Async event recieved() with
status 0. TDA Version : REL_VISION_SDK_03_07_01_00 Aug 5 2019 15:04:45[17:00:42] [RadarAPI]:
Device 32 : CAPTURE_RESPONSE_ACK Async event recieved() with status 0
[17:00:44] [RadarAPI]: Device 1 : CAPTURE_RESPONSE_ACK Async event recieved() with status 0
[17:00:44] [RadarAPI]: ConnectTDACaptureCard Status: Passed
[17:00:44] TDA Capture Card Status : CONNECTED!
```

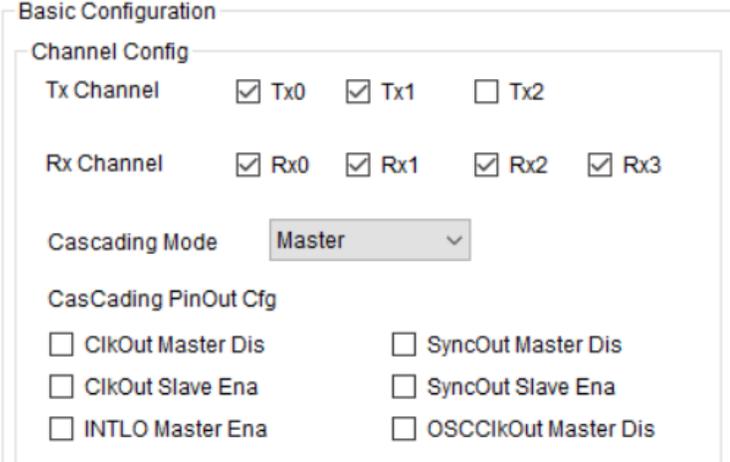
7.3.2 Device Configuration

- Before configuring any device, select the “Radar Device” checkbox corresponding to that specific device in the device Selection box present in the left. In the current hardware setup, “RadarDevice1” maps to Master device and remaining to the slave devices.



NOTE: Master provides the reference clock to all the slave devices in the current hardware setup. Hence, the master device has to be configured first till Basic Configuration (in static configuration tab) to enable the clock to the Slave devices. Only after this, slaves devices can be configured

- Go to the connection tab and select the SOP Mode as “Mode 4 (Functional-SPI)”. Functional mode is the only supported mode with the TDA2XX-AWR1243 Cascade solution.
- Click on SPI Connect and wait the connection to complete (the button will change to SPI Disconnect)
- Select the Meta Image from the BSS FW Selection and click Load. This step might take some time
- Click on RF Power-Up
- Move to Static configuration tab for Channel configuration. Select the Cascading Mode as “Master” for “RadarDevice1” and click Set. This will enable the clock to the slave devices and must be done before any configuration of slave devices.



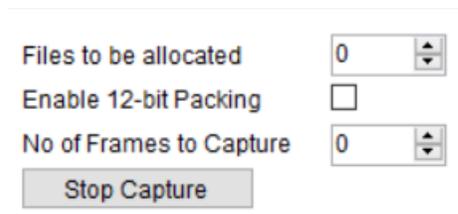
- Perform SOP mode and SPI Connect for each one of the slave devices **individually**.

NOTE: There is also an option to issue API’s together by selecting multiple devices from the checkbox. (apart from SOP mode, SPI Connect and RF Power Up)

- Perform the firmware download for all the slaves together. Perform RF Power Up for each one of the slave devices **individually**. Perform the basic configuration in Static configuration tab for all the slaves together. Also, the cascading mode for all slave devices should be selected as “Slave” during their basic configuration.
- The other configuration API’s are similar to the sequence followed in single chip apart from the frame configuration (refer to the next point for frame configuration). These API’s can be given by selecting all the devices from the checkbox.
- For Frame Configuration, select the “Hardware Trigger” for all the enabled slave devices and “Software Trigger” for Master device.

7.3.3 Capturing Data

- Once the configuration is done for all the devices, go to “SensorConfig” tab.
- Before clicking on “TDAXx ARM” button, update the following fields if required



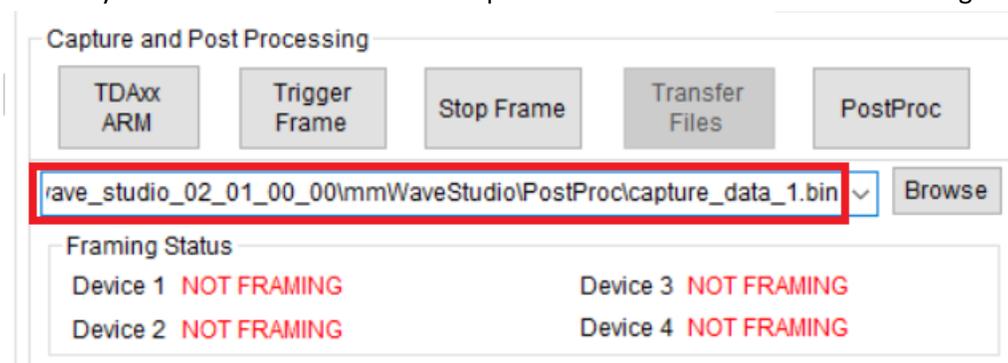
Files to be allocated: 0

Enable 12-bit Packing:

No of Frames to Capture: 0

Stop Capture

1. Number of files to be pre-allocated on the SSD. If this is non-zero, each file created (for each device) is a fixed 2047 MB even if a smaller number of frames are captured (remaining portion of the file is filled with zeroes). This might improve capture reliability while switching files. By default, no allocation is done and the file size is exactly the ADC data being captured.
2. Select whether to use 16-bit ADC data packing (native) or 12-bit ADC data packing (in a packed form). The default is no-packing (16-bit ADC data) (the box is unchecked as showed in the screenshot above).
3. The data will be captured on the SSD with separate files for each device in a directory. Directory name is the same as the name provided in GUI. Refer to the below image



Capture and Post Processing

TDAxx ARM Trigger Frame Stop Frame Transfer Files PostProc

/ave_studio_02_01_00_00\mmWaveStudio\PostProc\capture_data_1.bin Browse

Framing Status

Device 1	NOT FRAMING	Device 3	NOT FRAMING
Device 2	NOT FRAMING	Device 4	NOT FRAMING

In this case, a folder named “capture_data_1” will be created under “/mnt/ssd” directory in the TDA2XX Host file system. The captured files will be stored separately for each device under this folder.

4. Number of frames to capture is a special case where the number of frames to be captured by the TDA2XX Host is informed before-hand via this parameter. By default, the value is kept zero. (i.e TDA captures till the device is framing). This is typically useful when the devices are framing continuously and TDA is needed to selectively capture a number of frames at required intervals. In that case, a non-zero value is provided to this parameter and “TDAxx ARM” is issued whenever it is required to capture. TDA stops capturing automatically once it has captured the above mentioned frames.

NOTE: There is also an option to inform TDA2xx Host to stop capturing voluntarily through “Stop Capture” button. This should be issued (only if necessary) when TDA2xx is currently capturing the raw ADC data.

- Click on “TDAXx ARM” with any one of the device selected. This should be done only once for a capture session. By default, the framing status (refer the below screenshot) is “NOT FRAMING” for all the devices.
- Click on Trigger frame for all the enabled slave devices (one device at a time). The framing status will be updated to “WAITING FOR TRIGGER” once the frame start async event is obtained for the respective devices.
- Choose the master (device 1) and click on Trigger frame. Once the frame start async event for the master is received, the framing status is updated.

Framing Status	
Device 1	FRAMING
Device 2	FRAMING
Device 3	FRAMING
Device 4	FRAMING

- To stop the framing (in case of infinite frames), click on the stop frame for slaves first and then for the master. (one device at a time)

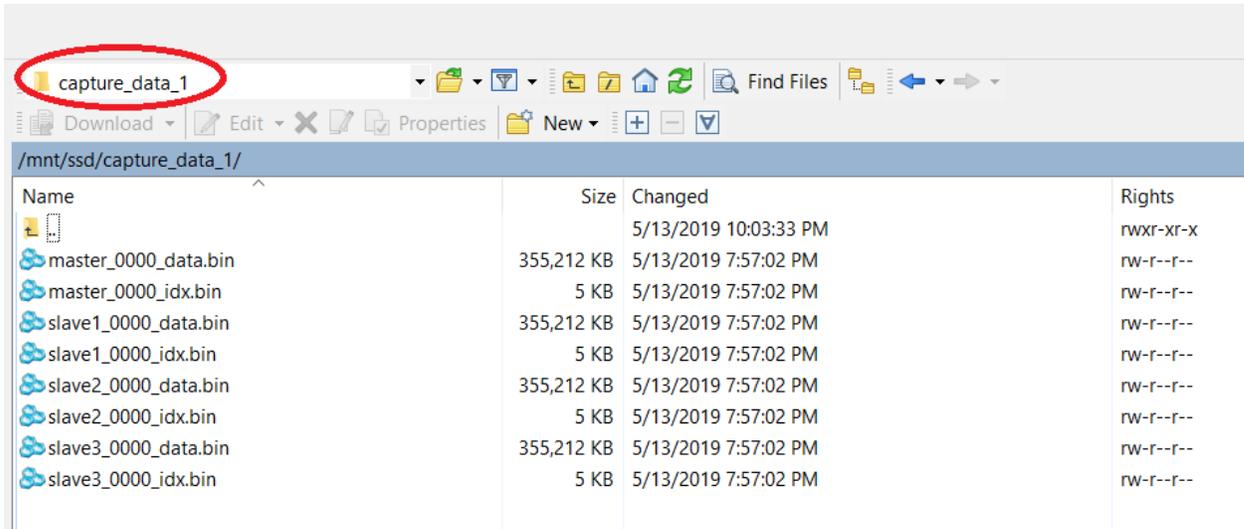
NOTE: Invoke command to all the Slave devices (one device at a time) first and then to the Master device for “Trigger Frame” and “Stop Frame”.

- Once the frame for the master is stopped, the TDA gets dis-ARMed and stops the capture. The framing status is also updated accordingly based on the frame stop async events received from the device.

NOTE: Instructions on using the WinSCP are explained in [Section 8.1.2](#).

- The below screenshot from WinSCP shows the data captured in the folder mentioned during TDAXx ARM. There are two types of files created for each device. (assuming *16-bit packing mode*)
 1. *deviceName_000x_data.bin*: This file contains the actual ADC data being captured from the respective device.
 2. *deviceName_000x_idx.bin*: This file contains the information regarding the number of frames captured, timestamp of each frame captured, packing type etc. for a corresponding *deviceName_000x_data.bin* file. This file is useful in detecting frame drops etc.

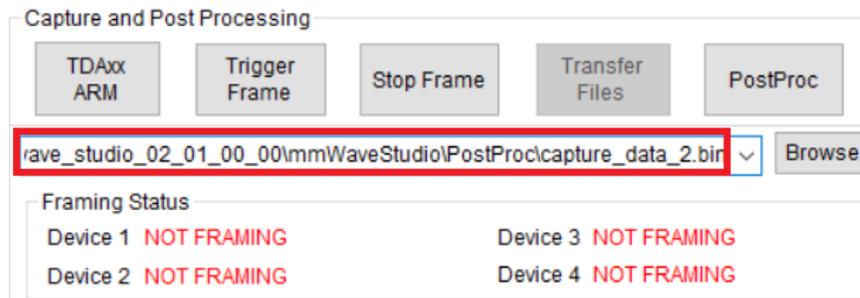
NOTE: In the case of 12-bit packing mode, the filenames are of the form *deviceName_000x_packed_data.bin* and *deviceName_000x_idx.bin*



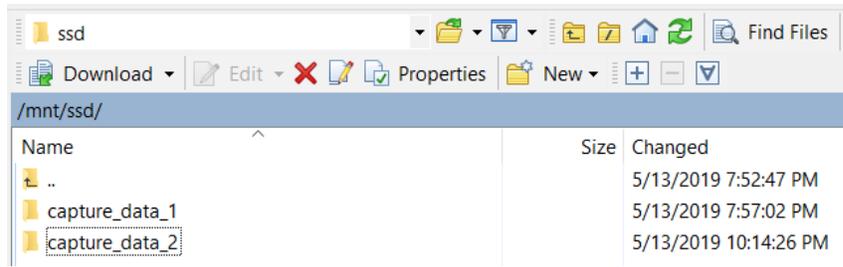
NOTE: If the data file crosses the 2047MB limit, then a new file will be created dynamically for storing the incoming data with the same name/prefix and an incremented file index.

For example in the above case, if the master’s data had crossed 2047MB, then a new file named `master_0001_data.bin` and `master_0001_idx.bin` would be created.

- If you want multiple capture sessions, do the following:
 - a. Update the required device configurations.
 - b. Update the name of the ADC file (It is highly recommended to do this as it would give a clear distinction between the capture sessions) to store the data at different directory.



- c. Follow the previous sequence of clicking on “TDAX ARM” and subsequent “Trigger Frame” for the devices.
- The new capture will create a new directory and will not impact the previous captures (as shown in the screenshot below). If the folder already exists it will create data files inside the same folder with incremented file indexes.



NOTE: If needed, the data files and the trace files created on the SSD needs to be deleted manually. This is explained in the [Section 9](#).

8. Transferring Files and Post-Processing

8.1 Transferring the Data

The data is captured on the SSD and needs to be transferred to the PC. There are 2 options

- Automatic transfer using mmWaveStudio (recommended)
- Manual transfer using standalone WinSCP application

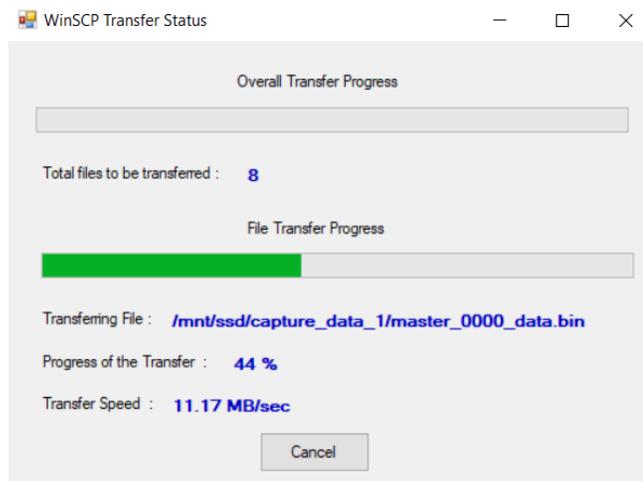
8.1.1 Using mmWaveStudio



- As shown in the [Section 7.3.3](#), the data is stored in dedicated folders mentioned during TDAxx ARM operation. In order to transfer files using mmWaveStudio, edit the filename in the highlighted box above, same as that of the directory name to transfer from the SSD.

NOTE: The path to store the captured files in the PC file system need not always be in the PostProc folder. Make sure the “filename.bin” provided as input matches with the directory name in the /mnt/ssd/ folder in the TDA2XX Host file system.

- Click on “Transfer Files” button.
- In the above case, a directory named *capture_data_1* (if found in the /mnt/ssd/ folder) would be transferred to PostProc directory. In case, there are no files present in the directory, there would be a pop-up message displaying “The no of files captured in the directory is zero! Nothing to transfer!”



NOTE: The speed at which the files are being transferred varies from network to network.

- The progress of the WinSCP transfer is shown through a pop-up which indicates the following
 1. Name of the file that is being transferred.
 2. Progress of the transfer (for each file)
 3. Number of files to be transferred (in overall) from that directory.
 4. Speed at which the transfer takes place (in MB/sec)

NOTE: The transfer of files using WinSCP (in mmWaveStudio) can also be aborted during the file transfer by pressing the “Cancel” button. However, it is not recommended to use this, unless it is really required.

On successful abort, a message appears in the Output log window displaying “*WinSCP session Aborted!*”

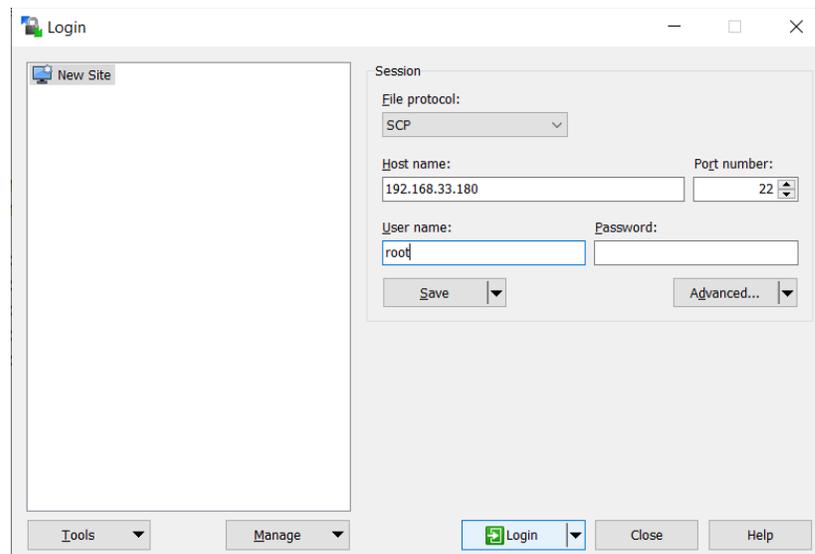
- A brief version of the file transfer status can also be viewed in the output log window. Once the entire transfer is complete, a message appears displaying *Transfer of the Captured files COMPLETE!*

```
[11:04:38] Downloading file /mnt/ssd/capture_data_1/master_0000_data.bin...
[11:04:42] Downloaded file /mnt/ssd/capture_data_1/master_0000_data.bin from SSD!
[11:04:42] Downloading file /mnt/ssd/capture_data_1/master_0000_idx.bin...
[11:04:42] Downloaded file /mnt/ssd/capture_data_1/master_0000_idx.bin from SSD!
[11:04:42] Downloading file /mnt/ssd/capture_data_1/slave1_0000_data.bin...
[11:04:45] Downloaded file /mnt/ssd/capture_data_1/slave1_0000_data.bin from SSD!
[11:04:45] Downloading file /mnt/ssd/capture_data_1/slave1_0000_idx.bin...
[11:04:45] Downloaded file /mnt/ssd/capture_data_1/slave1_0000_idx.bin from SSD!
[11:04:45] Downloading file /mnt/ssd/capture_data_1/slave2_0000_data.bin...
[11:04:50] Downloaded file /mnt/ssd/capture_data_1/slave2_0000_data.bin from SSD!
[11:04:50] Downloading file /mnt/ssd/capture_data_1/slave2_0000_idx.bin...
[11:04:50] Downloaded file /mnt/ssd/capture_data_1/slave2_0000_idx.bin from SSD!
[11:04:50] Downloading file /mnt/ssd/capture_data_1/slave3_0000_data.bin...
[11:04:55] Downloaded file /mnt/ssd/capture_data_1/slave3_0000_data.bin from SSD!
[11:04:55] Downloading file /mnt/ssd/capture_data_1/slave3_0000_idx.bin...
[11:04:56] Downloaded file /mnt/ssd/capture_data_1/slave3_0000_idx.bin from SSD!
[11:04:56] Export Operation was successful!
[11:04:56] [RadarAPI]: Transfer of the Captured files COMPLETE!
```

- Once the transfer is complete to the PC file system, along with the data files, there are also other files exported to the same directory (which will be used for Post Processing).
 1. *Directory_name.mmwave.json*: contains the mmWave configuration used for all the devices.
 2. *Directory_name.setup.json*: contains the information regarding the meta image used and the directory path used for the capture.
 3. *Directory_name_LogFile.txt*: contains the log sent to Matlab used for the in-built basic Matlab Post processing.

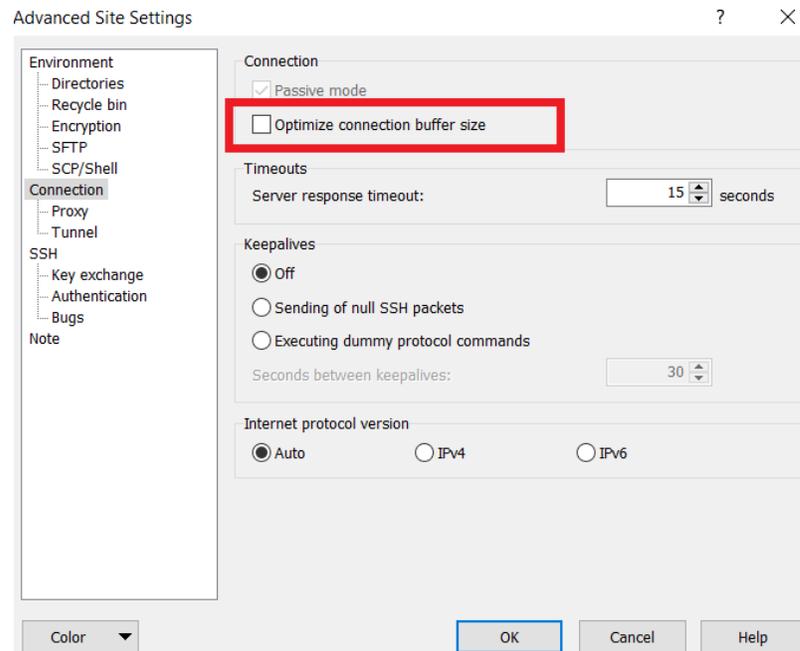
Name	Date modified	Type	Size
 capture_data_1.mmwave.json	8/7/2019 12:51 PM	JSON File	31 KB
 capture_data_1.setup.json	8/7/2019 12:51 PM	JSON File	1 KB
 capture_data_1_LogFile.txt	8/7/2019 12:48 PM	Text Document	6 KB
 master_0000_data.bin	5/14/2019 1:27 AM	BIN File	355,212 KB
 master_0000_idx.bin	5/14/2019 1:27 AM	BIN File	5 KB
 slave1_0000_data.bin	5/14/2019 1:27 AM	BIN File	355,212 KB
 slave1_0000_idx.bin	5/14/2019 1:27 AM	BIN File	5 KB
 slave2_0000_data.bin	5/14/2019 1:27 AM	BIN File	355,212 KB
 slave2_0000_idx.bin	5/14/2019 1:27 AM	BIN File	5 KB
 slave3_0000_data.bin	5/14/2019 1:27 AM	BIN File	355,212 KB
 slave3_0000_idx.bin	5/14/2019 1:27 AM	BIN File	5 KB

8.1.2 Using WinSCP

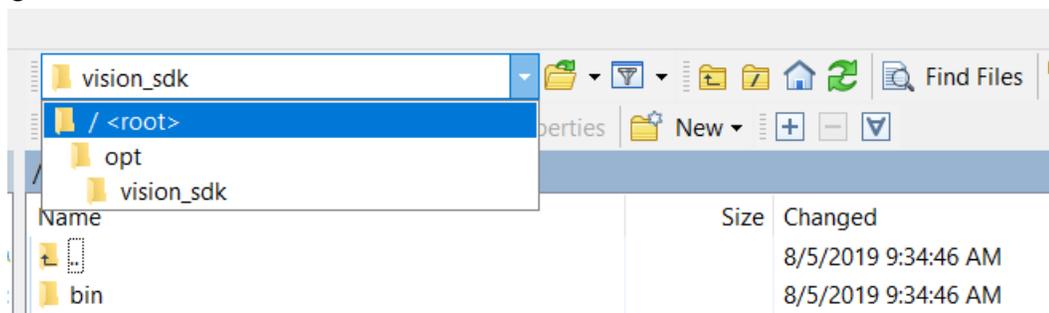


- Connect to the Host over WinSCP with following configuration
 - File Protocol - SCP
 - Host Name - 192.168.33.180
 - Port Number - 22
 - User name - root

Before clicking on Login, choose advanced option present in the opening window.



- As shown in the screenshot above, navigate to “Connection” in the tree structure of Advanced Site Settings. Uncheck the Optimize connection buffer size option. Once done, click OK and Login.
- The default folder that the GUI opens up is “/home/root”.
- Navigate to “/<root>” folder

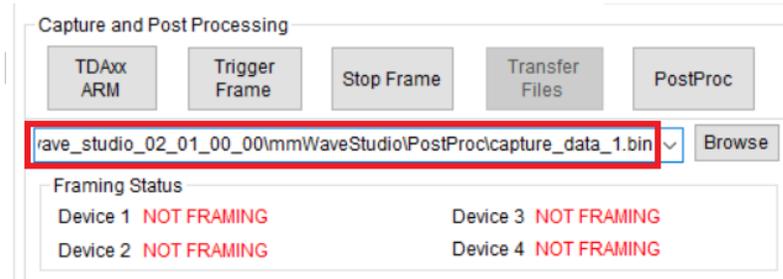


- Go to /mnt/ssd folder. This folder contains the data folders corresponding to all the capture sessions. Choose the directory to transfer and drag them to the left hand side window (PC file system). Make sure the left hand side window is in the correct directory where the data files are intended to be present. It is ideal to match the path present in the left hand side window to the path provided for Post Processing in Sensor config tab.
- The transfer of files along with the status will be shown until it is completely transferred.

8.2 Post Processing the ADC Data

8.2.1 Default processing (done in a same session as configuration and capture)

- After the files have been transferred from the SSD to PC, the basic post-processing of the data can be done. Make sure the path provided is same as that of the files that have been transferred from the previous step.



- Click on the Post Processing button to view the basic plots obtained from the captured data.

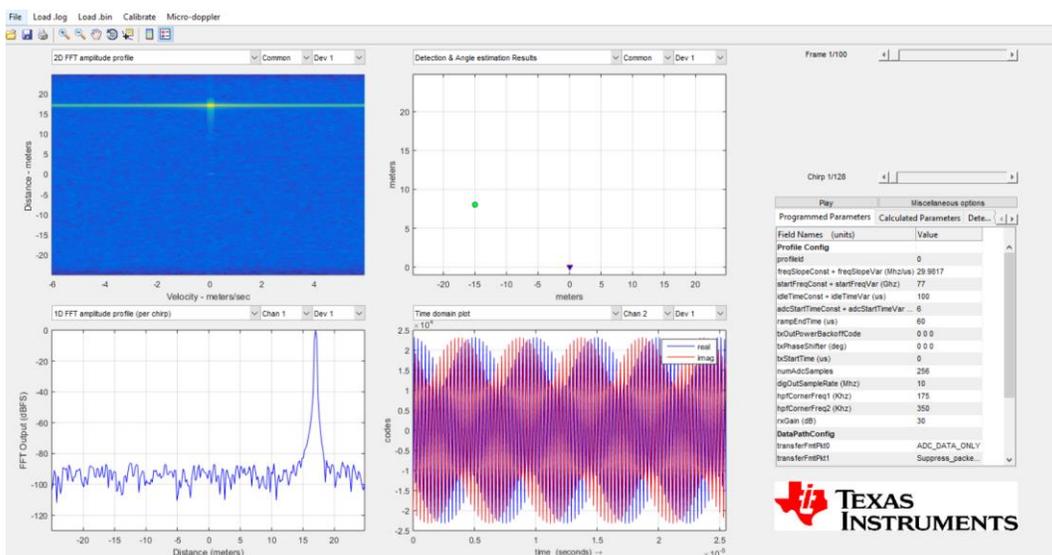
The plots can be viewed for a particular device using the device selection drop down option available against each plot.

NOTE: The in-built basic post-processing has following limitations:

- It Post-processes the data individually for each device.
- It assumes same configuration for all devices.

Sample Post Processing Plots

Device – 1 – Test Source at 17m (x = 15m and y = 8m)

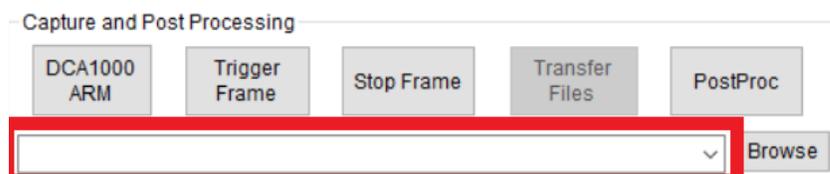


8.2.2 Standalone processing (done in a different session from configuration and capture)

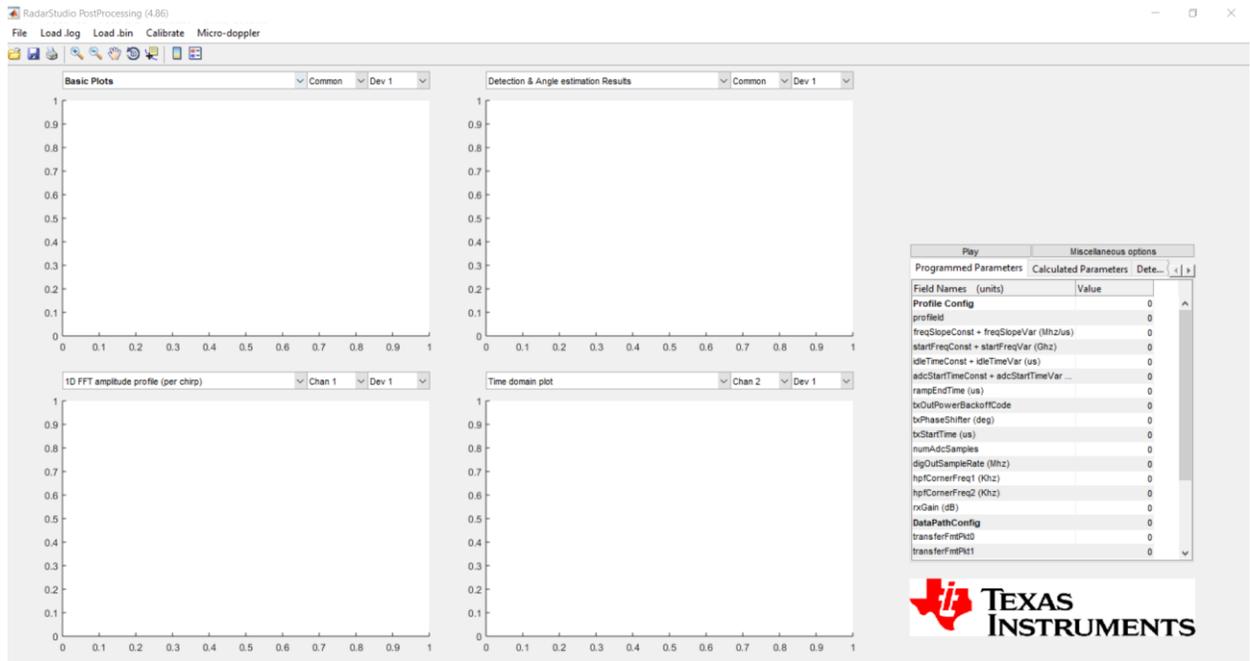
- This is particularly useful when the data is already captured from a previous session and the user just needs to post process the data using Studio.
- If the data to be analyzed is for multiple devices, then choose the Radar System as “Cascade” from the left pane.



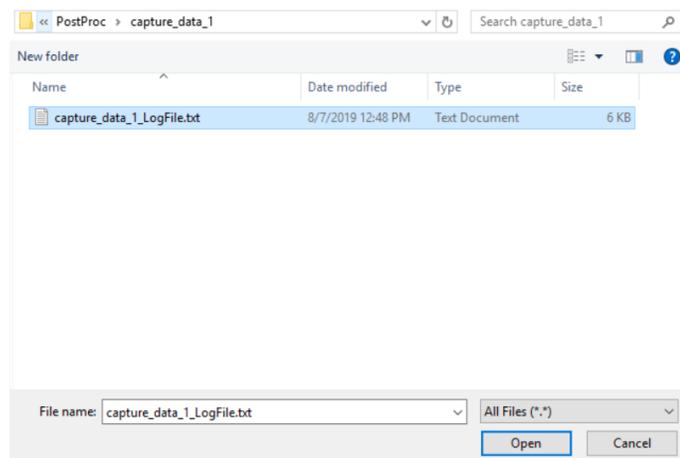
- Go to the “Sensor Config” tab directly once Studio is opened.
- Provide an empty path (like in the screenshot below) and click on “PostProc” button.



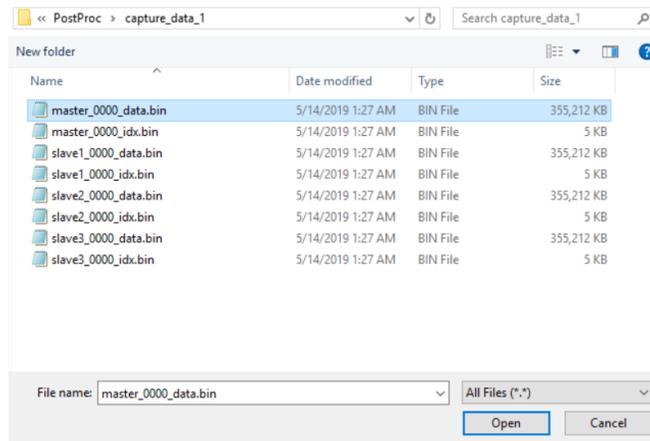
- An empty Post processing window will be opened



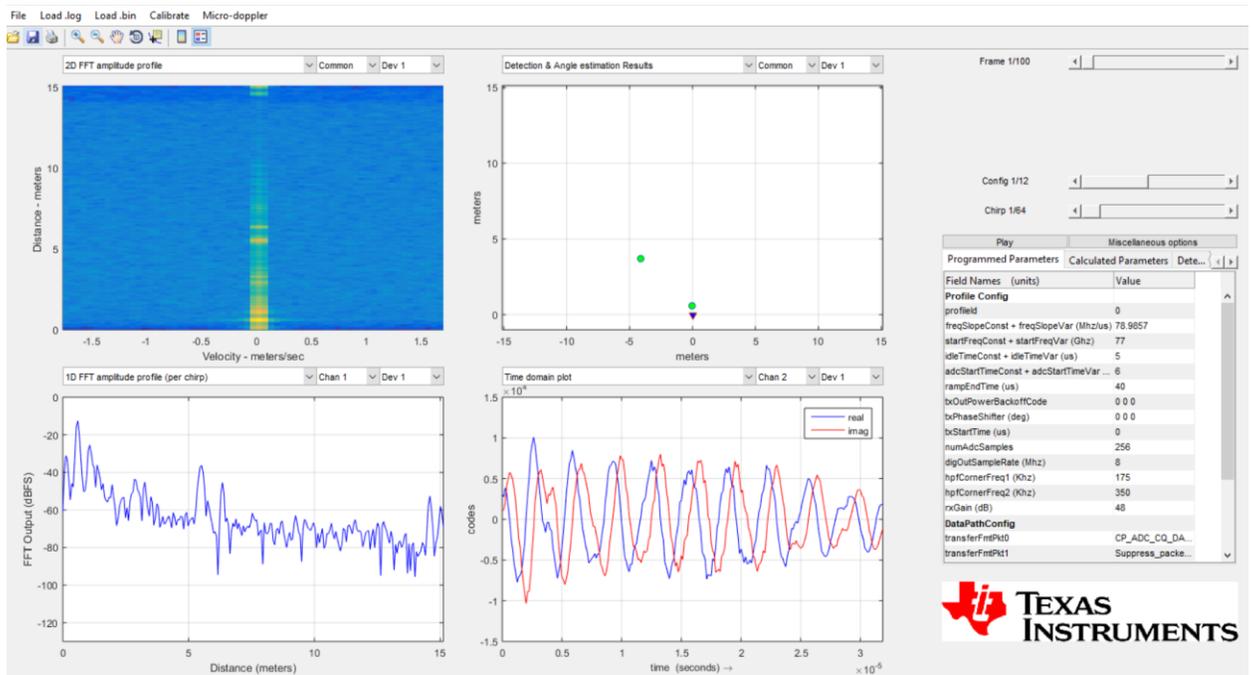
- Click on “Load .log” on the toolbar and select the corresponding log file.



- Click on “Load .bin” on the toolbar and select the corresponding bin file.



- **Note:** It post processes the data individually for each of the device.
- The basic plots will be obtained for the bin file and the log file selected



8.3 Cascade Use Cases – Configuration and Post-processing

This section is relevant for post-processing the data captured for MIMO or Tx Beamforming scenario. Refer (<http://www.ti.com/lit/ug/tiduen5/tiduen5.pdf>) for more information.

8.3.1 Calibration

8.3.1.1 Inter-channel mismatch calibration

The inter-channel mismatch calibration is done using a corner reflector at about 5m and using a TDM MIMO configuration. Inter-channel mismatch is expected to be different on each board and must be explicitly calibrated for. By default, the post-processing scripts include a dummy calibration matrix. Users should calibrate each board explicitly by following the procedure in [Section 8.3.2.2](#).

8.3.1.2 Phase Shifter Calibration

The post-processing scripts currently use the same phase-shifter calibrations for all boards. Customers wishing to get better phase-shifter accuracy may want to calibrate the phase shifters explicitly for all TXs and devices.

8.3.2 TDM MIMO

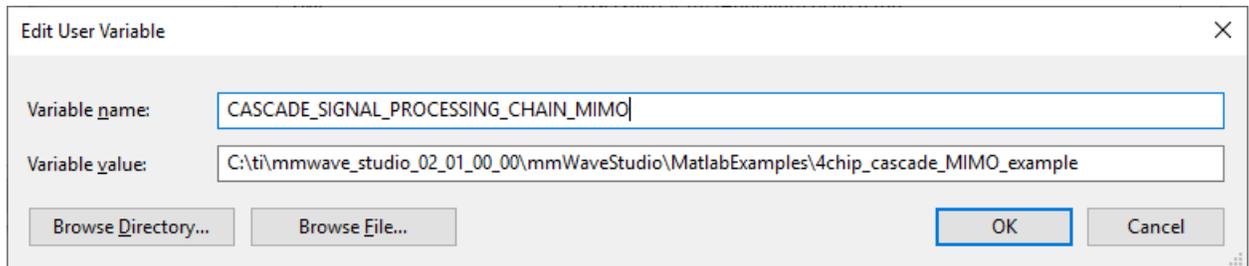
Reference example for Post-processing the raw ADC data (captured for Cascade MIMO scenario) is already included (`\mmWaveStudio\MatlabExamples\4chip_cascade_MIMO_example`) in the mmWaveStudio installation. Matlab installation is required to run this example. Refer to *use case-2* of [Section 7.2](#) to capture the Data. This section explains the steps to run this example. The user is expected to make changes to this example based on the updates to the capture scenario.

There are 3 steps for running this example:

- Environment setup
- Generating the Calibration matrix (requires only once for a RF-EVM board)
- Post-processing the actual scenario data using the Calibration matrix.

8.3.2.1 Environment Setup

- Go to Control Panel -> System and Security -> System -> Advanced system settings -> Environment Variables.
- Add the Environment variable “CASCADE_SIGNAL_PROCESSING_CHAIN_MIMO” pointing to “<Installation_path>\mmWaveStudio\MatlabExamples\4chip_cascade_MIMO_example” to your PC.



NOTE: Restart MATLAB after adding the environment variable to reflect the changes made.

- Open Matlab and Set the current path to
 “<Installation_path>\mmWaveStudio\MatlabExamples\4chip_cascade_MIMO_example”
- Run “*add_paths.m*” in the command window to add the current folders to Matlab path

8.3.2.2 Calibration

This step is required only once for each Cascade RF-EVM. Once you have the calibration matrix, it can be used for any future captures with the same EVM setup.

NOTE: Although there is a file present in the folder for calibration, it is always recommended for the user to calibrate it once.

- Capture the data by running the lua script for MIMO use-case while keeping the Corner-reflector in boresight at around 5 meters. There should not be any strong reflector nearby.

NOTE: To understand the antenna calibration, refer to the “Antenna Calibration” section in the “*signal_processing_4chip_cascade.pdf*” in the “MatlabExamples” folder

- Once the calibration data is collected, go to the `.\main\cascade` folder in matlab and update the “*dataFolder_calib_data*” in “*cascade_MIMO_antennaCalib.m*” to point to calibration data folder.
- Run the “*cascade_MIMO_antennaCalib.m*”. It should update the “*calibrateResults_high.mat*” file in input folder. This file is the calibration matrix for your EVM.

NOTE: If the user is going with the default calibration already present, make sure that the working directory is “`.\main\cascade`” folder in MATLAB before proceeding to the post processing step.

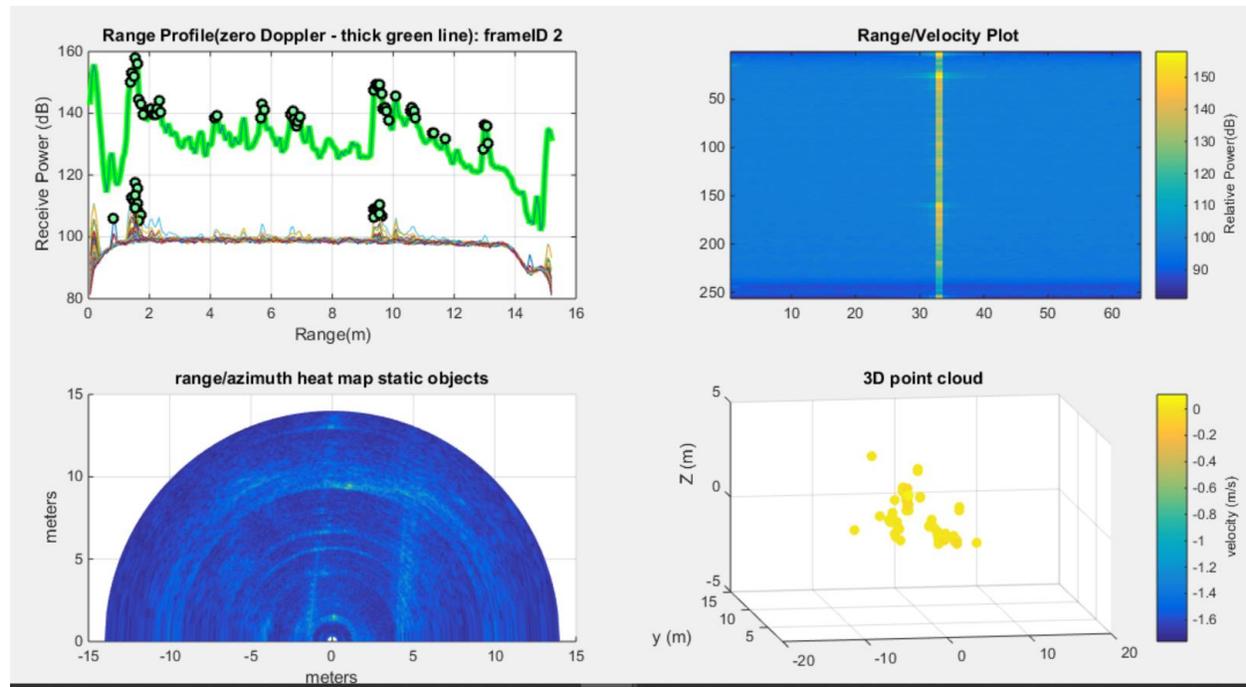
8.3.2.3 Post-processing

- Collect the raw adc data for the actual scenario using the lua script provided for MIMO.
- Update the first line of the \input\testList.txt file to point to the Captured Data folder (as shown below).
- Update the second line of the \input\testList.txt file to point to the corresponding calibration vector generated from the calibration step
- Update the third line for algorithm parameter tuning if needed; the default values are in \paramGen\module_param.m

```

1 C:\ti\mmwave_studio_02_01_00_00\mmWaveStudio\PostProc\MIMO_Capture\
2 .\input\calibrateResults_high.mat
3 .\paramGen\module_param.m
4
    
```

- Run the “cascade_MIMO_signalProcessing.m” file. If the post-processing script is run correctly, you will see the following GUI.



8.3.3 TX Beamforming

8.3.3.1 Steps to capture the data for TX Beam Forming

- Go to “<mmWaveStudio Installation>\mmWaveStudio\MatlabExamples\4chip_cascade_TxBF_example\”.
- Run the “mmWaveStudio\Scripts\Cascade\Cascade_Basic_Configuration.lua” from mmwavestudio GUI to configure the devices till datapath configuration. Change the required settings in the LUA script if required.

NOTE: Only the Profile, Chirp Configuration and Phase Shifter Configuration is done from Matlab. The rest of the configuration is still done from lua.

- Open Matlab and go the “<mmWaveStudio Installation>\mmWaveStudio\MatlabExamples\4chip_cascade_TxBF_example\”
- Open “cascade_TxBF_dataCapture.m” to update following paths
 - phaseShiftCalibfile -> Phase shifter calibration file, the same calibration file is used by default for all boards
 - phaseMismatchCalibfile -> Phase Mismatch calibration file, the same calibration matrix generated in the MIMO mode will apply
 - RSTD_DLL_Path -> Path to RttNetClientAPI.dll file in studio installation
 - paramFile -> Parameter file according to the use case to specify the chirp and profile configuration; an example parameter file is provided
- Execute the following command in matlab command prompt
 - “run cascade_TxBF_dataCapture.m”

NOTE: This step assumes that the Studio is already running (corresponding to the RSTD_DLL_Path)

- Once the execution of the configuration script is completed, run the “mMWaveStudio\Scripts\Cascade\Cascade_Capture.lua” script for capturing data.

8.3.3.2 Steps to process the captured data

- Script `cascade_TxBF_signalProcessing.m` at `'mmWaveStudio\MatlabExamples\4chip_cascade_TxBF_example'` directory can be used to process the captured data from above steps.
- Update the following variables in `cascade_TxBF_signalProcessing.m` file with values corresponding to the captured data.

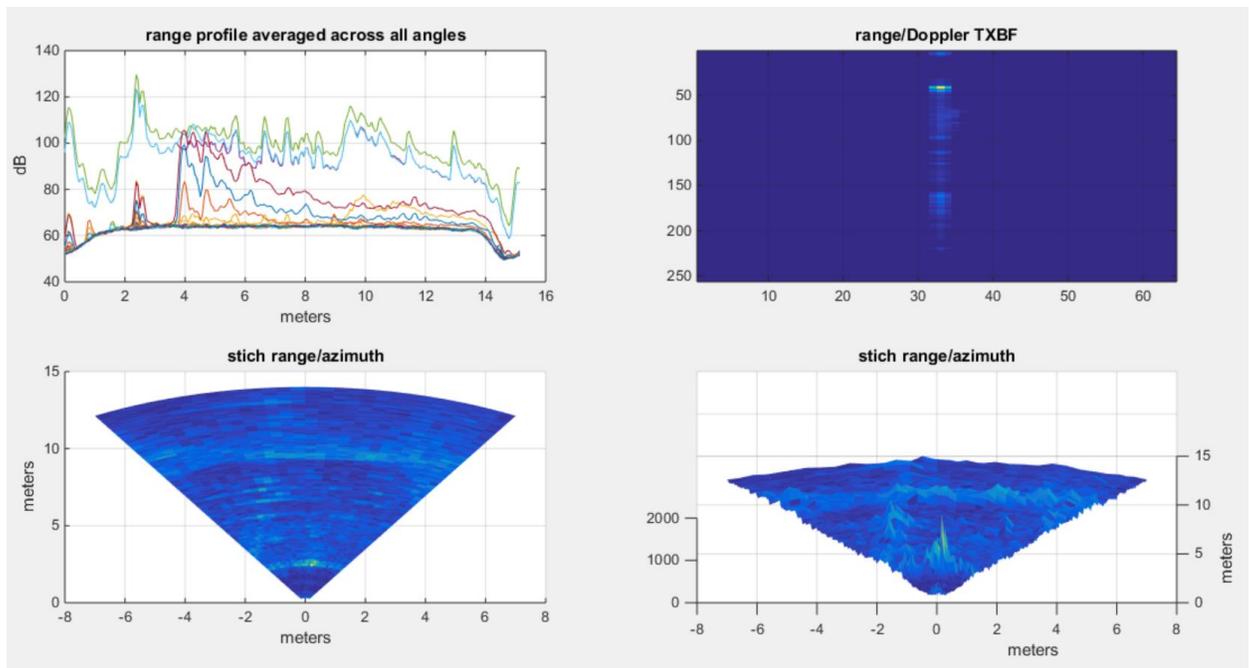
BF_data_dest0: adc test data path

paramsConfig.anglesToSteer: steering angles associated with the collected test data, this value **MUST be the same as** the “`params.anglesToSteer`” value specified in `paramFile` used during data capture.

paramsConfig.Chirp_Frame_BF: specify if the collected data is chirp based or frame based beam steering, this value **MUST be the same as** the “`params.Chirp_Frame_BF`” value specified in `paramFile` used during data capture

calibrationFilePath : specify the calibration file for the specific board to compensate the channel mismatch, the same calibration matrix generated in the MIMO mode will apply.

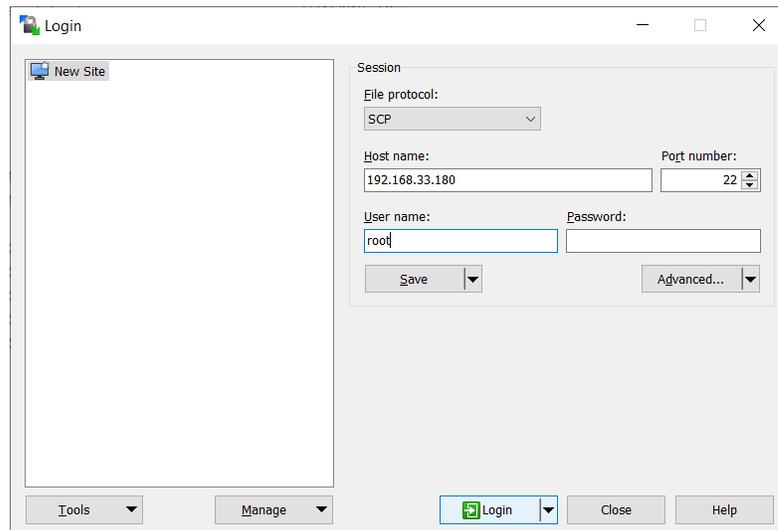
- If the `cascade_TxBF_signalProcessing.m` is run correctly, you will see the following GUI. The top left figure shows the range profiles averaged across the steering angles with different color referring to different Doppler line.



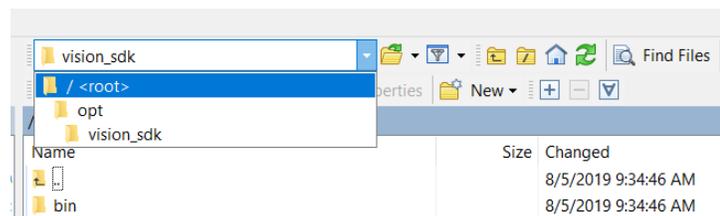
9. Deleting data on SSD

Since mmWaveStudio does not delete any data that is stored on the SSD, the SSD might fill up after a while. This section explains how to delete the data files from the SSD along with the trace files to free up the memory for further captures.

- Connect to the Host over WinSCP with following configuration
 - File Protocol - SCP
 - Host Name - 192.168.33.180
 - Port Number - 22
 - User name - root



- The default folder that the GUI opens up is “/home/root”.
- Navigate to “/<root>” folder



Navigate to “/opt/vision_sdk” folder. It contains the trace files for all the sessions with file name as Trace_TDA_<Date>_<time>.txt. Delete all these files if it is not required.

- Navigate to “/<root>” folder. Go to /mnt/ssd folder. This folder contains the data folders corresponding to all the sessions
- Delete the folders. Make sure the data is copied on the PC already before deleting data

10. Miscellaneous Features

10.1 Read/Write GPIO in AWR

Reading/Writing the GPIO pins present in AWR devices is a useful way of debugging in the case of an issue/fault. Please refer to TDA2XX and RF board schematic to find these GPIO locations.



- Go to RegOp Tab. There is a dedicated “TDA GPIO” section present in the bottom right corner of the tab.
- It automatically populates the GPIO Base and GPIO Pin based on the Pin Description.

The following pins are supported as of now

- AWR1_RESETN
- AWR2_RESETN
- AWR3_RESETN
- AWR4_RESETN
- AWR_WARM_RST
- AWR_SOP2
- AWR_SOP1
- AWR_SOP0

The CUSTOM_GPIO pin is not supported as of now in this release.

- For Read operation, click on “Read” button and the value read will be displayed.
- For Write operation, provide the value to be written in the GPIO Value (Hex) option. And then click on “Write”.

10.2 Read Width and Height from TDA



- After the capture is complete, the user can verify whether TDA was configured properly for capturing data by pressing the “Get” button in Sensor Config Tab.

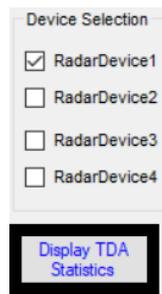
- This information is purely for debug and not otherwise needed.
- This displays the width and height obtained from TDA. It is recommended to use this *only* after a capture operation.
- TDA2XX captures the frames from AWR devices in terms of width and height programmed during “TDAXX ARM” operation. Width is defined to be half the number of bytes per chirp per device. Height is the total number of chirps per frame (or subframe).
The frame size per device can be calculated using

$$\text{Frame size (in bytes)} = 2 * \text{Width} * \text{Height}$$

NOTE: The maximum width that is supported with TDA2XX is 4096. This corresponds to a maximum of 512 complex samples or 1024 real samples per chirp per RX.

NOTE: In the case of Advanced frame, if the sub frames configured have different heights, the height that is programmed in TDA is the maximum of the heights of sub frames configured.

10.3 Display CPU Statistics from TDA



- During framing, the CPU core usage and data rate statistics of the VIP ports on TDA can be obtained in the teraterm on clicking “Display TDA Statistics”. It is recommended to use this *only* when TDA is capturing data from the AWR devices.

NOTE: The statistics can be viewed on the teraterm console only when apps.out (present in the /opt/vision_sdk folder) is ran manually from teraterm.

In order to accomplish this, please refer [Section 10.4](#).

10.4 Running apps.out application manually from teraterm

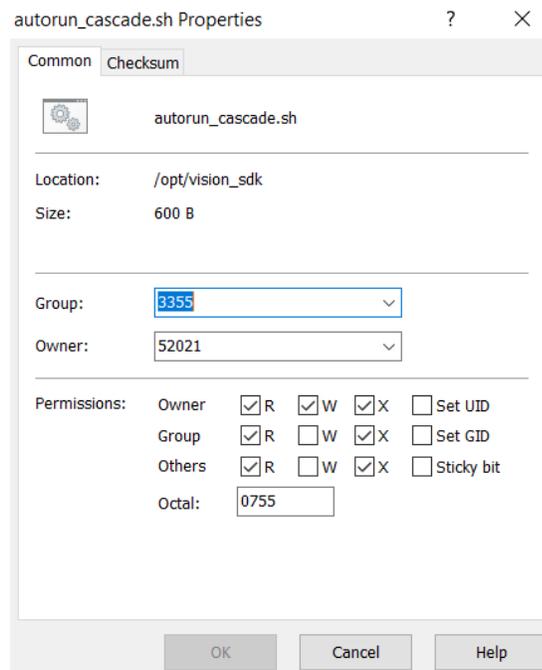
The apps.out application in the TDA2XX image is responsible for interacting with mmWaveStudio for the support of cascade operation.

By default, the apps.out application in the TDA2XX side is running automatically. This does not provide a debug interface where the commands logs are captured in teraterm. In order to debug an issue, it is useful to run apps.out manually so as to obtain the command logs in teraterm.

- Open WinSCP and login into the TDA2XX host using the IP address.
- Navigate to /opt/vision_sdk folder.
- Edit the autorun_cascade.sh script (script which runs automatically during boot up) and comment the line (using “#”) containing apps.out like the one below

#./apps.out

- Save the script and change the owner/group and executable permissions
 - Right click on autorun_cascade.sh script and choose Properties
 - Change the group and owner (from root to the next option available in the dropdown) and provide execute permissions as follows



- Open teraterm corresponding to the COM port of TDA. If it prompts for username, enter “root”. Once done, enter “sync” -> synchronizes corresponding file data in volatile memory and permanent storage.
- Reopen the script in WinSCP (or in teraterm using VI editor) to verify whether the changes made are in effect.
- Reboot the TDA2xx Host board.

```

Arago Project http://arago-project.org dra7xx-evm ttyS0
Arago 2016.12 dra7xx-evm ttyS0

dra7xx-evm login: root
root@dra7xx-evm:~# cd /opt/vision_sdk/
root@dra7xx-evm:/opt/vision_sdk# ps | grep apps
root@dra7xx-evm:/opt/vision_sdk# ./apps.out

```

- Once booted, enter the login username as “root”
- Navigate to /opt/vision_sdk folder using the below command

```
cd /opt/vision_sdk
```

- In order to verify whether apps.out is running automatically or not, issue the command

```
ps | grep apps
```

This should not show `./apps.out` as one of the results.

If it does show up, (like in the screenshot below) then the script changes made in the previous session are not in effect. In that case, restart the same procedure from beginning.

```

root@dra7xx-evm:/opt/vision_sdk# ps | grep apps
793 root      1763m R    ./apps.out
885 root      1968 S    grep apps

```

- Once it is confirmed that apps.out application is not running, provide the command in the console

```
./apps.out
```

- On successful execution, it should end up with the following information in the console.

```

[HOST] [HOST] ] =====
[HOST] [HOST] ] Vision SDK Usecases
[HOST] [HOST] ] =====
[HOST] [HOST] ] 1: Single Camera Usecases
[HOST] [HOST] ] 2: Multi-Camera LWDs Usecases
[HOST] [HOST] ] 3: Open-Compute Usecases
[HOST] [HOST] ] 6: Radar Usecases
[HOST] [HOST] ]
[HOST] [HOST] ] 7: PCIe SSD card Write Test
[HOST] [HOST] ] p: CPU Status
[HOST] [HOST] ] i: Show IP Addr <IPU + NDK + AUB>
[HOST] [HOST] ]
[HOST] [HOST] ] s: System Settings
[HOST] [HOST] ]
[HOST] [HOST] ] x: Exit
[HOST] [HOST] ] z: Exit - AND SHUTDOWN Remote CPUs
[HOST] [HOST] ]
[HOST] [HOST] ] Enter Choice:
[HOST] [HOST] ]

```

- After this, follow the regular steps with mmWaveStudio (starting from [Section 7](#)). All the commands received by the TDA2xx Host will be shown in the teraterm console.

```

[HOST] [HOST] 1 779.171597 s:
[HOST] [HOST] 1
[HOST] [HOST] 1 =====
[HOST] [HOST] 1 Vision SDK Usecases
[HOST] [HOST] 1 =====
[HOST] [HOST] 1 1: Single Camera Usecases
[HOST] [HOST] 1 2: Multi-Camera LUDS Usecases
[HOST] [HOST] 1 3: Open-Compute Usecases
[HOST] [HOST] 1 6: Radar Usecases
[HOST] [HOST] 1
[HOST] [HOST] 1 7: PCIe SSD card Write Test
[HOST] [HOST] 1 p: CPU Status
[HOST] [HOST] 1 i: Show IP Addr (IPU + NDK + AUB)
[HOST] [HOST] 1
[HOST] [HOST] 1 s: System Settings
[HOST] [HOST] 1
[HOST] [HOST] 1 x: Exit
[HOST] [HOST] 1 z: Exit - AND SHUTDOWN Remote CPUs
[HOST] [HOST] 1
[HOST] [HOST] 1 Enter Choice:
[HOST] [HOST] 1
[HOST] [HOST] 1 830.978740 s: Received NETWORK_RADAR_CAPTURE_CONFIG_PING command
[HOST] [HOST] 1 830.982889 s: Received NETWORK_RADAR_CAPTURE_CONFIG_TRACE_START command
[HOST] [HOST] 1 830.996666 s: Received NETWORK_RADAR_CAPTURE_CONFIG_VERSION_GET command
[HOST] [HOST] 1 830.996644 s: Received NETWORK_RADAR_CAPTURE_CONFIG_DEVICE_MAP command
[HOST] [HOST] 1 831.053193 s: Received NETWORK_RADAR_CAPTURE_CONFIG_CONNECT command
[HOST] [HOST] 1 832.653480 s: UTILS_MCSPI: McSPI is configured in interrupt mode!!
[HOST] [HOST] 1 832.653846 s: SYSTEM: UARI: INTERRUPT Mode is Selected
[HOST] [HOST] 1 832.654242 s: UTILS_MCSPI: McSPI is configured in interrupt mode!!
[HOST] [HOST] 1 832.654700 s: UTILS_MCSPI: McSPI is configured in interrupt mode!!
[HOST] [HOST] 1 832.655035 s: UTILS_MCSPI: McSPI is configured in interrupt mode!!
[HOST] [HOST] 1 832.788019 s: Received NETWORK_RADAR_SENSOR_CONFIG_SET_SOP command
[HOST] [HOST] 1 833.089123 s: Received NETWORK_RADAR_SENSOR_CONFIG_DEVICE_RESET command

```

11. Configuring IP Address

By default, the IP address for eth0 is statically set to 192.168.33.180. The following sections explain how to configure a different static IP address and how to configure the system to acquire an IP address using DHCP. Use TeraTerm and the vi editor command to edit these files.

11.1 Static IP Address (default)

By default, the IP address is statically set to 192.168.33.180. Modify the eth0 section in the `/etc/network/interfaces` file as needed to select a different IP address. The following shows the default configuration.

```
# Wired or wireless interfaces
auto eth0
iface eth0 inet static
    address 192.168.33.180
    netmask 255.255.255.0
    gateway 192.168.33.1
#iface eth0 inet dhcp
#   pre-up /bin/grep -v -e "ip=[0-9]\+\.[0-9]\+\.[0-9]\+\.[0-9]\+" /proc/cmdline > /dev/null
#   udhcpc_opts -R -b
```

Make sure that the two lines shown in bold below in the [Network] and [DHCP] sections of the `/lib/systemd/network/80-container-host0.network` file are commented out.

```
[Network]
#DHCP=yes
LinkLocalAddressing=yes

[DHCP]
#UseTimezone=yes
```

Make sure that the ExecStart line in the [Service] section of the `/lib/systemd/system/systemd-networkd.service` file is as shown below

```
[Service]
Type=notify
Restart=on-failure
RestartSec=0
ExecStart=/sbin/ifup eth0
#ExecStart=/lib/systemd/systemd-networkd
CapabilityBoundingSet=CAP_NET_ADMIN CAP_NET_BIND_SERVICE CAP_NET_BROADCAST CAP_NET_RAW CAP_SETUID
CAP_SETGID CAP_SETPCAP CAP_CHOWN CAP_DAC_OVERRIDE CAP_FOWNER
ProtectSystem=full
ProtectHome=yes
WatchdogSec=3min
```

11.2 DHCP

Modify the eth0 section in the `/etc/network/interfaces` file as below to comment out the static configuration and enable DHCP.

```
# Wired or wireless interfaces
auto eth0
#iface eth0 inet static
#     address 192.168.33.180
#     netmask 255.255.255.0
#     gateway 192.168.33.1
iface eth0 inet dhcp
    pre-up /bin/grep -v -e "ip=[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+" /proc/cmdline > /dev/null
    udhcpd_opts -R -b
```

Make sure that the two lines shown in bold below in the [Network] and [DHCP] sections of the `/lib/systemd/network/80-container-host0.network` file are not commented out.

```
[Network]
DHCP=yes
LinkLocalAddressing=yes

[DHCP]
UseTimezone=yes
```

Make sure that the ExecStart line in the [Service] section of the `/lib/systemd/system/systemd-networkd.service` file is as shown below

```
[Service]
Type=notify
Restart=on-failure
RestartSec=0
#ExecStart=/sbin/ifup eth0
ExecStart=/lib/systemd/systemd-networkd
CapabilityBoundingSet=CAP_NET_ADMIN CAP_NET_BIND_SERVICE CAP_NET_BROADCAST CAP_NET_RAW CAP_SETUID
CAP_SETGID CAP_SETPCAP CAP_CHOWN CAP_DAC_OVERRIDE CAP_FOWNER
ProtectSystem=full
ProtectHome=yes
WatchdogSec=3min
```

12. Known Issues and Limitations

Serial Number	Description
1	The first frame may be corrupted in every capture. It is recommended to ignore the first frame for all captures.
2	If the “num_frames_to_capture” parameter is 0, then the number of frames captured by the TDA2 is one less than the number of frames sent by the AWR devices when the AWR devices are configured to send out a finite number of frames. However if the “num_frames_to_capture” parameter is set to a non-zero value, then the TDA2 will correctly capture exactly that many number of frames.
3	The maximum number of samples per chirp is currently limited to 512 in complex 1x/2x modes, and 1024 in real mode. This is a limitation of the TDA2. There is no workaround at this time.
4	The 12-bit and 14-bit ADC data format modes are not supported. Only 16-bit ADC data is supported and verified.
5	The basic built-in post processing tool in mmWave Studio assumes that the configuration is the same for all devices. The angle estimation done by the tool only takes into account data from the 4 RXs of one device at a time.
6	Logs from the TDA2 apps.out application are only generated when running the apps.out binary manually. The default behavior in which this application is started automatically at bootup does not generate the logs. These logs are required only for debug. Refer to Section 10.4 for instructions on how to run apps.out manually.
7	Support for the continuous streaming mode is currently limited. When this mode is enabled, only one frame of data is captured.
8	Sometimes mmWave Studio closes itself without any notification due to some random action performed on it.

13. Troubleshooting

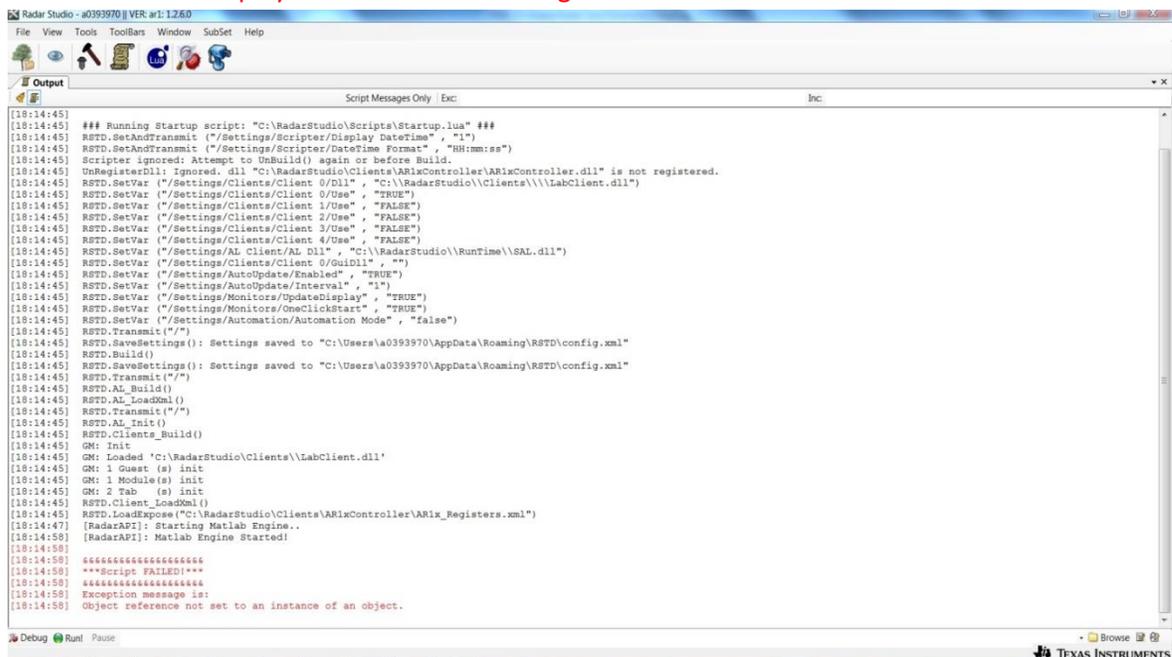
- **Not able to connect TDA2XX from mmWaveStudio over the network**
 - a. Check the Ethernet cable connection, LED status on TDA2xx Ethernet port connection.
 - b. Ensure the PC IP address to be the same as mentioned in the [Section 7.3.1](#).
 - Try to ping TDA2XX IP address from PC to verify if the network connection is proper.
 - c. Check the terminal screen if it had any error message while booting up.
 - If yes, then try to reboot.
 - Flash the SD card again if the error persists.
 - d. At the terminal screen, check if *apps.out* process is running with the command `'ps | grep 'apps.out''`
apps.out is the application running on TDA2XX which establishes network connection with mmWaveStudio.
 - Ideally this application should run automatically on boot up. If not, then run it manually. Refer [Section 10.4](#) on how to run it manually.
 - e. Restart the TDA2XX board if none of above solution works.

- **Not able to connect mmWave Sensor over SPI (via SPI Connect button)**
 - a. Make sure that you have clicked 'Set' button in SOP Control GUI block before connecting to the specific device.
 - b. For the Slave SPI connection, make sure that you have configured Master device with Channel configuration API (cascading mode = Master).
 - c. Check the mmWave Sensor RF board connection with TDA2xx board.
 - d. Check the RF board LEDs which maps to each mmWave sensor.
 - At the boot up, all the LEDs (LEDs DS4, DS5, DS6 and DS7) on RF board will be ON and will be switched OFF when you connect TDA2 from mmWaveStudio over network.
 - When you click 'Set' button in SOP control GUI block, then the corresponding single LED which maps to a specific device will glow up. Only then you should click 'SPI Connect' button to do SPI connection.
 - e. Reset all the devices by clicking 'Set' button with 'RadarDevice1' (master) checkbox selected. This will reset all the mmWave Sensors.
 - f. Re-connect TDA2XX network connection from 'Setup TDAxx Capture Card' pop-up GUI.

- **Facing issues with running LUA script**
 - a. Make sure that you have followed the steps mentioned in [Section 7.2](#)

- **RF Power-up fails**
 - a. Make sure that
 - Correct firmware has been downloaded to the device
 - No error is observed during the firmware download
 - Async event (SBID: 0x5005) of file download done is received.
 - b. RF power up draws more current in general. Hence, make sure that the right power supply is connected to the TDA2XX board (as mentioned in [Section 2](#)).

- c. If you have selected multiple slave checkbox and clicked 'RF Power-up' button, then it will effectively send commands to all the selected devices. So it is not required to click next 'RF Power-up' button (else it will throw an error).
- **Error while transferring file from TDA2XX SSD using mmWaveStudio**
 - a. Check if SSD is connected correctly on the TDA2XX board.
 - b. Check if SSD is mounted in TDA2xx: In the serial terminal check for '/mnt/ssd' directory.
 - c. Sometimes after SSD cleanup, you may see this error. In that case, reset the board.
 - **mmWaveStudio displays an error when starting for the first time**



```

Radar Studio - a0393970 | VER: a1.1.2.6.0
File View Tools Toolbars Window SubSet Help
Output
Script Messages Only | Exc: Inc:
[18:14:45] ### Running Startup script: "C:\RadarStudio\Scripts\Startup.lua" ###
[18:14:45] RSTD.SetAndTransmit ("/Settings/Scripter/Display DateTime", "1")
[18:14:45] RSTD.SetAndTransmit ("/Settings/Scripter/DateTime Format", "HH:mm:ss")
[18:14:45] Scripter ignored: Attempt to UnBuild() again or before Build.
[18:14:45] UnregisterDll: Ignored. dll "C:\RadarStudio\Clients\ARIXController\ARIXController.dll" is not registered.
[18:14:45] RSTD.SetVar ("/Settings/Clients/Client 0/Dll", "C:\RadarStudio\Clients\LabClient.dll")
[18:14:45] RSTD.SetVar ("/Settings/Clients/Client 0/Use", "TRUE")
[18:14:45] RSTD.SetVar ("/Settings/Clients/Client 1/Use", "FALSE")
[18:14:45] RSTD.SetVar ("/Settings/Clients/Client 2/Use", "FALSE")
[18:14:45] RSTD.SetVar ("/Settings/Clients/Client 3/Use", "FALSE")
[18:14:45] RSTD.SetVar ("/Settings/Clients/Client 4/Use", "FALSE")
[18:14:45] RSTD.SetVar ("/Settings/AL Client/AL Dll", "C:\RadarStudio\Runtime\SAL.dll")
[18:14:45] RSTD.SetVar ("/Settings/Clients/Client 0/Guid1", "")
[18:14:45] RSTD.SetVar ("/Settings/ToUpdate/Enabled", "TRUE")
[18:14:45] RSTD.SetVar ("/Settings/ToUpdate/Interval", "1")
[18:14:45] RSTD.SetVar ("/Settings/Monitors/UpdateDisplay", "TRUE")
[18:14:45] RSTD.SetVar ("/Settings/Monitors/OneClickStart", "TRUE")
[18:14:45] RSTD.SetVar ("/Settings/Automation/Automation Mode", "false")
[18:14:45] RSTD.Transmit("/")
[18:14:45] RSTD.SaveSettings(): Settings saved to "C:\Users\ao393970\AppData\Roaming\RSTD\config.xml"
[18:14:45] RSTD.Build()
[18:14:45] RSTD.SaveSettings(): Settings saved to "C:\Users\ao393970\AppData\Roaming\RSTD\config.xml"
[18:14:45] RSTD.Transmit("/")
[18:14:45] RSTD.AL_Build()
[18:14:45] RSTD.AL_LoadXml()
[18:14:45] RSTD.Transmit("/")
[18:14:45] RSTD.AL_Init()
[18:14:45] RSTD.Clients_Build()
[18:14:45] GM: Init
[18:14:45] GM: Loaded 'C:\RadarStudio\Clients\LabClient.dll'
[18:14:45] GM: 1 Guest (s) init
[18:14:45] GM: 1 Module(s) init
[18:14:45] GM: 2 Tab (s) init
[18:14:45] RSTD.Client_LoadXml()
[18:14:45] RSTD.LoadExpose("C:\RadarStudio\Clients\ARIXController\ARIX_Registers.xml")
[18:14:47] [RadarAPI]: Starting Matlab Engine..
[18:14:50] [RadarAPI]: Matlab Engine Started!
[18:14:50]
[18:14:50] ****Script FAILED****
[18:14:50] ****Script FAILED****
[18:14:50] Exception message is:
[18:14:50] Object reference not set to an instance of an object.
Debug Run Pause
Browse
TEXAS INSTRUMENTS

```

If the error message is displayed during the opening of mmWaveStudio in its Output window,

- a. Make sure the combination of DevPack/DCA1000 and AWR BOOST-EVM are connected to the PC while opening mmWaveStudio for the first time.
 - b. Make sure that you have installed Matlab runtime engine v8.5.1 (32-bit), link provided in mmWave_studio_user_guide.pdf
- **No files have been captured in the SSD directory**
 - a. Make sure that SSD is mounted on TDA2x and it is formatted.
 - b. Try rebooting the board and carry out the procedure again.
 - **If I close mmWave Studio or it closes by itself (due to some issue) while TDA2xx is connected in GUI then next time it doesn't connect to TDA2xx.**
 - a. Ideal way is to disconnect the TDA2xx first then close the mmWave Studio GUI.
 - b. If it happens then try rebooting the board and carry out the procedure again.