Developer's Guide

mmWave Studio CLI Tool

Overview

This tool is a demonstration of controlling mmwave sensor using command line interface (CLI). Here is the list of features this tool provides

1. Command line interface to configure the mmwave sensor device over UART.
2. It uses all the basic inputs from a text file, so user can control the tool's functionalities from this input text file.
3. Support CFG and JSON file format for mmwave sensor configuration parameters.
4. Configure the DCA1000EVM and capture the ADC data from mmwave sensor to PC.
5. Capture the monitoring report generated from mmwave sensor to JSON file and later plot it.
6. Post processing of the captured ADC data.

Following figure shows the high level of flow diagram of this tool.

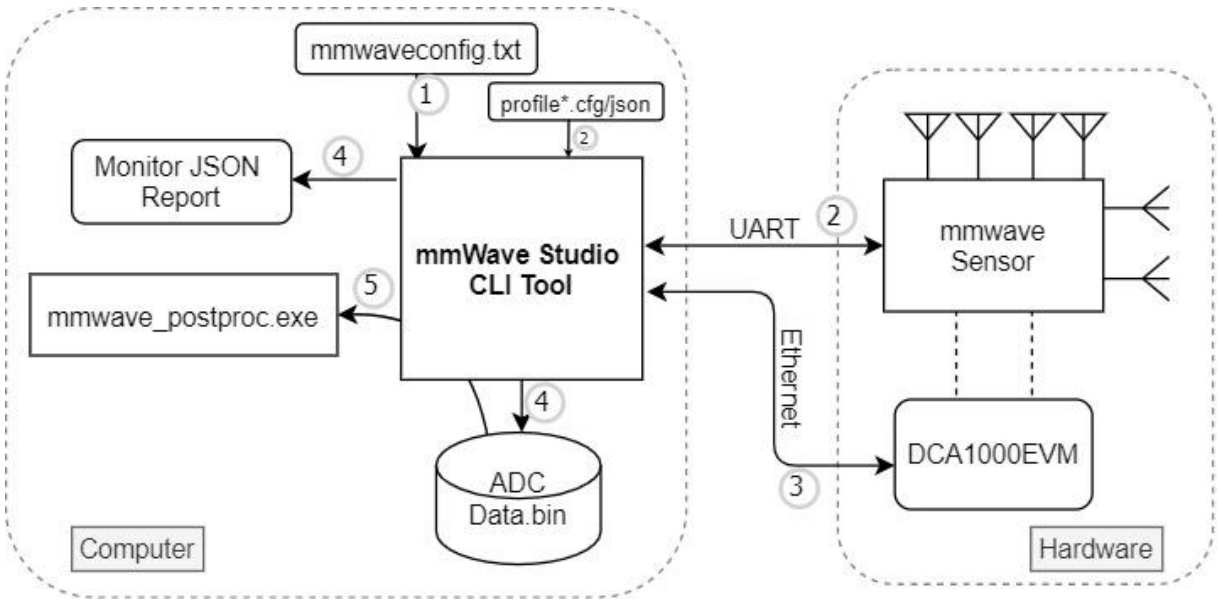

Figure 1 mmWave Studio CLI Tool Flow Diagram

Pre-requisite

1. mmWave Sensor device board or EVM.
2. DCA1000EVM to capture the ADC data over LVDS.
3. Two Micro USB cables & power adaptors for above two boards.
4. Ethernet cable to connect DCA1000EVM with PC.
5. PC running on Windows 10 OS.
6. FTDI driver installation on PC to connect with DCA1000EVM.
    a. This is available in mmWave Studio package under 'ftdi' directory.
    b. For driver update steps, please follow section 2.3 of mmwave_studio_user_guide.pdf from mmwave Studio installation.
7. Install 32-bit Matlab Runtime Engine (Version 8.5.1): It is used to run the Post-Processing utility within this tool.

NOTE: Please make sure the Matlab Runtime Engine installed is exactly same as 32-bit Version 8.5.1
https://in.mathworks.com/supportfiles/downloads/R2015a/deployment_files/R2015aSP1/installers/win32/MCR_R2015aSP1_win32_installer.exe

Directory Structure

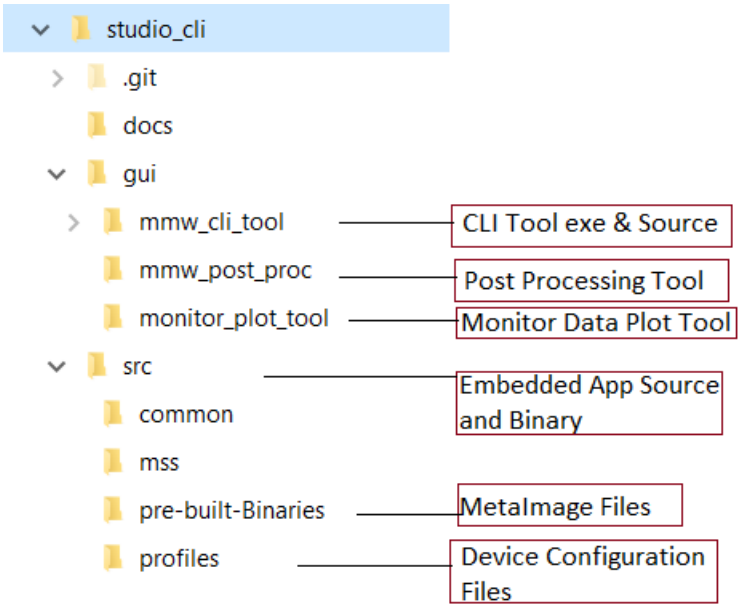

Figure 2 Directory Structure

Package contains the tool and source codes. Directory structure is broadly divided in GUI and SRC, where GUI content is for PC end and SRC for mmWave Device.

Within the GUI, it contains mmWave CLI, Post-processing and Monitor Report plotting tools.

Steps to run

1. Flash the binary file from 'src\pre-built-Binaries' to mmwave sensor device.
2. Boot the device in functional mode.
3. Connect DCA1000EVM with mmWave Sensor EVM/board and connect USB & Ethernet cables to PC.
    a. If you are using DCA1000EVM first time them, refer DCA1000_Quick_Start_Guide.pdf from mmWave Studio for details steps.
4. Get the COM port number which is used for flashing the binary.
    a. In case of xWR1843BOOST, it is enumerated as 'Application/User' COM port.
5. mmwave_studio_cli.exe which is available at 'gui\mmw_cli_tool' directory reads mmwaveconfig.txt for miscellaneous inputs
    a. mmwaveconfig.txt file must be at same location as this exe.
    b. Update the input parameters in this txt file as required, refer table-1 for description of each input parameters.

List of parameters to change in mmwaveconfig.txt for each type device variants.
For xWR18xx: MMWAVE_DEVICE_VARIANT=AWR1843 (or IWR1843) CONFIG_FILE_FORMAT=1 CONFIG_JSON_CFG_PATH=..\..\src\profiles\profile_monitor_xwr18xx.cfg DCA_LVDS_LANE_MODE=2
xWR68xx: MMWAVE_DEVICE_VARIANT=AWR6843 (or IWR6843) CONFIG_FILE_FORMAT=1 CONFIG_JSON_CFG_PATH=..\..\src\profiles\profile_monitor_xwr68xx.cfg DCA_LVDS_LANE_MODE=2
xWR16xx: MMWAVE_DEVICE_VARIANT=AWR1642 (or IWR1642) CONFIG_FILE_FORMAT=1 CONFIG_JSON_CFG_PATH=..\..\src\profiles\profile_monitor_xwr16xx.cfg DCA_LVDS_LANE_MODE=2

6. After setting right values to input parameters in mmaveconfig.txt, execute mmwave_studio_cli.exe from command prompt.
7. At any of point of error, mmWave Sensor Device returns error code over UART. Refer Table-2 for error code mapping.
8. After the entire task is done by this exe, input 'quit' command to this tool to terminate the exe. This command will first close the Monitor (if enable) JSON report files and then terminate the exe.

mmWaveConfig.txt Input Parameters

mmWave Studio CLI tool uses mmwaveconfig.txt file to read all the defined task. User can change this file as required to select different type of task for this tool.

Following table explains all the input parameters of mmWaveConfig.txt file.

Input Parameters	Description
MMWAVE_DEVICE_VARIANT	Supported options: xWR1443, xWR1642, xWR1843, xWR6843, AWR2243 Note: For AWR2243, tool uses DCA1000EVM or DevPack based SPI interface to communicate to mmwave sensor.
COM_PORT_NUM	Tool uses UART interface for xWR1443, xWR1642, xWR1843, xWR6843 devices. • This is CLI COM port number which is also being used for flashing binary to the device. • In case of xWR1843BOOST, it is enumerated as 'Application/User' COM port in device manager. • Baud rate is 921600 to communicate over UART to mmwave sensor.
ENABLE_DCA_CAPTURE	Enable/Disable capture Raw ADC data using DCA1000: 1- Enable, 0-Disable
ENABLE_CONFIG_MMWAVE	Enable/Disable configure the mmWave sensor: 1- Enable, 0-Disable
ENABLE_MONITOR_CAPTURE	Enable/Disable capture Monitoring report received from mmwave sensor to JSON files: 1- Enable, 0-Disable Note: make sure to give correct directory path to 'MONITORING_JSON_PATH'
ENABLE_POSTPROC	Enable/Disable post process the captured ADC data from DCA1000EVM: 1- Enable, 0-Disable Post processing tool path is give in 'POST_PROC_EXE_PATH' which is default set to '..\mmw_post_proc\mmwave_postproc.exe' (available within this package)
CONFIG_FILE_FORMAT	Type of mmwave config file, it can be cfg or JSON file format. 1-cfg, 0-json Reference CFG file is provided with the package at 'src\profiles\' directory. JSON File can be generated from Sensing Estimator

	or mmWave Studio tool based on device selection.
CONFIG_JSON_CFG_PATH	Full Path for mmWave Config JSON or .cfg file Note: make sure that this file type must match according to CONFIG_FILE_FORMAT
MONITORING_JSON_PATH	Path to store the monitoring report received from device in JSON file format. Each report will be stored in separate JSON file. Note: • At each new tool execution, new directory will be created within this given path based on current timestamp to avoid any overwrite. • Tool adds start JSON header at beginning of execution and end JSON header at the exit time of execution, so you would get JSON report data visible only after you terminate the tool or its execution (CLI CMD: monCaptureStop).
NUMBER_OF_MONITOR_REPORT_STORE	Number of monitoring report to store in JSON file. Set to zero for infinite time till termination (CLI CMD: "quit" or CLI CMD : "monCaptureStop")
CAPTURED_ADC_DATA_PATH	Path to store Captured ADC data binary file from DCA1000EVM.
DCA_FILE_PREFIX	File prefix for each ADC data DCA1000 captures. e.g. with 'adc_data' prefix, file name would be adc_data_Raw_0.bin
DCA_MAX_REC_FILE_SIZE_MB	Max size of each ADC data file DCA1000 captures, at this size limit it splits captured data into next binary file.
DCA_DATA_FORMAT_MODE	ADC data bit format. 1: 12Bit, 2: 14Bit, 3: 16Bit mode Note: make sure this format matches with configuration provided in cfg/JSON file. e.g. If device is configured for 16bit ADC format then set this to '3'.
DCA_LVDS_LANE_MODE	Number of LVDS lane over which DCA1000 will capture the ADC data. Valid value: 2 or 4 AWR2243 - 4 lanes, xR1642/1843/6843 - 2 lanes
POST_PROC_EXE_PATH	Post Processing Tool path which is fixed for this tool as this exe is provided within this package.

Table 2 mmWaveConfig.txt Input Parameters

Following table explains error codes for this tool.

Error Value	Explanation
-50	Invalid CLI Command sent to mmwave sensor over UART. Command is not supported by the device's application.
-51	Invalid usage of CLI command, wrong number of parameters in the command.
-52	Invalid input parameter in the CLI command.
-53	Sensor Start reconfiguration is not supported by the application.
-54	Sensor already started or stopped and got the same command again.
-55	LVDS Software session and HSI header is not supported.
-56	Execution time out in the application
-57	Error in DataPath Configuration
-58	Command is not supported while frame is running
Other values	Refer mmWaveLink.h or ICD for the other error codes.

Table 3 mmWave Sensor Error Code

Configuration Command Detail

Tool uses *.cfg file from 'src\profiles' directory and sends each configuration command to device over serial COM port. Most of these commands used in this application are similar to mmWave SDK except few newly added commands for Monitoring feature. Please refer mmave SDK user guide for existing configuration command details. New or updated CLI commands are explained here

Configuration Command	Command details	Command Parameters
calibMonCfg	The values in this command are to control the monitoring and calibration duration. This is a mandatory command.	<calibMonTimeUnit> FTTI duration for monitoring. 0: periodic Calibration and monitoring is DISABLED, else valid value (>0) to enable. <calibPeriodicity> Calibration periodicity calibPeriodicity = 0: to disable periodic calibration, value (>0) to set the calibration period. <periodicCalibEnMask> optional field to enable Mask for Periodic calibration.
monCalibReportCfg	This command is to control monitor/calibration reporting mode as application sends over UART.	<enable calib report> Enable the periodic calibration reporting <enable failure report>

	By default application sets monitor reportMode to '2' (Report is sent every monitoring period with threshold check) but user can change that in rfmonitor_configdefaults.c	Set the monitor report mode. [0]: Quit mode, don't send reports over UART. [1]: Reports are sent over UART as it is received from BSS. [n>1]: report is sent over UART at every Nth frame interval. <reserved>
txPowerMonCfg	This command is to control TX power monitor.	<enable> Enable Tx Power Monitor <TxAnt> Index of Tx for which this monitor is enabled. <profileIdx> Profile ID which will be used for this monitor.
txBallbreakMonCfg	This command is to control TX Ballbreak monitor.	<enable> Enable TX Ballbreak Monitor <TxAnt> Index of Tx for which this monitor is enabled.
rxGainPhaseMonCfg	This command is to control RX Gain Phase Monitor	<enable> Enable RX Gain Phase Monitor <profileIdx> Profile ID which will be used for this monitor.
tempMonCfg	This command is to control Temperature Monitor	<enable> Enable Temperature Monitor <tempDiffThresh> Temperature Difference Threshold.
synthFreqMonCfg	This command is to control Synthesizer Frequency Monitor	<enable> Enable Synth Frequency Monitor <profileIdx> Profile ID which will be used for this monitor.
pllConVoltMonCfg	This command is to control APLL and Synthesizer's control voltage signals Monitor	<enable> Enable PLL Voltage Monitor
dualClkCompMonCfg	This command is to control Dual Clock Compare Monitor	<enable> Enable DCC Monitor
rxIfStageMonCfg	This command is to control RX IF Filter Attenuation Monitor	<enable> Enable RX IF Stage Monitor <profileIdx> Profile ID which will be used for this monitor.
extAnaSigMonCfg	This command is to control External DC signals Monitor	<enable> Enable External Analog Signal Monitor
pmClkSigMonCfg	This command is to control Power Management, Clock generation and LO distribution circuits' Internal Analog Signals Monitor. It requires FMCW_SYNC signal (in/out) to monitor.	<enable> Enable PM Clock Signal Monitor <profileIdx> Profile ID which will be used for this monitor.

rxIntAnaSigMonCfg	This command is to control RX Internal Analog Signals Monitor.	<enable> Enable RX Internal Analog Signal Monitor <profileIdx> Profile ID which will be used for this monitor.
gpadcSigMonCfg	This command is to control GPADC Internal Analog Signals Monitor	<enable> Enable GPADC Signal Monitor

NOTE: The entire Monitor threshold and other parameters are set within the application (rfmonitor_configdefaults.c), user can change required parameter and rebuild the application.

Tool also can use JSON file generated from SensingEstimator or mmWave Studio, which internally converted to *.cfg format and sent to device over UART. Change 'CONFIG_FILE_FORMAT' input value in mmwaveconfig.txt file.

Tool's CLI Command

Tool supports few CLI commands which user can feed to control different functionalities. Each of these commands are case in-sensitive.

CLI Command	Command Detail	Usage, Precaution & Limitation
quit	Terminate the command line session. This command gracefully close all the internal task or modules.	It is recommended using this command to close the CLI tool.
sensorstop	Stop the frame at any given time. This command sends 'sensorStop' CLI command over UART to device to stop the frame.	It may not print 'done' after this command is being sent, so user needs to ignore this behavior.
sensorstart	Start the frame at any given time. This command sends 'sensorStart' CLI command over UART to device to trigger the frame.	It may not print 'done' after this command is being sent, so user needs to ignore this behavior.
moncapstop	Stop the monitor report capture.	At this command only JSON files will get updated on the disk. User needs to give either 'monCapStop' or 'quit' command so Monitor JSON files reflected on disk else files' content won't be visible.
moncapstart	Start the monitor report capture.	At this command tool creates new timestamp directory where it stores the monitor JSON files.
dcastop	To stop the DCA1000 capture, this further invokes to post-processing task.	At this command ADC data being written to file completely. If limited no. of frames are requested then after last frame

		over event, DCA will internally update the ADC file from its local buffer. So with small no. of frames, user doesn't need to provide dcaStop command.
dcastart	To trigger DCA1000 capture	NOT Supported By default DCA will start capturing the data as sensorStart CLI CMD is sent to device.

Monitor Report Capture & Plot

mmWave Studio CLI tool captures the monitor report received from device and stores it in JSON file format for each type of monitor. Make sure that ENABLE_MONITOR_CAPTURE is set to '1' in mmwaveconfig.txt. This tool buffers all JSON monitor data and finally stores to JSON file once user provides either 'monCapStop' or 'quit' command in CLI tool screen.

There is a separate tool provided in this package to plot the captured JSON monitor files. This tool is based on Python script and available at '\gui\monitor_plot_tool' path.

Pre-requisite of this tool

- Python v3.8
- SciPhy
- MatPlotLib
- Numpy

Steps to execute this tool

1. Install the Python and other dependent libraries.
2. First capture the Monitor report (JSON) by running mmWave Studio CLI exe with mmwave Sensor.
3. After Monitor JSON files are available with you at known path, data can be plotted by running following command in the CMD screen
    a. plots_gui.py [<path_to_JSPN_reports>]
4. It opens a GUI with buttons for all the available Monitor types. Click on required monitor type to open the plot screen for that monitor type.
5. Plot screen shows x/y axis values on bottom right of the screen on mouse hover. This provides the accurate values on any plot screen.

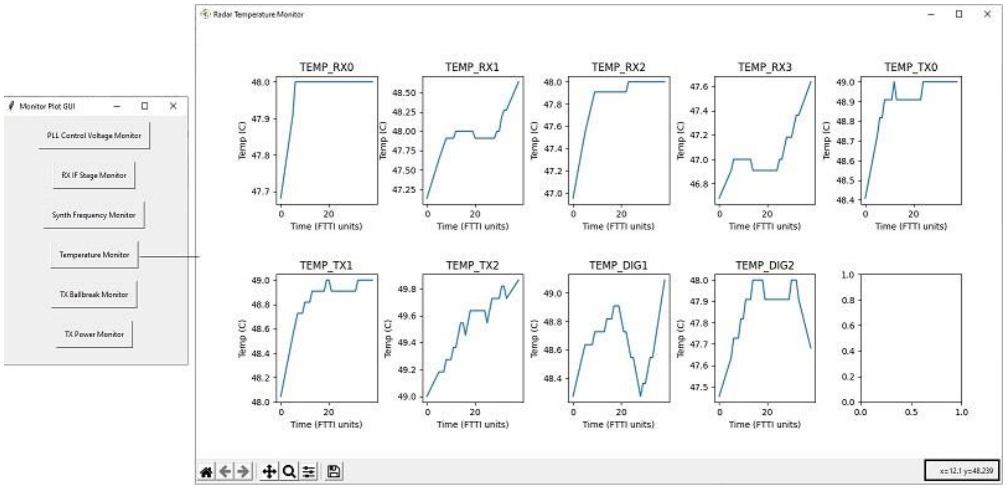

Figure 3 Monitor Report Plot

Rebuild Application & Tool

Steps to rebuild the mmWave Sensor application for different device variants

1. Using CCS:
    a. CCS project files (studio_cli_xwr<devType>.projectspec) are provided for each type of device variant.
    b. Import the CCS project and compile it for specific device variant.
2. Using Makefile:
    a. Open command prompt to 'C:\ti\mmwave_sdk_03_05_xx_xx\packages\scripts\windows' path.
        i. Make sure you have installed mmWave SDK 3.5 and other dependency tools.
    b. Set required device type in setenv.bat file.

```
@REM Select your device. Options (case sensitive) are: awr14xx, iwr14xx, awr16xx, iwr16xx, awr18xx, iwr18xx, awr68xx, iwr68xx
set MMWAVE_SDK_DEVICE=awr18xx
```

    c. Run setenv.bat in command prompt which should set environment variables w.r.t. mmWave SDK 3.5
    d. In the same command prompt screen, move to the 'src' directory of mmWave Studio CLI project.
    e. Execute 'gmake all' command to compile the application.
        i. After successful compilation, it generates '<deviceType>_mmw_demo.bin' which can be flashed to the device. This metaImage binary file contains only MSS and BSS images (no DSS).

Steps to rebuild the mmWave Studio CLI tool

1. Pre-requisite to re-build mmw_cli_tool poject
    a. Visual Studio 2017
    b. Latest mmWave DFP
2. Open mmwave_studio_cli.sln file (gui\mmw_cli_tool) in Visual Studio
3. Rebuild the project.

Studio CLI with AWR2243

Steps to run this tool with AWR2243 ES2.0 device

1. This tool doesn't contain the entire firmware version within it, so it is recommend flashing the firmware to the device beforehand.
2. For AWR2243 ONLY JSON file format is supported.
    a. Set 'CONFIG_FILE_FORMAT=0' and 'CONFIG_JSON_CFG_PATH' to correct JSON file path in mmwaveconfig.txt
3. AWR2243 supports max 4 LVDS lanes. So user can set LVDS lane option for DCA1000. This setting must match with the device LANE configuration.
   'DCA_LVDS_LANE_MODE' in mmwaveconfig.txt while using with AWR2243.
4. Connect AWR2243 BOOST EVM with DCA1000EVM and execute mmwave_studio_cli.exe

List of Parameters need to be updated on given mmwaveconfig.txt file for AWR1243/2243 device.
MMWAVE_DEVICE_VARIANT=AWR2243
CONFIG_FILE_FORMAT=0
CONFIG_JSON_CFG_PATH=..\..\src\profiles\22xx_monitor.mmwave.json
DCA_LVDS_LANE_MODE=4
FW_DOWNLOAD_DISABLE=1

mmWave Device Application

Device application contains only MSS not DSS. So metaImage binary contains MSS and BSS images.

Here is high level application's flow diagram

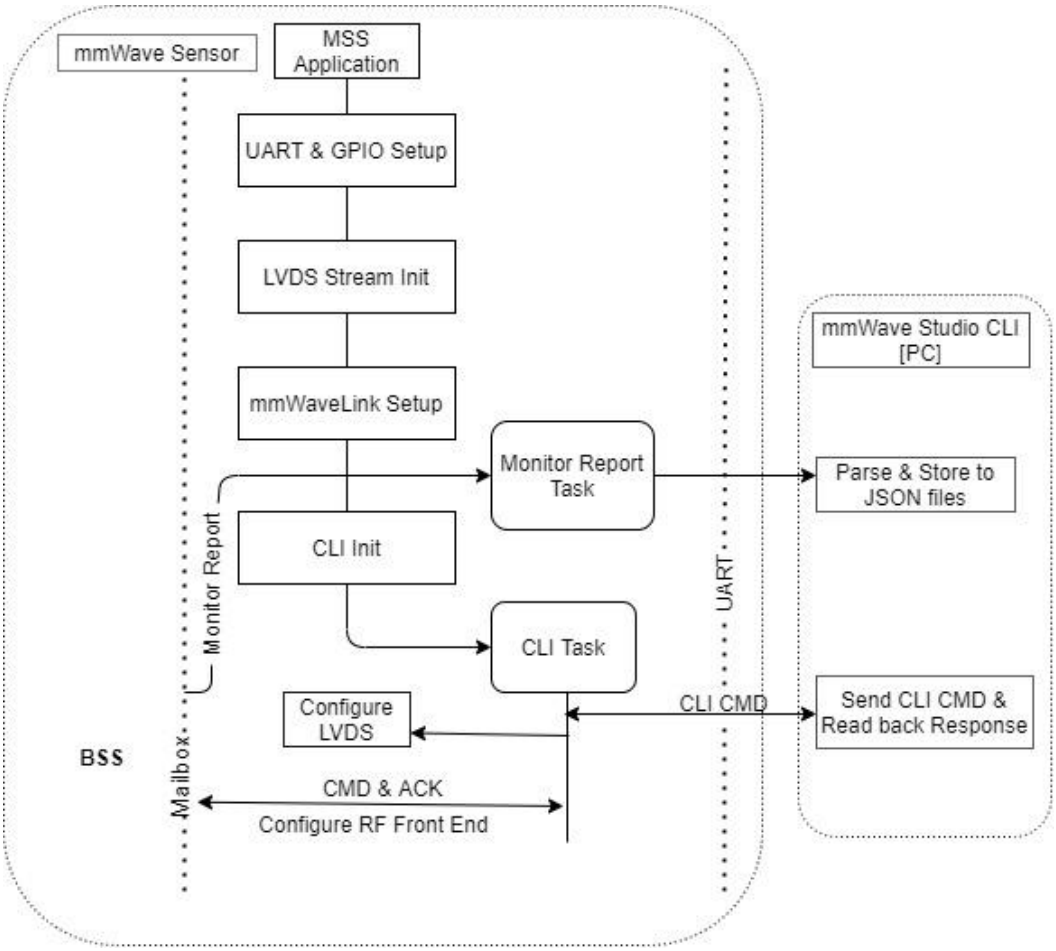

Figure 4 Application Flow

Application sends monitor report over same UART in a fixed data format at every FTTI. Following table explains byte detail of that data structure which application sends over UART to PC tool.

Header	4 Bytes 0x1234 5678
version	4 Bytes [32:24] Major, [23:16] Minor, [15:8] bugFix, [7:0] build version
totalPacketLen	4 Bytes total packet length
frameNumber	4 Bytes current frame count
platform	2 Bytes 0xFFFF [DevID] DevID: A1843, A1642, A6843
numTlv	2 Bytes total no. of TLV in this packet
TLV format: TLV Type	 2 Bytes

	TLV Type
TLV length	2 Bytes Current TLV Length
TLV Data	[TLV_Len]