MMWAVE SDK Release Notes

Product Release 2.1.0

Release Date: Oct 5, 2018

Release Notes Version: 1.0

CONTENTS

1. Introduction

The mmWave SDK enables the development of millimeter wave (mmWave) radar applications using TI mmWave sensors (see list of supported Platform/Devices). The SDK provides foundational components which will facilitate end users to focus on their applications. In addition, it provides few demo applications which will serve as a guide for integrating the SDK into end-user mmWave application.

Key mmWave SDK features:

- Building blocks
  - Full driver availability
  - Layered approach to programming analog front end
  - Catalog of mmwave algorithms optimized for C674x DSPs
- Demonstrations and examples
  - TI RTOS based
  - Out of box demo with easy configurability via TI cloud based GUI
  - Representation of "point cloud" and benchmarking data from demo via GUI
  - Profiles tuned to common end user scenarios such as Range, Range resolution, Velocity, Velocity resolution
- Documentation

mmWave SDK works along with the following external tools:

- Host tools including Pin Mux, Flashing utilities
- Code Composer Studio™ IDE for RTOS development

2. Release overview

2. 1. What is new

- Support for production devices mentioned in the "Platform and Device Support" section below
- Bug fixes

More details can be found in NewFeatures section.

2. 2. Platform and Device Support

The devices and platforms supported with this release include:

Supported Devices	Supported EVM
AWR1443 ES3.0	AWR1443BOOST - AWR1443 Evaluation Module RevB/RevA
AWR1642 ES2.0	AWR1642BOOST - AWR1642 Evaluation Module RevB
AWR1642_HS ES 2.0 *	
IWR1443 ES3.0	IWR1443BOOST - IWR1443 Evaluation Module RevB/RevA
IWR1642 ES2.0	IWR1642BOOST - IWR1642 Evaluation Module RevB
IWR1642_HS ES 2.0*	

* xWR16xx HS devices need additional MMWAVE-SECDEV package

xWR14xx terminology is used in sections that are common for AWR14xx and IWR14xx

xWR16xx terminology is used in sections that are common for AWR16xx and IWR16xx

xWR14xx ES2.0 is not supported in this release. Prebuilt binaries in this release will not work on ES2.0 silicon.

xWR16xx ES1.0 is not supported in this release. Prebuilt binaries in this release will not work on ES1.0 silicon

This release of mmWave SDK supports the foundation components for the devices mentioned in the table above . At system level, the mmWave SOC/EVM may interface with other TI ecosystem SOCs/Launchpads/EVMs and software for these other devices will not be a part of the mmWave SDK foundation components.

2. 3. Component versions

Components inside mmwave_sdk that have their own versions are shown below.

Component		Version	Type	Comment
mmwave sdk		2.1.0	Source and Binary	Overall package release version
RadarSS firmware (patch)		1.2.0	Binary	RadarSS firware is in ROM. Only the patch is included in the mmwave sdk release
mmWaveLink Framework		1.2.0	Source and Binary	
FTDI		2.12	Binary	
Image Creator	gen_bincrc32	1.0	Windows and Linux binary	
	out2rprc	2.0	Windows binary	Need mono to run this on Linux
	Crc multicore image	1.0	Windows and Linux binary	
	Multicore image generator	1.0	Windows and Linux binary	
	create_ConfigRPRC	1.0	Windows and Linux binary	

2. 4. Tools dependency

For building and using mmwave sdk the following tool versions are needed.

Tool	Version	Download link
CCS	7.4 or later	download link
TI SYS/BIOS	6.53.02.00	Included in mmwave sdk installer
TI ARM compiler	16.9.6.LTS	Included in mmwave sdk installer
TI CGT compiler	8.1.3	Included in mmwave sdk installer
XDC	3.50.04.43	Included in mmwave sdk installer
C64x+ DSPLIB	3.4.0.0	Included in mmwave sdk installer
C674x DSPLIB	3.4.0.0	Included in mmwave sdk installer
C674x MATHLIB (little-endian, elf /coff format)	3.1.2.1	Included in mmwave sdk installer
Mono JIT compiler	4.2.1	Only for Linux builds
mmWave Radar Device support package	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)
MMWAVE-SECDEV	2.0.0	Needed for xWR16xx high secure (HS) devices only Can be requested from link
Pinmux tool (optional)	Latest	Used to generate pinmux configuration for custom board https://dev.ti.com/pinmux (Cloud version)

Doxygen (optional)	1.8.11	Only needed if regenerating doxygen docs
Graphviz (optional)	2.36.0 (20140111.2315)	Only needed if regenerating doxygen docs

The following tools are needed at runtime

Runtime tool	Version	Link
Uniflash	Latest	Uniflash tool is used for flashing xWR1xxx devices Cloud version (Recommended): https://dev.ti.com/uniflash Offline version: http://www.ti.com/tool/uniflash
mmWave Demo Visualizer	Latest	TI Gallery APP for configuring mmWave sensors and visualizing the point cloud objects generated by the mmWave SDK demo https://dev.ti.com/mmWaveDemoVisualizer

2. 5. Licensing

Please refer to the mmwave_sdk_software_manifest.html, which outlines the licensing status for mmwave_sdk package.

3. Release content

3. 1. New Features

- mmWave Suite enhancement
  - Support for xWR14xx ES3.0 production device.
    - Increased L3 memory
    - Support for RadarSS patch binary
    - Updated flash image format to match xWR16xx
  - Drivers
    - mmWaveLink updated to support the RadarSS firmware version as noted above
    - mmWave API: Add support to bypass and restore boot-time device calibration from user
- mmWave Demos enhancement
  - Enhanced mmWave Demo Visualizer to allow users control dynamic settings such CFAR threshold, peak grouping and clutter removal in real time
  - Added advanced mode in mmWave Demo Visualizer to send dynamic CLI commands in real time

3. 2. Migration section

This section describes the changes that are relevant for users migrating to the mmWave SDK 2.1.0 release from 2.0.0 release. See release notes archive in the SDK release package for migrating from other older releases.

Summary	Component/s	Subcomponent	Behavior of Impact
Visualizer: Moved Object detection knobs from Configure tab to Plots tab to enable real time tuning	Demos	GUI/Visualizer	The knobs for CFAR threshold, peak grouping and clutter removal have been moved to "Real time tuning" within the plots tab to allow real time tuning of these features. When using the configure tab, Visualizer now will send default value for these features and user can tune it from plots tab while observing immediate feedback in the plots. The knob for Range/Angle Bias compensation config has been moved to "Advanced Commands" within the plots tab.
xWR14xx mmW demo: lowPower command	Demos	CLI	lowpower command is now mandatory in mmw demo CLI configuration for xWR14xx device. Refer to ti\demo\xwr14xx\mmw\profiles\profile_2d.cfg for example. mmWaveDemoVisualizer will automatically append this command in all generated configs. It will also throw an error if this command is found missing in the user provided config. When using the perl script mmwDemo_xwr14xx_update_config.pl to automatically update the existing 1.2 based SDK profiles, it will throw an error if the "lowPower" command is not present.
Remove generateBin step from makefile and incorporate it into generateMetaImage	Build	Demo makefiles, generateMetaImage script	All currently supported devices only support metaimage file for flashing (this is the combined MSS/RadarSS/DSS binary). Earlier (due to legacy devices) there was a separate binary generation step for each out file. The creation of flashable binary is now simplified since the generateBin step does not need to be called for MSS and DSS out files and it is handled inside the generateMetaImage step. Please follow the example in the mmw demo makefile for your device and incorporate similar change in user application makefile that may be based on the mmw example. See mmWave SDK user guide for more details.
xWR1443 ES3: Add Out2rprc image generation by combining the MSS and BSS patch image (similar to xWR16xx)	Build, Uniflash	Demo makefiles, generateMetaImage script	For xwr14xx 3.0 only meta image format binary is supported (similar to xwr16xx). All the tools used to create meta image are common for xwr14xx and xwr16xx (and are under ImageCreator folder). The binary generation step is handled by makefile/scripts and is transparent to user. Uniflash usage for xwr14xx is different since it takes only the meta image as input now (for older version of xwr14xx the BSS and MSS files were specified separately in uniflash). Select the generated metaimage file for "Meta Image 1/BSS" option in Uniflash
xWR1443 ES3: Remove the requirement for alignment from the platform linker command file	Build	xWR14xx platform linker file	Explicit alignment of section to 8 or 16 byte boundary is not needed for xWR14xx ES 3.0 and xWR16xx ES 2.0 and such alignment can be removed from the R4F linker cmd file.

<table>
<tr><td>Linker warning regarding "file cannot be loaded and run on a target system" should be considered error</td><td>Build</td><td>C674x Linker flags</td><td>linker warning 10015 will be considered as error and if seen in a give build, user will need to address the issue to be able to build successfully. If left un-noticed, this warning results in a out file that can demonstrate unpredictable behavior and hence the linker flags have been adjusted to convert this warning to error.</td></tr>
<tr><td>Upgrade QSPI flash driver to handle at runtime the different flash parts populated on TI EVMs</td><td>Drivers</td><td>QSPI Flash</td><td>- QSPI driver is enhanced to handle two different flash parts - spansion and macronix. It queries the flash for the ID and other details and appropriately loads the handler for that particular flash part.
There are no changes to be done by the user to enable this feature.

- QSPIFlash_configMmapRead() is updated with supporting for 4 read modes.
For specific reading modes such as FLASH_SINGLE_MODE, FLASH_DUAL_MODE , FLASH_QUAD_MODE, Flash memory is mapped with corresponding reading configurations.
For FLASH_AUTO_MODE, flash memory is mapped with the first mode supported by the flash device in the order of FLASH_QUAD_MODE, FLASH_DUAL_MODE, FLASH_SINGLE_MODE.</td></tr>
<tr><td>XWR14xx ES3.0: enable WDT(RTIB) on init</td><td>Drivers</td><td>Watchdog</td><td>This is handled internally by the Watchdog driver and no other change is needed by the application</td></tr>
<tr><td>CSI2: DataRate enum names imply input clock selection and not actual datarate seen on the CSI2 O/P lanes</td><td>Drivers</td><td>CSI</td><td>CSI_DataRate enumeration which describes the data rates supported by the CSI interface have changed in this release. Old definitions seemed to imply data rate when it was really the HSClk rate which is twice the data rate (DDR mode).
The mapping from old values to new values is given below

<table><tr><td>Defines in SDK 2.0/older releases
(this was actually used internally as HSICLK_xxxMhz)</td><td>Equivalent defines in SDK 2.1 /newer releases
This is true datarate=<old_define>/2</td></tr><tr><td>CSI_DataRate_1200Mhz</td><td>CSI_DataRate_600Mbps</td></tr><tr><td>CSI_DataRate_900Mhz</td><td>CSI_DataRate_450Mbps</td></tr><tr><td>CSI_DataRate_600Mhz</td><td>CSI_DataRate_300Mbps</td></tr><tr><td>CSI_DataRate_450Mhz</td><td>CSI_DataRate_225Mbps</td></tr><tr><td>CSI_DataRate_400Mhz</td><td><invalid/not supported></td></tr><tr><td>CSI_DataRate_300Mhz</td><td>CSI_DataRate_150Mbps</td></tr><tr><td>CSI_DataRate_225Mhz</td><td><invalid/not supported></td></tr><tr><td><No definition></td><td>CSI_DataRate_400Mbps</td></tr></table>
Old definitions in user code will result in compilation error and user should use the above mapping table to find the new definition corresponding to the old one and use that in their code.</td></tr>
</table>

mmWave API: Add support to bypass and restore boot-time device calibration from user	Control	mmWave API	This is a new feature. In order to enable an application to save /restore calibration: - Reading/Writing to the Flash for calibration data is outside the scope of the MMWave API. - Extend the input structure MMWave_OpenCfg of MMWave_open API to add the following new fields: -- Custom Calibration Enable Mask: If this is set to a non-zero value, mmWave module will use the provided calibration mask to configure the RF. In the existing SDK demos and applications this field will be 0; so there should be no change in behavior. -- [Optional]Pointer to the Calibration Data: If this is set to NULL, mmWave module will not restore the calibration data. If this is set to Non-NULL, mmWave module will use this and restore the calibration data in the BSS using the mmWave Link API “rlRfCalibDataRestore” - In order to get the calibration data the application would need to invoke the MMWave Link API “rlRfCalibDataStore” after the MMWave_open has been invoked. There will be no extension added to the existing MMWave API to get this information.
DFP 1.2 integration: Changes in names for "reservedxx" fields	control	mmWavelink	Reserved fields in many APIs were changed from 'reserved0' to 'reserved' and so on. Since the reserved fields should not be accessed within the calling code, the impact of this change should be minimal to none
DFP 1.2 integration: Change in storage types for some existing fields	control	mmWavelink	- rlRfRunTimeCalibReport_t->temperature field changed from unit16 to int16 - rlChirpCfg_t->freqSlopeVar field changed from int16 to unit16 - rlTempMonConf_t->tempDiffThresh field changed from int16 to unit16 - rlTxGainPhaseMismatchMonConf_t->txPhaseMismatchThresh field changed from int16 to unit16
DFP 1.2 integration: Daisy chain mode for cascade feature	control	mmWavelink	Added new parameters in rlChanCfg_t API to support daisy chain mode of cascading the AWR devices. This should be ignored for single chip mode
DFP 1.2 integration: Optimize Dynamic chirp configuration	control	mmWavelink	Added new option in rlSetDynChirpCfg API which will configure 48 chirps dynamically in one call. This option speedup the dynamic chirp configuration.
DFP 1.2 integration: LDO short circuit monitor	control	mmWavelink	Added new parameter ldoScEn in rlMonAnaEnables_t to enable /disable LDO short circuit monitor.
DFP 1.2 integration: 20GHz sync-in monitoring	control	mmWavelink	This is applicable only on devices which support cascading. - Updated rlMonPmclkloIntAnaSigRep structure to include 20 GHz sync in monitoring results. - Updated rlPmClkLoIntAnaSignalsMonConf structure to enable 20 GHz sync monitoring.
DFP 1.2 integration: PA LDO knob for simultaneous 3TX usecases	control	mmWavelink	Added a new option in rlRfLdoBypassCfg API which will disable the PA LDO in the LDO bypass mode. In this mode, the VIN_13RF2 should be shorted to VOUT_PA on board. This is needed specifically in simultaneous 3 TX use cases to improve the package reliability.
DFP 1.2 integration: new feature: TX phase shift calibration	control	mmWavelink	These APIs are applicable only on those devices which have support for TX phase shifter. - Defined a new bit for enabling/disabling TX phase shifter calibration in rlRfInitCalConf_t API. This calibration can be enabled /disabled on devices which have support for TX phase shifter. - Defined a new bit for TX phase shifter calibration status in rlRfInitComplete_t async event. The status bit will be set only on successful completion of TX phase shifter calibration on devices which support TX phase shifter. On other devices this status bit should be ignored. - Added 2 new APIs, rlRfPhShiftCalDataStore and rlRfPhShiftCalDataRestore for storing and restoring phase shifter calibration values. - Updated rlMonTxBpmRep structure to include the TX phase shifter monitoring results. - Added new parameters in rlTxBpmConf structure to enable phase shifter monitoring configuration.
DFP 1.2 integration: new feature: PD calibration	control	mmWavelink, mmwave	Added PD calibration in runtime calibration structure rlRunTimeCalibConf_t. mmWave API - MMWave_start enables this by default. No action from user is requried when using the MMWave_start API.

DFP 1.2 integration: New structure for BSS ESM fault	control	mmWavelink	Added new rlBssEsmFault_t structure for BSS ESM fault report.
DFP 1.2 integration: TX calibration enhancement	control, Demos	mmWavelink, mmW Demo CLI handler	-Added a new parameter txCalibEnCfg in rlProfileCfg_t API. This parameter indicates to the firmware if more than one transmitter is to be turned on during the TX power calibration. If the same number of transmitters are turned on during TX power calibration as in functional use-case, then it will improve the output power accuracy by about 1 dB. Application needs to set this parameter to the correct value before calling the rlProfileCfg API. - mmW demo CLI handler for profileCfg hardcodes this field to 0 for backward compatibility reasons. User can update this CLI handler to enable this feature
DFP 1.2 integration: New API for retrieving Die id	control, Demos	mmWavelink, mmW Demo CLI output	- Added a new mmwavelink API for reading device Die ID status. - mmW demo version command uses this API to provide yje die ID information
Expose shared memory allocation for L3/ MSS TCMA/MSS TCMB from build env	Build, Demos, ccsdebug	Makefiles	Added a new optional environment build variable MMWAVE_SDK_SHMEM_ALLOC to influence the shmem settings of demo and ccsdebug flashable images. See mmWave SDK User Guide for more details. User doesn't need to explicitly set this build variable - there are defaults in mmwave_sdk_<platform>.mak file that should work for demos from older SDK releases as well. This option should be exercised by a user if they need to divert some of the shared memory from L3 to MSS TCMA/TCMB.

3. 3. Issues fixed

The following issues from previous releases were fixed in this release

Issue Type	Key	Summary
Bug	MMWSDK-1469	Linker warning regarding "file cannot be loaded and run on a target system" should be considered error
Bug	MMWSDK-1460	Cleanup EDMA low level define for number of edma channels in device
Bug	MMWSDK-1459	Bug fix related to type conversion in mmwavelib_maxpow
Bug	MMWSDK-1458	HWA: Update window start documentation in paramset structure
Bug	MMWSDK-1449	HWA silicon issue workaround: set the reserved fields to 0 in paramset registers
Bug	MMWSDK-1448	Near range phase error estimation seems incorrect in mmw doxygen
Bug	MMWSDK-1425	DC Compensation(antenna coupling signature removal) is broken for negative indices
Bug	MMWSDK-1423	CSI2: DataRate enum names imply input clock selection and not actual datarate seen on the CSI2 O/P lanes
Bug	MMWSDK-1417	xWR16xx demo: DC range calibration buffer contents may be lost in case of advanced frame
Bug	MMWSDK-1416	Visualizer formula for non-coherent combining loss is grossly in error for 12 virtual antennas (14xx case)
Bug	MMWSDK-1414	Visualizer formula for max range for desired RCS is in error

Bug	MMWSDK-1413	IWR1443BOOST: DemoVisualizer: Wrong azimuth resolution for 1Tx2Rx
Bug	MMWSDK-1410	Watchdog unit test: Watchdog reset test mode doesnt warm reset the device
Bug	MMWSDK-1409	Windows checkenv.bat script corrupts PATH environment variable
Bug	MMWSDK-1406	User guide explanation of frameCfg needs to say maximum, not minimum 50 % duty cycle
Bug	MMWSDK-1400	xWR16xx demo: missing subframeindex arg in the help for clutterRemoval and nearFieldCfg
Bug	MMWSDK-1393	use case of 2048 range bins (with 1 or 2 rx antennas) is not working in oob demo
Bug	MMWSDK-1391	xWR14xx OOB demo crashes when peak grouping is disabled
Bug	MMWSDK-1389	HWA_close: calling sequence of reset and disable need to be reversed
Bug	MMWSDK-1388	Incorrect Log 2 and Abs computation in radarlib_log2Abs16
Bug	MMWSDK-1385	CANFD and CAN has same structure name and can't be used together

Known Issues

The following issues are known at the time of this release.

Issue Type	Key	Summary	Comments
Bug	MMWSDK-1478	Intermittent failure in "monitoring results" for mmwavelink unit test for awr16xx	This issue is seen in noisy lab environment only. One out of many reports for noise figure has failure status. Observed noise figure from that report are logged at the end of the test run and can be used for debugging further, in case this is seen in other scenarios.
Bug	MMWSDK-1078	Limitation in mmW demo + LVDS instrumentation use case	See limitations section below
Task	MMWSDK-533	GUI of mmw demo running slow from Firefox browser	Workaround: Please switch to Chrome browser.
Story	MMWSDK-319	CAN driver: DMA mode is not supported	
Story	MMWSDK-252	UART driver has not tested for Data Length 5 and 6	
Bug	MMWSDK-1157	Rare failure seen in UART loopback driver unit test - HW limitation	

3. 4. Limitations

Some of these limitations are captured in the "known issues" list shown in previous section.

1	CAN driver: • DMA and FIFO mode are not supported

2	CANFD driver: • DMA and Timestamping are not supported

3	CBUFF/CSI2/LVDS:
	• Driver does not support the following functionality:
	  • Multiple packets
	  • 3 channels
	• CSI2: ADC streaming has only been tested under 1 configuration in csi_stream usecase

4	CRC driver: "Auto" mode is not implemented.
5	DMA driver: MPU and Parity Feature not implemented.
6	EDMA driver: Privilege feature not implemented.
7	HWA driver: Any modes/algorithm outside the scope of mmWave demo are not tested (however they are implemented in the driver).
8	I2C driver: Verified only loopback mode on mmWave device TI EVM (however all features are implemented in the driver).
9	QSPI/QSPI Flash driver: ▪ dual-Read/Quad read in configuration mode is not supported ▪ setting write protections bits is not supported
10	SPI (MIBSPI) Limitations: • For xWR14xx, MIBSPI is only supported on SPIA, hence driver only supports SPIA. SPIB is not supported in xWR14xx. In xWR16xx, both instances are MIBSPI and are supported within the driver. • When MIBSPI mode is used in 4-pin slave mode, for every CHARLEN (8 bits or 16 bits), CS signal(from Master) has to be toggled and 2 VBUSP cycles need to be inserted. This needs to be taken care on SPI master device.
11	DMA based transactions are not supported for CRC and Mailbox driver.
12	mmW demo: See demo's doxygen page for more details.
13	mmW demo + LVDS instrumentation: ▪ This feature is not available for xWR14xx due to ADC Buffer being unavailable for streaming while datapath processing is active. ▪ For xWR16xx, CQ cannot be streamed out reliably when datapath processing is also enabled. The data corruption for CQ data over LVDS lanes is seen more pronounced when multiple chirps/chirp event is enabled

4. Test reports

Results of the unit tests can be found in the docs/test folder. The test folder has separate folders for all the SoC variants. System level test is run using demos.

5. Installation instructions

mmwave_sdk installer is available as a Windows Installer and a Linux installer.

- **mmwave_sdk_<version>-Windows-x86-Install.exe: Windows installer verified on Windows 7 and Windows 10 machines**
- **mmwave_sdk_<version>-Linux-x86-Install.bin: Linux installer verified on Ubuntu 14.04 & Ubuntu 16.04 64 bit machines.**

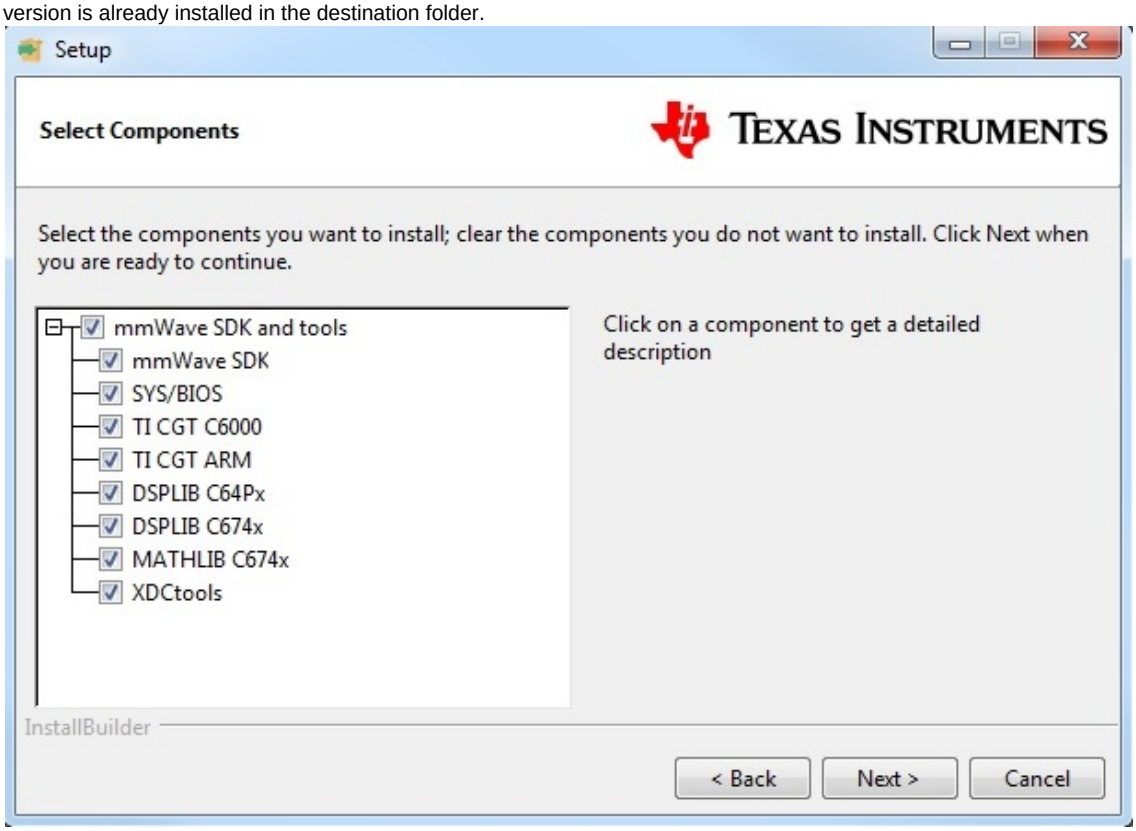
Depending on your development environment run the appropriate installer

- In Windows environment, double clicking the Windows installer from Windows explorer should start the installation process
- If in Linux environment,
  - Enable execute permission for the Linux installer by running "chmod +x mmwave_sdk_<version>-Linux-x86-Install.bin" command
  - Run the installer using "./mmwave_sdk_<version>-Linux-x86-Install.bin" command

Installation steps:

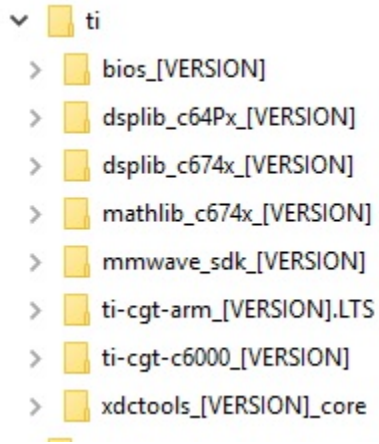
- Setup

- Choose Destination Location: Select the folder to install (default is c:\ti on windows and ~/ti on linux). The installation folder selected should not have spaces in its full path.
- Select Components: The installer includes all the tools needed for building the mmWave SDK. You should see a screen like below (except that each component will also have version information appended). The only reason to deselect a tool is if the exact tool version is already installed in the destination folder.

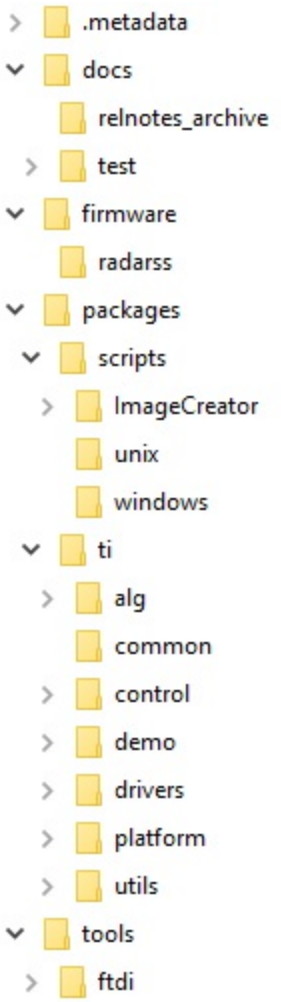


- Review installation decisions
- Ready to install
- Once installation starts all the selected components will be installed (if a component with the same version exists in the destination folder it will be overwritten)
- Installation complete

After the installation is complete the following folder structure is expected in the installation folder (except that each component will have appropriate version number in place of the VERSION placeholder shown below)



Under the mmwave_sdk <ver> folder you should have the following directory structure.

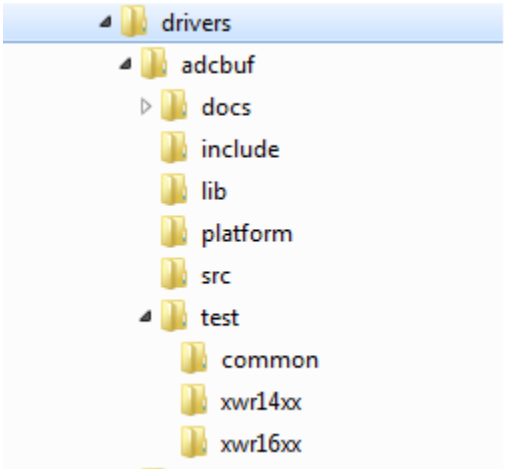


6. Package Contents

The mmwave sdk release package contains the following major components/folders.

6. 1. Drivers

Drivers can be found under mmwave_sdk_<ver>/packages/ti/drivers folder. The directory structure of all drivers is similar to the one shown below for adcbuf (some drivers do not have a unit test as shown in the table below)



- docs: Driver API documentation done with doxygen
- include: Include files
- lib: Prebuilt libraries
- platform: Platform files
- src: Driver Source files
- test/<platform>: Unit test src files and prebuilt unit test binary for that <platform: xwr14xx, xwr16xx>
- test/common: Unit test src files common for all platforms
- driver base folder has external header file, make files

Content of each driver is indicated in the table below.

Component	Source & prebuilt library	API Document (doxygen)	Unit test (source & prebuilt binary)
ADCBUF	X	X	X
CAN	X	X	X
CANFD	X	X	X
CBUFF/LVDS	X	X	X
CRC	X	X	X
CRYPTO[1]	X	X	X
CSI2	X	X	X
DMA	X	X	X
EDMA	X	X	X
ESM	X	X	
GPIO	X	X	X
HWA	X	X	X
I2C	X	X	X
MAILBOX	X	X	X
OSAL	X	X	
PINMUX	X	X	
QSPI	X	X	X
QSPIFLASH	X	X	X
SOC	X	X	
SPI	X	X	X
UART	X	X	X
WATCHDOG	X	X	X

[1] CRYPTO is only supported on xWR16xx high secure (HS) devices

6. 2. Control

Control modules can be found under mmwave_sdk_<ver>/packages/ti/control folder. Content of each of the control module is shown below

Component	Source & Prebuilt Library	API Document (doxygen)	Unittest (source & prebuilt binary)
mmwavelink framework	X	X	X

mmwave high level api	**X**	**X**	**X**

6. 3. Algorithm

Algorithms can be found under mmwave_sdk_<ver>/packages/ti/alg folder. Currently algorithms applicable for mmwave functionality are provided under this folder:

Component	Source & Prebuilt Library	API Document (doxygen)	Unittest (source & prebuilt binary)
gtrack	**X**	**X**	**X**
mmwavelib	**X**	**X**	**X**

6. 4. Usecases

Usecases can be found under mmwave_sdk_<ver>/packages/ti/drivers/test folder.

Component	Source	API Document (doxygen)	Unittest (source & prebuilt binary)
csi_stream (IWR14xx only)	**X**	**X**	**X**
dsp_edma (xWR16xx only)	**X**	**X**	**X**
hwa_edma (xWR14xx only)	**X**	**X**	**X**
mem_capture	**X**	**X**	**X**

6. 5. Demos

Demos can be found under mmwave_sdk_<ver>/packages/ti/demo/<platform>. The following demos are included in the mmwave sdk package. Details on running demos can be found in the mmwave_sdk_user_guide.

Component	Source & Prebuilt Binary	Demo document (doxygen)	Demo GUI
mmw	**X**	**X**	**X**

6. 6. Misc folders

Following folders are also part of mmwave_sdk_<ver>/packages/ti folder.

- common: Common header files needed across all components
- platform: platform specific files
- utility: Contains
  - ccs debug utility which is the MSS/DSSbinary that needs to be flashed when connecting/developing using CCS (details can be found in mmwave_sdk_user_guide)
  - cli which is the cli helper utility used by the demos
  - cycleprofiler which is the helper utility used for profiling the various components inside the SDK
  - hsiheader which is a helper utility that creates a header for the data to be shipped over LVDS lanes.
  - testlogger which is the helper utility for driver unit tests

6. 7. Scripts

Build scripts can be found in mmwave_sdk_<ver>/packages/scripts folder. Build instructions can be found in mmwave_sdk_user_guide.

6. 8. Firmware

RadarSS firmware for all supported devices is included under mmwave_sdk_<ver>/firmware/radarss folder. Procedure to flash the radarss is covered in the mmwave_sdk_user_guide.

6. 9. Tools

The following tools are included in the release in binary form. These can be found under mmwave_sdk_<ver>/tools folder.

- **Ftdi:** These Windows PC drivers are needed when interfacing to the board via MMWAVE-DEVPACK

6. 10. Docs

mmwave_sdk_<ver>/docs folder contains important documents related to the release such as

- mmwave_sdk_software_manifest.html: Software Manifest
- mmwave_sdk_release_notes.pdf: Release Notes (this document)
- mmwave_sdk_user_guide.pdf: User guide
- mmwave_sdk_module_documentation.html: Links to individual module's documentation

mmwave_sdk_<ver>/docs/relnotes_archive contains release notes from previous releases. Release notes contain migration information.

mmwave_sdk_<ver>/docs/test folder contains test results for each SoC. Each SoC folder in turn may contain multiple test group folders (such as module_test, alglib_test) which have the following files

- Report.html: Detailed Test report with links to logs
- *.log: Test logs for unit tests

7. Related documentation/links

Other than the documents included in the mmwave_sdk package the following documents/links are important references.

- SoC links:
  - AWR1443
  - AWR1642
  - IWR1443
  - IWR1642
- EVM links:
  - AWR1443BOOST
  - AWR1642BOOST
  - IWR1443BOOST
  - IWR1642BOOST
  - MMWAVE-DEVPACK