Understanding the Out of Box Demo Data Output

- An output packet is sent out every frame through the UART.
- Every packet consists of a Frame Header. Additional TLV items may also be sent each frame depending on the options enabled in the guiMonitor commmand and the number of detected objects.
- Each TLV item consists of a TLV header and value (payload) information.
- The length of the packet can depend on the number of detected objects and vary from frame to frame.
- The end of the packet is padded so that the total packet length is always multiple of 32 Bytes.

Frame Header

Length: 44 Bytes
A Frame Header is sent at the start of each packet. Use the Magic Word to find the start of each packet.

Value	Type	Bytes	Details
Magic Word	uint16_t	8	Output buffer magic word (sync word). It is initialized to {0x0102,0x0304,0x0506,0x0708}
Version	unint32_t	4	SDK Version represented as (MajorNum x 2^24 + MinorNum x 2^16 + BugfixNum x 2^8 + BuildNum)
Total Packet Length	unint32_t	4	Total packet length including frame header length in Bytes
Platform	unint32_t	4	Device type (ex 0xA6843 for IWR6843 devices)
Frame Number	unint32_t	4	Frame number (resets to 0 when device is power cycled or reset. Not when sensor stop/start is issued.)
Time [in CPU Cycles]	unint32_t	4	Time in CPU cycles when the message was created.
Num Detected Obj	unint32_t	4	Number of detected objects (points) for the frame
Num TLVs	unint32_t	4	Number of TLV items for the frame.
Subframe Number	unint32_t	4	0 if advanced subframe mode not enabled, otherwise the sub-frame number in the range 0 to (number of subframes - 1)

TLV Header

Length: 8 Bytes

- The number of TLVs in the frame packet is extracted from the Frame Header.
- For each TLV in the packet, there is a TLV Header containing Type and Length information.
    - The Type identifier indicates what kind of information is contained in the playload.
    - The Length value gives the length of the payload.

Value	Type	Bytes	Details
Type	unint32_t	4	Indicates types of message contained in payload.
Length	unint32_t	4	Length of the payload in Bytes (does not include length of the TLV header)

TLV Type Identifier

The parameters in the CLI command **guiMonitor** are used to enable or disable whether the TLV type is included in the output frame packet. The parameters are as follows: `guiMonitor <subFrameIdx> <detected objects> <log magnitude range> <noise profile> <rangeAzimuth(Elevation)HeatMap> <rangeDopplerHeatMap> <statsInfo>`

For the Out of Box demo, if type contains the following value then the payload contains the information listed under value type and should be parsed accordingly.

TYpe Identifier	Value Type	Conditions for output
1	Detected Points	`<detected objects>` is set to 1 or 2 AND there are detected objects for the frame, else this type is not sent for that frame
2	Range Profile	`<log magnitude range>` is set to 1; occurs every frame
3	Noise Floor Profile	`<noise profile>` is set to 1; occurs every frame
4	Azimuth Static Heatmap	`<rangeAzimuth(Elevation)HeatMap>` is set to 1 AND demo is not for AOP or ODS which use AOA2D; occurs every frame
5	Range-Doppler Heatmap	`<rangeDopplerHeatMap>` is set to 1; occurs every frame
6	Statistics (Performance)	`<statsInfo>` is set to 1; occurs every frame
7	Side Info for Detected Points	`<detected objects>` is set to 2 AND there are detected objects for the frame, else this type is not sent for that frame
8	Azimuth/Elevation Static Heatmap	`<rangeAzimuth(Elevation)HeatMap>` is set to 1 AND demo is for AOP or ODS which use AOA2D; occurs every frame
9	Temperature Statistics	`<statsInfo>` is set to 1; occurs every frame

TLV Payload

Detected Points

Type: 1
Length: 16 Bytes x Num Detected Obj
Value: Array of detected points. Each point is represented by 16 bytes giving position and radial Doppler velocity as shown in the table below:

Value	Type	Bytes
X [m]	float	4
Y [m]	float	4
Z [m]	float	4
doppler [m/s]	float	4

Side Info for Detected Points

Type: 7
Length: 4 Bytes x Num Detected Obj
Value: The payload consists of 4 bytes for EACH point in the point cloud. The values for snr and noise are determined as described in the doxygen documentation for the mmWave SDK Demo.

Value	Type	Bytes
snr [dB]	uint16_t	2
noise [dB]	uint16_t	2

Range Profile

Type: 2
Length: 2 Bytes x Range FFT size
Value: Array of profile points at 0th Doppler (stationary objects). The points represent the sum of log2 magnitudes of received antennas expressed in Q9 format.

Noise Profile

Type: 3
Length: 2 Bytes x Range FFT size
Value: Array of profile points at max Doppler bin. In general for stationary scene, there would be no objects or clutter at maximum speed so the range profile at such speed represents the receiver noise floor.

Azimuth Static Heatmap

Type: 4
Length: (Range FFT size) x (Number of "azimuth" virtual antennas) x (4Bytes)
Value: Samples to calculate static azimuth heatmap (no moving object signal). This is a 2D FFT array in range direction (x[numRangeBins][numVirtualAntAzim]), at doppler index 0. The antenna data are complex symbols, with imaginary first and real second in the following order:

Imag(ant 0, range 0), Real(ant 0, range 0),...,Imag(ant N-1, range 0),Real(ant N-1, range 0)
...
Imag(ant 0, range R-1), Real(ant 0, range R-1),...,Imag(ant N-1, range R-1),Real(ant N-1, range R-1)

This means the first 4 bytes of the payload is the radar cube complex value of the first range bin for the first virtual antenna(N=1). The last 4 bytes is for the last range bin (R) and the last virtual antenna (N). The values from the radar cube are used to construct the range-azimuth heatmap in the visualizer.

Azimuth/Elevation Static Heatmap

Type: 8
Length: (Range FFT size) x (Number of all virtual antennas) x (4Bytes)
Value: Samples to calculate static azimuth or elevation heatmap (no moving object signal). This is a 2D FFT array in range direction (x[numRangeBins][numVirtualAntAzim]), at doppler index 0. The antenna data are complex symbols, with imaginary first and real second in the following order:

Imag(ant 0, range 0), Real(ant 0, range 0),...,Imag(ant N-1, range 0),Real(ant N-1, range 0)
...
Imag(ant 0, range R-1), Real(ant 0, range R-1),...,Imag(ant N-1, range R-1),Real(ant N-1, range R-1)

NOTE: The demo will only output either the Azimuth Static Heatmap or the Azimuth/Elevation Static Heatmap

Range-Doppler Heatmap

Type: 5
Length: (Range FFT size) x (Doppler FFT size) x (2Bytes)
Value: Range/Doppler detection matrix.

X(range bin 0, Doppler bin 0),...,X(range bin 0, Doppler bin D-1),
...
X(range bin R-1, Doppler bin 0),...,X(range bin R-1, Doppler bin D-1)

Statistics

Type: 7
Length: 24 Bytes
Value: Stats information from data path. See the doxygen for detailed explanation of each stat.

Value	Type	Bytes
interFrameProcessingTime [usec]	uint32_t	4
transmitOutputTime[usec]	uint32_t	4
interFrameProcessingMargin [usec]	uint32_t	4
interChirpProcessingMargin [usec]	uint32_t	4
activeFrameCPULoad [%]	uint32_t	4
interFrameCPULoad [%]	uint32_t	4

Temperature Statistics

Type: 9
Length: 28 Bytes
Value: Temperature report - snapshot taken just before shipping data over UART

Value	Type	Bytes
tempReportValid (used to know if values are valid)	uint32_t	4
time (radarSS local Time from device powerup) [1LSB = 1ms]	uint32_t	4
tmpRx0Sens [1 LSB = 1 deg C]	uint16_t	2
tmpRx1Sens [1 LSB = 1 deg C]	uint16_t	2
tmpRx2Sens [1 LSB = 1 deg C]	uint16_t	2
tmpRx3Sens [1 LSB = 1 deg C]	uint16_t	2
tmpTx0Sens [1 LSB = 1 deg C]	uint16_t	2
tmpTx1Sens [1 LSB = 1 deg C]	uint16_t	2
tmpTx2Sens [1 LSB = 1 deg C]	uint16_t	2
tmpPmSens [1 LSB = 1 deg C]	uint16_t	2
tmpDig0Sens [1 LSB = 1 deg C]	uint16_t	2
tmpDig1Sens [1 LSB = 1 deg C] (Not valid for devices without DSP)	uint16_t	2

Need More Help?

- Additional resources in the documentation of the mmWave SDK:
    - mmWave SDK Module Doc located at `<mmwave_sdk_install_dir>/docs/mmwave_sdk_module_documentation.html`
    - mmWave SDK User's Guide located at `<mmwave_sdk_install_dir>/docs/mmwave_sdk_user_guide.pdf`
- Search for your issue or post a new question on the mmWave E2E forum (https://e2e.ti.com/support/sensor/mmwave_sensors/f/1023)