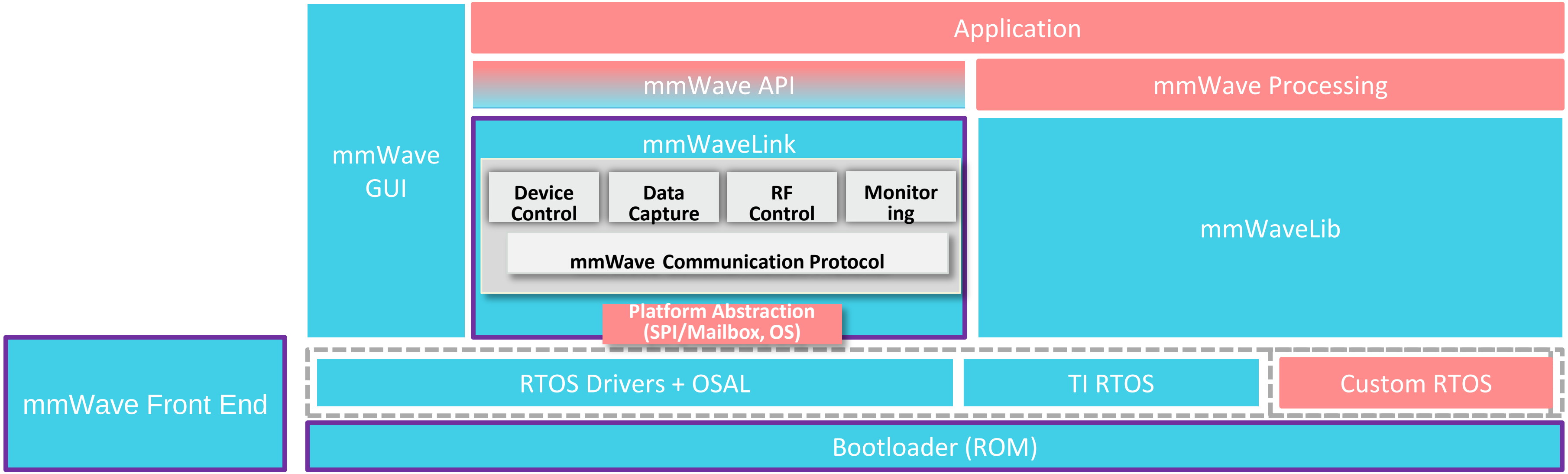
Radar Programming Model

Kaushal Kukkar, Texas Instruments

Tue 5/9/2017

mmWaveLink Framework



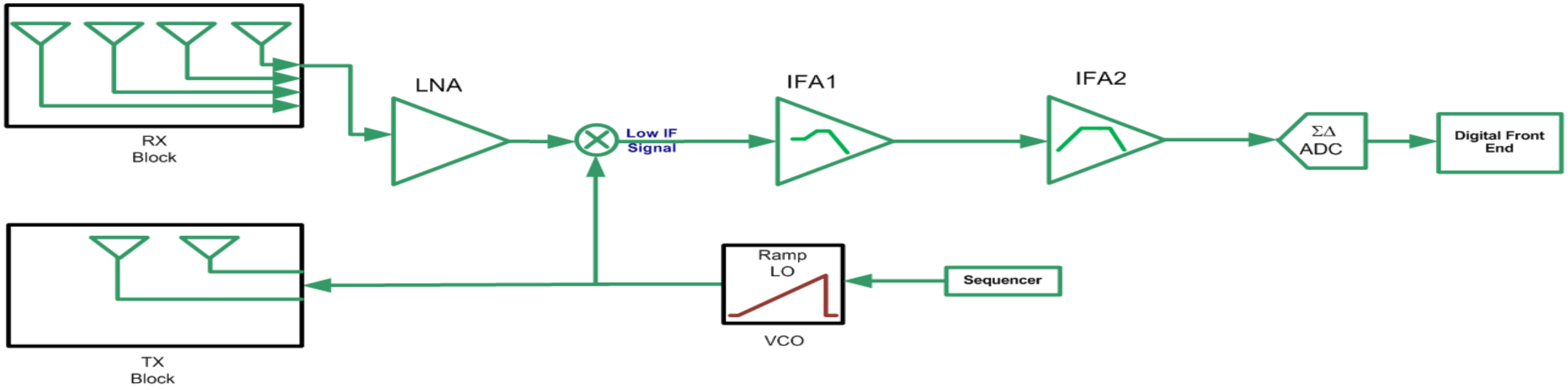
TI's mmWaveLink framework

- Is a link between Application and mmWave Radar device
- Runs on Cortex R4F or DSP and provides low level APIs to control the mmWave Front end
- Is platform independent and OS agnostic
- Can be ported to any External MCU and communicates with mmWave device over SPI

mmWaveLink Framework APIs

APIs	Description
Device Manager APIs	
rlDevicePowerOn	Initializes the driver and handshake with mmWave Front End
rlDevicePowerOff	De-initializes the driver
Sensor Control APIs	
rlSetChannelConfig	Configures the number of RX and TX channels
rlSetAdcOutConfig	Configures the ADC format (# bits, Real/Complex)
rlSetProfileConfig	Configures the profile (Frequency start, Frequency slope, Idle time, RX gain, ADC sampling rate, HPF1 and HPF2 cutoff, TX output power, TX phase shifter)
rlSetChirpConfig	Configures variable part for frequency start, slope, ADC start time, idle time and selection of TX for each chirp
rlSetFrameConfig	Configures the frame (start chirp index, end chirp index, number of chirp loops, frame periodicity)
rlSensorStart	Triggers the transmission of the frames as per the frame and chirp configuration
rlSensorStop	Stops the transmission of the frames.

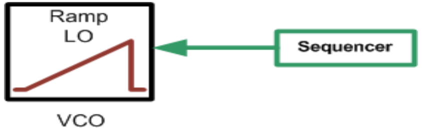
RF/Sensor Control

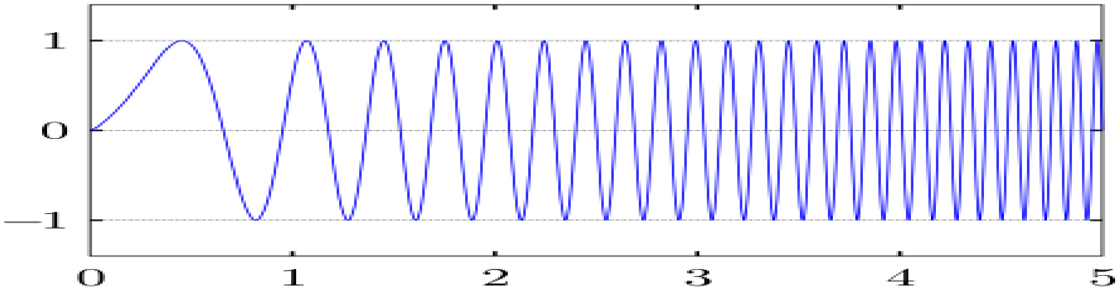


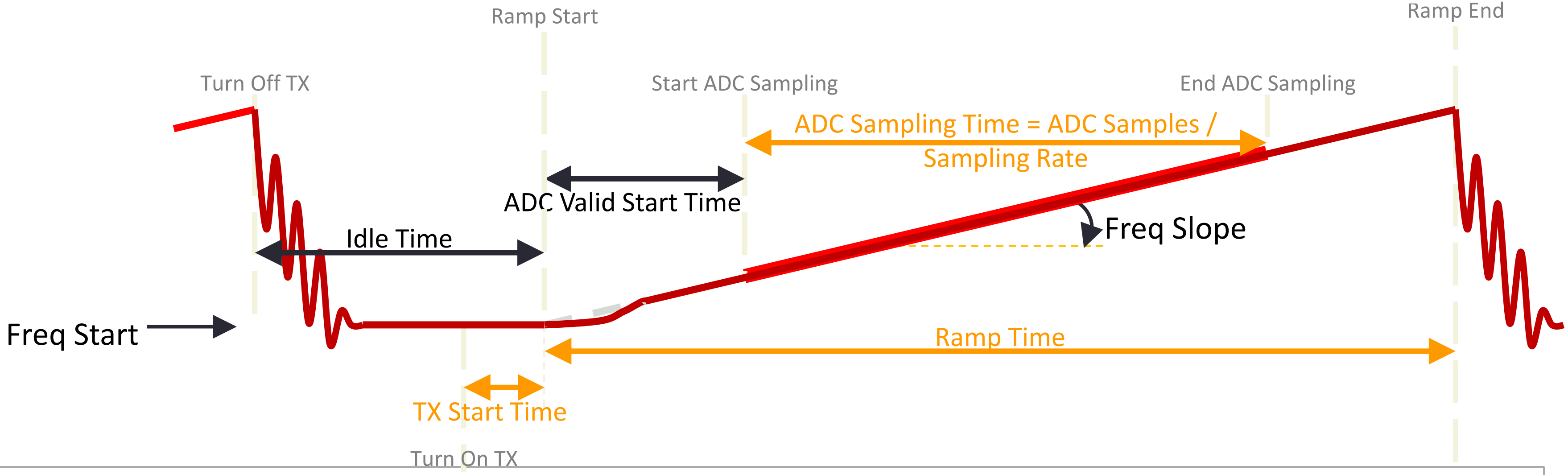
RF/Sensor Control

- Chirp sequencer (Radar Timing Engine)
- Rx/Tx Channel
- Rx Analog Chain
- ADC and Digital Front End Configuration

FMCW Chirp







Ramp Start

Ramp End

Turn Off TX

Start ADC Sampling

End ADC Sampling

ADC Sampling Time = ADC Samples / Sampling Rate

ADC Valid Start Time

Idle Time

Freq Slope

Freq Start

Ramp Time

TX Start Time

Turn On TX

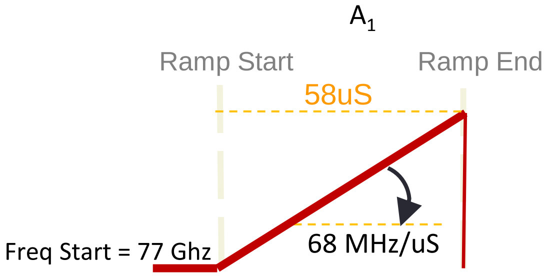
FMCW Chirp

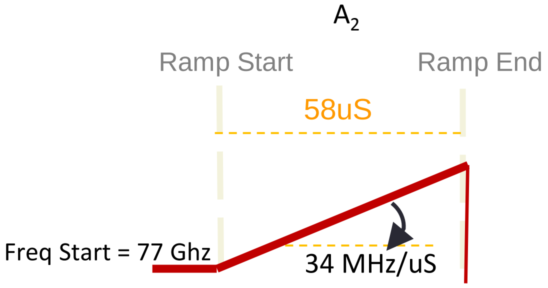
- Range resolution:
  - Directly proportional to the bandwidth (B) spanned by the chirp.
  - Bandwidth of 4GHz => ~4cm Resolution

$$d_{res} = \frac{c}{2B}$$

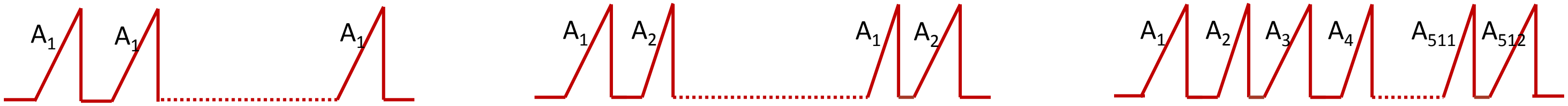
- IF Bandwidth: The required IF bandwidth increases with chirp slope (S).
  - Required IF bandwidth depends on chirp slope and maximum range(d_{max})
  - For given IF bandwidth, maximum range(d_{max}) is inversely proportional to Slope (S)

$$f_{IF\_max} = \frac{S2d_{max}}{c}$$

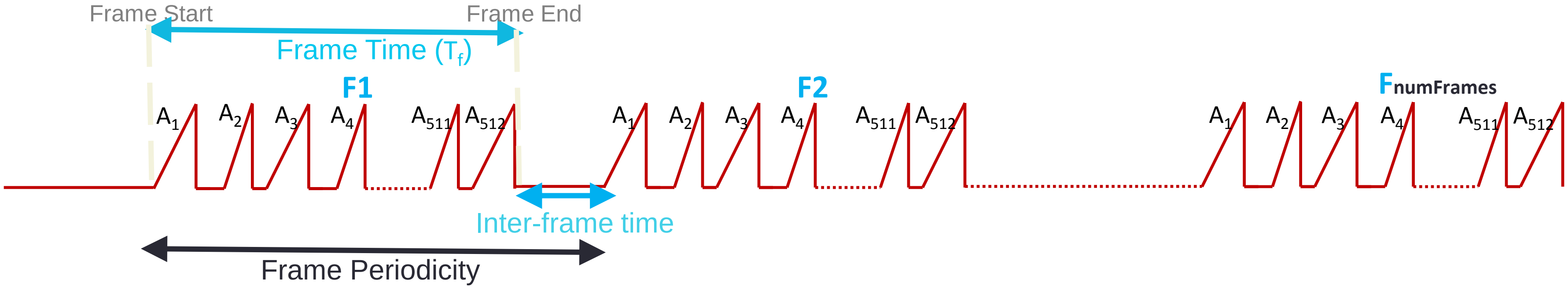




FMCW Frame



Frame = Sequence of chirps and periodicity of the sequence



Frame Start

Frame End

Frame Time (T_f)

F1

F2

$F_{numFrames}$

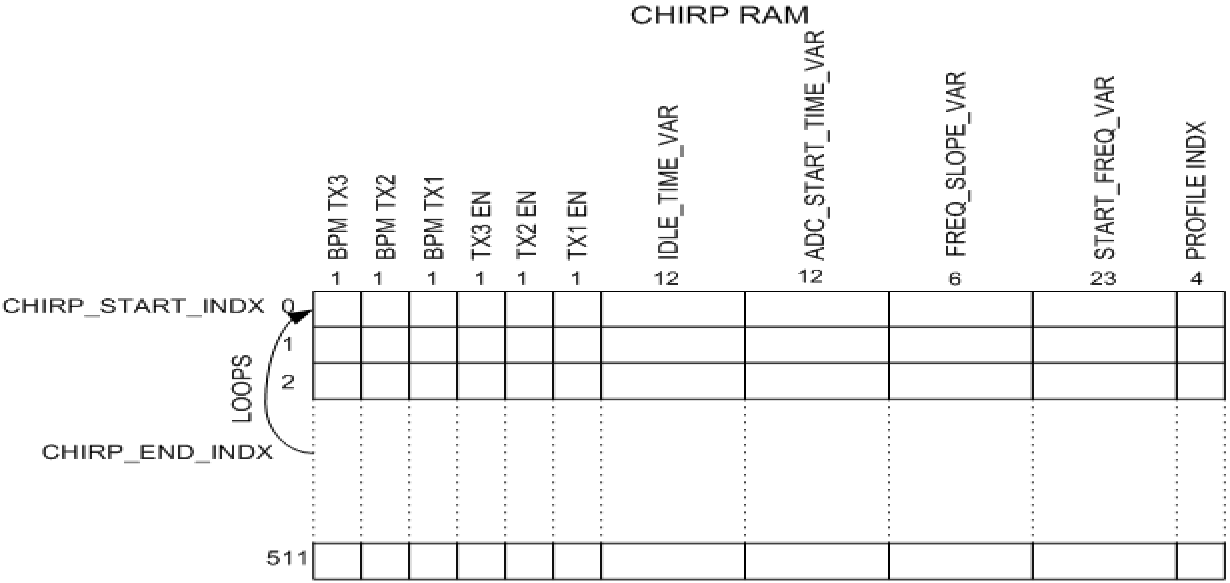
Inter-frame time

Frame Periodicity

- Velocity resolution:
  - Velocity resolution can be improved by increasing frame time (T_f)
  - A T_f of 5ms => v_{res} of 1.5 kmph

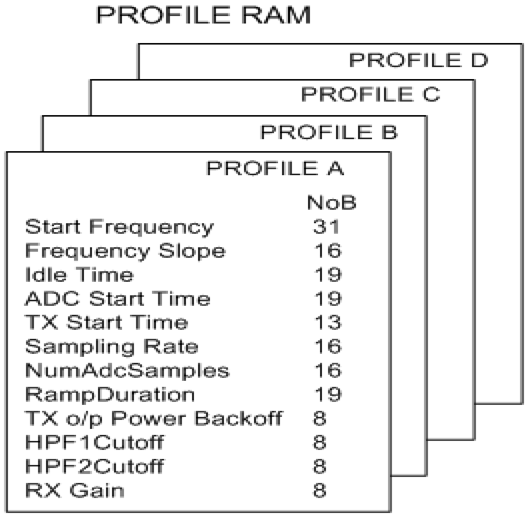
$$v_{res} = \frac{\lambda}{2T_f}$$

Profile RAM and Chirp RAM



CHIRP RAM

BPM TX3	BPM TX2	BPM TX1	TX3 EN	TX2 EN	TX1 EN	IDLE_TIME_VAR	ADC_START_TIME_VAR	FREQ_SLOPE_VAR	START_FREQ_VAR	PROFILE INDX
1	1	1	1	1	1	12	12	6	23	4

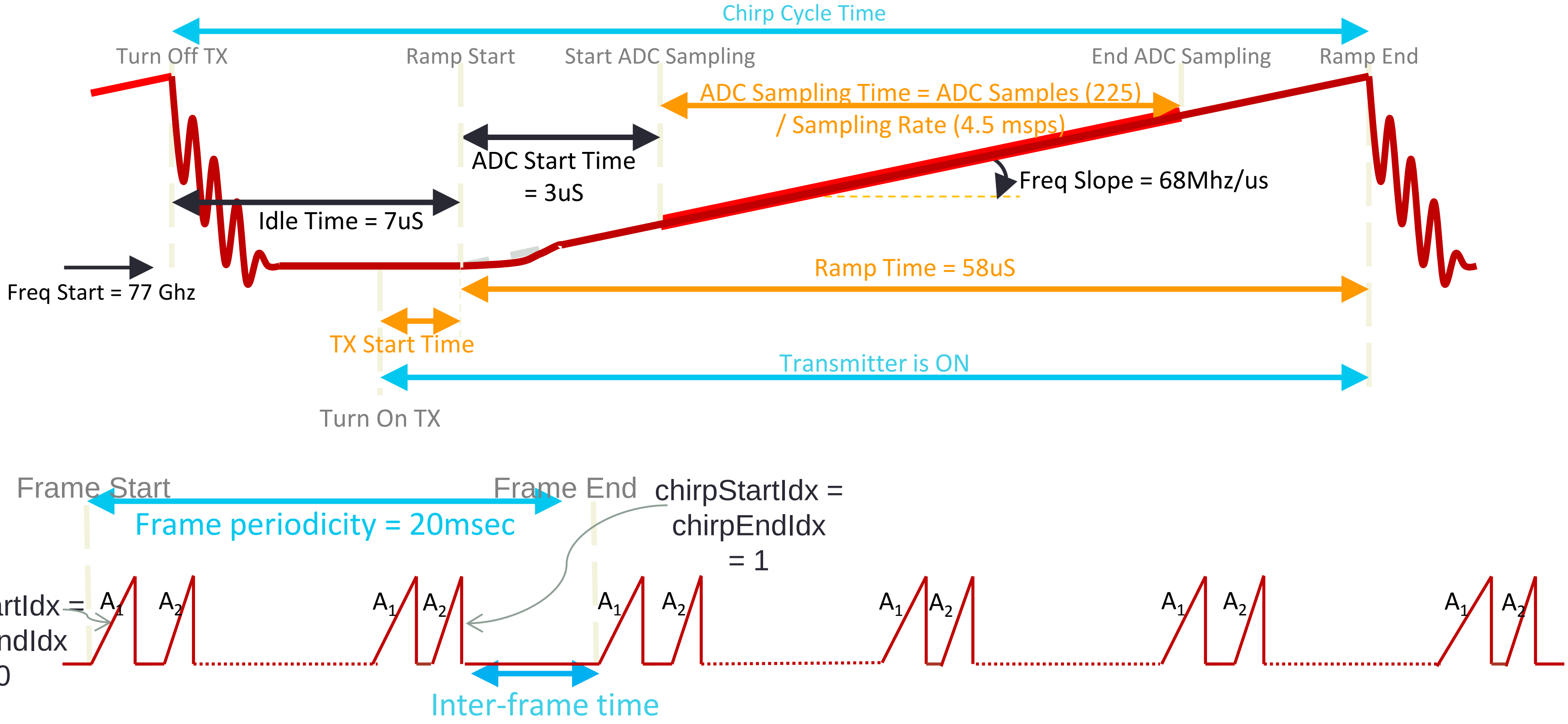


PROFILE RAM

	NoB
Start Frequency	31
Frequency Slope	16
Idle Time	19
ADC Start Time	19
TX Start Time	13
Sampling Rate	16
NumAdcSamples	16
RampDuration	19
TX o/p Power Backoff	8
HPF1Cutoff	8
HPF2Cutoff	8
RX Gain	8

*Both RAMs are ECC protected

Example Usecase



Profile Configuration

Type	Name	Description	Units	Value to be programmed
rlUInt16_t	profileId	Profile index for which rest of the properties are applicable	-	0
rlUInt32_t	startFreqConst	Frequency profile for each profile. 1 LSB = 3.6e9/2^26 = 54 Hz	Hz	0x558E38E3
rlUInt32_t	idleTimeConst	Idle time for each profile 1 LSB = 10 ns	ns	700
rlUInt32_t	adcStartTimeConst	ADC capture time for each profile. 1 LSB = 10 ns	ns	300
rlUInt32_t	rampEndTime	Ramp end of each profile. 1 LSB = 10ns	ns	5800
rlUInt32_t	txOutPowerBackoffCode	TX channel output power backoff. b7:0- TX0 output power backoff b15:8-TX1 output power backoff b23:16-TX2 output power backoff 1 LSB = 1 dB	dB	0
rlUInt32_t	txPhaseShifter	TX channel phase shift value b7:0 - TX0 phase shift value b15:8 - TX1 phase shift value b23:16 - TX2 phase shift value 1 LSB = 5°	degree	0
rlInt16_t	freqSlopeConst	Frequency slope for each profile 1LSB=3.6e9*900/2^26 = 48 kHz/µs	kHz/µs	0x580

Profile Configuration

Type	Name	Description	Units	Value to be programmed
rlInt16_t	txStartTime	TX start time 1 LSB = 10ns	ns	0
rlUInt16_t	numAdcSamples	Number of ADC samples to capture in a chirp for each RX		225
rlUInt16_t	digOutSampleRate	ADC sampling rate 1 LSB = 1 ksps	ksps	4500
rlUInt8_t	hpfCornerFreq1	HPF1 corner frequency for each profile 0x00– 175kHz 0x01- 235kHz 0x02- 350kHz 0x03- 700kHz	Hz	0x00
rlUInt8_t	hpfCornerFreq2	HPF2 corner frequency for each profile 0x00– 350kHz 0x01- 700kHz 0x02- 1.4MHz 0x03- 2.8MHz	Hz	0x00
rlUInt8_t	rxGain	RX gain for each profile 1LSB = 1dB	dB	0x1E

Configure profile

```
rlProfileCfg_t  profileCfgArgs;
/* Fill profile configuration structure */
rlSetProfileConfig(RL_DEVICE_MAP_CASCADED_1,  &profileCfgArgs);
```

Chirp Configuration (1/2)

Type	Name	Description	Units	Value to be programmed
rlUInt16_t	chirpStartIdx	Chirp start index valid range 0 to 511	-	0
rlUInt16_t	chirpEndIdx	Chirp start index valid range chirpStartIdx to 511	-	0
rlUInt16_t	profileId	Profile index valid range 0 to 3	-	0
rlUInt32_t	startFreqVar	Ramp start frequency variation 1 LSB = 3.6e9/2^26 = 54 Hz	Hz	0
rlInt16_t	freqSlopeVar	Ramp slope variation 1 LSB = 3.6e9*900/2^26 = 48 kHz/µs	kHz/µs	0
rlUInt16_t	idleTimeVar	Idle time for each chirp 1 LSB = 10 ns	ns	0 (0ns)
rlUInt16_t	adcStartTimeVar	ADC start time for each chirp 1 LSB = 10 ns	ns	0 (0ns)
rlUInt16_t	txEnable	TX enable bitmask b0 : TX0 Enable b1 : TX1 Enable b2 : TX2 Enable	-	1 (TX0)

Chirp Configuration (2/2)

Type	Name	Description	Units	Value to be programmed
rlUInt16_t	chirpStartIdx	Chirp start index valid range 0 to 511	-	1
rlUInt16_t	chirpEndIdx	Chirp start index valid range chirpStartIdx to 511	-	1
rlUInt16_t	profileId	Profile index valid range 0 to 3	-	0
rlUInt32_t	startFreqVar	Ramp start frequency variation 1 LSB = 3.6e9/2^26 = 54 Hz	Hz	0
int16_t	freqSlopeVar	Ramp slope variation 1 LSB = 3.6e9*900/2^26 = 48 kHz/µs	kHz/µs	0
rlUInt16_t	idleTimeVar	Idle time for each chirp 1 LSB = 10 ns	ns	0 (0ns)
rlUInt16_t	adcStartTimeVar	ADC start time for each chirp 1 LSB = 10 ns	ns	0 (0ns)
rlUInt16_ta	txEnable	TX enable bitmask b0 : TX0 Enable b1 : TX1 Enable b2 : TX2 Enable	-	4 (TX2)

Configure chirp

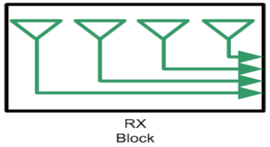
```
rlChirpCfg_t  chirpCfgArgs[2U];
/* Fill chirp #0(chirpCfgArgs[0]) and chirp #1(chirpCfgArgs[1]) configuration structure*/
rlSetChirpConfig(RL_DEVICE_MAP_CASCADED_1, 2U, chirpCfgArgs);
```

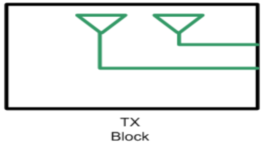
Frame Configuration

Type	Name	Description	Units	Value to be programmed
rlUInt16_t	chirpStartIdx	Chirp start index. Valid range 0-511	-	0
rlUInt16_t	chirpEndIdx	Chirp end index. Valid range from chirpStartIdx to 511		1
rlUInt16_t	numLoops	Number of times to repeat from chirpStartIdx to chirpEndIdx in each frame		32
rlUInt16_t	numFrames	Number of frames to transmit Set 0 for infinite frames	-	0 (Infinite)
rlUInt32_t	framePeriodicity	Frame repetition period 1LSB = 5ns	ns	4000000
rlUInt16_t	triggerSelect	Frame trigger method 0x0001: SW API based triggering. 0x0002: HW SYNC_IN based triggering.		1
rlUInt32_t	frameTriggerDelay	Optional time delay from the SYNC_IN trigger to the occurrence of frame chirps. 1LSB = 1ns	ns	0

Configure frame

```
    rlFrameCfg_t  frameCfgArgs;
    /* Fill frame configuration structure */
    rlSetFrameConfig(RL_DEVICE_MAP_CASCADED_1,  & frameCfgArgs);
```



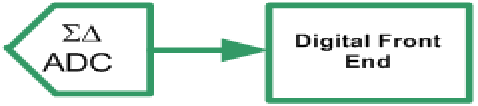


Rx/Tx Channel configuration

Type	Name	Description	Units	Value to be programmed
rlUInt16_t	rxChannelEn	RX Channel Enable/disable b0 – RX Channel 0, b1 – RX Channel 1 b2 – RX Channel 2, b3 – RX Channel 3 b15:4 – Reserved	-	0xF
rlUInt16_t	txChannelEn	TX Channel Enable/disable b0 – TX Channel 0, b1 – TX Channel 1 b2 – TX Channel 2, b15:3 – Reserved	-	0x5 (TX1, TX3)
rlUInt16_t	cascading	Cascading enable 0x0000 SINGLECHIP: Single mmWave sensor application 0x0001 MULTICHIP_MASTER: This mmwave device is the master chip and generates LO and conveys to other mmwave sensors 0x0002 MULTICHIP_SLAVE: This mmwave device is a slave chip and uses LO conveyed to it by the master mmwave sensor.	-	0

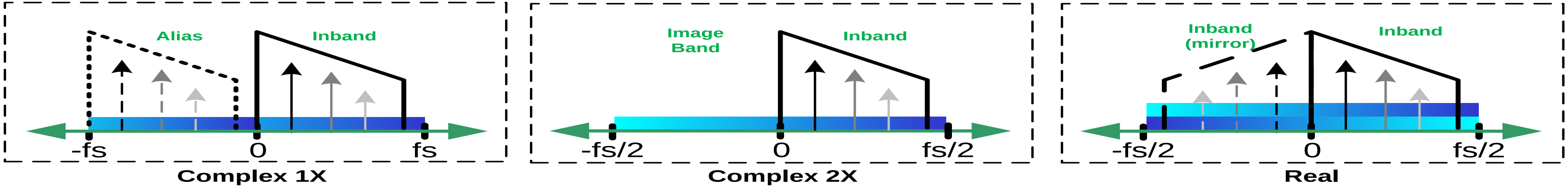
Configure Tx/Rx Channels

```
rlChanCfg_t rfChanCfgArgs = {0};
/* Fill profile configuration structure */
rlSetChannelConfig(RL_DEVICE_MAP_CASCADED_1,  & rfChanCfgArgs);
```



ADC configuration

Type	Name	Description	Units	Value to be programmed
rlUInt32_t	adcBits	Number of ADC bits 00: 12 bits 01: 14 bits 10: 16 bits	-	0b10
rlUInt32_t	adcOutFmt	ADC output Format 00:Real 01:Complex 1x (image band filtered output) 10:Complex 2x (Image band visible)	-	0b01



Configure ADC output

```
rlAdcOutCfg_t adcOutCfgArgs = {0};
/* Fill ADC output configuration structure */
rlSetAdcOutConfig(&adcOutCfgArgs);
```

Sensor Control

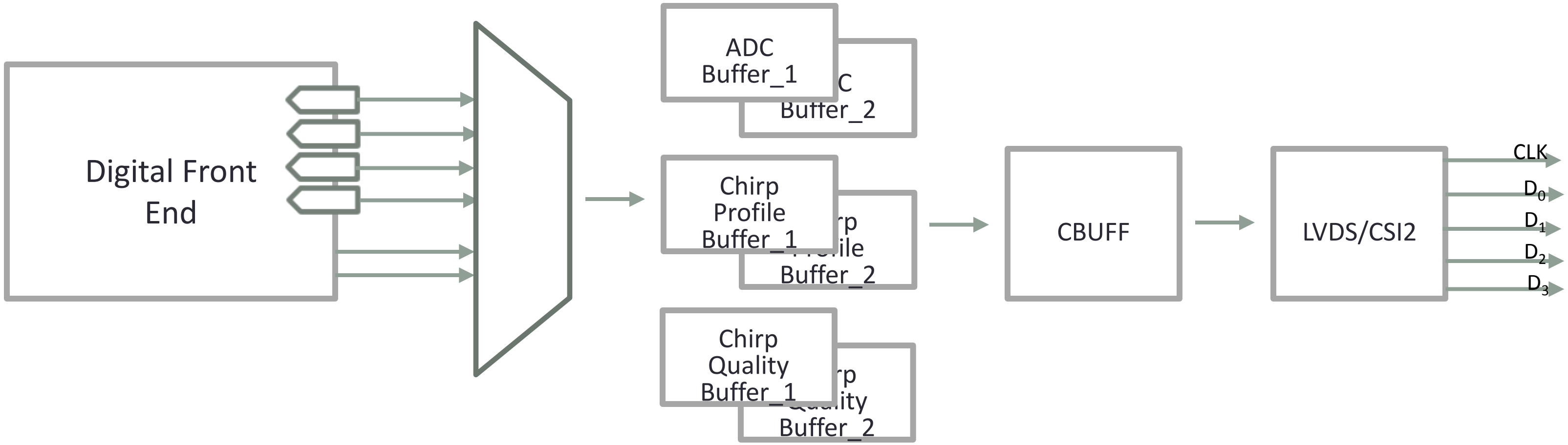
- Trigger the frame

  rlSensorStart(RL_DEVICE_MAP_CASCADED_1)

- Stop the frame

  rlSensorStop(RL_DEVICE_MAP_CASCADED_1)

Data Capture



Data Capture

- High Speed interface(LVDS/CSI2) selection
- Data format and rate configuration
- Lane configuration
- Interface specific configurations

mmWave Data Control (1/4)

Type	Name	Description	Units	Value to be programmed
rlUInt8_t	intfSel	Data path interface select 0 – CSI2 1 – LVDS	-	1
rlUInt8_t	transferFmtPkt0	Data Output format 000001b ADC 000110b CP_ADC 001001b ADC_CP 110110b CP_ADC_CQ Others Reserved	-	1
rlUInt8_t	transferFmtPkt1	000000b Suppress Packet 1 transmission. 001110b CP_CQ 001011b CQ_CP Others Reserved	-	0

Configure data path interface

```
 rlDevDataPathCfg_t devDataPathArgs = {0};
/* Fill data path configuration structure */
rlDeviceSetDataPathConfig(RL_DEVICE_MAP_CASCADED_1, &devDataPathArgs);
```

mmWave Data Control (2/4)

Type	Name	Description	Units	Value to be programmed
rlUInt8_t	laneClkCfg	Bitmap for the lane clock selection 0 : SDR Clock 1 : DDR Clock	-	1
rlUInt8_t	dataRate	Bitmap for data rate selection 0000b 900 Mbps (DDR only) 0001b 600 Mbps (DDR only) 0010b 450 Mbps (SDR, DDR) 0011b 400 Mbps (DDR only) 0100b 300 Mbps (SDR, DDR) 0101b 225 Mbps (DDR only) 0110b 150 Mbps (SDR, DDR)	-	2 (450 Mbps DDR)

Configure HSI clock

```
rlDevDataPathClkCfg dataPathClkCfgArgs = {0};
/* Fill data path clock configuration structure */
retVal = rlDeviceSetDataPathClkConfig(RL_DEVICE_MAP_CASCADED_1, &dataPathClkCfgArgs);
```

Ensure data rate meets this criterion

NumBitsPerChirp * NumRxChannels/NumLanes < RampDuration * dataRate

For e.g. $(256 * 4 * 8) * 4/2 < 60 \mu s * 450 \text{Mbps}$

mmWave Data Control (3/4)

Type	Name	Description	Units	Value to be programmed
rlUInt8_t	rxChannelEn	Bitmap for each of the RX channel	-	0xF
rlrlUInt32_t	b2AdcBits:2	Number of ADC bits 00: 12 bits 01: 14 bits 10: 16 bits	-	01 (14 Bits)
rlUInt32_t	b2AdcOutFmt:2	ADC output Format 00: Real 01: Complex 1x(image band filtered output) 10: Complex 2x(Image band visible)	-	01 (Complex 1x)
rlUInt8_t	iqSwapSel	IQ bit swap selection 00 – Sample Interleave Mode - I first 01 – Sample Interleave Mode – Q first	-	0
rlUInt8_t	chInterleave	00 – Interleaved mode of storage 01 – Non-Interleaved mode	-	0

Configure data format

```
rlDevDataFmtCfg dataFmtCfgArgs = {0};
/* Fill data format configuration structure */
retVal = rlDeviceSetDataFmtConfig(RL_DEVICE_MAP_CASCADED_1, &dataFmtCfgArgs);
```

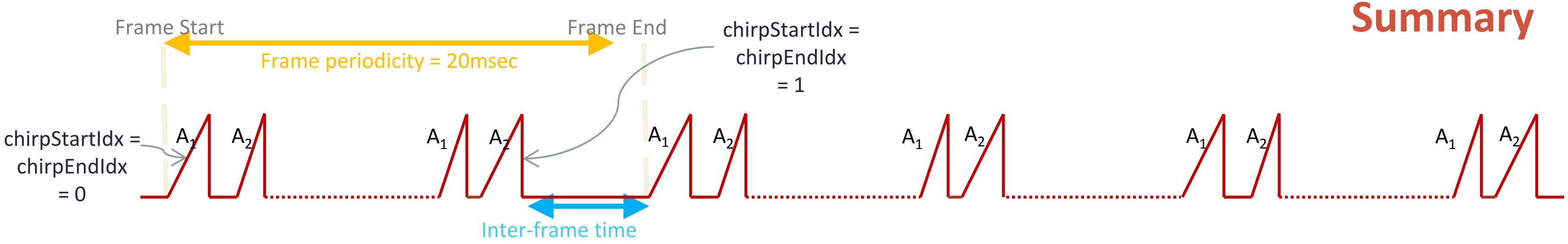
mmWave Data Control (4/4)

Type	Name	Description	Units	Value to be programmed
rlUInt16_t	laneEn	Lane enable bitmap. b0 - LANE0_EN 0 Disable lane 0 1 Enable lane 0 b1 - LANE1_EN 0 Disable lane 1 1 Enable lane 1 b2 - LANE2_EN 0 Disable lane 2 1 Enable lane 2 b3 - LANE3_EN 0 Disable lane 3 1 Enable lane 3 b15:4 Reserved		0xF
rlUInt16_t	Reserved	Reserved		

Configure the lanes

```
rlDevLaneEnable_t laneEnCfgArgs = {0};
/* Fill Lane enable parameters */
retVal = rlDeviceSetLaneConfig(RL_DEVICE_MAP_CASCADED_1, &laneEnCfgArgs);
```

Summary


Frame Start
Frame End
Frame periodicity = 20msec
chirpStartIdx = chirpEndIdx = 1
chirpStartIdx = chirpEndIdx = 0
A_1
A_2
Inter-frame time

- rlDevicePowerOn →Initializes the device driver and opens SPI/Mailbox for communication
- rlDeviceRfStart → Turns on mmWave Front End

- rlSetChannelConfig → Configures the RF channels which will be in use
- rlSetAdcOutConfig →Sets the ADC output format, real/complex mode
- rlSetLowPowerModeConfig → Configures the low power options
- rlRfInit → Initializes the RF subsystem with one time calibrations
- rlSetProfileConfig → Configure one profile (say A)
- rlSetChirpConfig → Configure all chirps and associate them to a profile. Also configure variable parameters per chirp
- rlSetFrameConfig → Configure the frame

- rlDeviceSetDataPathConfig → Sets the Data Path I/F
- rlDeviceSetDataPathClkConfig→ Selects the data rate
- rlDeviceSetDataFmtConfig→ Selects the data format
- rlDeviceSetLaneConfig→ Selects the CSI2/LVDS lanes

- rlSensorStart → Starts the frame