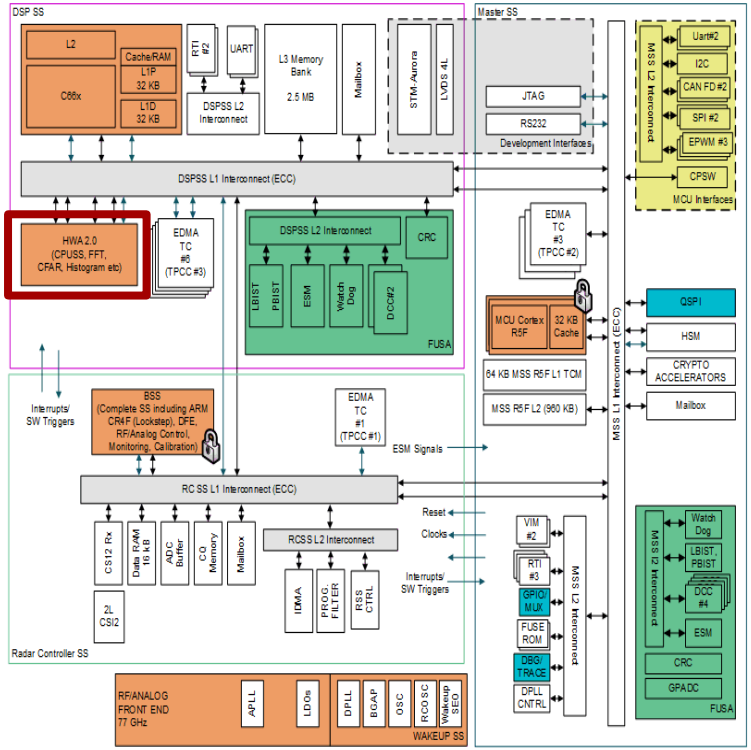
Hardware Accelerator (HWA) 2.0 overview

Agenda

- Key features
- Function of each sub-block
- Programming the HWA
- Example use-case of HWA

Introduction to hardware accelerator

- The radar hardware accelerator (HWA) is a hardware IP that enables off-loading the burden of some frequently used computations in FMCW radar signal processing from the main processor/DSP.
- FMCW radar signal processing involves pre-processing of the input data followed by multiple FFT's to obtain range, velocity and angle from the radar image and also supports detection of objects from it.
- The HWA 2.0 version is introduced in AWR294x devices.
- It is much more advanced and is able to perform complex operations than HWA1.0 version which was introduced in generation-1 devices (like AWR1443, AWR1843 etc).



AWR 294x block diagram

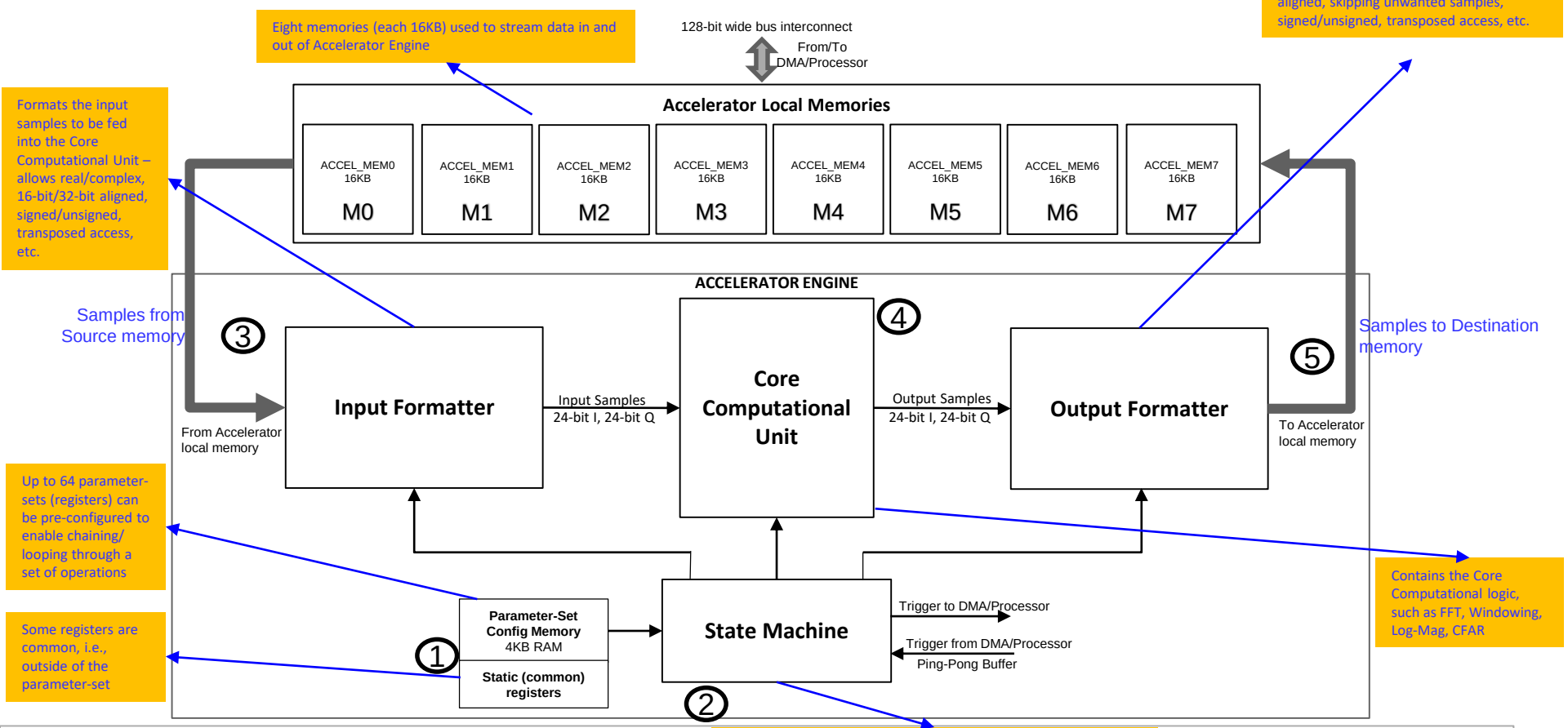
Key features

- **HWA 2.0 operates at a clock of 300MHz.**
- **Larger size local RAM – 128KB (16KB x 8)**
  - Flexible data flow and data sample arrangement to support efficient multi-dimensional FFT operations and transpose access.=
- **Pre-processing**
  - DC estimation and correction
  - Interference localization & mitigation with interference statistics
  - Complex multiplication (multiple modes) with both scalar and vector multiplication
  - Zero-Insertion/padding, Channel combination
- **FFT computation with programmable stage**
  - Programmable windowing (dynamic window update)
  - Up to 2048-pt complex FFT
  - Support for various FFT sizes – 2^N and 3*2^N
  - Supports 2D-FFT: M x N <=2048
  - Internal FFT bit-width of 24-bits (each of I and Q) for good Signal to Quantization-Noise Ratio (SQNR)
  - Programmable FFT at every stage for flexibility

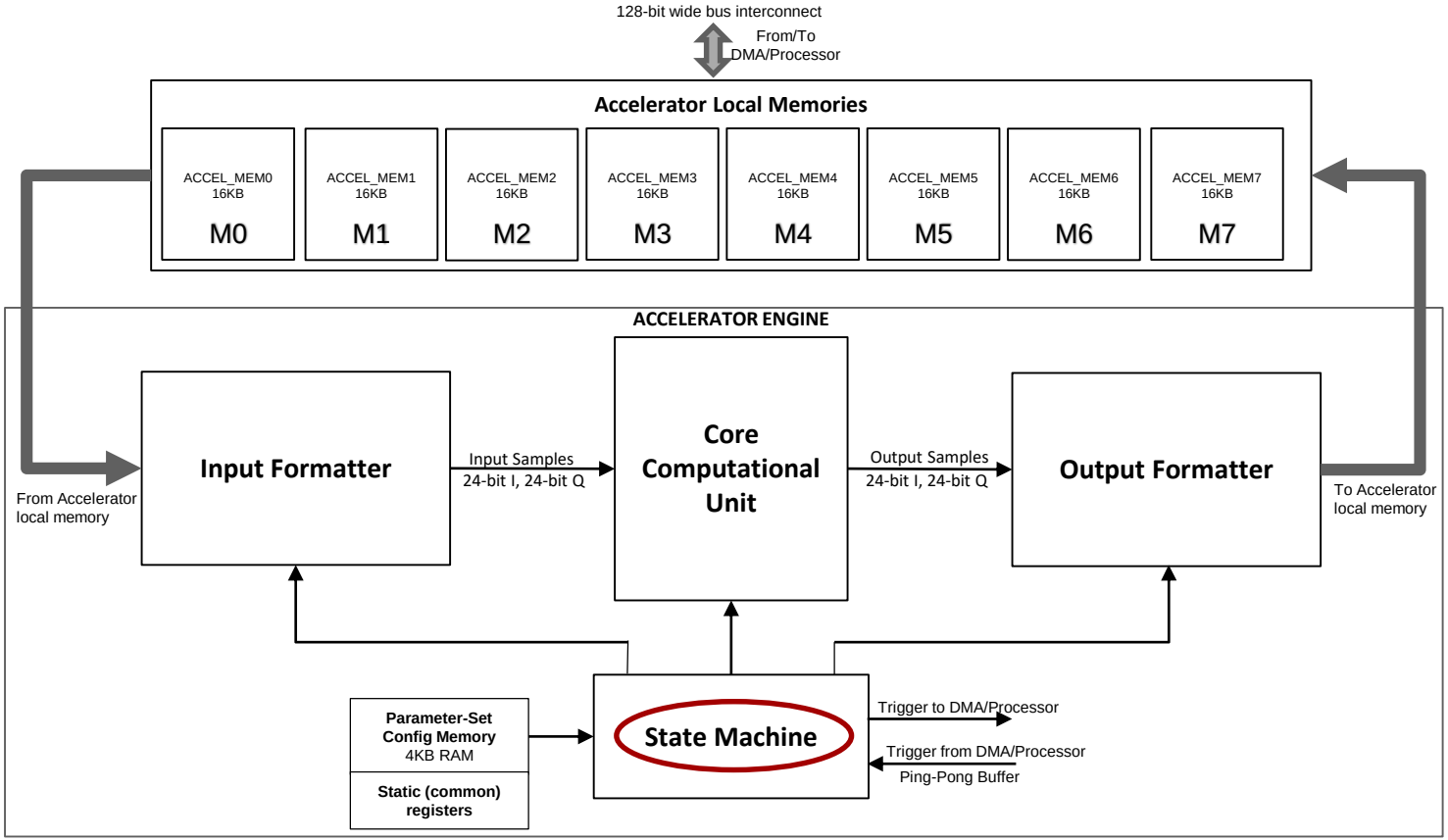
Key features (contd..)

- **Compression and decompression engine**
  - Supports compression and decompression of radar data using EGE (Exponential Golomb Encoder) format.
- **Post-FFT processing**
  - Advanced 2D-FFT statistics (max across) each dimension including basic statistics can be calculated.
  - Histogram/CDF statistics can also be calculated.
  - Log & magnitude computation is possible
  - CFAR CA (Cell Averaging) and CFAR OS (Ordered Statistics) detection are supported
  - Local maxima computation across 2D-plane is also supported with 4-modes of operation
- **Context switching**
  - Supports context switching which enables us to interrupt in order to perform a higher priority task before resuming the current task execution.
- **Miscellaneous other capabilities of HWA**
  - Stitching two or four 2K-point FFTs to get the equivalent of 4096-point or 8192-point FFT
  - Slow DFT mode, with resolution equivalent to 16K size FFT, for FFT interpolation purposes
  - Safety features are present like parity for memories and lock-step for FSM

Accelerator engine block diagram



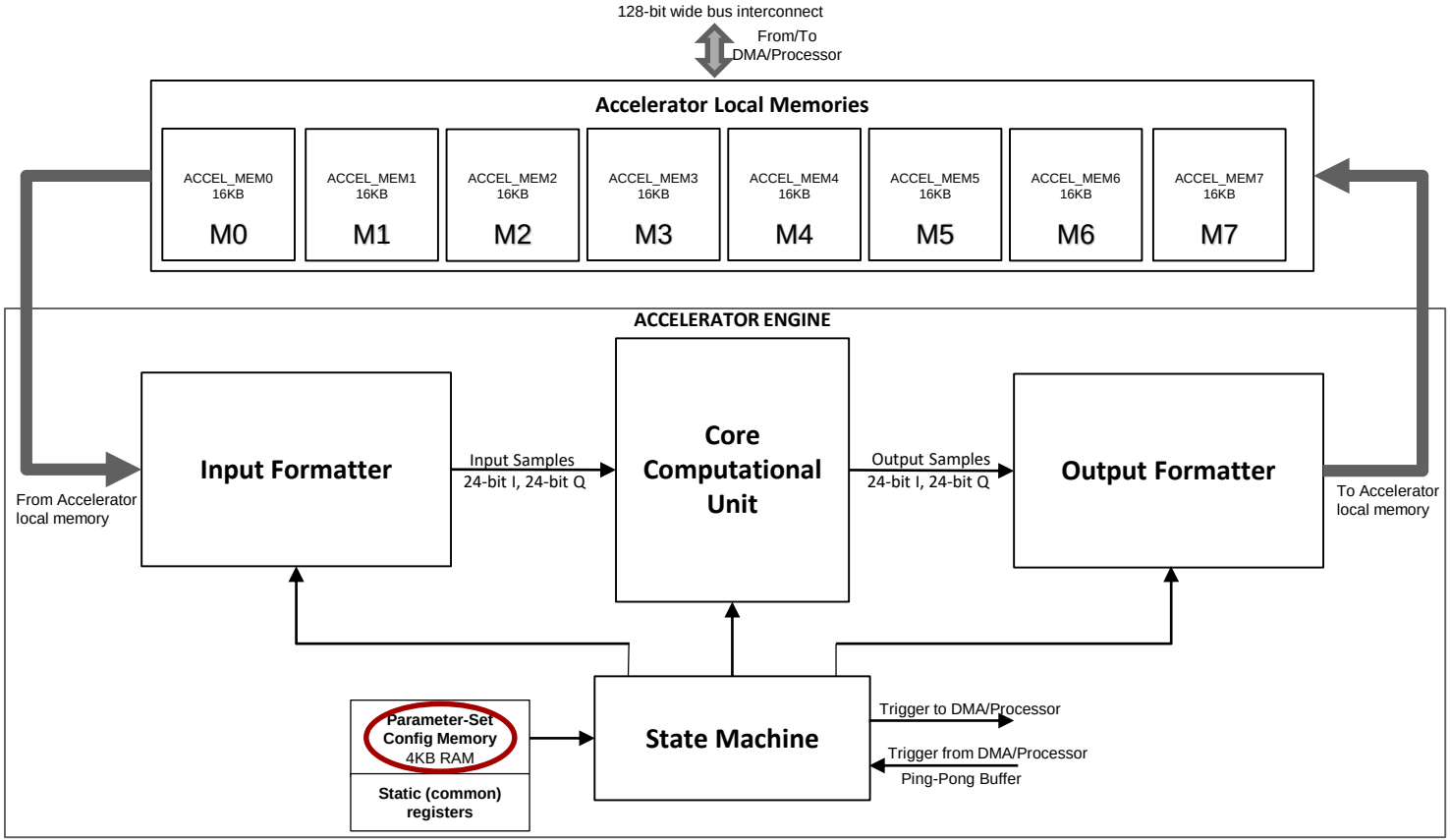
Accelerator engine block diagram



State machine

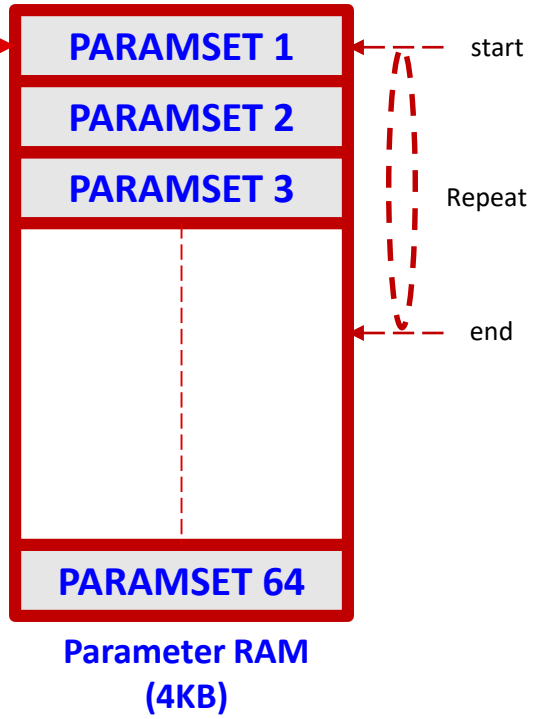
- **Operation**
  - Controls the overall functioning of HWA by enabling and disabling it
  - Sequencing a set of operations and looping through those operations one after another
- **Trigger mechanisms**
  - 4-incoming trigger
    - Immediate Trigger
    - Software Trigger
    - DMA based trigger
    - Hardware trigger
  - 2-outgoing trigger
    - Interrupt to main processor
    - Trigger to DMA
- **Supports an advanced operation called “Context Switching”**
  - Sequence of operations running in HWA is interrupted to run a different (high priority) sequence of operations
  - On finishing those operations, it returns to resume the execution of original sequence

Accelerator engine block diagram

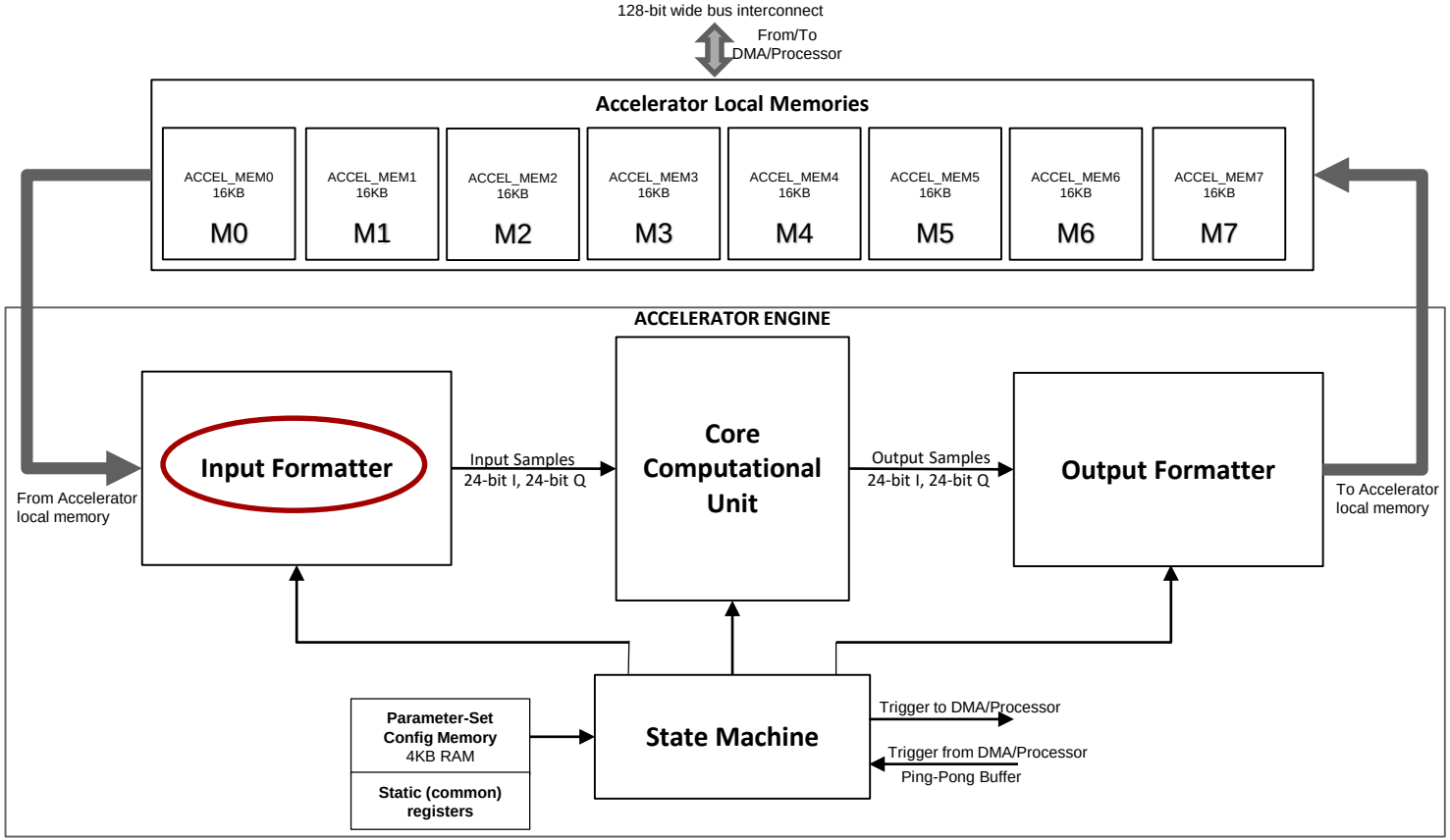


Parameter-set configuration memory

- Structure
  - 64 parameter sets
  - Each set consists of 16, 32-bit registers
- State machine can be programmed to loop through a specific subset of parameter sets based on param_start_idx, param_stop_idx (in PARAM_RAM_IDX struct) and numloops (in PARAM_RAM_LOOP struct).
- Functions:
  - Choosing and configuring each of blocks of computational engine
  - Configuring the input/output data format
  - Configuring the trigger for the parameter set
  - Configuring 2D-memory indexing
- State machine can also be configured to loop through another set of parameter sets for context switching based on param_start_idx, param_stop_idx and numloops in PARAM_RAM_IDX_ALT struct.

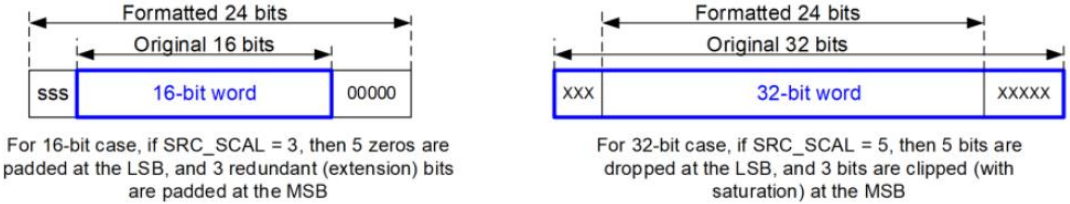


Accelerator engine block diagram



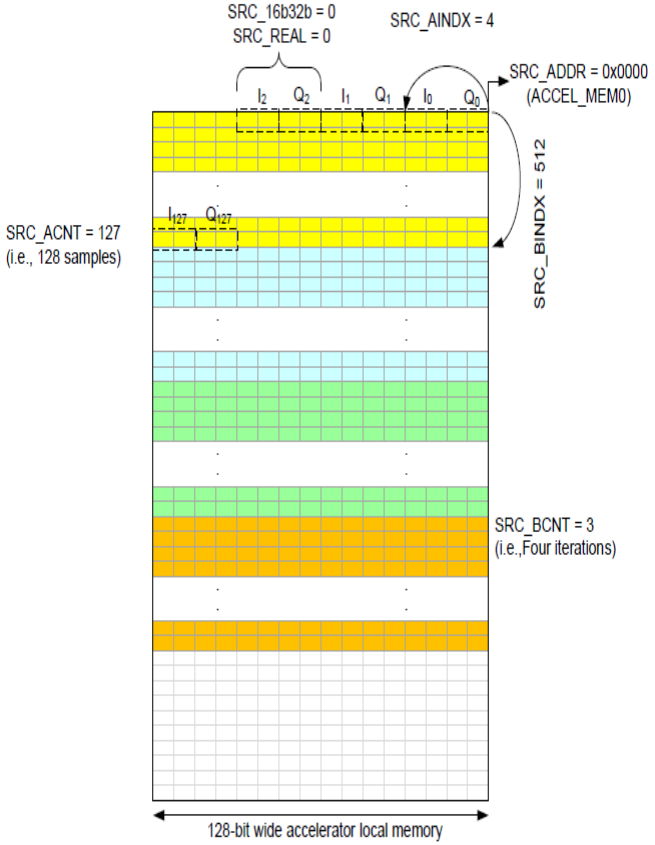
Input formatter

- Used to access, format and feed data from local memories of HWA to core computational unit as 24-bit complex samples.
- Supports input samples of 16-bit or 32-bit real or complex, variable arrangement of input data along with circular and shuffled addressing.
- Support for conjugation (to perform IFFT), taking only complex or real samples, sign extension and; swapping I and Q bits of complex samples are also provided.
- Also supports scaling and formatting of input samples to convert from 16bit or 32-bit to 24-bit for core computational unit.

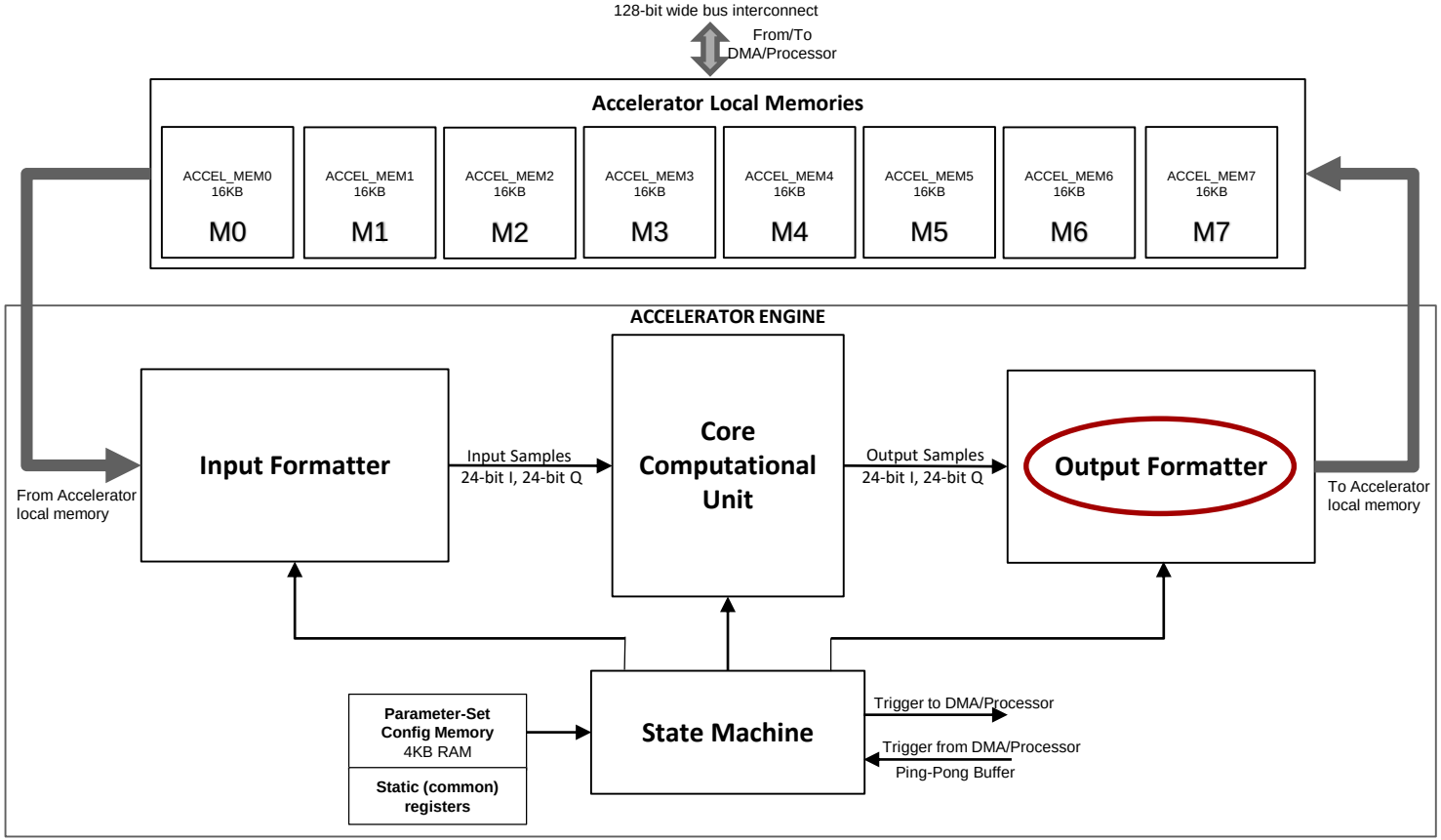


For 16-bit case, if SRC_SCAL = 3, then 5 zeros are padded at the LSB, and 3 redundant (extension) bits are padded at the MSB

For 32-bit case, if SRC_SCAL = 5, then 5 bits are dropped at the LSB, and 3 bits are clipped (with saturation) at the MSB

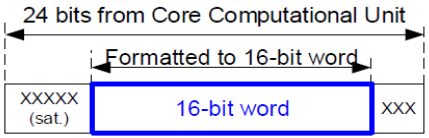


Accelerator engine block diagram

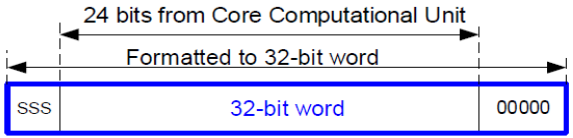


Output formatter

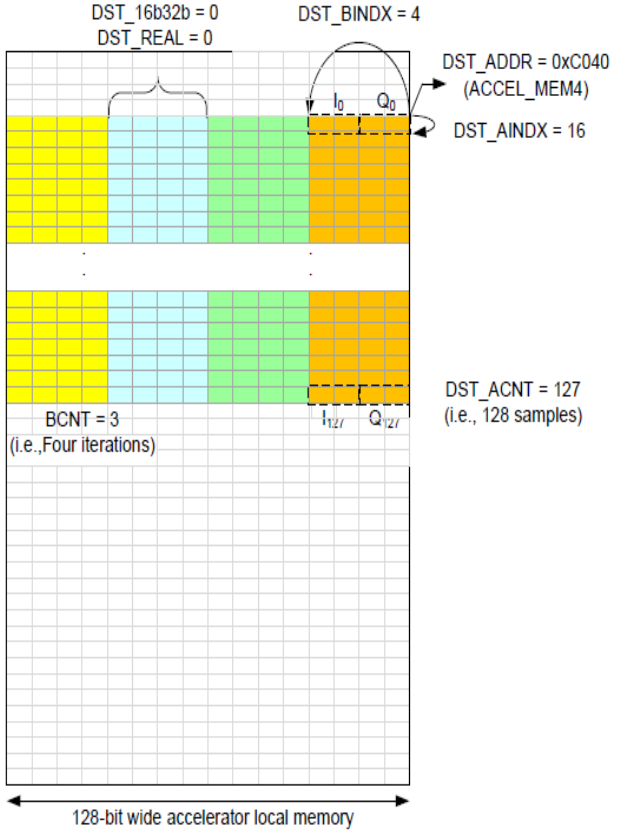
- Used to format and write data coming out of core computational unit into the HWA memory.
- Similar to input formatter, it supports output samples of 16-bit or 32-bit real or complex and variable arrangement of output data.
- Support for conjugation (to perform IFFT), taking only complex or real samples, skipping of samples, sign extension and; swapping I and Q bits of complex samples are also provided.
- Also, supports scaling and formatting of output samples of 24-bit from core computational unit to convert to 16bit or 32-bit.



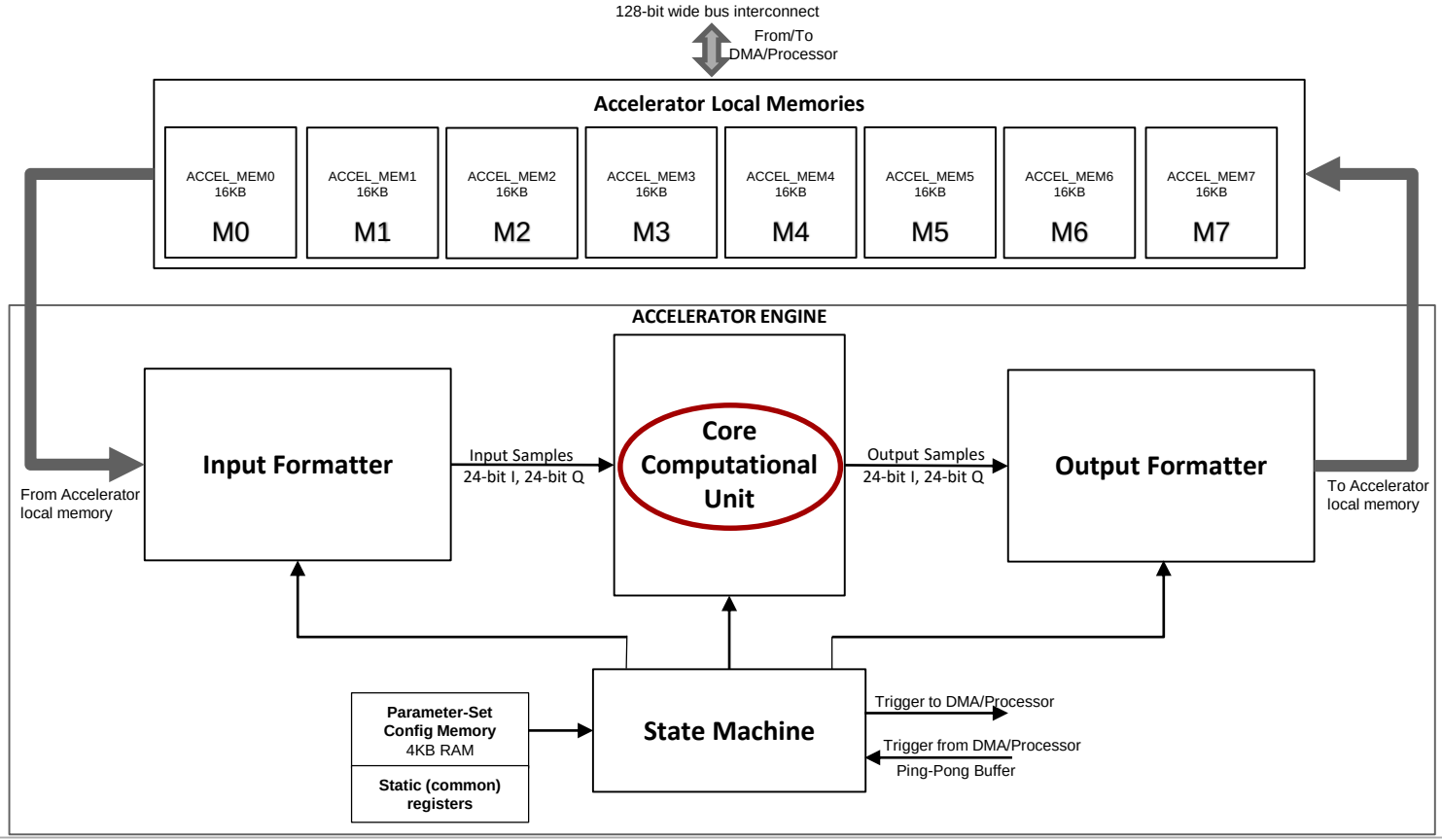
For 16-bit case, if DST_SCAL = 3, then 3 btis are dropped at the LSB, and 5 bits are clipped (saturated) at the LSB



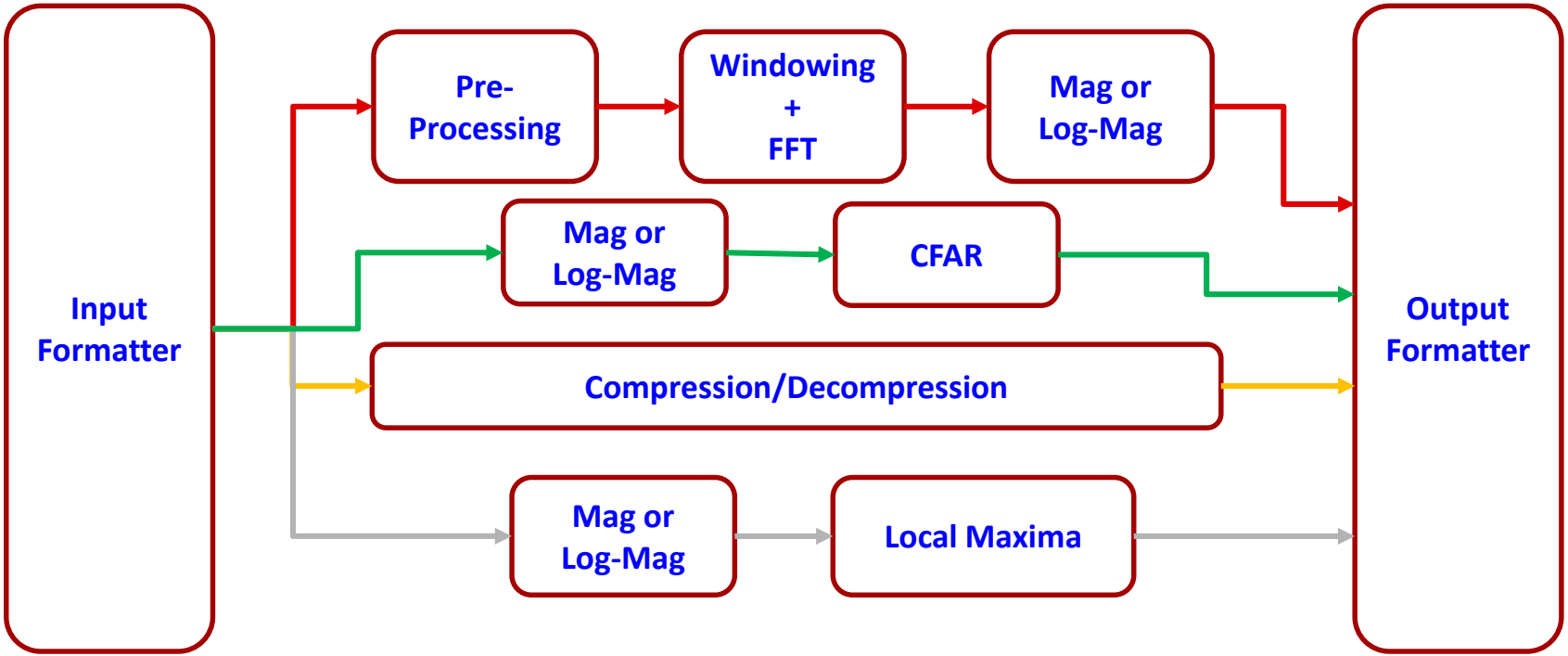
For 32-bit case, if DST_SCAL = 3, then 5 zeros are padded at the LSB, and 3 bits are extended at the MSB



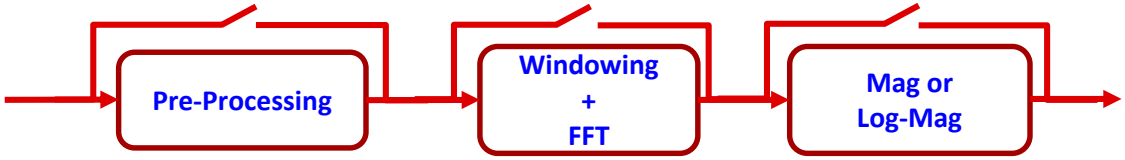
Accelerator engine block diagram



Core computational unit



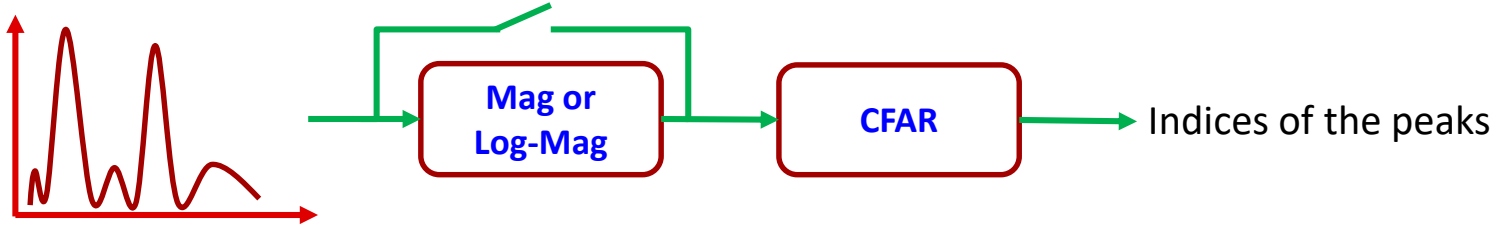
Hardware accelerator: FFT path



- Each block operates on a streaming input of data and produces a streaming output of data at a throughput of one clock sample. Bypassing of blocks are controlled through the param-set registers.
- Pre-processing block performs operations of dc correction, interference localization and mitigation, complex scalar and vector multiplication, zero padding and channel combining.
- For the purpose of windowing, window RAM can hold up to 2048 32-bit words of window coefficients in three formats (half of the size is only required in RAM if window function is symmetric)
- Magnitude or Log-Magnitude operations can be performed as a part of post-processing step.

Example	FFT Size	Number of Back-to-Back Iterations	Number of cycles (Initial Latency + Computation)	Total Duration
1	256	4	256 + (256 x 4)	4.27µs
2	128	4	128 + (128 x 4)	2.13µs
3	8	64	8 + (64 x 8)	1.73µs

Hardware accelerator: CFAR path

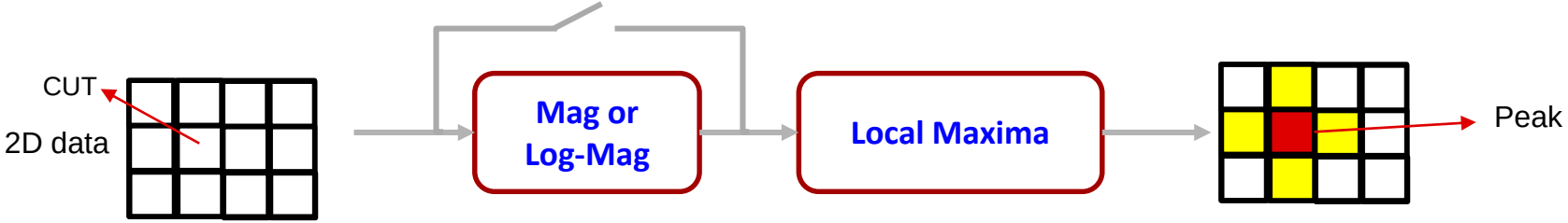


- Supports two types of CFAR algorithms: CFAR-CA (Cell-Averaging, CASO and CAGO) and CFAR-OS (Ordered Statistics) which determines the threshold of indices of peaks.
- CFAR is implemented on a real input vector, therefore provides us the option of Magnitude or Log-Magnitude incase the input to it is complex.
- Also, supports cyclic and non-cyclic mode of CFAR operation.
- The output of this path is a list of indices corresponding to detected peaks.

Hardware accelerator: Compression/decompression

- Has a compression engine which takes a fixed number of samples and returns a ‘block of bits’ such that the block’s size (in bits occupied) is a fraction of the size of the input samples. The compression module can achieve an arbitrary compression ratio.
- For e.g. , a 33 % compression-ratio, results in the average bit-width after compression being one-third of the bit-width before compression.
- Also, has a decompression engine which when provided with the a compressed block of bits, regenerates the original samples (with a possibility of some quantization error).
- Algorithms for compression/decompression is based on Exponential Golomb Encoding (EGE).
- A compression ratio in the range 33%-50% is good with 50% ratio being the nearly loss-less.

Hardware accelerator: Local maxima

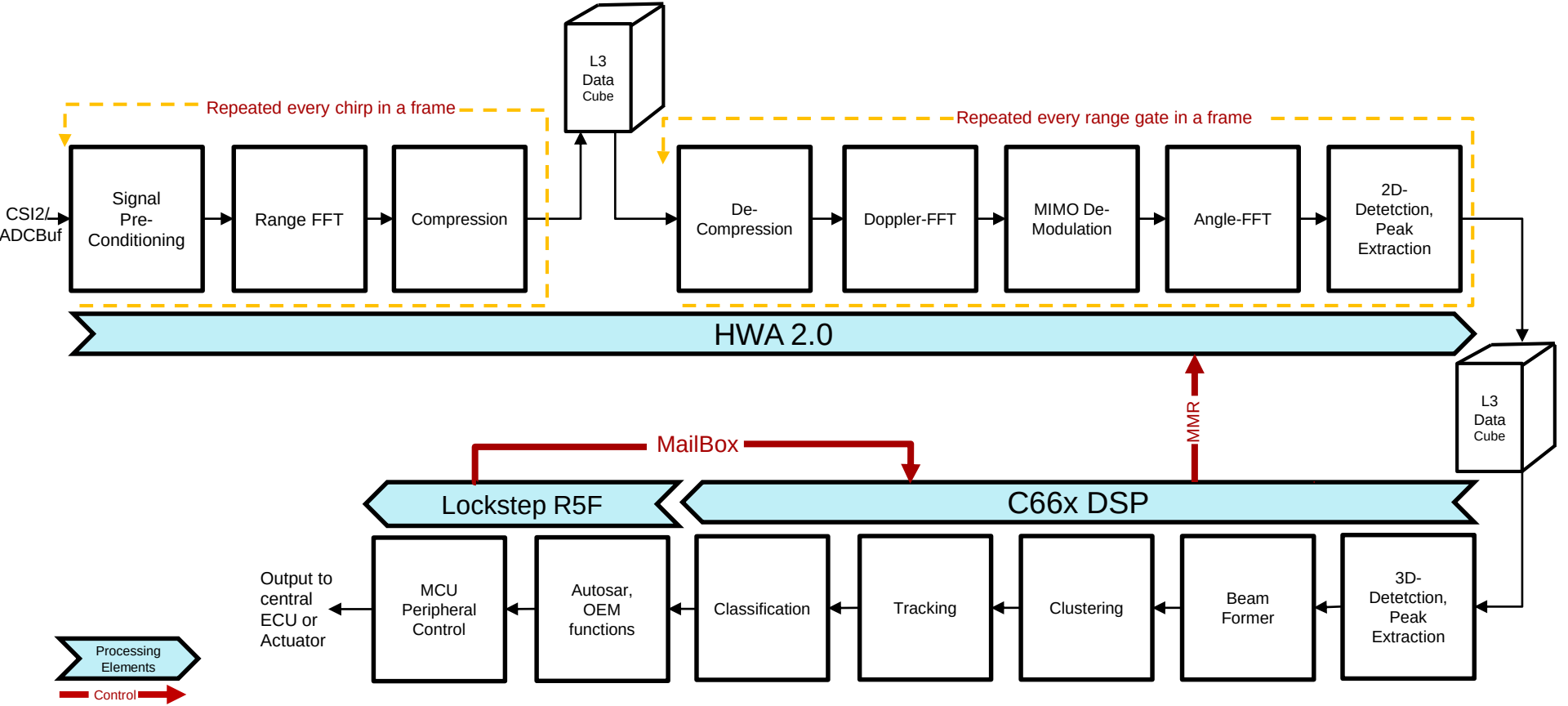


- The local maxima is used to find the maximum in a 2D-plane especially in the Doppler Angle dimension after angle-FFT.
- Usually for finding the local maxima of a cell under test (CUT), whose values after magnitude/log-magnitude (if the input is complex) are compared against the neighboring cells and the detection thresholds (row and column threshold which can be configured in registers or using advanced statistics).
- The output of the local maxima computations is stored into the destination memory as a bit pattern, where each bit indicates whether the specific sample/CUT was detected as a valid local peak or not.
- Supports wrap combination or circular shifting for edge cells in 2D plane if required.

Programming the HWA 2.0

- The HWA 2.0 can be programmed using the driver available in “mcu_plus_sdk”.
- HWA driver has multiple APIs and struct variables to implement functions of HWA2.0 with ease.
- Some of the basic APIs include HWA_init(), HWA_deinit(), HWA_open(), HWA_close(), HWA_reset() and HWA_enable().
- To configure the static and parameter-set registers of the HWA, we have two APIs of HWA_configCommon() and HWA_configParamSet() respectively.
- Also, there are APIs like HWA_paramSetDonePolling(), HWA_singleParamSetDonePolling() to query the finish of parameter-set and HWA_enableParamSetInterrupt(), HWA_disableParamSetInterrupt() etc. to issue interrupts to/from HWA from/to DSP/processor.
- In addition to the above APIs, there are multiple ones to configure/clear RAM, handle the operation of context-switching in HWA, trigger the DMA and also to read the status of registers.

Example use case of HWA 2.0



Comparison between HWA 1.0 vs HWA 2.0

	HWA 1.0	HWA 2.0
Operating Clock	200MHz	300MHz
Local RAM	16KB x 4 = 64KB	16KB x 8 = 128KB
Max FFT Size	1024(2^N only)	2048 (with 2^N and 3*2^N FFT's supported) along with 2D-FFT
Parameter-Set	16(16 x 8 x 32bits = 512B)	64 (4KB = 64 x 16 x 32bits)
Interference Statistics	Interference Zero-Out based on threshold (Statistics not taken into consideration)	Interference zero-out/Interpolation/zeroing with window (Statistics taken into consideration for maximum of 12 accumulations/iterations)
DC-correction estimation	-	DC estimation and correction for up to 12-iterations
CMULT mode	101-Complex scalar multiplier that remains constant	0101-Complex scalar multiplier that remains constant across all iterations or changes per iteration
CFAR	CFAR-CA	CFAR-CA and CFAR-OS
ACCEL-MODE	00 – FFT mode 01 – CFAR mode	000 – FFT mode, 001 – CFAR mode, 010 – Compression mode, 011 – Local Max Engine
Statistics	Supports basic statistics of maximum, sum etc.	Supports basic statistics of maximum, sum along with 2D-statistics of maximum, Histogram/CDF

References

1. Radar Hardware Accelerator 2.0 chapter in AWR294x TRM(SPRUIV5)
2. MCU PLUS SDK 8.x for HWA 2.0 driver and test application
3. Mmwave demo showcases HWA 2.0 features as part of DDMA processing chain

Thank You