EVE Architecture

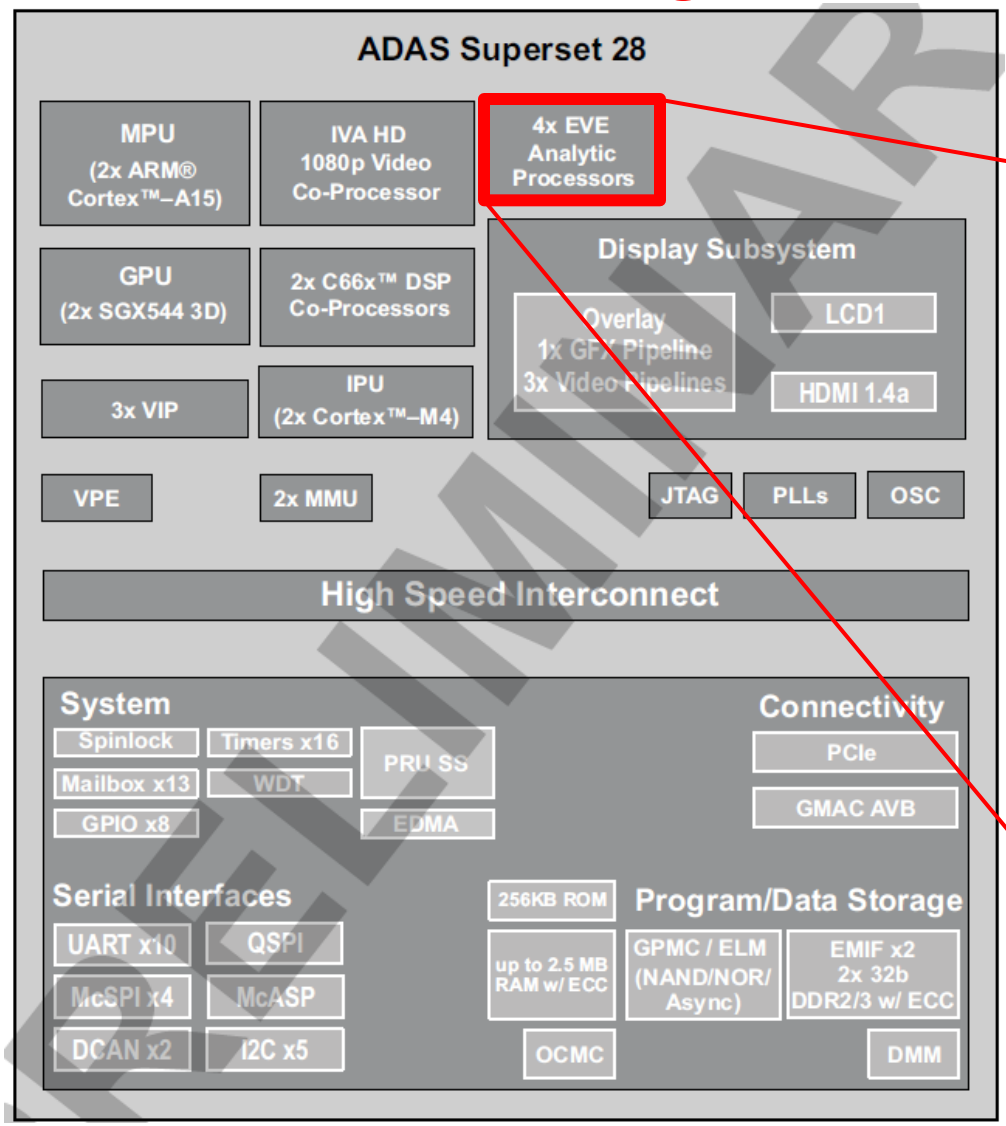
Contents

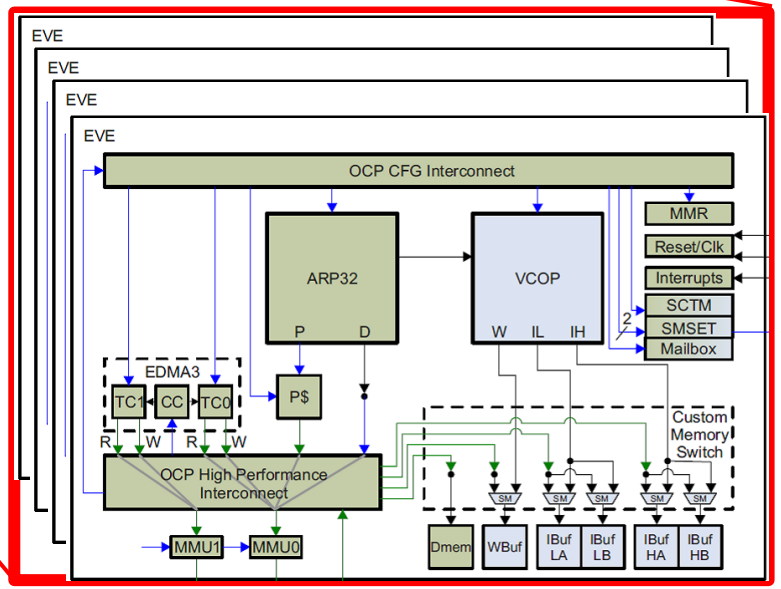
Vision Processing - Motivation for EVE

- **Low-level vision**
  - Examples: FIR filters, IIR filters, linear transforms, gradient, color space conversion, image integral, binary morphology.
  - Same mathematical operations applied to every pixel -> very SIMD friendly.
- **Mid-level vision**
  - Examples: Harris corners, Non Maximum Suppression, connected components, hough transform, sorting, clustering, optical flow, stereovision.
  - More than one math operations + ‘if’ conditions applied to every pixel -> SIMD friendly.
- **High-level vision**
  - Examples: object classifier, face recognition, object tracker.
  - Does not operate at pixel level but on information or features extracted from mid-level pixel processing.

How good?	DSP	EVE
Low-level	**	***
Mid-level	**	***
High-level	***	*

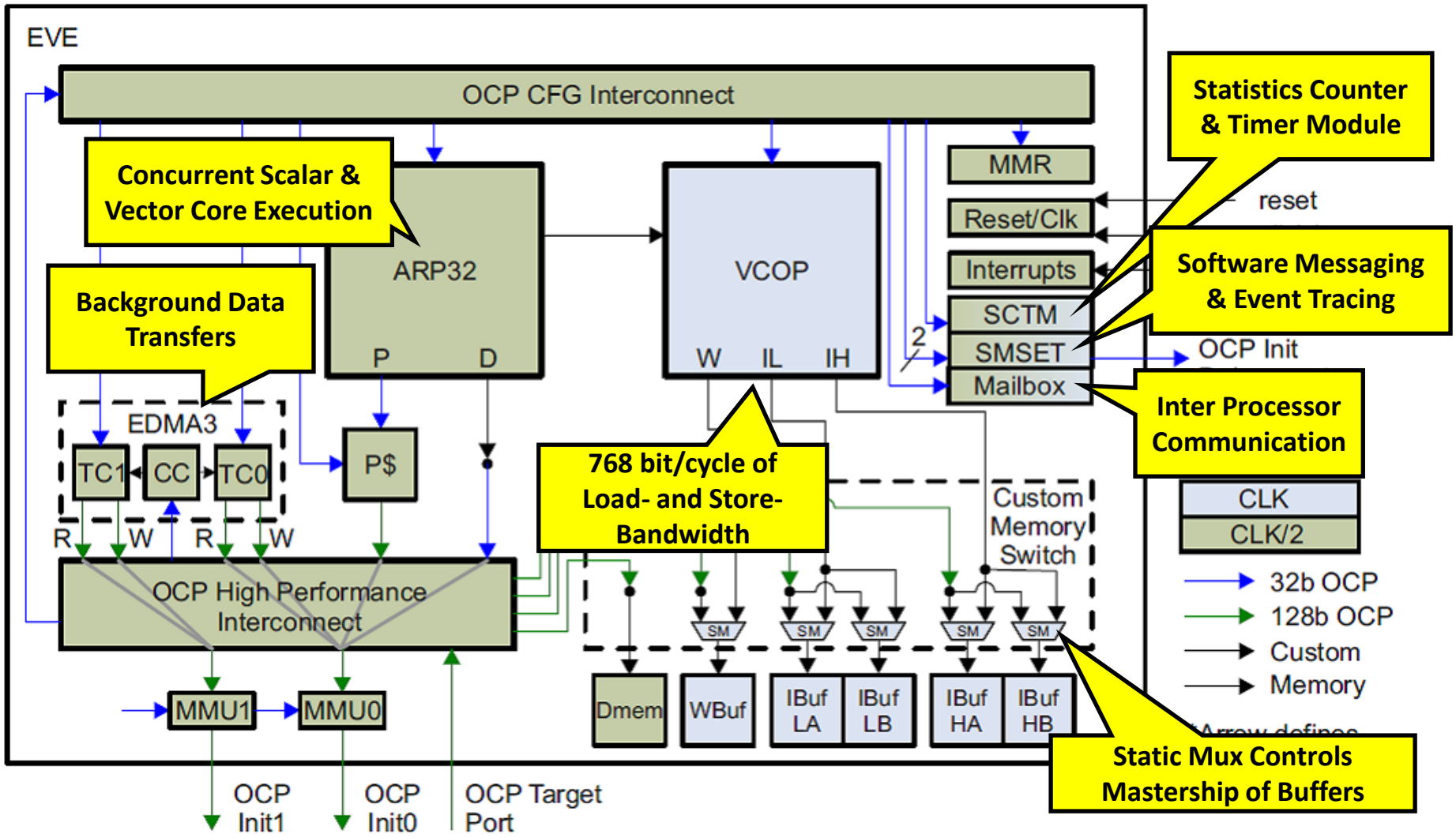
TDA2x Block Diagram





EVE – Elements and Topology

EVE – Elements and Topology



ARP32 Features

- Application Specific RISC Processor (ARP32) runs at EVE CLK/2
- Controls EVE subsystem, VCOP, EDMA3 and IRQ based on user code in C/C++
- Programs the VCOP to execute various kernels
- Runs concurrently with the vector core and DMA engine to maximize performance.
- Controls interaction with the host processor (ARM or C6000) using built-in mailboxes
- Program memory uses a direct-mapped program cache
- Data memory accesses are serviced by tightly coupled data memory block (DMEM).
- Accesses other memory blocks as well as both internal and external MMRs.
- Hardware loop acceleration for 2 levels of zero overhead nested loops.

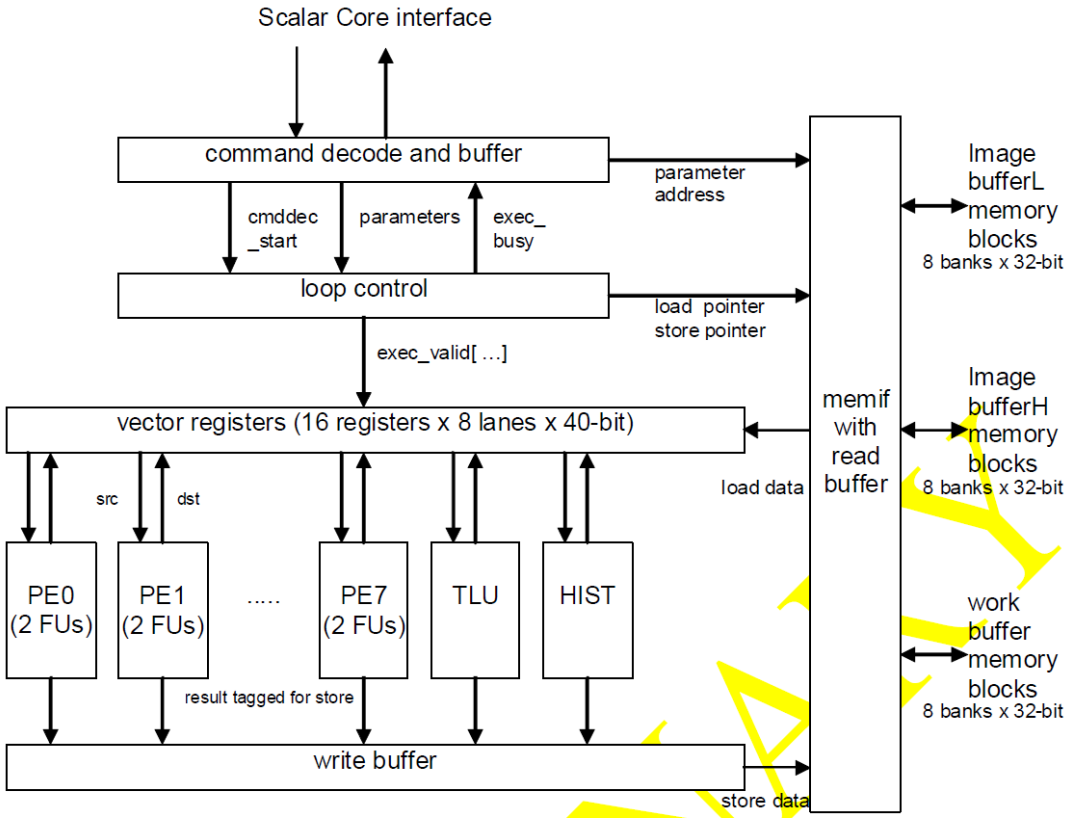
VCOP Features

- Vector co-processor (VCOP) runs at EVE CLK
- Implements algorithms without data dependencies across iterations
- 8-way SIMD engine with dual operation issue slots
- Operations on 8 bit, 16 bit and 32 bits with intermediate results in 40bit precision
- 16 16x16 bit multipliers, with add/subtract at 500 MHZ within 8 GMAC/sec throughput
- Support for built-in loop control with zero overhead for 4-levels of nested looping
- 8 address generation units to sustain addressing for all 4 loop levels
- Three memory interfaces IBUFL, IBUFH and WBUF, each supporting 256 bits / cycle
- Each memory interface is organized as 8 word-banks each of 32 bits
- Specialized pipelines for accelerating histogram and lookup tables
- Supports concurrent computation and background data movement
- Additional throughput from 3-src operations, 32 32-bit adds per cycle

VCOP Details

- Four nested for loops, with loop variables i1, i2, i3, and i4
- 8 address generators, each capable of 4-dimensional addressing
  i.e.: base + i1 × const1 + i2 × const2 + i3 × const3 + i4 × const4
- Two general-purpose functional units, fix-point, each 8-way SIMD
- 8 load units
- 8 store units
- Flat vs. Aliased view of EVE memory

- ➢ Generic compute
- ➢ Table Look-Up
  (up to 8 parallel lookups)
- ➢ Histogram and weighted histogram
  (up to 8 parallel histogram updates)

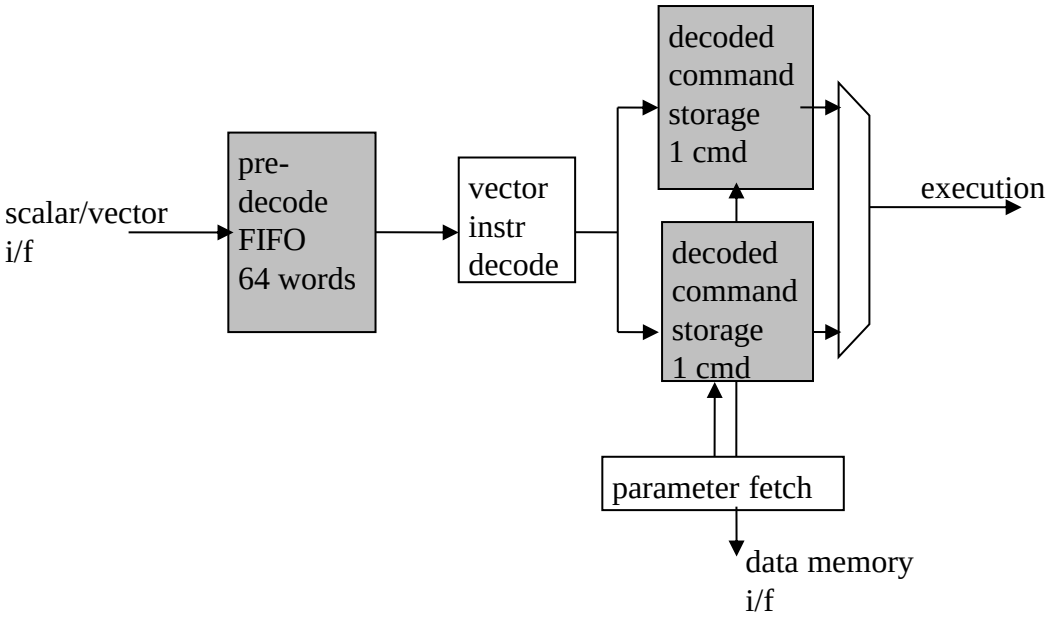


VCOP – Vector Commands

Each vector command can have

- Up to a maximum of 8 address generators
- Up to a maximum of 16 vector register initialization
- Up to a maximum of 8 loads
- Up to a maximum of 40 operations allowed only in innermost loop
- Up to a maximum of 8 stores with rounding and saturation

The maximal command length for the vector core is thus 81 instructions (VLOOP + 16 VINITs + 8 VAGENs + 8 VLDs + 40 operations + 8 VSTs).



VCOP : Per Loop Overheads

Overhead components are:

- Parameters can be read from data memory (wbuf, ibufL, or ibufH).
- Command decode time can be hidden if this is not the first in a set of loops.
- Execution pipeline has 15 cycles of ramp up/down time..
- Lookup Table and histogram loops, we have a longer pipeline, and overhead is 23 cycles.

Overhead component costs are:

- 1. Decode time: two cycles per instruction to decode + 4 cycles
- 2. Parameter fetch time: 9 + ceiling(num_param/16) cycles.

- Total execution time for first loop = Command decode + Parameter fetch + Execution time.
- Total execution time for back to back loops = Parameter fetch + Execution time

EVE Memory Blocks

Memory	Organization	Size	Function
PMEM	1024 x 256	32 KB	ARP32 program cache
DMEM	2048 x 128	32 KB	ARP32 data memory
WBUF	8 x 1024 x 32	32 KB	VCOP working buffer
IBUFLA	8 x 512 x 32	16 KB	Image buffer low copy A
IBUFLB	8 x 512 x 32	16 KB	Image buffer low copy B
IBUFHA	8 x 512 x 32	16 KB	Image buffer high copy A
IBUFHB	8 x 512 x 32	16 KB	Image buffer high copy B

REFERENCES

REFERENCES

- *Section 8.1 Embedded Vision Engine (EVE) Subsystem in ADAS Superset 28 ADAS Applications*
- *Processor Technical Reference Manual (SPRUHK5)*
- *Section 8.2 ARP32 CPU and Instruction Set in ADAS Superset 28 ADAS Applications Processor*
- *Technical Reference Manual (SPRUHK5)*
- *Section 8.3 VCOP CPU and Instruction Set in ADAS Superset 28 ADAS Applications Processor*
- *Technical Reference Manual (SPRUHK5)*
- *VCOP Kernel-C Reference Guide (SPRUHB9)*
- *ARP32 Assembly Language Tools Reference Guide (SPRUH23)*
- *ARP32 Compiler Reference Guide (SPRUH24)*

Thank You