Hardware Accelerators for Space Applications

Presenter: Lucana Santos, with contributions from TEC-EDM

EDHPC, Elche, 13/10/2025

ESA UNCLASSIFIED – For ESA Official Use Only

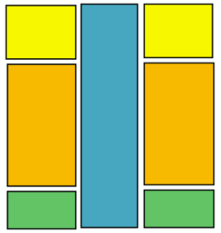
Introduction

- Overview of solutions to accelerate algorithms in space by parallelization
- Identification of challenges and potential solutions.

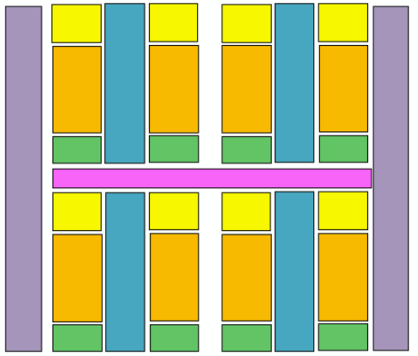
NB: Wide topic, presentation is more qualitative than quantitative – details in the sources/bibliographic references

Why Accelerators?

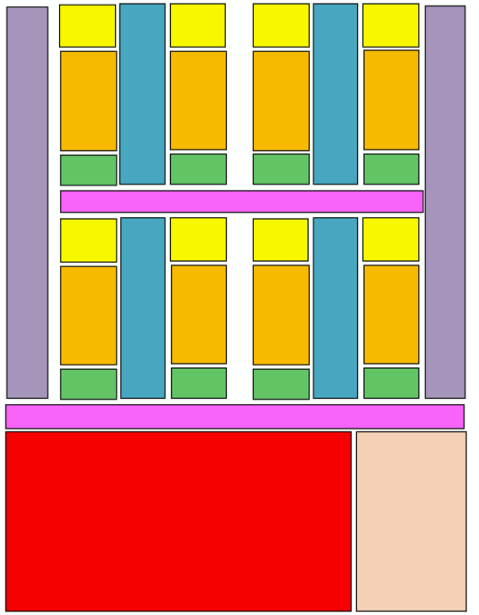
Single core



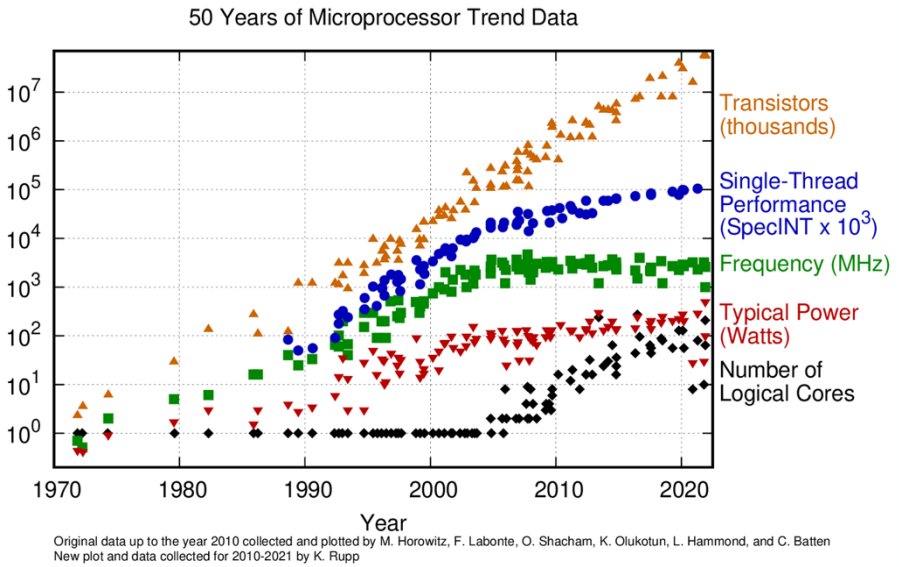
Multi core

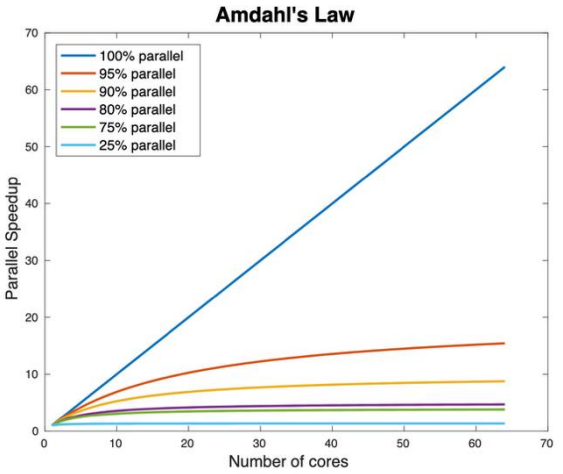


Heterogeneous



Next (?)





Local memory | Shared memory | Programmable logic | Accelerator | Processor | ALU | Interconnect | External I/F

Hardware Accelerators

Generic

- Instruction Set Extensions
- Processing in Memory (PIM)
- Systolic Arrays
- Vector Processor

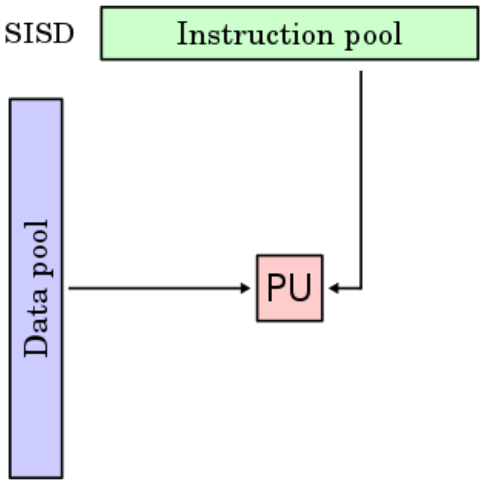
Application driven

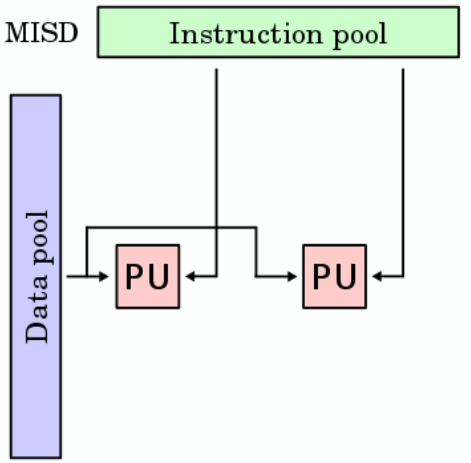
- Graphics Processing Units (GPUs)
- Video Processing Units (VPUs)
- Digital Signal Processors (DSPs)
- Tensor Processing Unit (TPUs)
- AI Engine
- Neuromorphic Processor

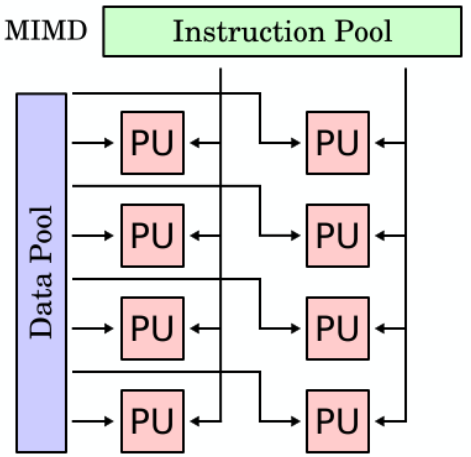
Application specific

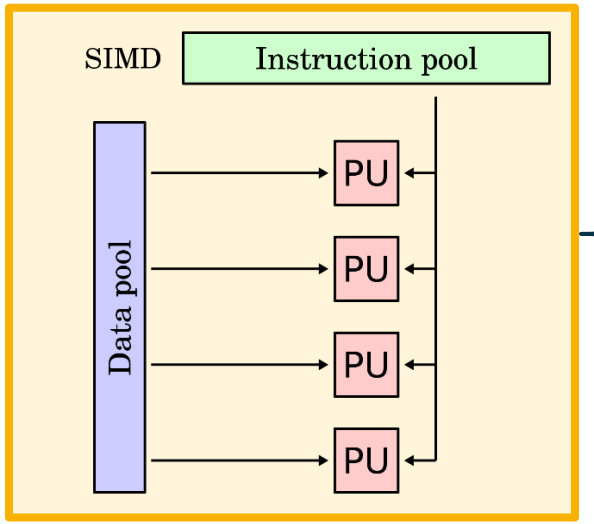
- Dedicated cores: GNSS, Image/video compressor, encryption

Parallel Computation Paradigm (Flynn's Taxonomy)









- Array processor or vector processor

AKA SIMT. Each PU has its own independent memory and register file

- Pipelined processor

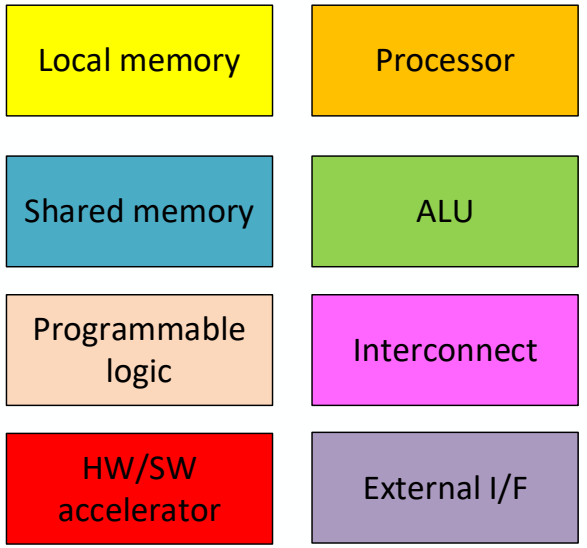
Data is read from a central resource, fragmented and processed by the PU to then store results in the same central resource.

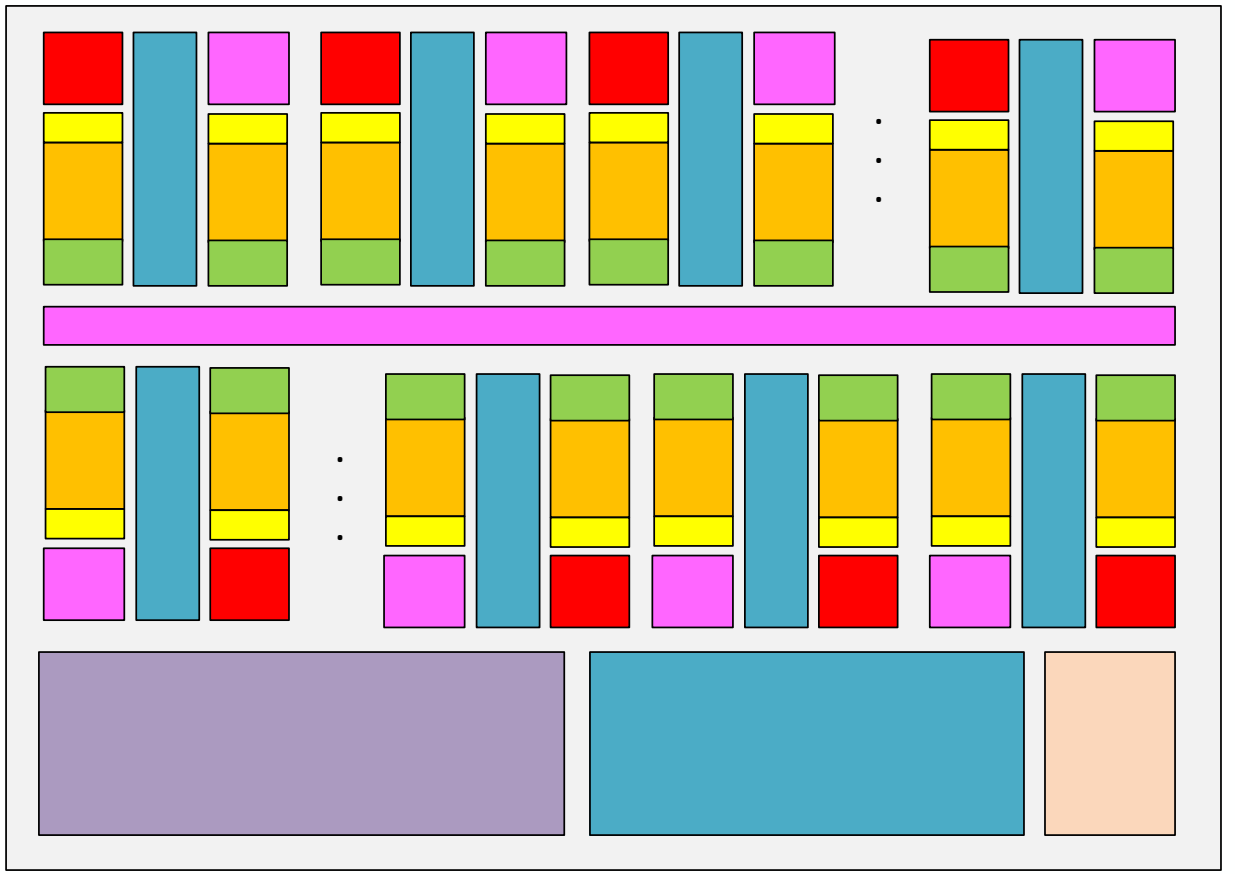
- Associative processor

AKA predicated SIMD, PU can make an independent decision based on local data.

Heterogeneous Computing: Hardware Accelerators

- **Multicore Systems with accelerators (GPU/ AI Core/ eFPGA) and ISA extensions (vector, packed).**
- FPGA with hard IPs (GPU/ AI Core/ Multicore processor).
- GPGPU

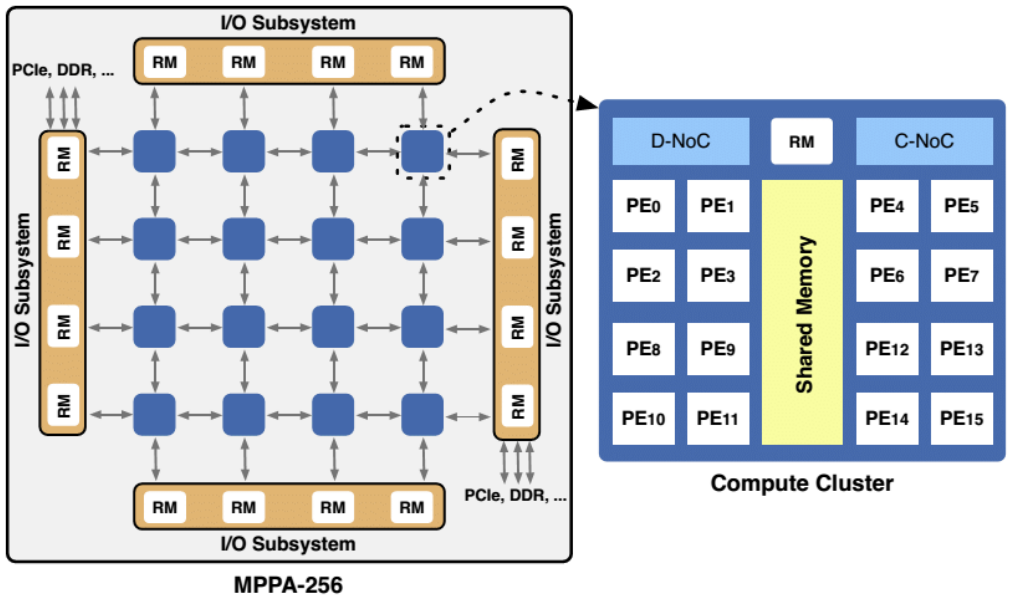




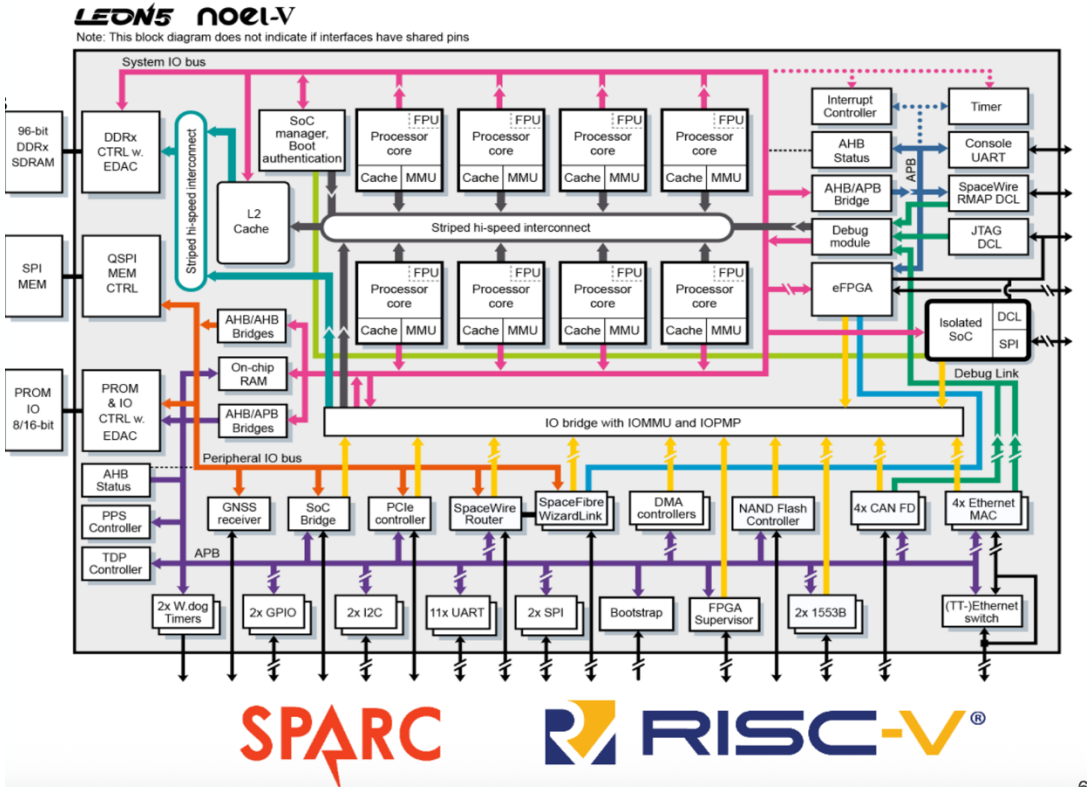
Heterogeneous Computing: Hardware Accelerators

- **Multicore Systems with accelerators (GPU/ AI Core/ eFPGA) and ISA extensions (vector, packed).**
- FPGA with hard IPs (GPU/ AI Core/ Multicore processor).
- GPGPU

Kalray MPPA



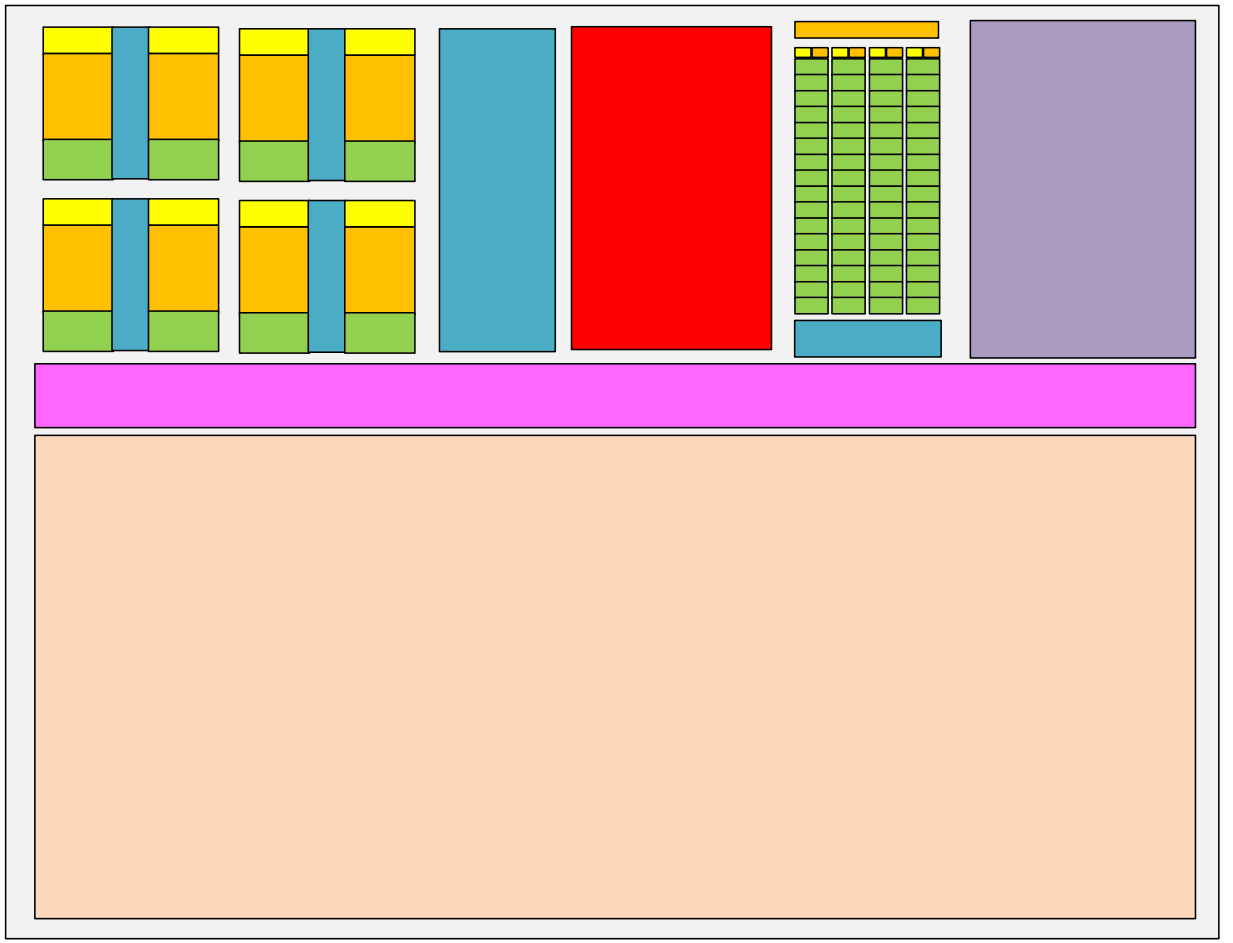
GR765



Heterogeneous Computing: Hardware Accelerators

- Multicore Systems with accelerators (GPU/ AI Core/ eFPGA) and ISA extensions (vector, packed).
- **FPGA with hard IPs (GPU/ AI Core/ Multicore processor).**
- GPGPU

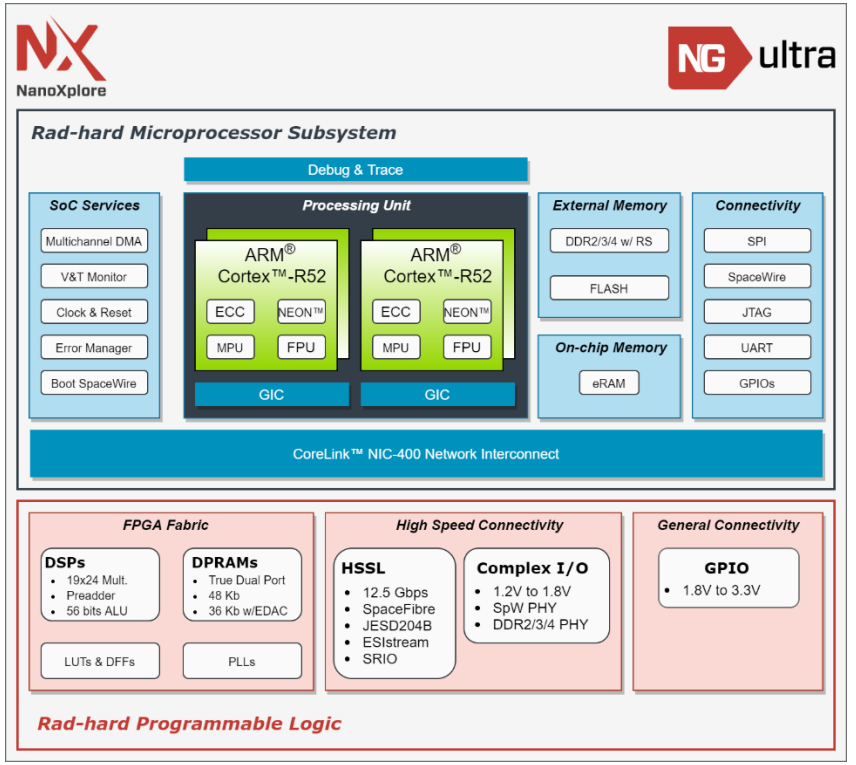
Local memory	Processor
Shared memory	ALU
Programmable logic	Interconnect
HW/SW accelerator	External I/F



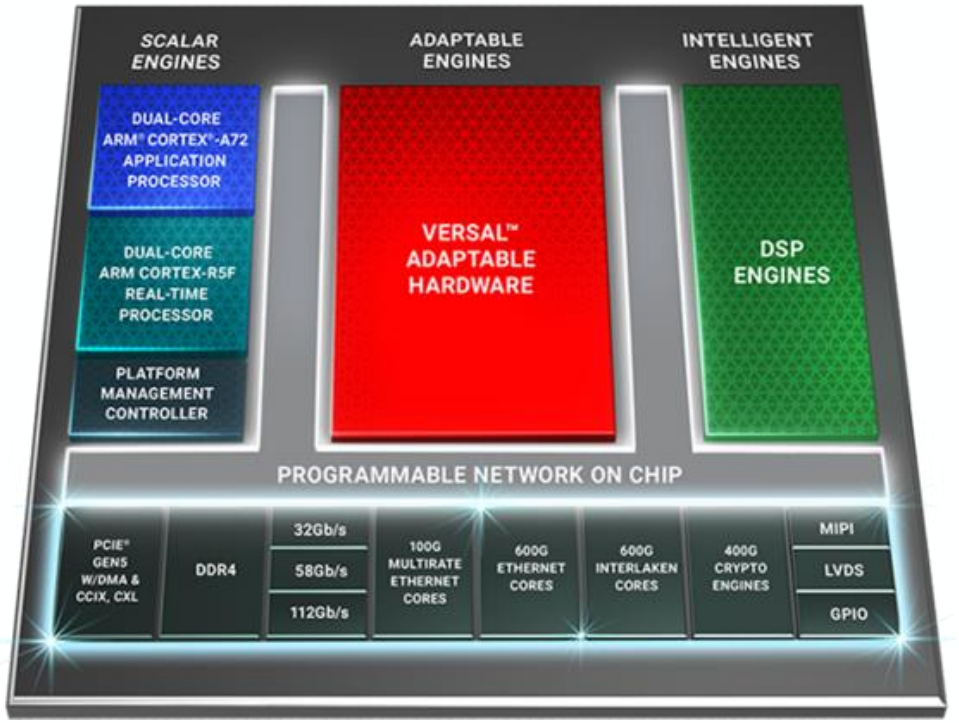
Heterogeneous Computing: Hardware Accelerators

- Multicore Systems with accelerators (GPU/ AI Core/ eFPGA) and ISA extensions (vector, packed).
- **FPGA with hard IPs (GPU/ AI Core/ Multicore processor).**
- GPGPU

NanoXplore NG-ULTRA



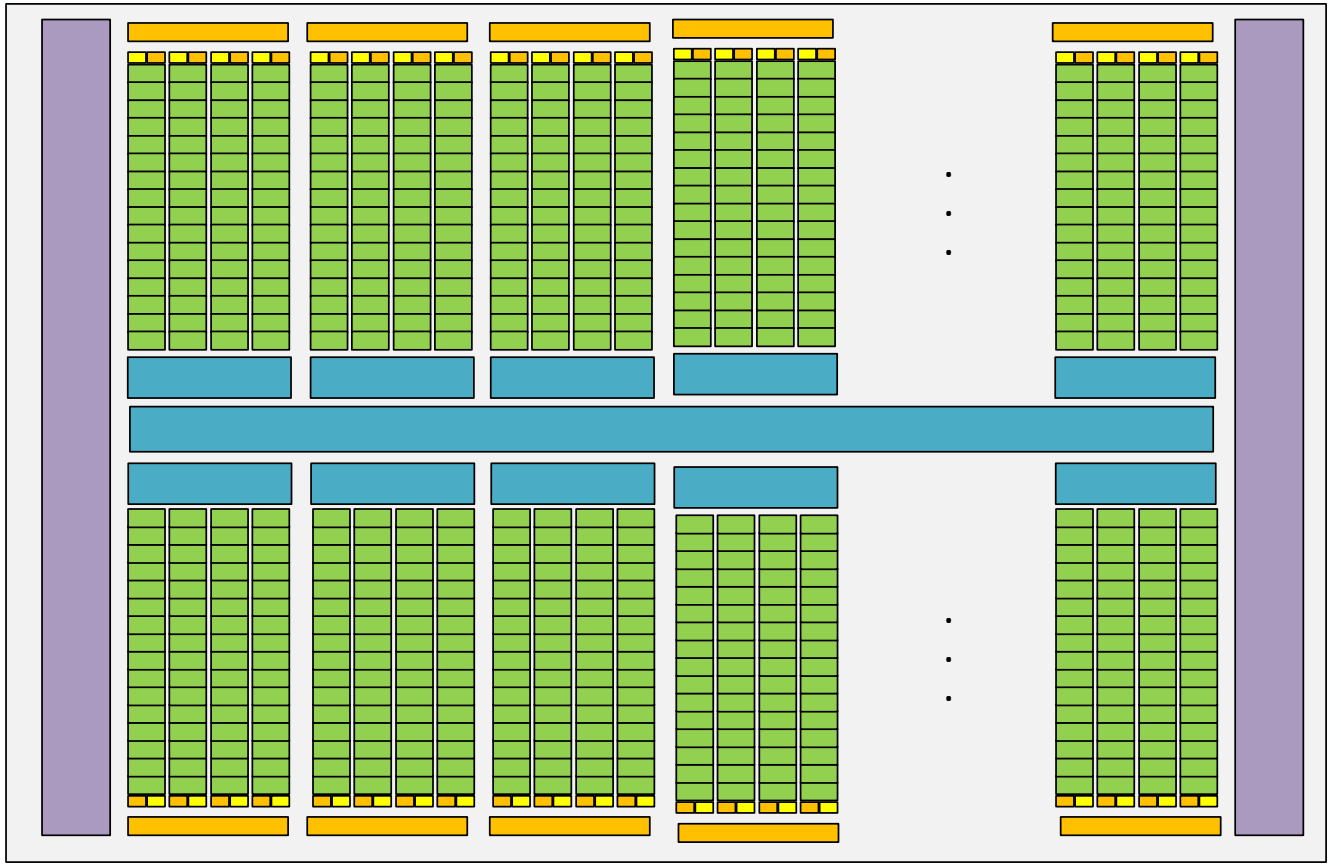
Xilinx Versal XQR



Heterogeneous Computing: Hardware Accelerators

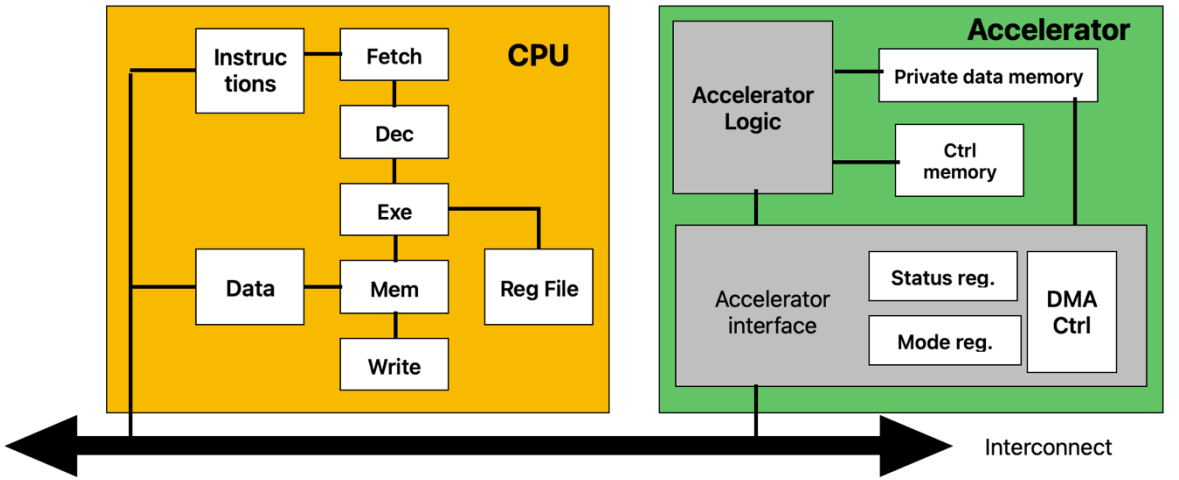
- Multicore Systems with accelerators (GPU/ AI Core/ eFPGA) and ISA extensions (vector, packed).
- FPGA with hard IPs (GPU/ AI Core/ Multicore processor).
- **GPGPU**

Local memory	Processor
Shared memory	ALU
Programmable logic	Interconnect
HW/SW accelerator	External I/F

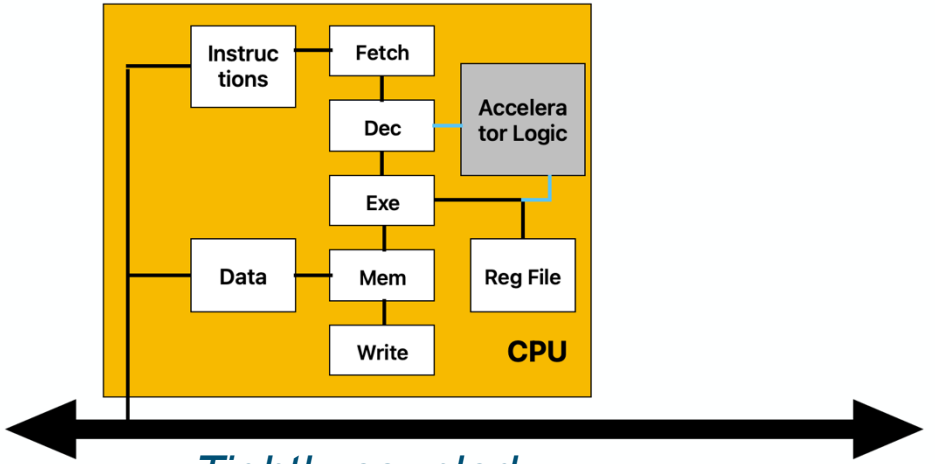


Tighly vs Loosely Coupled Accelerator

- Defines how the accelerator architecture is exposed to higher level programming model.
- It is crucial to ensure workloads are deployed on the accelerators quickly.
- **Loosely coupled architecture (LCA)**
  - Example: GPU
  - Communicates with processor through interconnect interfaces. (AXI, AHB, NoC)
- **Tightly coupled architecture (TCA)**
  - Example: Vector processor for ISA extensions
  - Accelerator can share processor key resources such as Register File, MMU, L1 cache.



Loosely coupled



Tightly coupled

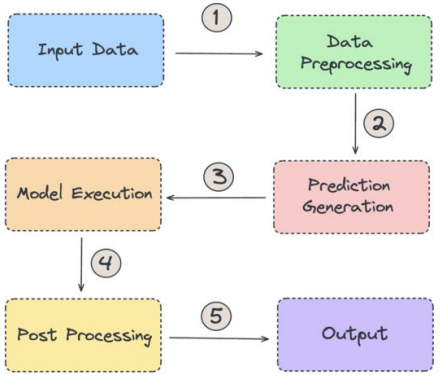
Software Ecosystem

Software ecosystem is key to ensure efficient use of HW accelerators.

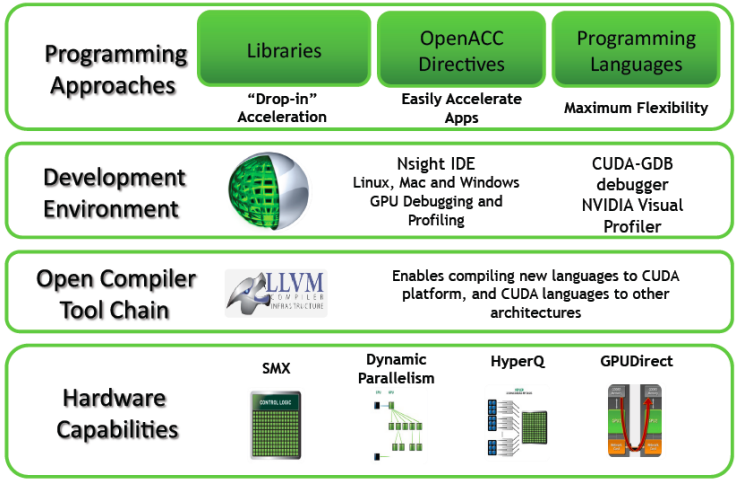
Simulators, object toolchain, debugger, compilers and libraries, bootloaders, monitors…

…and it increases in complexity!

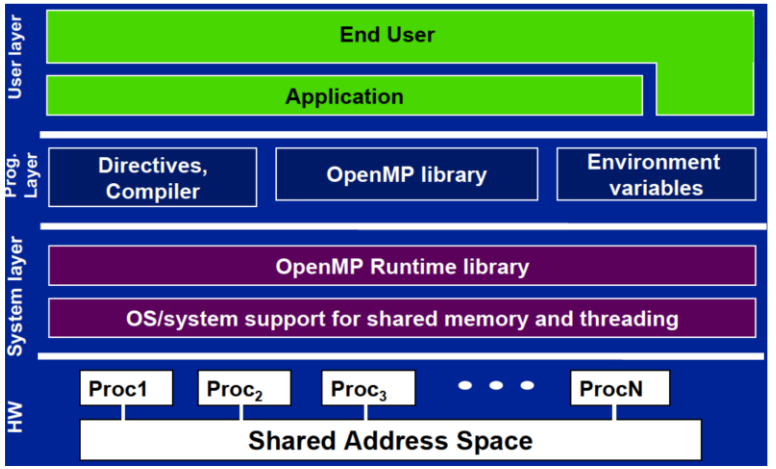
AI Inference Flow



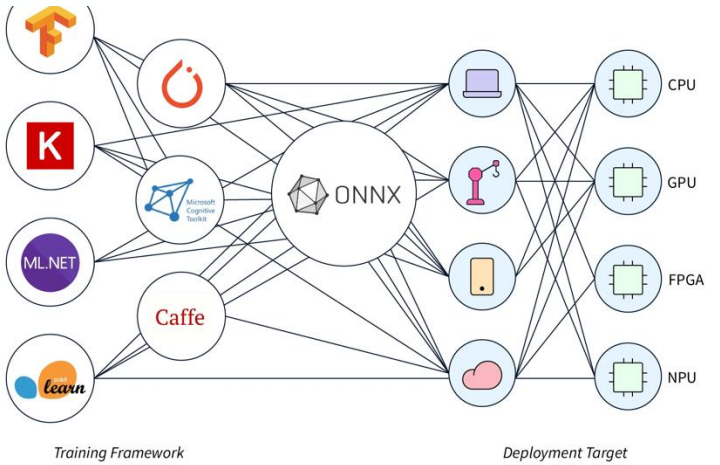
GPU Programming Models



OpenMP Stack



Open Neural Network Exchange (ONNX)



Space Specific Technical Constraints

Reliability

- **Radiation**
  - Total ionising dose (TID) and single event effects (SEU)
  - Commercial off-the-shelf (COTS) → needs expensive testing and system level mitigation
  - Radiation hardening by design (RHBD), redundancy
- **Vacuum**
  - Outgassing from package materials → contamination of sensors
- **Vibration**
  - At launch → stress on interconnects
- **Temperature**
  - Satellites can be very hot and very cold (-55 to 125 deg C) → challenge for timing closure.

Lifetime of satellites up to 20 years longer than commercial electronics → No repair!

Functional

- **Determinism**
  - Needed by timing critical control functions (launch, landers)
- **Mixed criticality software**
  - Different levels of trust, time and space partitioning for fault containment.
- **Specific interfaces**
  - MIL-1553B
  - SpaceWire
  - SpaceFibre
  - Time Sensitive Networking (TSN) / TimeTriggered-Ethernet (TTE)
- **Low power (TOTAL and operations/watt)**
- **Security**
  - PQC Encryption, root of trust, authenticated boot

Benchmarking/Metrics

- **CPU single core metrics**
  - Clock frequency
  - Core Count
  - Cache size
  - Thermal design power
  - Instructions per cycle (IPC)
  - Mega instructions per second (MIPS)
- **Accelerator (AI, GPU)**
  - Trillions of Operations per Second (TOPS) [int8, int16, int32]
  - Floating Point Operations Per Secod (FLOPS)) [FP32, FP64]
- **Synthetic benchmarks**
  - Dhrystone (integer) – DMIPS or DMIPS/Hz
  - Whetstone (floating point)
  - CoreMark, CoreMark PRO
- **Application benchmarks**
  - OBPMark & OBPMark-ML: https://github.com/OBPMark
- **How to compare two solutions?**
  - Speedup of execution time
- **Note:**
  - Keep track of anything affecting performance, compiler version, compiler flags…

Instruction Set Extensions

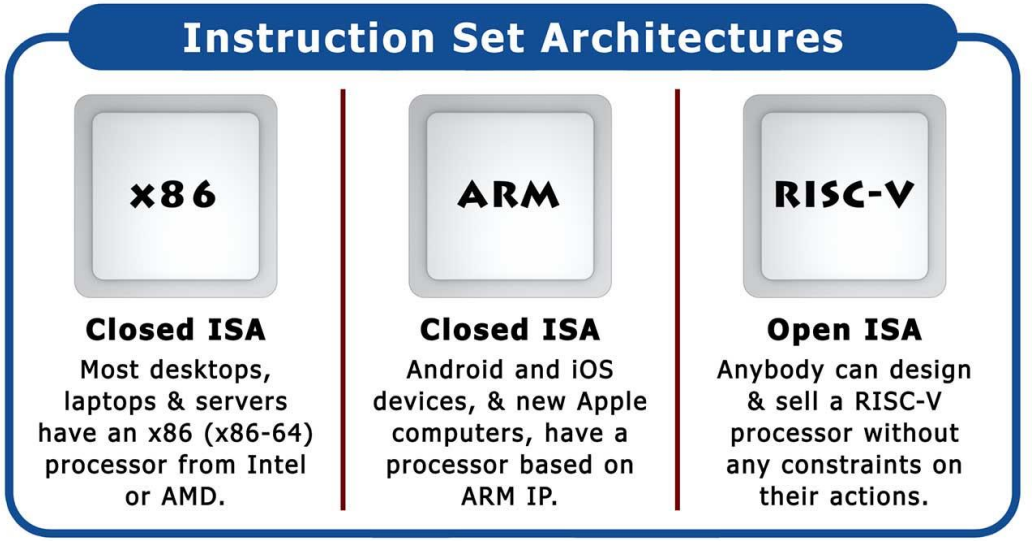
ESA UNCLASSIFIED – Releasable to the Public

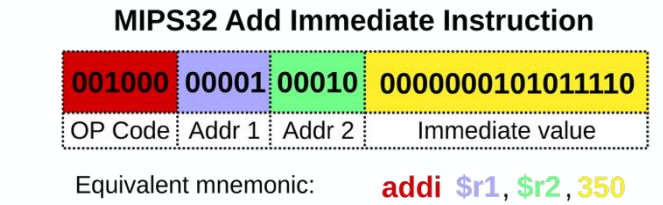
ISA – Instruction Set Architecture

ISA refers to **the set of instructions that a computer processor can understand and execute**.

Defines the instructions, data types, registers, and the programming interface for managing main memory such as addressing modes, virtual memory, and memory consistency mechanisms. The ISA also includes the input/output model of the programmable interface.

When an **Instruction Set Architecture (ISA)** is described as **open**, it means that its **specifications are publicly available and free to use**, without licensing fees or proprietary restrictions.

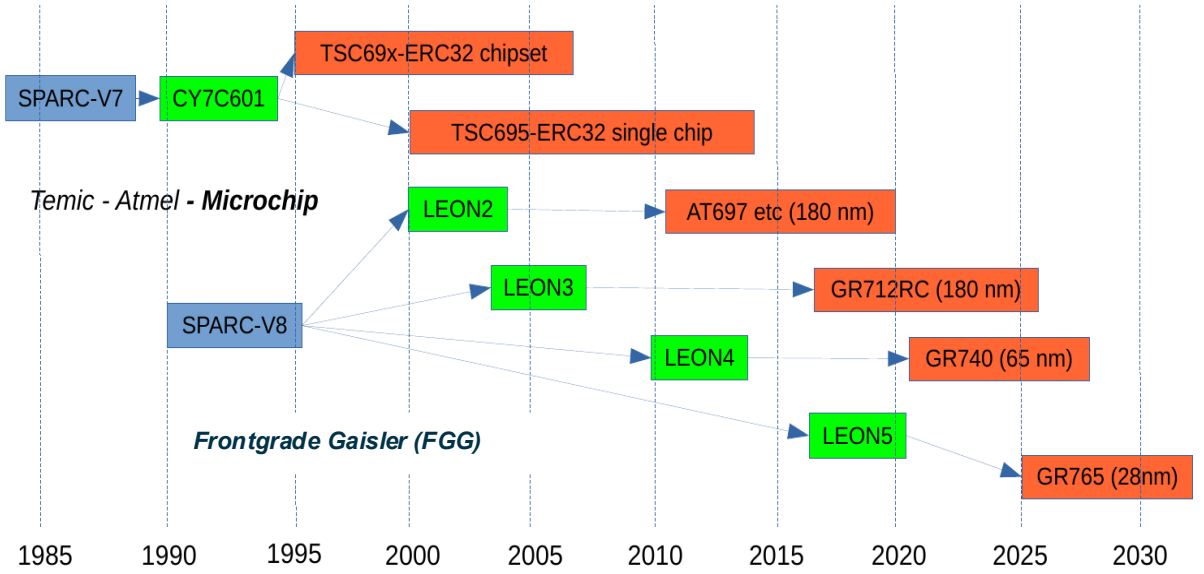




Why Open ISA for Space?

- Non-dependence: multiple IP sources, or own development.
- Free and open standard, allowing for customization.
- Easier to modify for radiation hardening when compared with proprietary ISA.
- Industry and academia in the game: commercial and open source.
- Stable specification, low complexity, GNU tool chain.

Open ISA allows for competition of multiple IP sources → warrants for flexibility and non-dependence



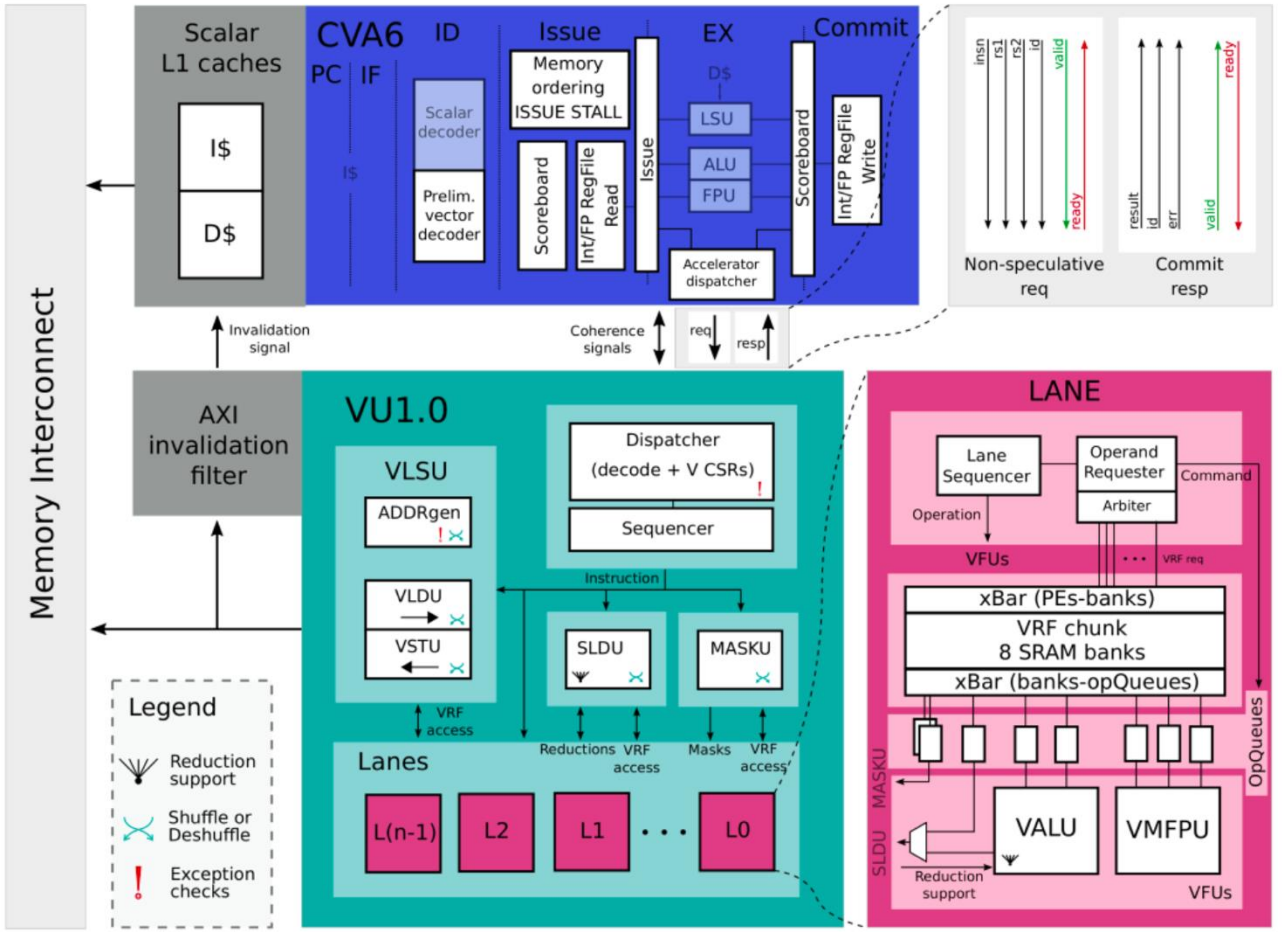
RISC-V ISA Extensions

ISA Extension	Notes
I/E	Instructions for basic Integer operations. This is the only extension that is mandatory. I requires 32 registers, E requires only 16.
M	Instructions for multiplication and division
C	Compact instructions that have only 16bit encoding. This extension is very important for applications requiring low memory footprint.
F	Single-precision floating-point instructions
D	Double-precision floating-point instructions
A	Atomic memory instructions
B	Bit manipulation instructions. The extension contains instructions used for bit manipulations, such as rotations or bit set/clear instructions.
V	Vector instructions that can be used for HPC.
P	DSP and packed SIMD instructions needed for embedded DSP processors.

RISC-V P and V Extensions

	P-extension (Packed-SIMD)	V-extension (Vector-SIMD)
Target	Array processors (parallel ops via multiple elements/partitionable datapath)	Vector processors (reuse processing elements over multiple cycles, scalable lanes)
Registers used	Reuses **general-purpose registers** (RV32I: 32×32-bit, RV64I: 32×64-bit)	Dedicated **vector registers** (32 registers, min 128-bit wide, scalable)
Parallelism	Limited by scalar register width (fewer elements in RV32I vs RV64I)	Scales with vector register width (≥128b, can be grouped for larger ops)
Instruction encoding	Separate instructions per element size (e.g., add8, add16, add32)	Element size selected dynamically (8–64b required, up to 1024b reserved)
Floating-point	Not supported (fixed-point only)	Supported (16-, 32-, 64-bit FP if scalar supports it)
Datapath reuse	Shares scalar datapath → saves area/power	Requires separate vector datapath (ALU, multiplier, FPU) → more complex
Scalability	Compact, lower-power, suitable for small designs	High performance, scalable with wider vectors & more lanes
Area & Power	Lower area, lower power (no FPU, no extra registers)	Higher area/power (extra vector register file, duplicate units, register file transfers)
Use cases	Embedded, low-power, simpler SIMD workloads	HPC, ML, DSP, floating-point heavy, scalable high-performance workloads

Vector Extensions



Architecture of the ARA accelerator as a co-processor with CVA6

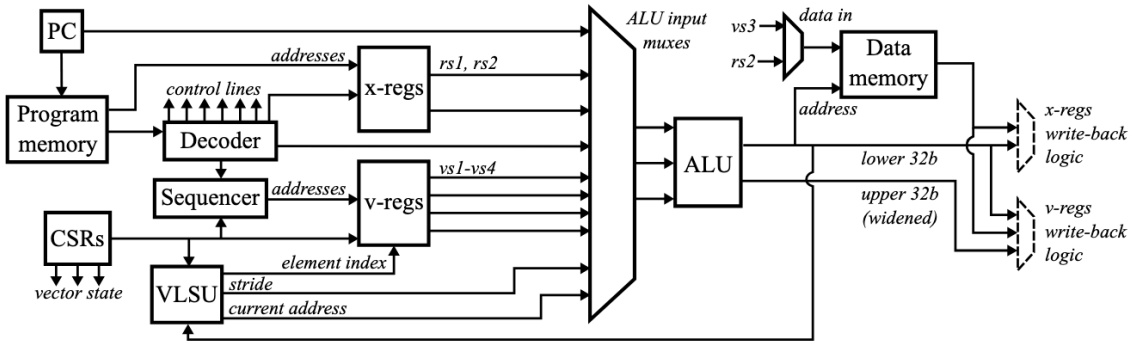
Assembly without vector extensions

```
vsetvli     x0, x0, e8, m1, ta, ma
;                       ^^
;                       m1 - use a single register

; perform 8 vector additions
vadd.vv     v24, v0, v8
vadd.vv     v25, v1, v9
vadd.vv     v26, v2, v10
vadd.vv     v27, v3, v11
vadd.vv     v28, v4, v12
vadd.vv     v29, v5, v13
vadd.vv     v30, v6, v14
vadd.vv     v31, v7, v15
```

Assembly with vector extensions

```
vsetvli     x0, x0, e8, m8, ta, ma
;                       ^^
;                       m8 - use groups of 8 registers

; perform 8 vector additions
vadd.vv     v24, v0, v8
```

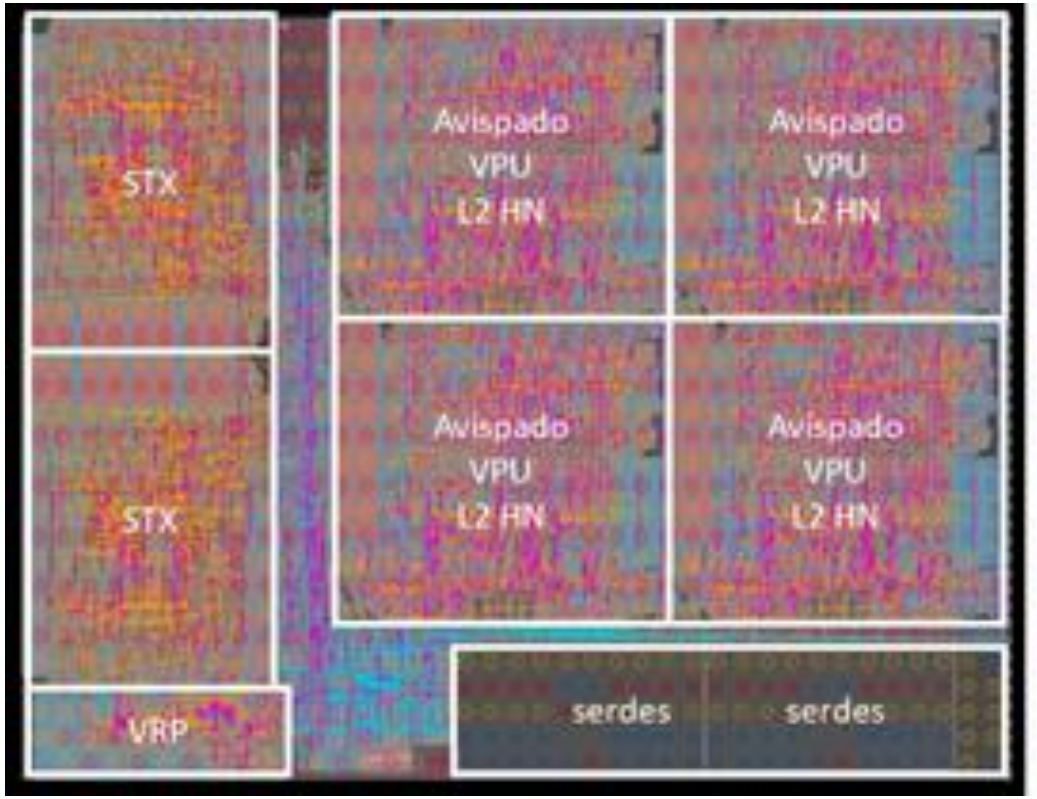
Block diagram of a vector processor



M. Johns and T. J. Kazmierski, "A Minimal RISC-V Vector Processor for Embedded Systems," *2020 Forum for Specification and Design Languages (FDL)*, Kiel, Germany, 2020, pp. 1-4

EPAC

- European Processor Initiative (EPI)
- Four RISC-V Vector (RVV) tiles composed of the scalar, two-way in-order Avispado core and 8-lane Vector Processing Unit (VPU) implementing v0.7 of RISC-V Vector extension ISA.
- Two STX tiles consisting of Stencil/Tensor Accelerator cores.
- One VRP tile consisting of Variable floating point precision core.

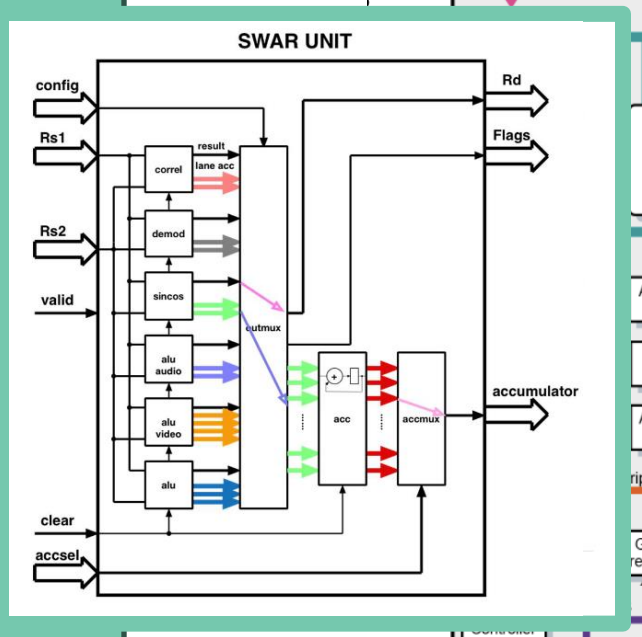


EPAC1.0 Test Chip in GF22 Technology

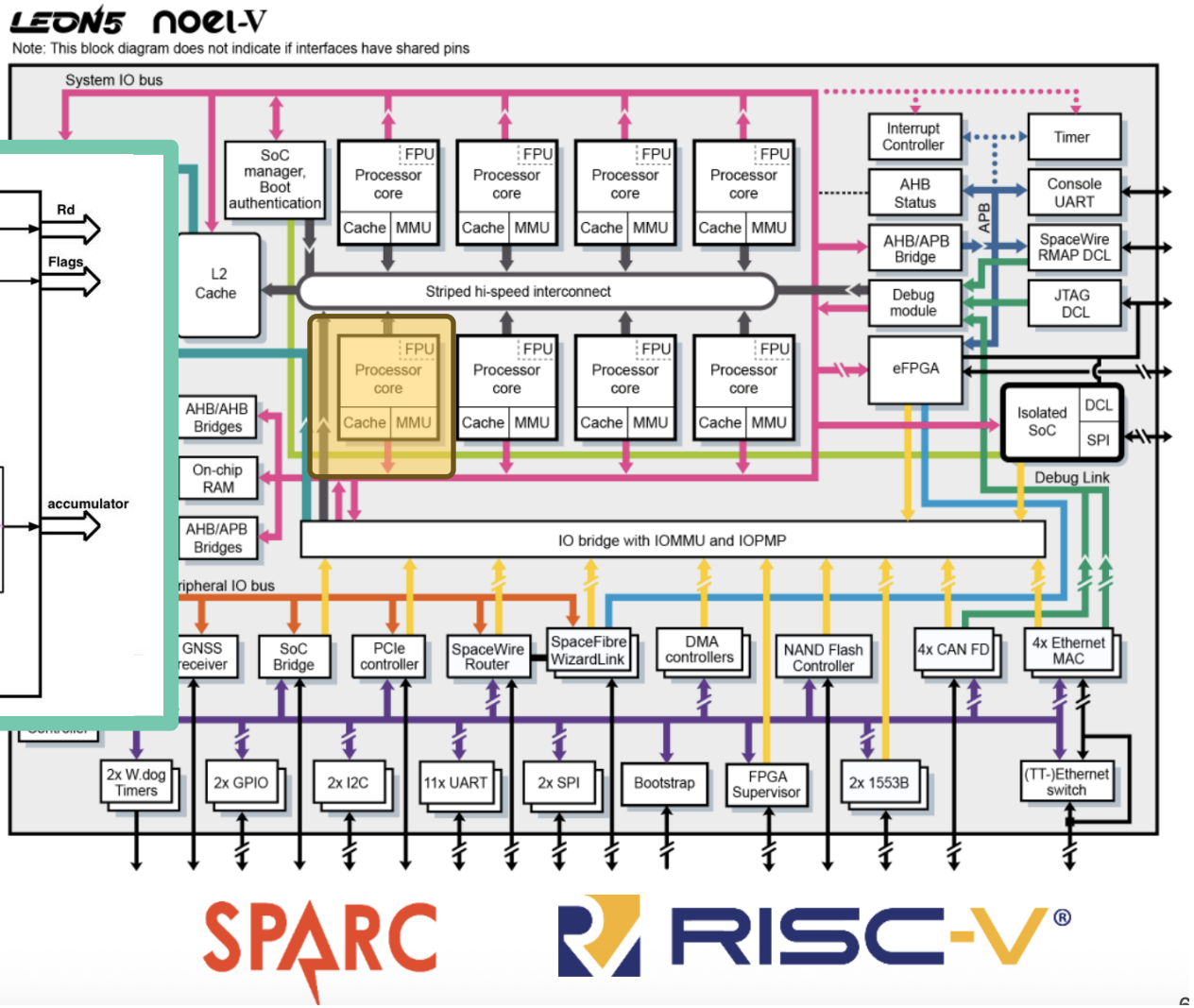
SIMD Within A Register SWAR for RISC-V and LEON5

- Octacore LEON5 SPARC or **NOEL-V** RISC-V in 28nm SOI
- SIMD extension (**SWAR**, daiteq s.r.o.)

- SIMD within a register (SWAR) extensions enable the use of packed instructions.
- Computation of a single instruction on multiple low-resolution operands without penalties.
- Targeted applications:
  - GNSS: 16x2b words; 10x3b words; 8x4b words
  - Audio: 2x16b words \
  - **Video/image: 4x8b words**
- New support: data compression; AI



SWAR extensions available as ESA IP Core



ISA Extensions – Programming

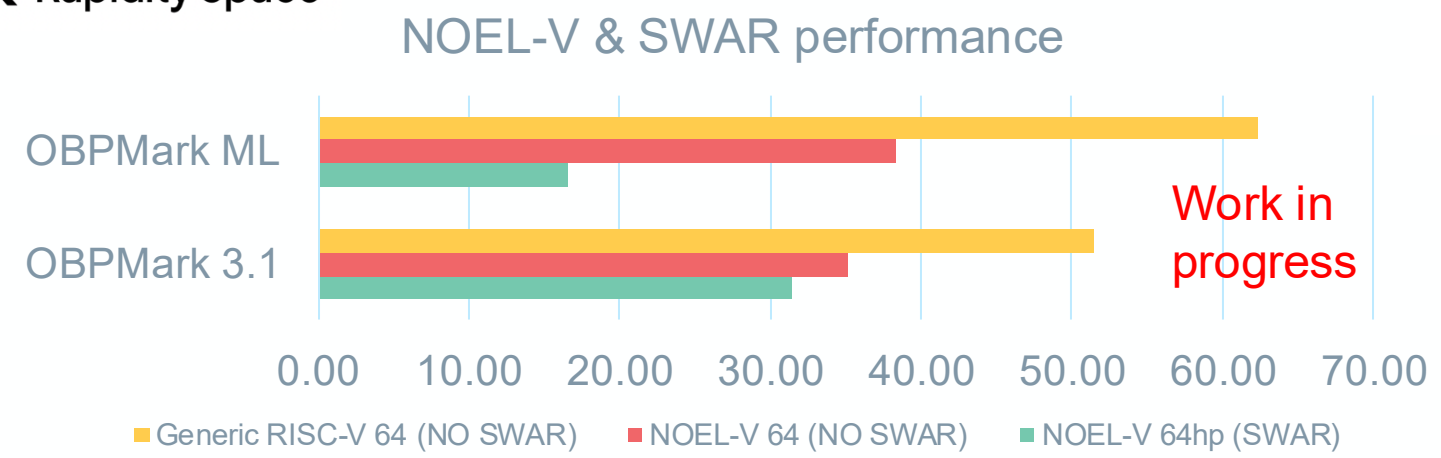
Hardware Side

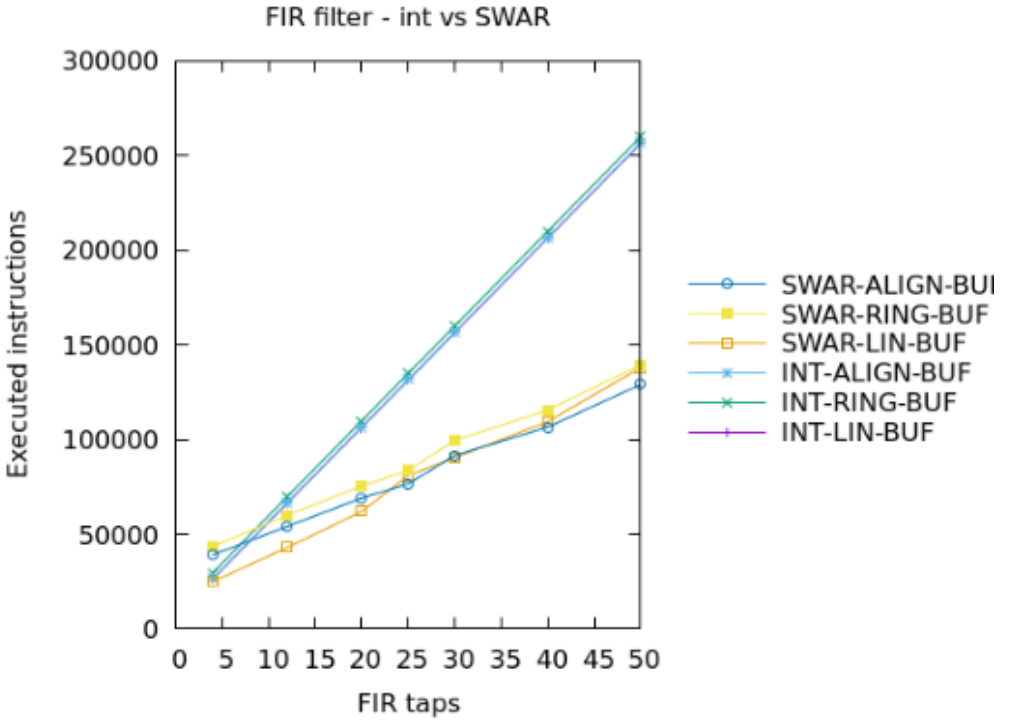
- Extensions need to exist in the processor architecture.
- Functional Unit design, RTL implementation…

Toolchain Side

- Add opcodes and types.
- Add compiler support to recognise new instructions.

Rapidity Space

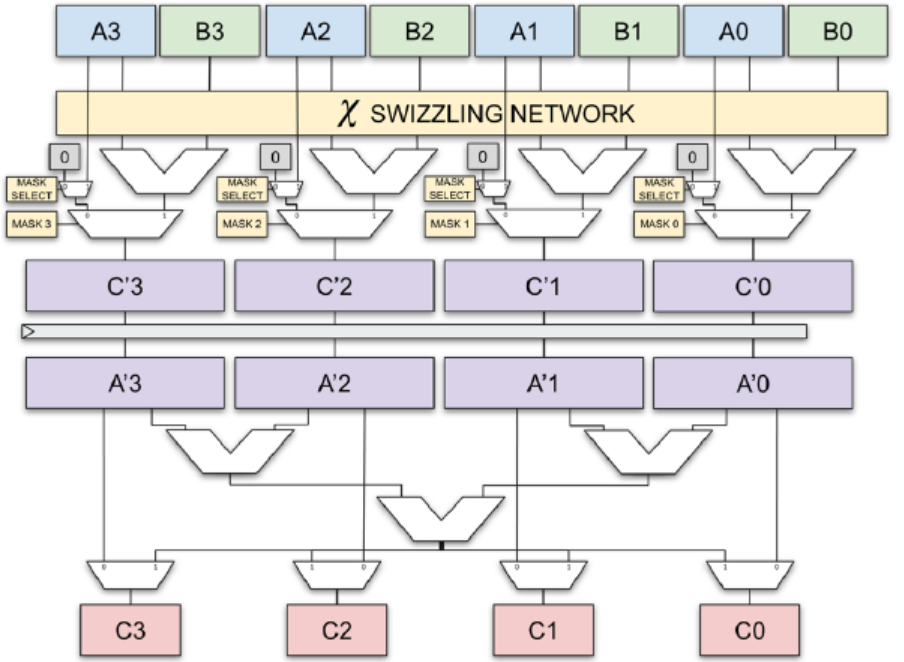




FIR Filter – Exec. Instructions for integer vs SWAR. With linear buffer, ring buffer, ring buffer aligned to 2^n.

SPARROW – SIMD Unit for AI Acceleration

- Increase the machine learning processing capabilities of space processors.
- Portable, open-source design, integrated with CAES processors NOEL-V (RISC-V) and LEON 3 (SPARC) High-performance, Low-cost targeting both ASIC and FPGA implementations, at least 30% smaller than conventional vector processors with similar performance.
- Key features: reuse of integer register file, short SIMD unit (8-bit), swizzling, reductions
- Area overhead, 30% when compared with LEON3.

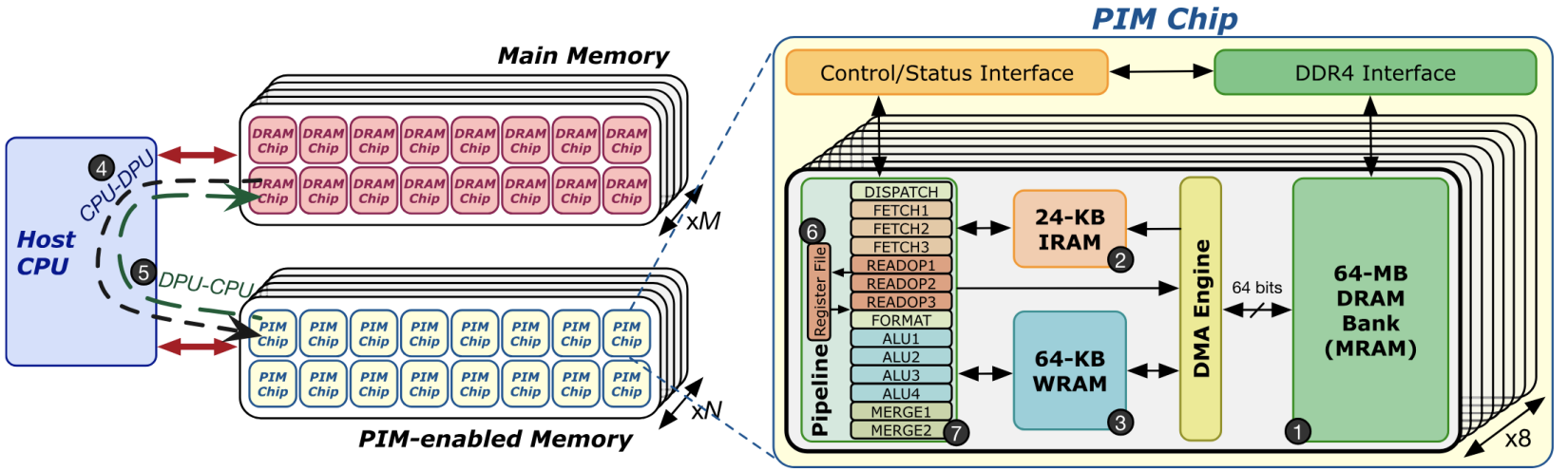


Program	Data size	LEON3	SPARROW	Speed-up
Matrix Mult.	120×120	117,310,797	12,632,882	9.3×
Grayscale	256×256	3,221,620	876,633	3.67×
Filter	256×256	39,019,342	11,855,218	3.3×
Cifar-10	32×32	3,784,906	649,618	5.83×
Polynomial	2048	88,145	19,537	4.5×

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Processing in Memory

- **Computation into or near the memory instead of transferring large amounts of data to and from the computing unit.**
  - **Reduced data movement**: Processing data within the memory chip minimizes the need for data transfers between the CPU and main memory.
  - **Improved bandwidth utilization and lower latency**: Direct connections between computational units and local DRAM banks allow for higher internal bandwidth.
  - **Energy Efficiency**: Reducing off-chip data movement lowers energy consumption.
  - **Scalability**
- **Challenges:**
  - Thermal management
  - Radiation tolerance
  - Programming complexity



AI Accelerators

ESA UNCLASSIFIED – Releasable to the Public

Accelerators for AI inference

AI Accelerator

AI accelerator is a class of specialized hardware accelerator designed to accelerate AI and ML applications, including artificial neural networks and machine vision.

Most times it is a many core design which features:

- Low precision arithmetic
- Specialized dataflow architectures
- In-memory computing capabilities

AI specialized accelerators is expected to be 10x more efficient, most times it is up to 100-1000x than general purpose solutions (GPU).

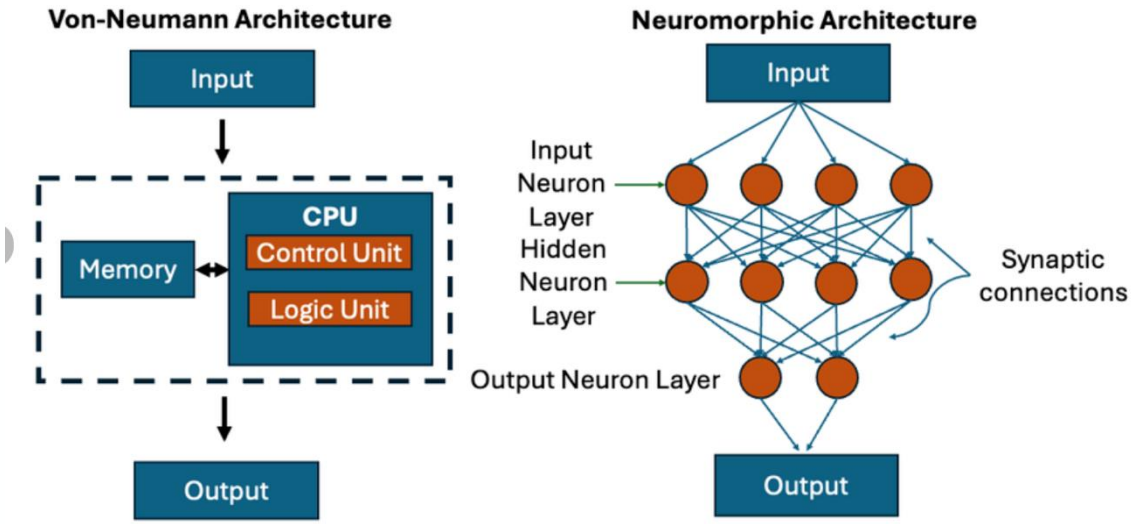
- Lower latency.
- Scalability.
- Associated SW stack.

Neuromorphic Processor

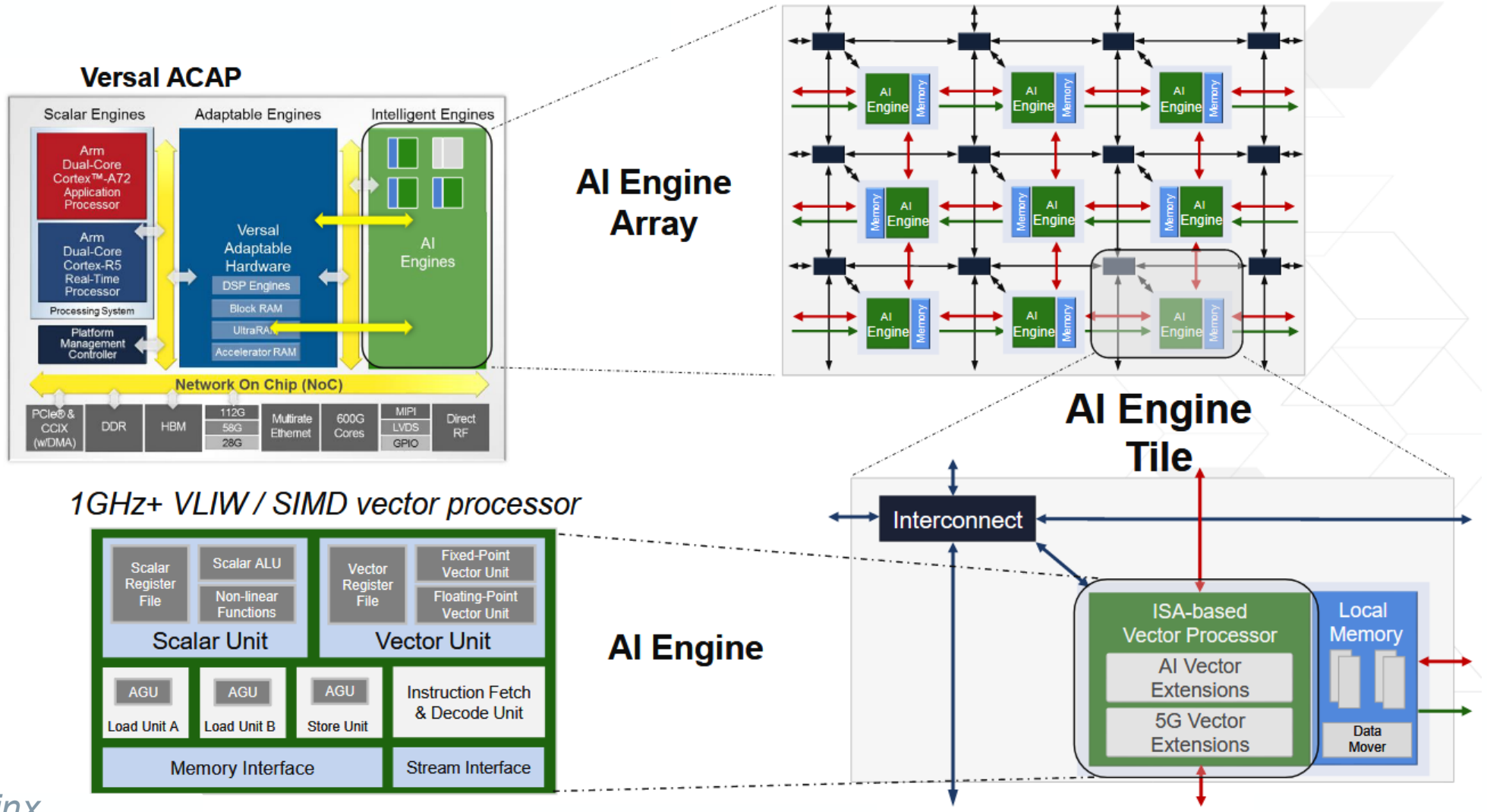
Brain-inspired: mimic how biological neurons and synapses process information.

Use **spiking neural networks (SNNs)** instead of traditional digital logic for computation.

Designed for **event-driven, parallel, and low-power** processing:



Xilinx Versal SoC and AI accelerators

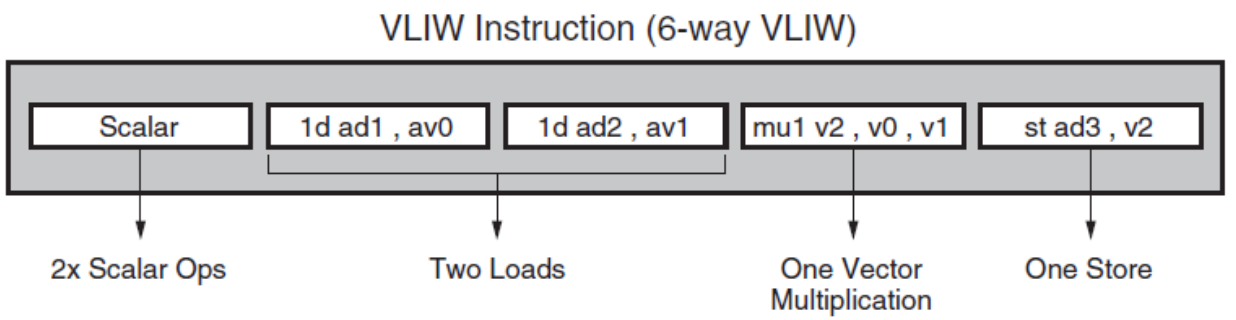


Source: Xilinx

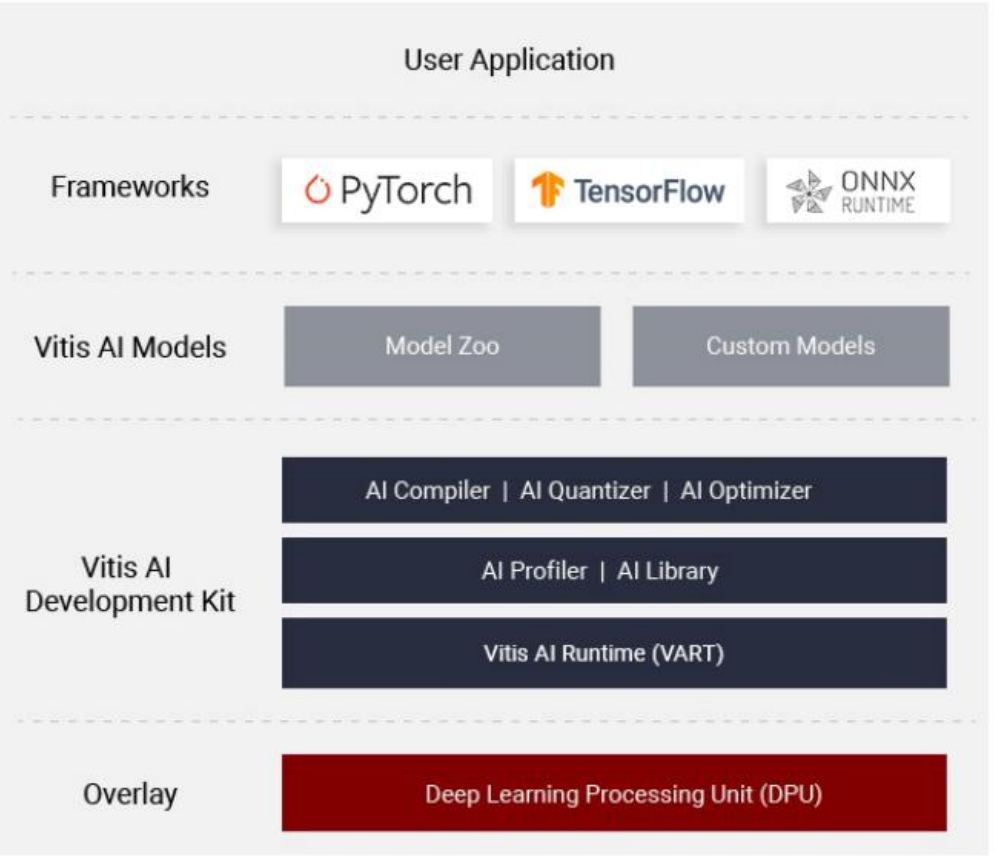
Xilinx Versal SoC and AI Accelerators

Targeting

- Software stack integrated in Vitis.
- Multi-precision math support (8/16/32 SPFP, Real, Complex, INT(4, 8, 16, 32), BFLOAT16) – MAC/cycle depend on selected precision.
- Instruction parallelism: VLIW enabling 7+ operations per clock cycle.
- Data parallelism: SIMD.
- Deterministic performance.
- Example applications: Radio solutions and CNN inference.

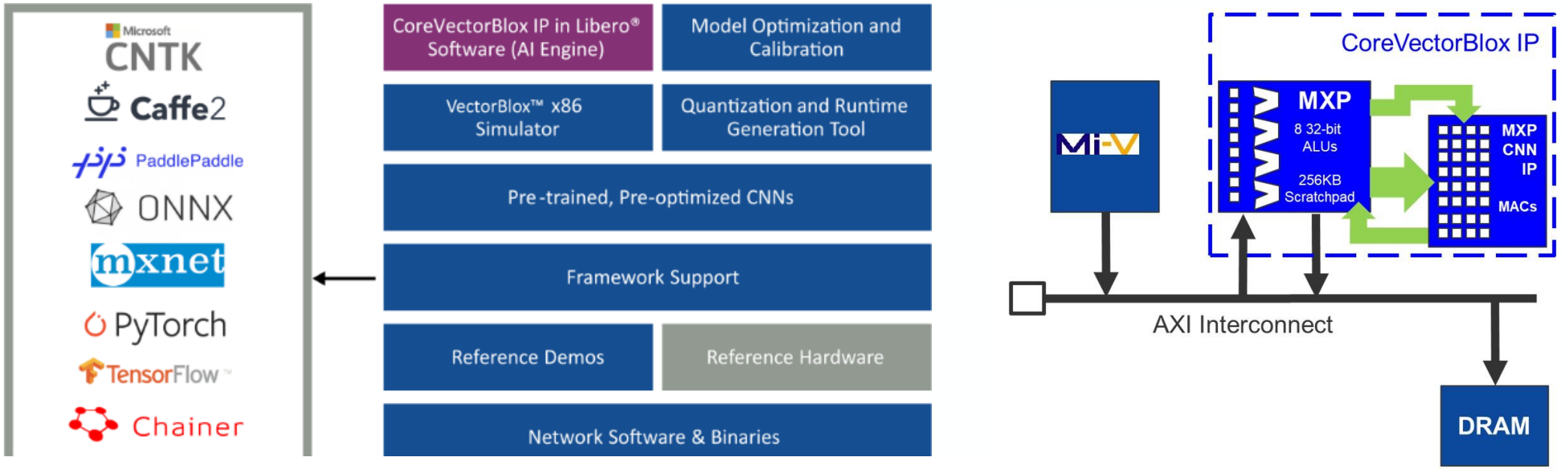


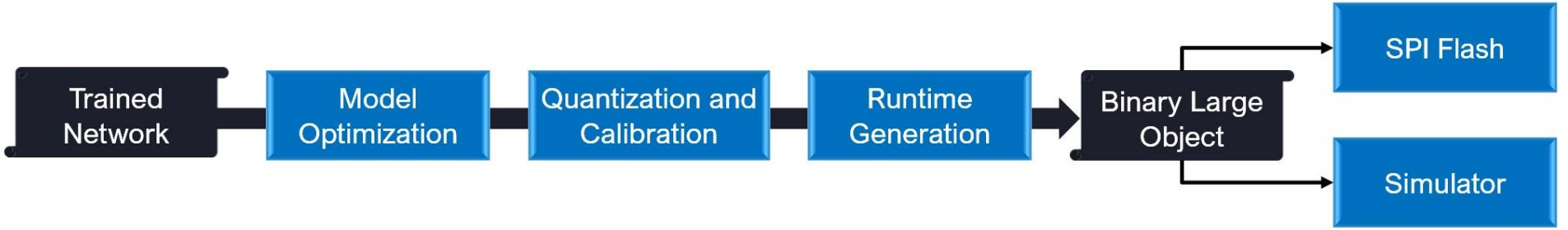
Vitis AI structure



Source: Xilinx

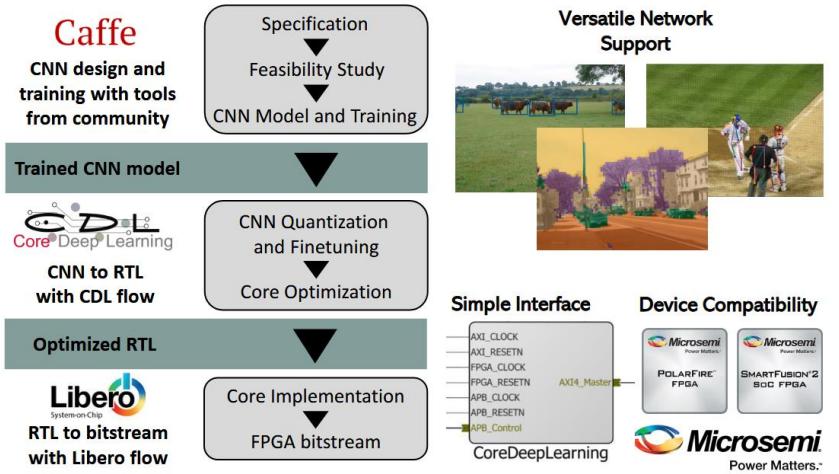
Microchip PolarFire SoC AI Flow



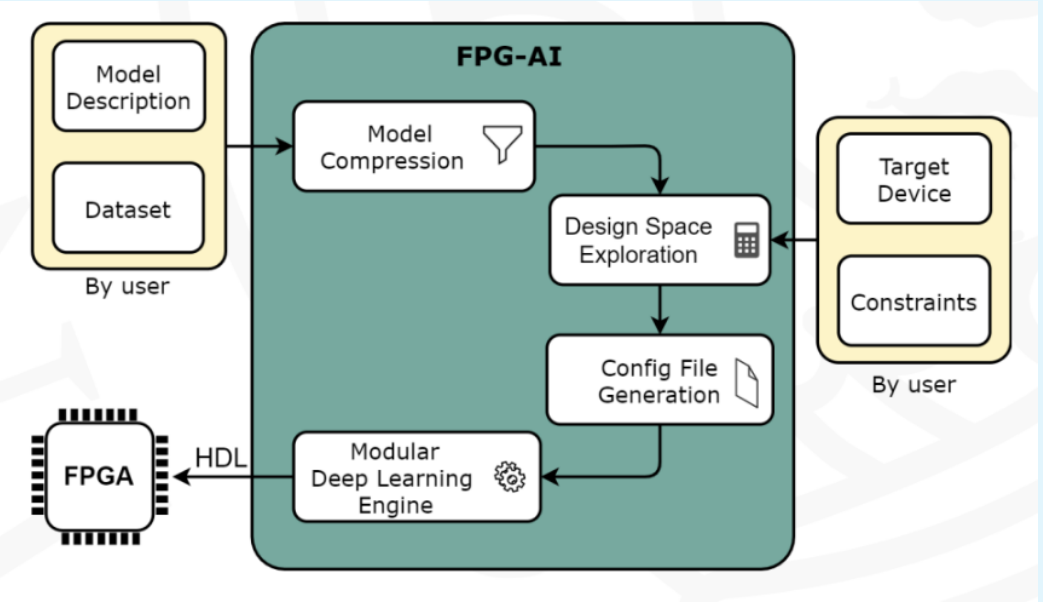


AI Accelerator Soft Cores

- Core Deep Learning
  - Works on 8-bit fixed point.
  - CNN solution for FPGA implementation.
  - Scalable.
- Solutions based on generation of the neural network with HLS, such as Xilinx FINN.

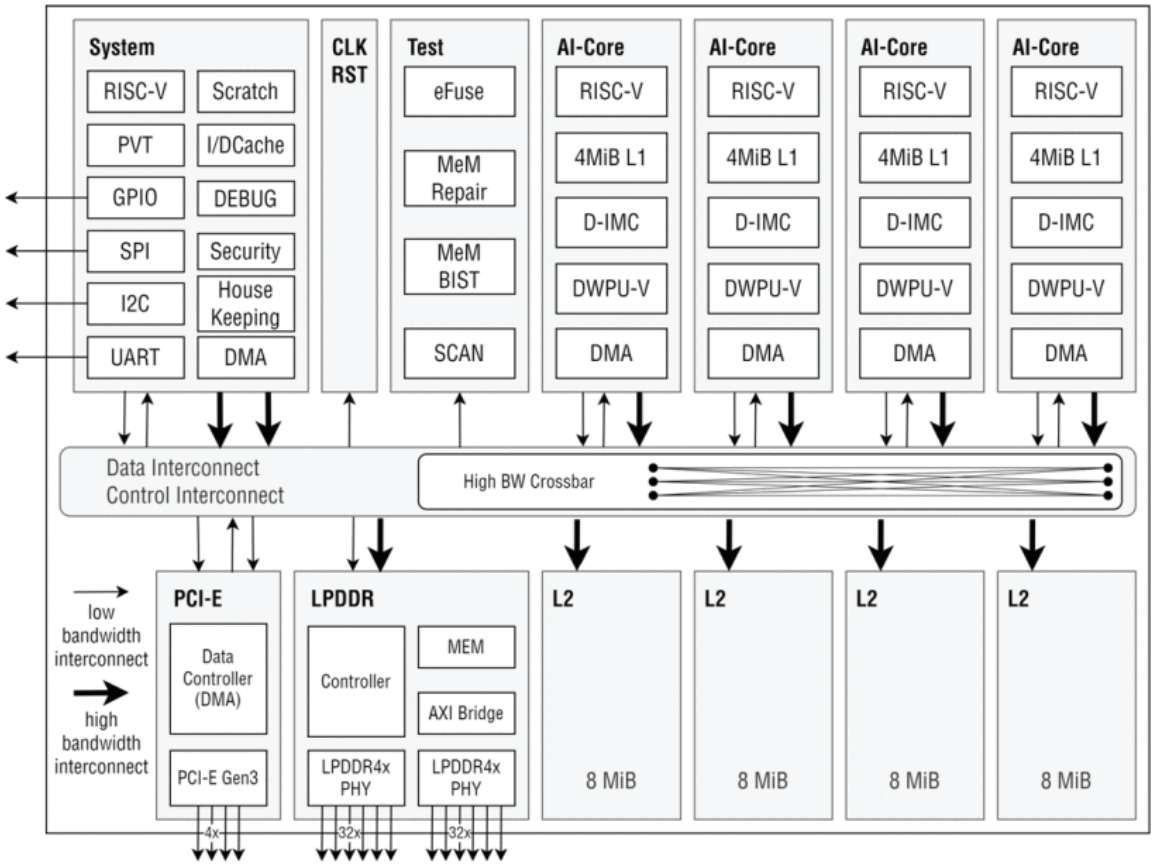


- **FPG-AI (University of Pisa)**
- Technology independent framework for Edge AI deployment onboard satellite.
- Mapping to NanoXplore FPGAs.



AI Accelerators - Axxelera METIS AIPU

- Metis AI Processing Unit (AIPU): Quad-core SoC designed for edge inference.
- **Performance:**
  - 52.4 TOPS per AI core
  - 209.6 TOPS total throughput
- **Architecture:**
  - Uses quantized digital in-memory computing (D-IMC).
  - 8-bit weights 8-bit activations
  - Full-precision accumulation
- **Benefits:**
  - Reduced memory cost for weights and activations
  - Lower energy consumption for matrix-vector multiplications (MVM)
  - Maintains neural network accuracy.

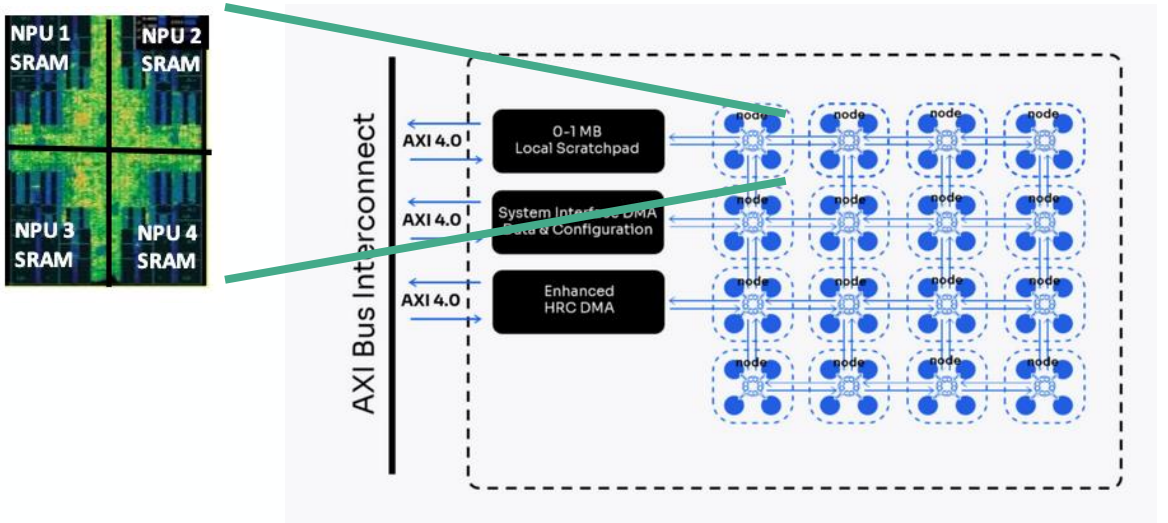


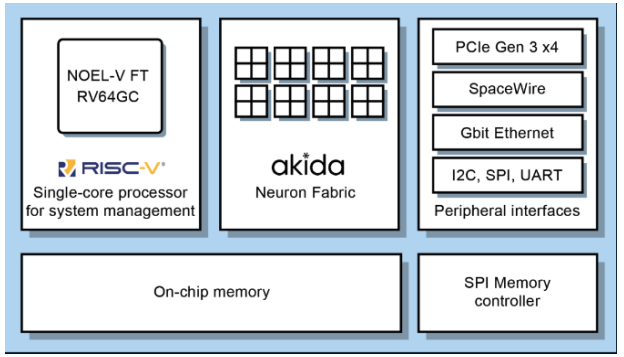
Metis AIPU SoC Architecture

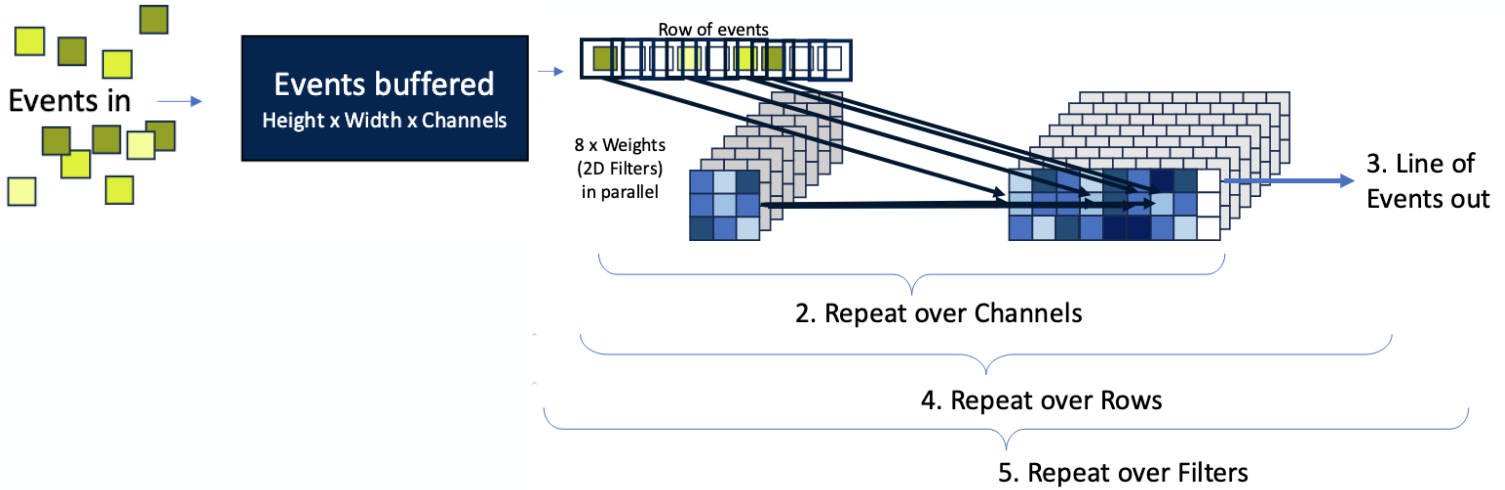
Source: P. A. Hager *et al.*, "11.3 Metis AIPU: A 12nm 15TOPS/W 209.6TOPS SoC for Cost- and Energy-Efficient Inference at the Edge," *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, San Francisco, CA, USA, 2024, pp. 212-214

Neural Processor - Akida Brainchip

- Scalable fabric of 1-128 nodes
- Each neural node supports 128 MACs.
- Configurable 50-130K embedded local SRAM per node.
- DMA for all memory and model operations.
- On-chip communication via mesh network.
- Multi-layer execution without host CPU
- Integrate with any microcontroller or application Processor.
- Efficient algorithmic mesh.







PIC64 High-Performance Spaceflight Computing

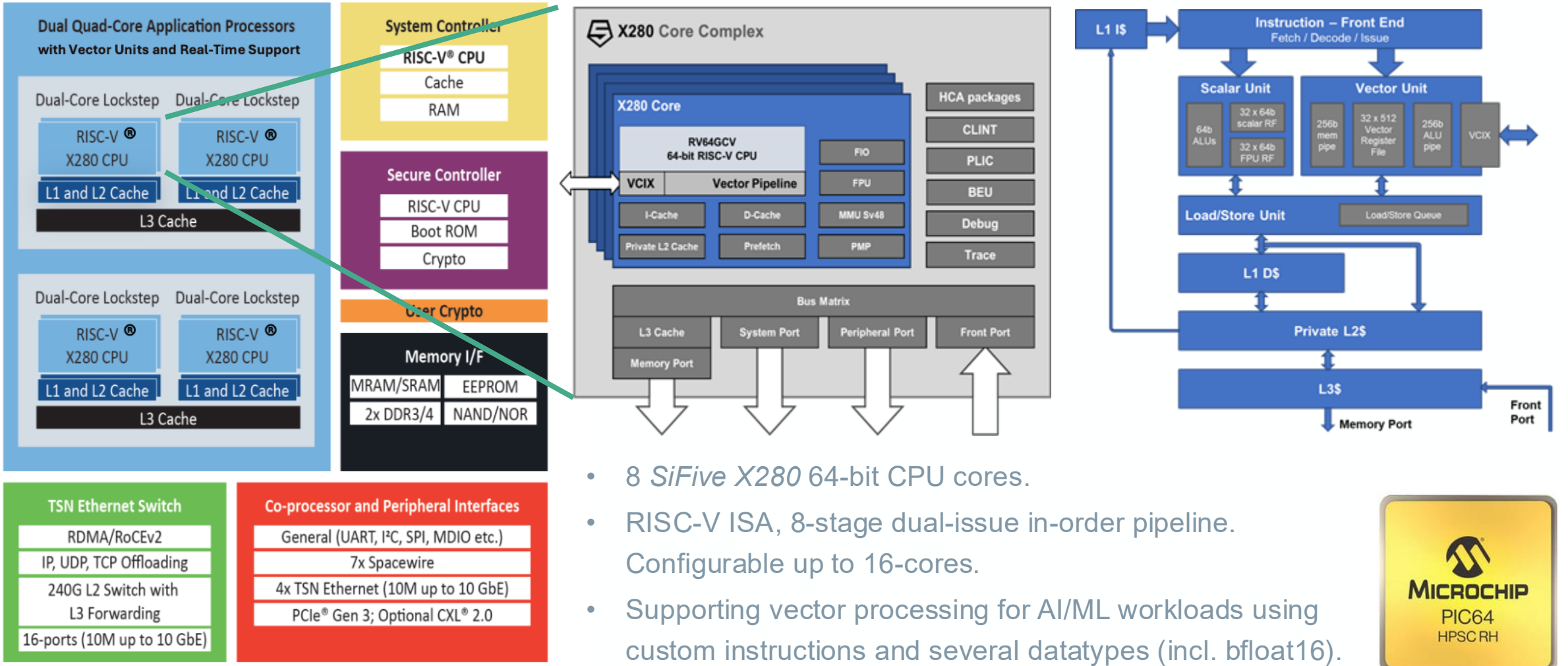


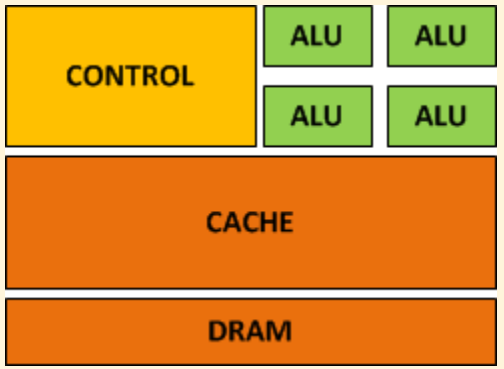
Figure 1 – HPSC Architecture *(courtesy Microchip Technology)*

- 8 *SiFive X280* 64-bit CPU cores.
- RISC-V ISA, 8-stage dual-issue in-order pipeline. Configurable up to 16-cores.
- Supporting vector processing for AI/ML workloads using custom instructions and several datatypes (incl. bfloat16).
- Implements RISC-V Vector ISA version 1.0

Graphics Processing Units (GPUs)

ESA UNCLASSIFIED – Releasable to the Public

GPGPUs – General Purpose GPUs



CPU

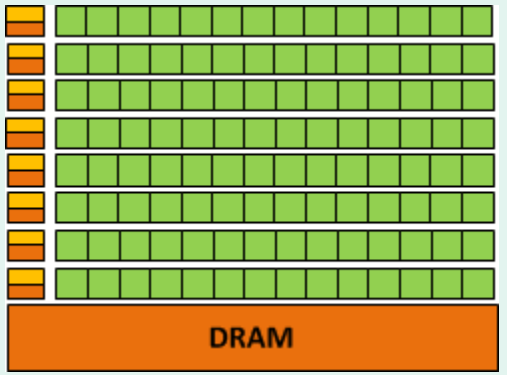
Most of the transistors devoted to control and memory.

Achieve high speedups in applications that are control-flow intensive.

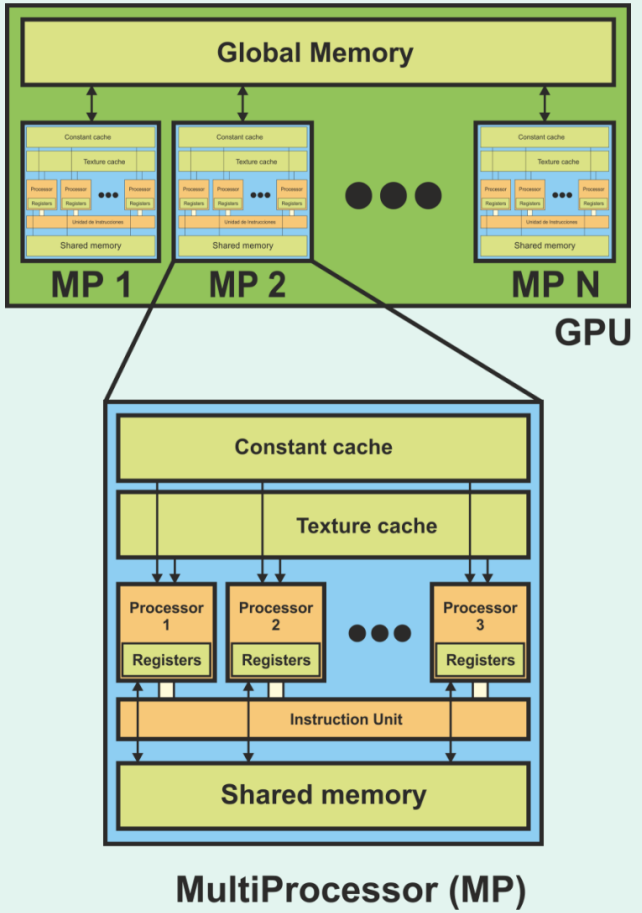
GPU

Most of the transistors devoted on small arrays of execution units, dispatches, small volumes of shared memory and memory controllers:

All this does not accelerate the execution of separate streams, but allows to process several thousands of threads in parallel.



Hierarchical hardware parallelism (architecture)



GPUs consist of a set of multiprocessors, each composed of a set of simple processing elements working in SIMD (single instruction multiple data) mode.

Each multiprocessor has different elements, including local registers, shared memories and the processing cores.

Dramatically increase computation speed in applications where **a huge amount of data can be processed in parallel.**

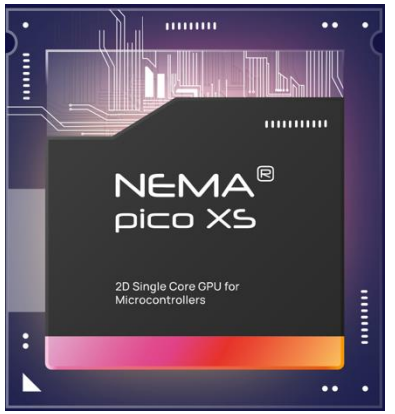
GPGPUs for space use

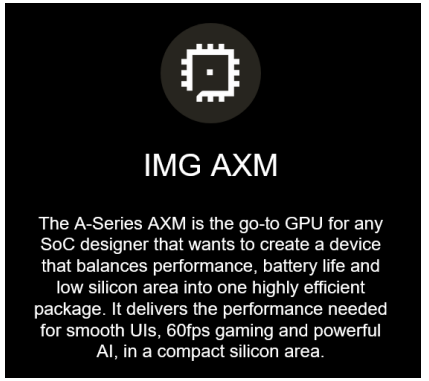
- GPUs in general are a good fit for highly flexibly high-performance on-board processing.
- COTS **low power** GPU SoCs have high-performance and are suited for new space applications where component qualification is not critical.
- Radiation testing & FDIR required.
- GPU IP Cores could be a way forward for increased reliability,but requires significant investment.
- OBPMark benchmarks available.

- **NVIDIA Jetson Orin NX**
- **AI Performance:** Up to **100 TOPS (INT8, sparse)** for the 16 GB model
- **Compute:** 8-core **ARM Cortex-A78AE** CPU + **1024-core Ampere GPU** with 32 Tensor Cores
- **Memory:** 8 GB or 16 GB **LPDDR5**, ~102.4 GB/s bandwidth.
- **Power:** Configurable **10–25 W**, up to ~40 W in Super Mode
- **Benchmarks:** Runs **YOLOv8n at ~52 FPS (FP16)**, or **~65 FPS (INT8)** on Orin NX 16 GB
- **Use-cases:** Edge AI, robotics, drones, and autonomous machines.
- Use in space with shielding.

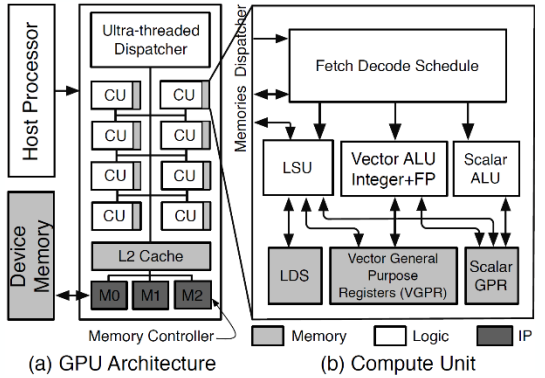
GPGPU – GPU-like IP Cores

- IngeniARS GPU@sat
- Imagination Technologies
  - IMG AXM series
- ThinkSilicion NEMA GPU
- Open source-research oriented:
  - MIAOW → AMD architecture synthesizable on FPGA, OpenCL
  - FlexGrip → NVIDIA architecture, CUDA
  - Limitations → memory interface, instruction support.
- Generally, IPs are very expensive, closed designs.



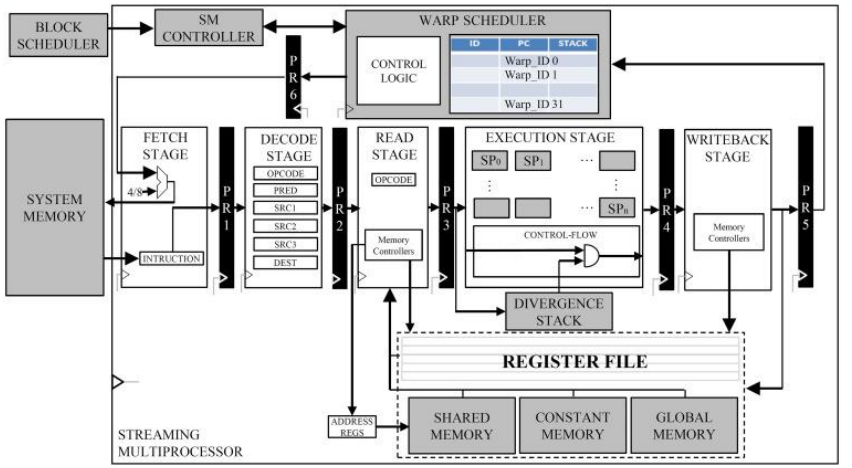


MIAOW Architecture



GPU@SAT
HIGH PERFORMANCE • HW-SW CONFIGURABLE • AI ENABLER

FlexGripPlus SM



GPGPU – GPU-like IP Cores

Vortex GPU

- **Open-source RISC-V GPGPU**: Designed to support general-purpose GPU workloads with RISC-V ISA extensions.
- **SIMT architecture**: Supports parallel threads/warps, multiple cores, units like ALU, FPU, LSU, SFU, etc.
- **Software support**: Compatible with OpenCL 1.2 for running kernels.
- Configurable pipeline: you can tune number of cores, threads, warps, cache hierarchy (L1, L2, L3) and shared memory.
- Hardware-software integration: extensions to RISC-V ISA to support OpenCL kernels.
- Supports both 32-bit and 64-bit RISC-V variants (RV32IMAF and RV64IMAFD) depending on configuration.
- Layout in 15 nm educational cell.
- Good scaling with #cores when using Rodinia benchmark.
- Arria 10 FPGA -> 192 MHz

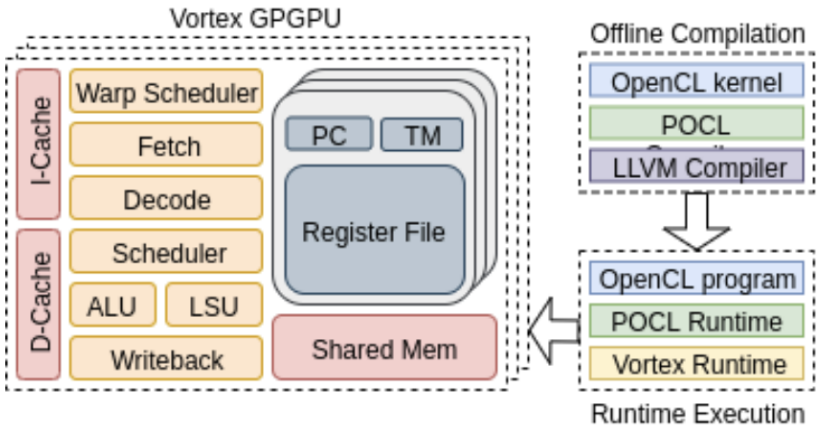


Fig. 1: Vortex System Overview

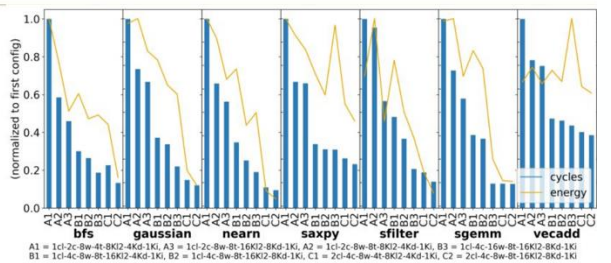
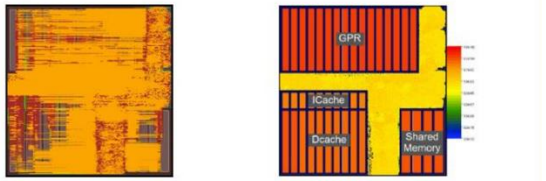


Fig. 4: Vortex v0.1 performance for Rodinia benchmark with normalized cycle and energy utilization on Arria 10 FPGA



(a) GDS Layout (b) Power density distribution

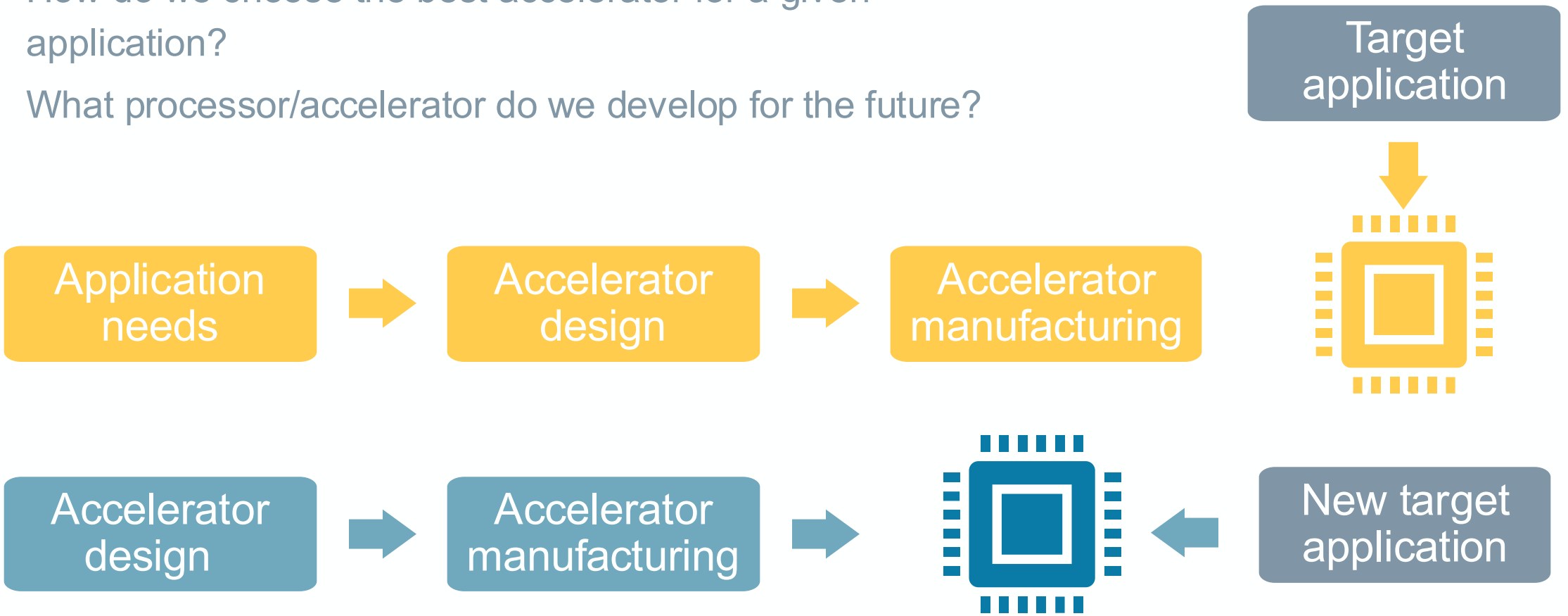
Fig. 3: GDS layouts for a single-core 8-warp 4-thread configuration synthesized @300Mhz produced 46.8mW total power

Design Challenges

ESA UNCLASSIFIED – Releasable to the Public

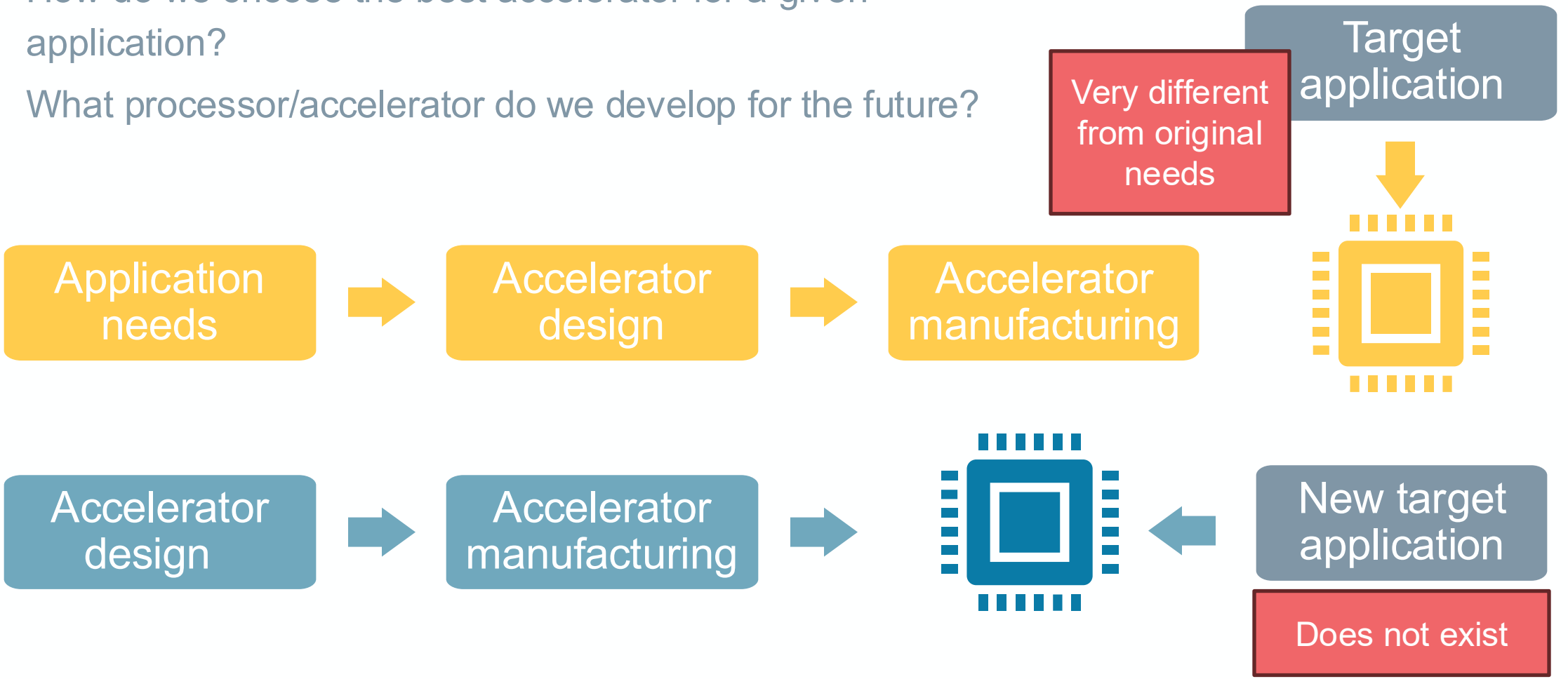
Application or Technology Driven?

- How do we choose the best accelerator for a given application?
- What processor/accelerator do we develop for the future?



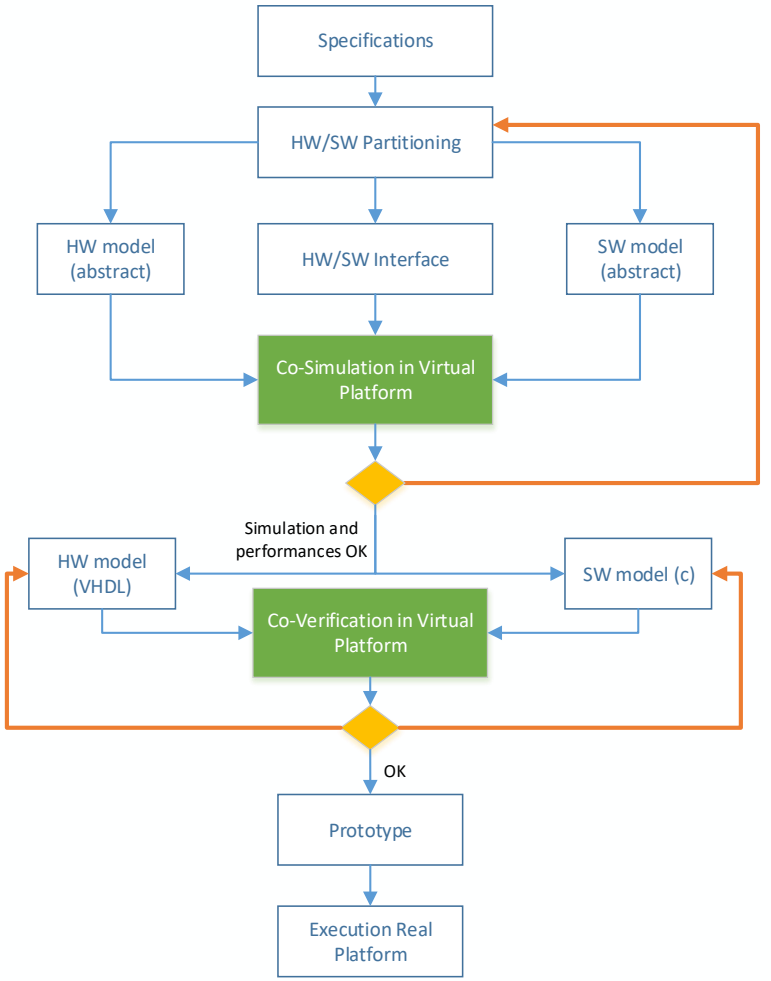
Application or Technology Driven?

- How do we choose the best accelerator for a given application?
- What processor/accelerator do we develop for the future?



Profiling, Co-design, Design Space Exploration

- **Profiling**: parallelisation capabilities, potential bottlenecks, type of operations, required accuracy…
- **Hardware/Software (HW/SW) co-design:** simultaneous consideration of hardware and software within the design process. It involves co-simulation in a Virtual Platform that models the hardware.
    - HW and SW are simultaneously developed
    - Fast simulation speed and high observability.
    - HW/SW partitioning.
    - Reduces the risk of choosing the right architecture to then realise there is no software ecosystem, software stack needs to be developed from scratch -> expensive!
- **Design Space Exploration:** find the right architecture: memory size, memory hierarchy, external (Host-Accelerator) Interface bandwidths, number of processing cores/clusters number of accelerator functional units, any other dedicated functional unit/IP.



Desing Space Exploration

- Functionality implementation on SoCs is becoming more sophisticated with heterogeneous complex computing units: CPU, GPU, VPU, e-FPGAs, systolic arrays, parallel vector engines, neuromorphic engines.
- Need for Cycle-exact microarchitectural simulation platform to simulate heterogeneous complex SoC.
- Full SoC can be mapped to one or more FPGAs, enables validation,

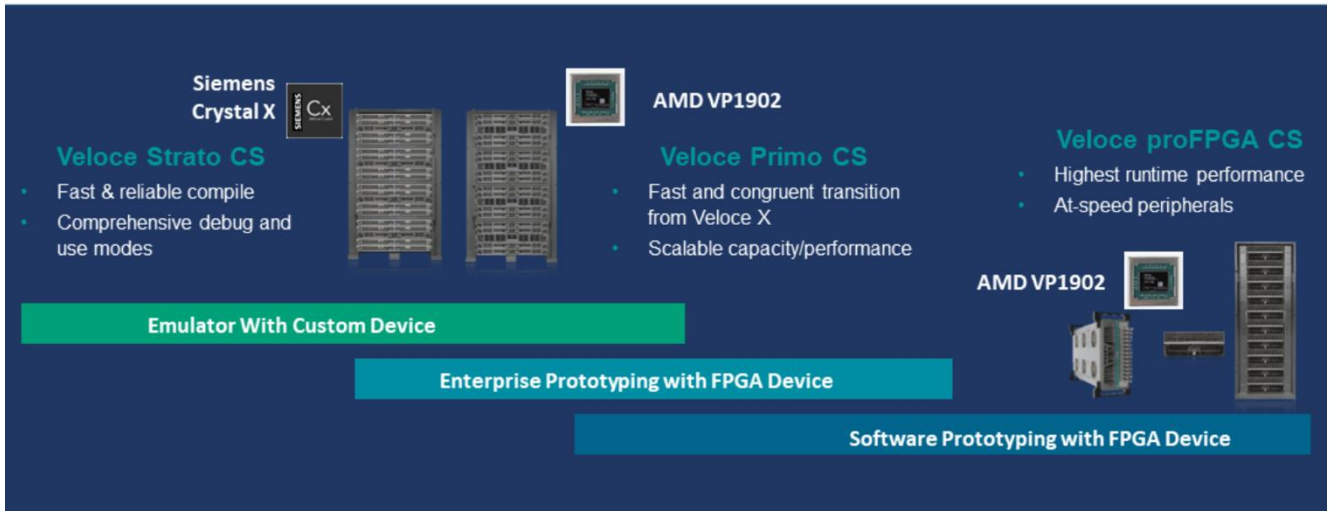
FireSim

Open-source simulation platform for cycle-accurate full-system benchmarking.

FireSim

SpaceSim esa

Mentor Veloce, ProFPGA



Application Examples

ESA UNCLASSIFIED – Releasable to the Public

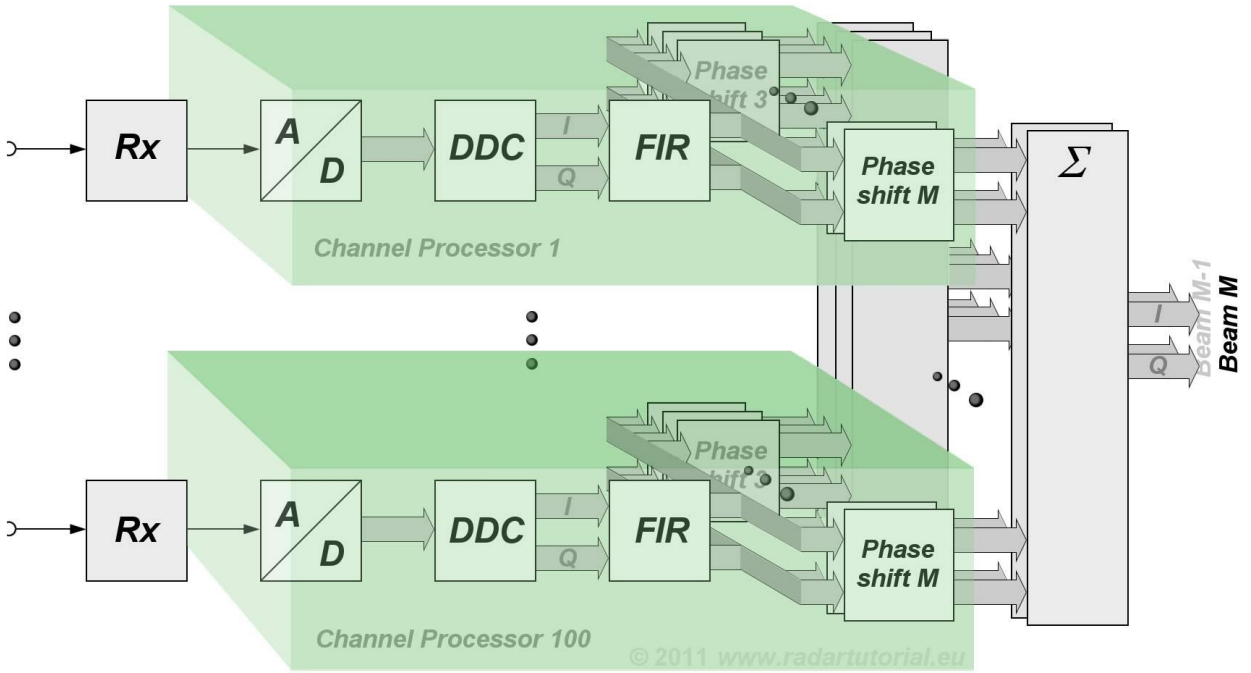
Future processing needs

- Application drivers: increasing demand for on-board processing.
  - Machine learnging powered FDIR, autonomy in exploration (landers, rovers), GNSS.
  - On-board data processing (Earth Observation), high data volumes.
  - Software Defined Radio (Beamforming, DVB-S, 5G-NTN).
  - Cryptography and quantum key distribution.
- Algorithms:
  - Machine Learning is increasingly important.
  - Filters, correlations, CNN with extensive use of parallel/vector oprations.
  - Memory footprint and organisation.
  - Non-linear functions.
  - Data type support for quantized computations (INT16, 8, 4), BFLOAT16.
- **Common denominator: vector and matrix operations**

The Copernicus Hyperspectral Imaging Mission for the Environment, CHIME, will carry a unique visible to shortwave infrared spectrometer to provide routine hyperspectral observations to support new and enhanced services for sustainable agricultural and biodiversity management, as well as soil property characterisation

Applications Needs

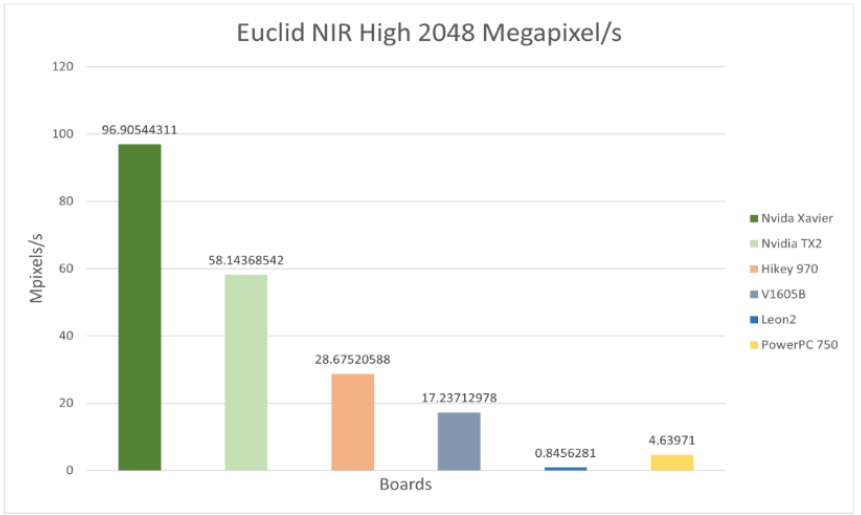
- **Signal processing example:**
  - **Digital beamforming on Rx and Tx**
  - **RFI detection and mitigations**
- Processing increasingly larger bandwidths of signals at increasingly fast rates
  - Fast filtering, FFTs, correlations, convolutions, resampling …
- High precision operations SPFP32.
- Potential HW targets:
  - FPGA + Accelerator
- Application domain:
  - Rx → EO and Telecom
  - Tx → NAV and Telecom

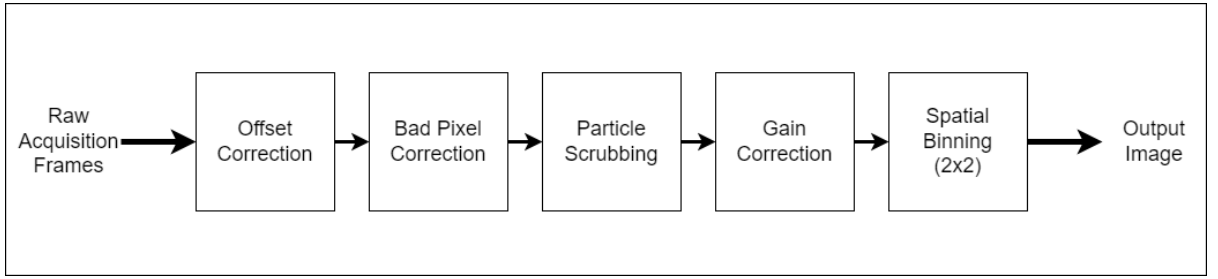


Applications Example

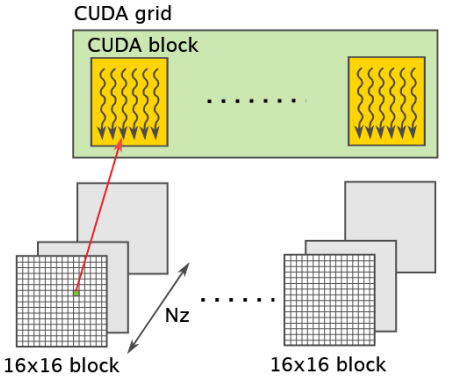
Image processing:

- Image processing generally involves processing of vectors of matrices in floating point precision.
- Amenable to GPU, but control or memory bound applications can limit the performance → low performance/watt ratio
- Many times FPGAs are selected due to the flexibility to optimize parts of the algorithms.
- Challenge: increase in sensor resolution.

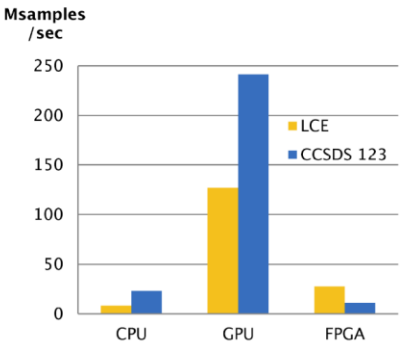




- **Example: Hyperspectral data compression:**
  - GPU-like: Block based, lots of operations in parallel.
  - Prediction-based → data dependencies.
  - Transform-based (DWT) → memory operations.
  - Bit-wise operations involved.



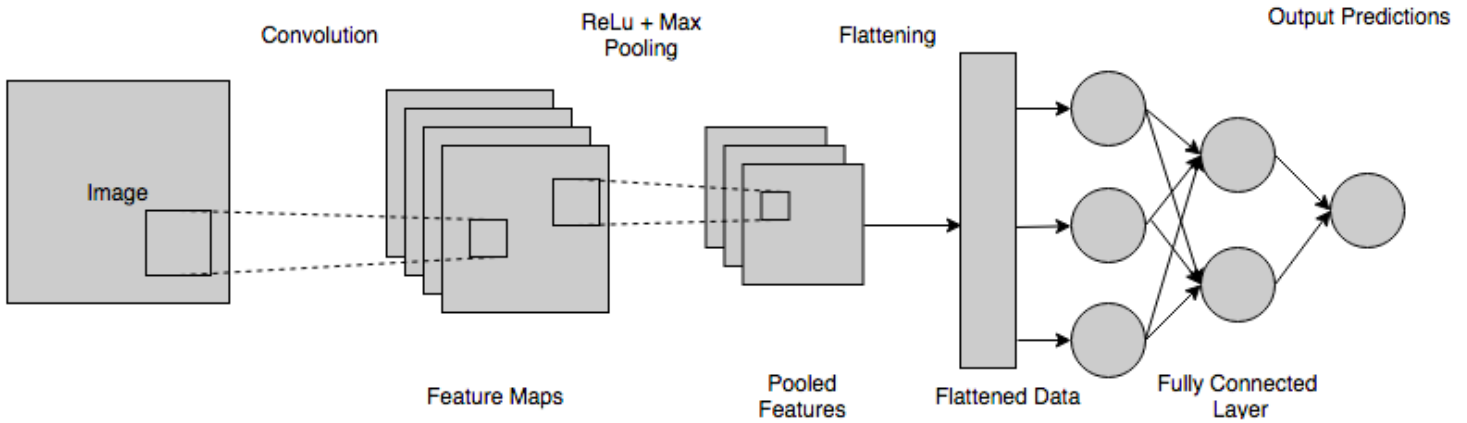
Mapping compression to GPU



Hyperspectral compression in different targets

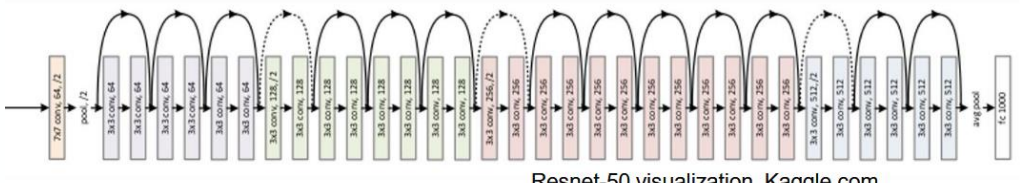
Applications Example

- **AI/ML example:**
  - **CNN for classification**
- Involving mainly low resolution floating-point MAC operations and memory operations.
- Target HW platform: dedicated accelerator, GPU, many core with instruction extensions, FPGA

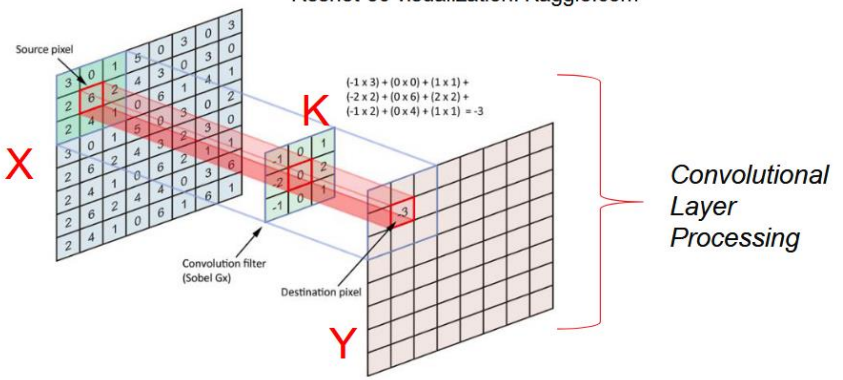


Deep Learning Inference (Conv. Nets)

Application: Modulation Recognition, Waveform Classification



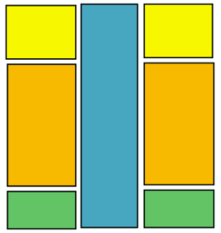
Resnet-50 visualization. Kaggle.com



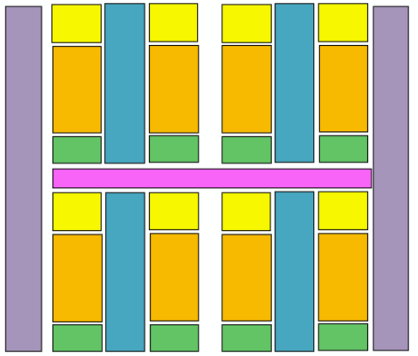
References: "Applied Deep Learning - Part 4: Convolutional Neural Networks", Towards Data Science (blog).

Why Accelerators?

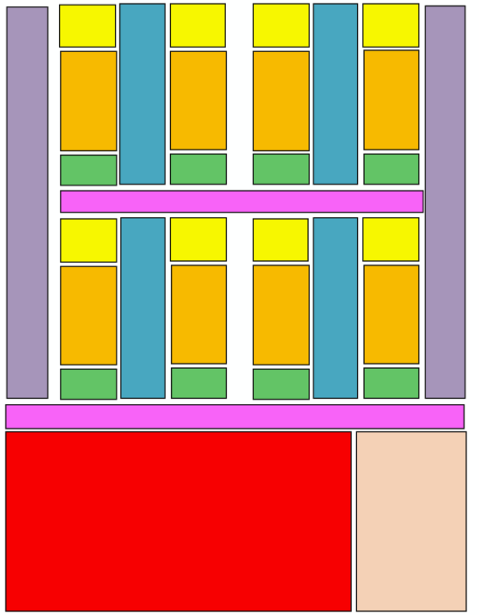
Single core



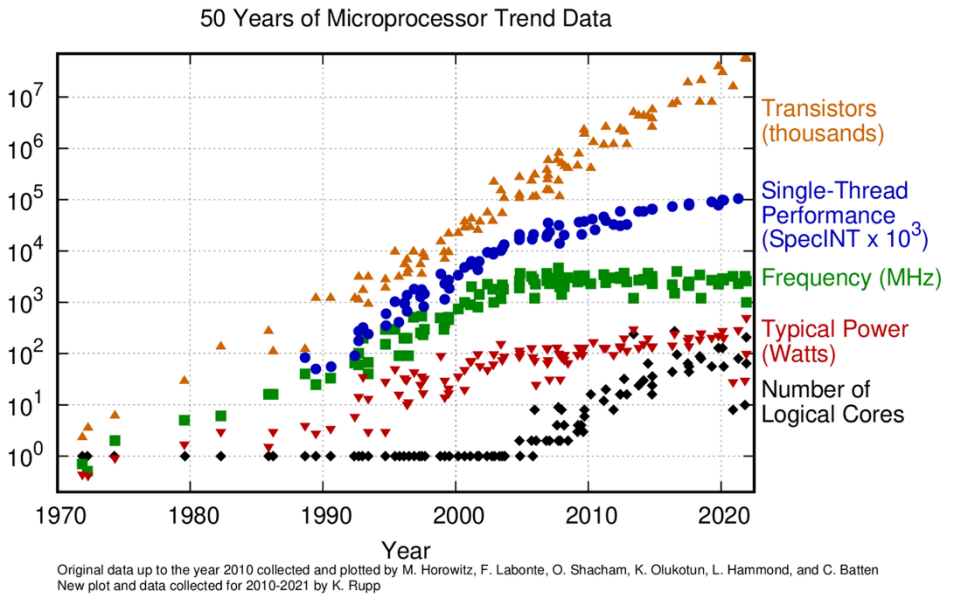
Multi core

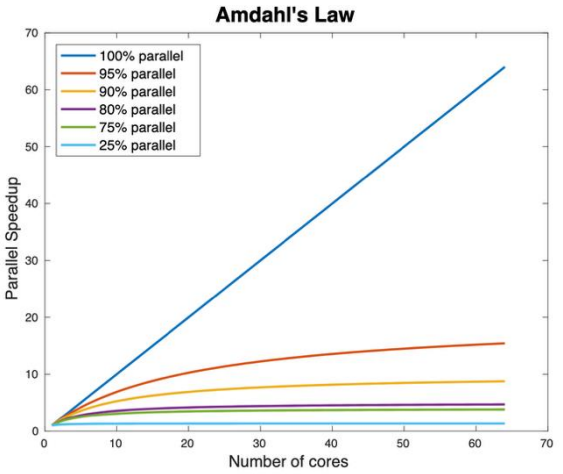


Heterogeneous



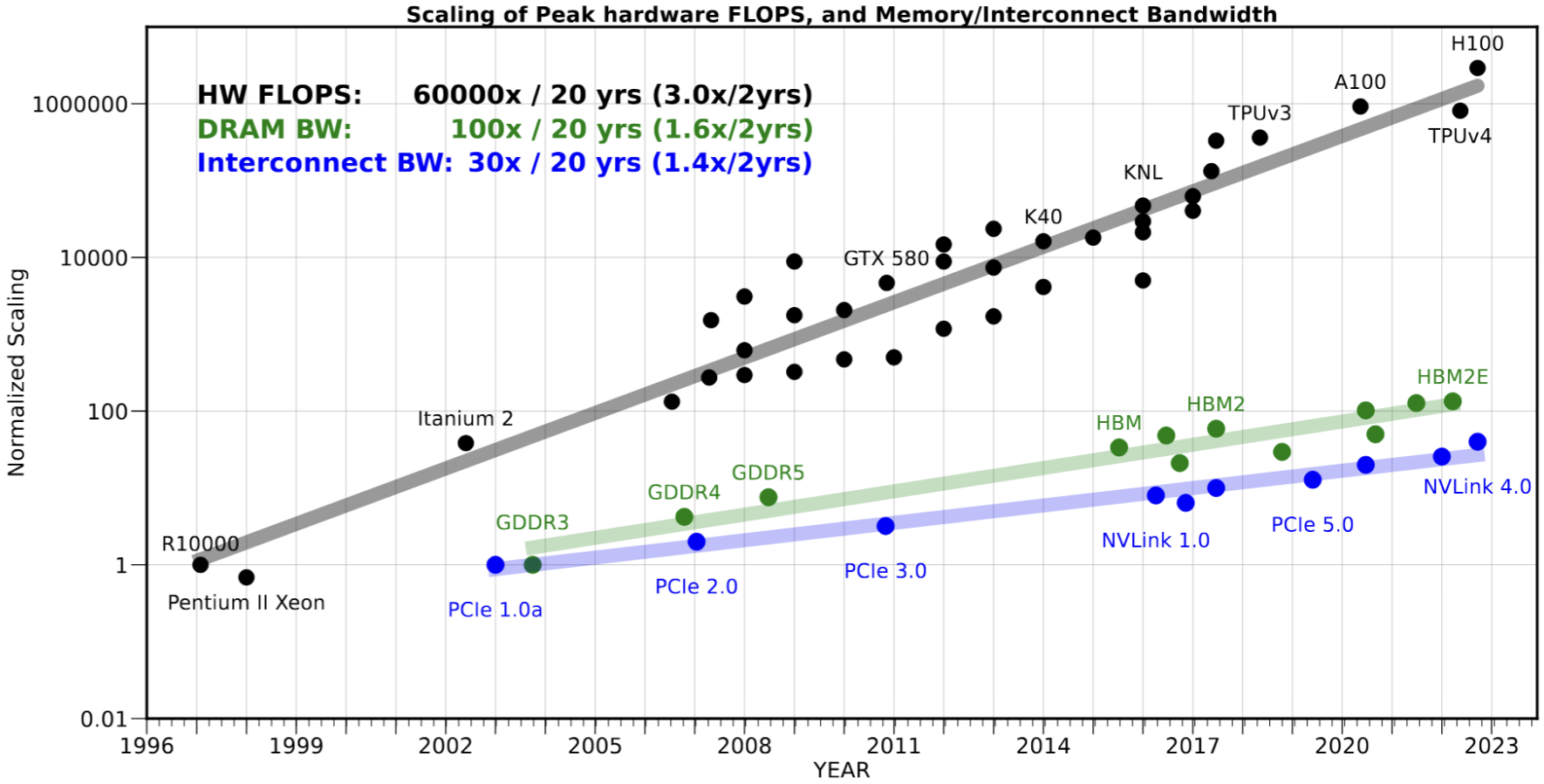
Next (?)





Local memory | Shared memory | Programmable logic | Accelerator | Processor | ALU | Interconnect | External I/F

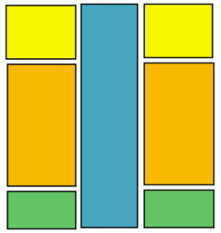
Bandwdith Interconnect and FLOPS increase



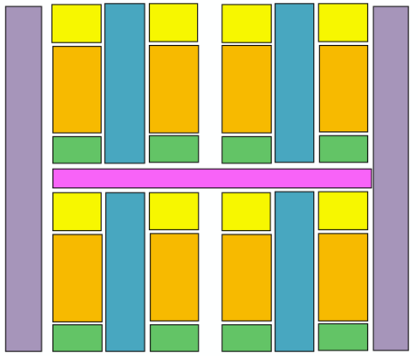
A. Gholami, Z. Yao, S. Kim, C. Hooper, M. W. Mahoney and K. Keutzer, "AI and Memory Wall," in *IEEE Micro*, vol. 44, no. 3, pp. 33-39, May-June 2024

Why accelerators?

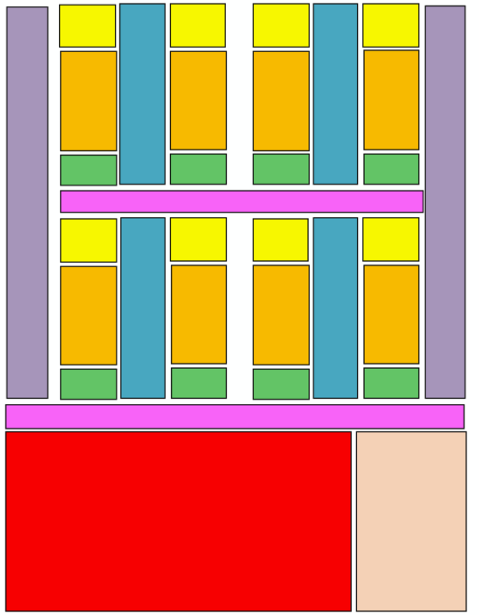
Single core



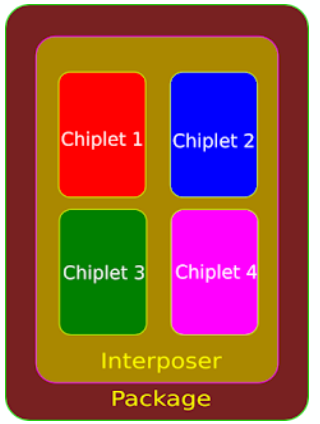
Multi core

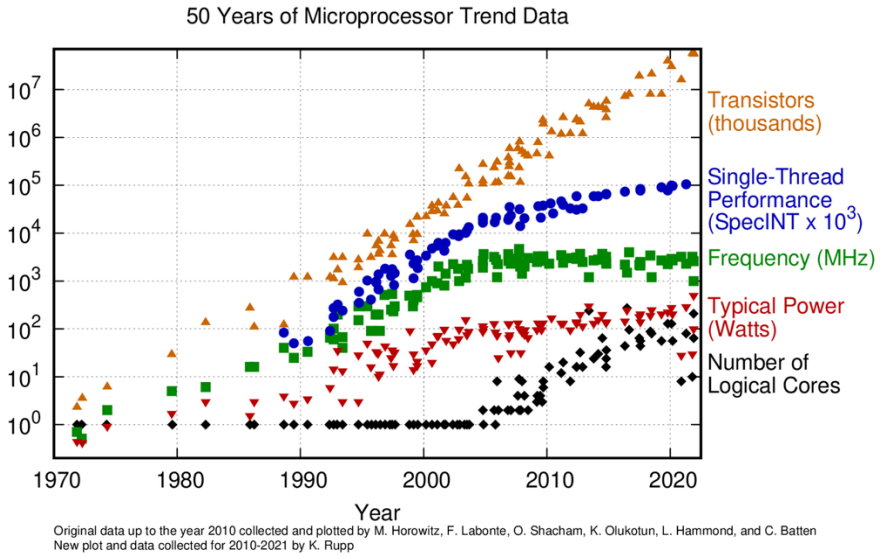


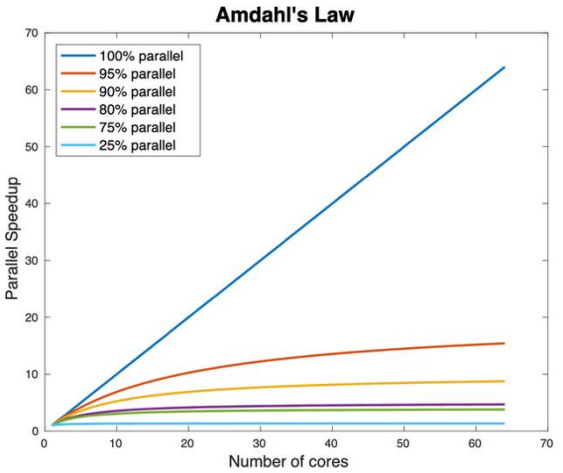
Heterogeneous



Chiplet







Local memory | Shared memory | Programmable logic | Accelerator | Processor | ALU | Interconnect | External I/F

Takeways

- Hardware accelerators constitute a fast evolving landscape.
- Use of COTS is many times limited: radiation tolerance and power comsumption.
- Efficiency of the solution will depend on many factors:
    - Accelerator design and adequacy to intended application.
    - Internal and external connectivity bandwidth.
    - Software stack.
- One single architecture will not fit all applications.
- Design methodology can help: hw-sw co-design, fast prototyping or emulation.
- Continuous development:
    - New IP cores focused on acceleration for space: ISA extensions, AI-inference, neuromorphic, PIM, GPGPU-like
    - IP building blocks: high-speed ADCs and DACs, SERDES, NoC, AXI interconnect.