DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

esa

UNIVERSITÀ DI PISA

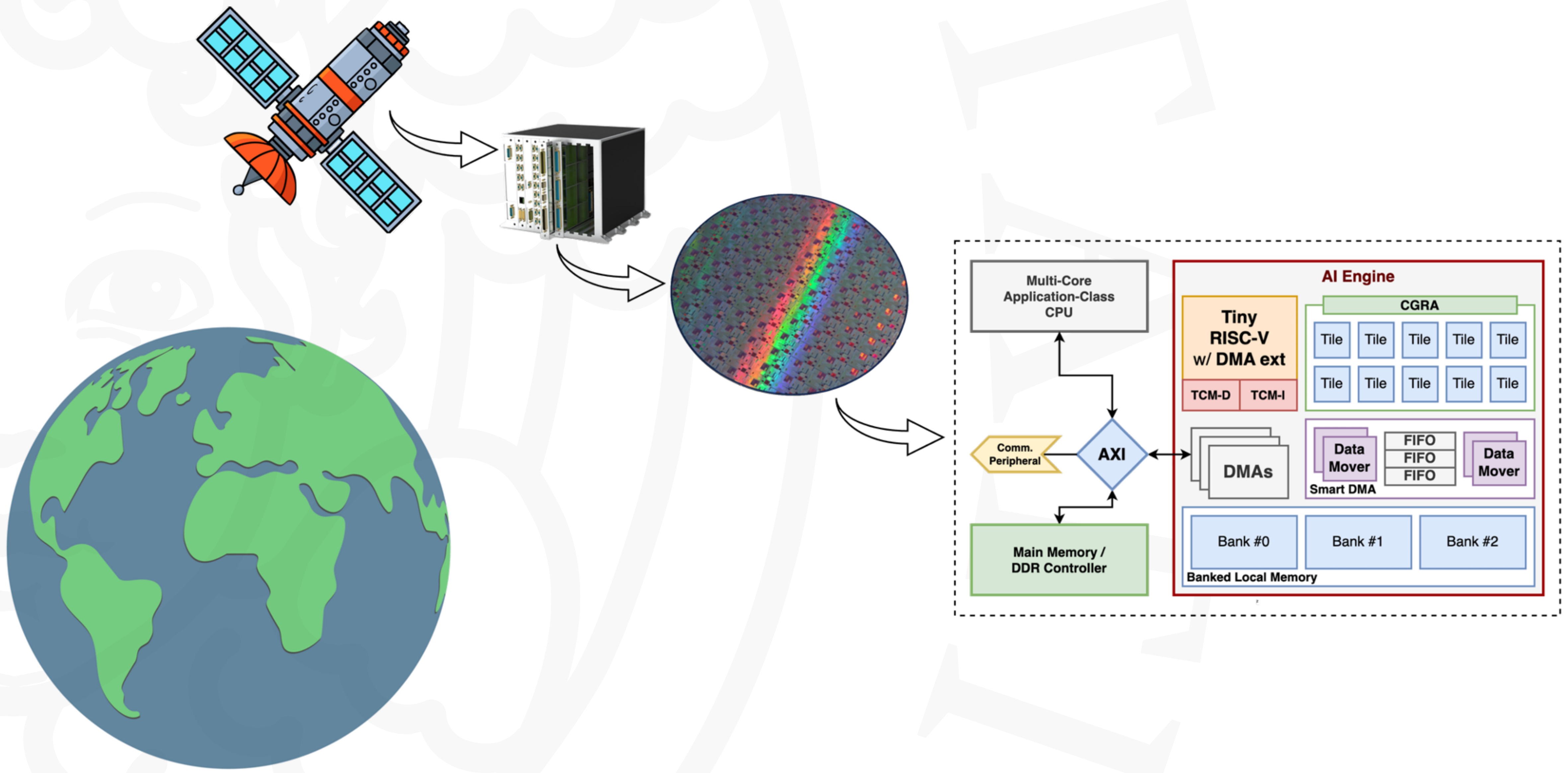
FPGA Prototype of CGR-AI Engine for Space Systems: Step Towards UDSM Implementation

G. Mystkowska, M. Monopoli, D. Merodio Codinachs, P. Nannipieri, L. Fanucci

This contribution is sponsored by ESA Education

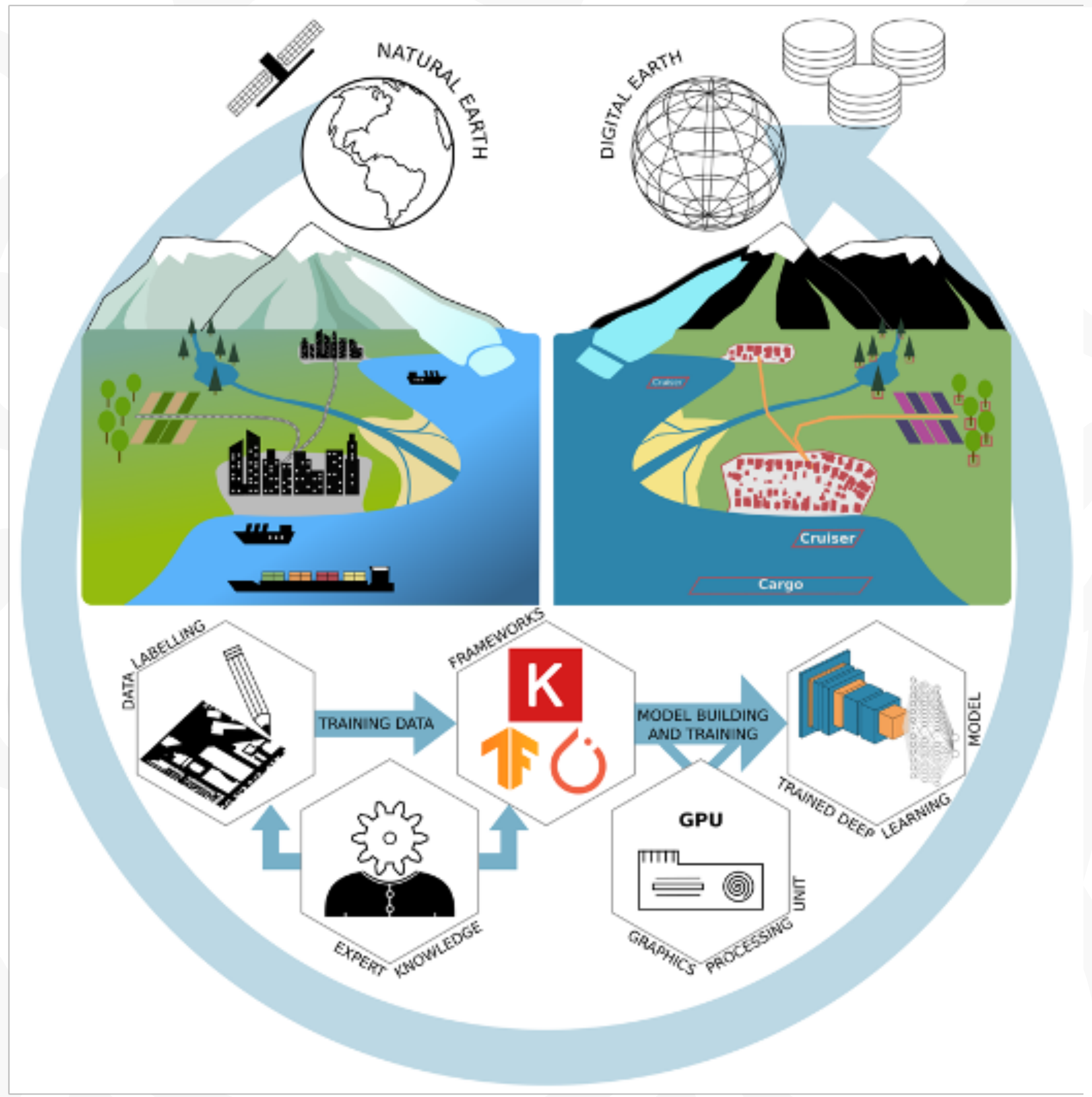
Outline

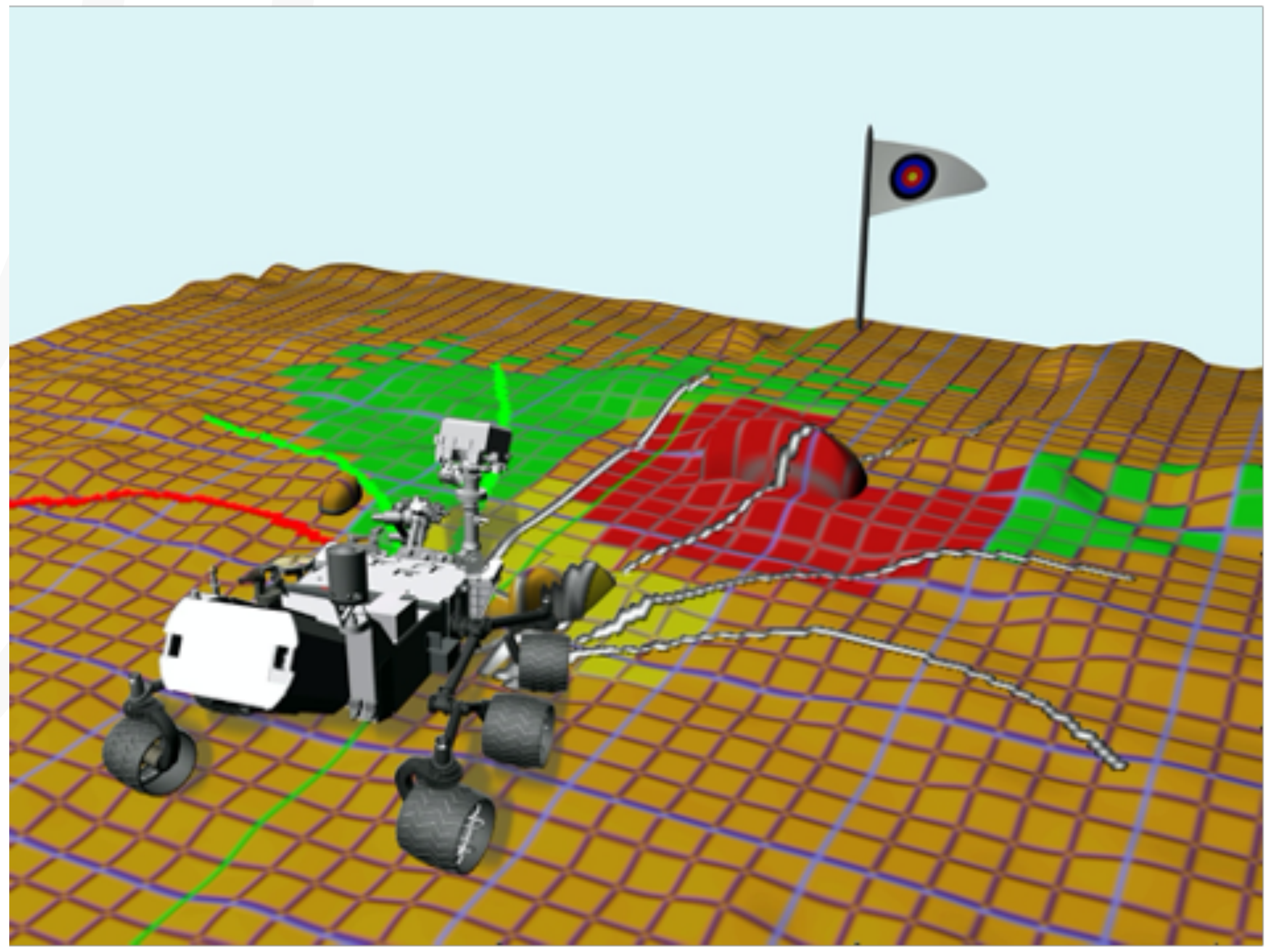
1. AI Applications in orbit
2. Hardware for AI On-board
3. State-of-the-Art
4. HW Accelerator - resource optimised CGRA
5. UniPi's CGRA for AI
    a. CGR-AI Functional Walkthrough
    b. CGR-AI and its Parameterisation
6. FPGA overlay
    a. Strategy
    b. Implementation
    c. Proof of Concept
7. Lessons learned for improved user experience



AI Applications in orbit

1. Remote Sensing
    a. Object detection
    b. Weather forecast
    c. Earth observation
2. Autonomous spacecrafts
    a. Vehicle docking
    b. Probes
    c. Landers, rovers
    d. Deep space missions
    e. Fault detection and isolation, recovery
3. Data privacy increase
    a. Downlink requirements reduction





Hardware for AI On-board

- High Computational Efficiency
  - **Inference**
  - **Real-time processing** in resource-constrained environments
  - Tailored for parallel processing, matrix multiplication and neural network operations
- **Energy efficient**
- Compact and Integrated Design
  - **Small** form factors
- Customizability and Flexibility
  - **Reconfigurable** to meet specific AI requirements
  - **Adaptable** to evolving AI models and algorithms without requiring new hardware
- Reliability
  - Must be able to **withstand exposure to radiation**
  - **Protection** against **Single Event Effects (SEEs)** through radiation-hardening or fault mitigation

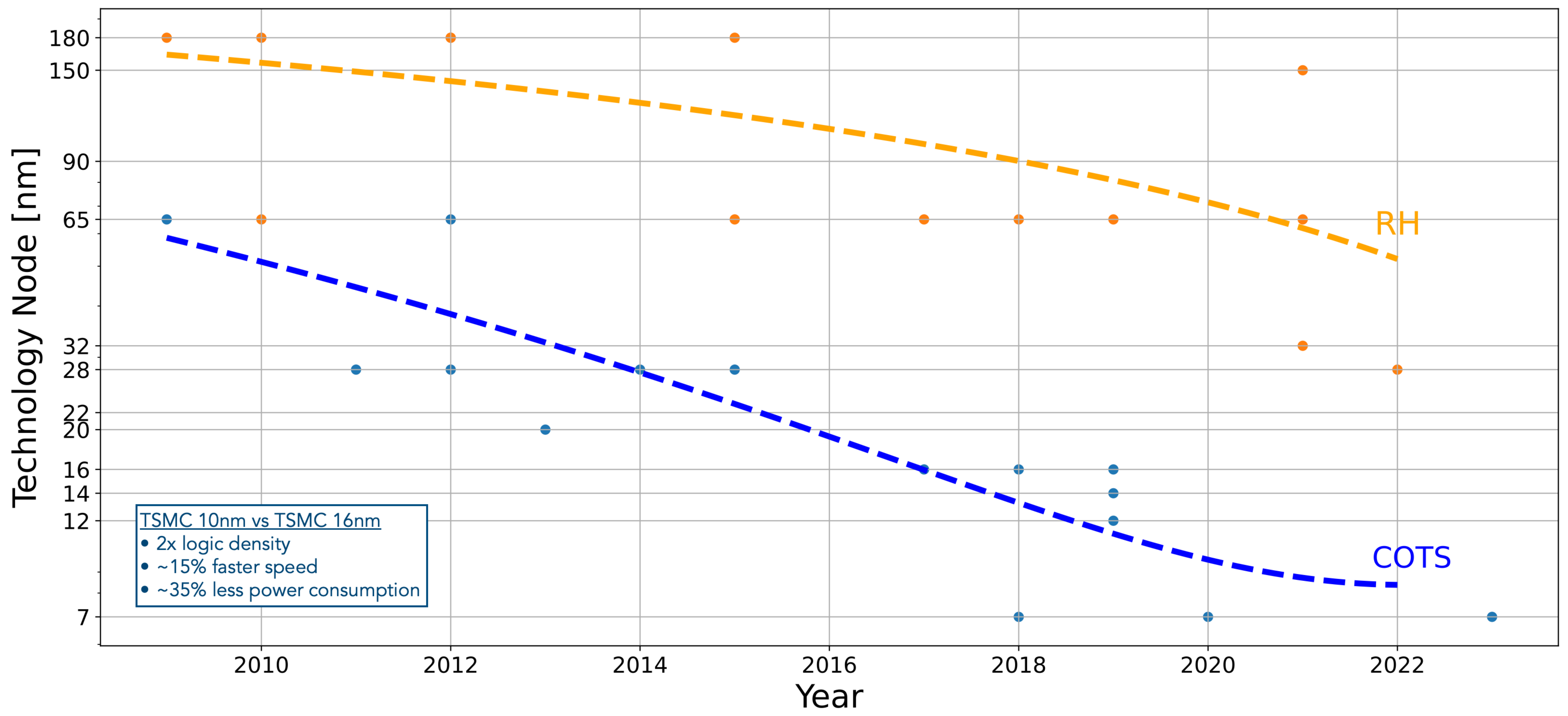
State-of-the-Art

NON RAD-HARD

RAD-HARD

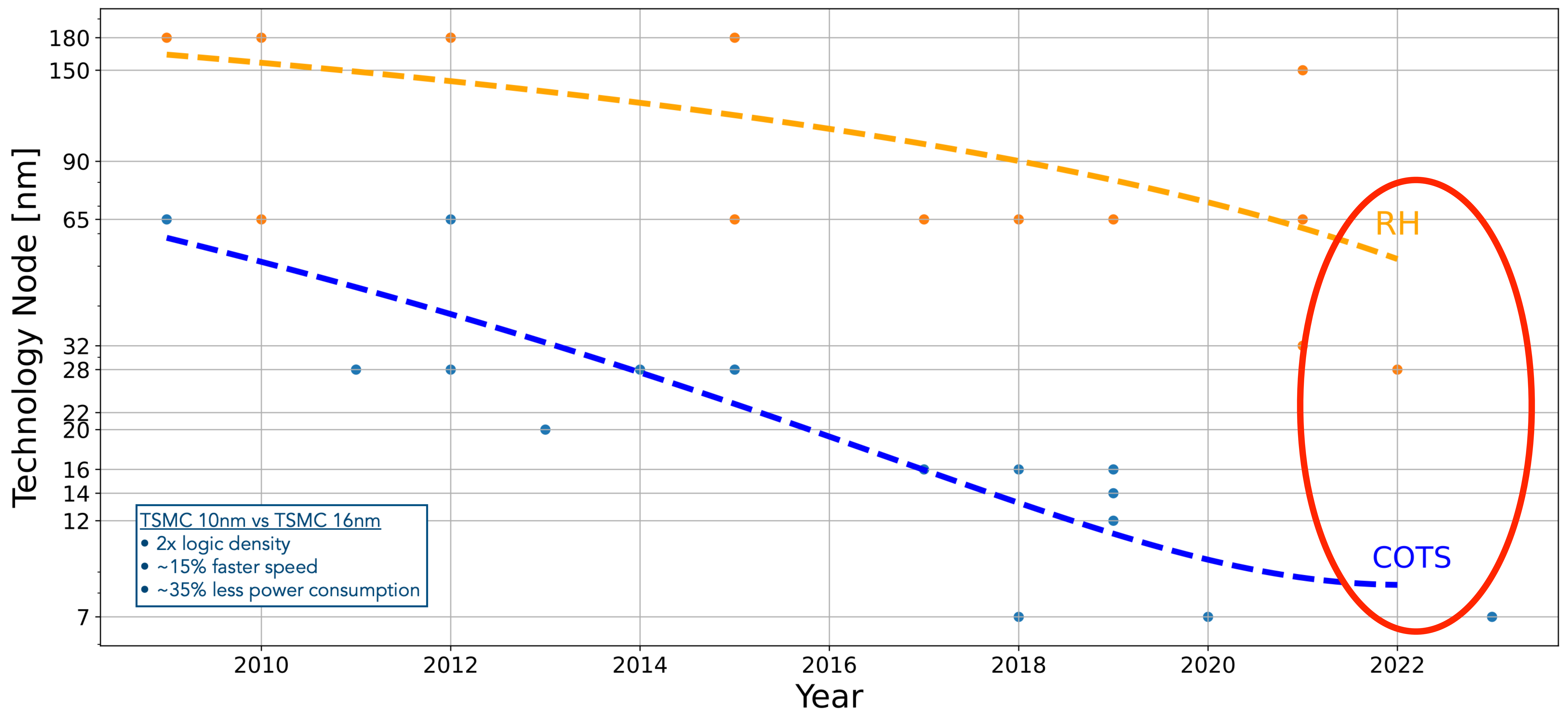
Architecture	Chip	Pros	Cons
GPU	NVIDIA Xavier, NVIDIA Jetson Orion, AMD Ryzen	COTS, High performance (up to 300 TOPS and 5 TOPS/W)	Not rad-hard, only short missions, only LEO
VPU	Intel Myriad X		
TPU	Google Coral		
FPGA	Xilinx Zynq UltraScale+, Microchip PolarFire	Flight heritage, Rad-hard, Flexible, Reprogrammable	General purpose, Relatively high energy consumption and low performance
CPU	Gaisler GR740		
Spatial Architecture	Xilinx Versal	Rad-tolerant, High performance (up to 430 TOPS)	Low interpretability (“black box”) Low predictability in time critical tasks, Relatively high power consumption (35W)
Systolic Array	HPDP	High flexibility High reconfigurability Low power consumption	Relatively old technology (STM 65 nm), Proprietary tools

SotA – Technology Available over the Years



G. Mystkowska, M. Monopoli, P. Nannipieri, L. Zulberti, D. Merodio Codinachs and L. Fanucci,
"Hardware Platforms Enabling Edge AI for Space Applications: A Critical Review," in IEEE Access, doi: 10.1109/ACCESS.2025.3596326.

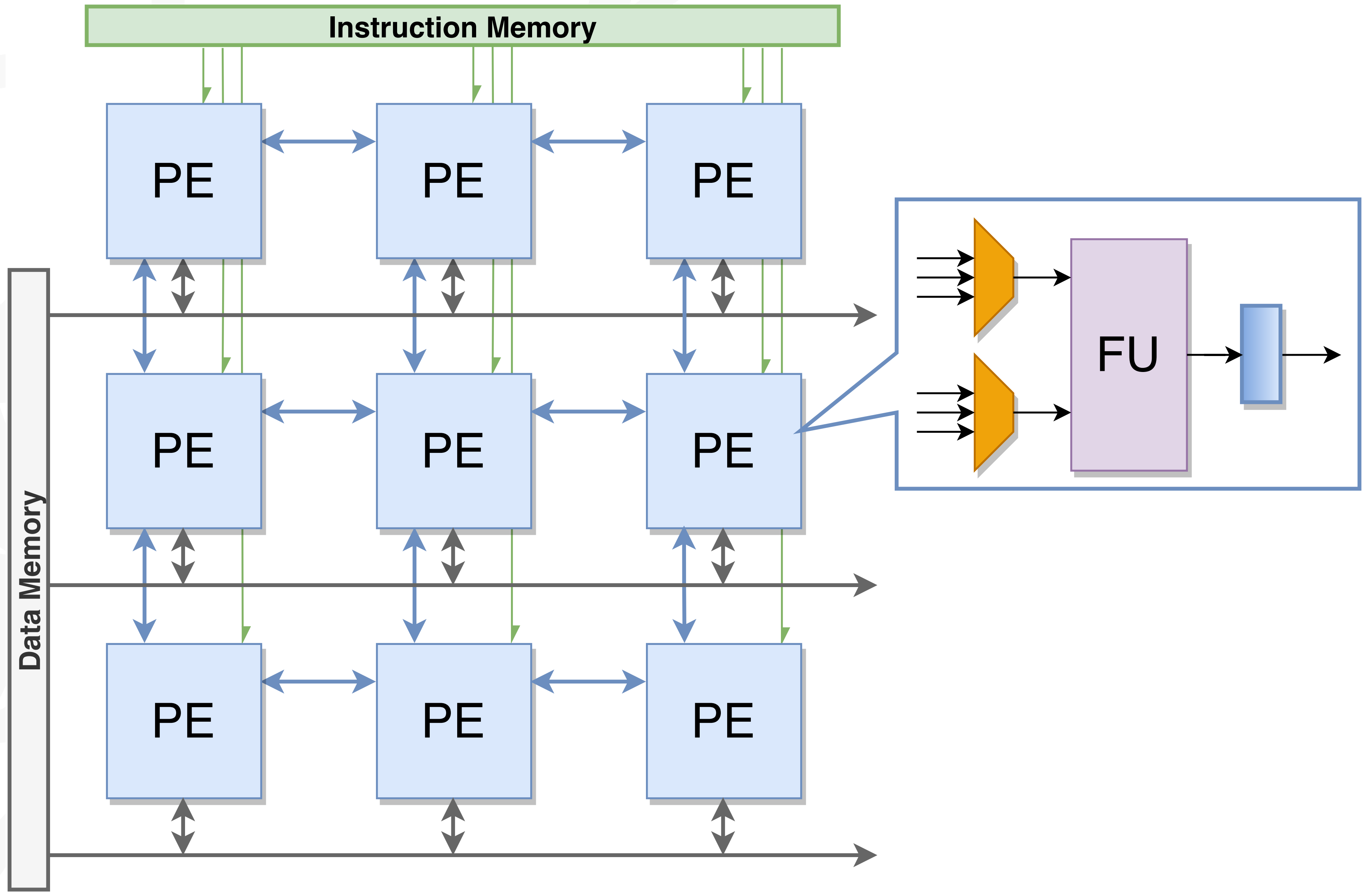
SotA – Technology Available over the Years



G. Mystkowska, M. Monopoli, P. Nannipieri, L. Zulberti, D. Merodio Codinachs and L. Fanucci,
"Hardware Platforms Enabling Edge AI for Space Applications: A Critical Review," in IEEE Access, doi: 10.1109/ACCESS.2025.3596326.

HW Accelerator - resource optimised CGRA

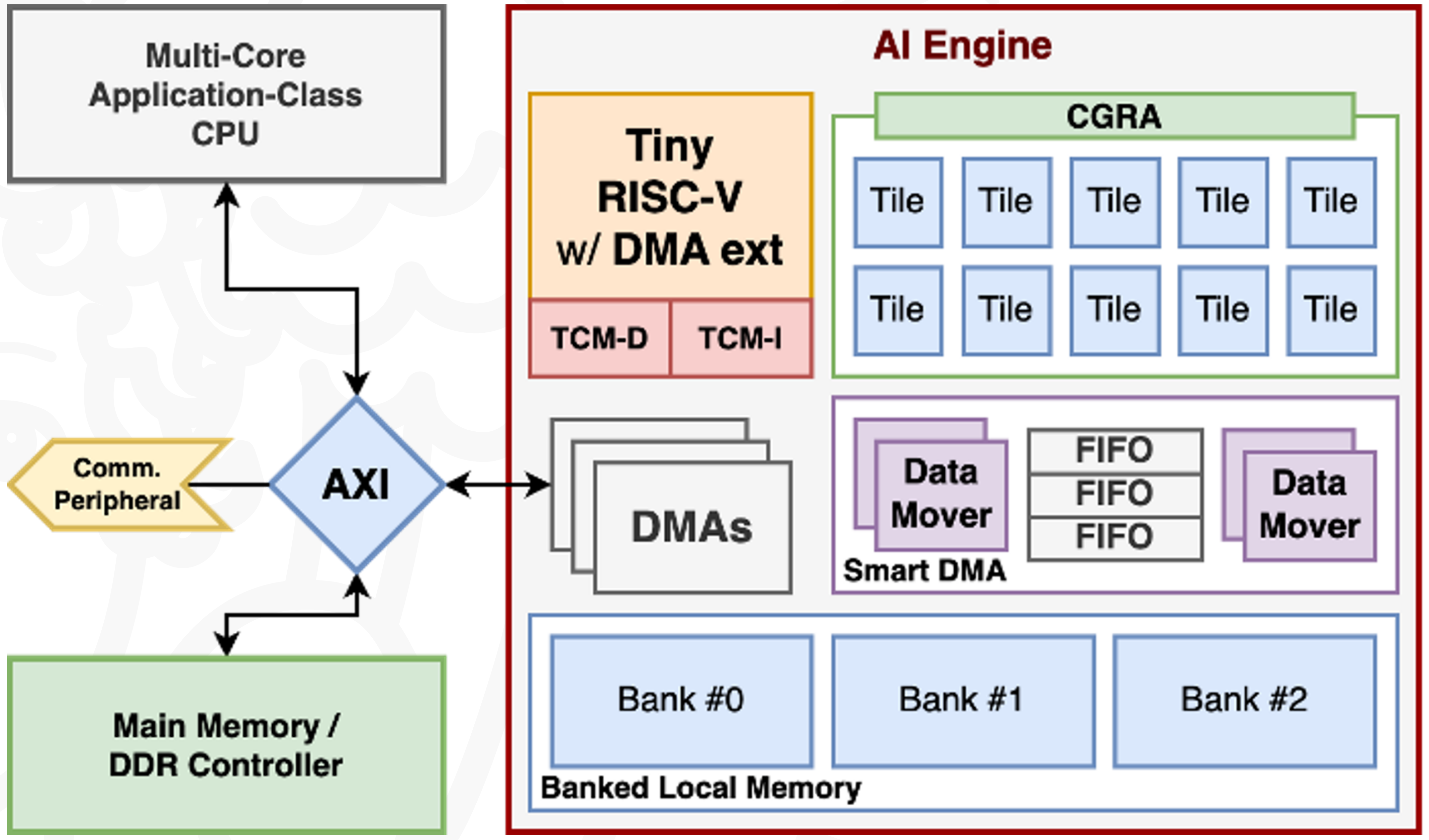
- 2D array of PE interconnected through NoC
- Speeding up **compute-intensive inner** loops
  - Signal processing
  - AI/ML workloads
- **Heterogeneous** computing, high **parallelism**
- Highly **predictable** (timing model)
- Highly **parametrised** architecture
  - Optimised resource utilisation
  - **Scalable** and **Flexible**
- Low reconfiguration time



CGRA is	power efficient	compared to	FPGA
	power efficient, high-performance		CPU
	more flexible		Systolic array
	more flexible		ASIC

UniPi's CGRA for AI

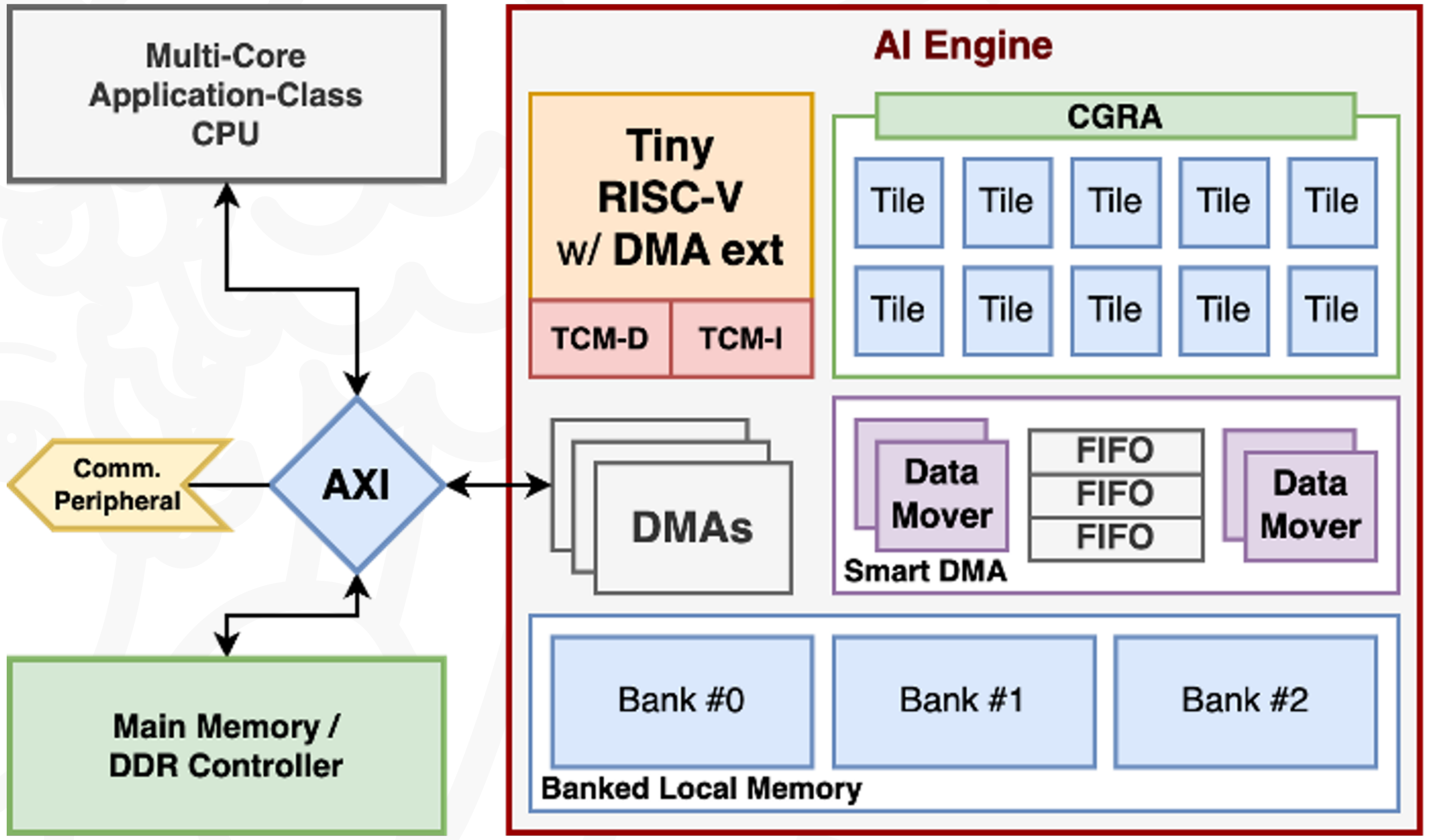
- CGRA-based accelerator for AI on-edge applications
  - **Time-constrained** applications (e.g., autonomous operations)
  - **High reliability** applications (e.g., space industry)
- Features
  - **Flexible**, runtime reconfiguration
  - **Energy-efficient**
- Technology:
  - 65 nm, 40 nm
  - **7 nm radiation-hardened** (in progress)
- Heritage of:
  - Three ongoing ESA supported projects
  - OPERAND project supported by Italian Ministry of Education and Research



OSIP I-2021-03237 Innovative Coarse-Grained Reconfigurable Array Platform for Computing Artificial Intelligence On-Board
OSIP I-2022-04765 Risc-V Based SoC Featuring A Soft-GPU Hardware Accelerator for Artificial Intelligence On-Board
OSIP I-2023-09415 UDSM AI-engine For Reliable, Energy-efficient Next-generation Satellites
L. Zulberti, M. Monopoli, P. Nannipieri and L. Fanucci, "Architectural Implications for Inference of Graph Neural Networks on CGRA-based Accelerators,", doi: 10.1109/PRIME55000.2022.9816810.
L. Zulberti, M. Monopoli, P. Nannipieri, L. Fanucci and S. Moranti, "Highly Parameterised CGRA Architecture for Design Space Exploration of Machine Learning Applications Onboard Satellites,", doi: 10.23919/EDHPC59100.2023.10396632.

CGR-AI Engine IP

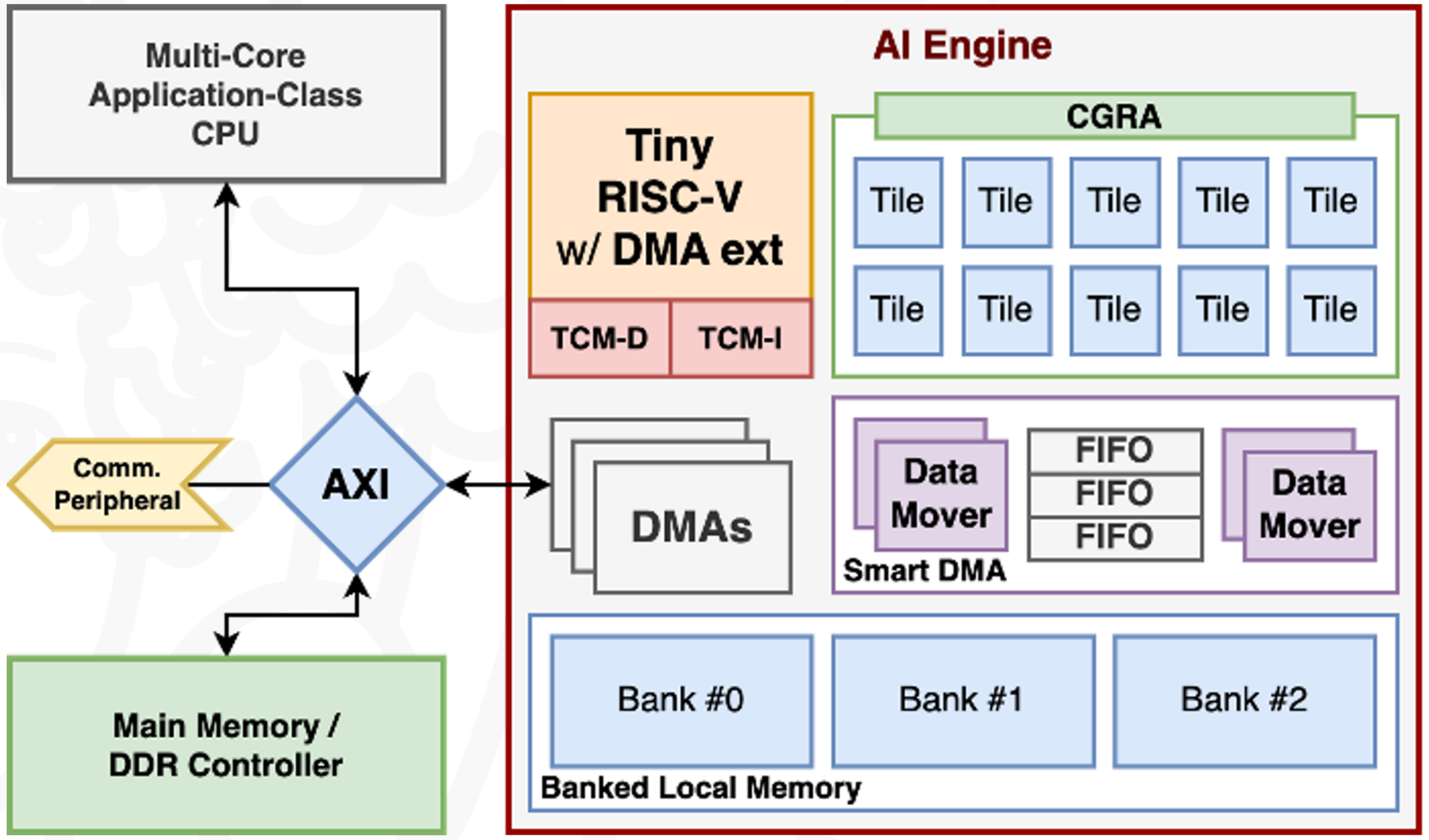
- AI-Engine IP
  - With CGRA core
  - With a simplified programming environment
    - CGRA Configurations
    - Scheduling Firmware
- What we have:
  - CGRA core
  - Smart DMA
  - Tiny RISC-V with DMA extension
- My contribution to the project:
  - SoC integration
  - Simplified programming environment
  - Synthesis in UDSM 7nm technology



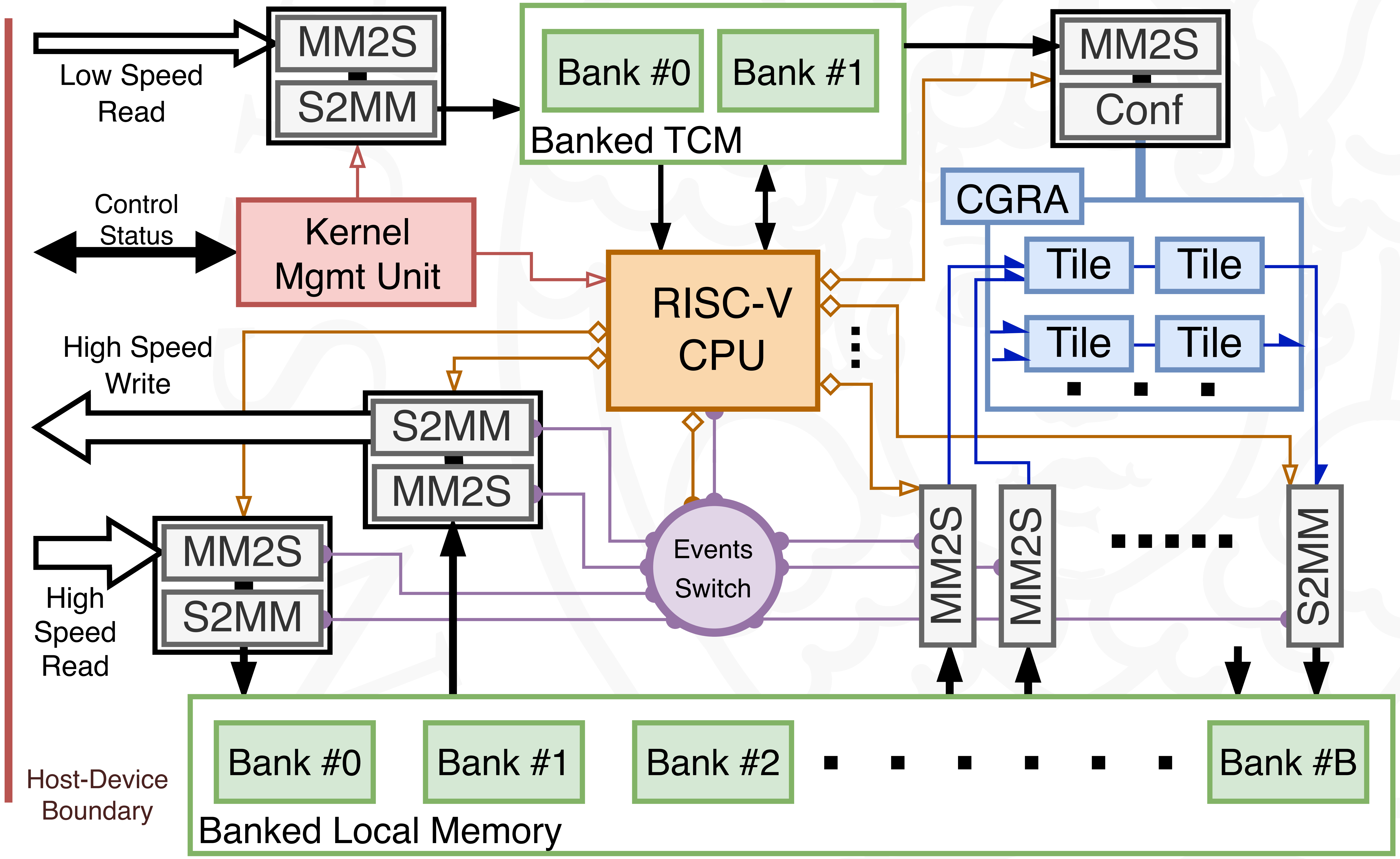
CGR-AI Engine IP

- AI-Engine IP
  - With CGRA core
  - With a simplified programming environment
    - CGRA Configurations
    - Scheduling Firmware
- What we have:
  - CGRA core
  - Smart DMA
  - Tiny RISC-V with DMA extension
- My contribution to the project:
  - SoC integration
  - Simplified programming environment
  - Synthesis in UDSM 7nm technology

Functional testing



CGR-AI Functional Walkthrough

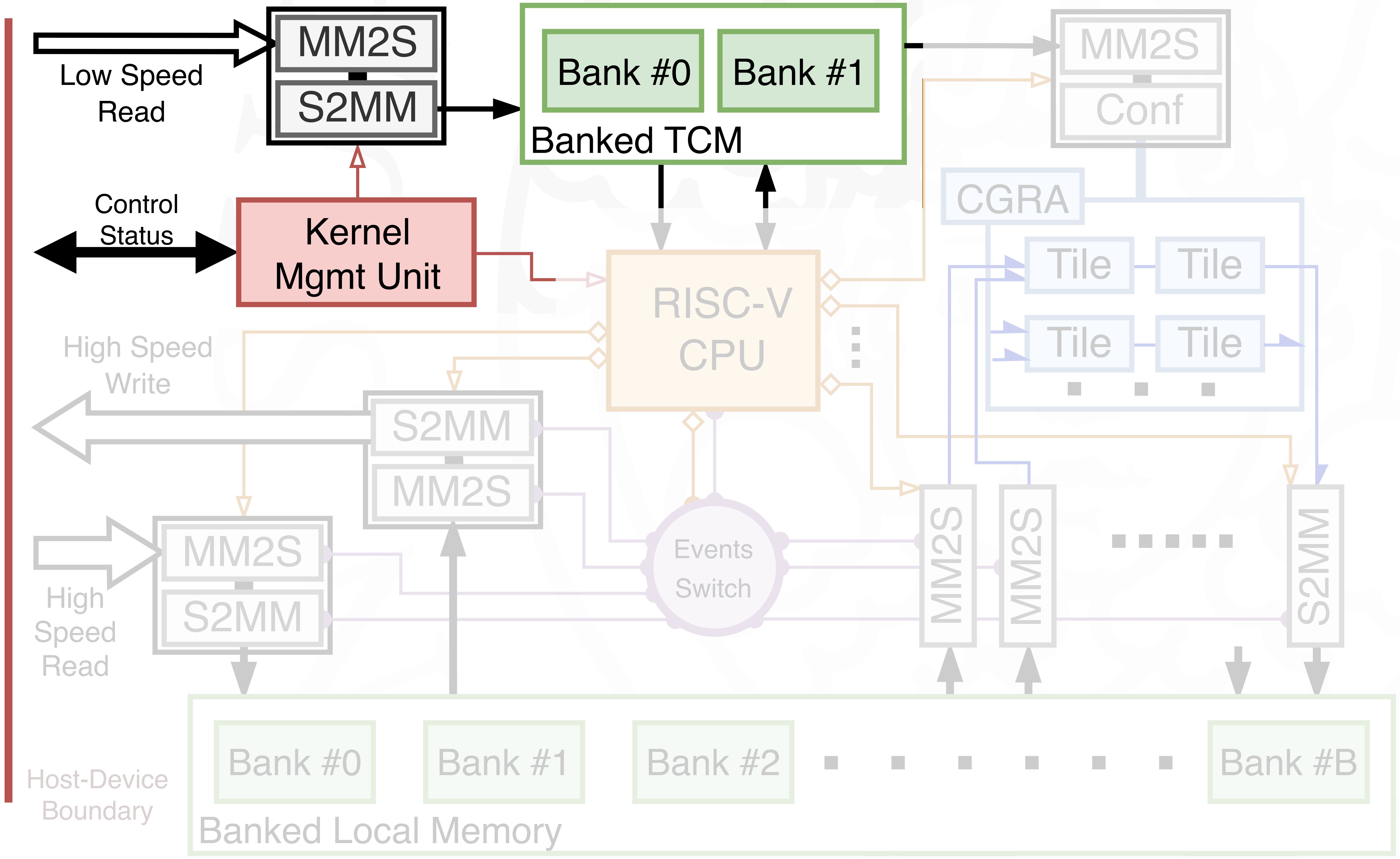


Operations

- **Kernel loaded** in TCM
- CGRA is **configured**
- Data **read from Ext** to L. Mem
- Data **processed in CGRA**
- Data **written to Ext** Mem

Functional Walkthrough

1/5

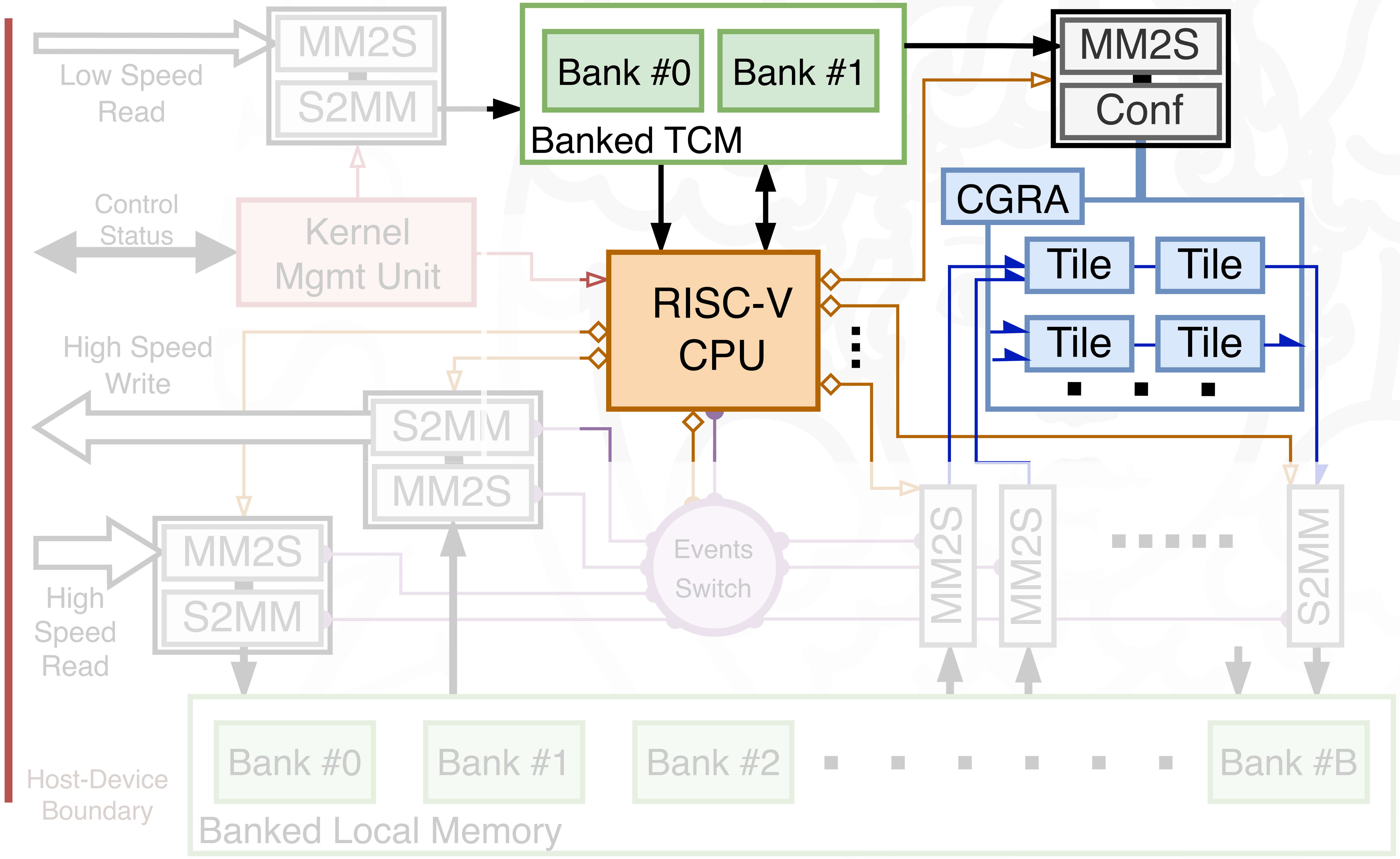


Operations

- **Kernel loaded** in TCM
- CGRA is **configured**
- Data **read from Ext** to L. Mem
- Data **processed in CGRA**
- Data **written to Ext** Mem

Functional Walkthrough

2/5

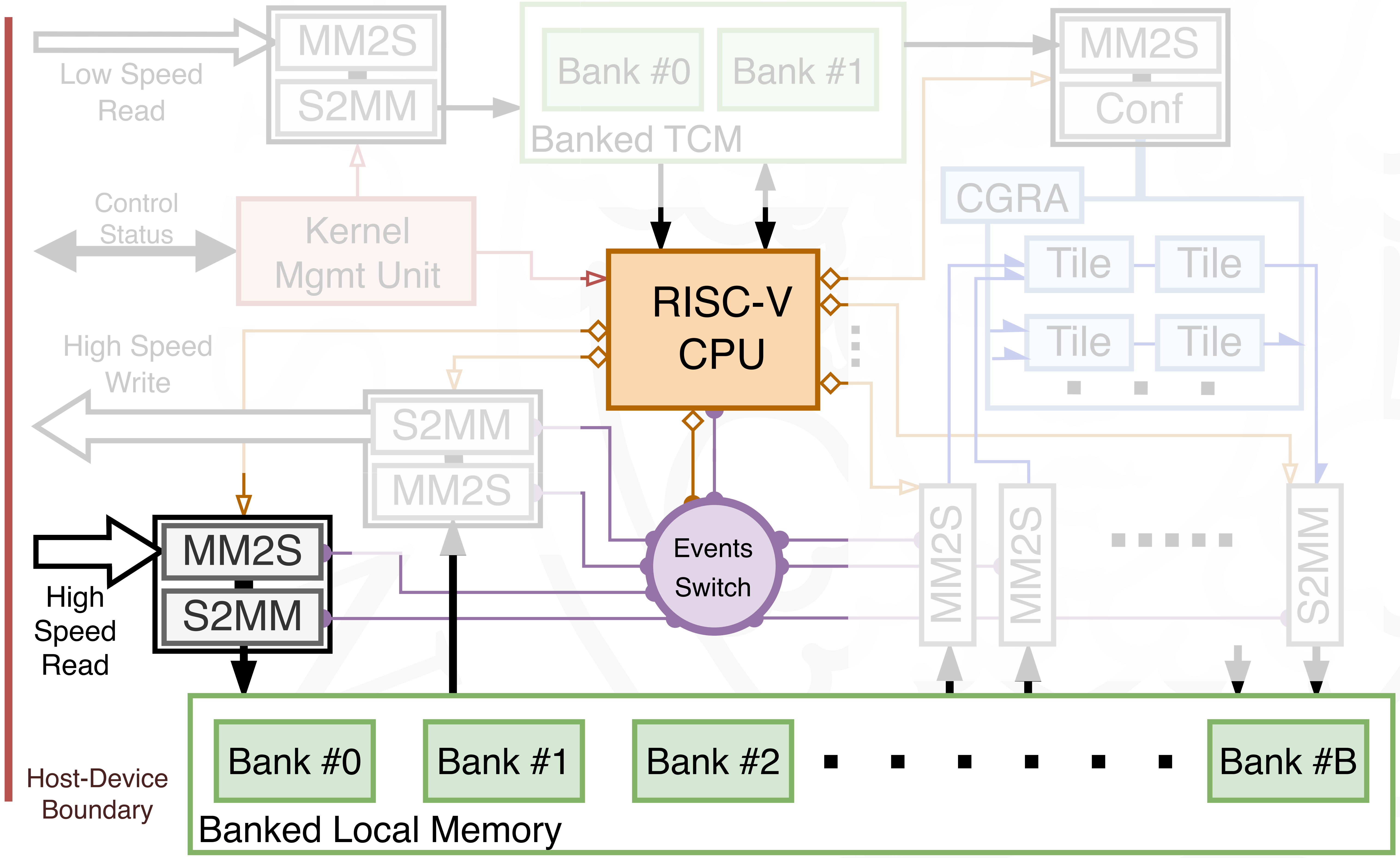


Operations

- **Kernel loaded** in TCM
- CGRA is **configured**
- Data **read from Ext** to L. Mem
- Data **processed in CGRA**
- Data **written to Ext** Mem

Functional Walkthrough

3/5

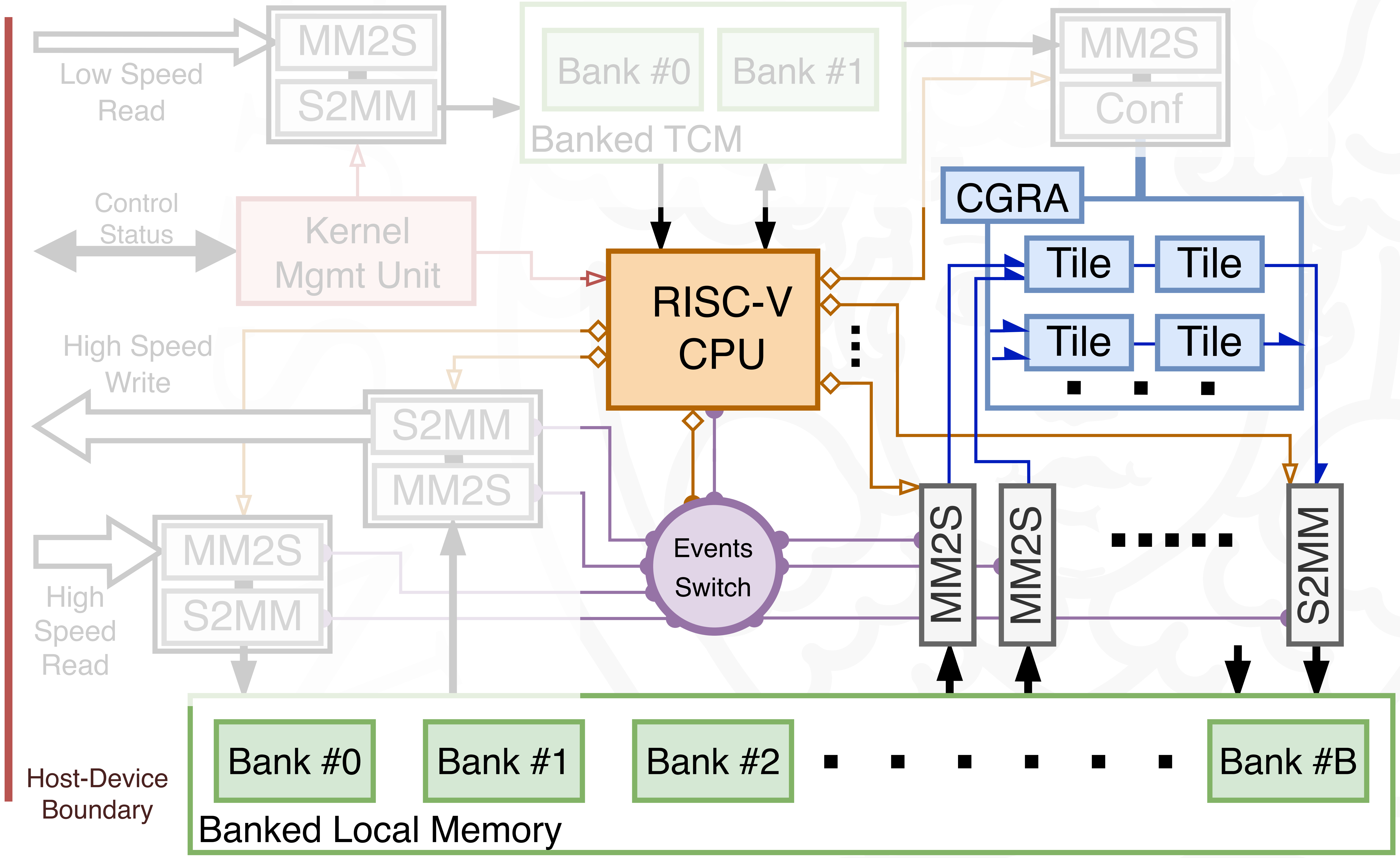


Operations

- **Kernel loaded** in TCM
- CGRA is **configured**
- Data **read from Ext** to L. Mem
- Data **processed in CGRA**
- Data **written to Ext** Mem

Functional Walkthrough

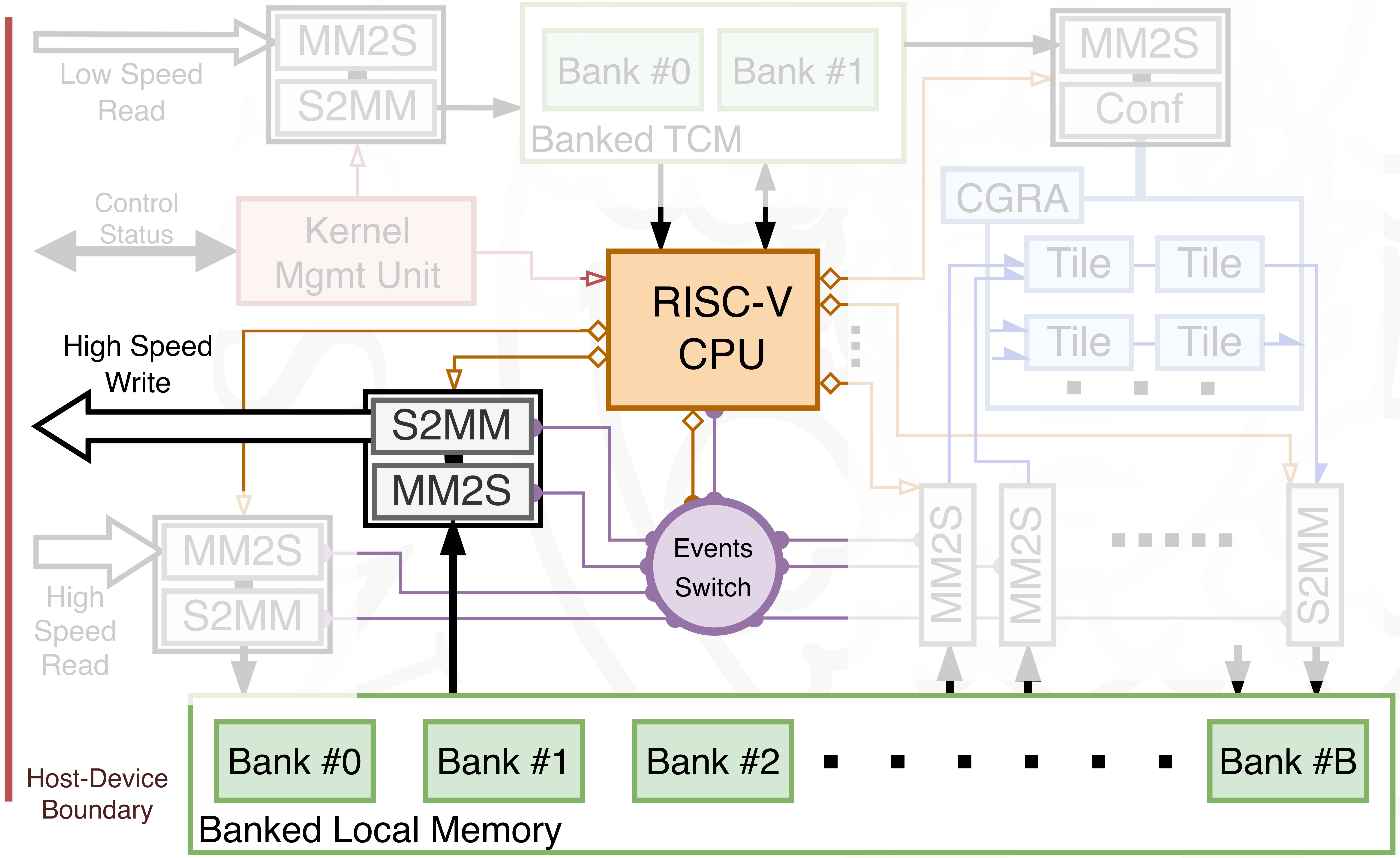
4/5



Operations

- **Kernel loaded** in TCM
- CGRA is **configured**
- Data **read from Ext** to L. Mem
- Data **processed in CGRA**
- Data **written to Ext** Mem

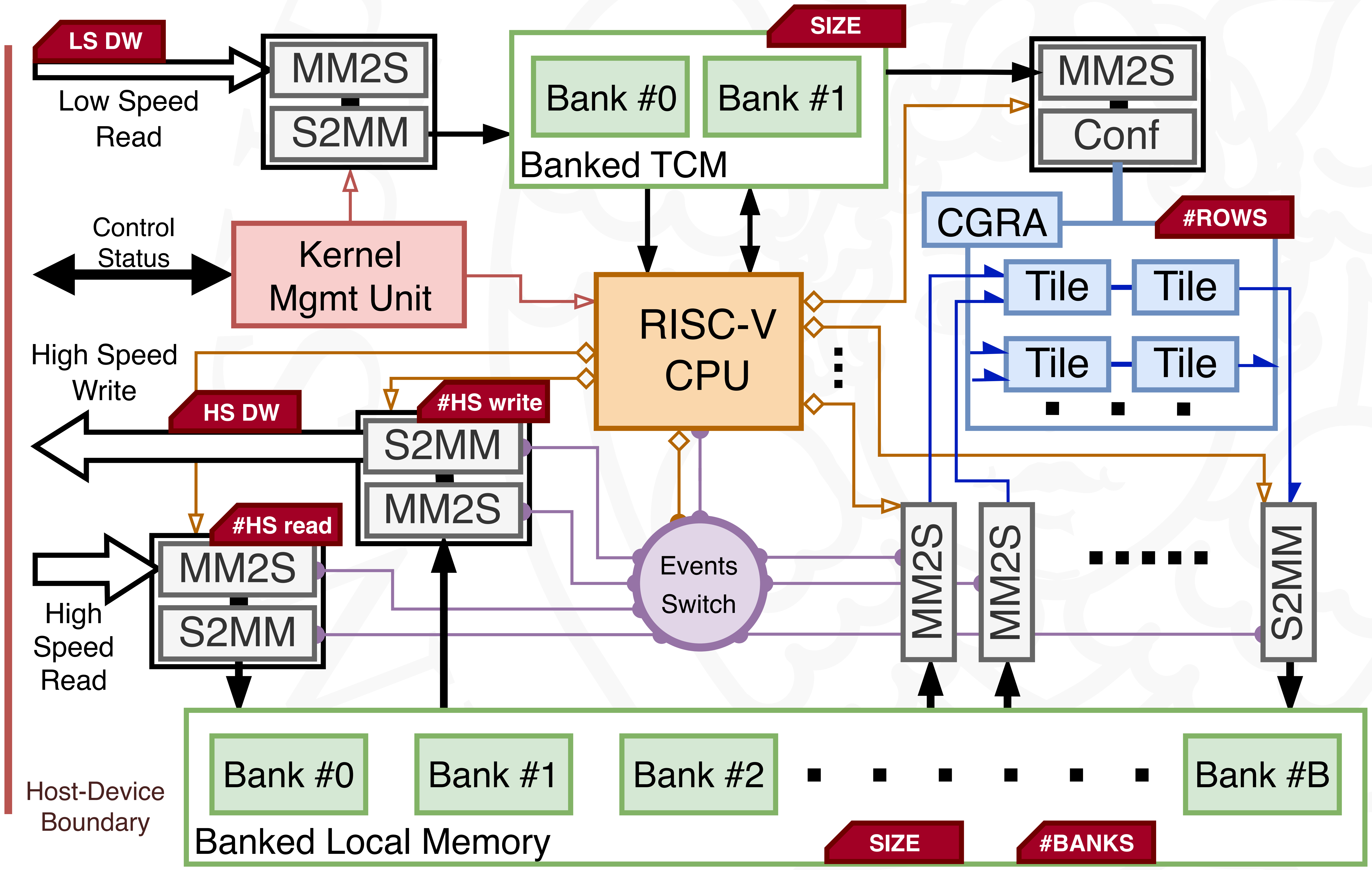
Functional Walkthrough 5/5



Operations

- **Kernel loaded** in TCM
- CGRA is **configured**
- Data **read from Ext** to L. Mem
- Data **processed in CGRA**
- Data **written to Ext** Mem

CGR-AI and its Parameterisation

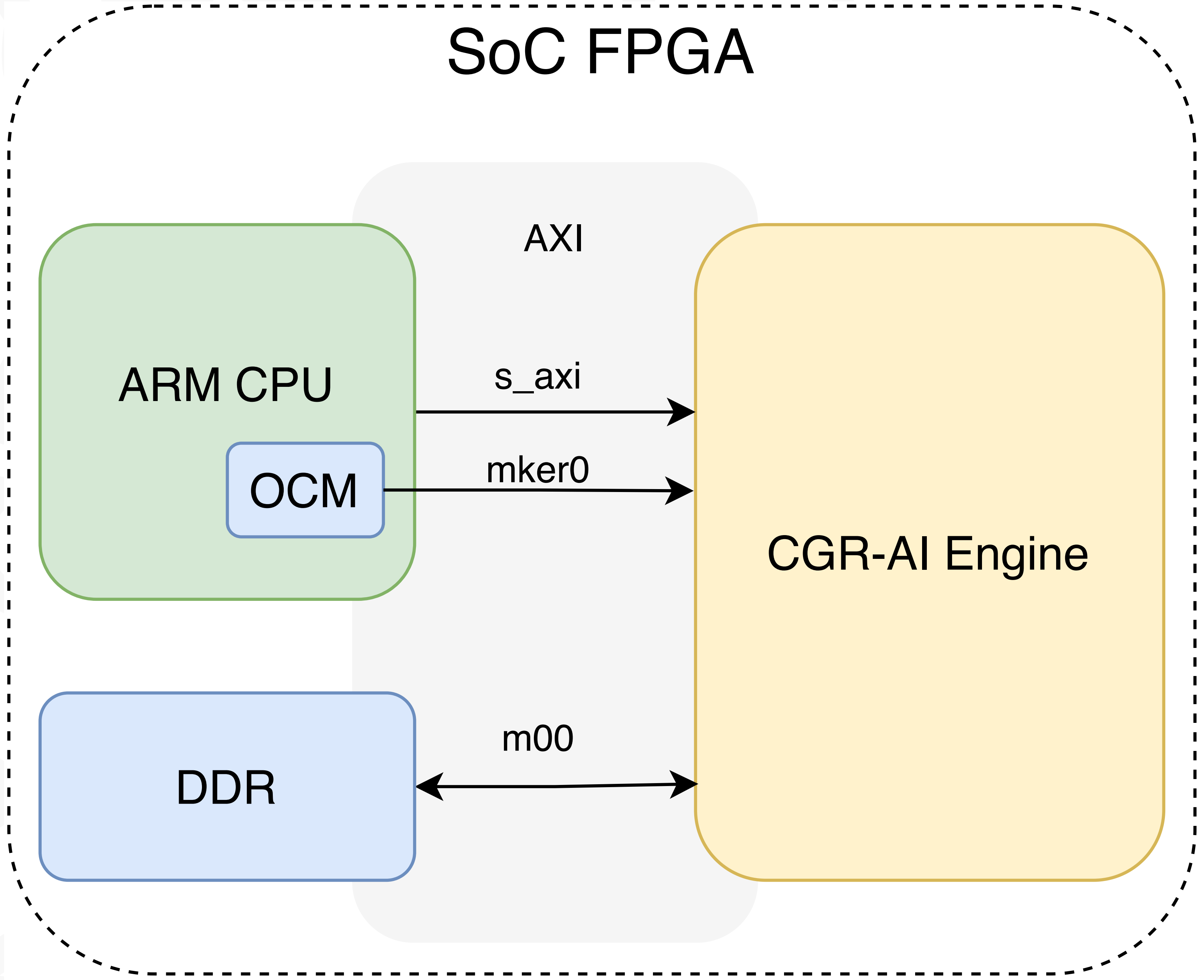


Parameters

- Number of **CGRA rows**
- TCM **size**
- Local Memory **size** and **banks**
- Number of memory **interfaces** and **data width**

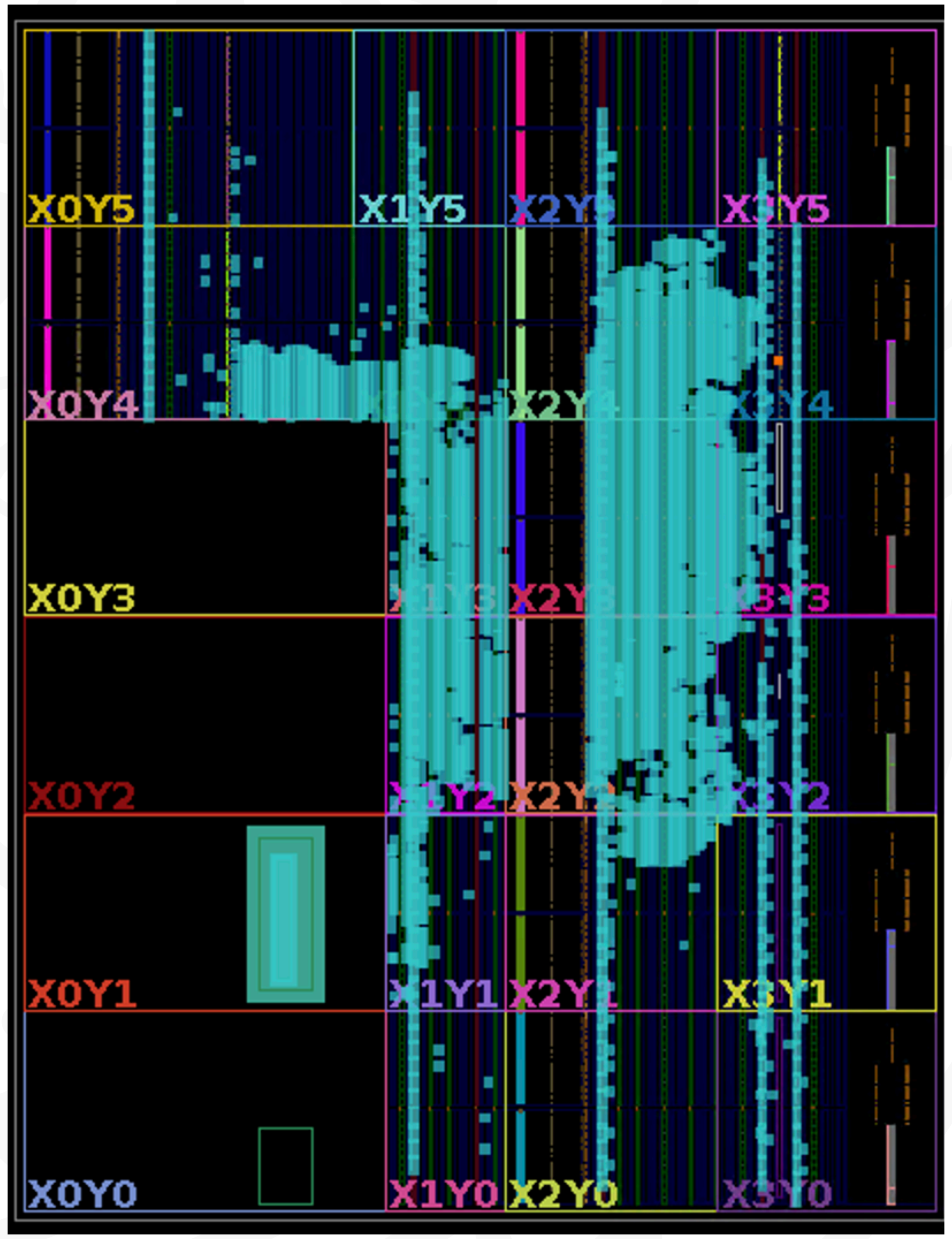
FPGA overlay - strategy

- To confirm **functionalities** and **performance**, which then will be scaled up directly with the numbers obtained by the std-cell synthesis
- IP approach for ease of portability
- Xilinx ZCU104
  - SoC ARM for:
    - CGR-AI configuration
    - Data transferes monitoring
    - Interface for debugging signals
- AXI
  - `s_axi` for CPU configuration
  - `mker0` for kernel `.bin` file load
  - `m00` for input and output data



FPGA implementation

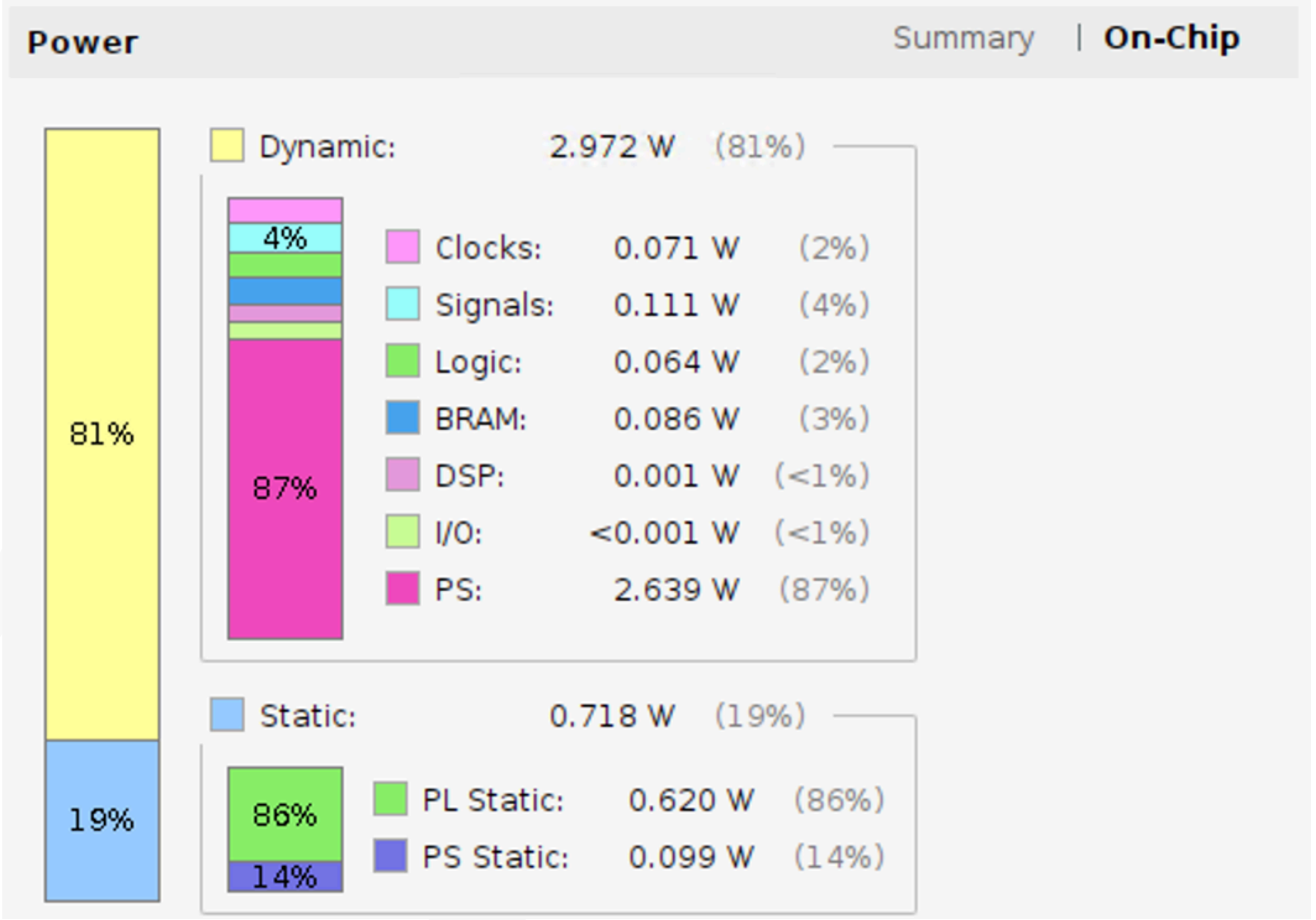
- Resource utilisation
  - LUT < 14%
  - FF < 7%
  - BRAM ≈ 90%
- Power → 3,7 W
  - Most power used by PS (≈ 90%)
- Timing @ 100 MHz
  - WNS → 2,1 ns



3x2 CGRA with 9MB of local memory

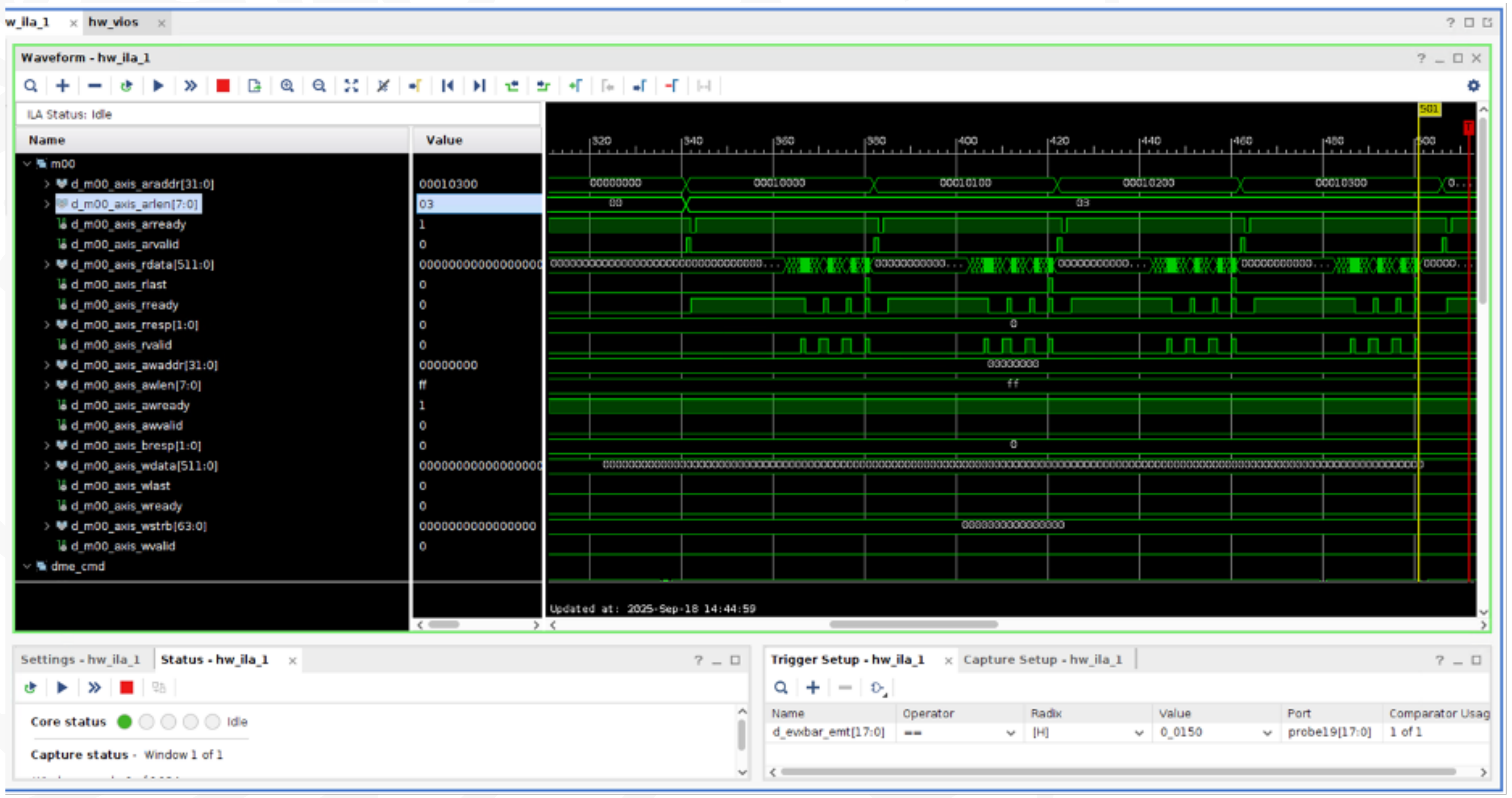
Utilization — Post-Synthesis | **Post-Implementation** — Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	30983	230400	13.45
LUTRAM	139	101760	0.14
FF	30055	460800	6.52
BRAM	272	312	87.18
DSP	4	1728	0.23
IO	1	360	0.28



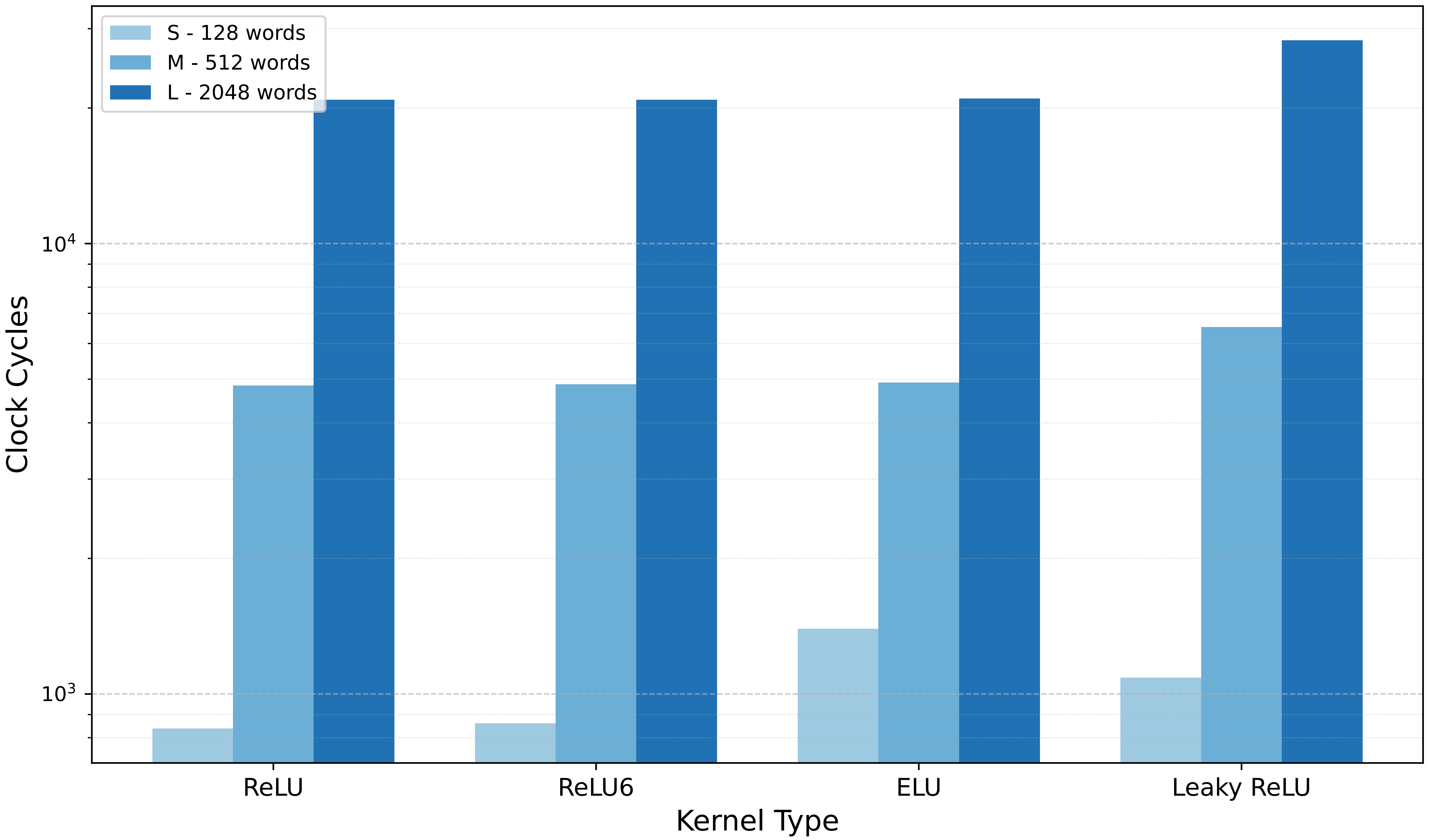
FPGA implementation

- Verification and debugging
  - Hardware in the loop
    - Use of ILA
  - FPGA implementation vs simulation
    - The output data matches the output data obtained in Questa simulation confirming the functional accuracy of the implementation



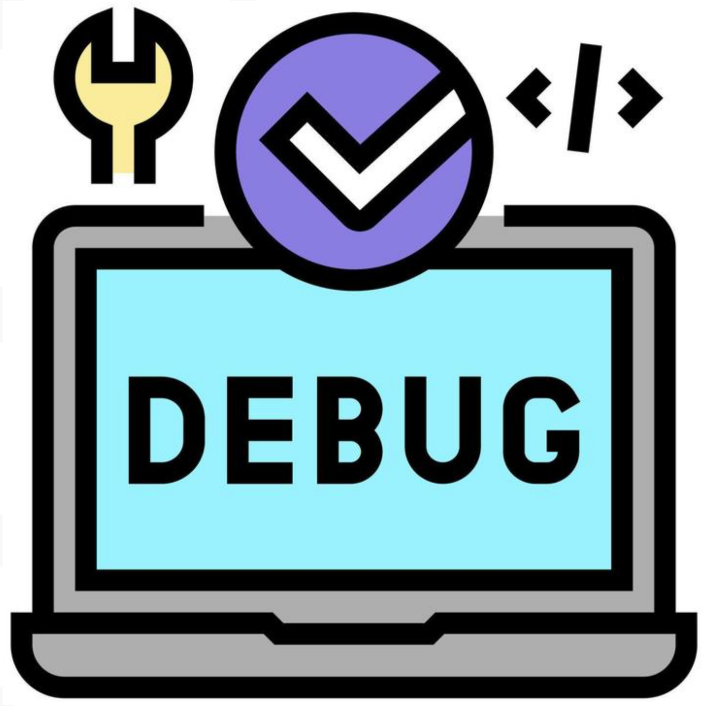
Proof of concept

- Processing data 4× larger required ≈ 5.25× more clock cycles.
  - Increased latency due to memory access and bandwidth limitations.
- Verified:
  - data flow
  - synchronization
- Successful kernel execution
- The Engine is ready for further performance optimization and scaling



Lessons learned for improved user experience

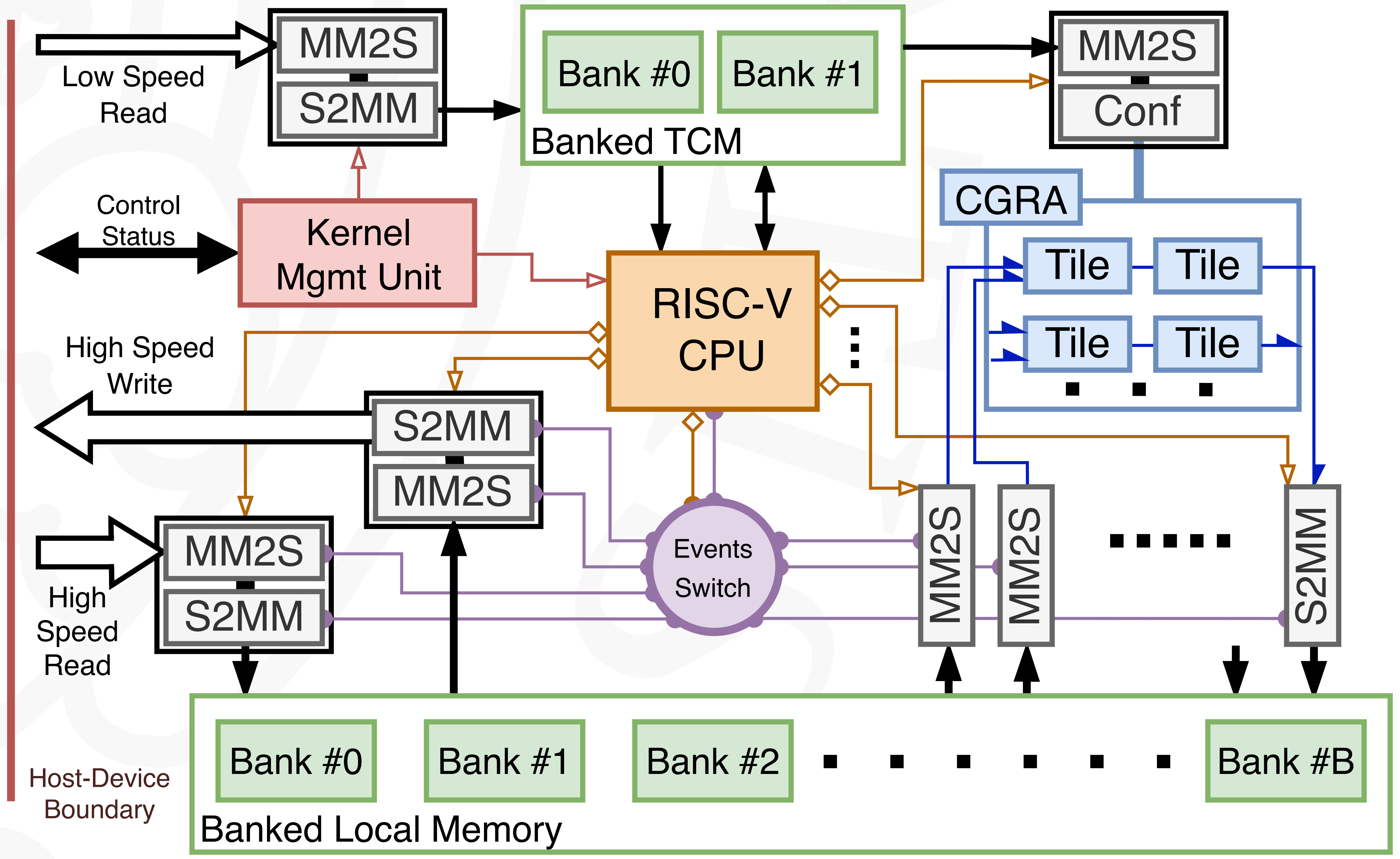
- Parametrisation
  - **Challenge:** Finding the optimal parameters was more complex than expected.
  - **Impact:** Suboptimal configurations led to difficulties in deployment.
  - **Lesson:** Establish a systematic approach for parameter tuning early in development, limit the number of parameters.
- RISC-V with debugging core
  - **Previous approach:** Debugging core was omitted to save silicon area and reduce power consumption.
  - **Result:** Limited visibility into internal operations made troubleshooting time-consuming and error-prone.
  - **Lesson:** Including a debugging core, even with slight overhead, is crucial for faster issue resolution, improved maintainability, and better overall user experience.



FPGA Prototype of CGR-AI Engine for Space Systems: Step Towards UDSM Implementation

Gabriela Mystkowska
gabriela.mystkowska@phd.unipi.it

- **CGRA-core** for calculation acceleration
  - Heterogeneous computing
  - Scalable and flexible
  - Low reconfiguration time
- **Successful** implementation of the CGR-AI Engine on FPGA
  - System integration ✅
  - Compliance with AXI interface ✅
- **Proof of concept**, successful kernel execution ✅
- The Engine is ready for further performance optimization
  - Stepping stone for **UDSM 7nm** synthesis and implementation



This contribution is sponsored by **ESA Education**

OSIP I-2023-09415 UDSM AI-Engine for Reliable, Energy-Efficient Next-Generation Satellites