CEI UPM | Centro de Electrónica Industrial

Self-adaptation of Cyber Physical Systems: Flexible HW/SW computing

Eduardo de la Torre
eduardo.delatorre@upm.es

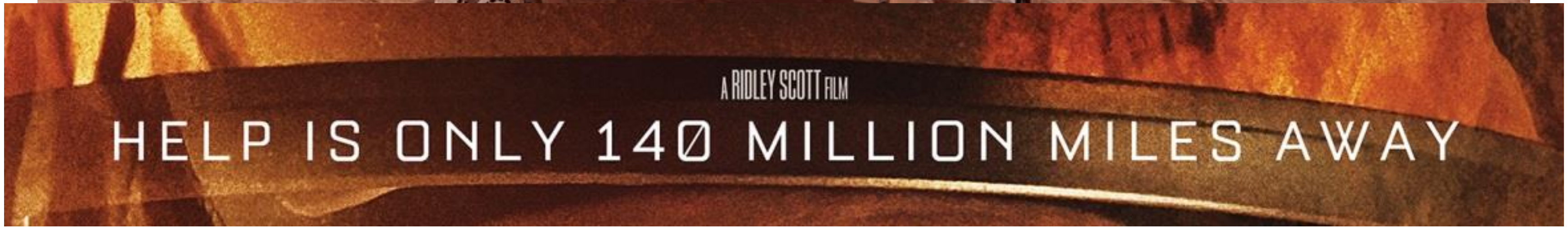
Summer School on Cyber Physical Systems
Alghero – 25/29 Septmber 2017

UNIVERSIDAD POLITÉCNICA DE MADRID

cei@upm.es

POLITÉCNICA

A use case for motivation: the SPACE environment



Why adaptation in space?

- Cost is becoming as determinant as radiation
- Resources are starting to be shared
  - Same cameras used for navigation and scientific missions
  - Algorithms change from one function to other → Reconfiguration needed
  - Flash memories for space cannot be rewritten that often
  - SRAM-based FPGAs with many design protections are being considered
- New business models are being projected
  - Farms of satellites offering specific services (Power, communications)
- But, out of LEO, MEO or GEO, systems are fully alone
  - Nobody knows what will be there
  - Some missions are commanded to define Science on the spot
  - Autonomy (Self-*) is required

Autonomous System Adaptation

Autonomic Computing

[Horn 2001; Kephart et al. 2003; March 2004]

Address complexity and adaptation needs of future applications

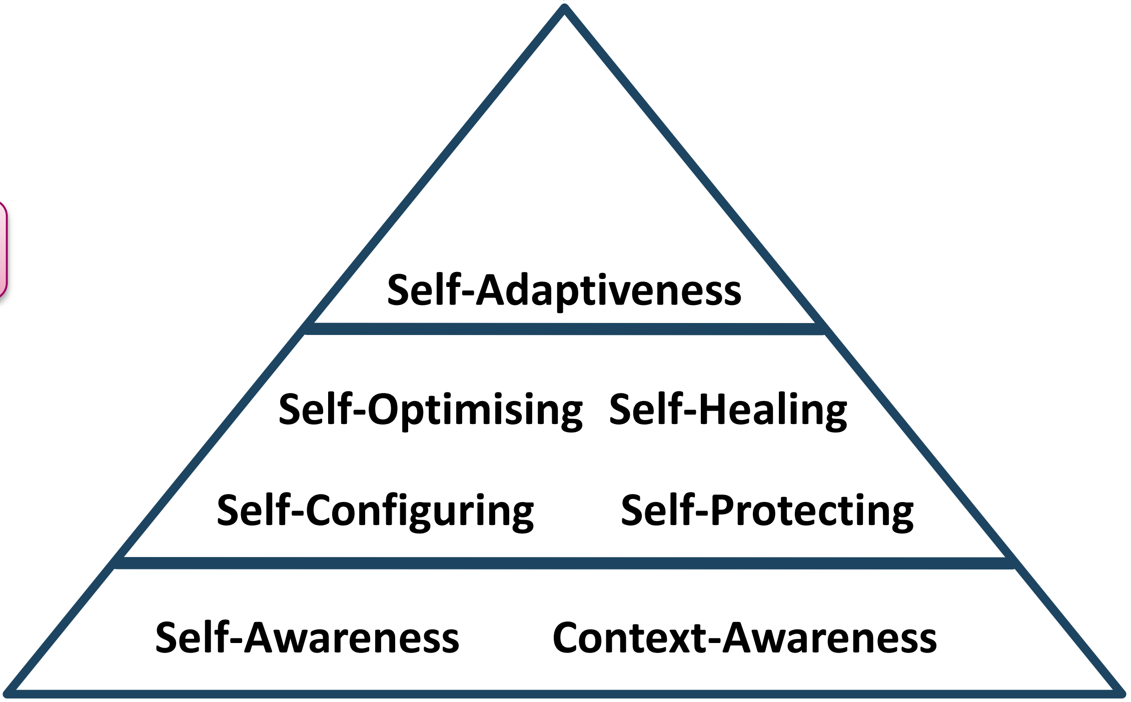
Accomplish high level goals

- **No external control**
- **No human intervention**
- **Runtime decision taking**
- **Optimising behaviour**

Self-adaptive systems

Self-* properties

[Salehie et al. 2009]



Towards more robust and autonomous systems

	Levels of autonomy	With reconf. devices
8	System learns	Evolvable HW
7	System grows	Scalable HW systems
6	System acts	Adaptability, self-healing
5	System searches	Evolutionary algorithms
4	System observes	Power-aware systems
3	Designer provides behaviour	Offline genetic algorithms
2	Designer provides capability	Self-configuration
1	Designer provides configuration	Remote configuration
0	Designer creates system	Off-line design

Source: 'HW autonomy and space systems', Steiner & Athanas
IEEE Trans on Automation Control, 2009

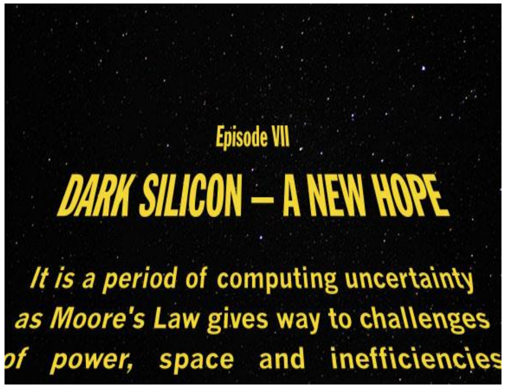
CEI UPM

POLITÉCNICA

Can reconfigurable computing help in this?

- **Time-multiplexing of tasks**
  - Complex systems in smaller devices
  - The IKEA concept applied to circuits

- **Reduction of energy consumption**
  - Dark silicon is reduced (dark silicon occupies area, spends energy, and does nothing!)
  - It allows to make more efficient circuits, since there is more place to do things better (quality, performance), or with less energy.


Episode VII
DARK SILICON – A NEW HOPE
It is a period of computing uncertainty
as Moore's Law gives way to challenges
of power, space and inefficiencies

- **Autonomous and adaptive systems**
  - Adaptable Security & Communication Standards
  - Bioinspired computing
  - Dependability by design

How to reconfigure?

Theoretically, reconfiguring an FPGA is as simple as changing a SW program:

It's just writing in a memory !!

Reconfigurable systems on heterogeneous platforms

FPGAs
and
MPRSoCs

FPGA-based conventional System

- Extended use in prototyping
- Competitive Solution in some products
  - Shortest Time to market
  - Largest time in Market
  - Lower Design Cost
- Mature Design methodologies

Reconfigurable Computing System

- Offers new SW/HW Design Space possibilities
- Real parallelism
- Chances for lowest energy consumption
- Requires New Design Methodologies

CEI UPM

POLITÉCNICA

THE ARTICO^3 ARCHITECTURE

ARTICo3: Motivation

+

Requirement

Independent of the task itself!

Urgent

Put as many people as possible to work, and save time

Confidential

Do not tell others what you are doing, protect from others

Critical

Let more than one do the same work and compare result

Dynamic and Partial Reconfiguration
(hardware acceleration + module replication)

How to transfer data efficiently

The memory side

The accelerators' side

Efficient transactions (coalesced accesses) are planned by the programmer,
So they can be more efficient than caches.
In Artico3, accelerators don't claim for data, these are given in an organized manner by a scheduler

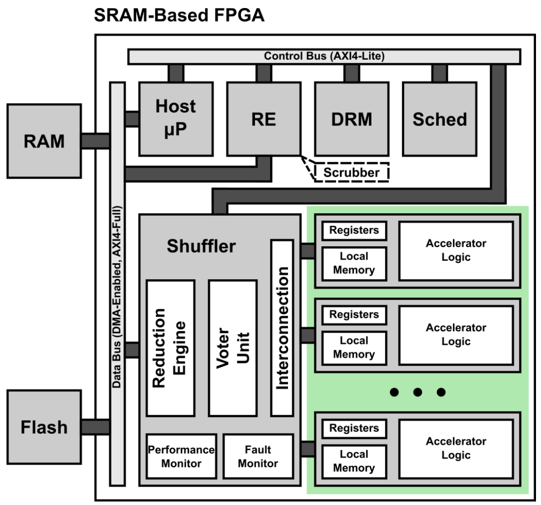
What is required?

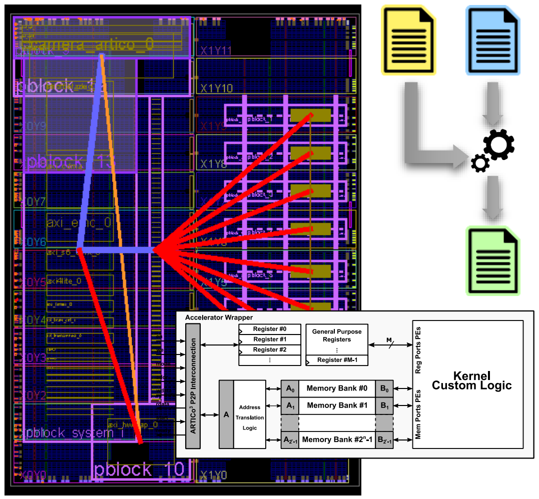
High Performance Embedded Computing Platforms

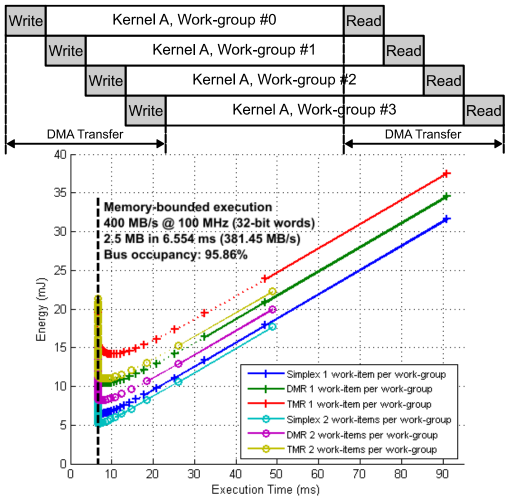
Architectures

Design Flows

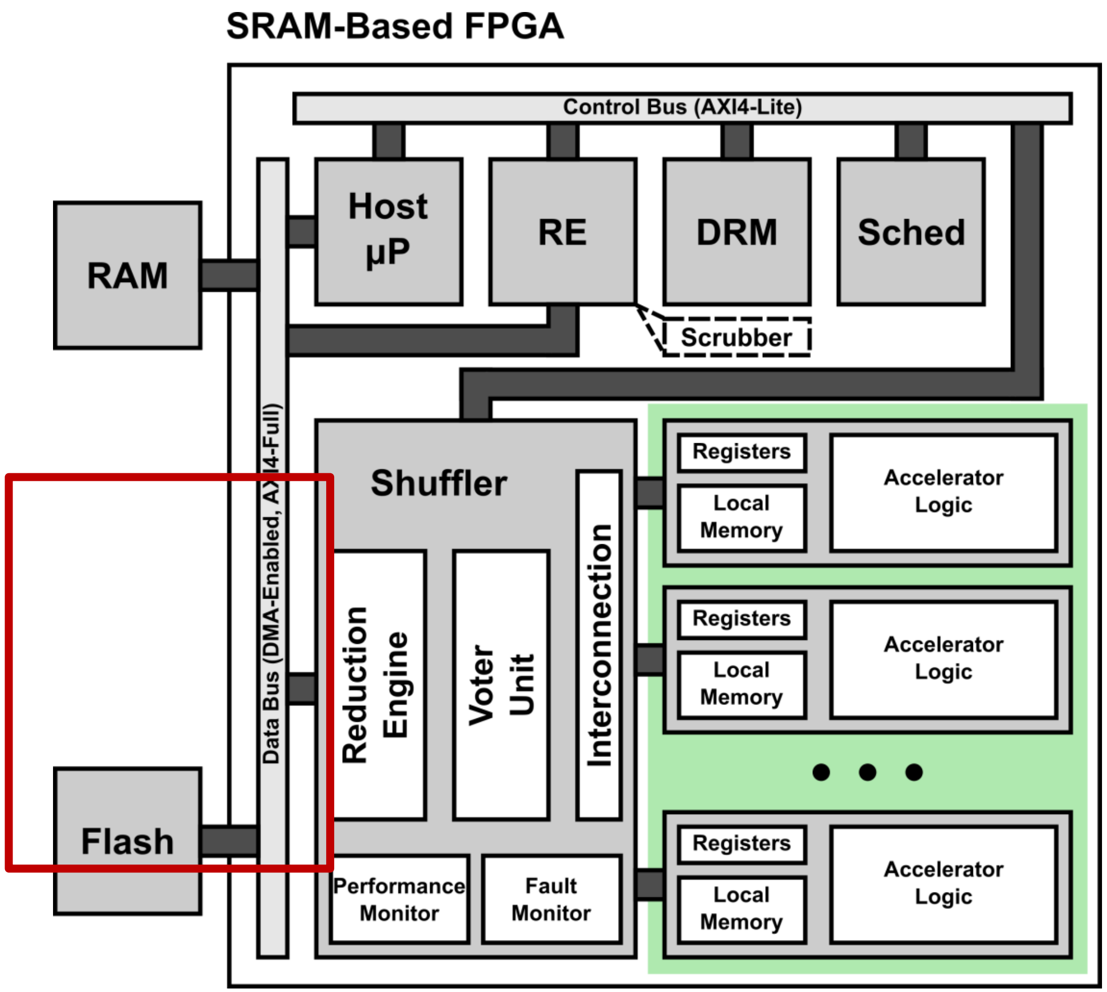
Runtime Environment







ARTICo3: Computing vs consumption vs fault tolerance



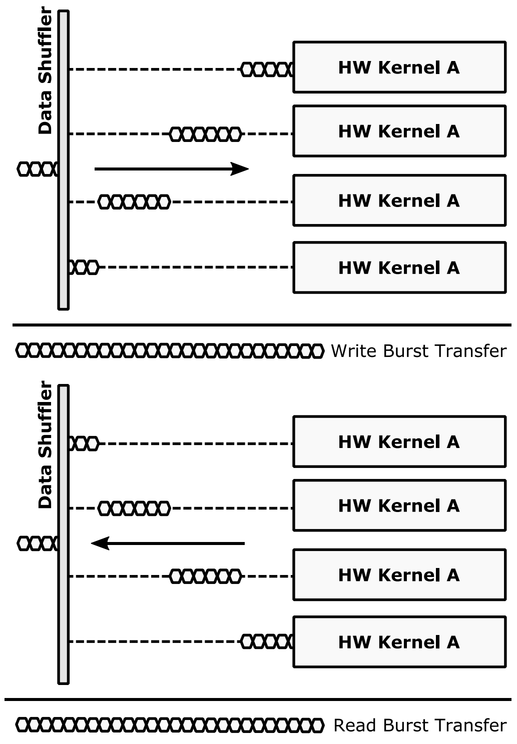
Smart Gateway between AXI4 and Custom P2P Communication

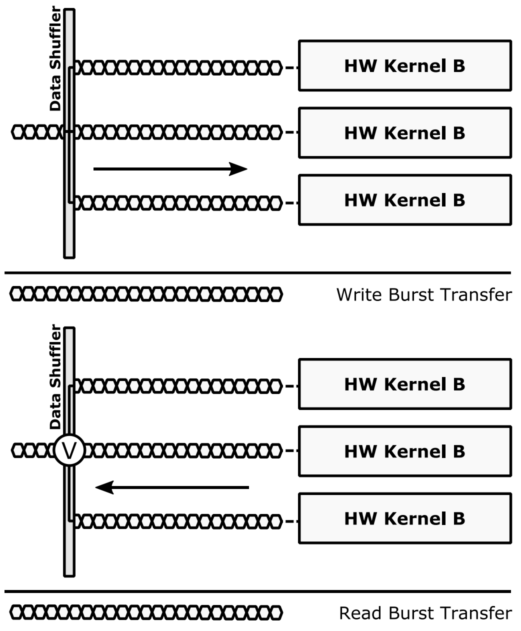
Data Transaction Modes

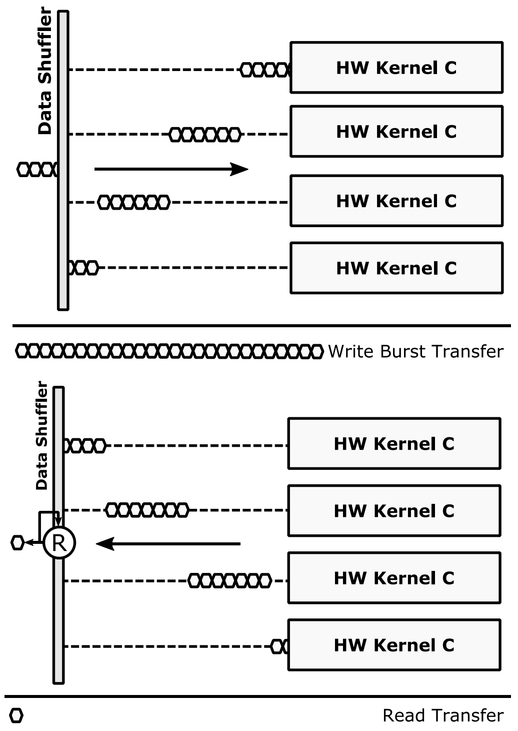
Parallel — Urgent

Redundant — Confidential Critical

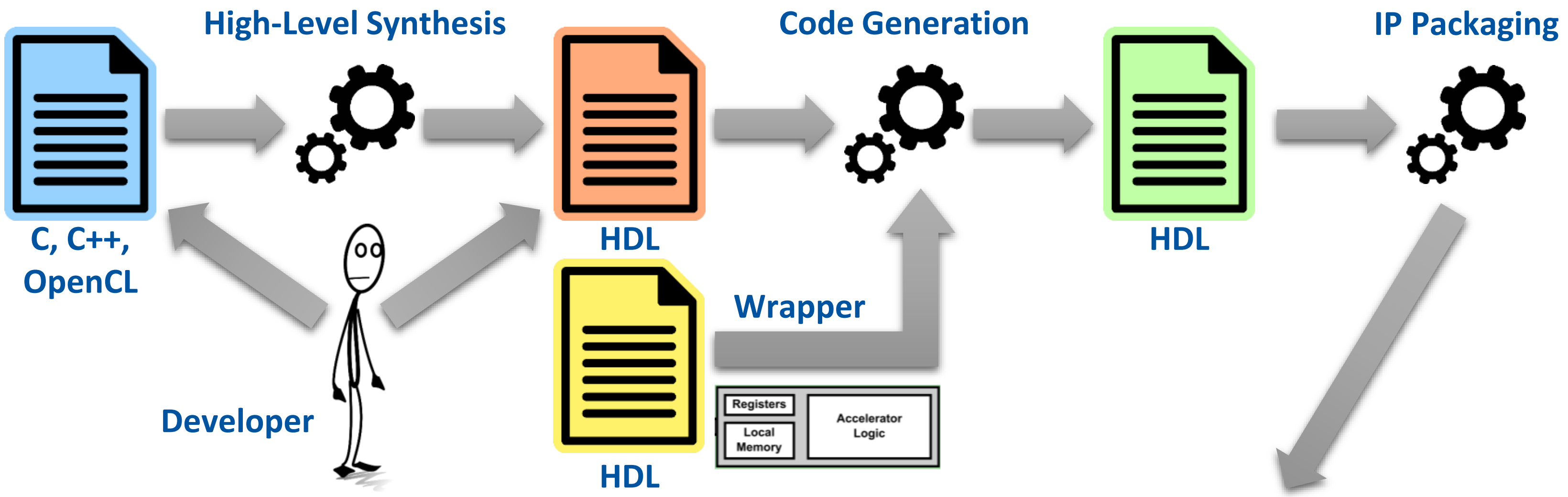
Reduction — Urgent

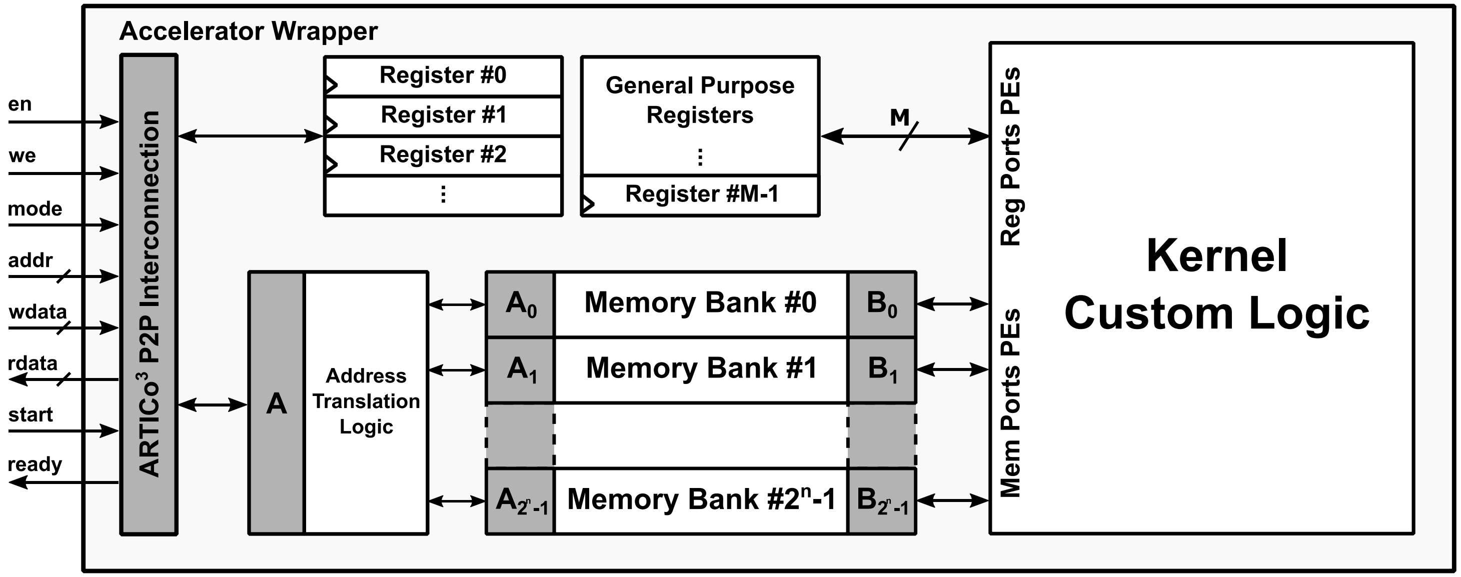




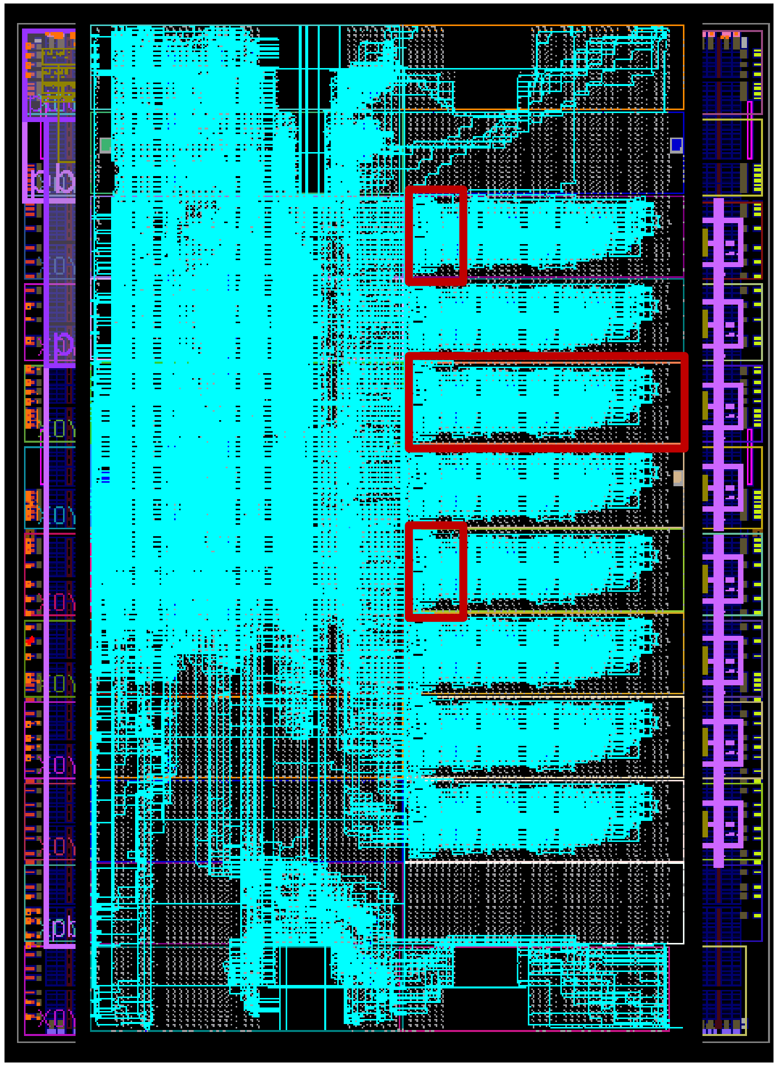


ARTICo3-Compliant Accelerator Design





DPR-Compatible Floorplanning



Low-Level Constraints

Design Placement

Design Routing

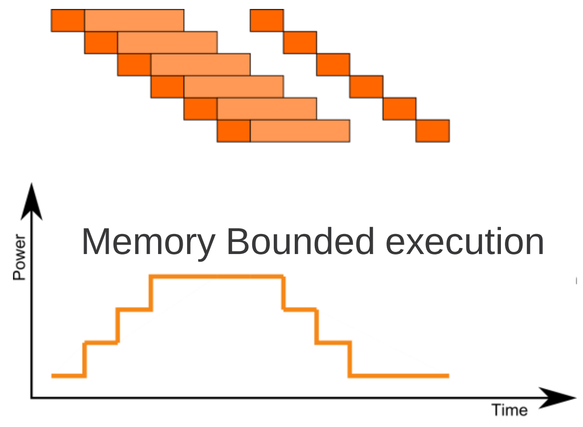
Technology Dependencies

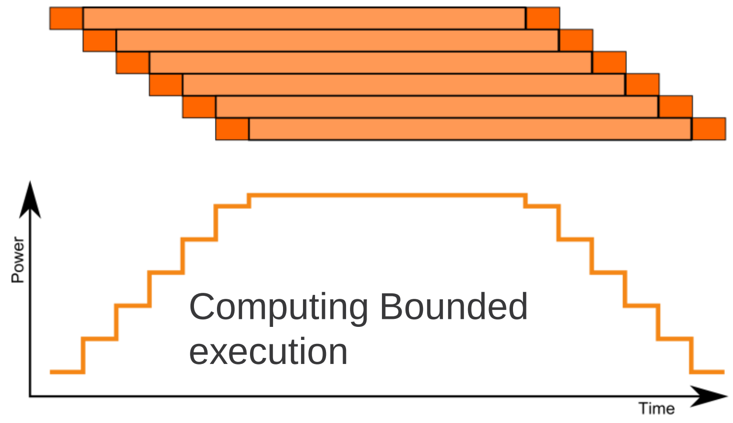
"Homogeneous" Fabric Layout

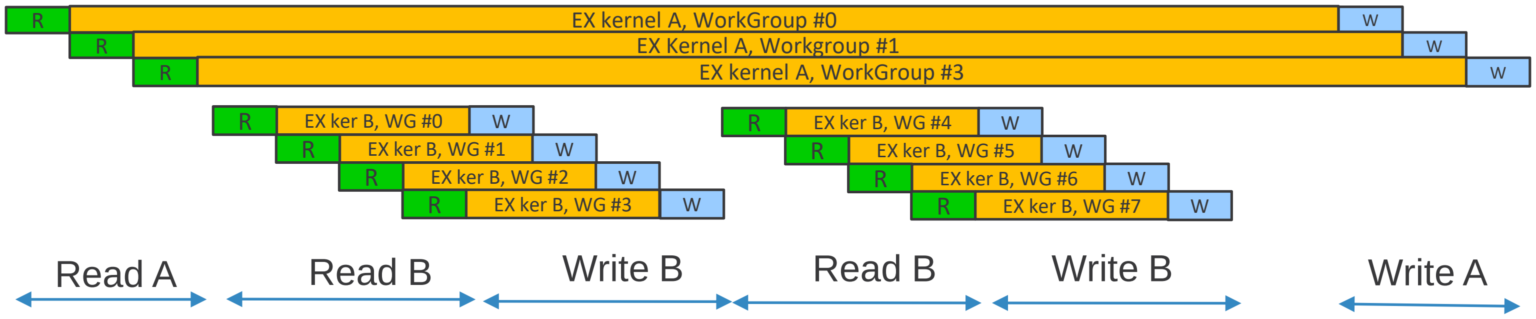
Common Interfaces

Hardware "Copy & Paste"

Execution and energy modelling







Example execution of two concurrent kernels. Kernel A is a long computing bounded execution run of 3 different Workgroups, while B uses memory access 'dead time' to execute two bunches of 4 Workgroups each for kernel B.
Memory accesses optimised: only 6 transactions, máximum memory utilisation

Dynamic Solution Space Exploration

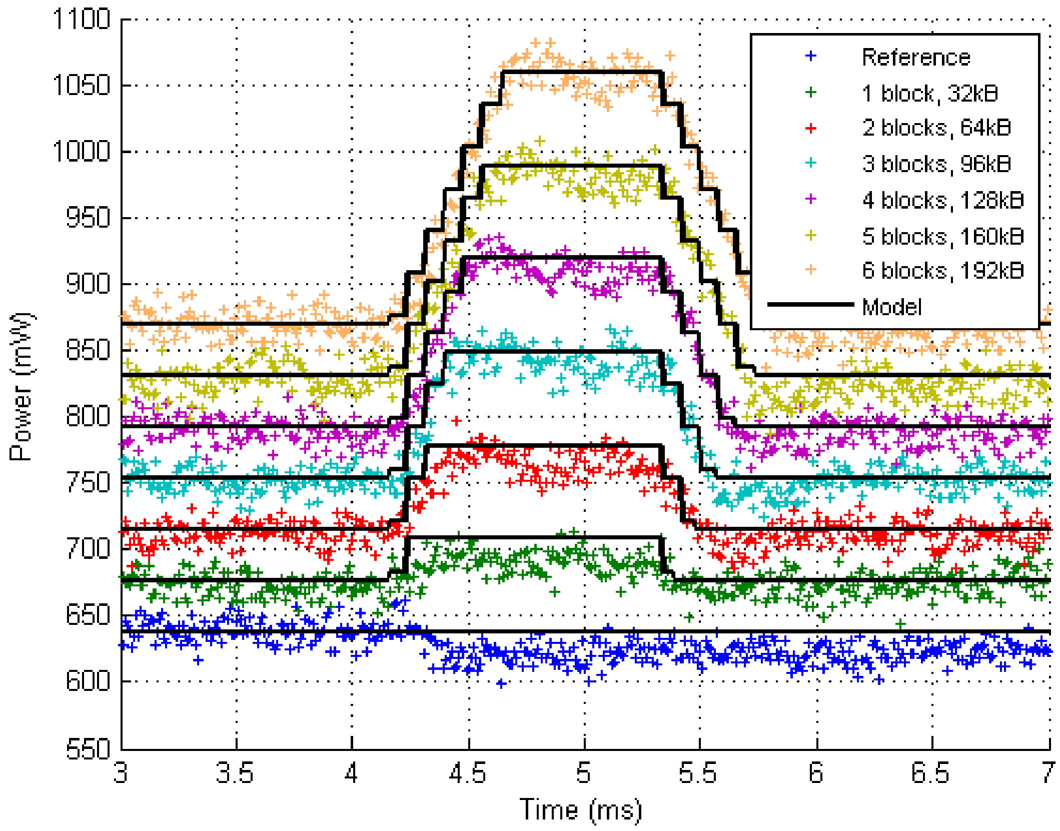
Embedded Models

+

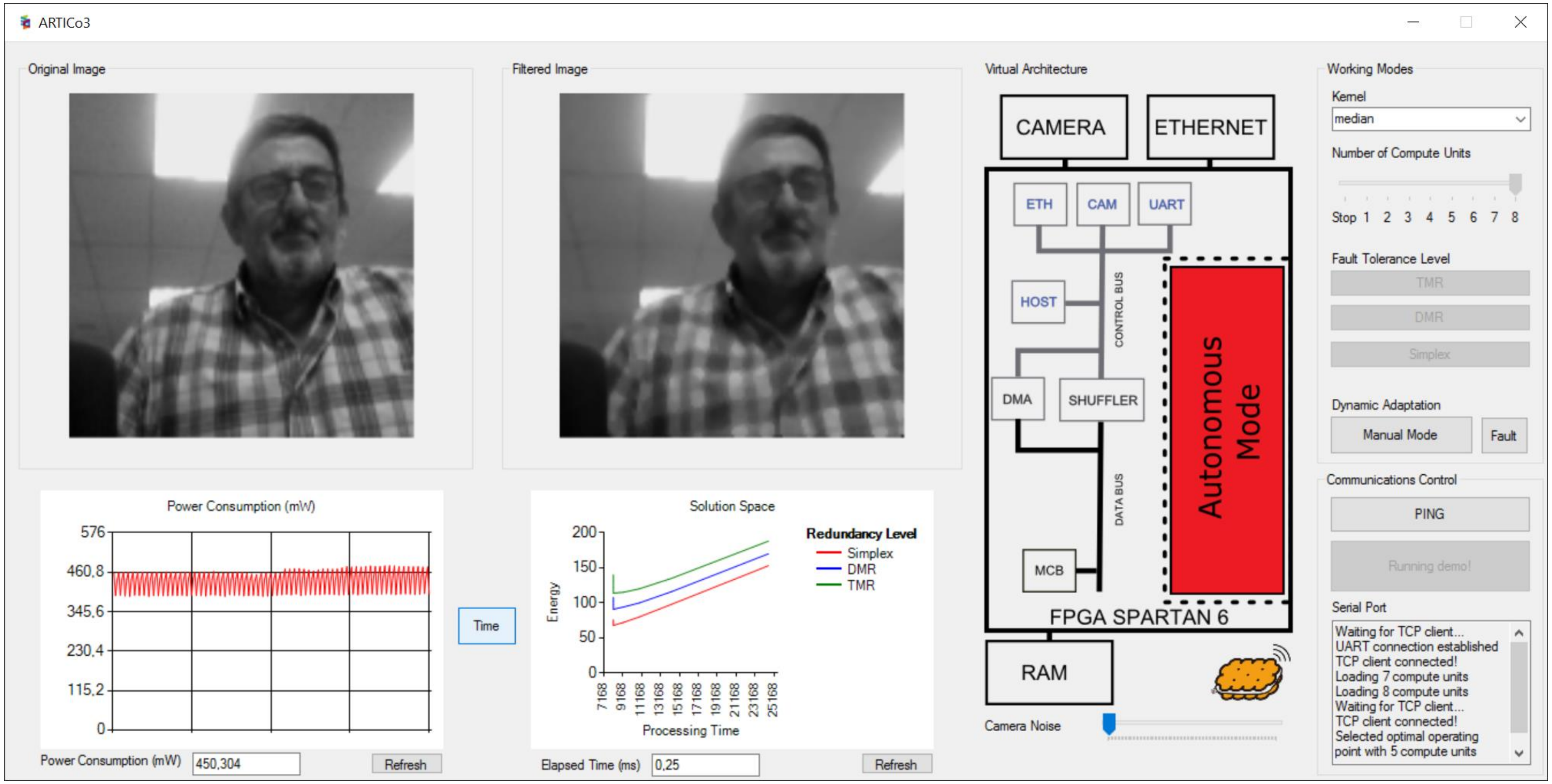
Runtime Monitoring & Profiling

=

Self-Aware Online Resource Management



Embedding the model for self-adaptation



Kernel: median

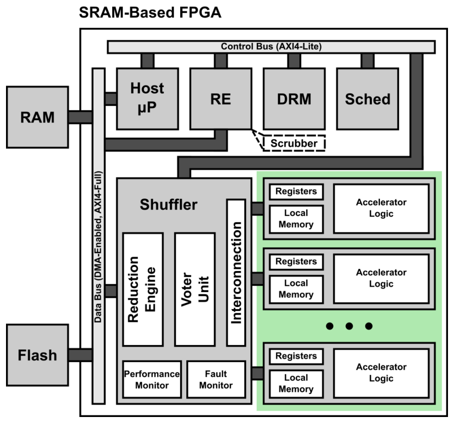
Power Consumption (mW): 450,304

Elapsed Time (ms): 0,25

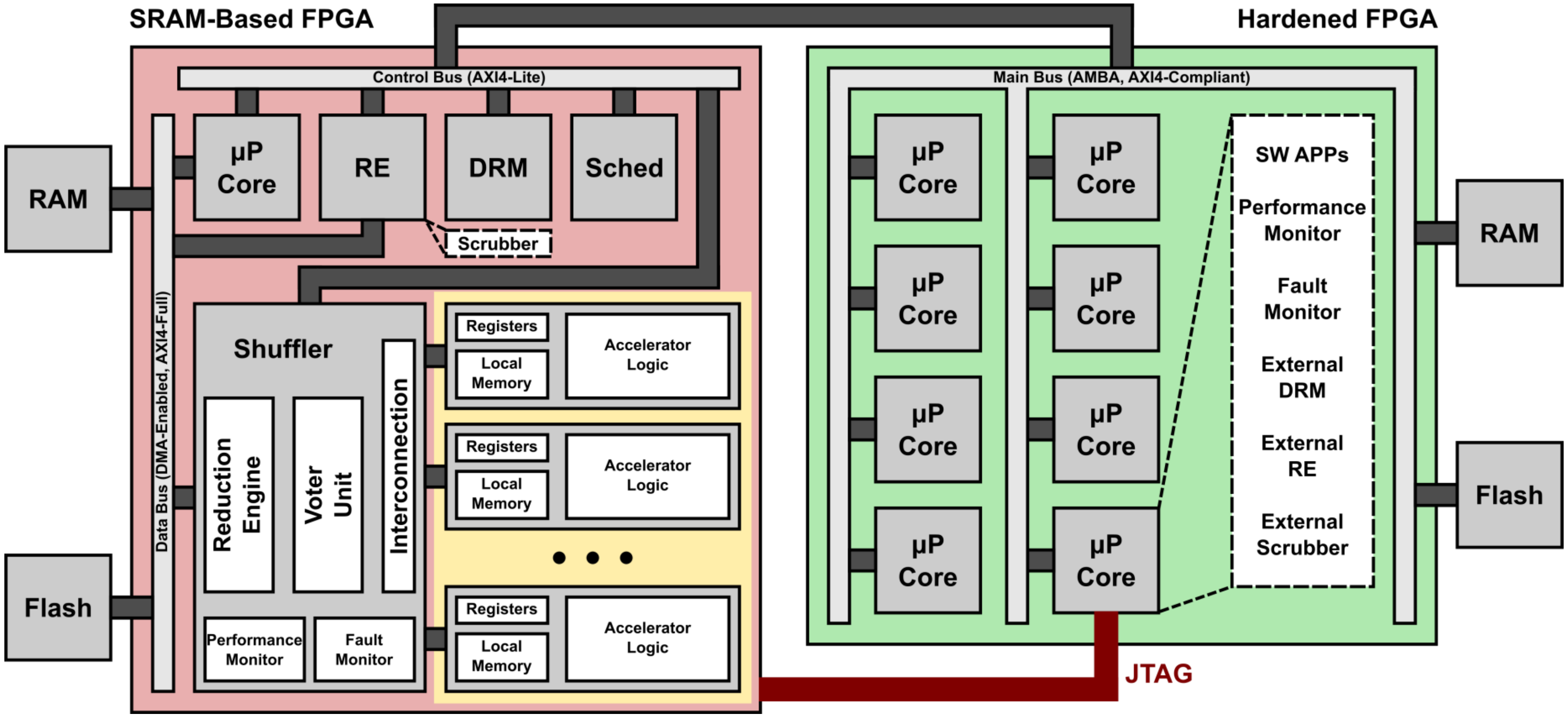
```
Waiting for TCP client...
UART connection established
TCP client connected!
Loading 7 compute units
Loading 8 compute units
Waiting for TCP client...
TCP client connected!
Selected optimal operating
point with 5 compute units
```

What's next?

- ReconOS + ARTICo^3 integration
  - Different programming models
    - HW/SW multithreading (ReconOS)
    - OpenCL-like coprocessing (ARTICo^3)
  - Common needs
    - Simultaneous multitasking support
    - SoA DPR support
  - Integration challenges
    - Is it possible/feasible to merge both programming models?
    - How can both frameworks benefit from multitasking capabilities (i.e. the reconfigurable fabric is used by several applications at the same time)?
    - Does it make sense to provide last ReconOS version with DPR capabilities using ARTICo^3 as part of the underlying hardware infrastructure?
    - Does it make sense to have an OpenCL-like coprocessing engine in ReconOS-based systems?



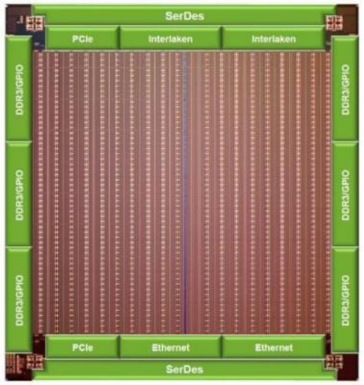
Next: Hardening for extending the applicability



Design diversity: 3 reconfiguration types: external (JTAG, other), internal (PCAP & ICAP)
2 scrubbers , external and internal (slow and fast loops)
HW & SW tasks, all relocatable → Heterogeneity to survive
Graceful degradation

Further into HPC

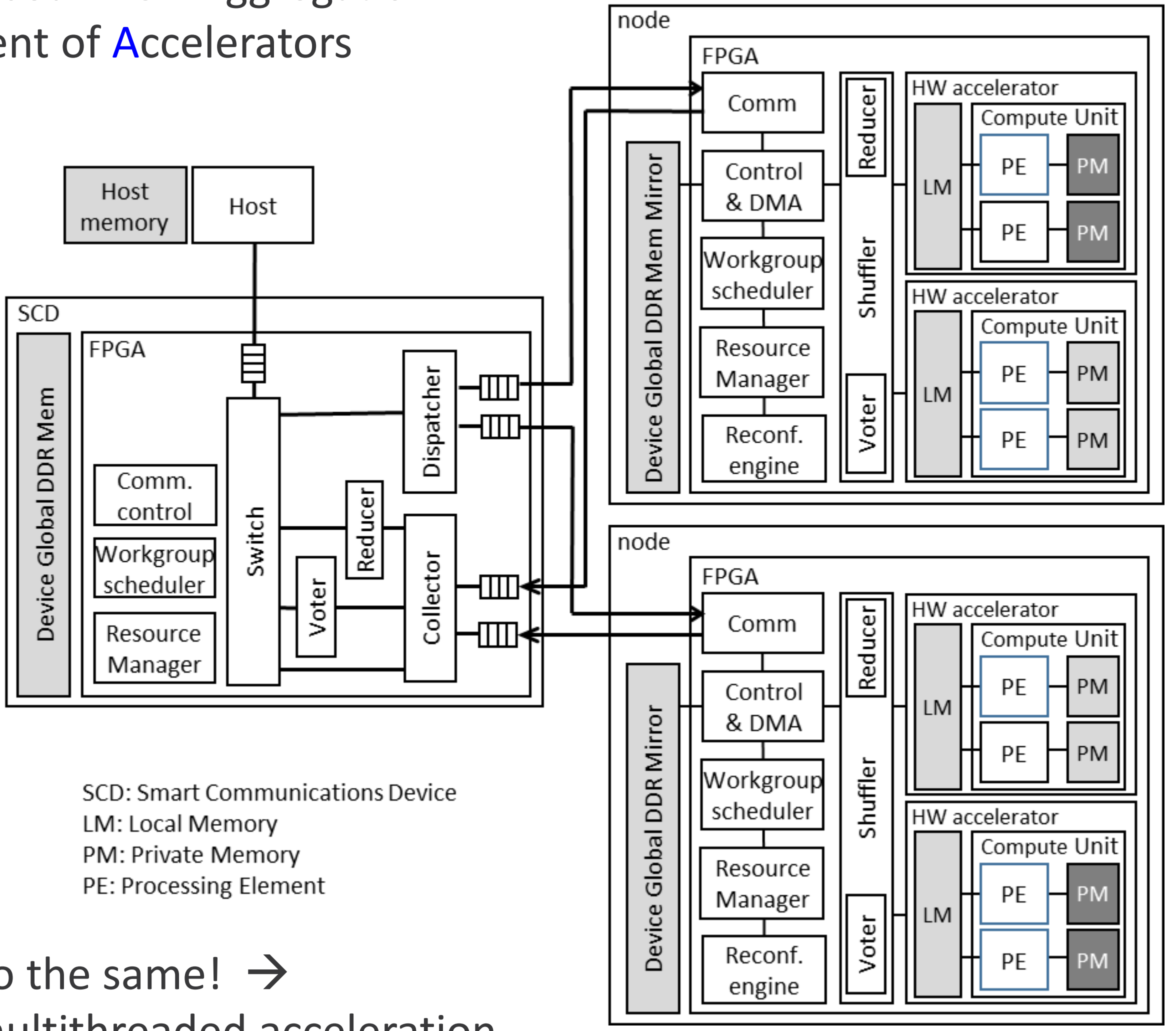
Intel and other chip makers are increasingly relying on accelerators to help improve the performance and energy efficiency of their processors and speed up the workloads that run on them. Nvidia and Advanced Micro Devices offer GPU accelerators. However, the company also is now using FPGAs, which can be reprogrammed through software after they've been manufactured. They're becoming more important for cloud and Web-scale environments, where workloads can change quickly. (Source: Jeffrey Burtenews, April 2016)



Achronix Introduces the Highest FPGA Memory Bandwidth, PCIe Acceleration Board for Data Center Applications (Source: Achronix, 2016 June 21st)
(700,000 LUTs, 6 memory controllers 690 Gbps, PCIe v3x8, 4x 40G Ethernet, …)
… and outstanding regularity!!! → Perfect for DPR

A cluster of ARTICo^3 nodes: towards HPC

MAFALDA: Multiple Artico3-based FPGA Aggregation for Large Deployment of Accelerators



SCD: Smart Communications Device
LM: Local Memory
PM: Private Memory
PE: Processing Element

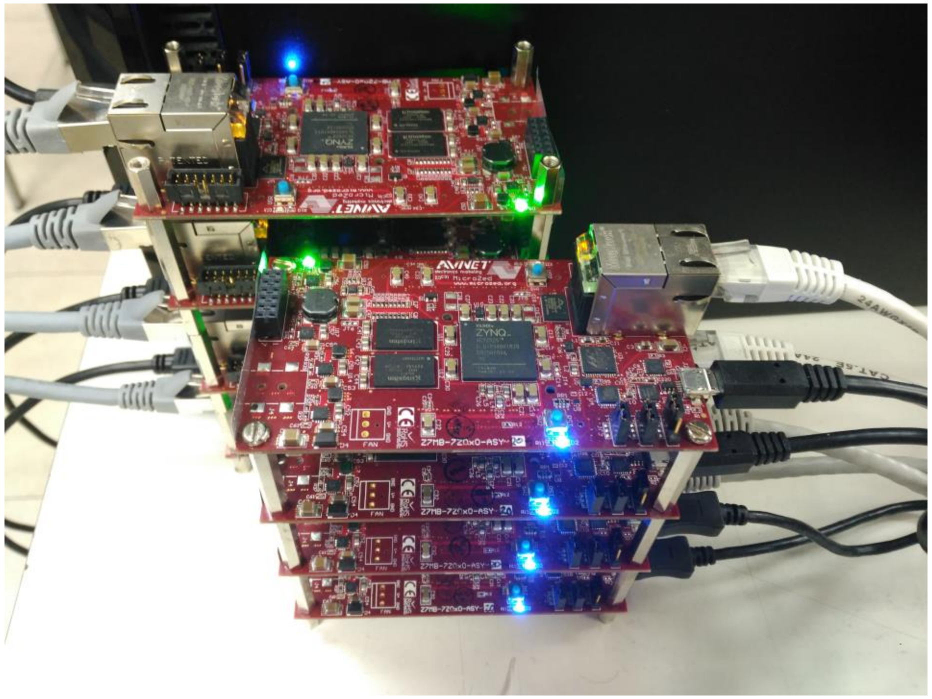
Smart comm. Device (SCD): a switch that mcasts, votes, reduces, etc…
Same as Artico3

Several reconf. engines in parallel!

MPI collective functions for distributed processing also do the same! → Distributed heterogeneous multithreaded acceleration

Multi-FPGA acceleration

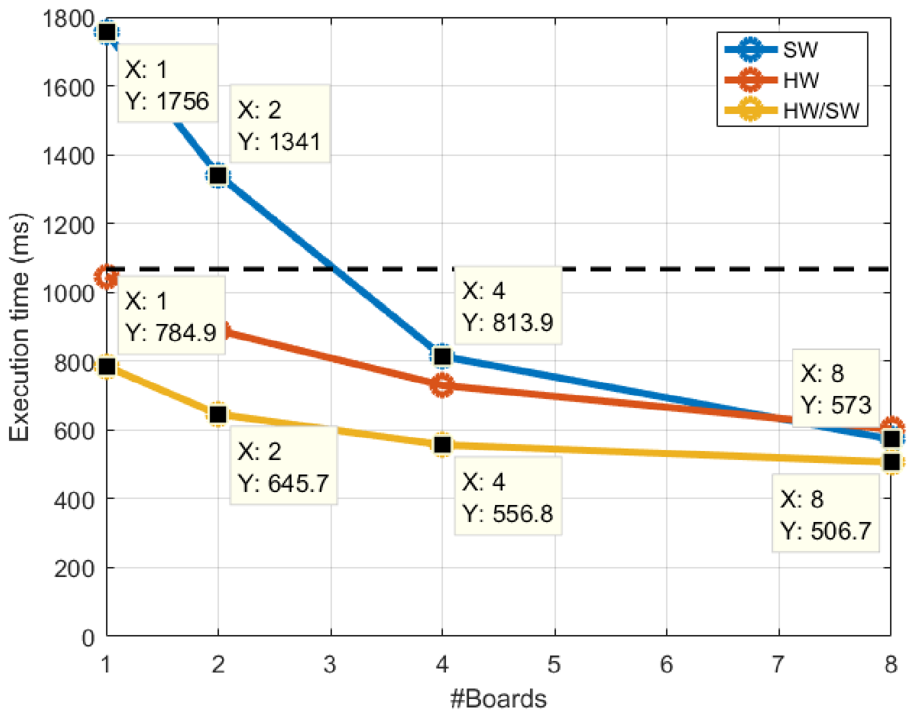
- Small FPGA-based computing cluster
  - ARTICo^3 infrastructure per node
  - Resilient MPI-based communication
- Work in progress
  - HLS-based standalone accelerators, need to use ARTICo^3-based ones
  - MPI-based communications, need to use fault-tolerant alternatives

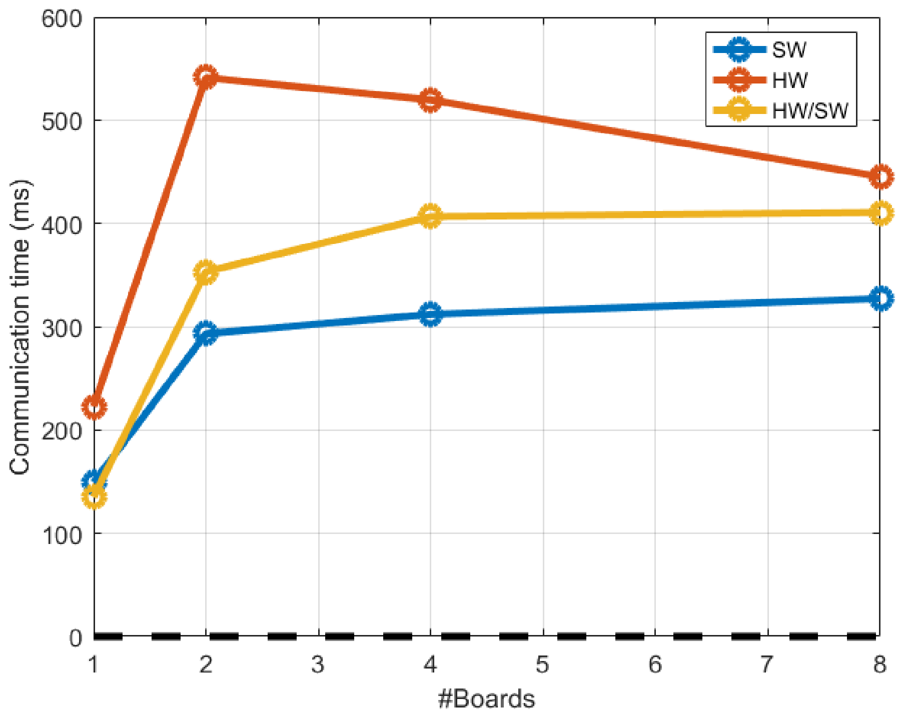


Results for multi-FPGA acceleration

Total Processing Time

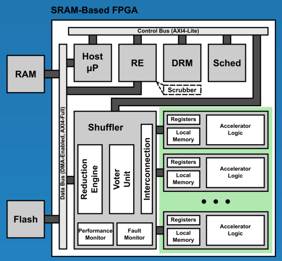
Communication Time





ARTICo3 Ecosystem

REBECCA



Basic ARTICo^3 Platform

- Basic Toolchain
- Execution Modeling
- Multi-FPGA Extension

- Resilience in Smart Cities

ENABLE-S3

- Hardened for Space
- Safe Reconfiguration
- Scrubbing Techniques
- Predictability

- Hyperspectral Image Compression
- Autonomous Satellite Navigation

CERBERO

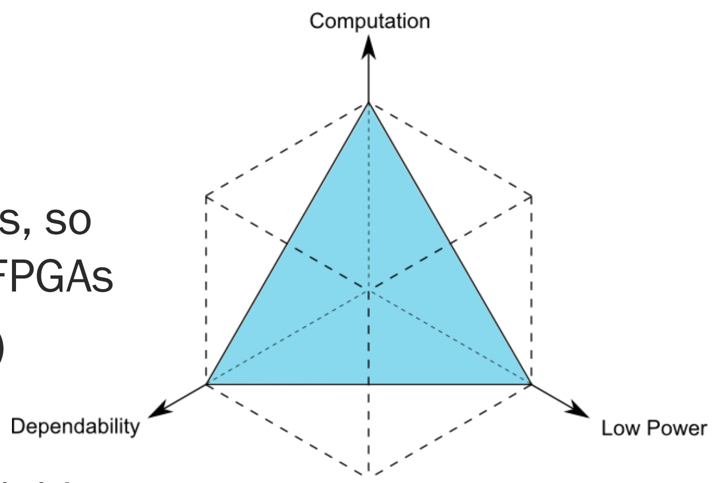
- Dataflow Extension
- Complete Toolset
- Full Adaptation
- CPSs & CPSoSs

- Robotic Arm for Space Rover

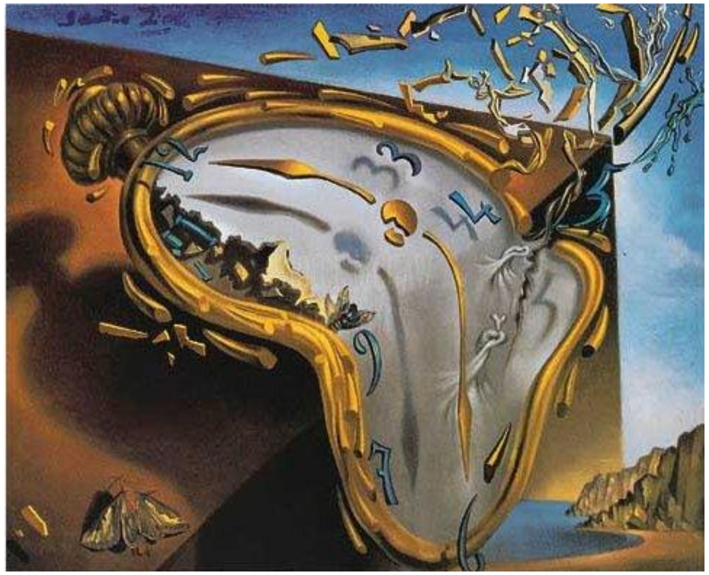
What we have seen so far:

- Adaptable platform
  - Trading-off computing performance, energy utilization and fault tolerance
  - Compatible with multithread execution models, so same code is OK for multicore, GPGPUs and FPGAs
  - Scalable to HPC by congruent clustering (WIP)
- Some Self-* features
  - Self adaption triggered by lightweight embeddable execution models
  - Self-awareness (conscious of energy and faults)
  - Self-protecting by fault tolerance and task movement

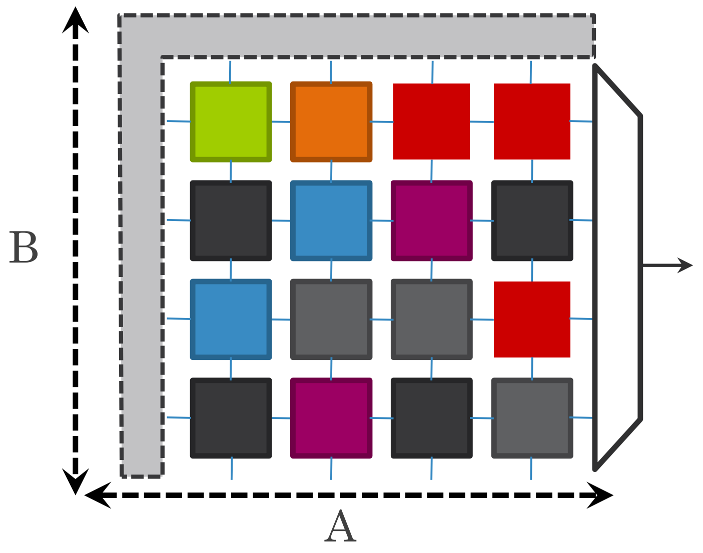
- Going up towards more levels of autonomy
  - Evolvable HW



EVOLVABLE HW: TOWARDS SELF-*



Computing Engine – Architecture



Adaptable Computing Template

Unknown Processing Behaviours

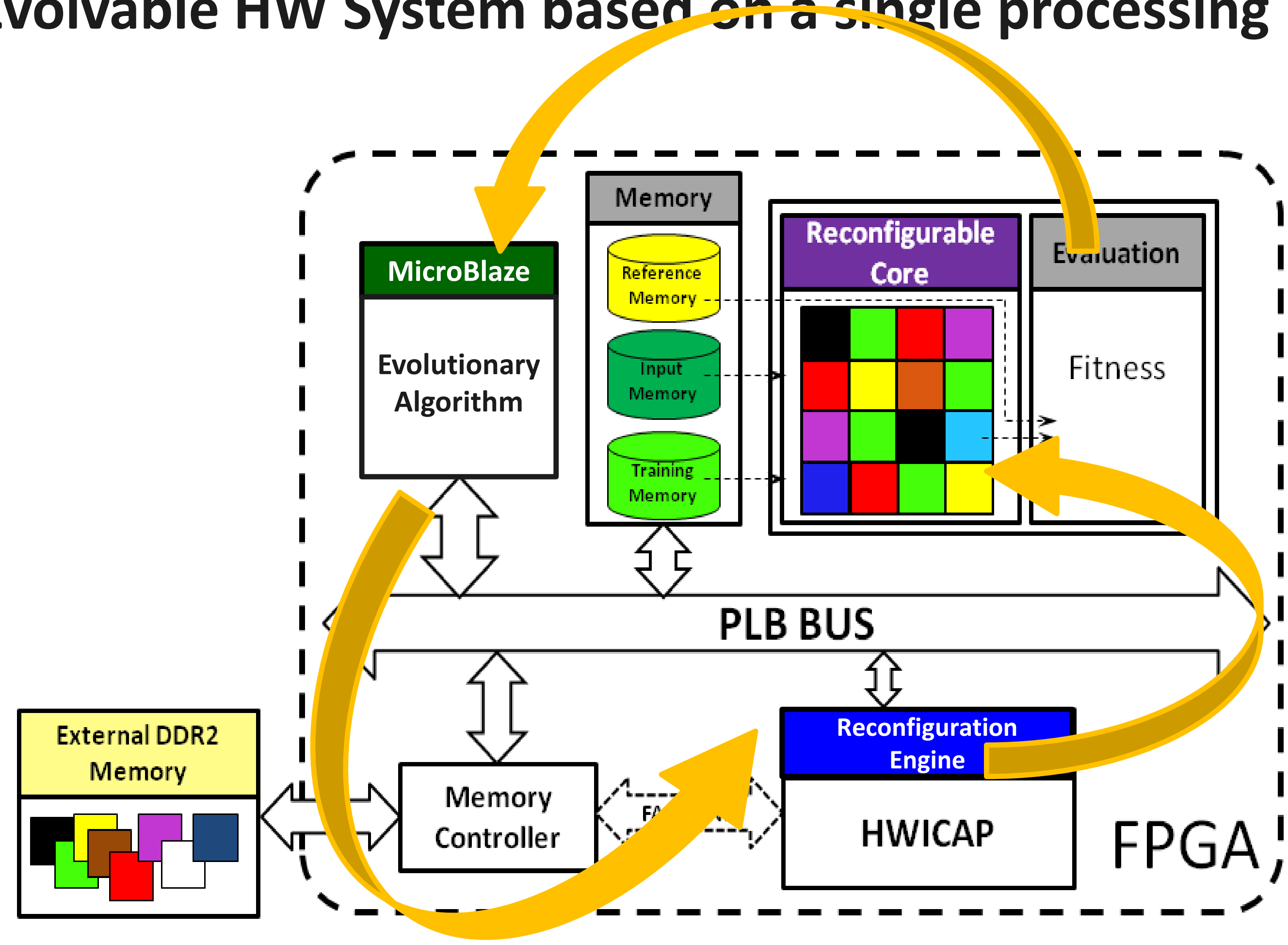
Runtime Reconfigurable

2D Systolic Array of Reconfigurable PEs

Parallelism	High throughput
Locality	Reduced routing overhead
Regularity	Reduced memory footprint
Modularity	Reduced reconfiguration time

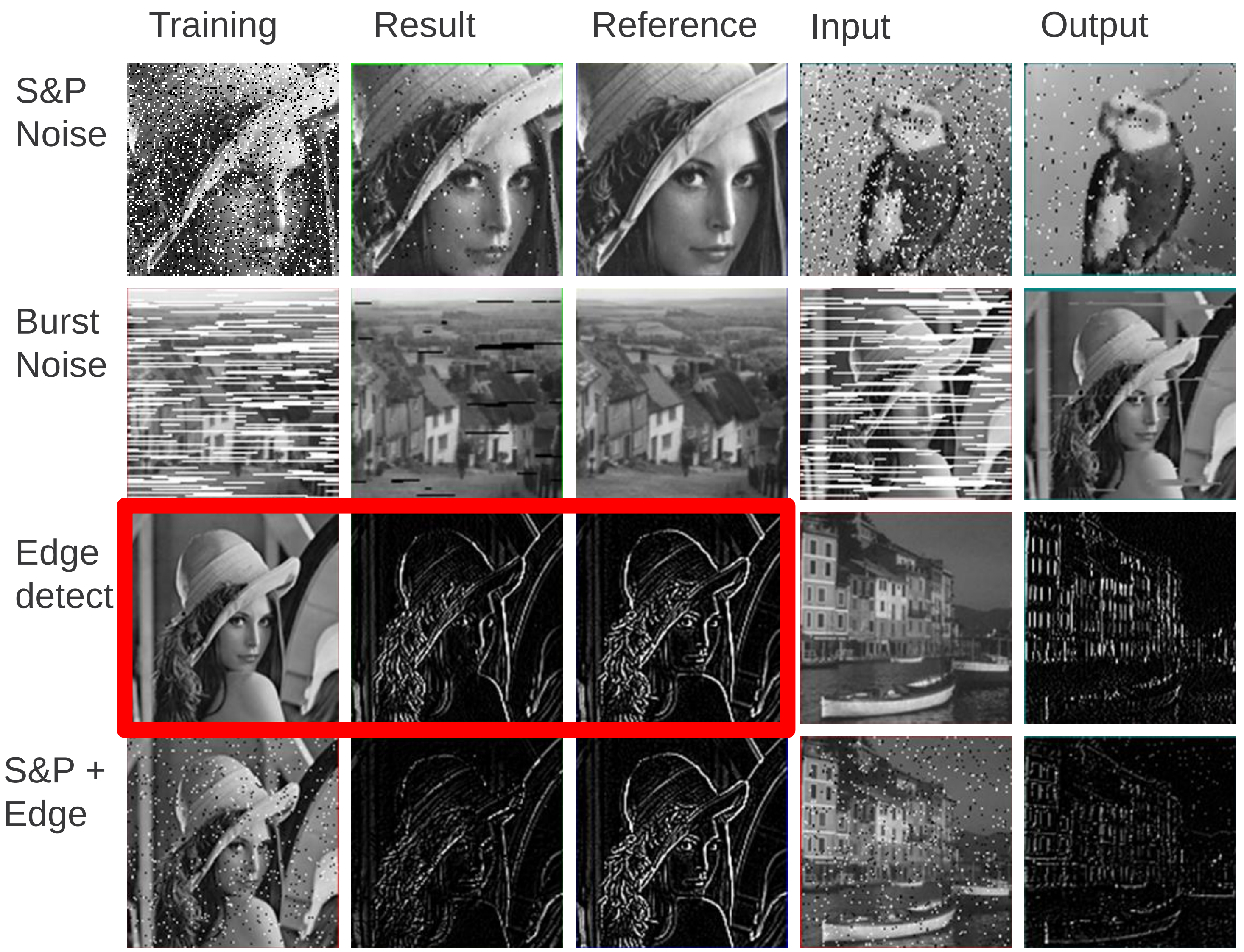
The SoPC architecture

An Evolvable HW System based on a single processing array



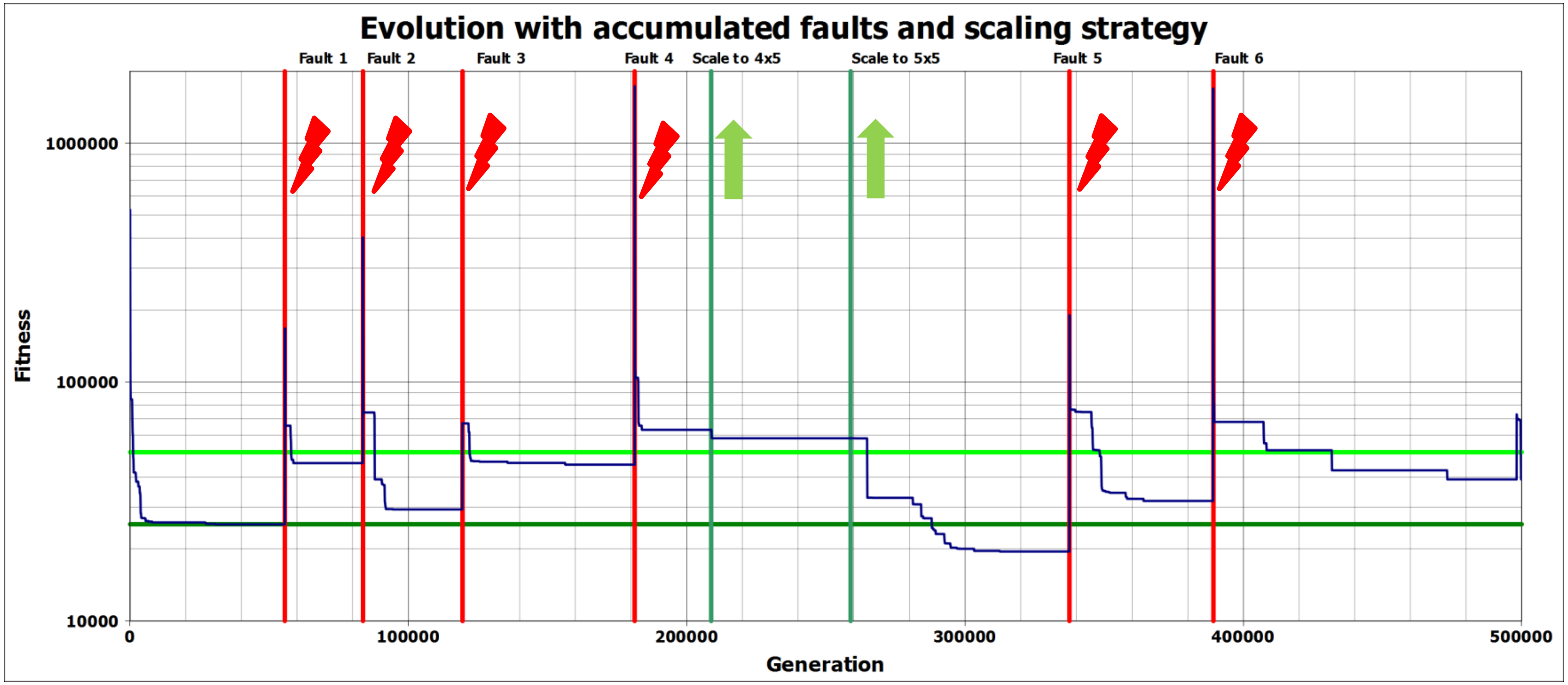
A. Otero, R. Salvador, J. Mora, E. de la Torre, T. Riesgo, L. Sekanina; "A fast Reconfigurable 2D HW core architecture on FPGAs for evolvable Self-Adaptive Systems", AHS 2011

System is adaptable and generalizable



Scalability and evolution for increased fault tolerance

Improved fault tolerance provided by intrinsic evolution



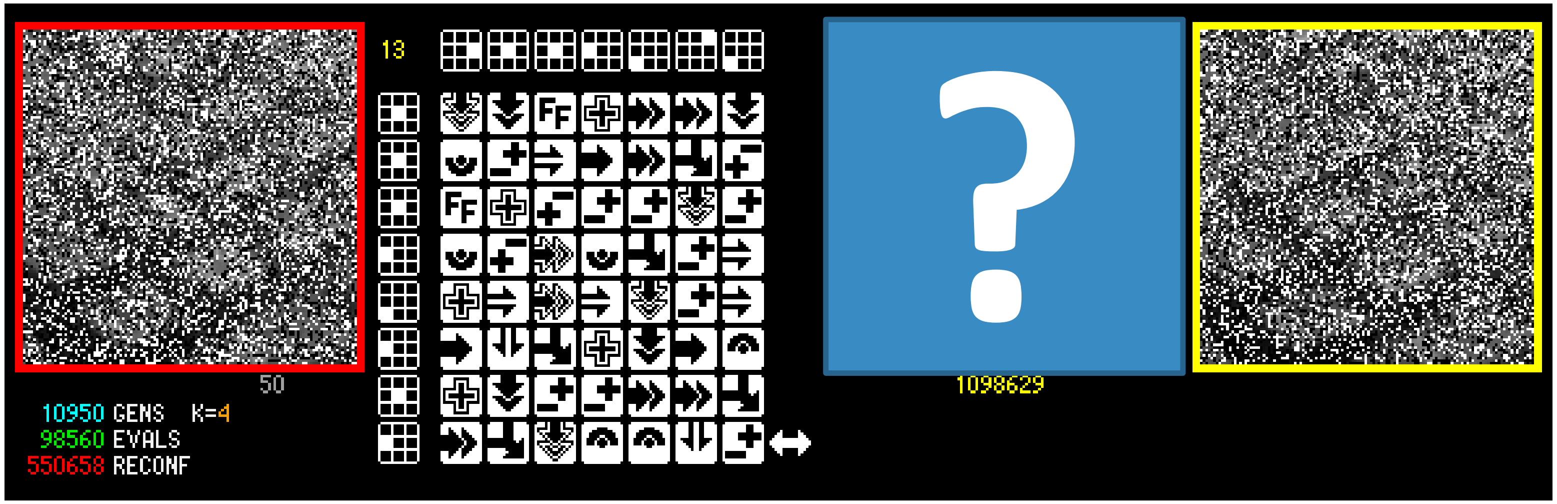
Example of evolution with accumulated faults (threshold at 2x initial fitness)

A 4x4 recovers from 2 faults in average

A 7x7 recovers from 12 faults in average

Lifetime of the system extended 6 times

Results of a large array with very noisy reference

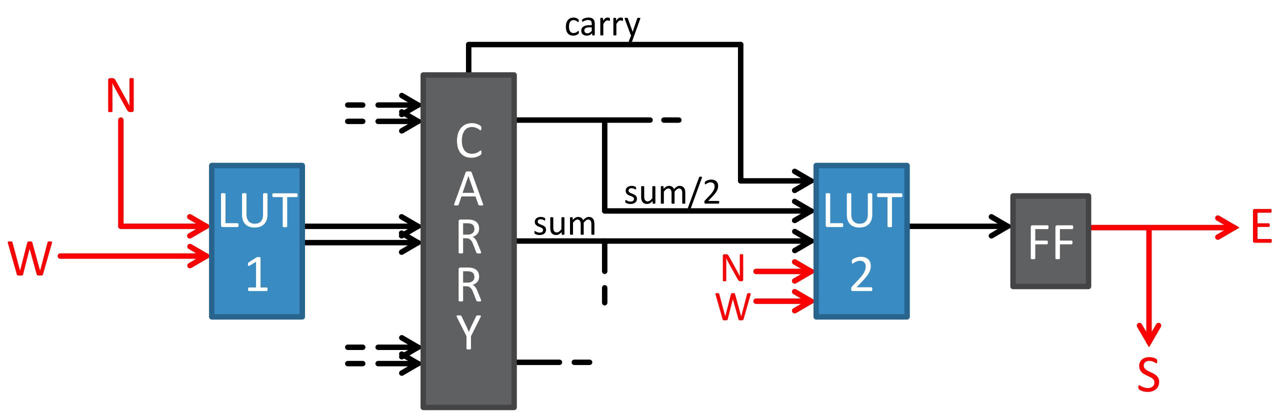


With these two inputs …. We get this output

Noise-agnostic filter with autonomous self-healing physically-sclable scalable evolvable hardware

Processing array improved results

LUT-based processing elements



- Small: 2 CLBs per PE
- Fast: 450 MHz (450 Mpixels/sec) ← Because of locality!!
- Same functions / Possible new functions
- Only LUTs change → No routing restrictions
- Less to reconfigure: From 36 frames → 3 frames
- Evolution speed: 135000 evaluations/second (12 arrays on Virtex5)

Conclusions

- The world is becoming complex, and systems might need to be autonomous and adapt to changes w/o external intervention
- Full autonomy is obtained as a contribution of other interesting characteristics
  - DPR, scalability, self-awareness, self-healing, …
- Performance goes together with energy efficiency and dependability
- FPGAs and DPR are key players, in embedded and HPC
- There are many things left to be done → research oportunities

Acknowledgements

- **Many things shown here today would not have been possible without the help of**
  - Teresa Riesgo, Jorge Portilla and Yago Torroja
  - Ana Belén Jimeno, Yana Krasteva, Andrés Otero, Wei He, Rubén Salvador, Javier Mora, Juan Valverde, Filip Veljkovic, Alfonso Rodriguez, Leonardo Suriano
  - Ángel Gallego, Blanca López, Julio Camarero, Santiago Muñoz, César Castañares, Carlos Giménez, Javi Vázquez, Gabriela Cabrera
  - Miguel de las Heras, Ángel Morales, Manu Llinás, Alex Fernández, Miguel López, Pablo Sánchez, Carlos Pizarro, Verónica Díaz, Clara Casas, Miguel Baquero, Antonio Niño,, Javier Polop, Diego Lanza, Victor López, Ramón Conejo, Carlos Correa, Pablo Iglesias, David Gozalo, Alberto Ortiz

  … **thank you all for being a great (if not the best) team**
- **We also acknowledge the support from institutions**
  - European Union, Ministerio de Ec. y competitividad, Min. de Industria, Comunidad de Madrid
- **Many thanks for the collaboration with past, present and future partners, and all those who shared the same illusions (and beers)**

… "Become wáter, my friend"

"You must be shapeless, formless, like water. When you pour water in a bottle, it becomes the bottle. When you pour water in a teapot, it becomes the teapot.
Water can drip and it can crash.
Become like water my friend."

Become like water, my hardware.
Be shapeless, formless, like water