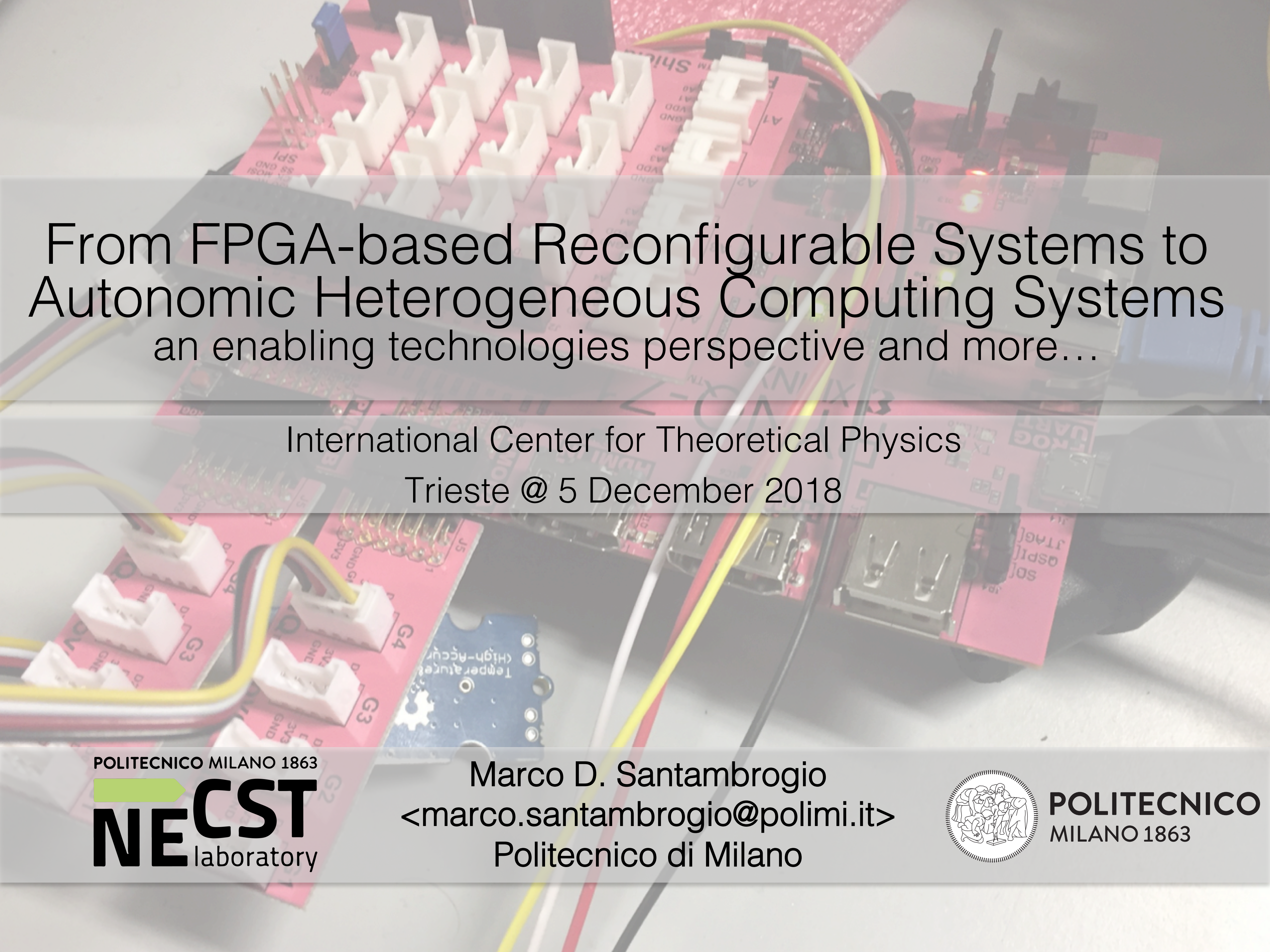


From FPGA-based Reconfigurable Systems to Autonomic Heterogeneous Computing Systems

an enabling technologies perspective and more…

International Center for Theoretical Physics

Trieste @ 5 December 2018

POLITECNICO MILANO 1863
NECST laboratory

Marco D. Santambrogio
<marco.santambrogio@polimi.it>
Politecnico di Milano

POLITECNICO
MILANO 1863

Computing systems are getting…

Computing systems are getting…

little…

Computing systems are getting…

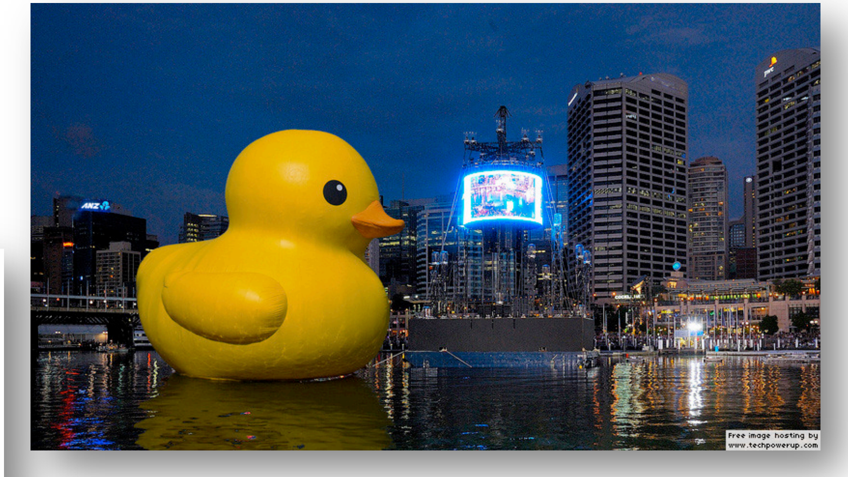
little…

little+Big

Computing systems are getting…

little…

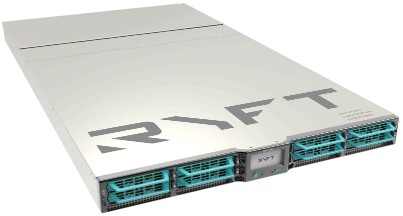
little+Big



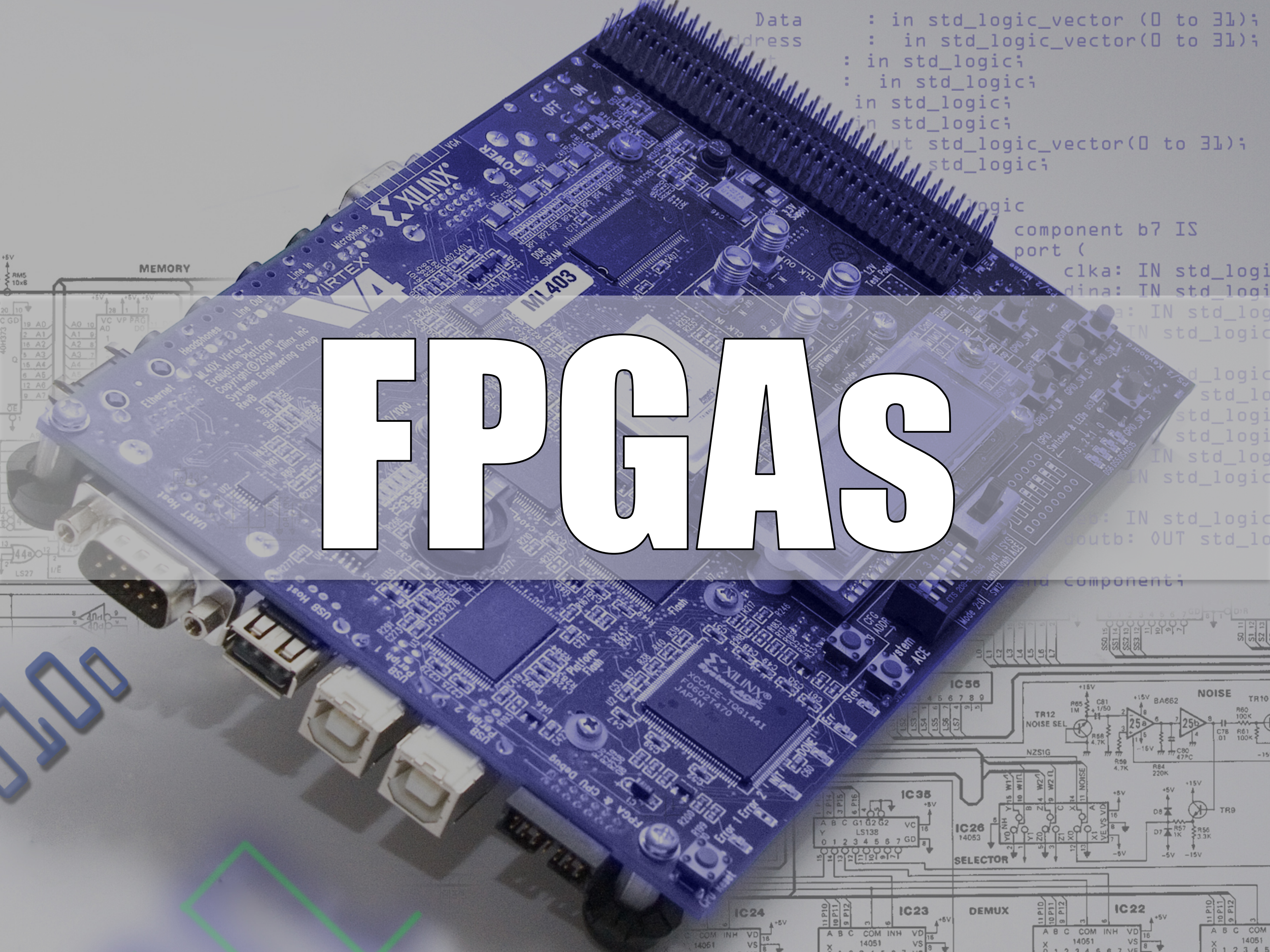
little+Big and heterogeneous

Heterogeneous Complex Systems

- Ryft ONE
  - Big Data infrastructure due to an FPGA-accellerated architecture
  - http://www.ryft.com/
- IBM Power8
  - Introducing the Coherent Accelerator Processor Interface (CAPI) port that is layered on top of PCI Express 3.0
  - http://www-304.ibm.com/webapp/set2/sas/f/capi/home.html
- Microsoft Catapult
  - Stratix V (Arria 10 FPGA)
  - http://research.microsoft.com/en-us/projects/catapult/
- Amazon EC2 F1 Instances
  - Xilinx UltraScale Plus FPGA
  - https://aws.amazon.com/about-aws/whats-new/2017/04/amazon-ec2-f1-instances-customizable-fpgas-for-hardware-acceleration-are-now-generally-available/
- OpenPower Foundation
  - http://openpowerfoundation.org/

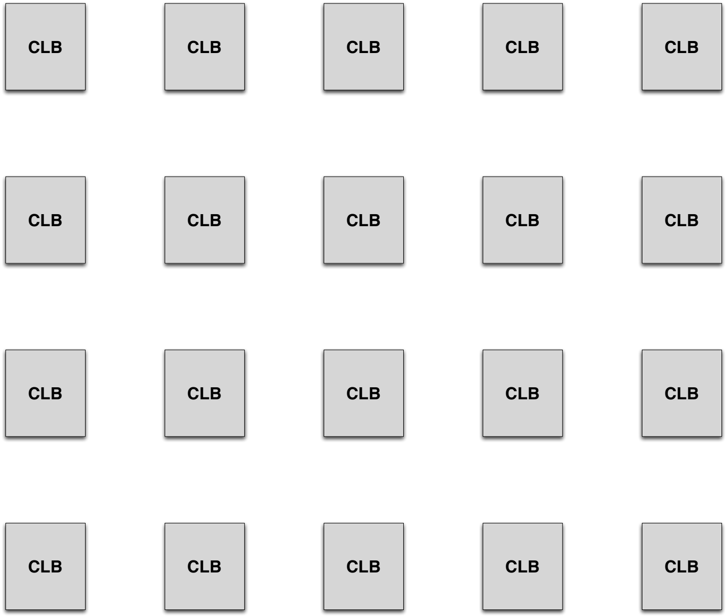


FPGAs


XILINX
VIRTEX-4
ML403
POWER
MEMORY
NOISE
SELECTOR
DEMUX
IC56
IC35
IC26
IC24
IC23
IC22
component b7 IS
port (
clka: IN std_logic
end component;

Field-Programmable Gate Arrays

- Configurable Logic Blocks
  - to implement combinational and sequential logic



CLB CLB CLB CLB CLB

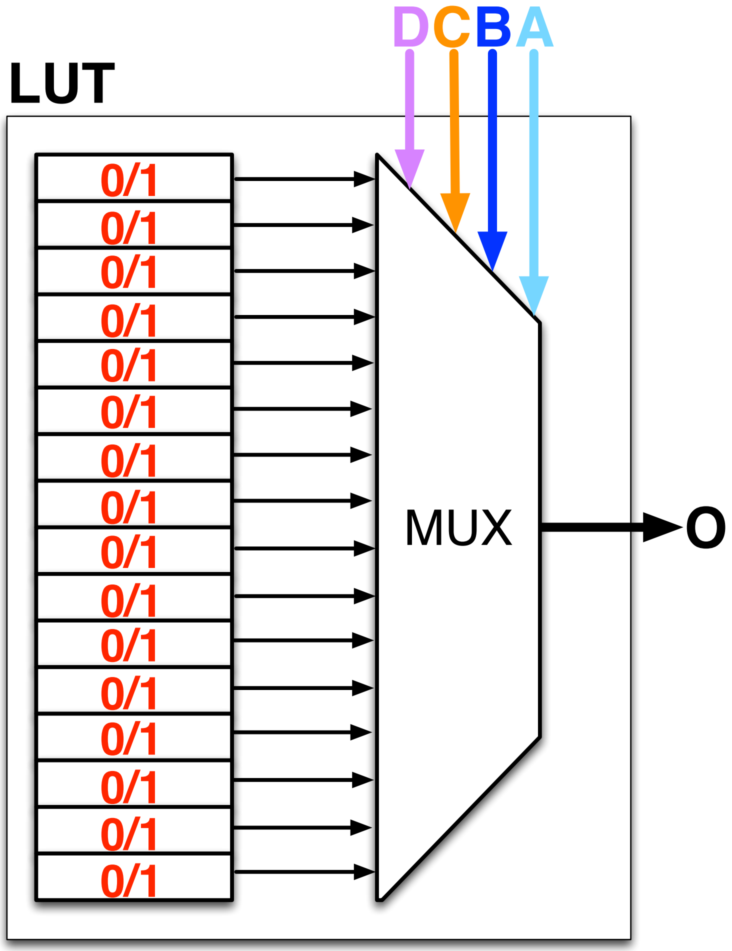
CLB CLB CLB CLB CLB

CLB CLB CLB CLB CLB

CLB CLB CLB CLB CLB

LookUp Tables: LUTs

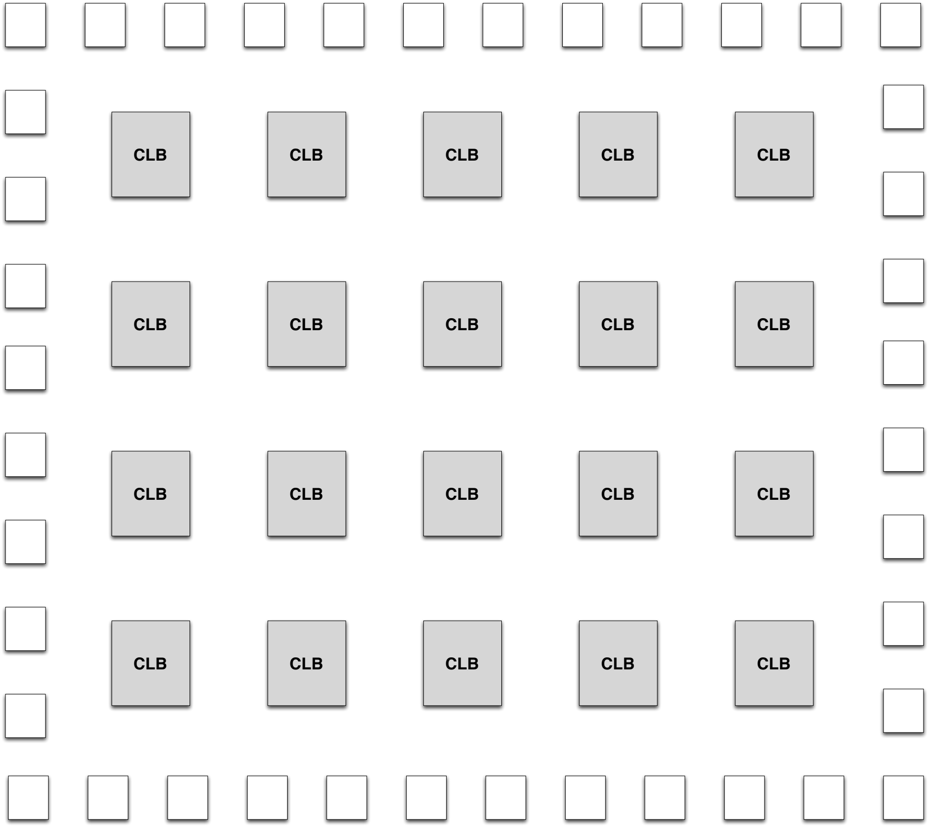
O = f(D, C, B, A)



- LUT contains Memory Cells to implement small logic functions
- Each cell holds ‘0’ or ‘1’ .
- Programmed with outputs of Truth Table
- Inputs select content of one of the cells as output

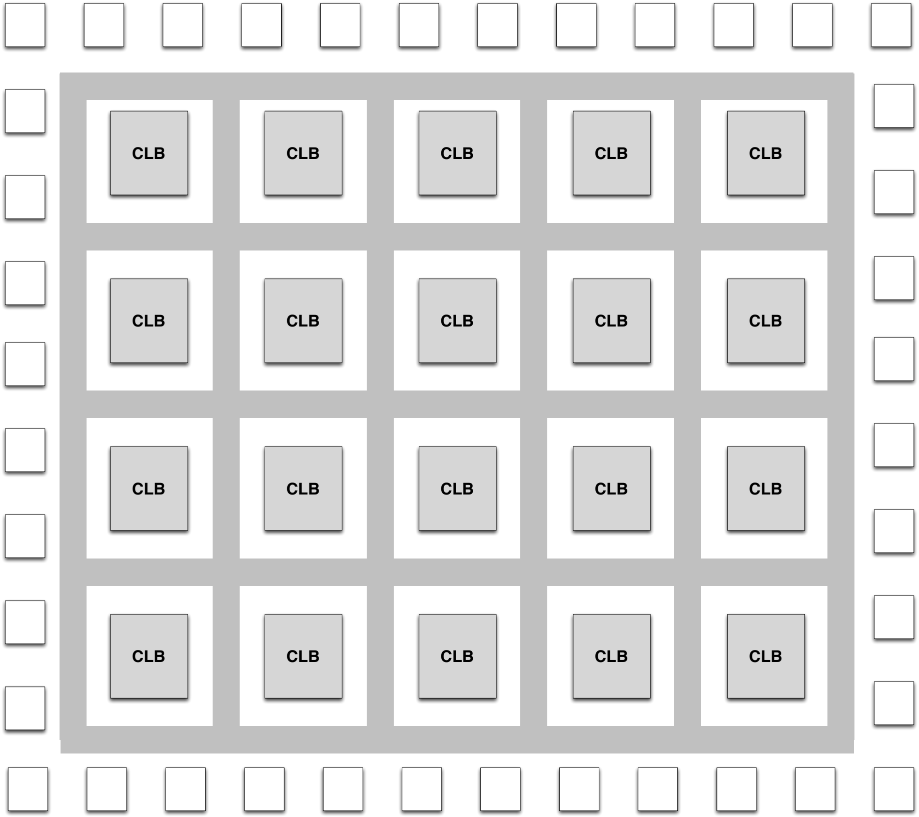
Field-Programmable Gate Arrays

- Configurable Logic Blocks
  - to implement combinational and sequential logic
- I/O blocks
  - special logic blocks at periphery of device for external connections



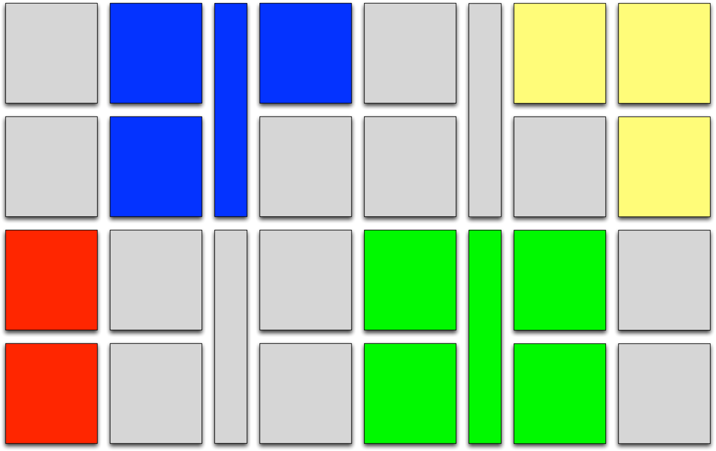
Field-Programmable Gate Arrays

- Configurable Logic Blocks
  - to implement combinational and sequential logic
- I/O blocks
  - special logic blocks at periphery of device for external connections
- Interconnections
  - wires to connect Inputs/ Outputs to configurable logic blocks



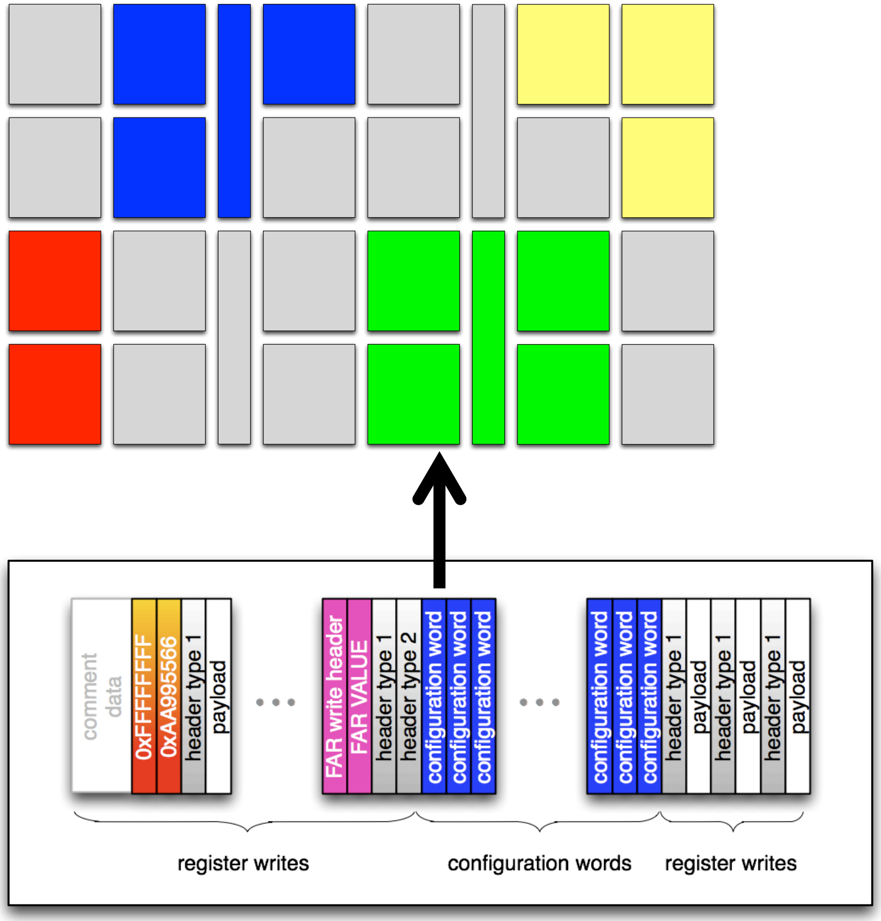
Xilinx FPGA and Configuration Memory

FPGA



Xilinx FPGA and Configuration Memory

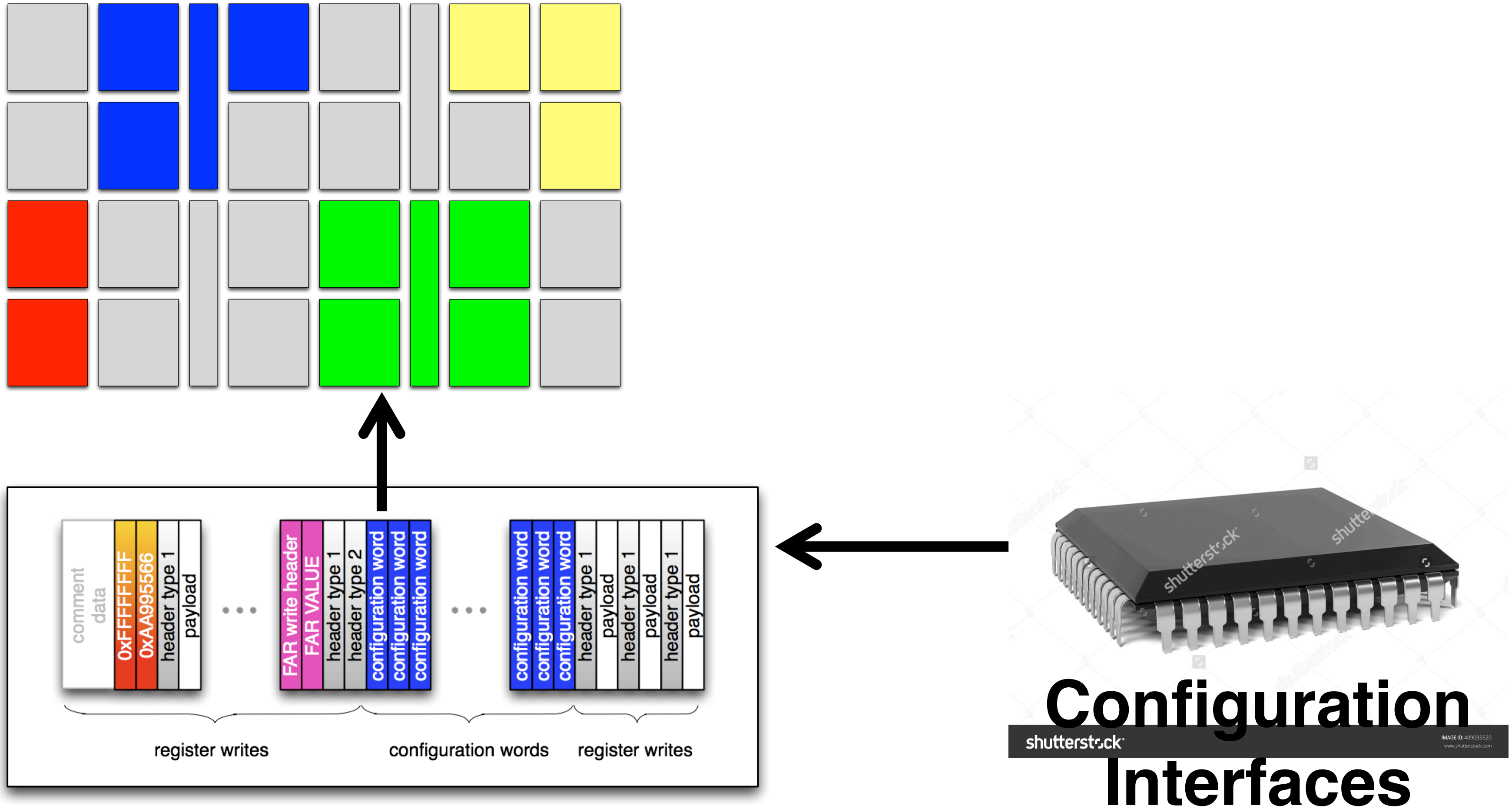
FPGA



Configuration Memory

Xilinx FPGA and Configuration Memory

FPGA

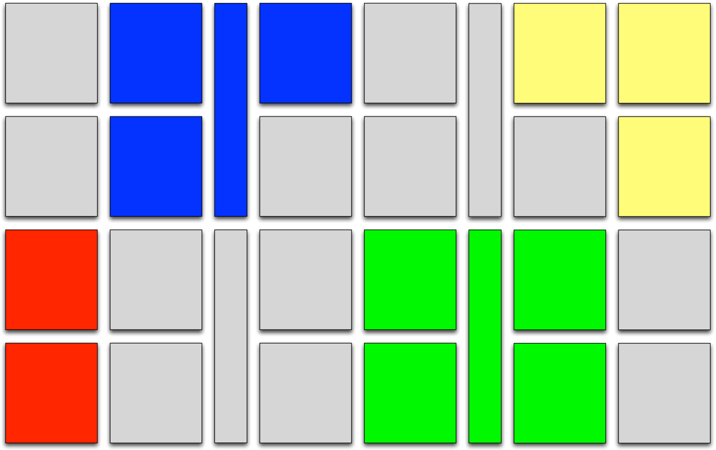


Configuration Interfaces

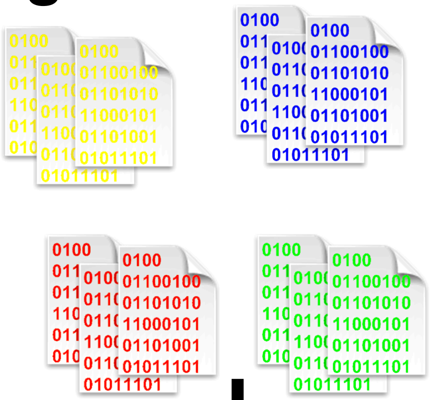
Configuration Memory

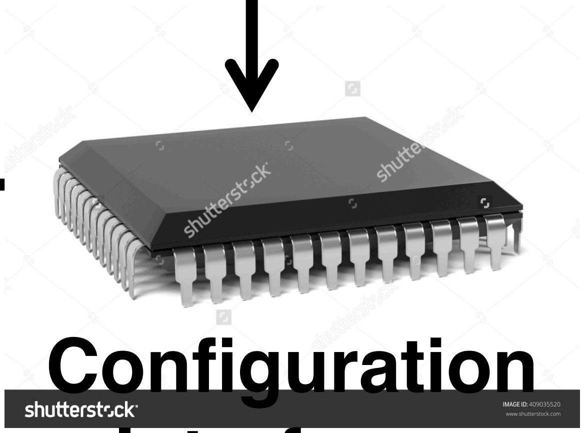
Xilinx FPGA and Configuration Memory

FPGA

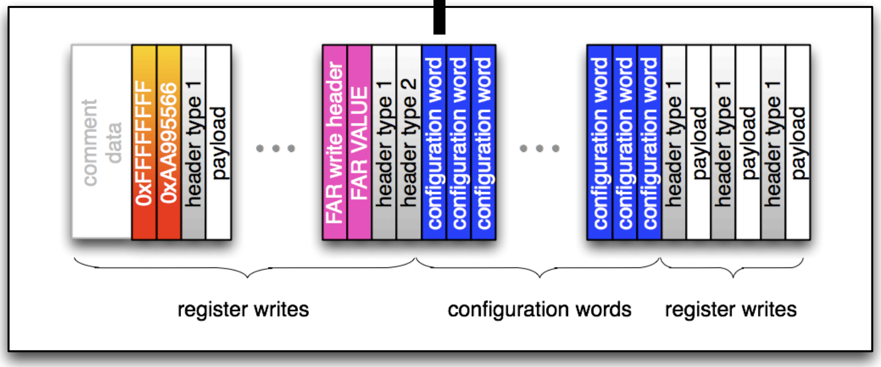


Configuration bitstream

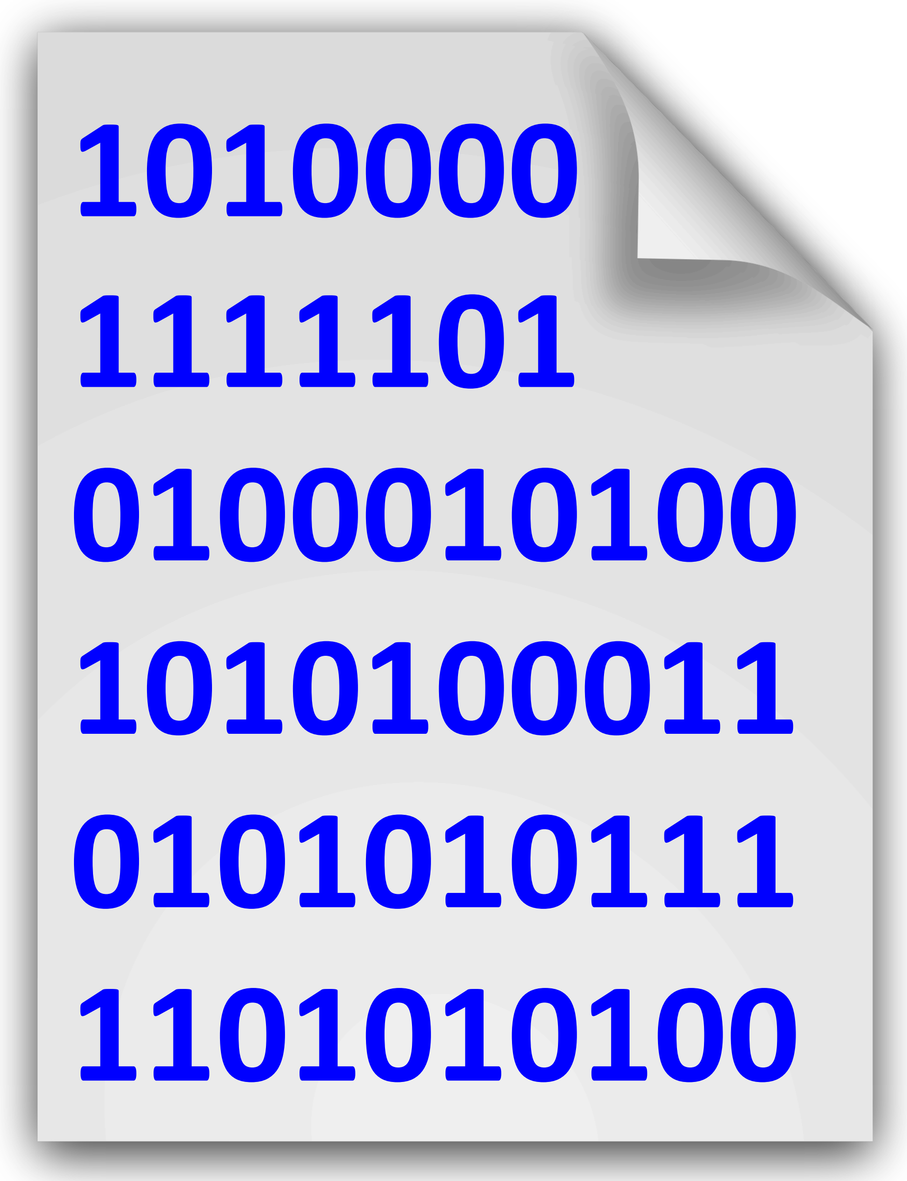




Configuration Interfaces



Configuration Memory



1010000

1111101

0100010100

1010100011

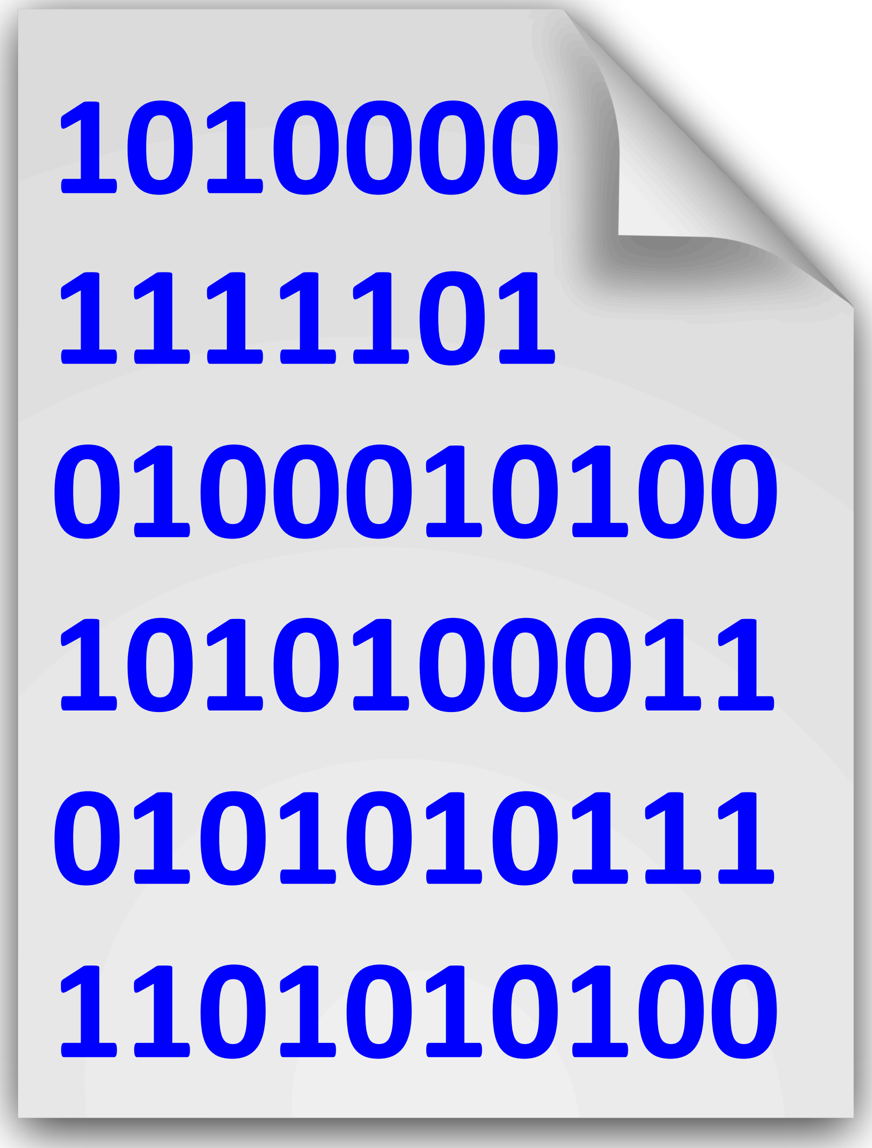
0101010111

1101010100

Reconfiguration in everyday life

Football

(Complete – Static)



1010000

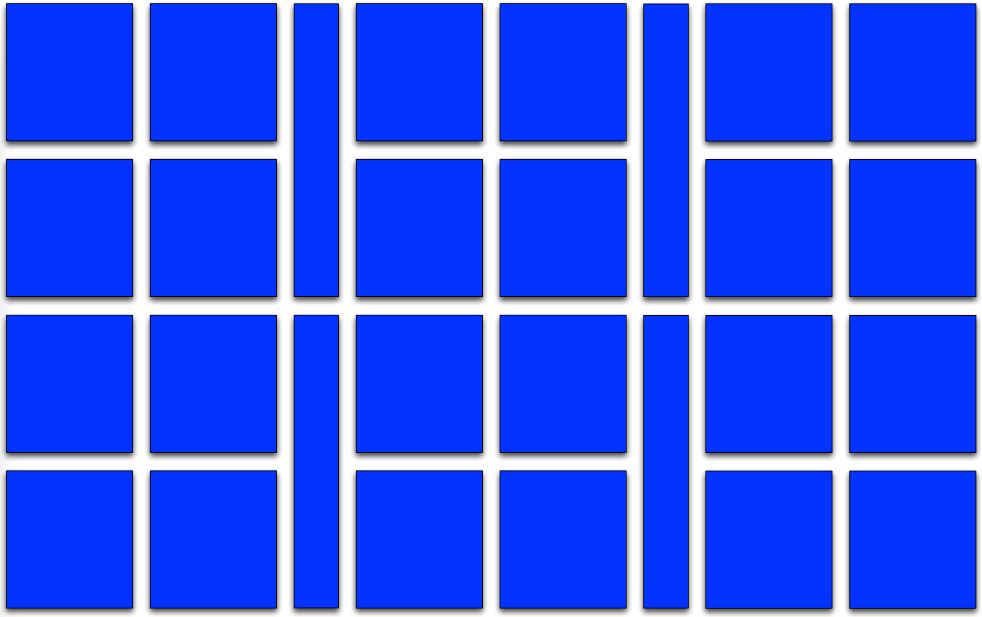
1111101

0100010100

1010100011

0101010111

1101010100



Reconfiguration in everyday life

Soccer

(Partial – Static)

Football

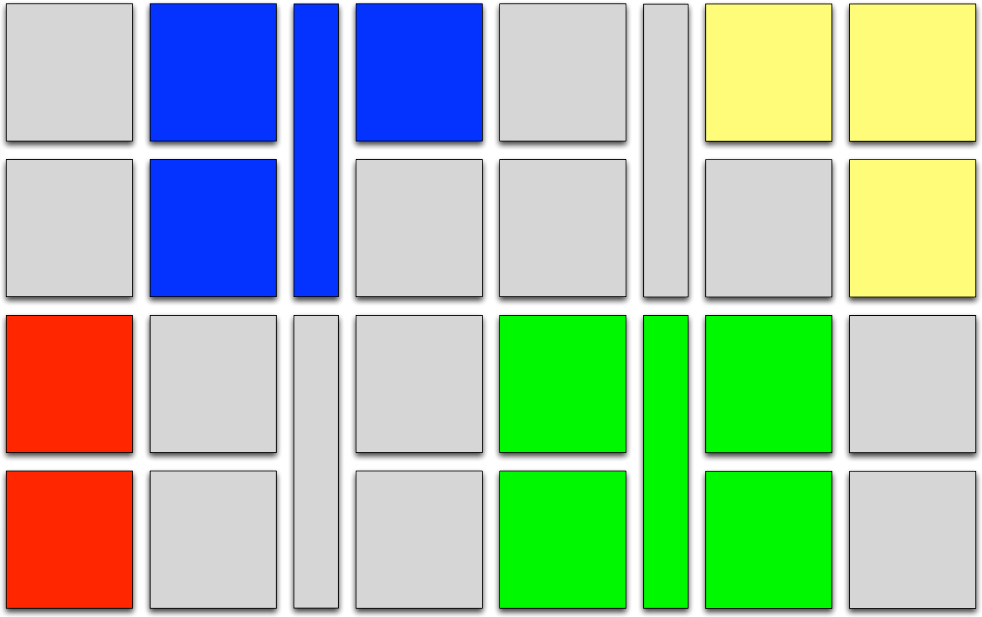
(Complete – Static)

1010000
1111101
0100010100
1010100011
0101010111
1101010100

1010000
1111101

1010000
1111101
0100010100

1010000
1111101
0100010100
1010100011
001010



Reconfiguration in everyday life

Soccer

(Partial – Static)

Football

(Complete – Static)

Hockey

(Partial – Dynamic)

SYSTEM HAS TO BE ADAPTIVE

TO RECOVER FROM DAMAGES

SYSTEM HAS TO BE ADAPTIVE

TO GUARANTEE SERVICES OVER
POWER CAP AND ENERGY SAVINGS

SYSTEM HAS TO BE ADAPTIVE

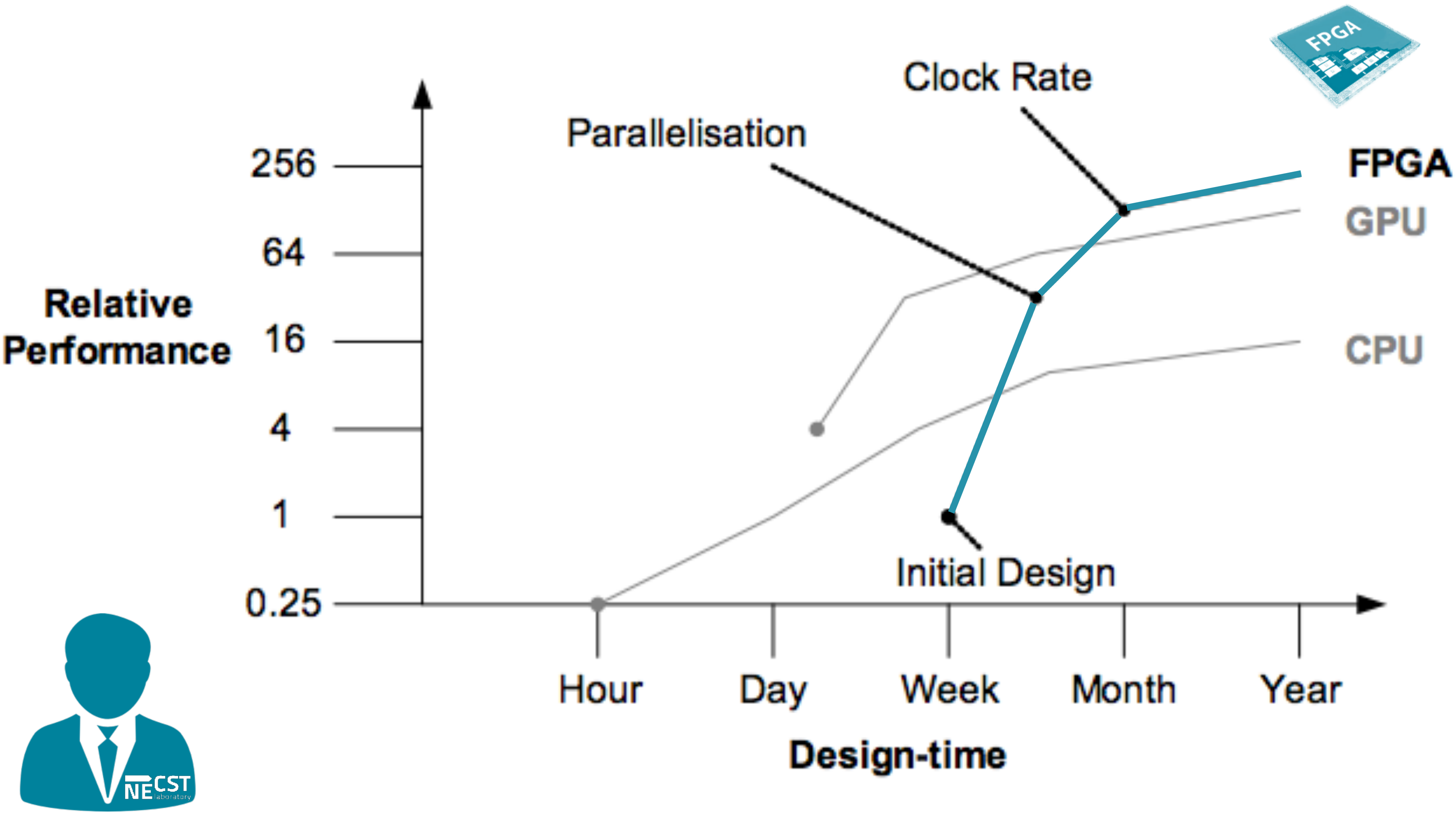
TO ADAPT TO

SYSTEM HAS TO BE ADAPTIVE

TO ADAPT TO UKNOWN CONDITIONS

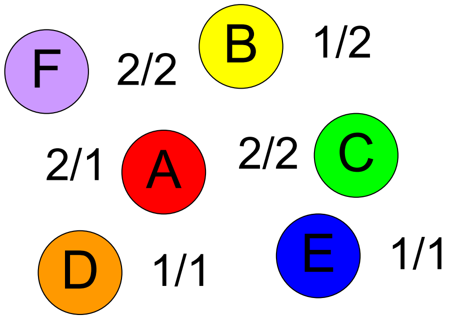
HOW TO MANAGE/DEAL WITH THESE HETEROGENOUS SYSTEMS

Research Challenge



Starting Scenario @ 2009

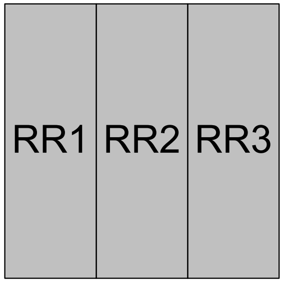
Set of Available Functionalities



Legenda:

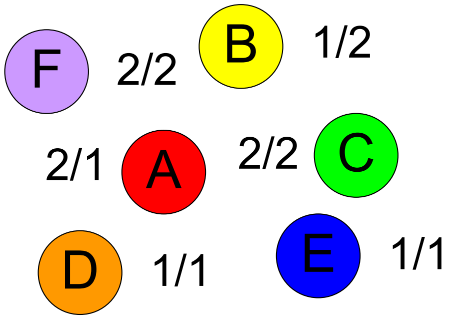
Area/Time F_i

FPGA

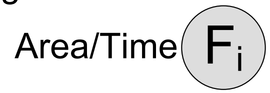


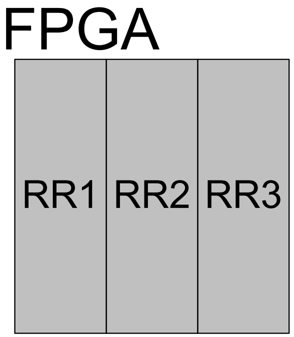
Starting Scenario @ 2009

Set of Available Functionalities

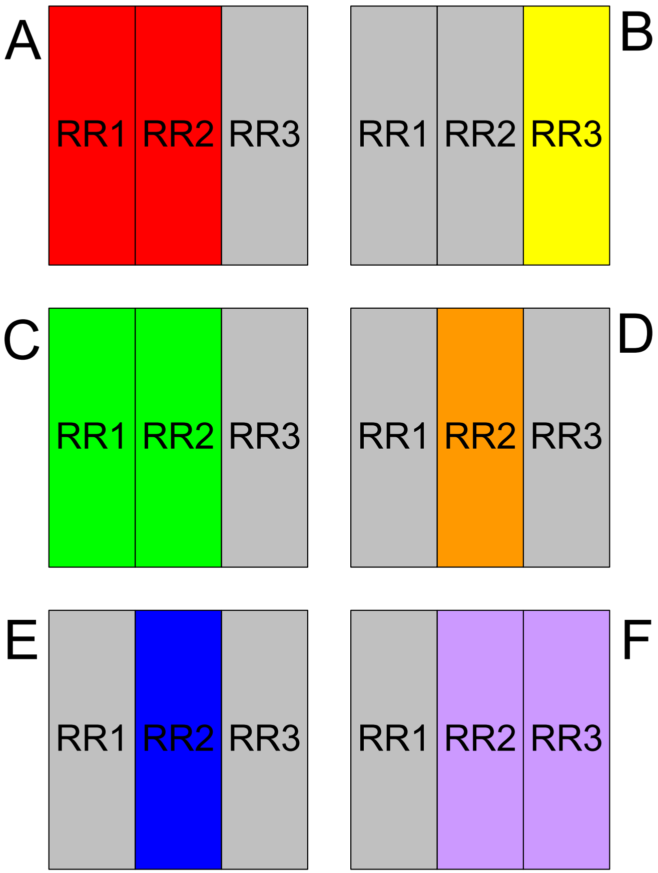


Legenda:



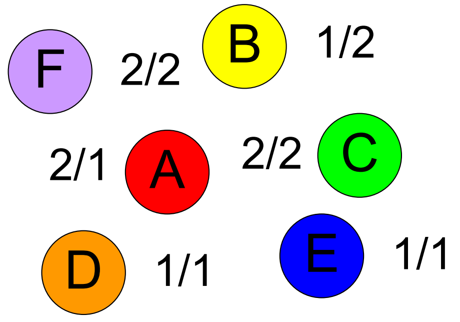


RC Implementations



Starting Scenario @ 2009

Set of Available Functionalities



F 2/2

B 1/2

2/1 A

2/2 C

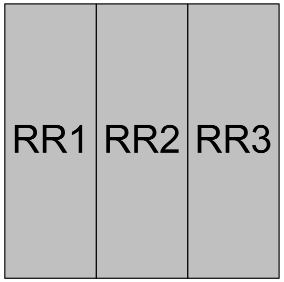
D 1/1

E 1/1

Legenda:

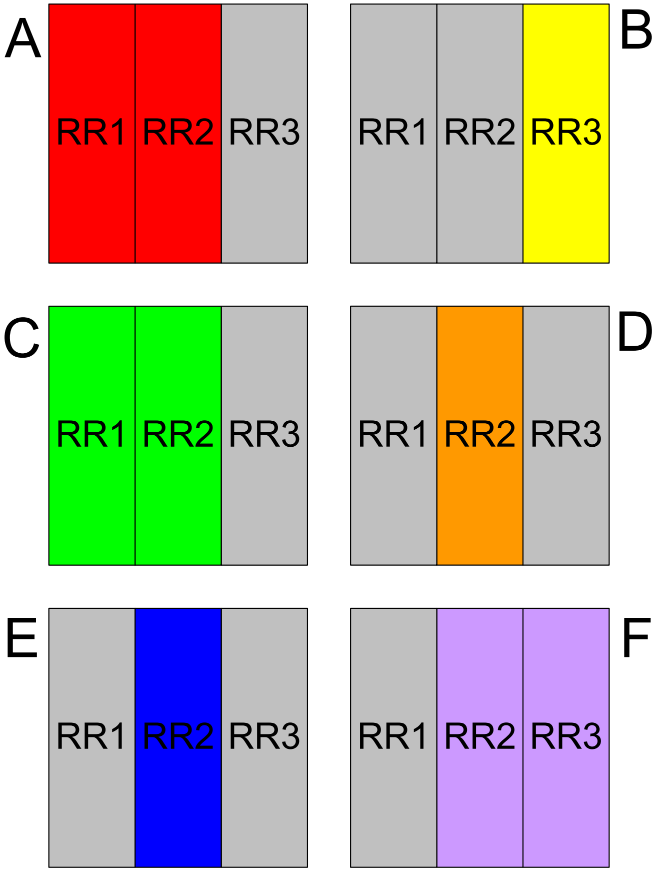
Area/Time F_i

FPGA



RR1 RR2 RR3

RC Implementations



A RR1 RR2 RR3

B RR1 RR2 RR3

C RR1 RR2 RR3

D RR1 RR2 RR3

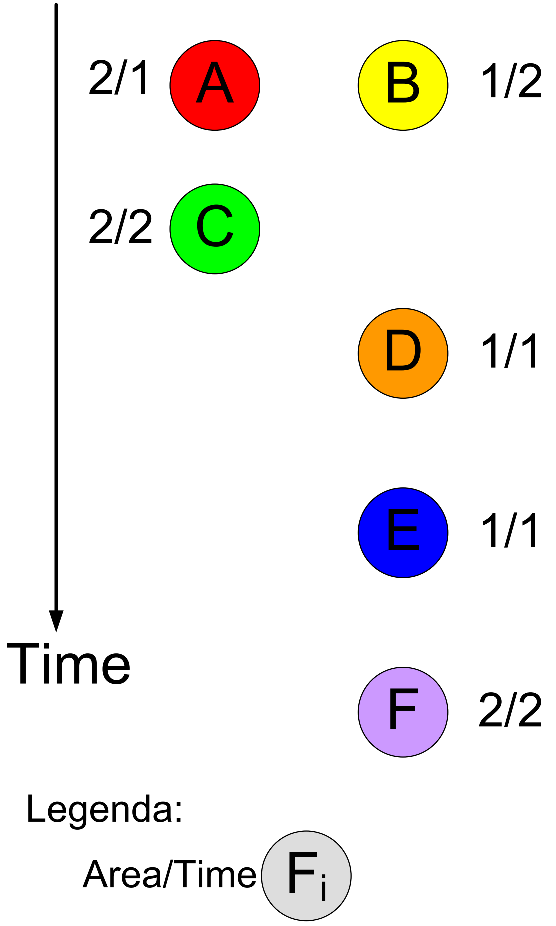
E RR1 RR2 RR3

F RR1 RR2 RR3

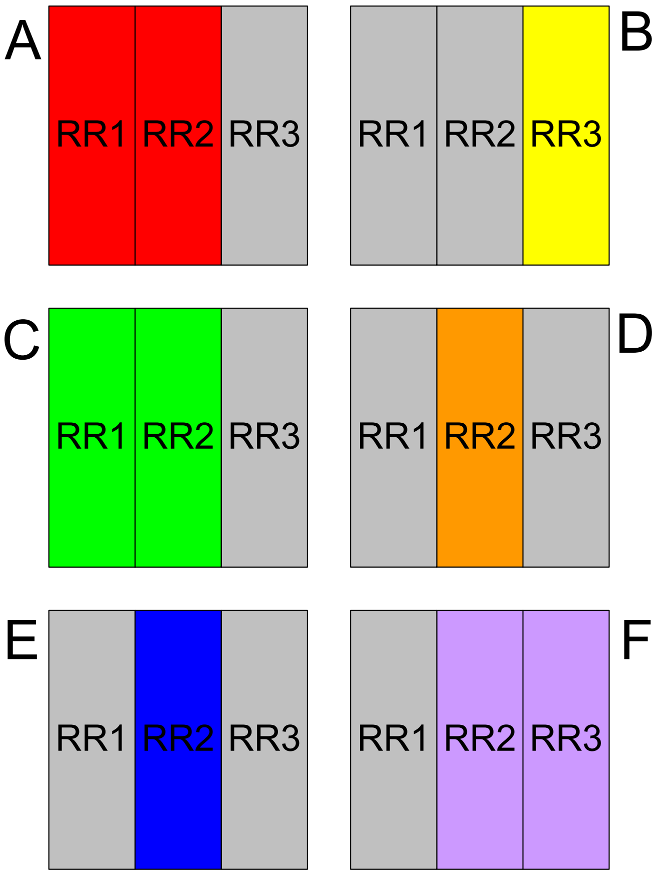
People Demanding for Functionalities

Starting Scenario @ 2009

A possible scenario



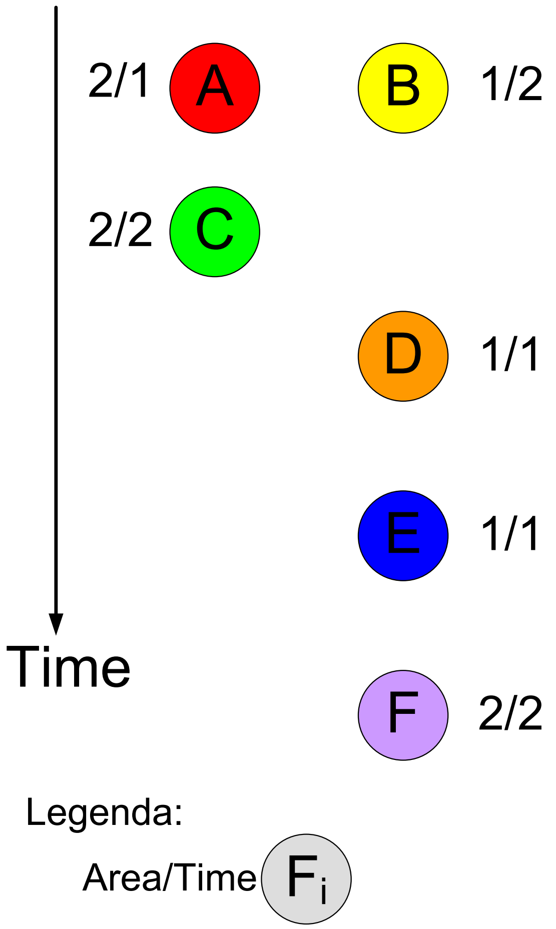
RC Implementations



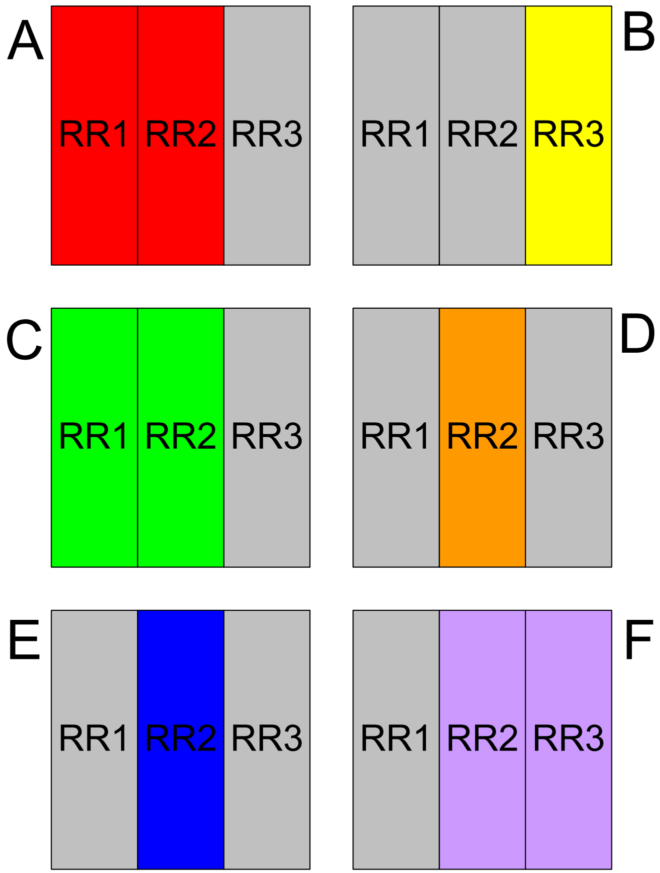
People Demanding for Functionalities

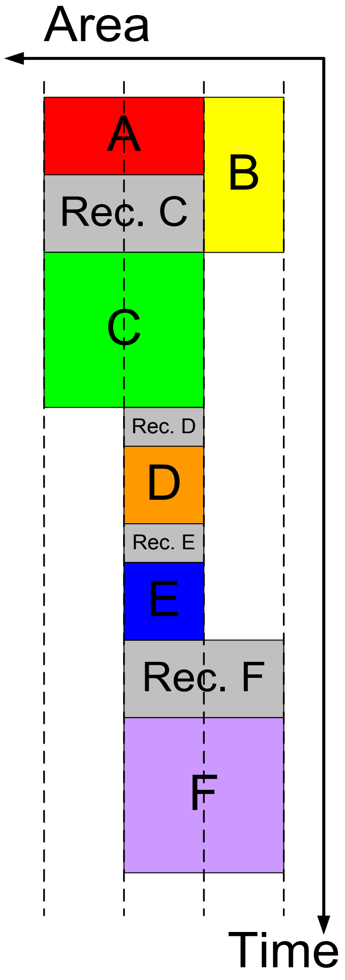
The Problem

A possible scenario



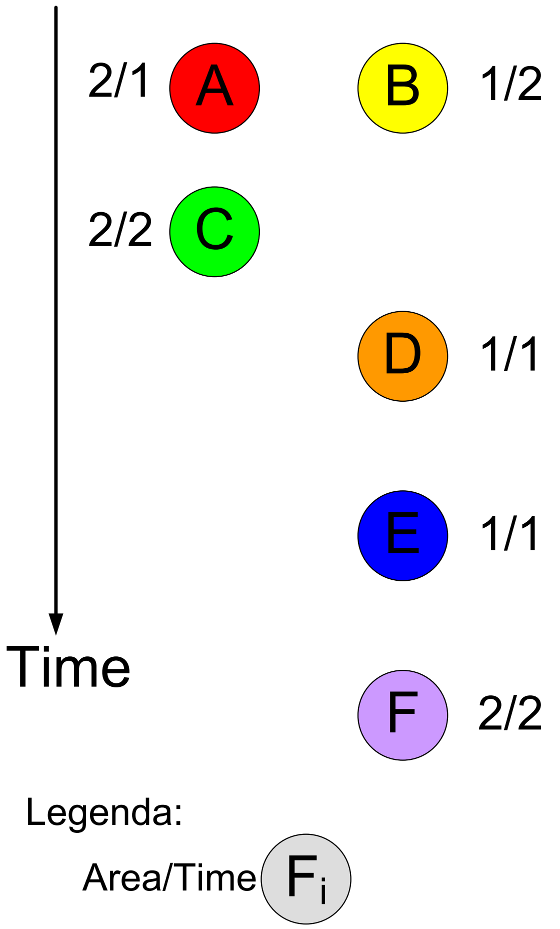
RC Implementations



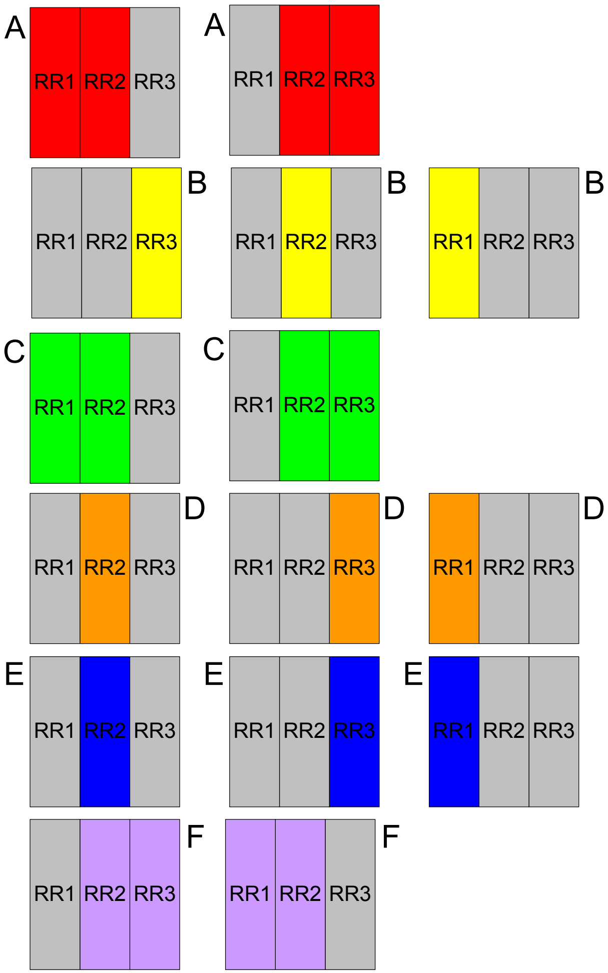


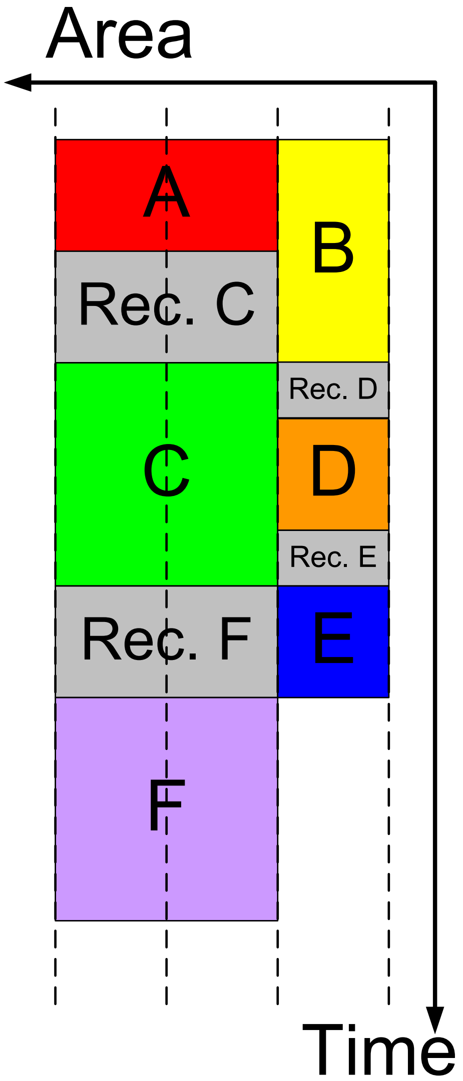
A Possible Solution

A possible scenario



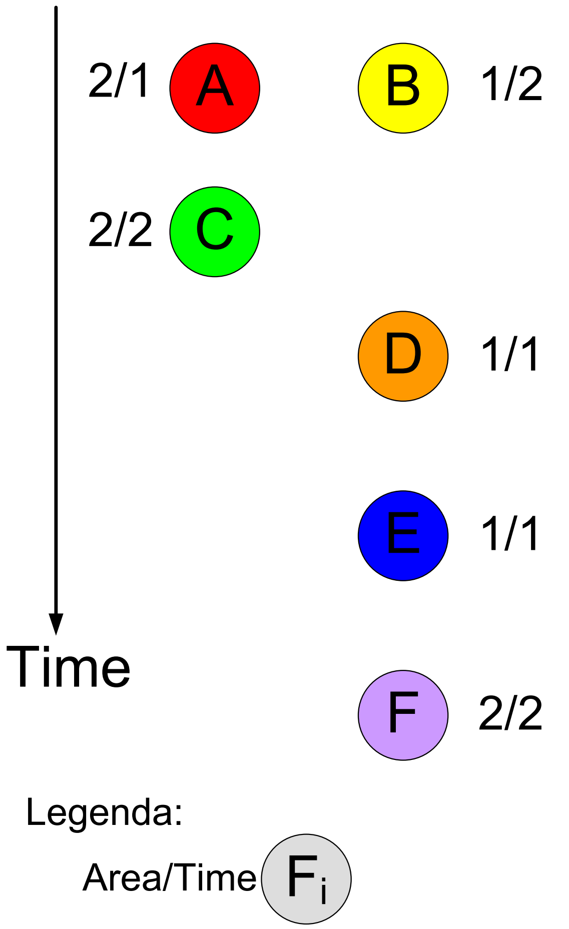
RC Implementations





Relocation

A possible scenario


2/1 A

B 1/2

2/2 C

D 1/1

E 1/1

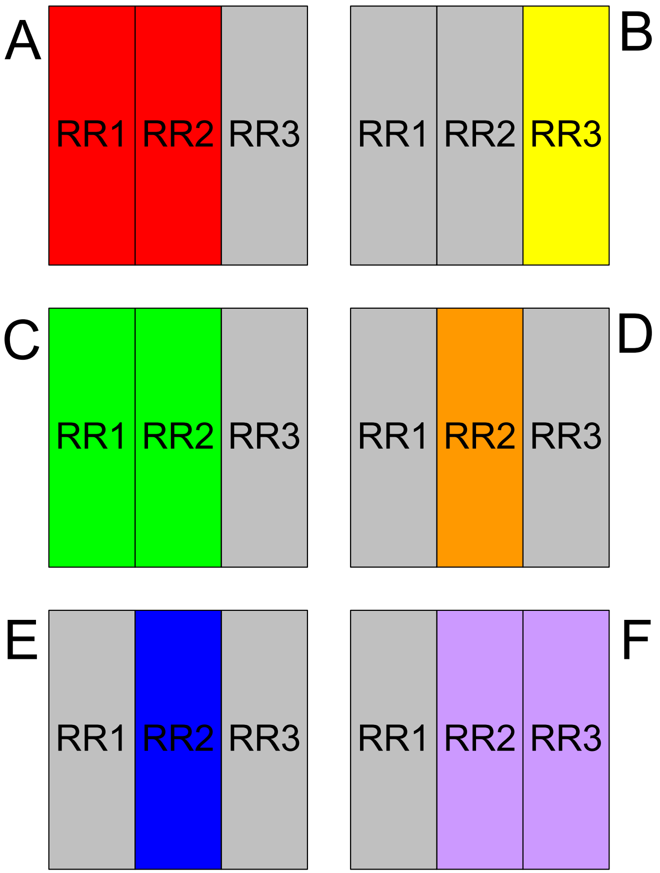
Time

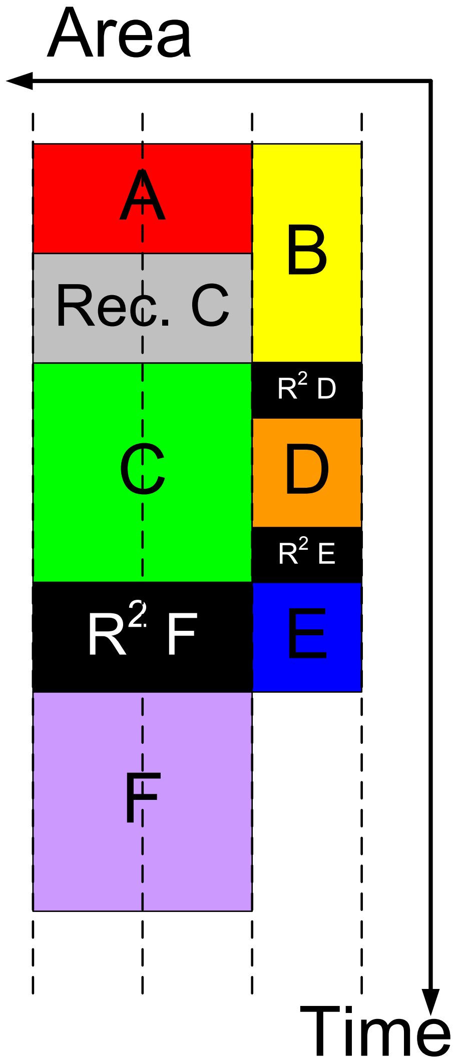
F 2/2

Legenda:

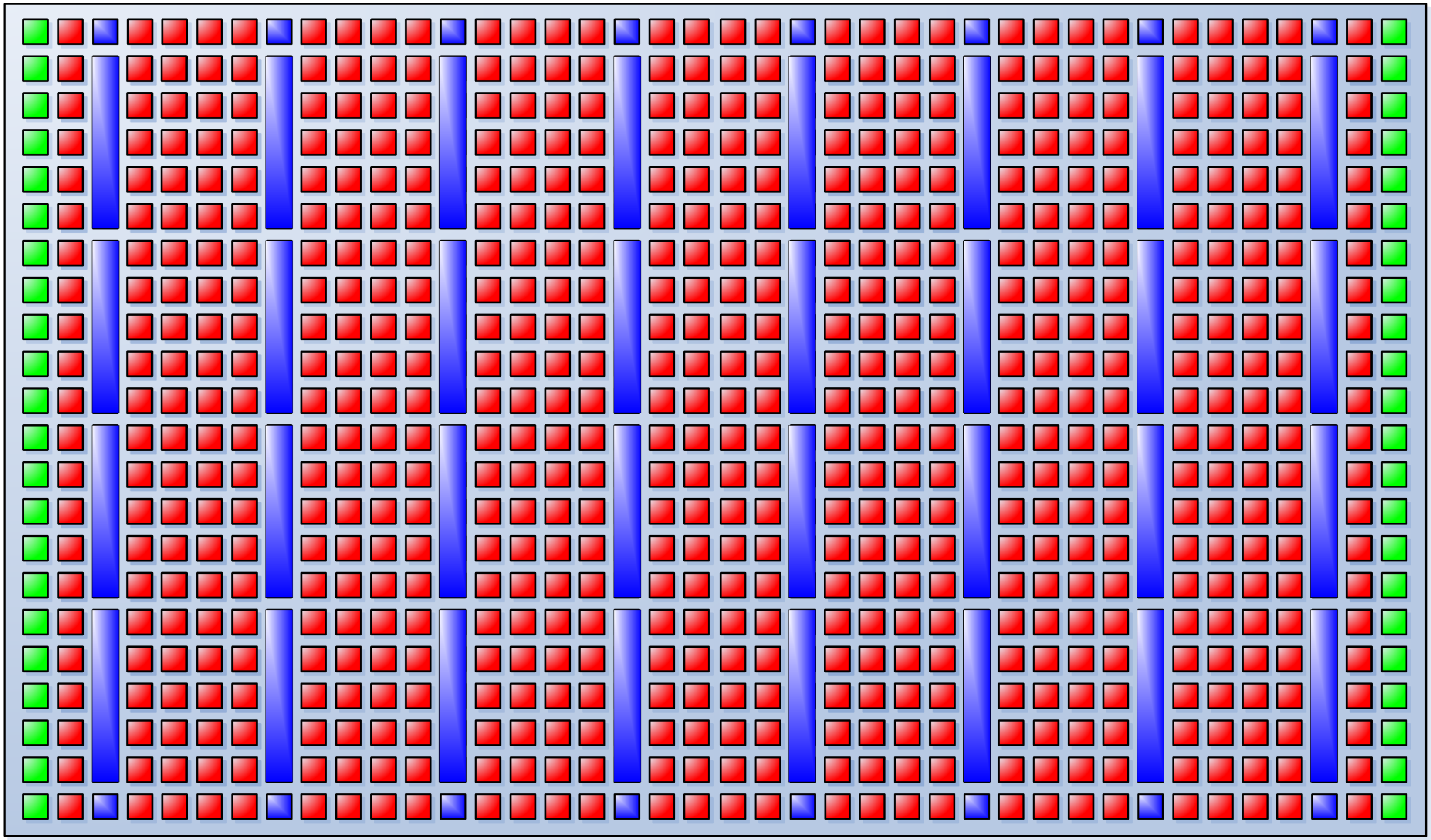
Area/Time F_i

RC
Implementations



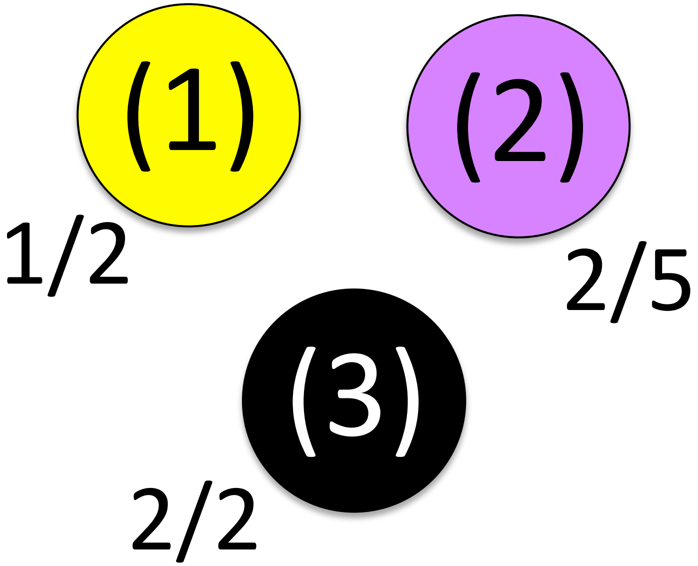


Relocation: Virtual homogeneity

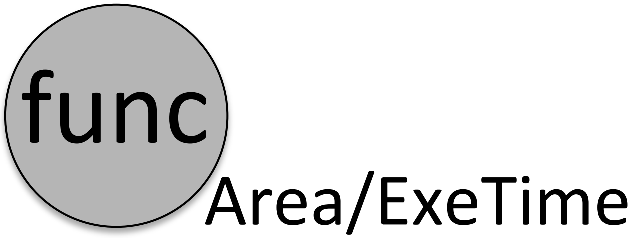


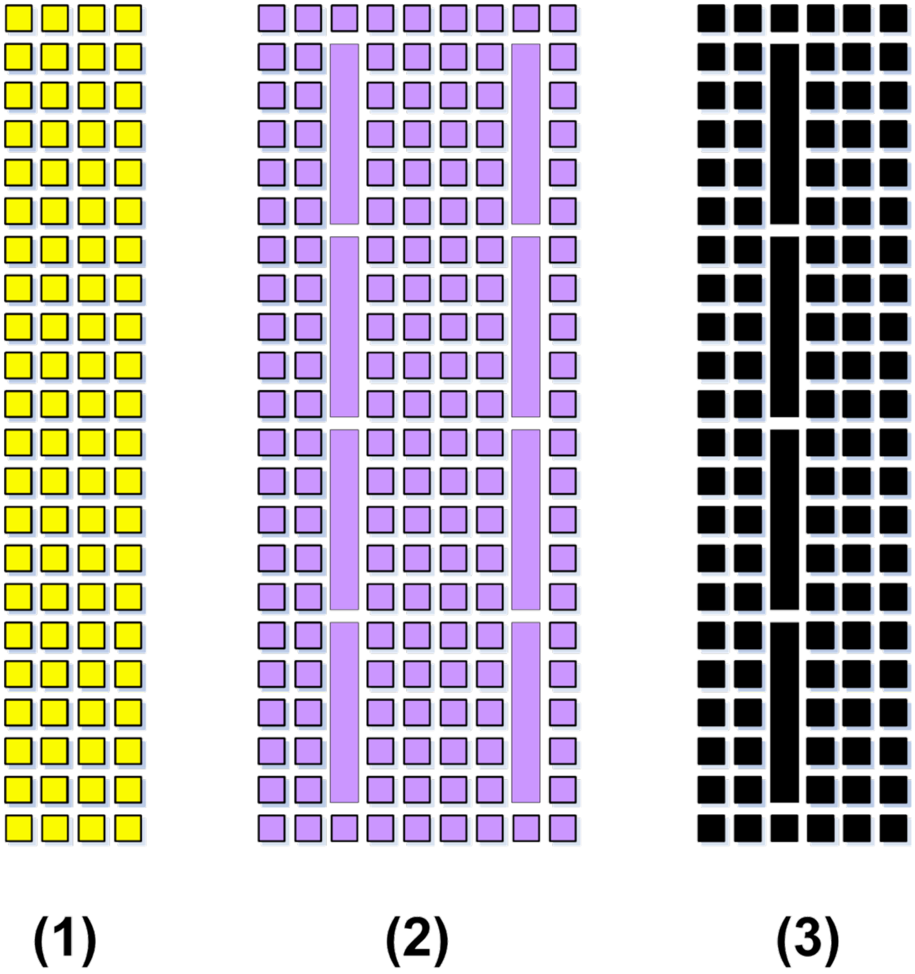
Relocation: Virtual homogeneity

Set of functionalities

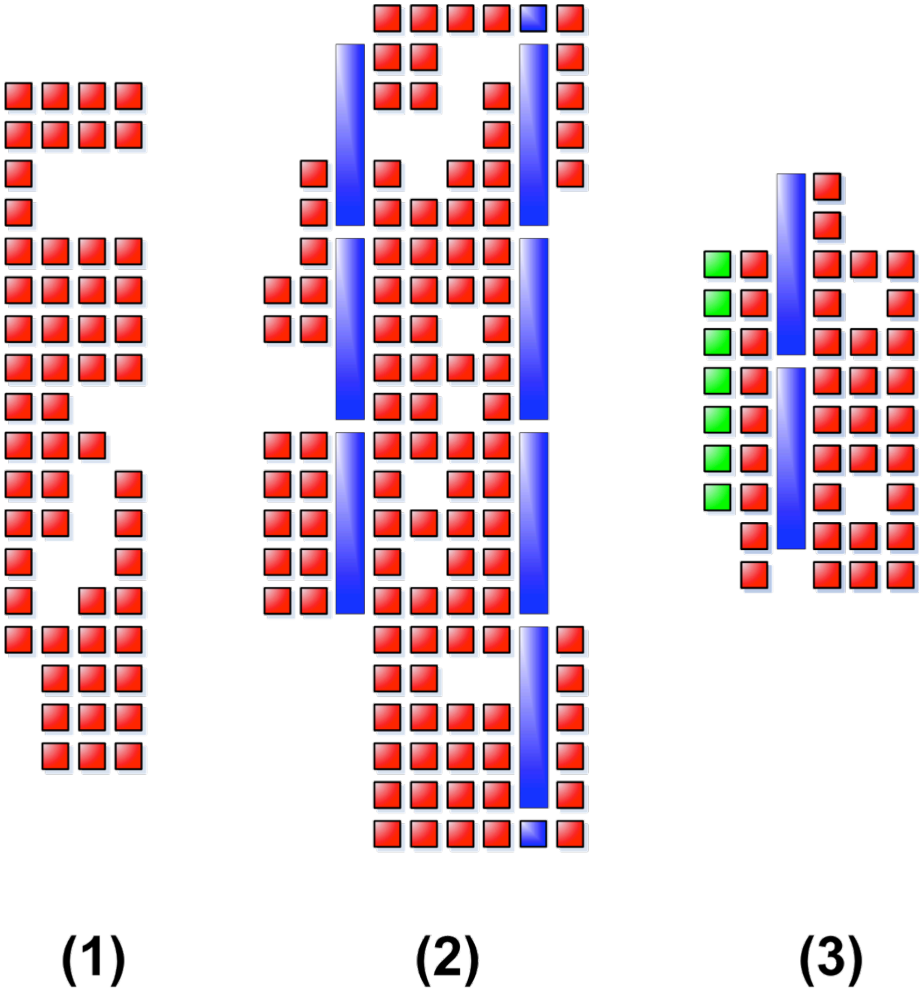


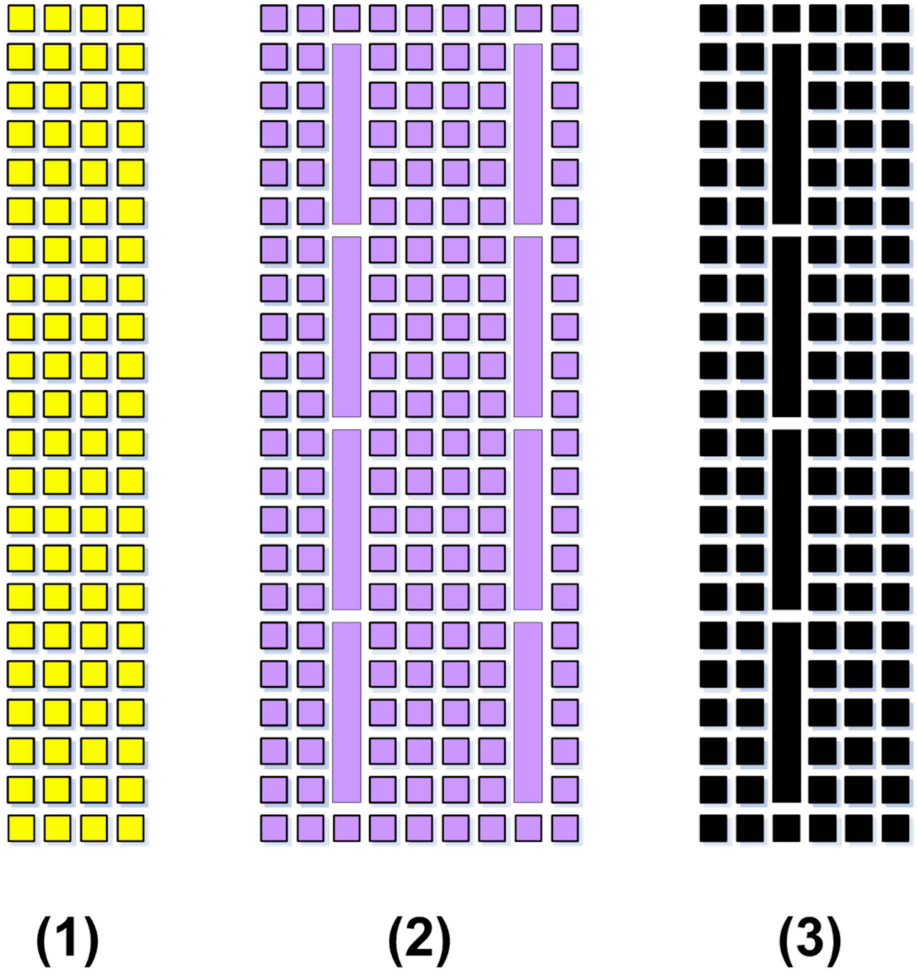
Legenda:





Relocation: Virtual homogeneity

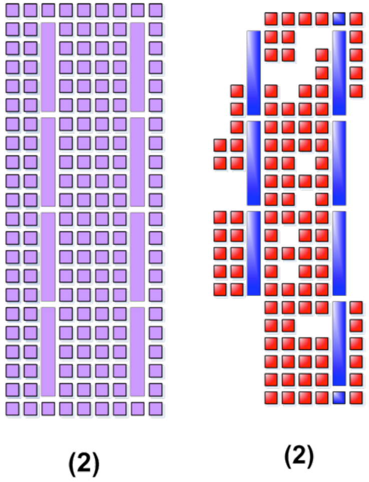


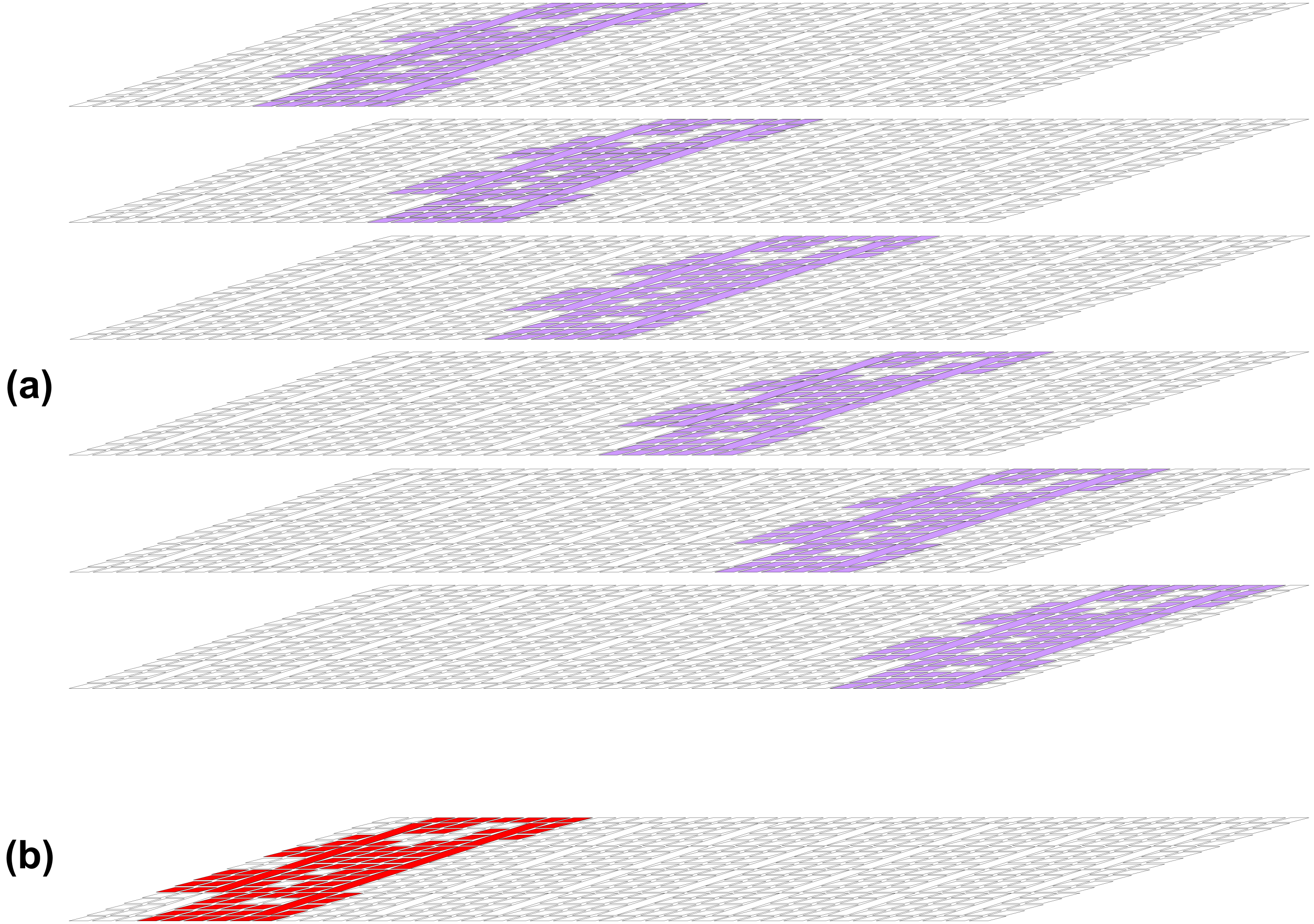


Relocation: Virtual homogeneity

(2)

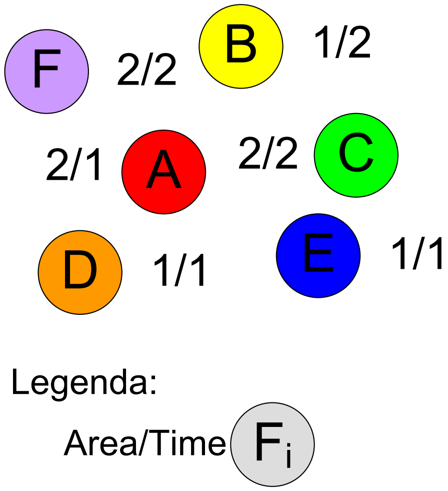
2/5



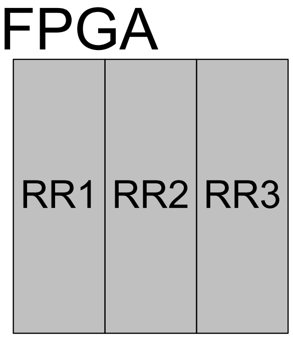


Starting Scenario @ 2009

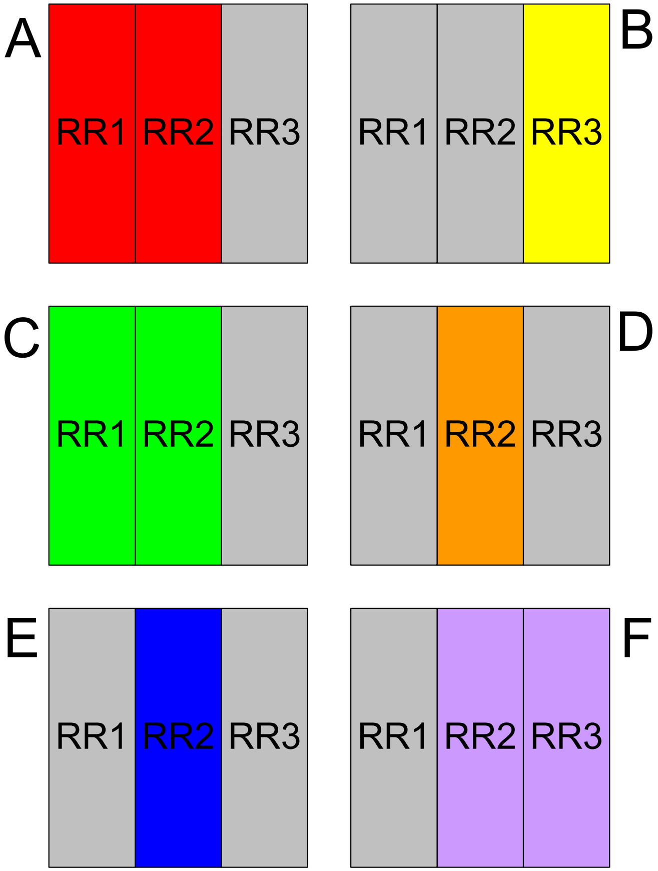
Set of Available Functionalities



FPGA



RC Implementations

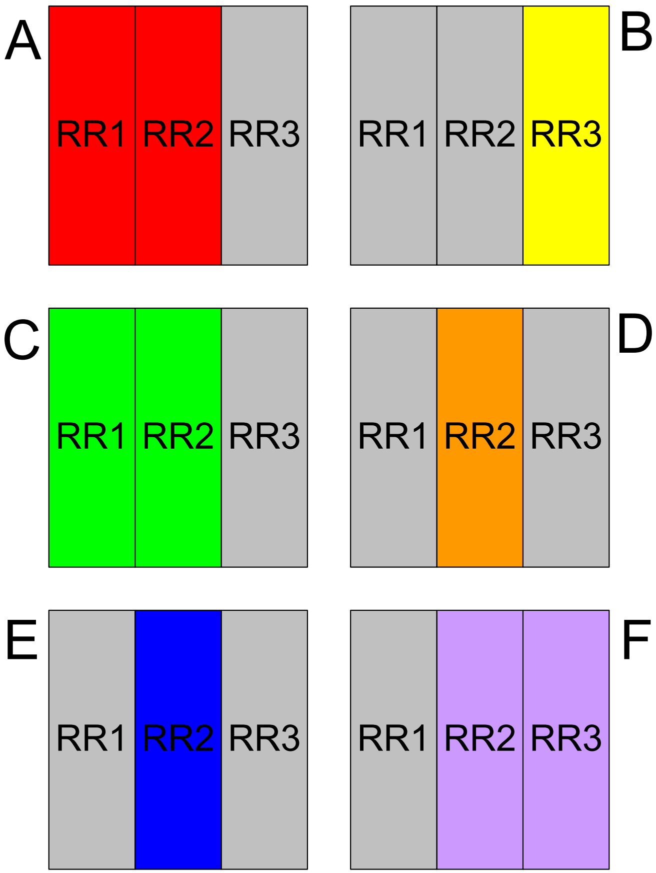


People Demanding for Functionalities

Starting Scenario

XILINX
ALL PROGRAMMABLE

RC Implementations



A
RR1 RR2 RR3

B
RR1 RR2 RR3

C
RR1 RR2 RR3

D
RR1 RR2 RR3

E
RR1 RR2 RR3

F
RR1 RR2 RR3

People Demanding for Functionalities

Starting Scenario

XILINX
ALL PROGRAMMABLE™

amazon
web services

People Demanding for Functionalities

Life is sharing

Starting Scenario

coursera

XILINX
ALL PROGRAMMABLE™

amazon
web services

NECST Scenario @ 2017/2018

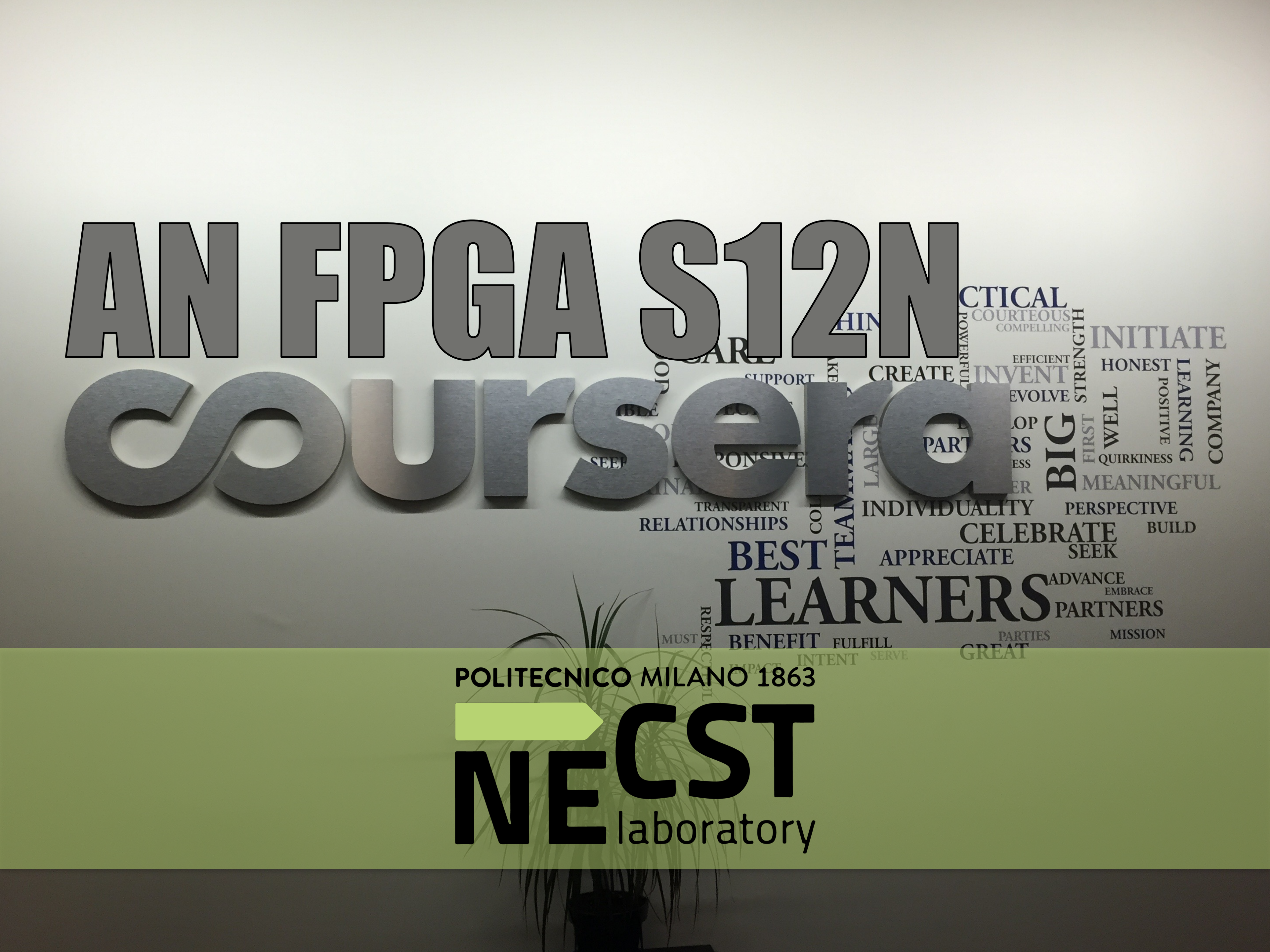
coursera

XILINX
ALL PROGRAMMABLE™

amazon web services

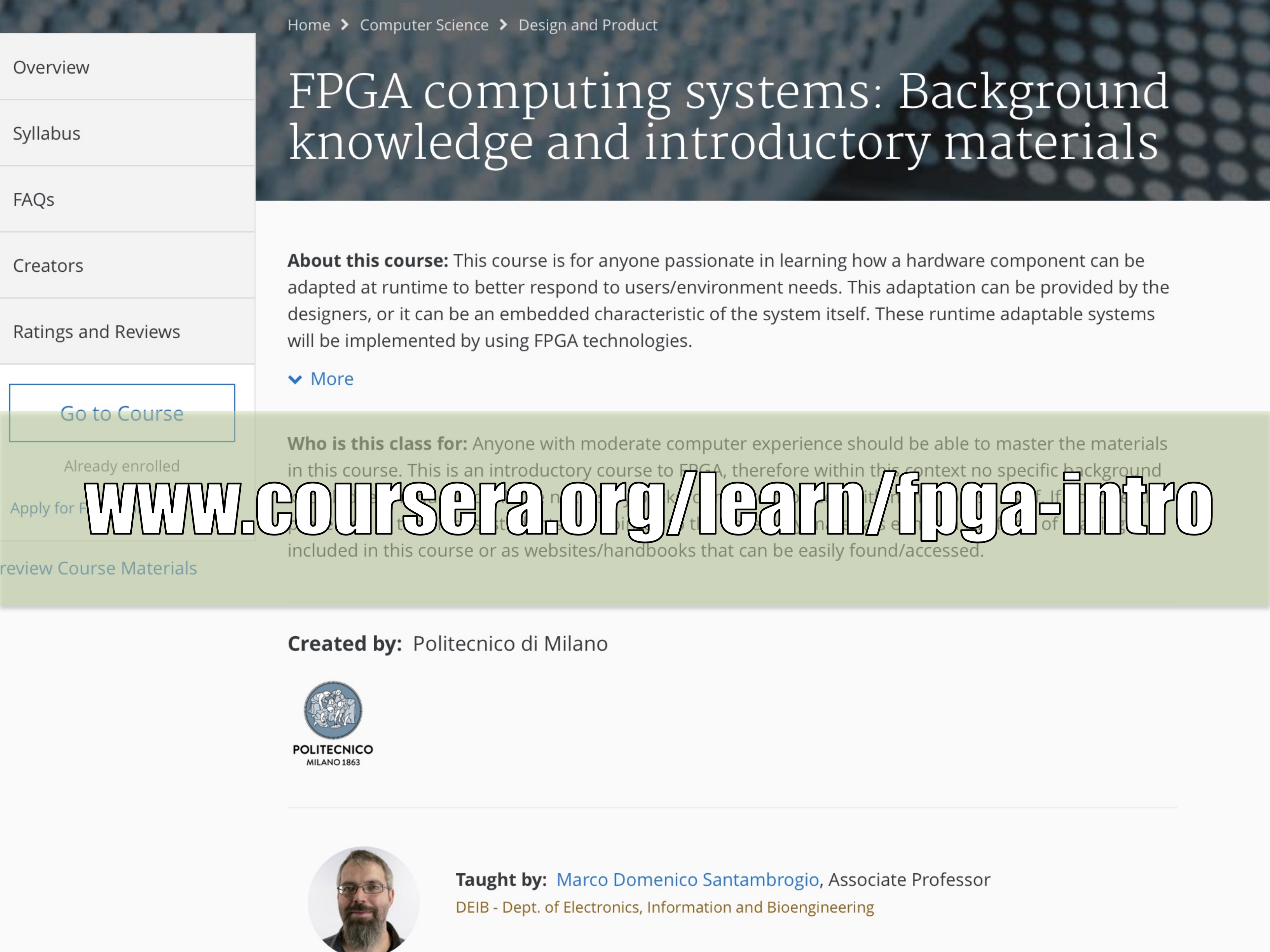
POLITECNICO MILANO 1863
NECST laboratory

AN FPGA S12N



POLITECNICO MILANO 1863

NECST laboratory



FPGA computing systems: Background knowledge and introductory materials

About this course: This course is for anyone passionate in learning how a hardware component can be adapted at runtime to better respond to users/environment needs. This adaptation can be provided by the designers, or it can be an embedded characteristic of the system itself. These runtime adaptable systems will be implemented by using FPGA technologies.

Who is this class for: Anyone with moderate computer experience should be able to master the materials in this course. This is an introductory course to FPGA, therefore within this context no specific background [illegible] included in this course or as websites/handbooks that can be easily found/accessed.

www.coursera.org/learn/fpga-intro

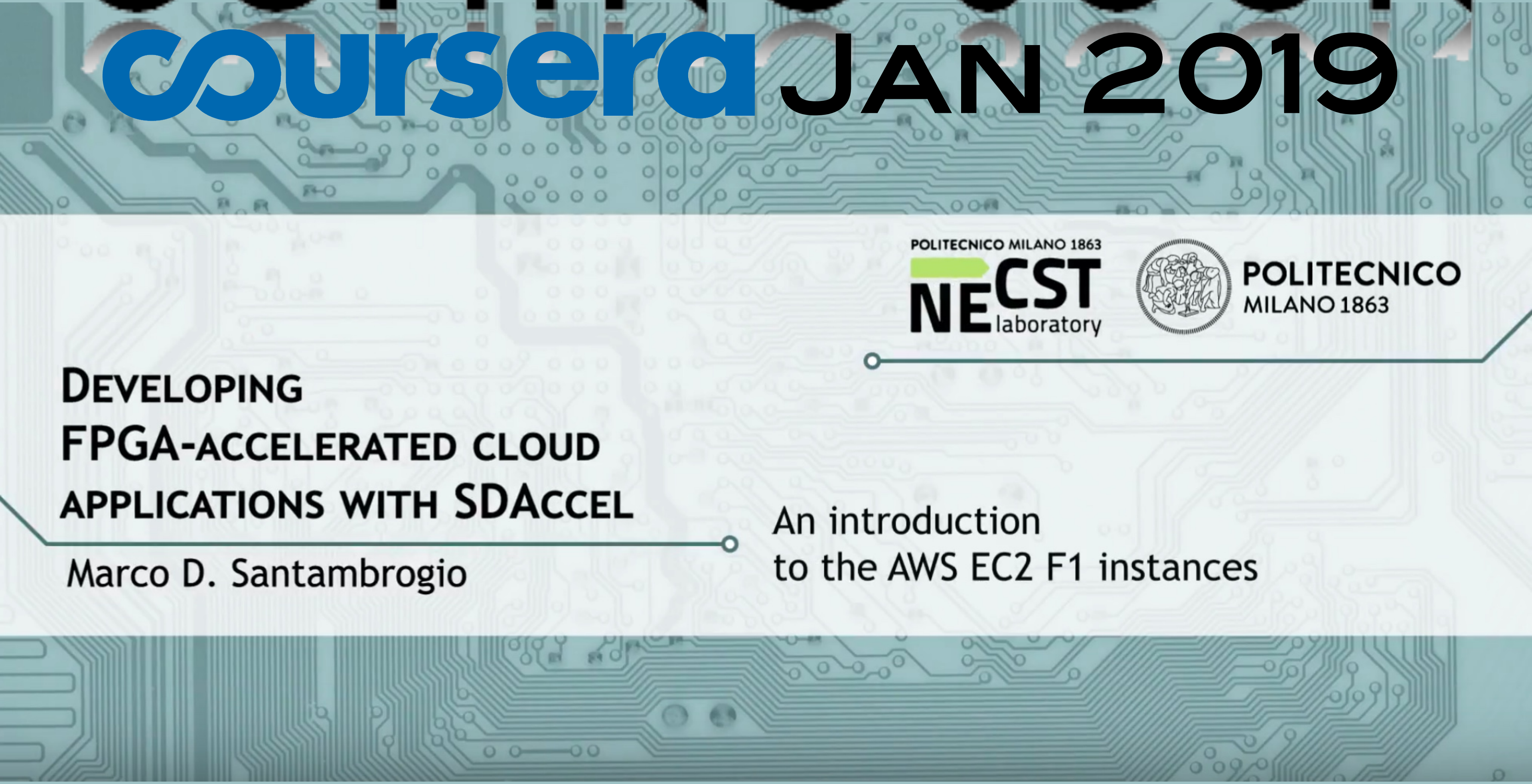
Created by: Politecnico di Milano

Taught by: Marco Domenico Santambrogio, Associate Professor

DEIB - Dept. of Electronics, Information and Bioengineering

COMING SOON

coursera JAN 2019



POLITECNICO MILANO 1863 NECST laboratory

POLITECNICO MILANO 1863

DEVELOPING FPGA-ACCELERATED CLOUD APPLICATIONS WITH SDACCEL

Marco D. Santambrogio

An introduction
to the AWS EC2 F1 instances

Some Applicative Domains for FPGA acceleration

- Image and Video Processing
- Security
- Machine Learning
- Genomics
- Financial Analytics
- Big Data Analytics

edico genome

MAXELER Technologies

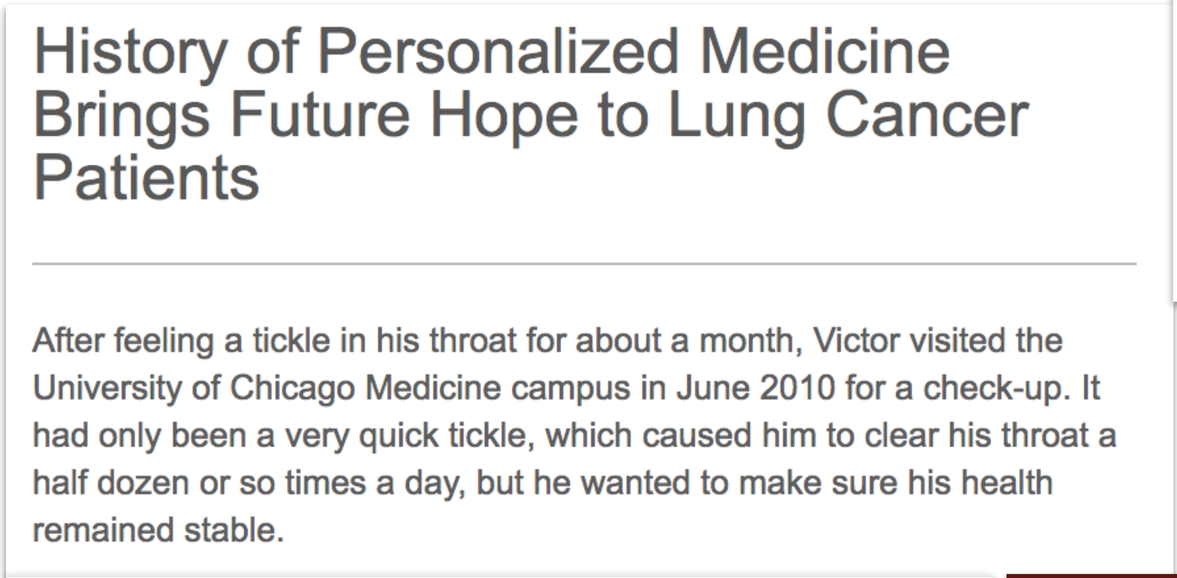
MAXIMUM PERFORMANCE COMPUTING

RYFT™

ACTIONABLE INTELLIGENCE FROM COMPLEX DATA

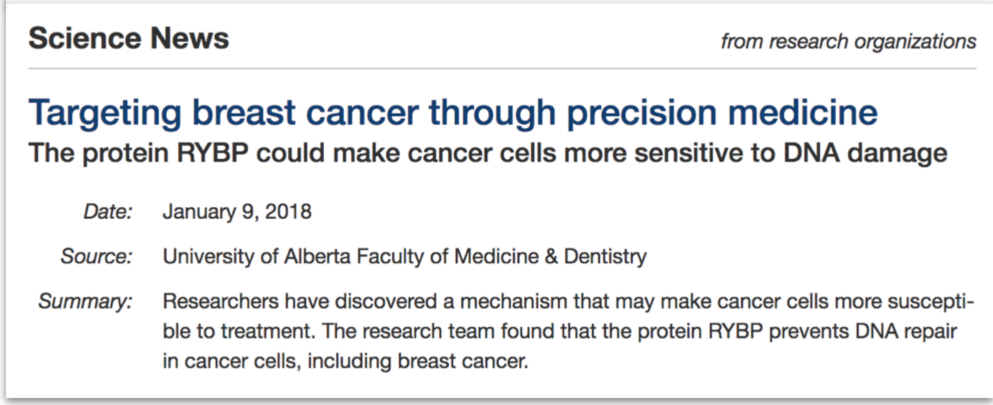
Who Victor is

How a genetic test changed Victor's life



History of Personalized Medicine Brings Future Hope to Lung Cancer Patients

After feeling a tickle in his throat for about a month, Victor visited the University of Chicago Medicine campus in June 2010 for a check-up. It had only been a very quick tickle, which caused him to clear his throat a half dozen or so times a day, but he wanted to make sure his health remained stable.



Science News *from research organizations*

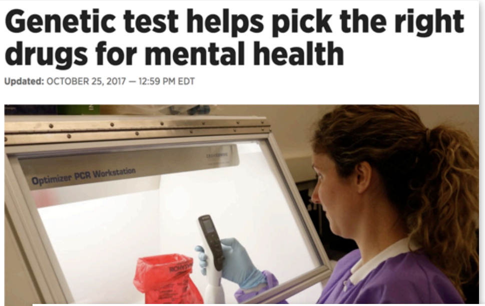
Targeting breast cancer through precision medicine

The protein RYBP could make cancer cells more sensitive to DNA damage

Date: January 9, 2018

Source: University of Alberta Faculty of Medicine & Dentistry

Summary: Researchers have discovered a mechanism that may make cancer cells more susceptible to treatment. The research team found that the protein RYBP prevents DNA repair in cancer cells, including breast cancer.

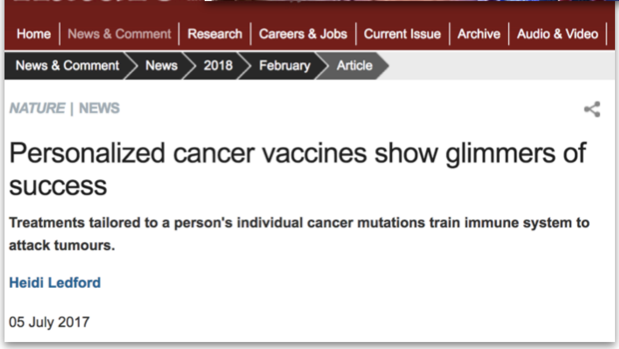


Genetic test helps pick the right drugs for mental health

Updated: OCTOBER 25, 2017 — 12:59 PM EDT

Personalized cancer vaccines successful in first-stage human trials

Rich Haridy | July 10th, 2017



nature
Home | News & Comment | Research | Careers & Jobs | Current Issue | Archive | Audio & Video
News & Comment > News > 2018 > February > Article

NATURE | NEWS

Personalized cancer vaccines show glimmers of success

Treatments tailored to a person's individual cancer mutations train immune system to attack tumours.

Heidi Ledford

05 July 2017

Open Challenges

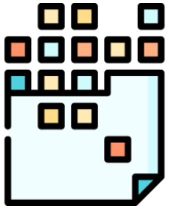
- It is necessary to keep-up with continuous development of biological research

Open Challenges

- It is necessary to keep-up with continuous development of biological research
- Each individual DNA provides huge amount of data

Open Challenges

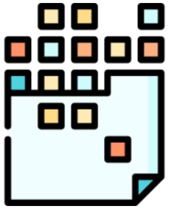
- It is necessary to keep-up with continuous development of biological research



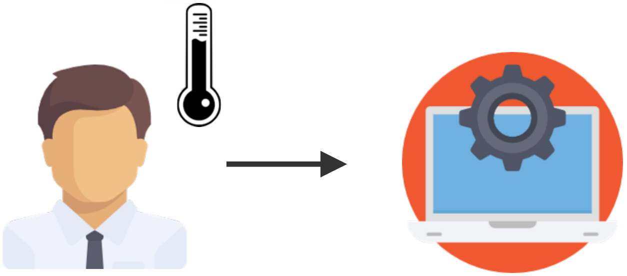
- Each individual DNA provides huge amount of data
- To produce a tailor-made drug, for each DNA:

Open Challenges

- It is necessary to keep-up with continuous development of biological research

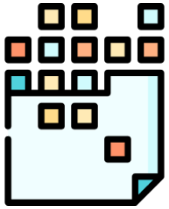


- Each individual DNA provides huge amount of data
- To produce a tailor-made drug, for each DNA:

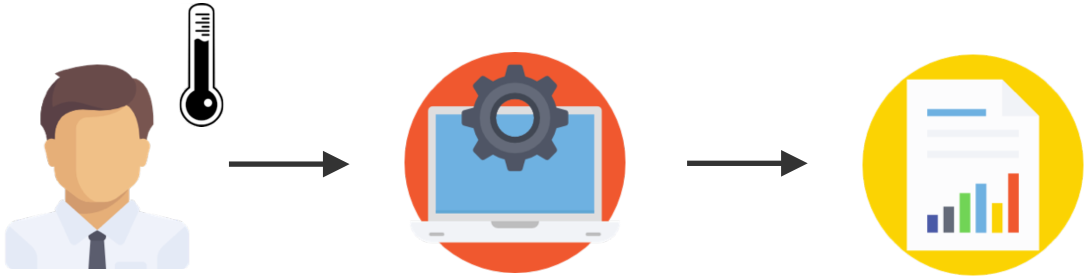


Open Challenges

- It is necessary to keep-up with continuous development of biological research



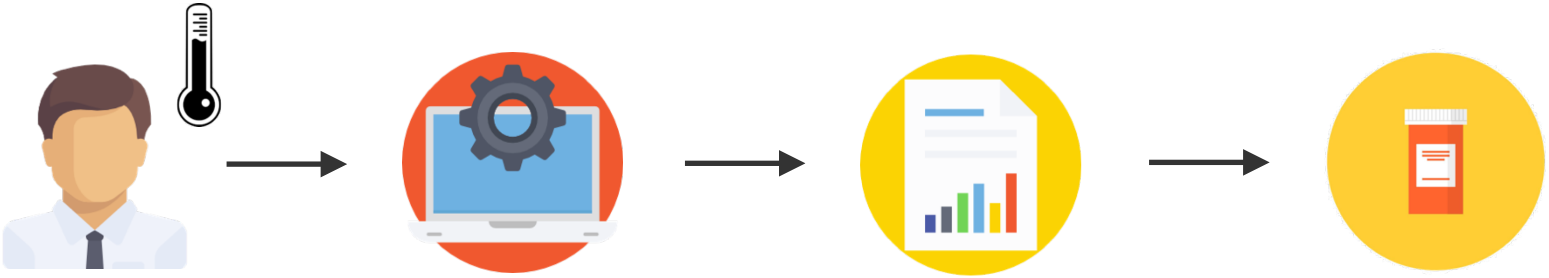
- Each individual DNA provides huge amount of data
- To produce a tailor-made drug, for each DNA:



Open Challenges

- It is necessary to keep-up with continuous development of biological research

- Each individual DNA provides huge amount of data
- To produce a tailor-made drug, for each DNA:



Personalized Medicine Today

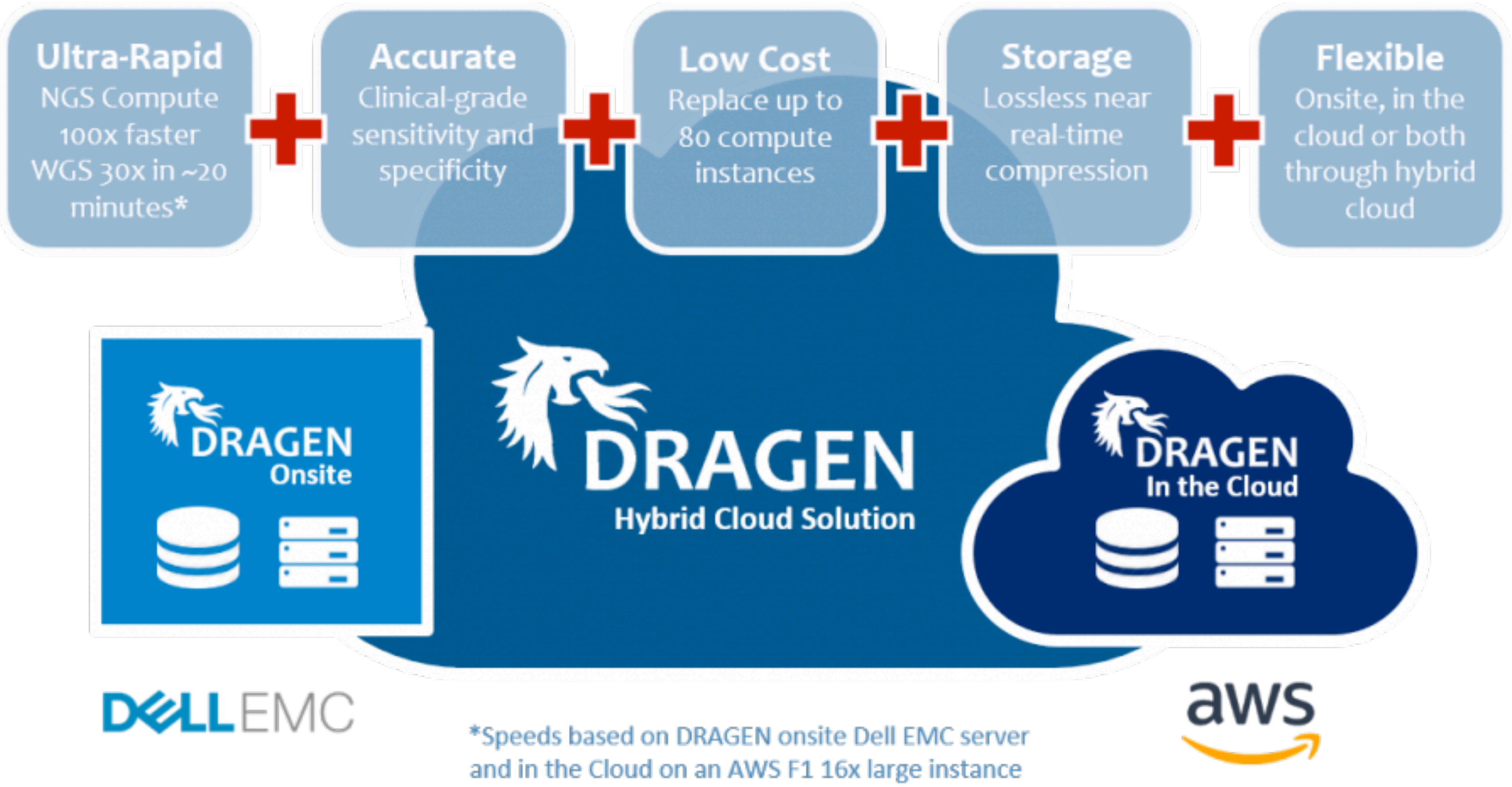
DRAGEN™

Bio-IT Processor

- FPGA-based acceleration
  - optimal ratio performance/power consumption
  - reconfigurability
- Possibility to use pre-accelerated biological pipelines
- Available on-site or for AWS cloud

edico genome

DRAGEN Hybrid Onsite & Cloud Solutions



Ultra-Rapid
NGS Compute 100x faster
WGS 30x in ~20 minutes*

Accurate
Clinical-grade sensitivity and specificity

Low Cost
Replace up to 80 compute instances

Storage
Lossless near real-time compression

Flexible
Onsite, in the cloud or both through hybrid cloud

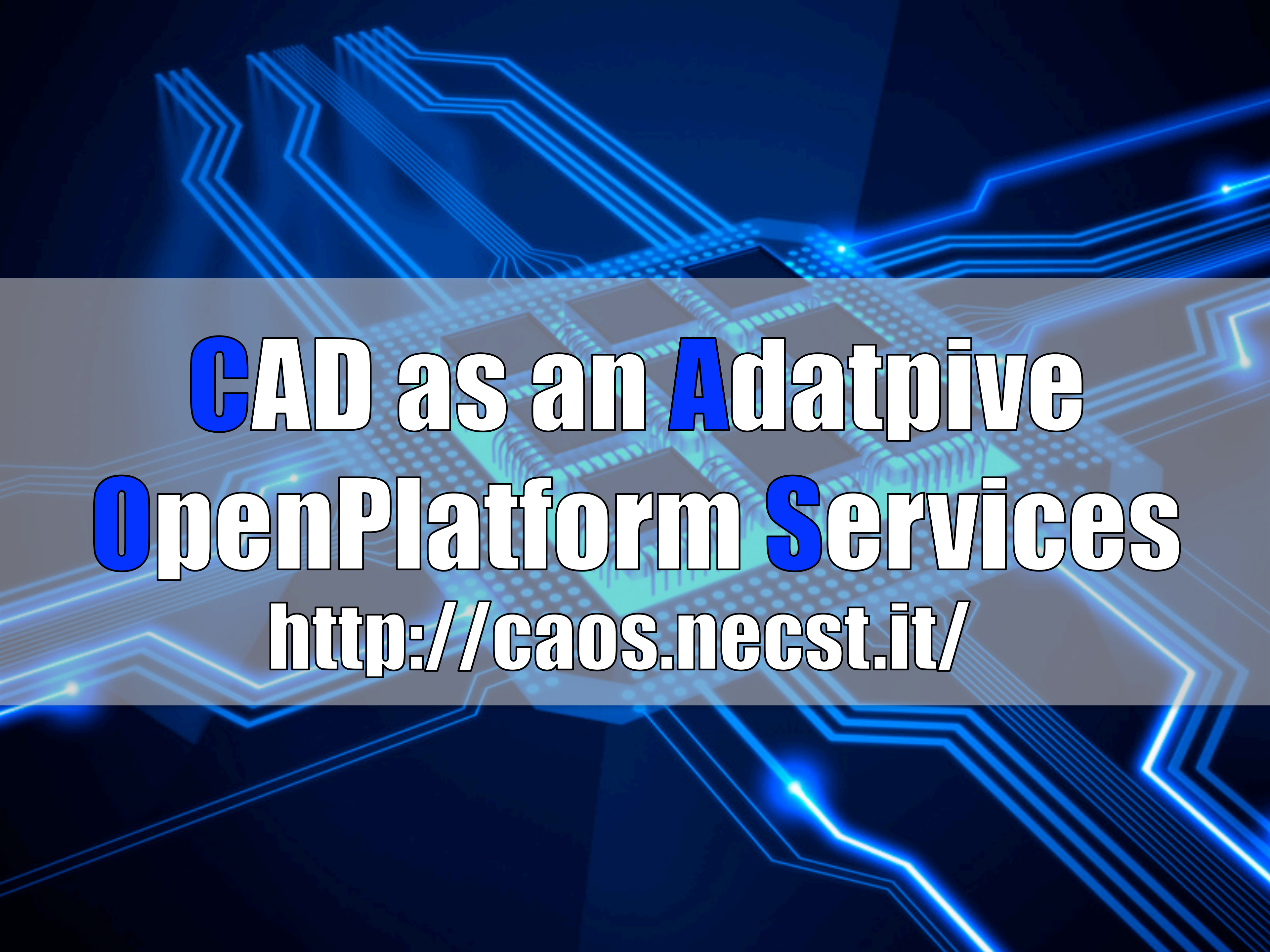
*Speeds based on DRAGEN onsite Dell EMC server and in the Cloud on an AWS F1 16x large instance

http://edicogenome.com/dragen-bioit-platform/

Benefits of the AWS F1 Cloud Compute Platform

- Makes FPGA acceleration available to a large community of developers, and to millions of potential AWS users
- Provides dedicated and large amounts of FPGA logic with elasticity to scale to multiple FPGAs
- Simplifies the development process by providing cloud-based FPGA development tools
- Provides a Marketplace for FPGA applications, giving more choice, secure and easy access to millions of AWS users

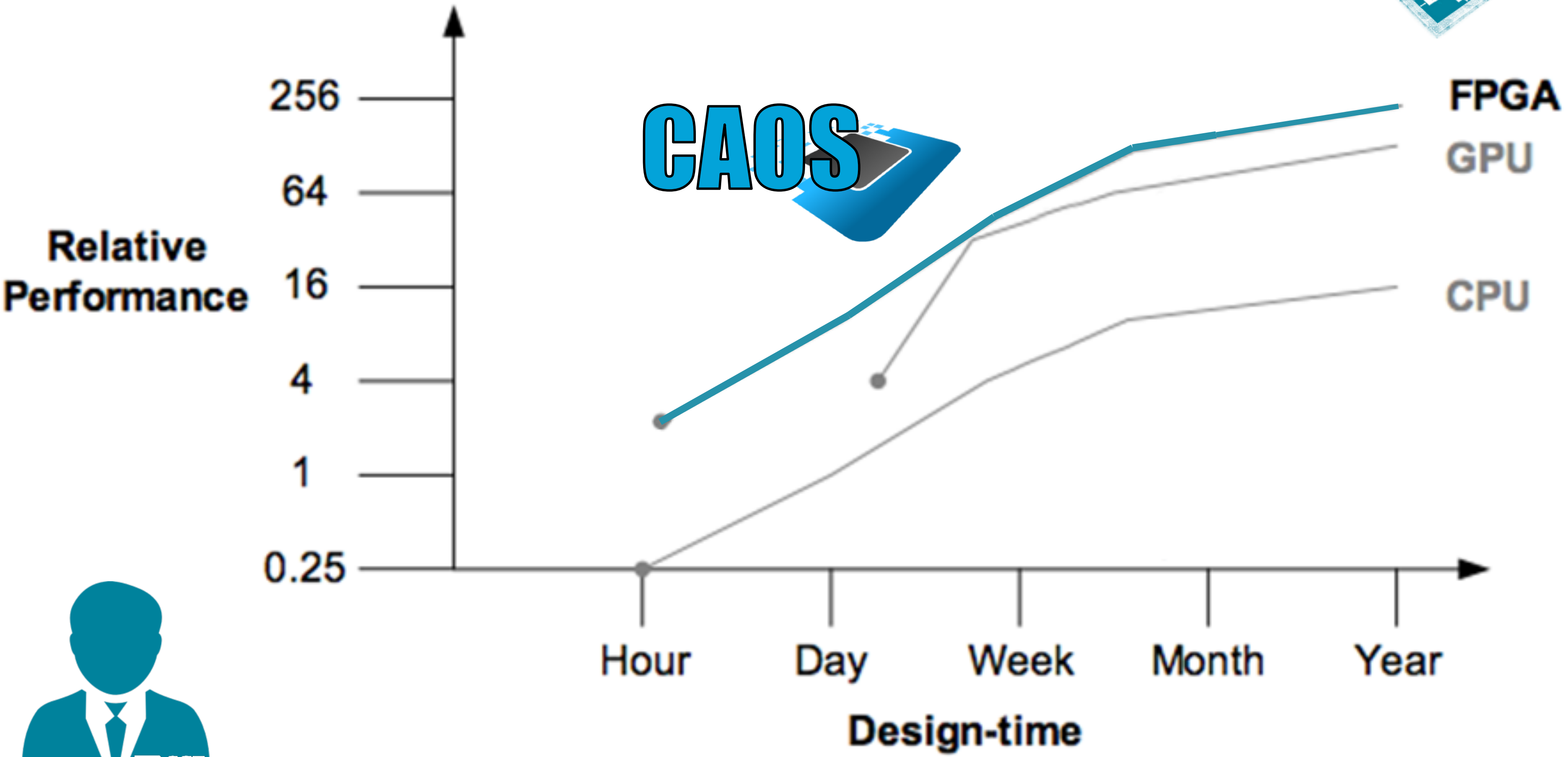
A RECONFIGURABLE-FRIENDLY ECOSYSTEM IS NEEDED



CAD as an Adatpive OpenPlatform Services

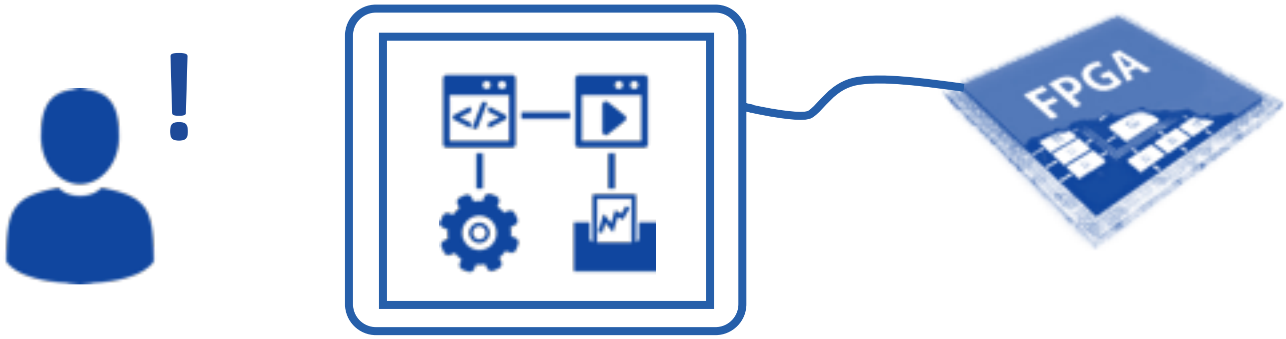
http://caos.necst.it/

CAD as an Adatpive OpenPlatform Services



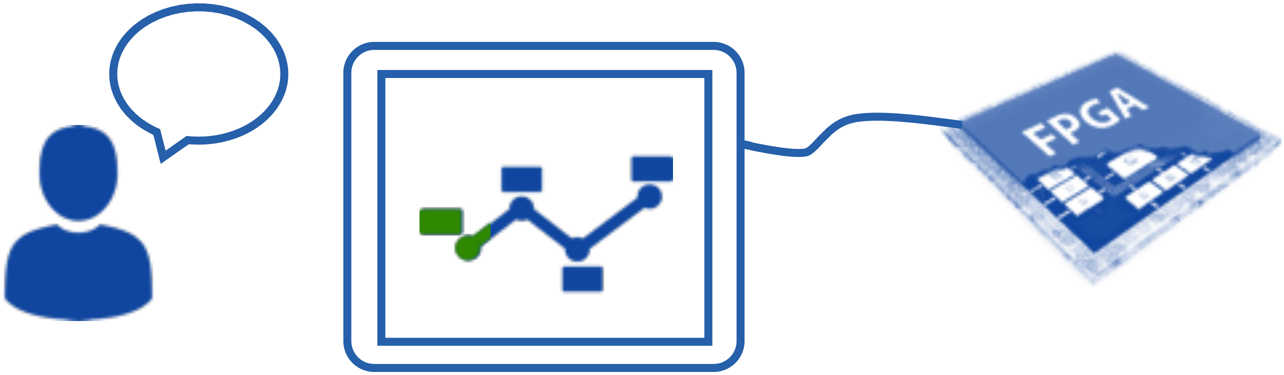
CAD as an Adatpive OpenPlatform Services

Usability



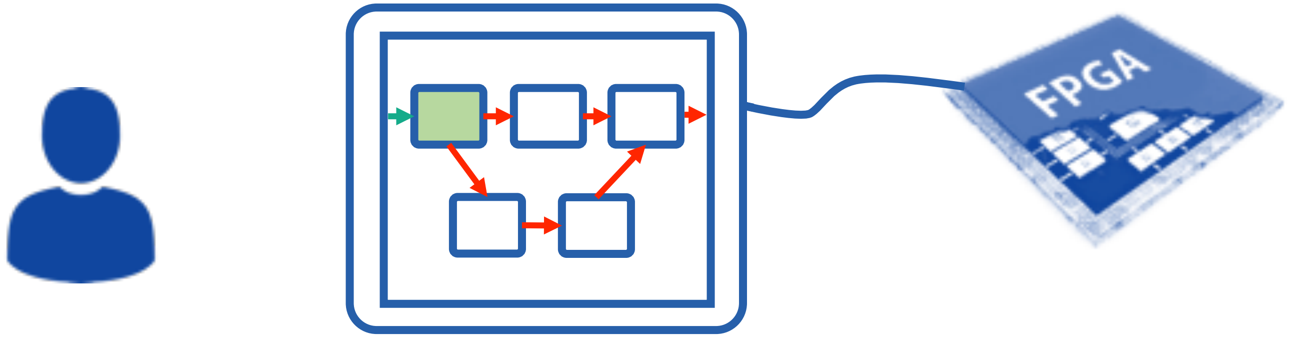
CAD as an Adatpive OpenPlatform Services

Interactivity



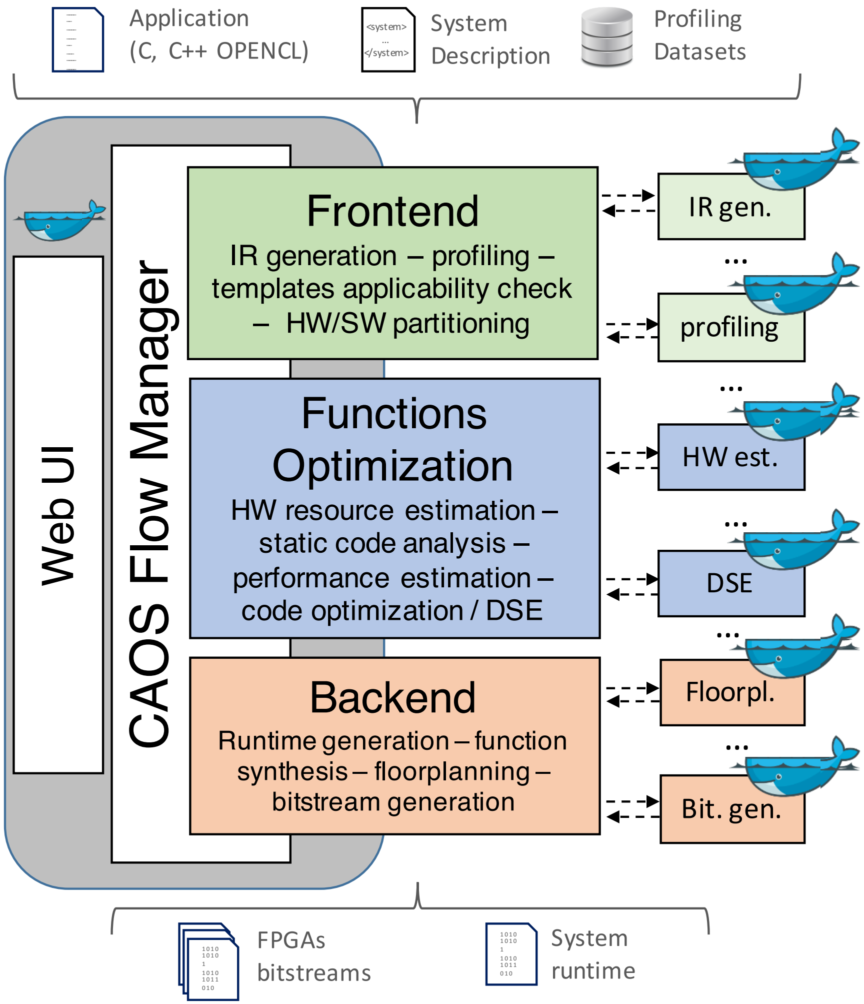
CAD as an Adatpive OpenPlatform Services

Modularity

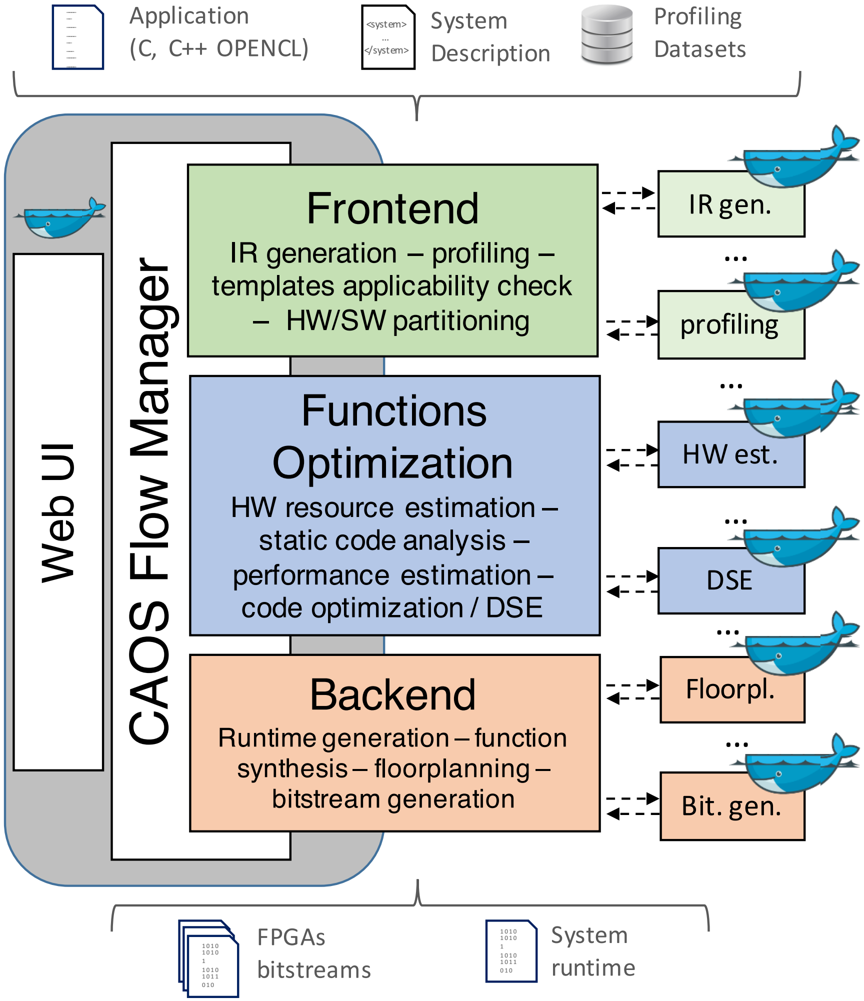


How to
do it?

The proposed CAOS framework


Application
(C, C++ OPENCL)
<system>
...
</system>
System
Description
Profiling
Datasets
Web UI
CAOS Flow Manager
Frontend
IR generation – profiling – templates applicability check – HW/SW partitioning
Functions Optimization
HW resource estimation – static code analysis – performance estimation – code optimization / DSE
Backend
Runtime generation – function synthesis – floorplanning – bitstream generation
IR gen.
...
profiling
...
HW est.
...
DSE
...
Floorpl.
...
Bit. gen.
FPGAs
bitstreams
System
runtime

The proposed CAOS framework



Application (C, C++ OPENCL)

System Description

Profiling Datasets

Web UI

CAOS Flow Manager

Frontend
IR generation – profiling – templates applicability check – HW/SW partitioning

Functions Optimization
HW resource estimation – static code analysis – performance estimation – code optimization / DSE

Backend
Runtime generation – function synthesis – floorplanning – bitstream generation

IR gen.

…

profiling

…

HW est.

…

DSE

…

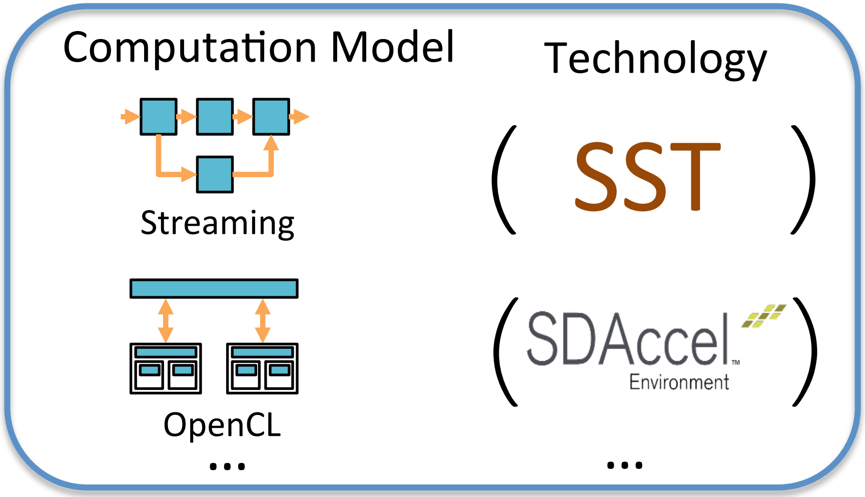
Floorpl.

…

Bit. gen.

FPGAs bitstreams

System runtime



Computation Model

Streaming

OpenCL

…

Technology

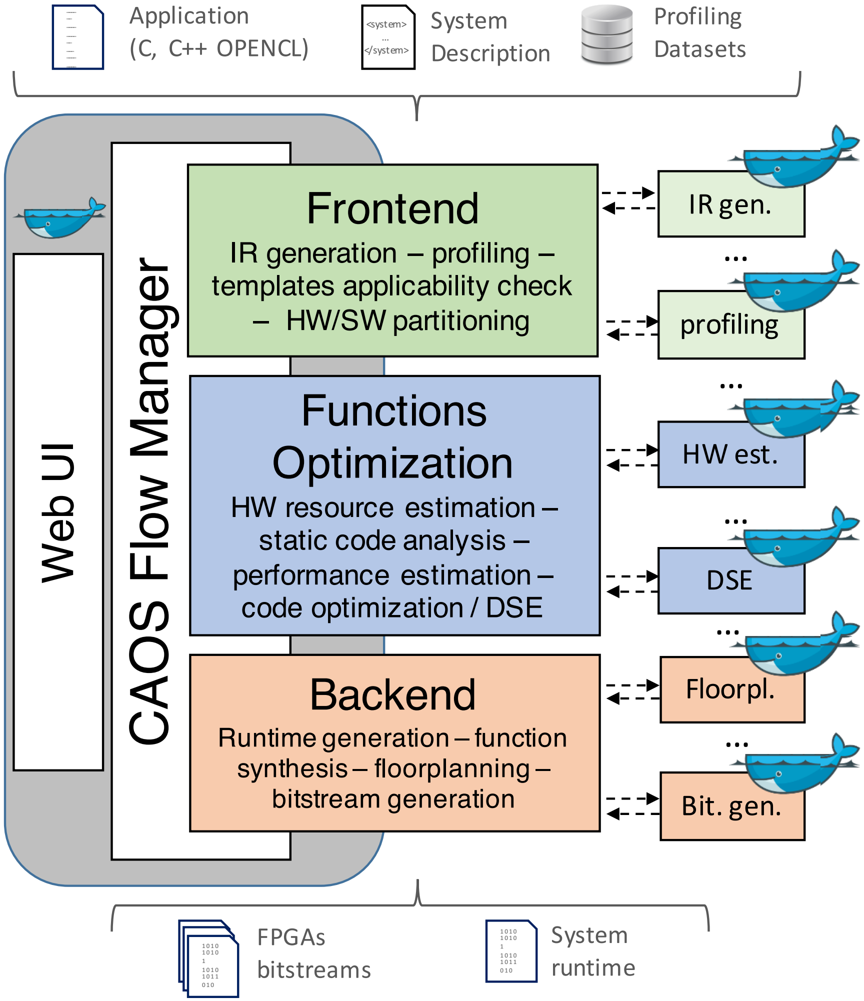
(SST)

(SDAccel Environment)

…

Architectural Templates

CAOS Frontend



Application (C, C++ OPENCL)

System Description

Profiling Datasets

Web UI

CAOS Flow Manager

Frontend

IR generation – profiling – templates applicability check – HW/SW partitioning

Functions Optimization

HW resource estimation – static code analysis – performance estimation – code optimization / DSE

Backend

Runtime generation – function synthesis – floorplanning – bitstream generation

IR gen.

profiling

HW est.

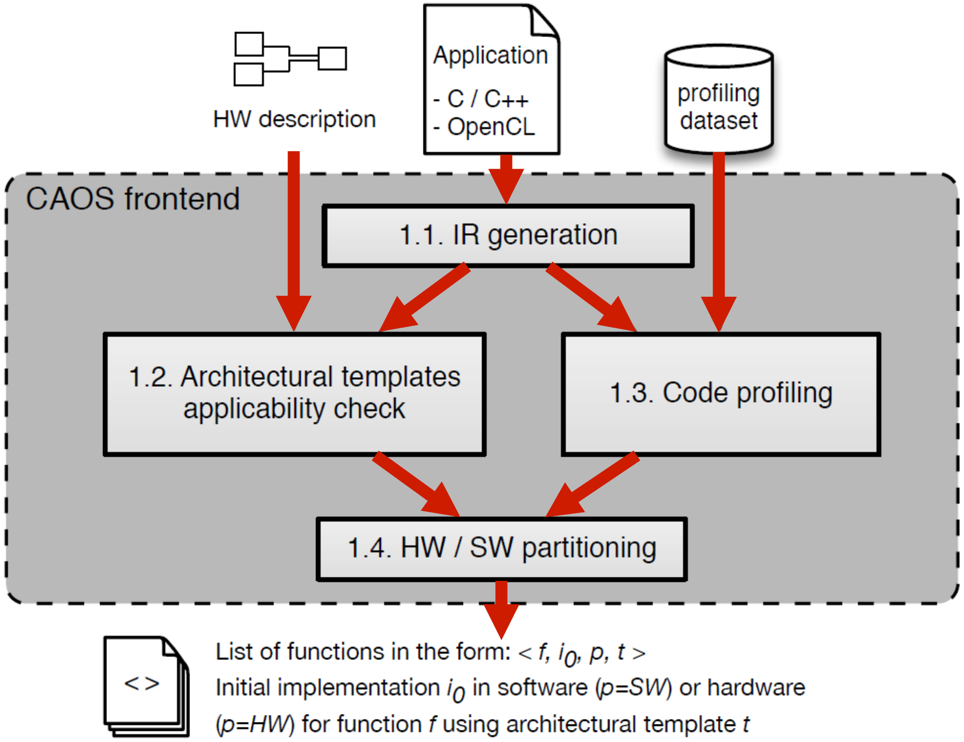
DSE

Floorpl.

Bit. gen.

FPGAs bitstreams

System runtime



HW description

Application

- C / C++
- OpenCL

profiling dataset

CAOS frontend

1.1. IR generation

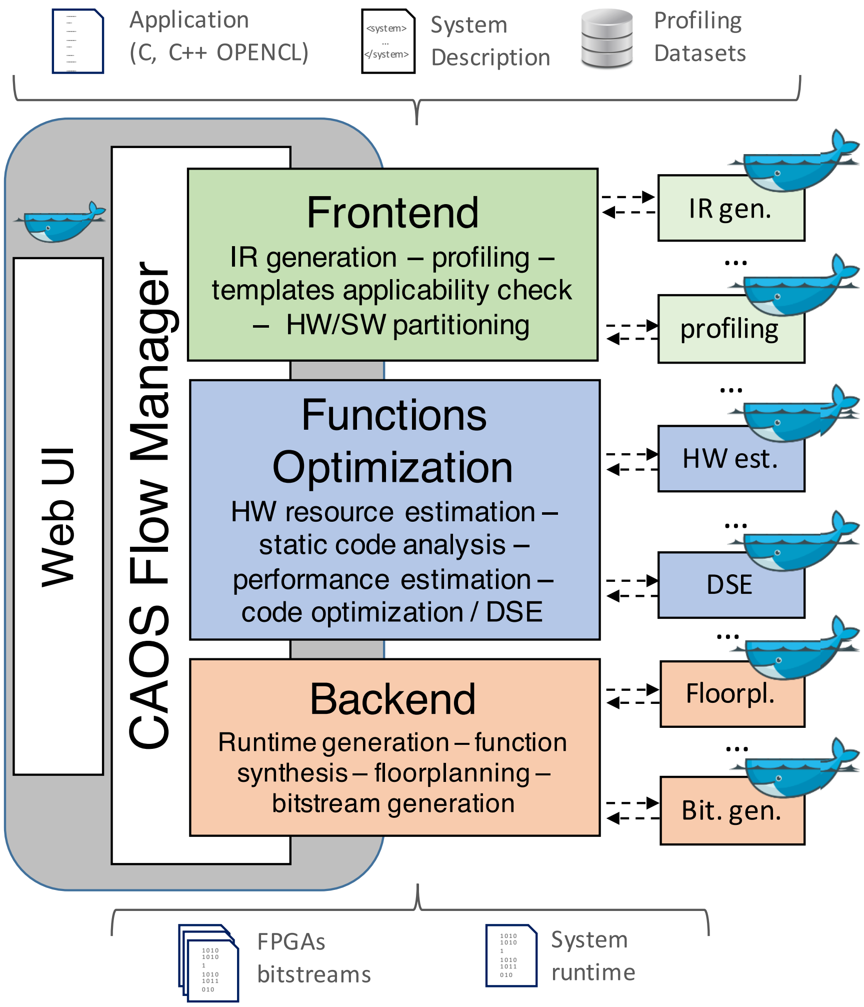
1.2. Architectural templates applicability check

1.3. Code profiling

1.4. HW / SW partitioning

List of functions in the form: $< f, i_0, p, t >$
Initial implementation i_0 in software ($p=SW$) or hardware ($p=HW$) for function *f* using architectural template *t*

CAOS Functions Optimization



Application (C, C++ OPENCL)

System Description

Profiling Datasets

Web UI

CAOS Flow Manager

Frontend

IR generation – profiling – templates applicability check – HW/SW partitioning

Functions Optimization

HW resource estimation – static code analysis – performance estimation – code optimization / DSE

Backend

Runtime generation – function synthesis – floorplanning – bitstream generation

IR gen.

…

profiling

…

HW est.

…

DSE

…

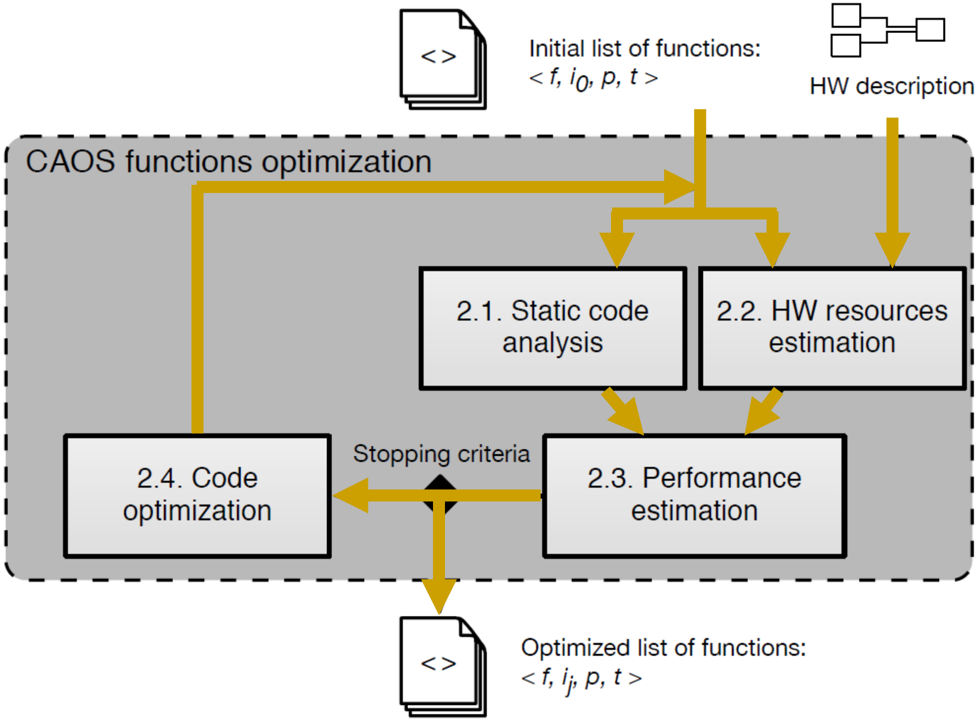
Floorpl.

…

Bit. gen.

FPGAs bitstreams

System runtime



Initial list of functions: $< f, i_0, p, t >$

HW description

CAOS functions optimization

2.1. Static code analysis

2.2. HW resources estimation

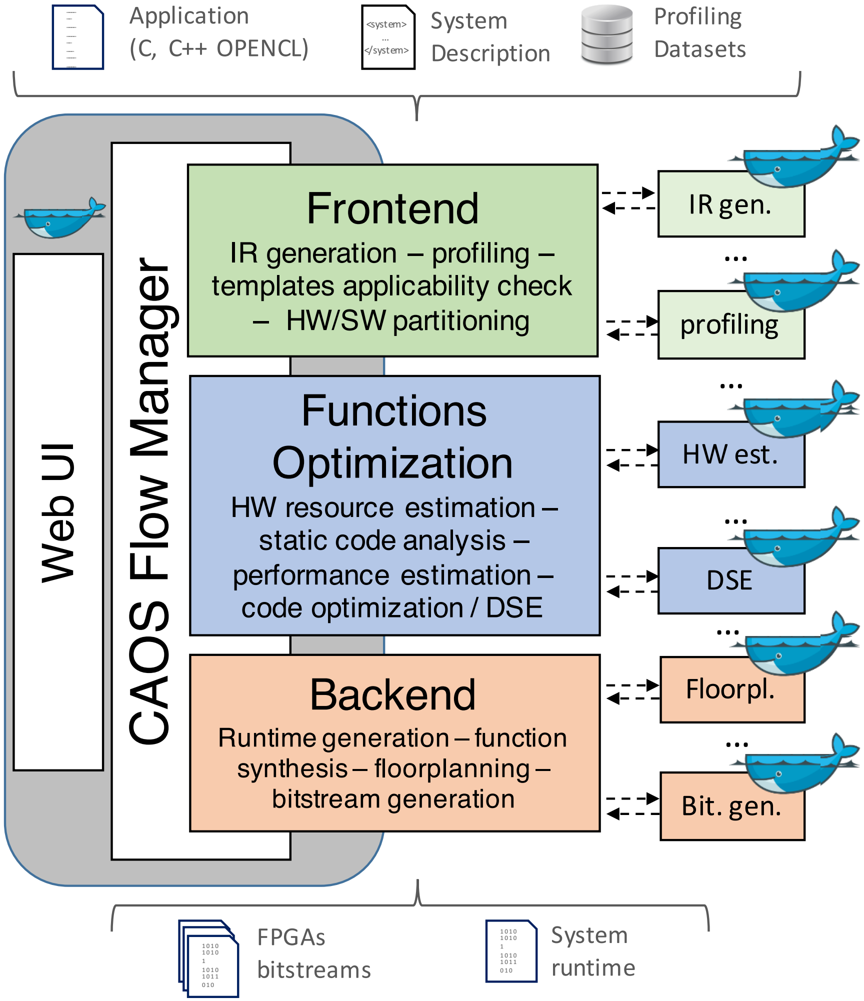
2.3. Performance estimation

2.4. Code optimization

Stopping criteria

Optimized list of functions: $< f, i_j, p, t >$

CAOS Backend



Application (C, C++ OPENCL)

System Description

Profiling Datasets

Web UI

CAOS Flow Manager

Frontend

IR generation – profiling – templates applicability check – HW/SW partitioning

Functions Optimization

HW resource estimation – static code analysis – performance estimation – code optimization / DSE

Backend

Runtime generation – function synthesis – floorplanning – bitstream generation

IR gen.

…

profiling

…

HW est.

…

DSE

…

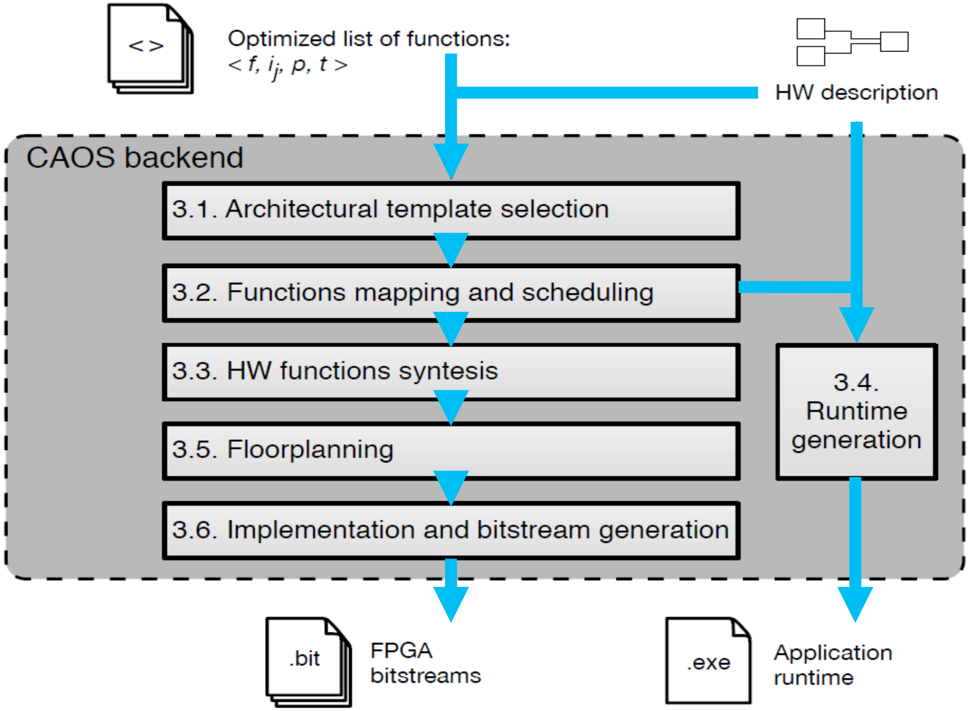
Floorpl.

…

Bit. gen.

FPGAs bitstreams

System runtime



Optimized list of functions: $< f, i_j, p, t >$

HW description

CAOS backend

3.1. Architectural template selection

3.2. Functions mapping and scheduling

3.3. HW functions syntesis

3.4. Runtime generation

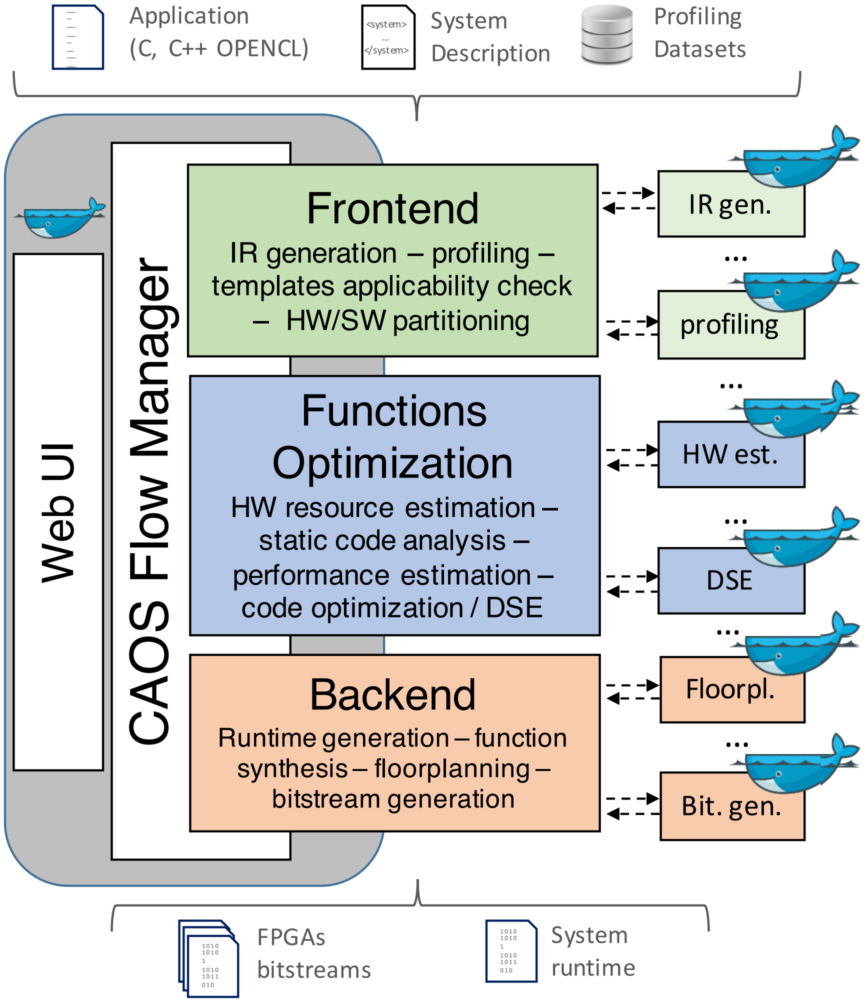
3.5. Floorplanning

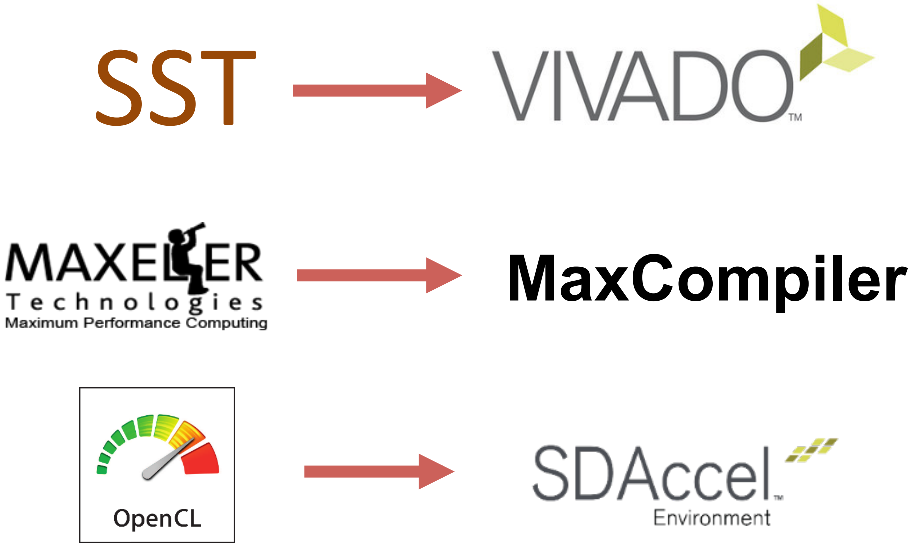
3.6. Implementation and bitstream generation

FPGA bitstreams

Application runtime

CAOS Backend

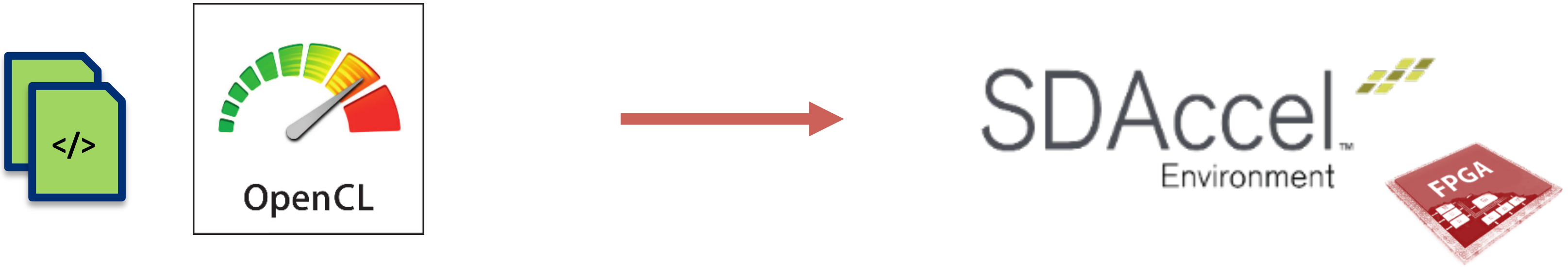

Application
(C, C++ OPENCL)
<system>
...
</system>
System
Description
Profiling
Datasets
Web UI
CAOS Flow Manager
Frontend
IR generation – profiling – templates applicability check – HW/SW partitioning
Functions Optimization
HW resource estimation – static code analysis – performance estimation – code optimization / DSE
Backend
Runtime generation – function synthesis – floorplanning – bitstream generation
IR gen.
...
profiling
...
HW est.
...
DSE
...
Floorpl.
...
Bit. gen.
FPGAs
bitstreams
System
runtime



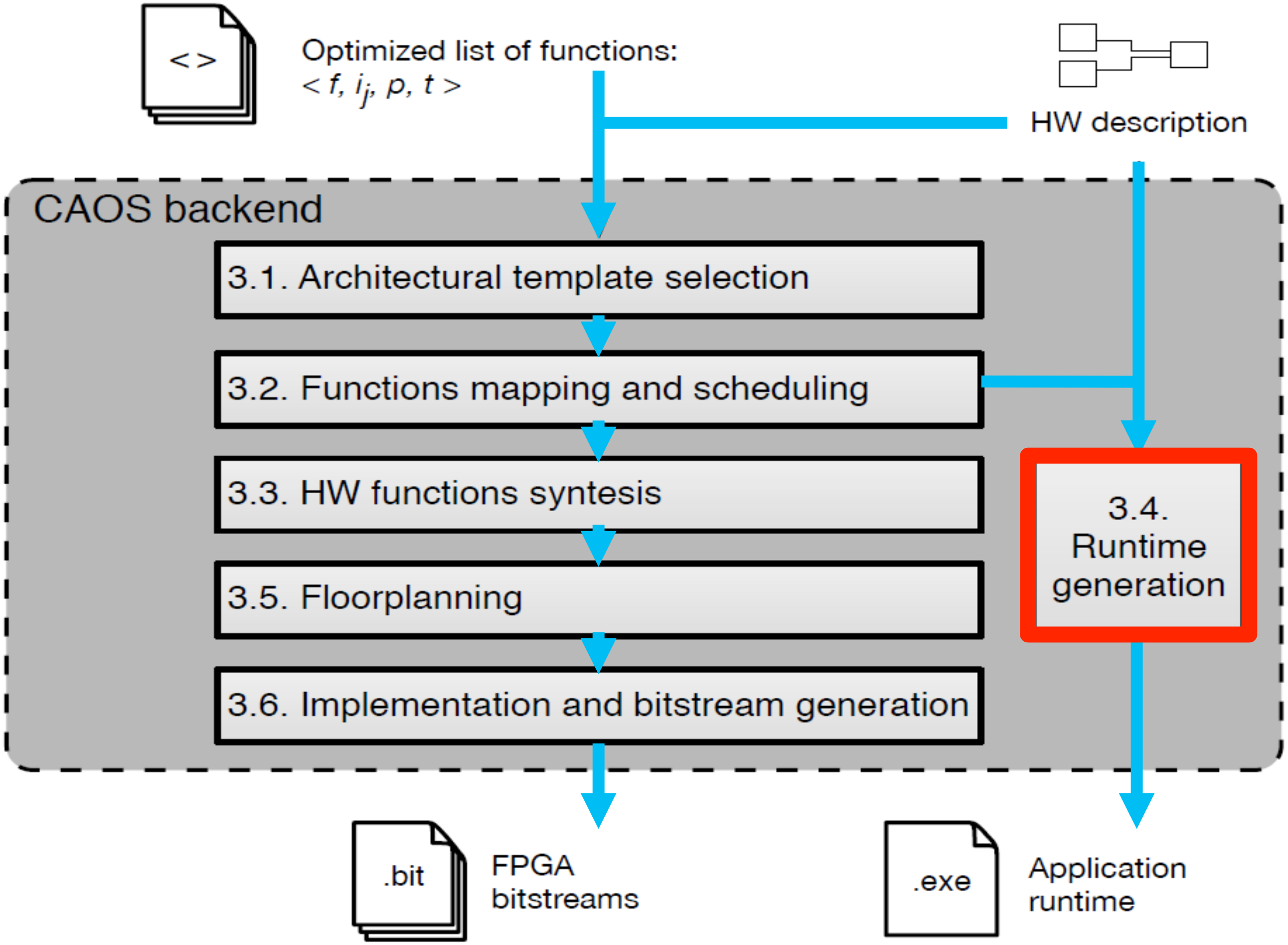
Output Generation

CAOS: OpenCL and SDAccel

- CAOS Frontend supports OpenCL code:
  - Intermediate representation support
  - Template applicability check for SDA
  - Code profiling through LTPV (OpenCL profiler)
  - Function optimization:
    - Static code analysis and HW resource estimation within SDA
  - Backend support for SDAccell



CAOS Backend for SDAccel


< >
Optimized list of functions:
$< f, i_j, p, t >$
HW description
CAOS backend
3.1. Architectural template selection
3.2. Functions mapping and scheduling
3.3. HW functions syntesis
3.5. Floorplanning
3.6. Implementation and bitstream generation
3.4. Runtime generation
.bit
FPGA bitstreams
.exe
Application runtime

SDAccel generates & provides:

- **XCLBIN** containing the bitstream
- **OpenCL Runtime** to manage kernel execution

CAOS Integrates SDAccel:

- Identifying I/O Variables
- Generating a specific OpenCL Host code for the application

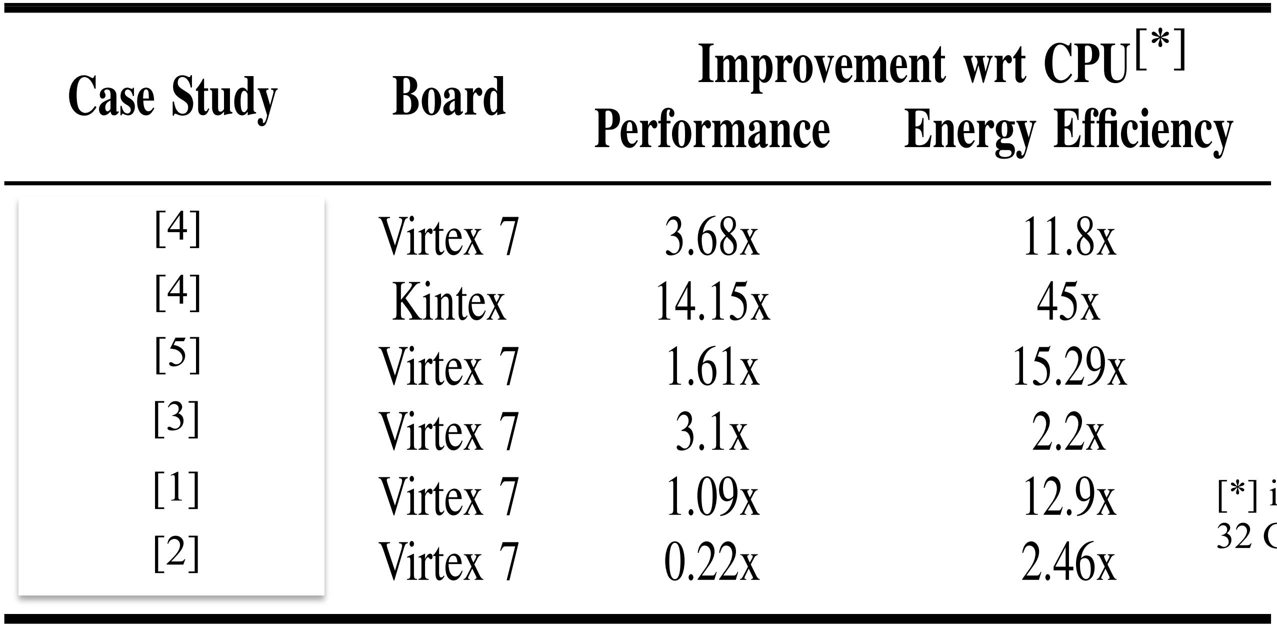
Evaluations: Evaluations

[1, 2] Streaming Stencil Time-step (SST)

[3] Pearson Correlation Coefficient, Asian Option Pricing

[5] Protein Folding

[4] Smith Waterman and Vessels Segmentation

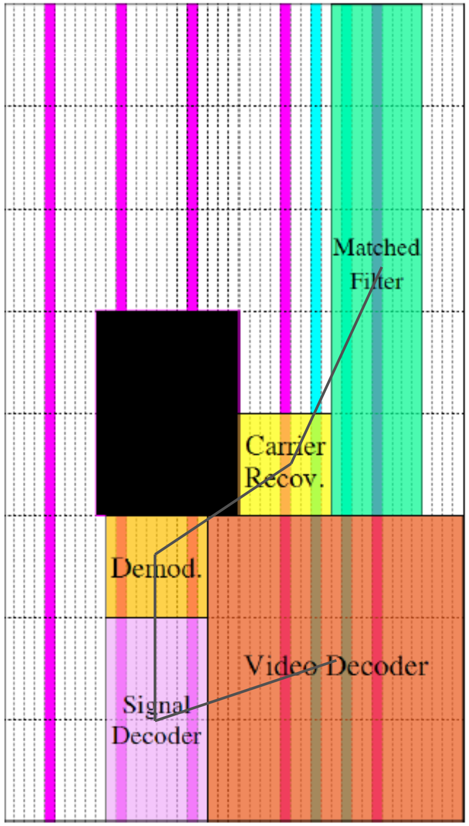


Case Study	Board	Improvement wrt CPU[*]	
		Performance	Energy Efficiency
[4]	Virtex 7	3.68x	11.8x
[4]	Kintex	14.15x	45x
[5]	Virtex 7	1.61x	15.29x
[3]	Virtex 7	3.1x	2.2x
[1]	Virtex 7	1.09x	12.9x
[2]	Virtex 7	0.22x	2.46x

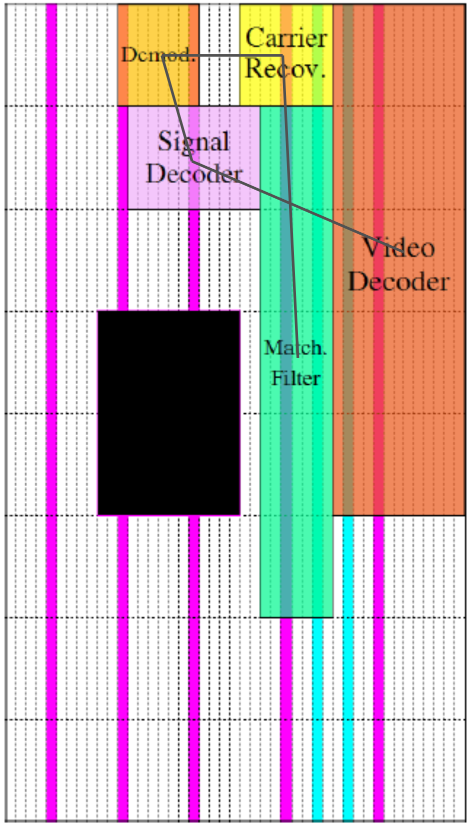
[*] intel Xeon E5 1410
32 GB RAM

RATIONALE BEHIND CAOS
A PRACTICAL EXAMPLE

Hints on the problem…



Floorplai produced by [*].

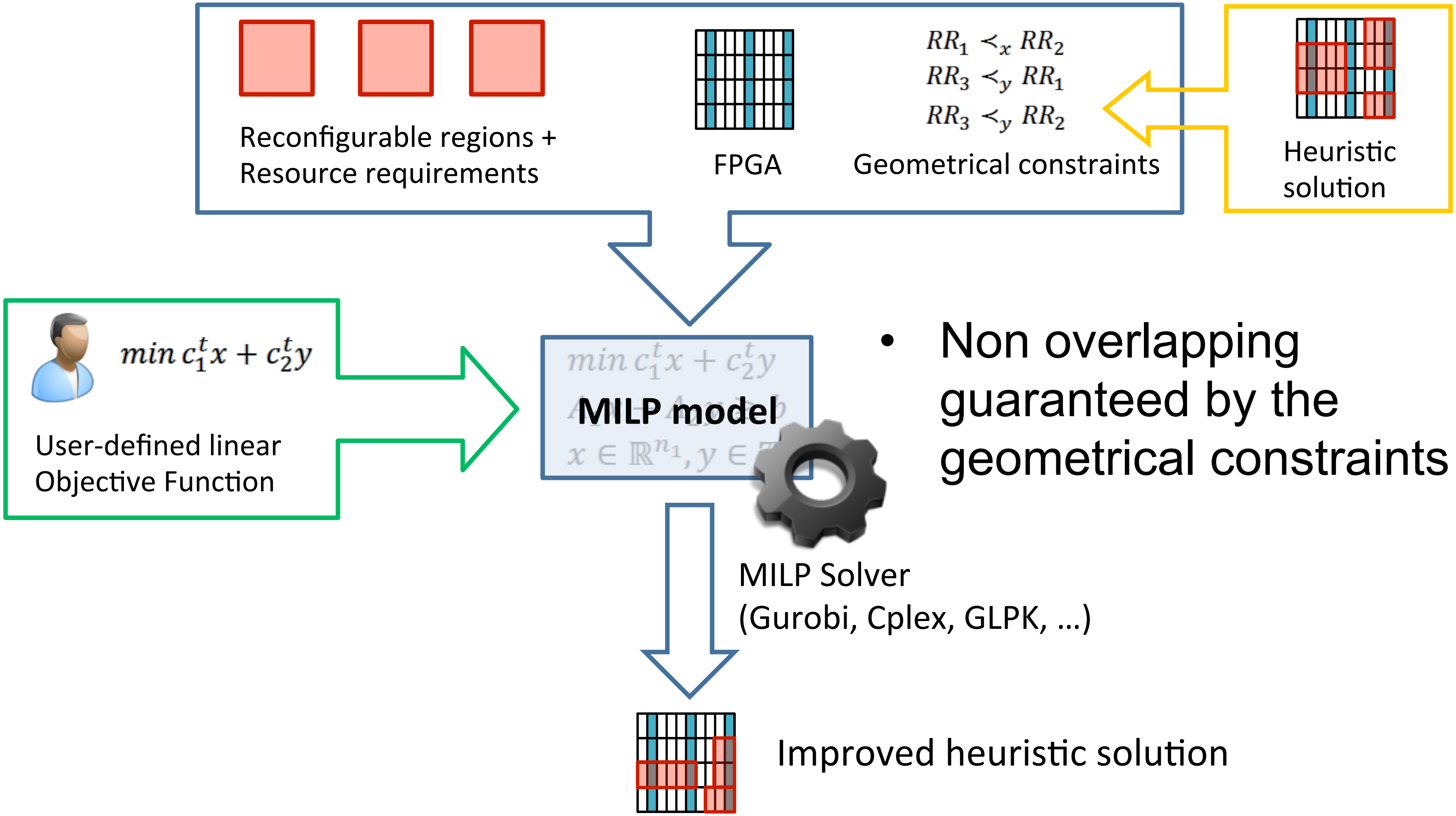


Floorplan produced by OF.

Which one is the best?

[*] Vipin, K. and Fahmy, S. A.: Architecture-aware reconfiguration-centric floorplanning for partial reconfiguration. In ARC, pages 13-25, 2012.

Heuristic-Optimal Floorplanner



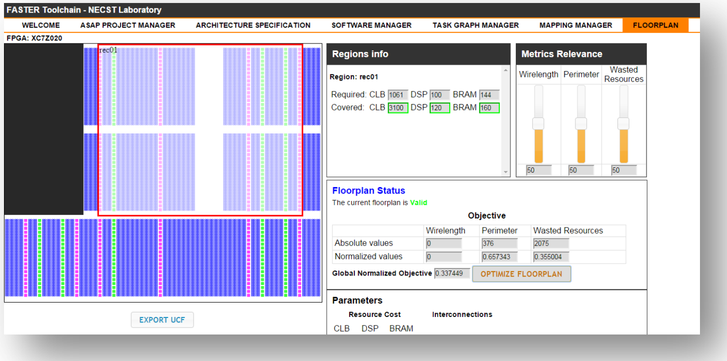
- Non overlapping guaranteed by the geometrical constraints

Objective function

- Cost function can be defined starting from the variables and parameters of the MILP model

- Implemented metrics:
  - Global wirelength measured using HPWL (WL_{cost})
  - Regions perimeter (P_{cost})
  - Wasted resources (R_{cost})

$$min\left\{q_1 \cdot \frac{WL_{cost}}{WL_{max}} + q_2 \cdot \frac{P_{cost}}{P_{max}} + q_3 \cdot \frac{R_{cost}}{R_{max}}\right\}$$



FPGA: XC7Z020

Regions info

Region: rec01

Required: CLB 1061 DSP 100 BRAM 144

Covered: CLB 3100 DSP 120 BRAM 160

Metrics Relevance

Wirelength	Perimeter	Wasted Resources
50	50	50

Floorplan Status

The current floorplan is Valid

Objective

	Wirelength	Perimeter	Wasted Resources
Absolute values	0	376	2075
Normalized values	0	0.657343	0.355004

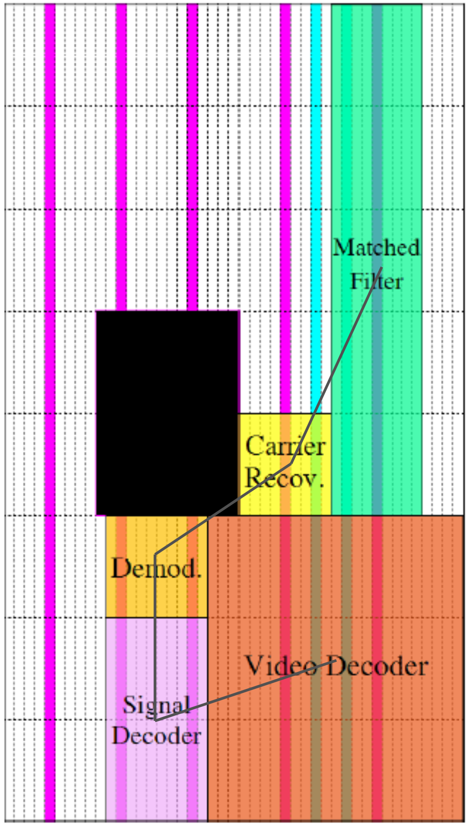
Global Normalized Objective 0.337449

Parameters

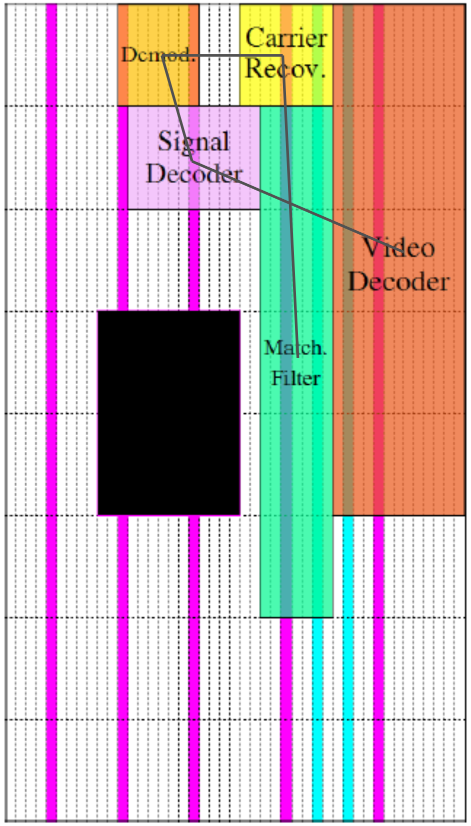
Resource Cost Interconnections

CLB DSP BRAM

Hints on the problem…



Floorplaı produced by [*].

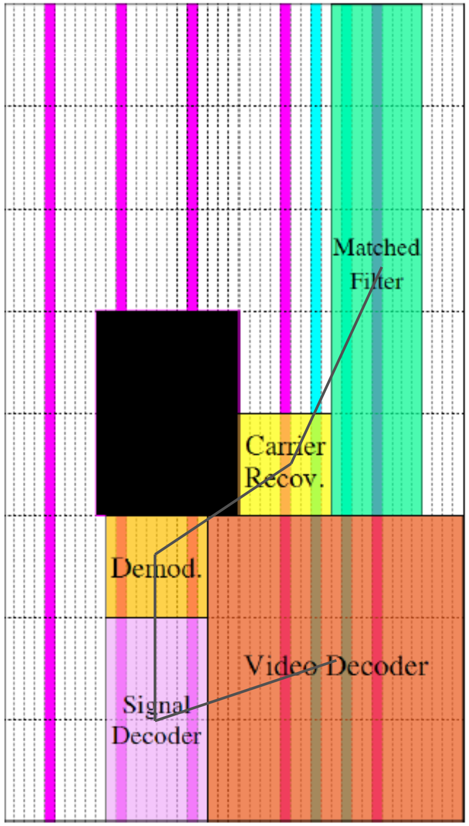


Floorplan produced by OF.

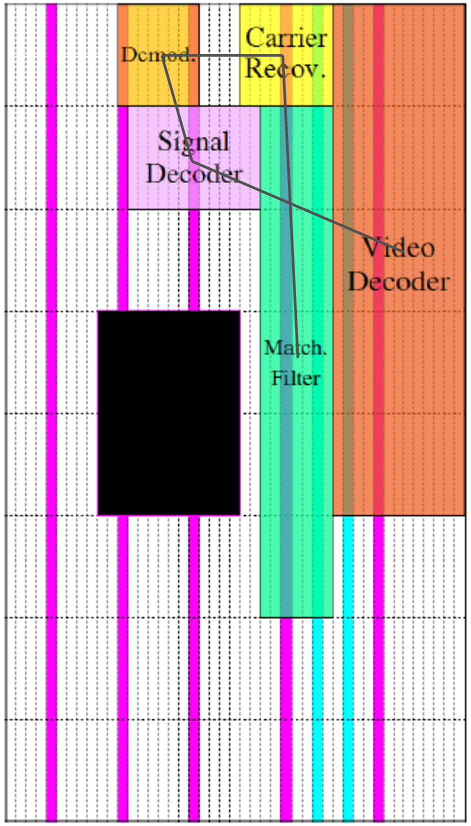
Which one is the best?

[*] Vipin, K. and Fahmy, S. A.: Architecture-aware reconfiguration-centric floorplanning for partial reconfiguration. In ARC, pages 13-25, 2012.

Hints on the problem…



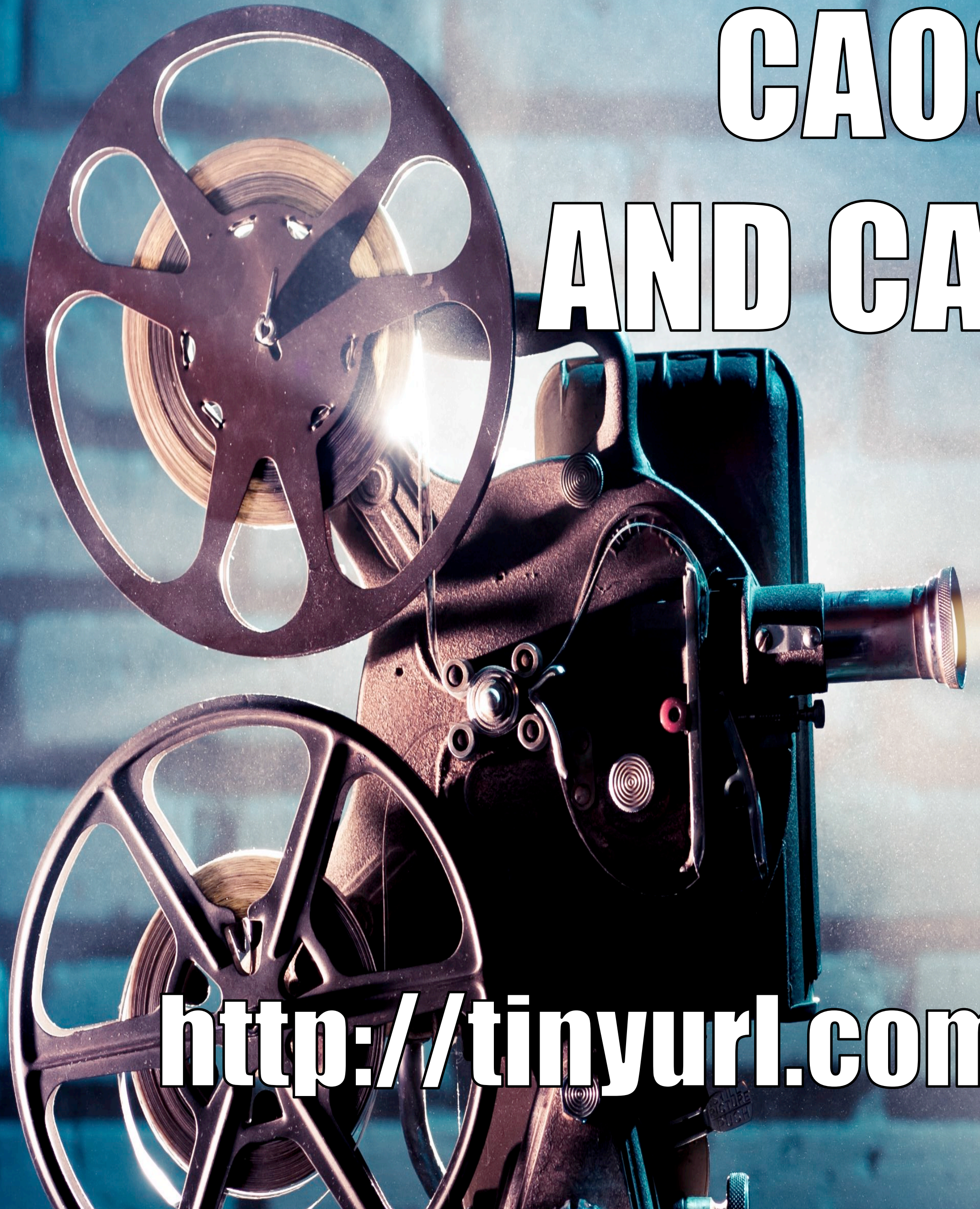
Floorplai produced by [*].



Floorplan produced by OF.

- Optimal solution in 29s
- 34% wasted frames reduction
  - No DSP and CLB wasted by the Video Decoder RR
  - No BRAM wasted by the Signal Decoder RR
- Approximately same wirelength

[*] Vipin, K. and Fahmy, S. A.: Architecture-aware reconfiguration-centric floorplanning for partial reconfiguration. In ARC, pages 13-25, 2012.



CAOS DEMOS AND CASE STUDIES

YouTube

http://tinyurl.com/CAOS-ICCD17

WHO IS DRIVING THE CAR?

(A.K.A) WHAT ABOUT THE RUNTIME?

Runtime Reconfiguration Management

- Reconfigurable architecture
  - Static Area: used to control the reconfiguration process
  - Reconfigurable Area: used to swap at runtime different cores
  - Reconfiguration- oriented communication infrastructure

- Runtime reconfiguration managed via SW
  - Standalone, Operating System
    - Increased portability of user applications
    - Inherited multitasking capabilities
    - Simplified software development process

- Bitstreams relocation technique to
  - speedup the overall system execution
  - achieve a core preemptive execution
  - assign at runtime the bitstreams placement
  - reduce the amount of memory used to store partial bitstreams

Runtime Reconfiguration Management

- Reconfigurable architecture
  - Static Area: used to control the reconfiguration process
  - Reconfigurable Area: used to swap at runtime different cores
  - Reconfiguration- oriented communication infrastructure

- Runtime reconfiguration managed via SW
  - Standalone, Operating System
    - Increased portability of user applications
    - Inherited multitasking capabilities
    - Simplified software development process

- Bitstreams relocation technique to
  - speedup the overall system execution
  - achieve a core preemptive execution
  - assign at runtime the bitstreams placement
  - reduce the amount of memory used to store partial bitstreams

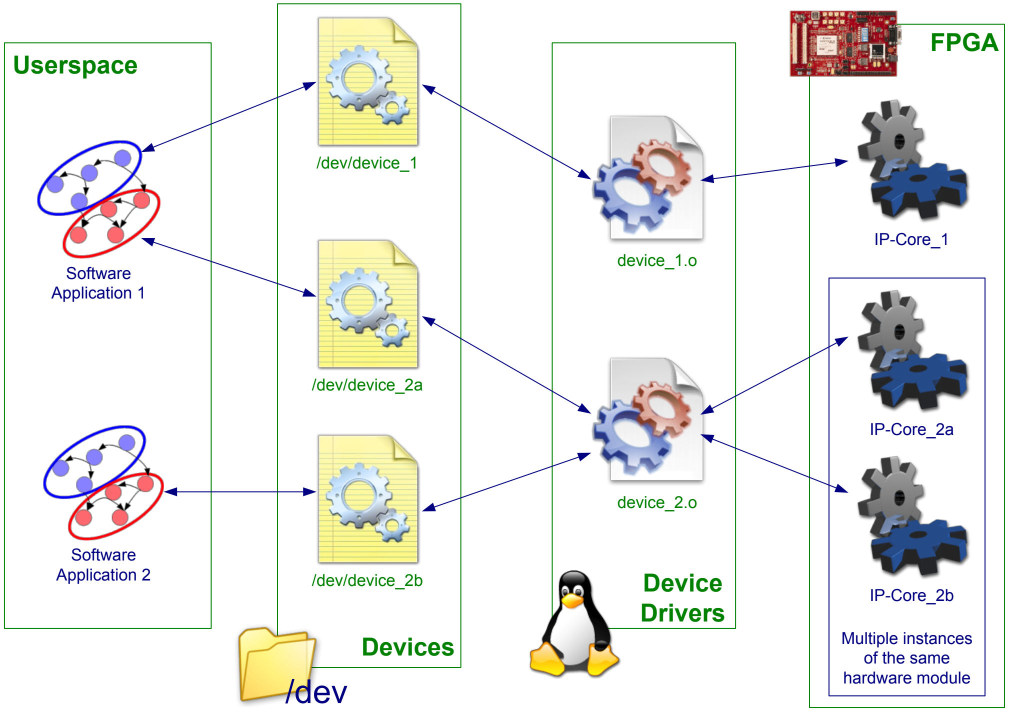
OS-based management of dynamic reconfiguration

- Provide software support for dynamic partial reconfiguration on Systems-on-Chip running an operating system (i.e., LINUX).
  - OS customization for specific architectures
  - Rec. Functional Unit caching policies to improve the performance
  - Partial reconfiguration process management from the OS
  - Addition and removal of reconfigurable components
  - Automatic loading and unloading of specific drivers for the IP-Cores upon components configuration and/or deconfiguration

- Hardware-independent interface for software developers based on the GNU/Linux
- Easier programming interface for specific drivers

IP-Core Devices Access

- Interaction with configured IP-Cores implemented by means of the standard Linux device access
  - *Open, Close, Read, write, ioctl* operations

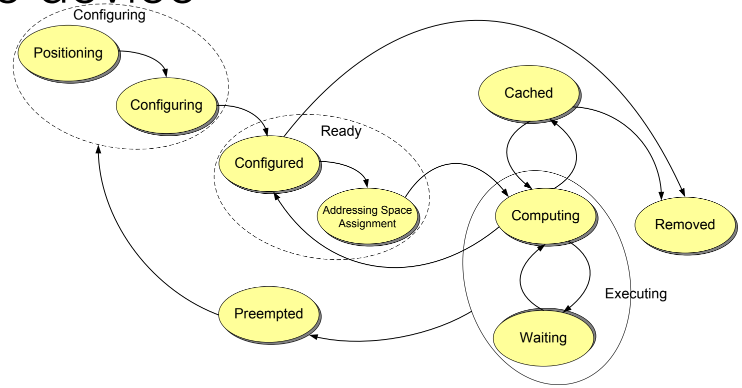


Reconfigurable Process Control Block

- Reconfigurable process: an RFU object code in execution

- Each reconfigurable process is represented in the system by a Reconfigurable Process Control Block (RPCB).

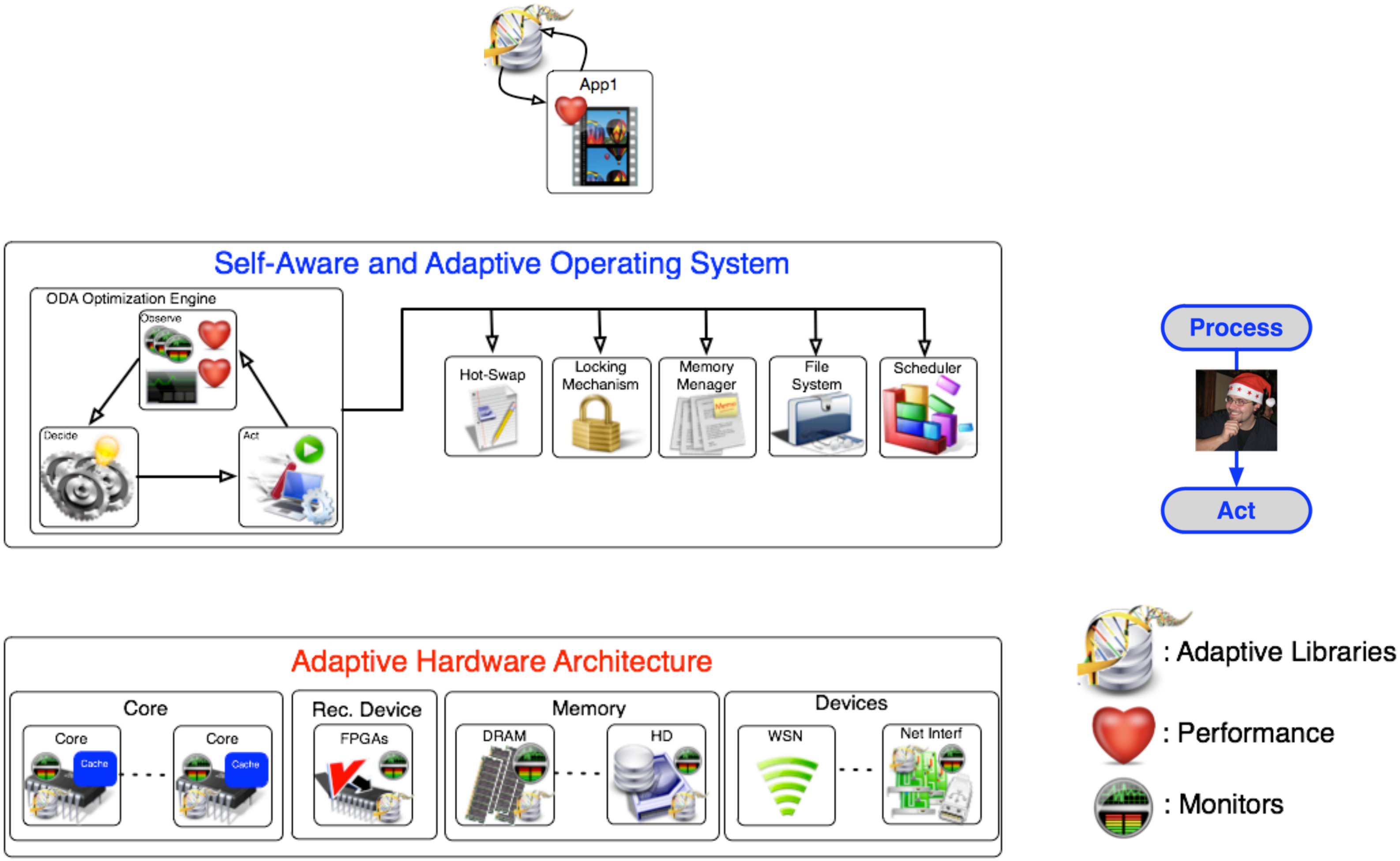
- A RPCB contains all the information associated with a specific reconfigurable process
  - State: the state in which the reconfigurable process control is at the current time
  - Position: the placement position on the device
  - Object Code Accounting Information:
    - Object Code
    - Configuration Priority
    - Resources
    - Position



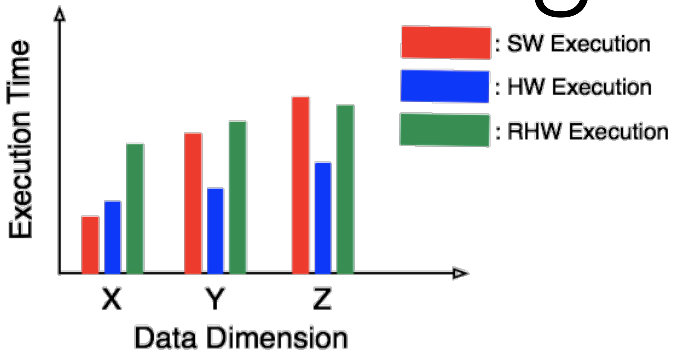
The Centralized Manager

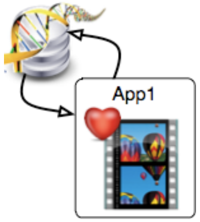
- Userspace applications are not allowed to explicitly request a bitstream
  - They request high-level functionalities
- Userspace requests are collected and served by a centralized manager (Linux Reconfiguration Manager)
  - The OS chooses the configuration code
  - A new reconfigurable process is created
- Only the LRM can ask for a bitstream to be configured on the FPGA
  - Centralized knowledge of the device status
  - Area management and module caching

Reconfiguration: Online Static

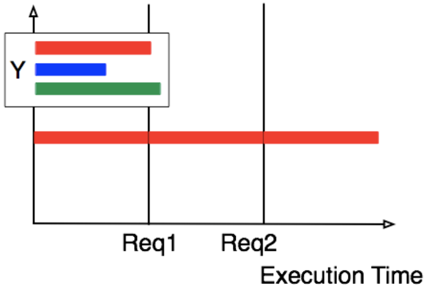


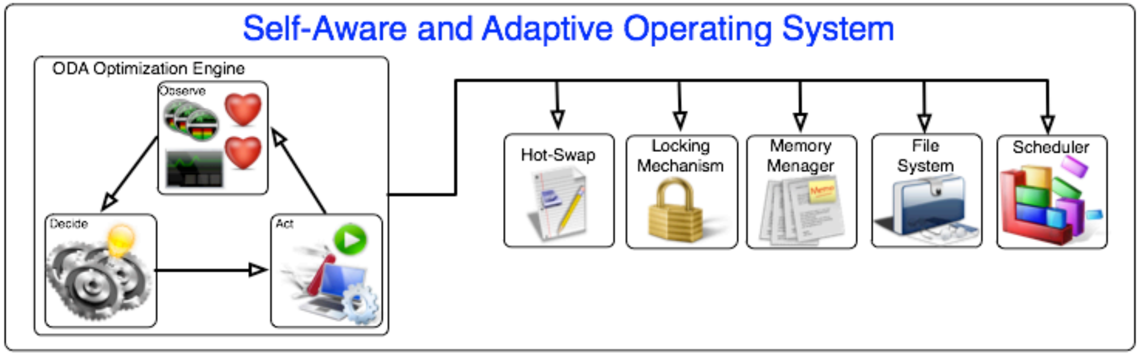
Reconfiguration: Online Static



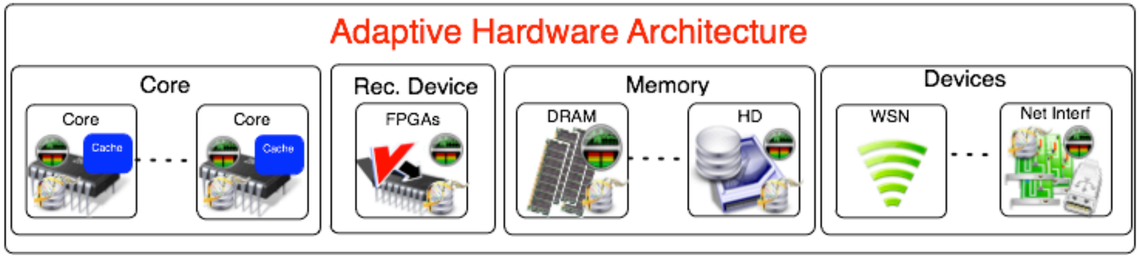


3 requests on Y data (NO HW configured)





Process
Act

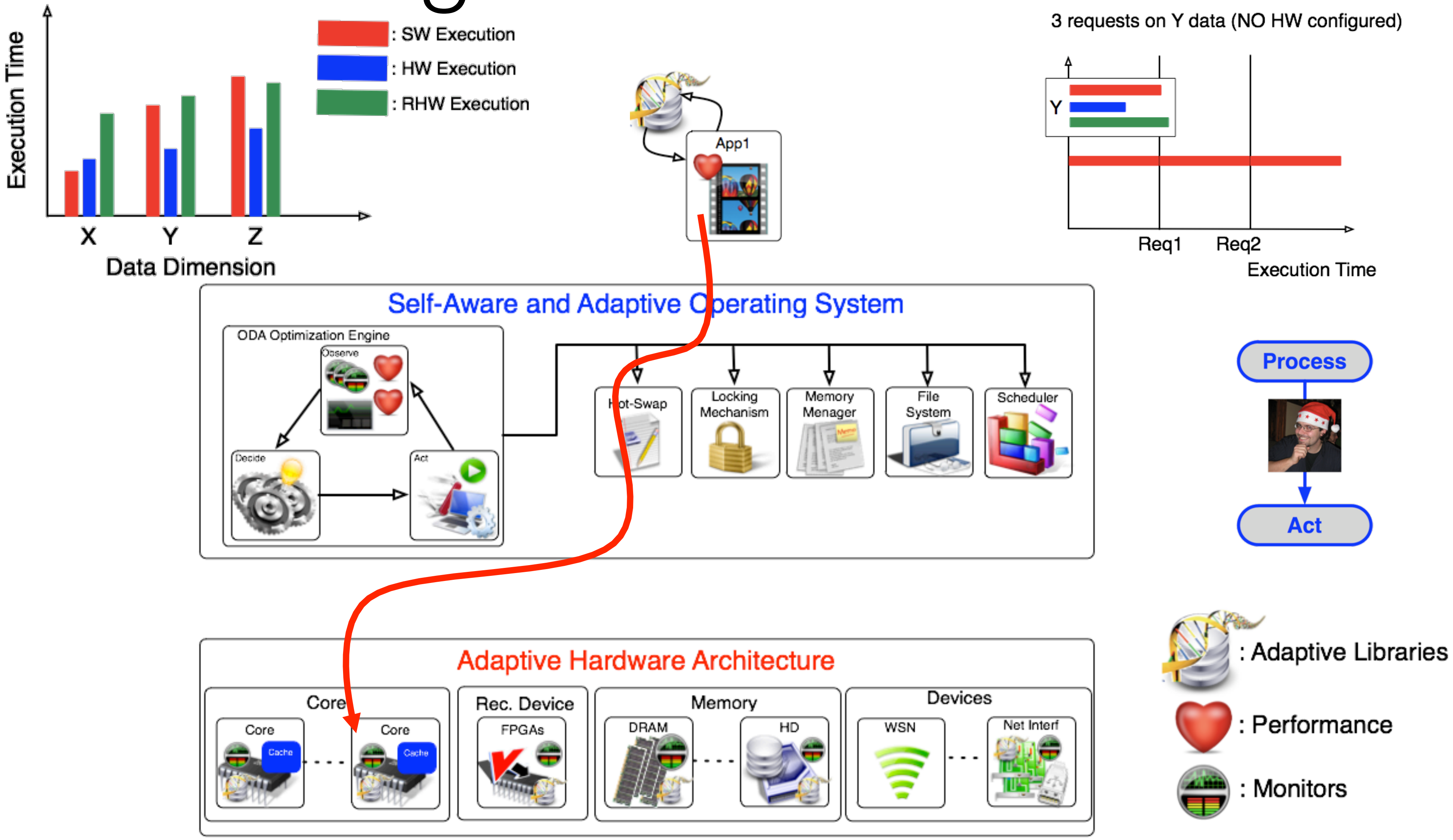


: Adaptive Libraries

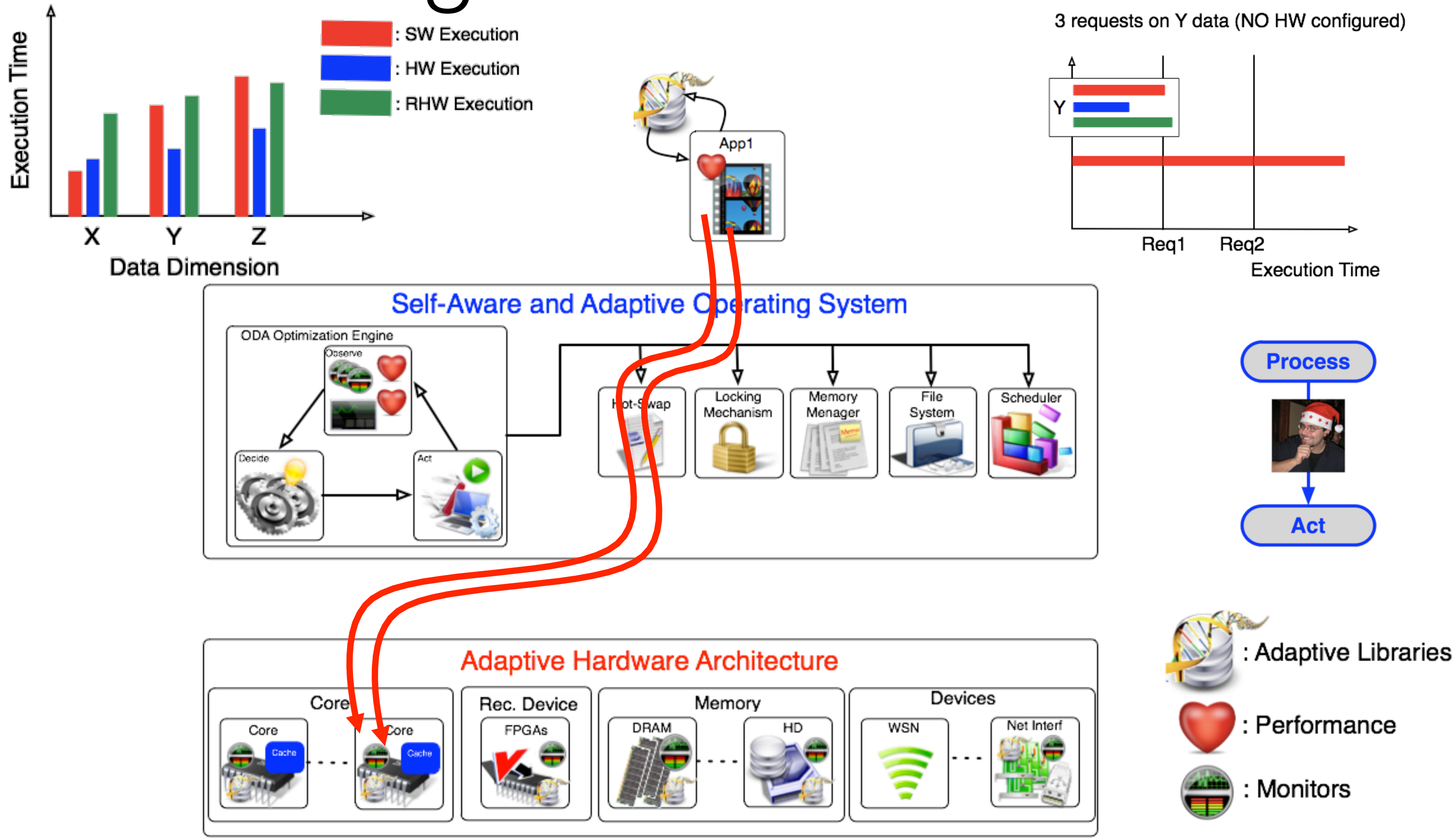
: Performance

: Monitors

Reconfiguration: Online Static

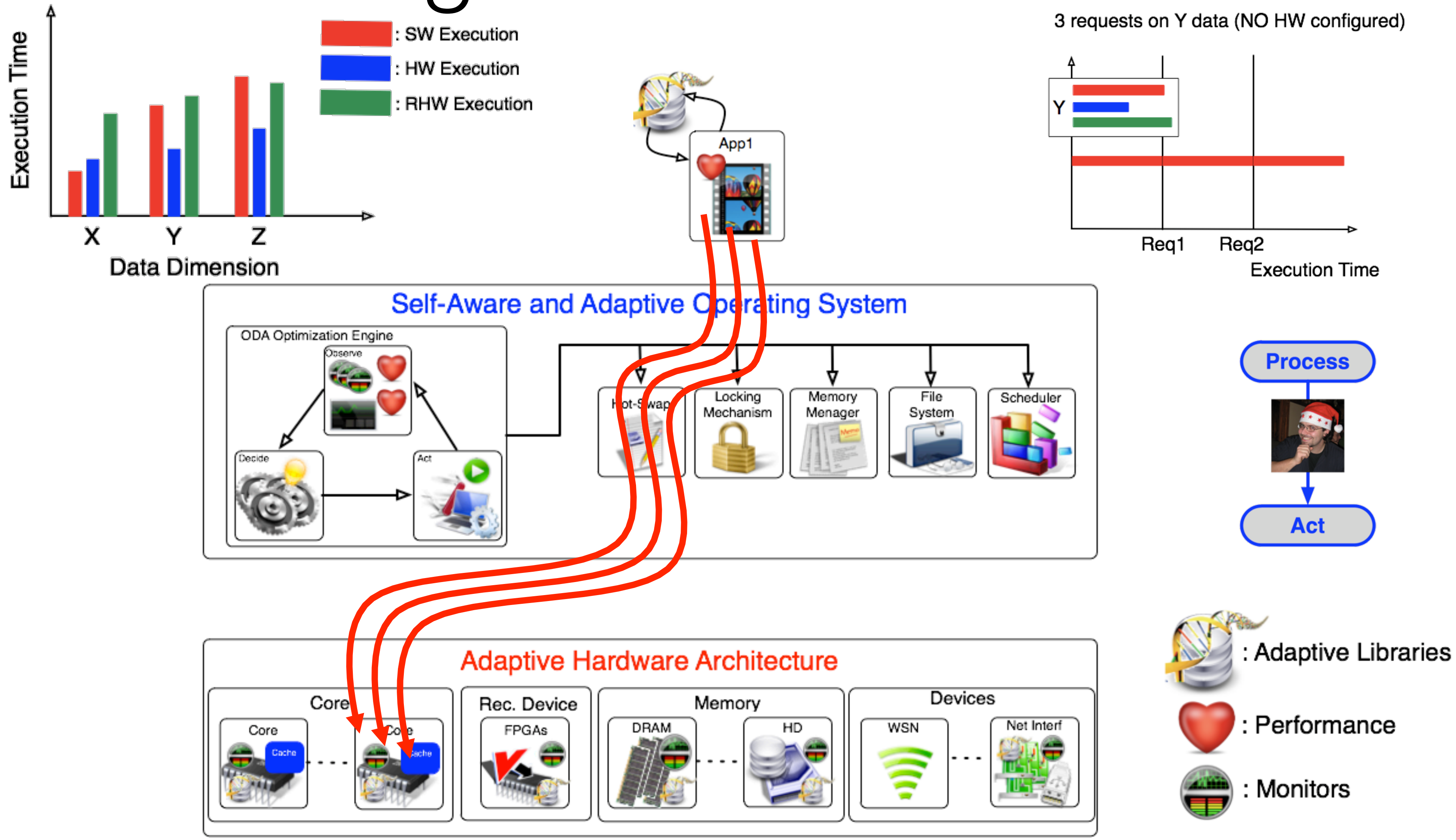


Reconfiguration: Online Static

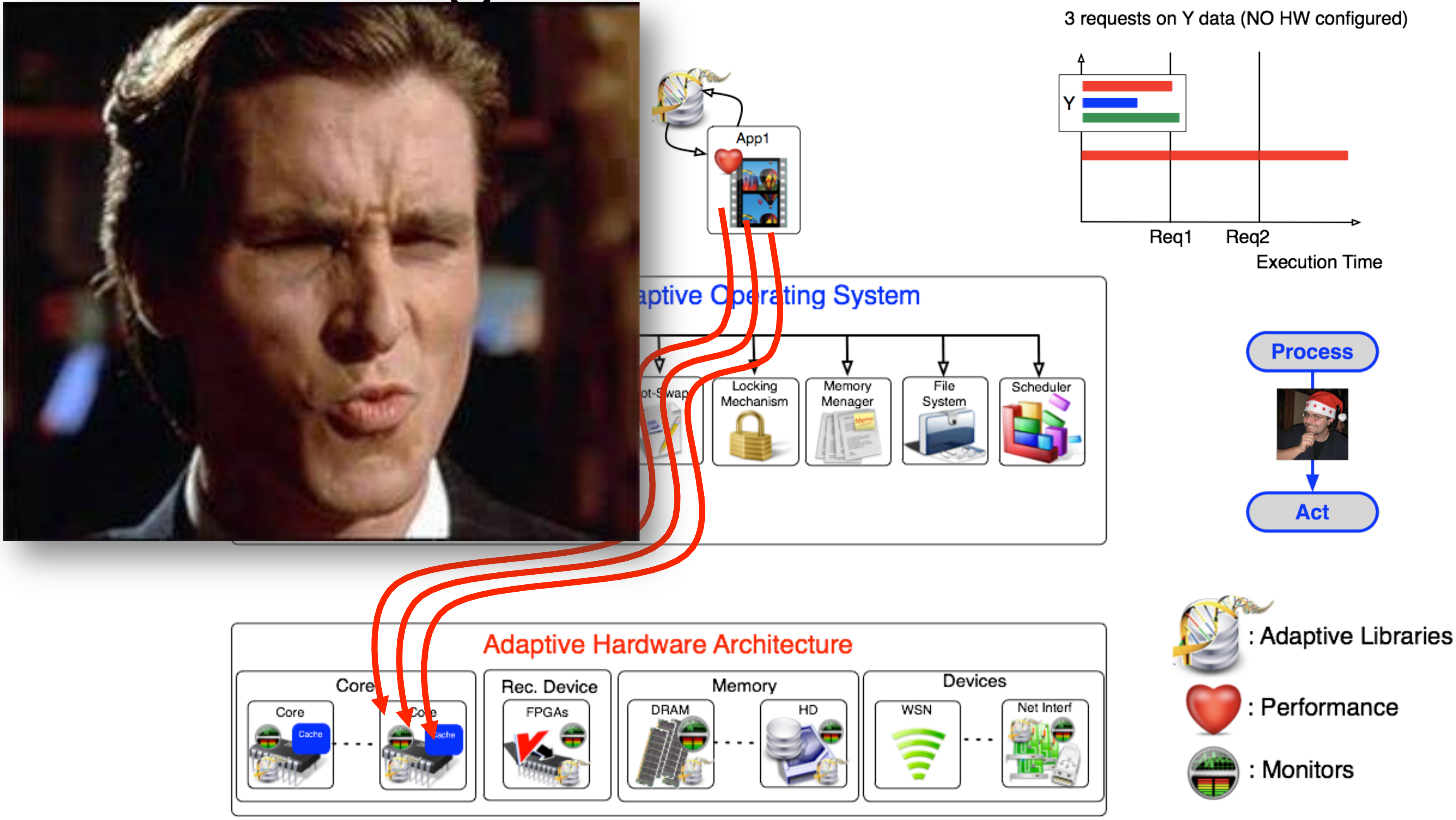


3 requests on Y data (NO HW configured)

Reconfiguration: Online Static



Reconfiguration: Online Static



Where to go necst?

Trying to raise the bar

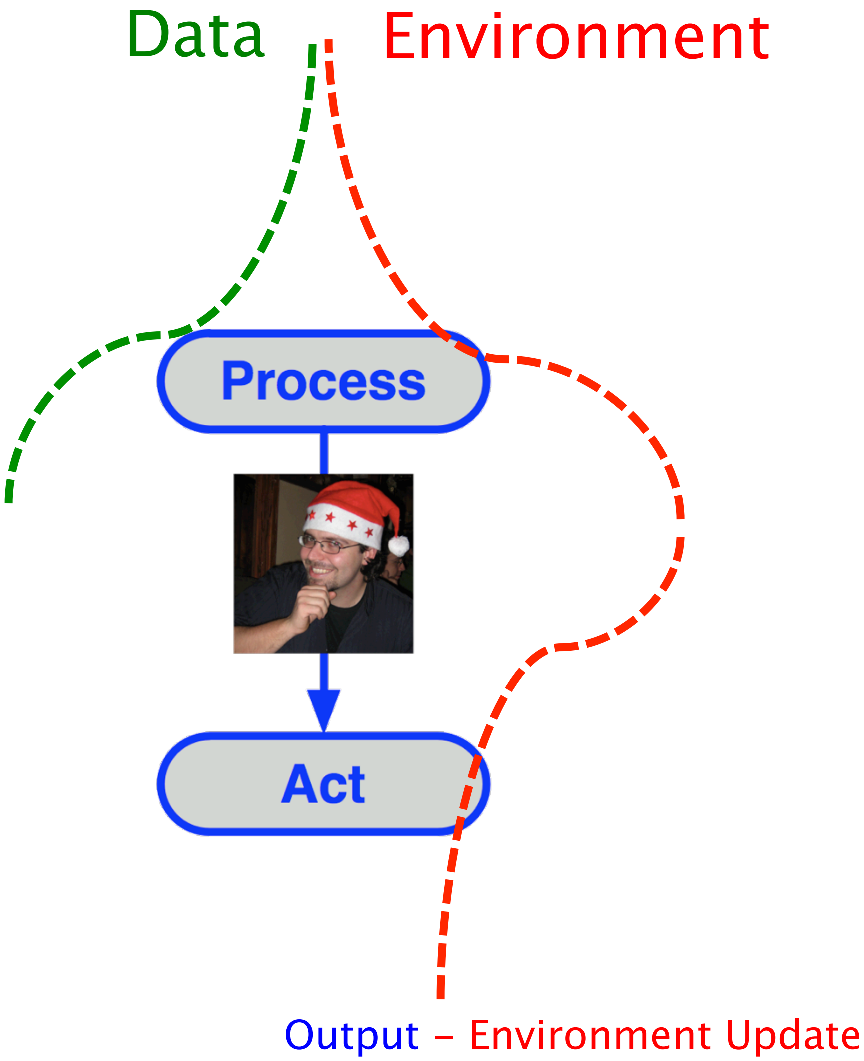
- Towards the design and implementaion of Self-adaptive and autonomic systems

Trying to raise the bar

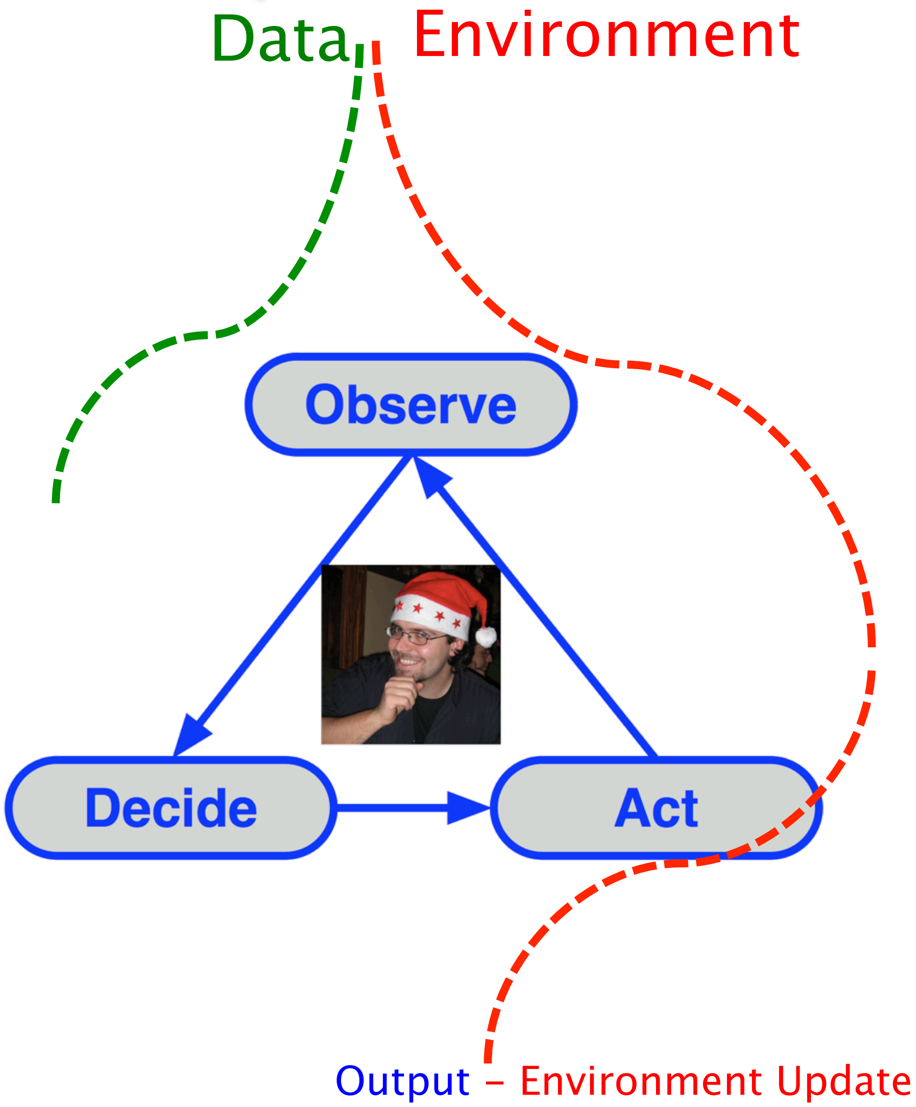
- Towards the design and implemententaion of Self-adaptive and autonomic systems

- We need to include 2 more features
  - Goal
    - Te
    - Sy
  - Appr
    - Do essary to meet goals

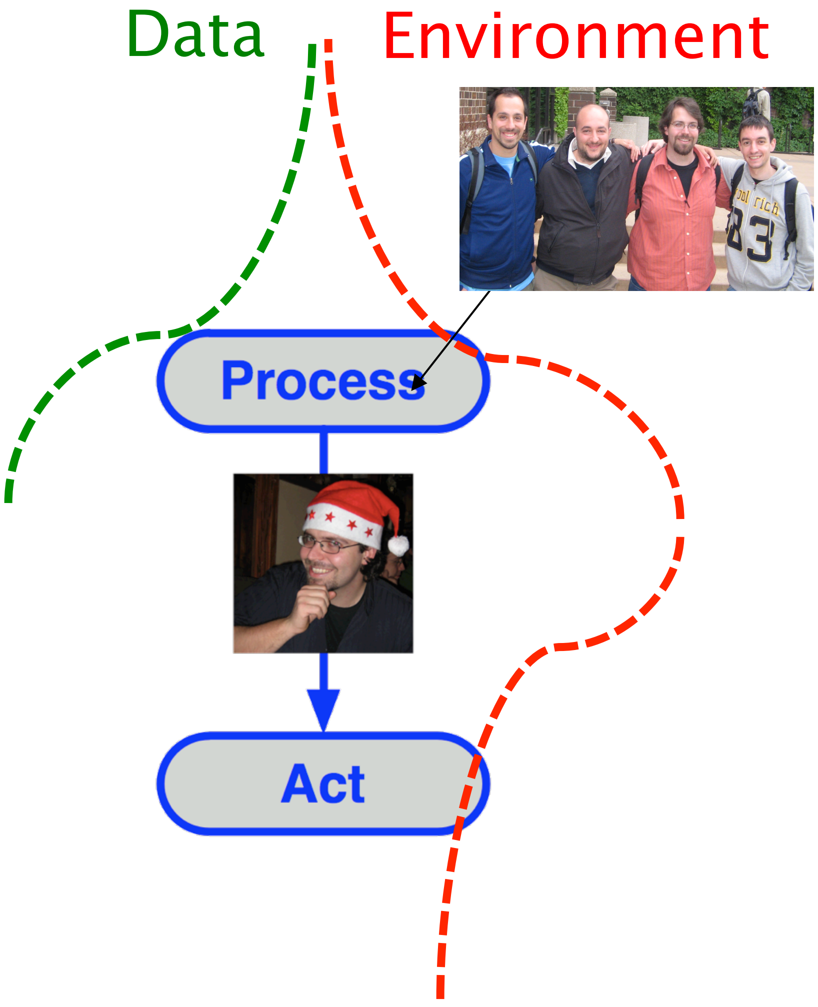
Online Static Solution



Adaptive Solution



Online Static Solution



Data

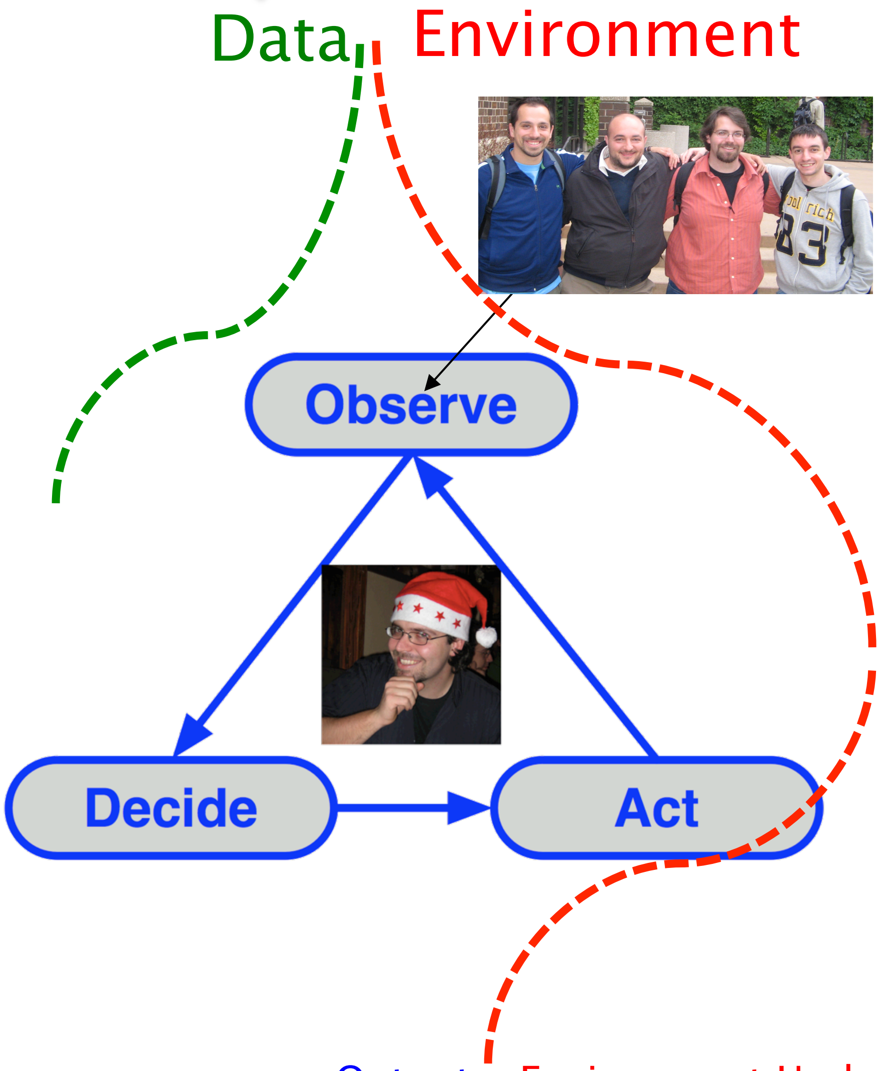
Environment

Process

Act

Output – Environment Update

Adaptive Solution



Data

Environment

Observe

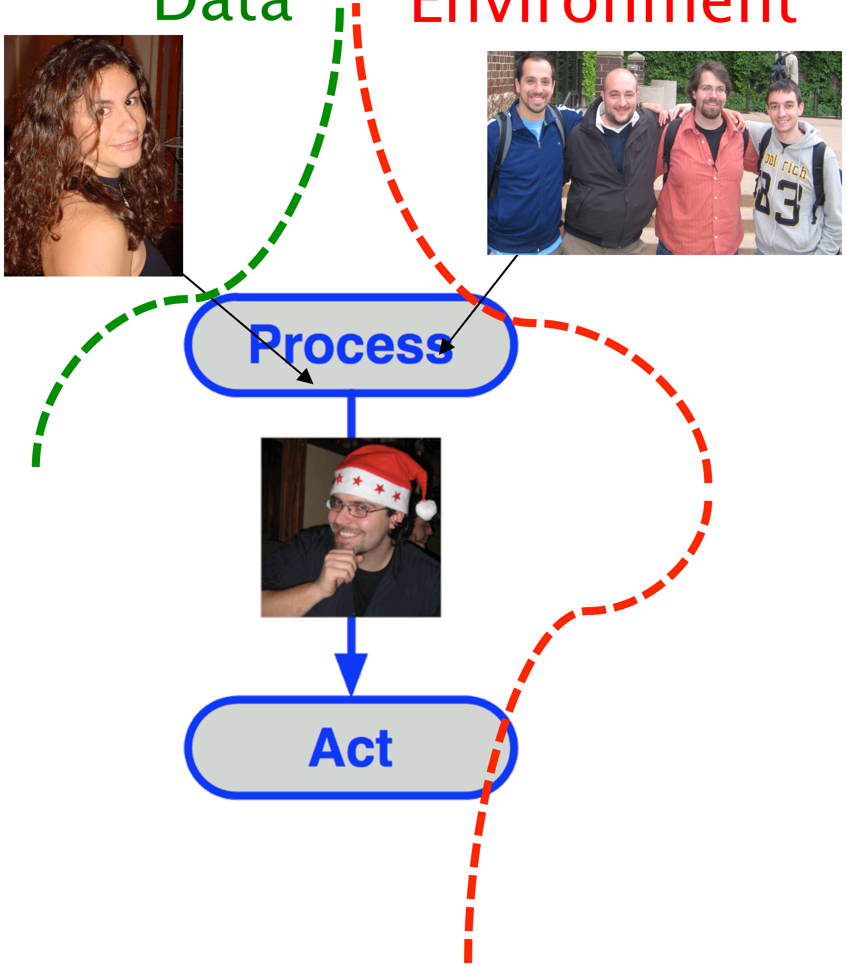
Decide

Act

Output – Environment Update

Online Static Solution

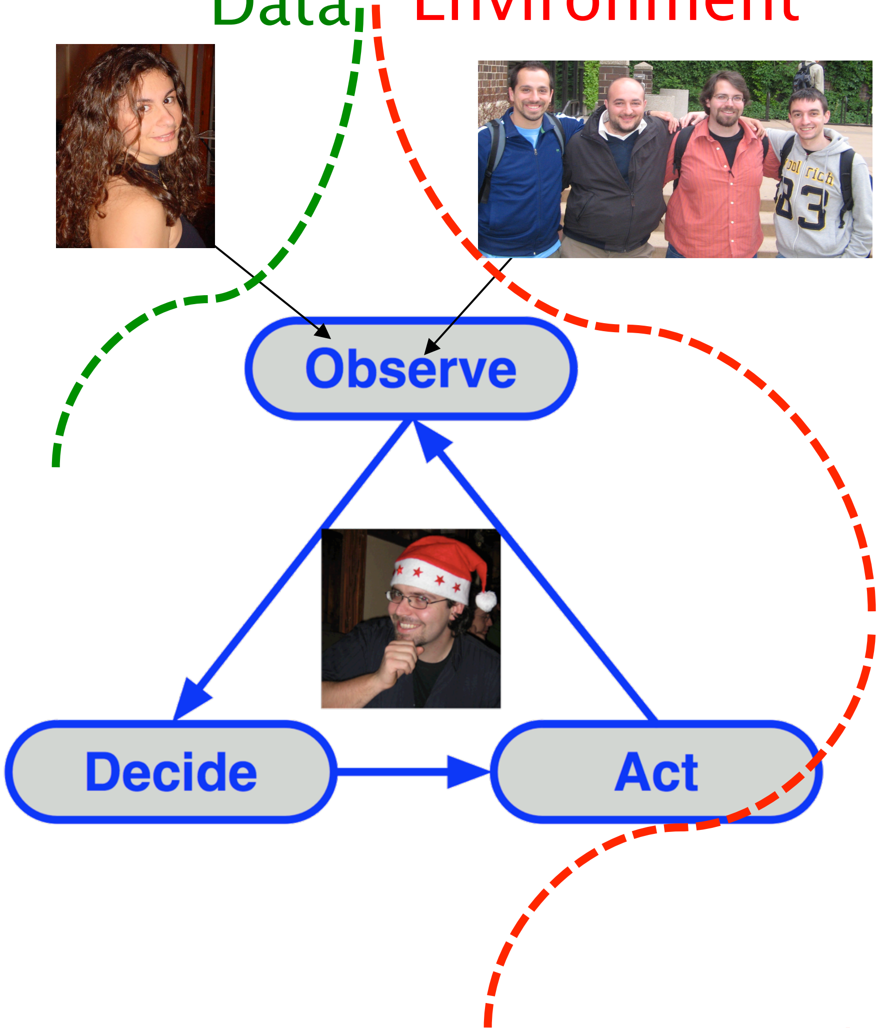
Data Environment



Output – Environment Update

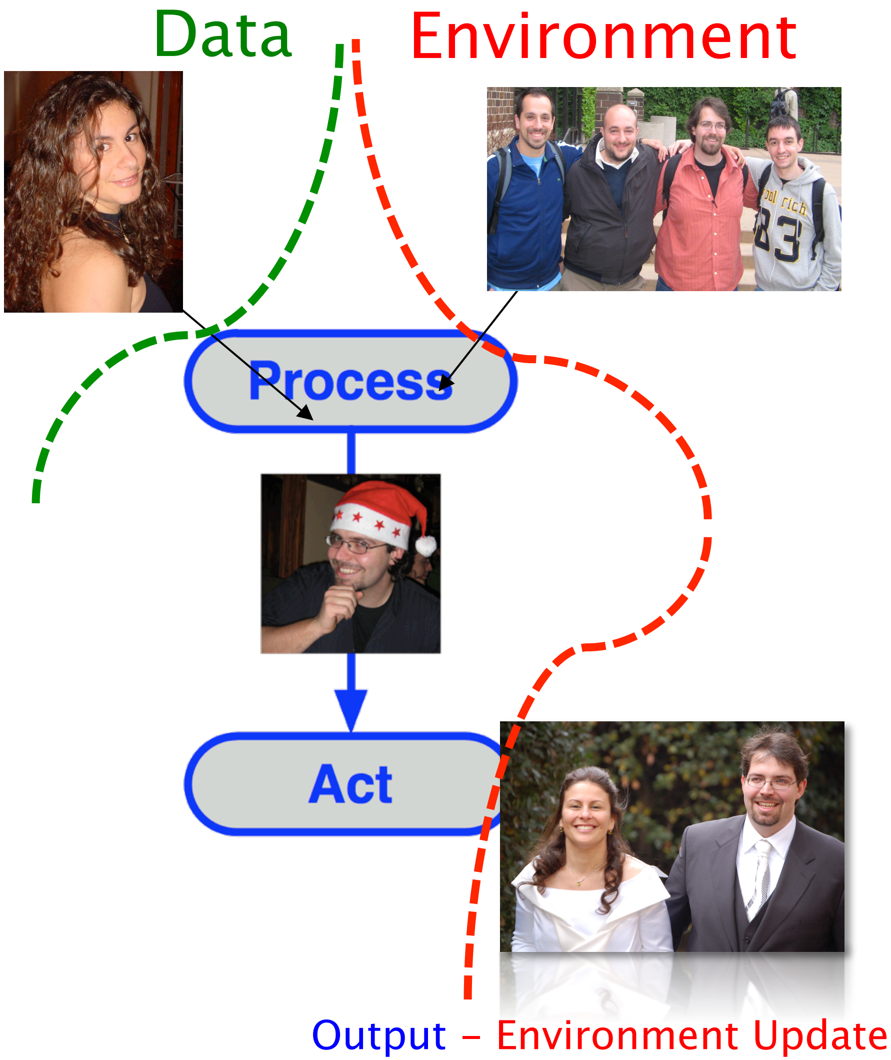
Adaptive Solution

Data Environment



Output – Environment Update

Online Static Solution



Data

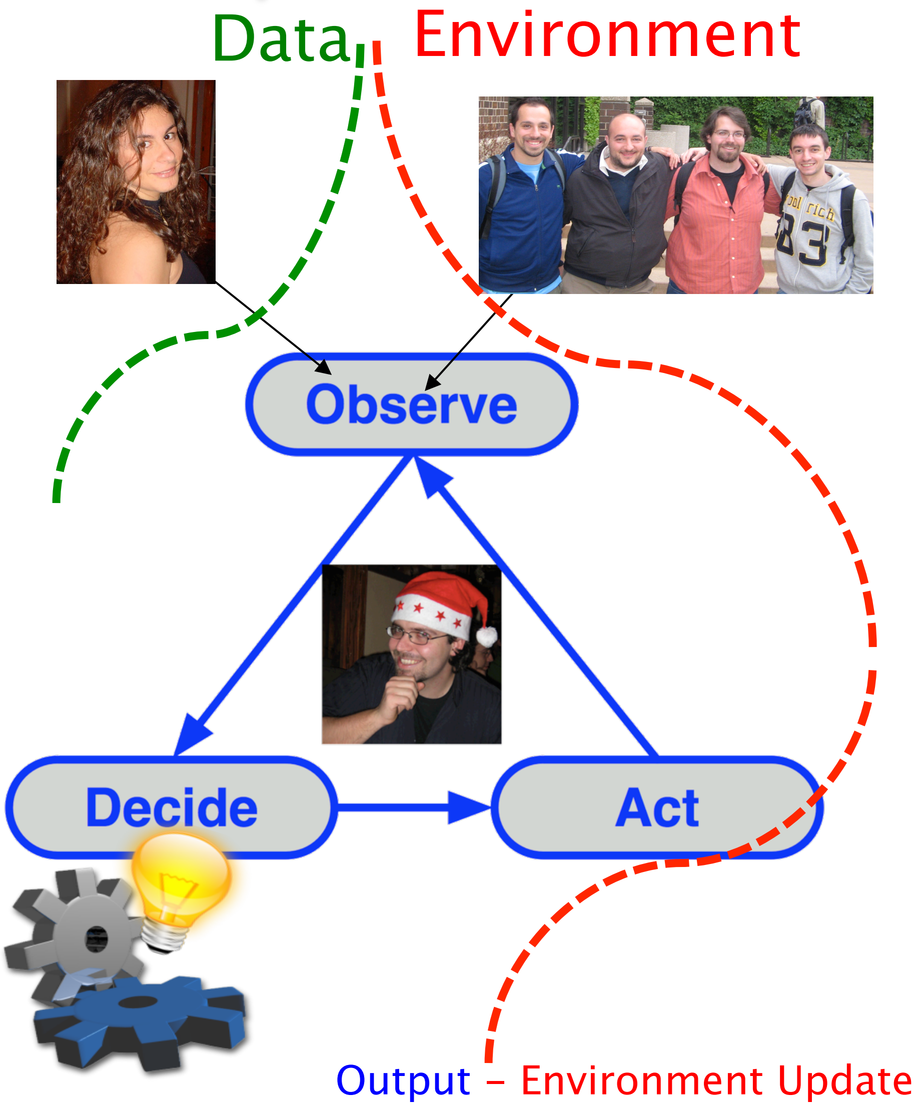
Environment

Process

Act

Output – Environment Update

Adaptive Solution



Data

Environment

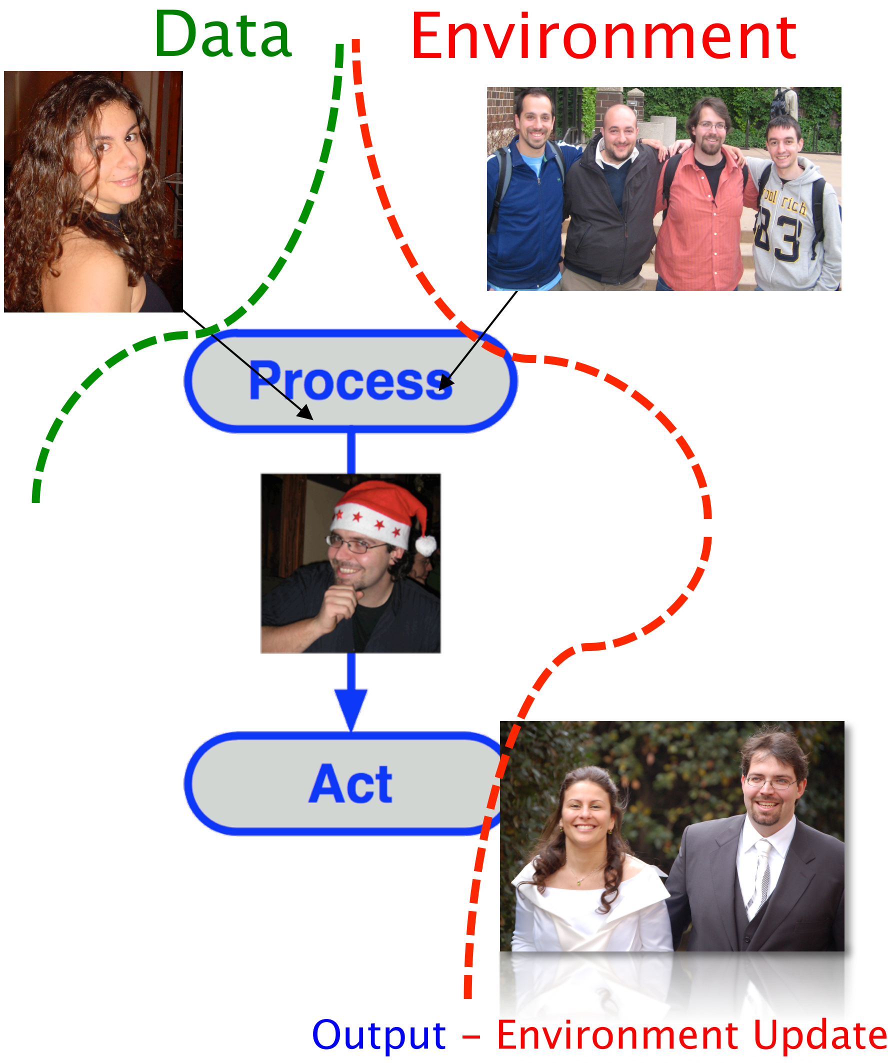
Observe

Decide

Act

Output – Environment Update

Online Static Solution



Data

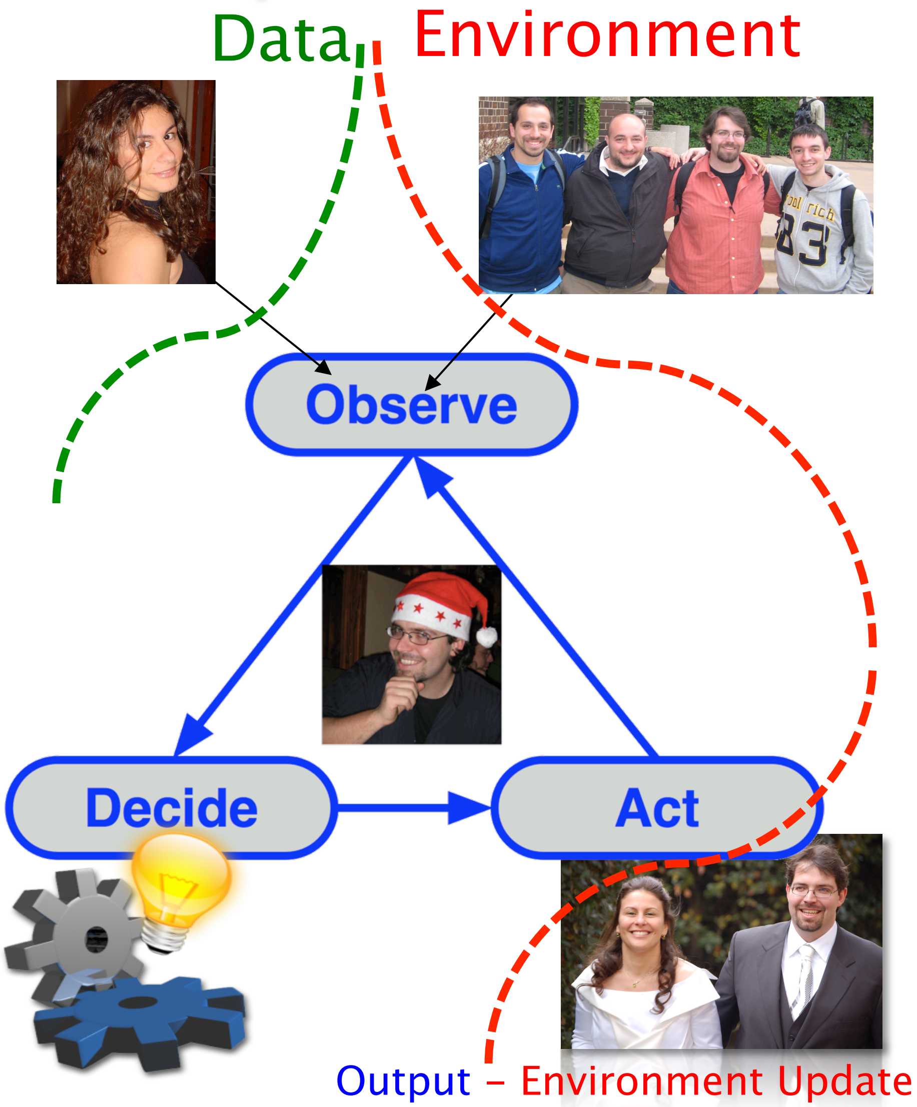
Environment

Process

Act

Output – Environment Update

Adaptive Solution



Data

Environment

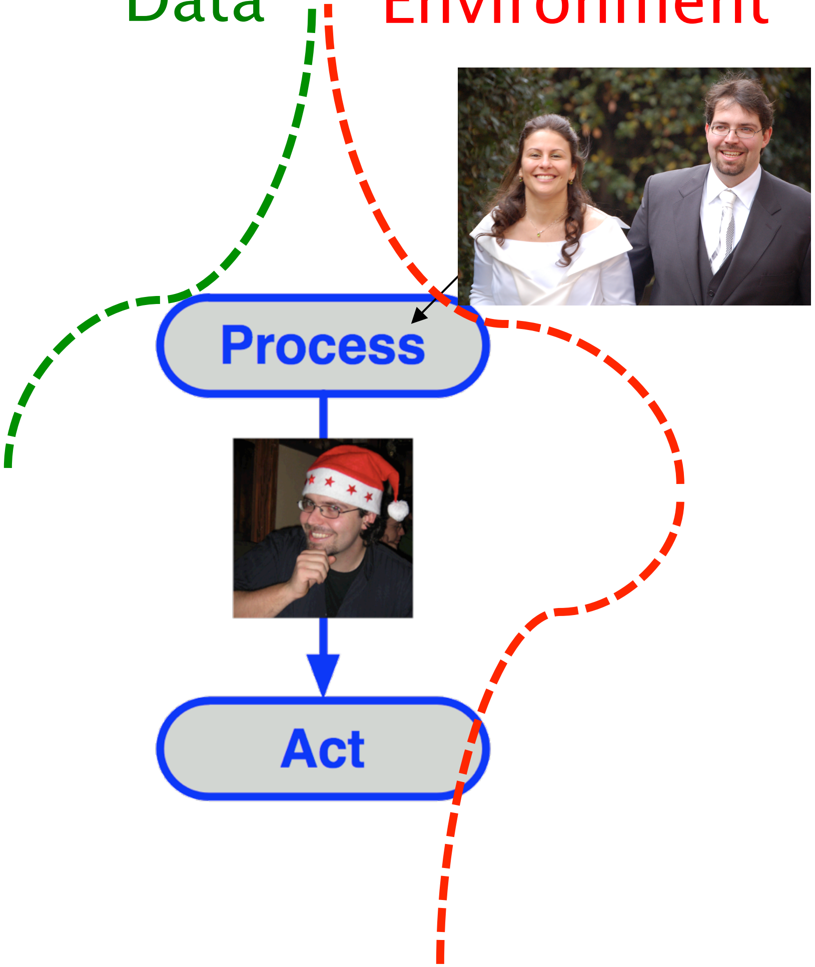
Observe

Decide

Act

Output – Environment Update

Online Static Solution



Data

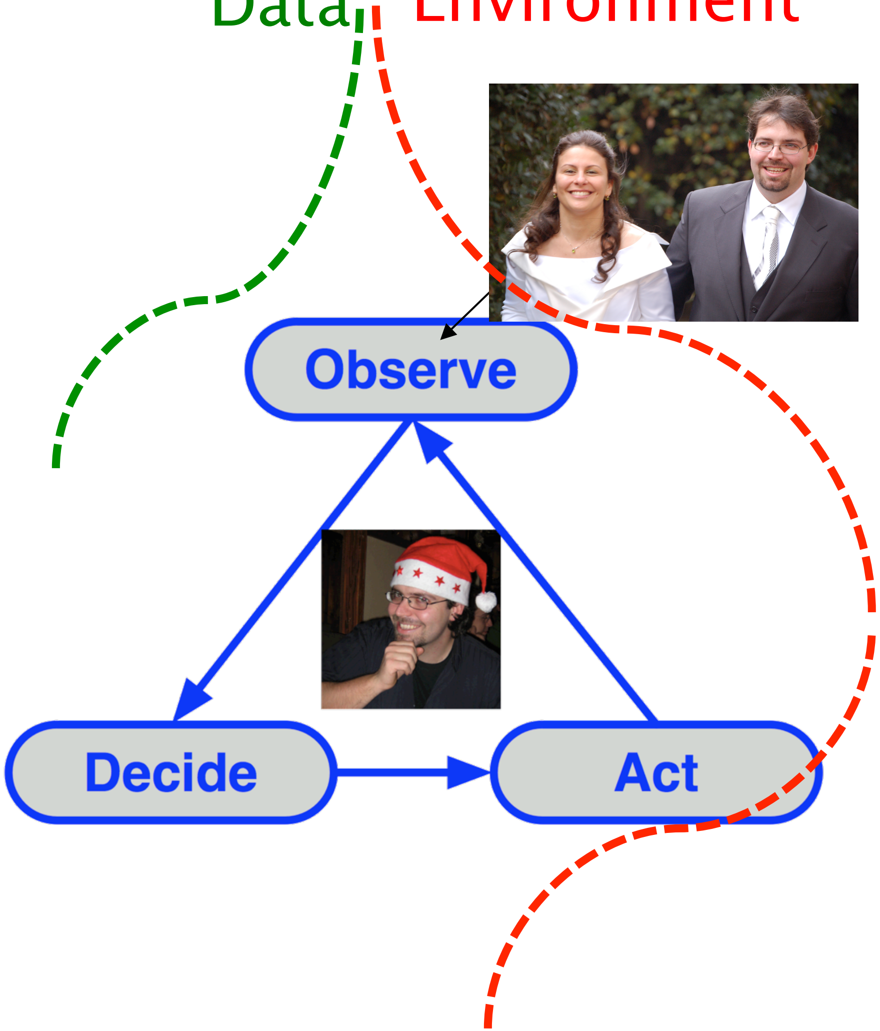
Environment

Process

Act

Output – Environment Update

Adaptive Solution



Data

Environment

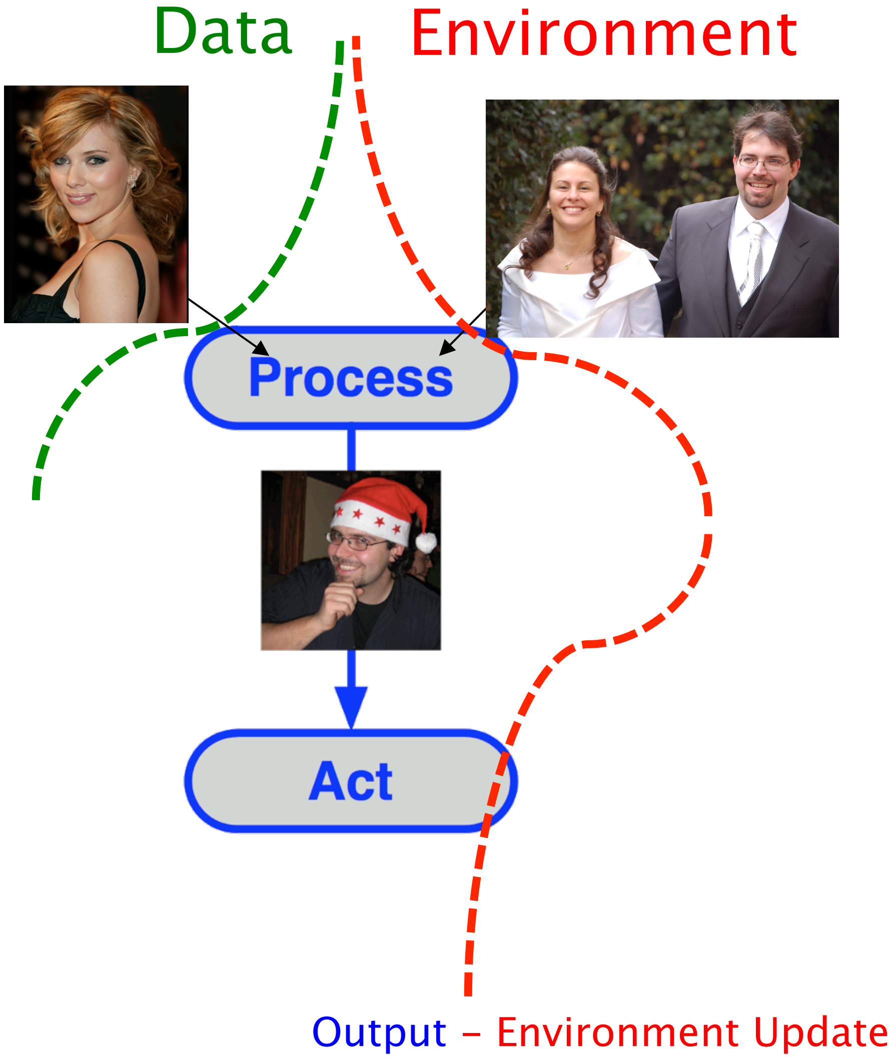
Observe

Decide

Act

Output – Environment Update

Online Static Solution



Data

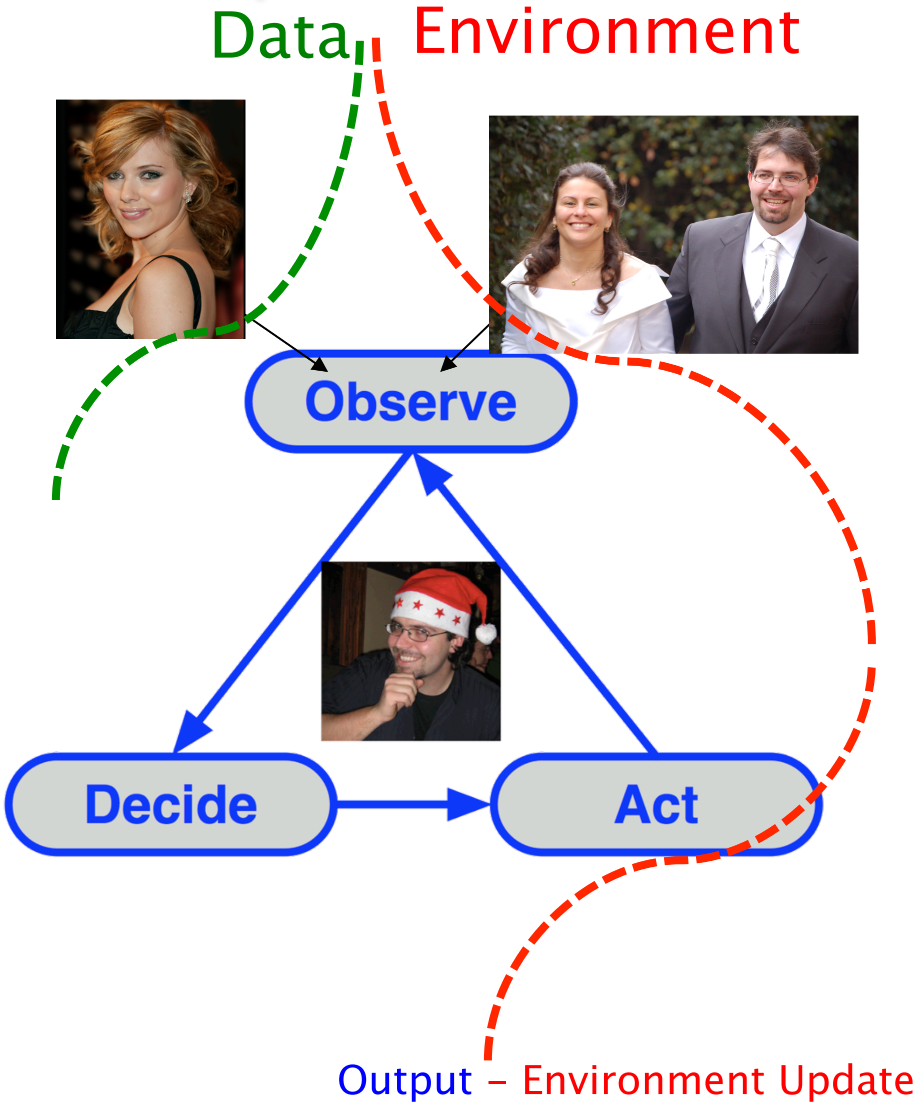
Environment

Process

Act

Output – Environment Update

Adaptive Solution



Data

Environment

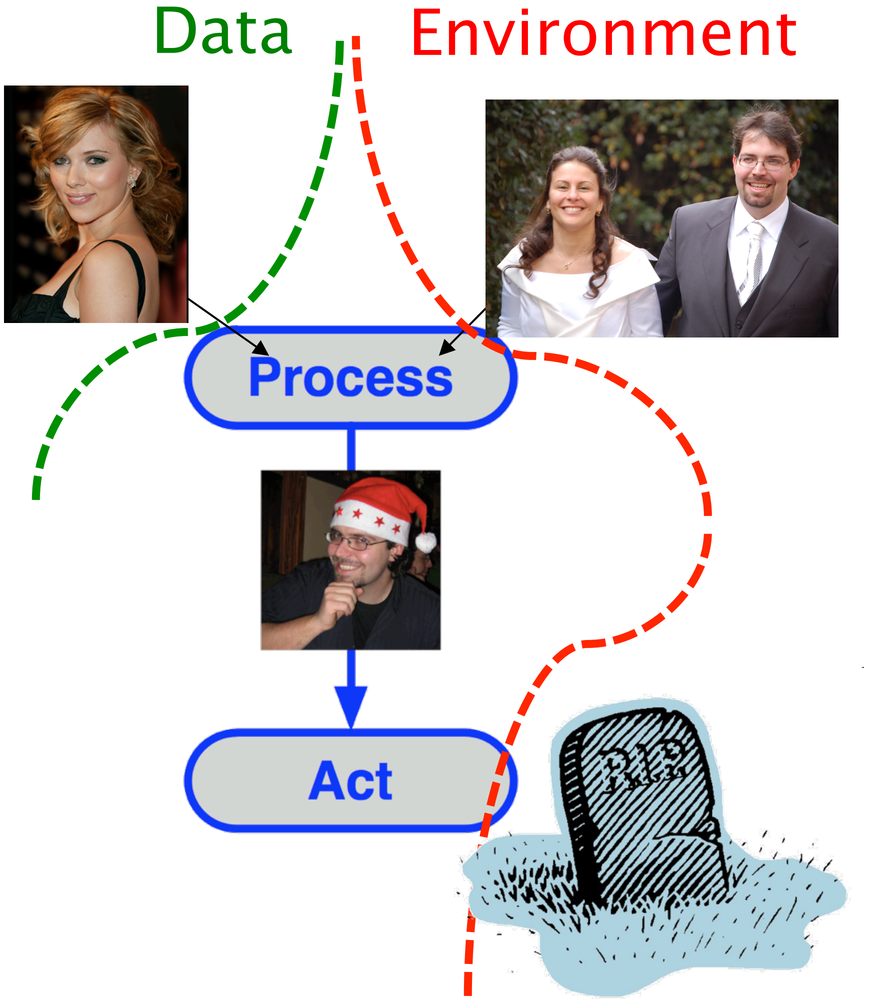
Observe

Decide

Act

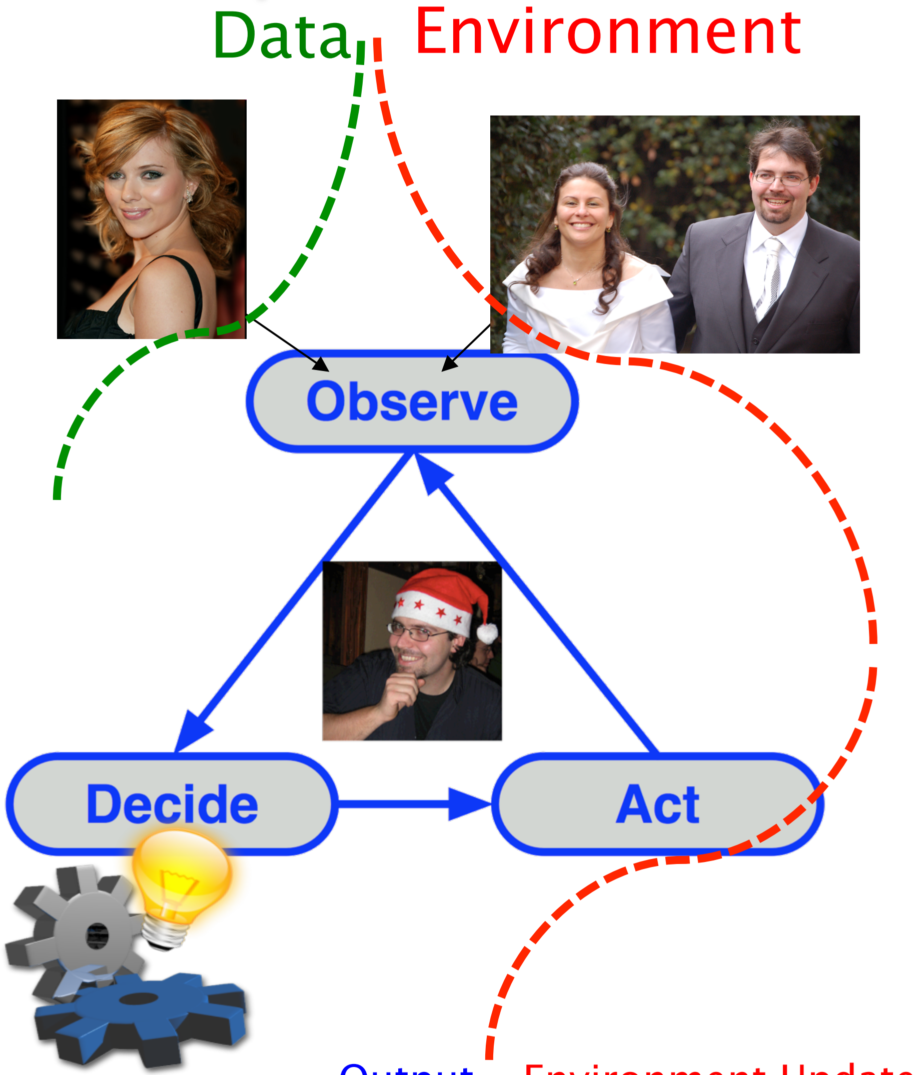
Output – Environment Update

Online Static Solution



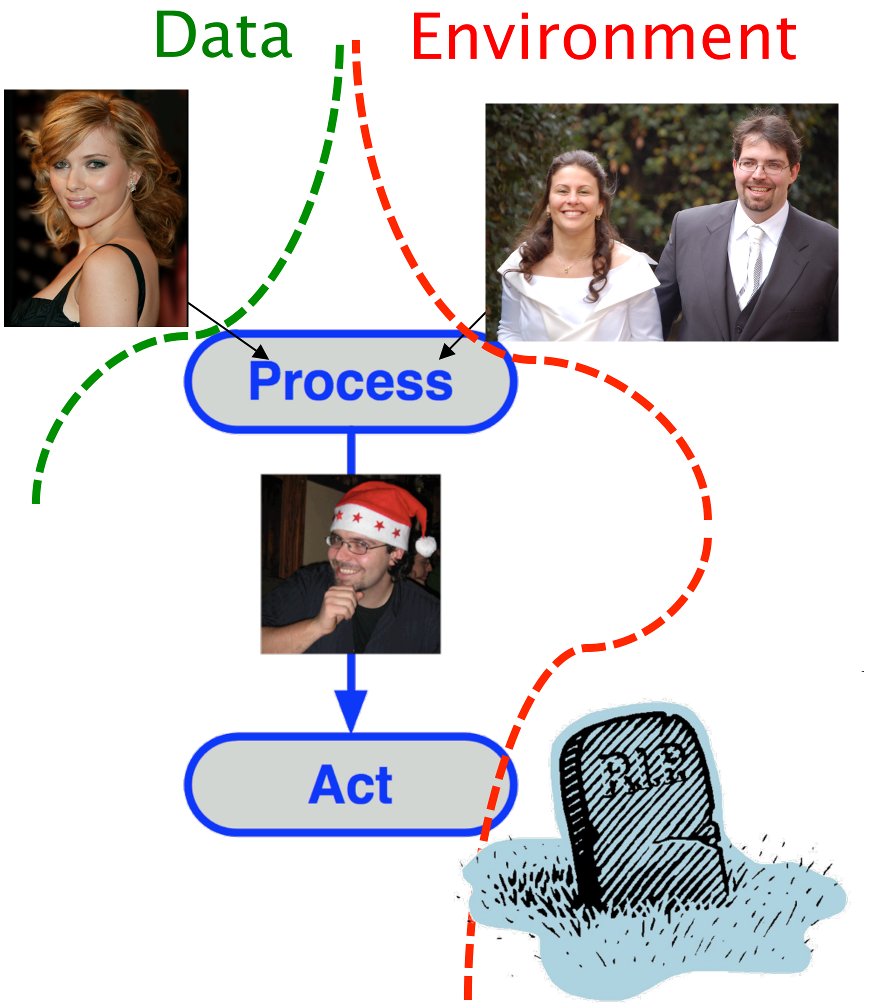
Output – Environment Update

Adaptive Solution



Output – Environment Update

Online Static Solution



Data

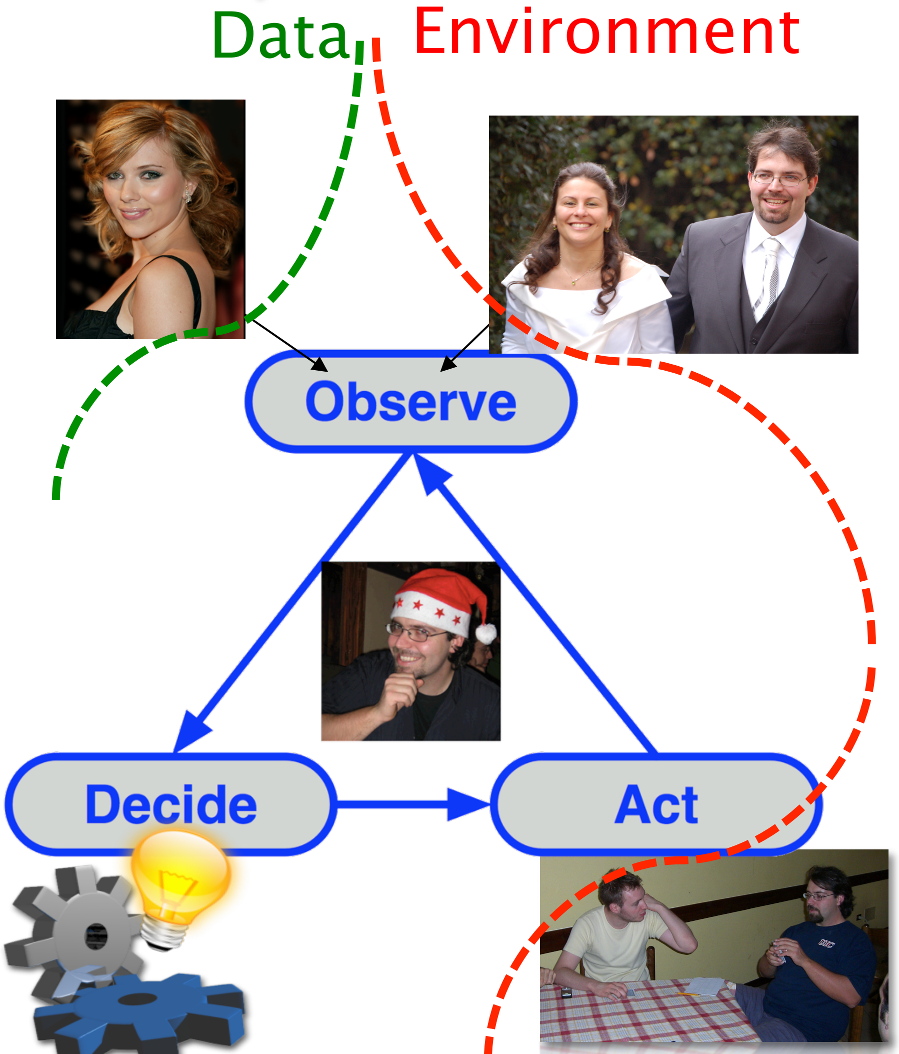
Environment

Process

Act

Output – Environment Update

Adaptive Solution



Data

Environment

Observe

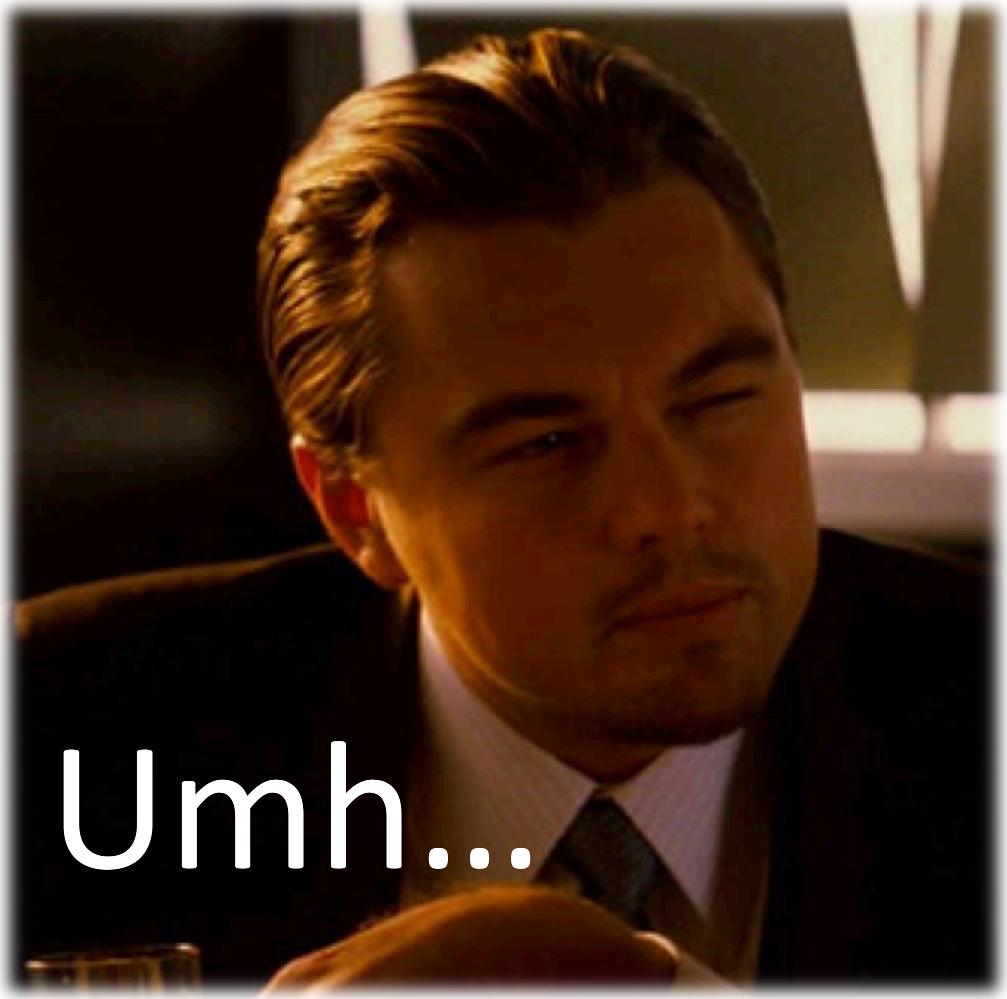
Decide

Act

Output – Environment Update

Nice idea, but

Nice idea, but how to use it!



Autonomic Operating System

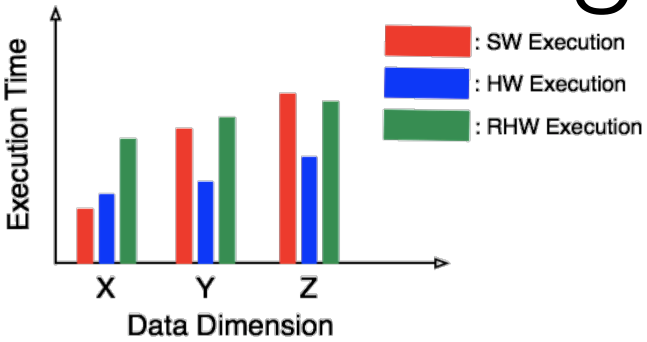
- The AcOS project aims at
  - designing and prototyping a patch for commodity operating systems (e.g. Linux, FreeBSD)
  - being capable to observe its own execution and optimize, in a self-aware manner, its behavior with respect to the external environment, to user needs and to applications demands

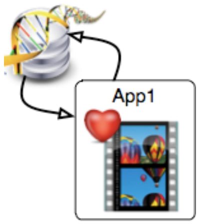
[integrated/used in different research projects]

The research effort

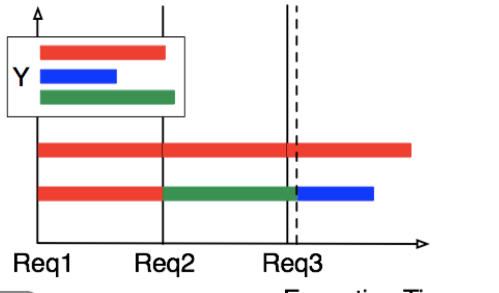
- K42
  - http://researcher.watson.ibm.com/researcher/view_project.php?id=2078
- The SElf-awarE Computing (SEEC) Model
  - http://groups.csail.mit.edu/carbon/seec/
- Angstrom
  - http://projects.csail.mit.edu/angstrom/
- The Swarm project and Tessellation OS
  - http://tessellation.cs.berkeley.edu

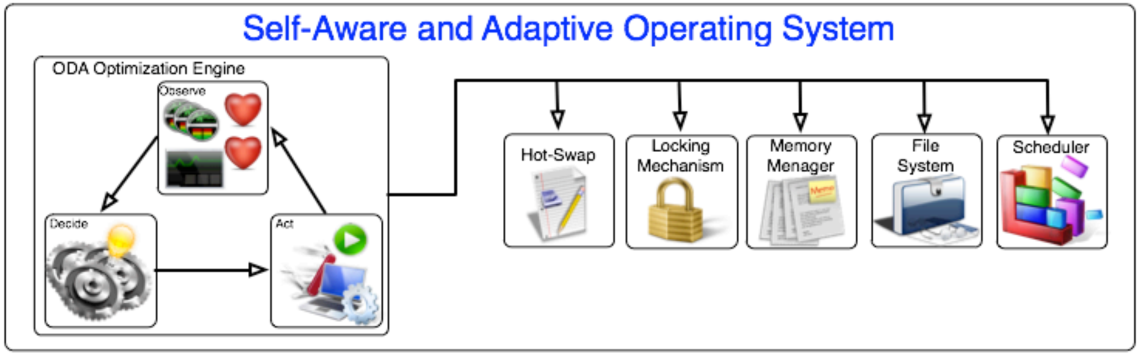
Reconfiguration: Self-Aware

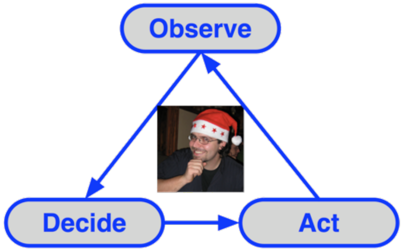


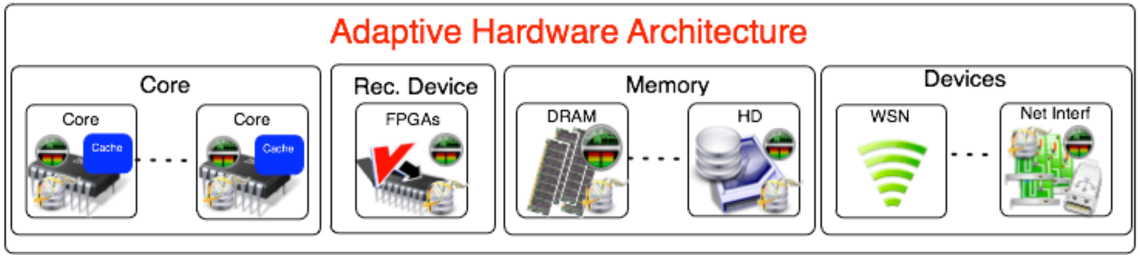


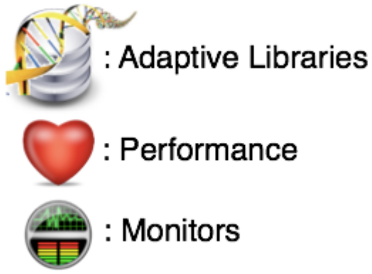
3 requests on Y data (NO HW configured)



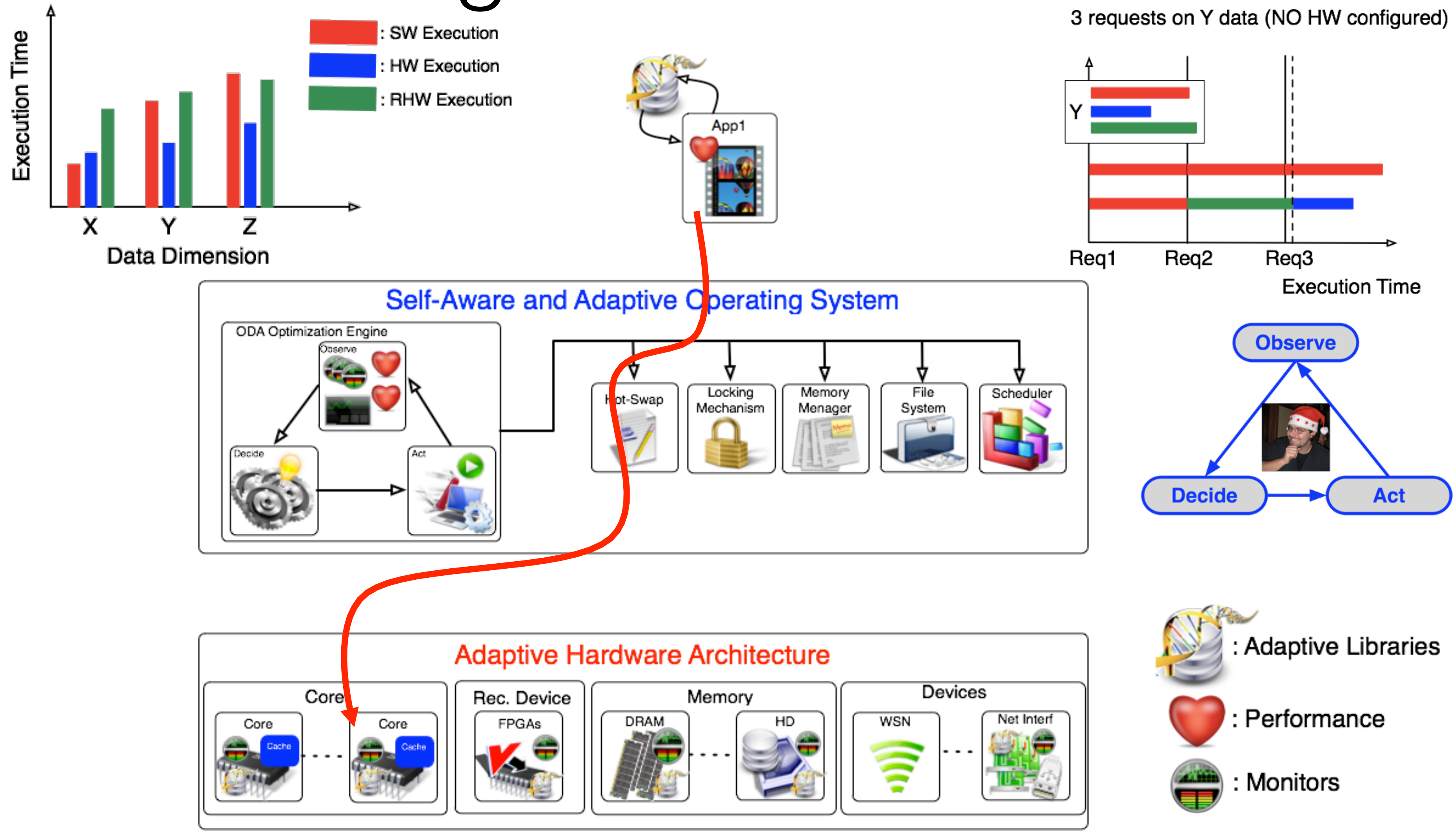




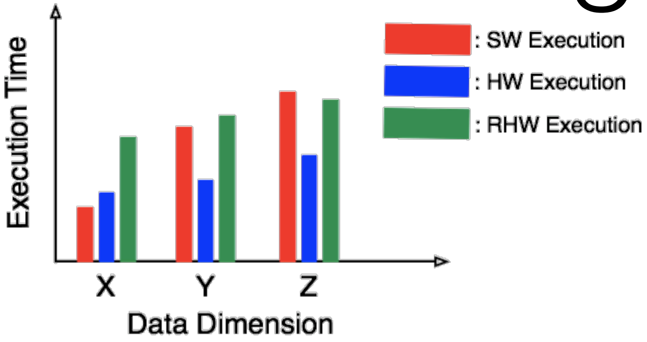


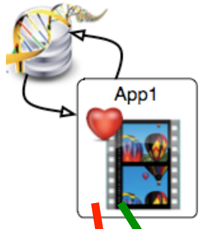


Reconfiguration: Self-Aware

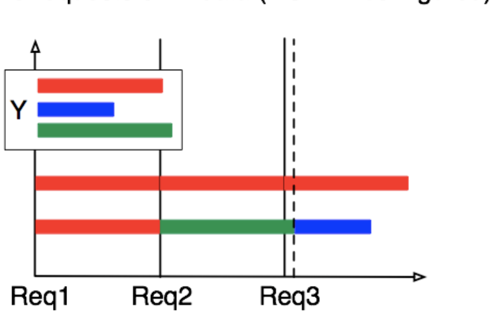


Reconfiguration: Self-Aware

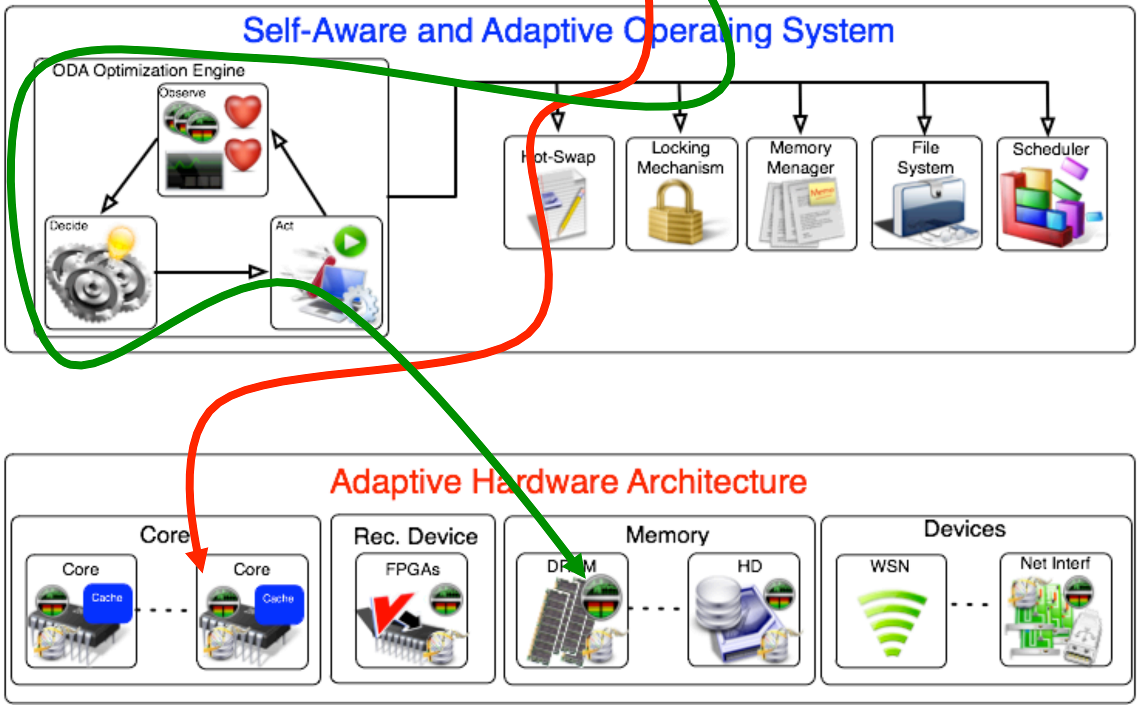


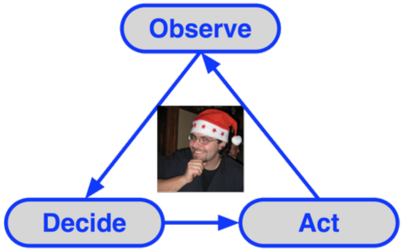


3 requests on Y data (NO HW configured)



Execution Time





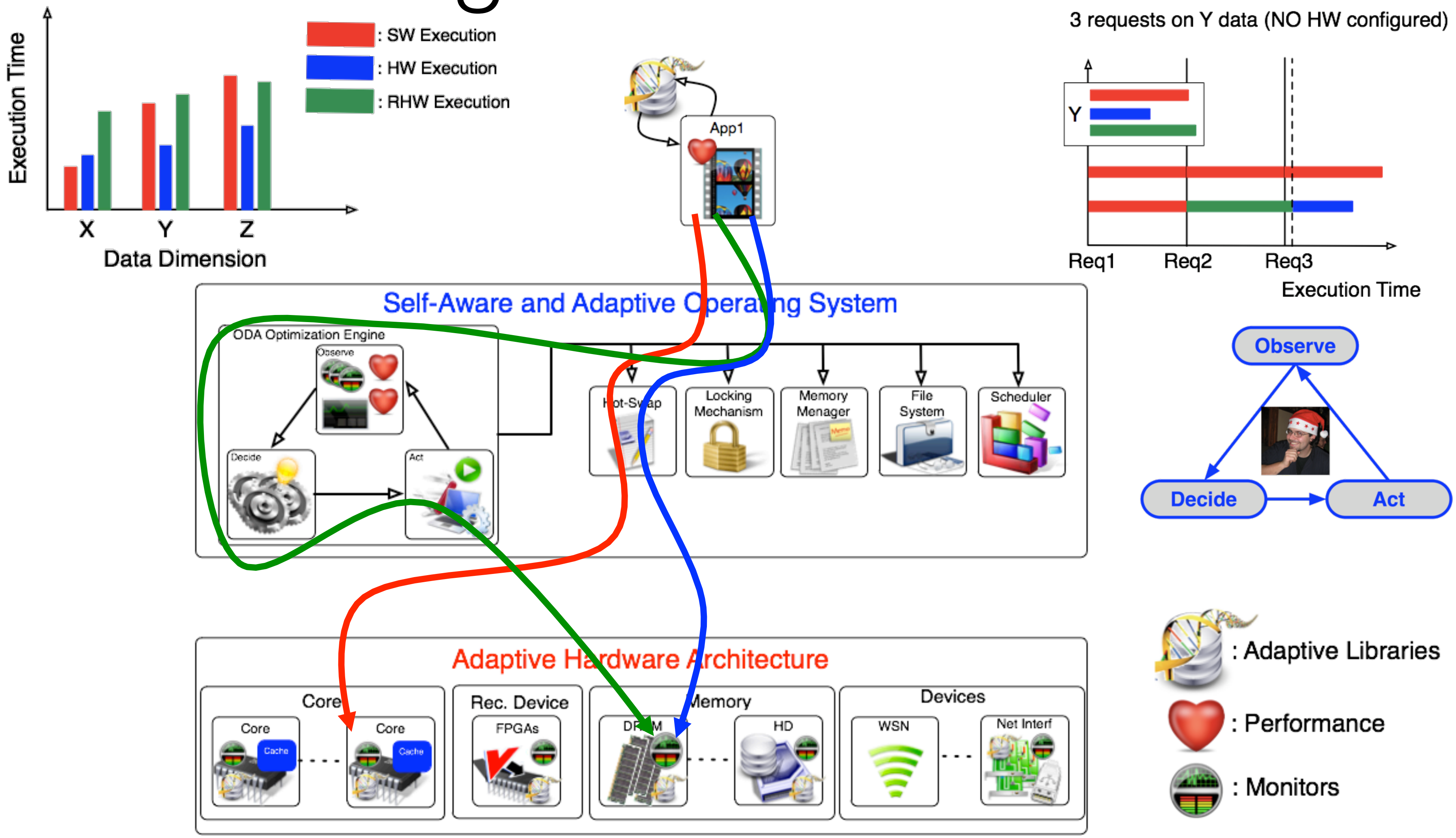
: Adaptive Libraries

: Performance

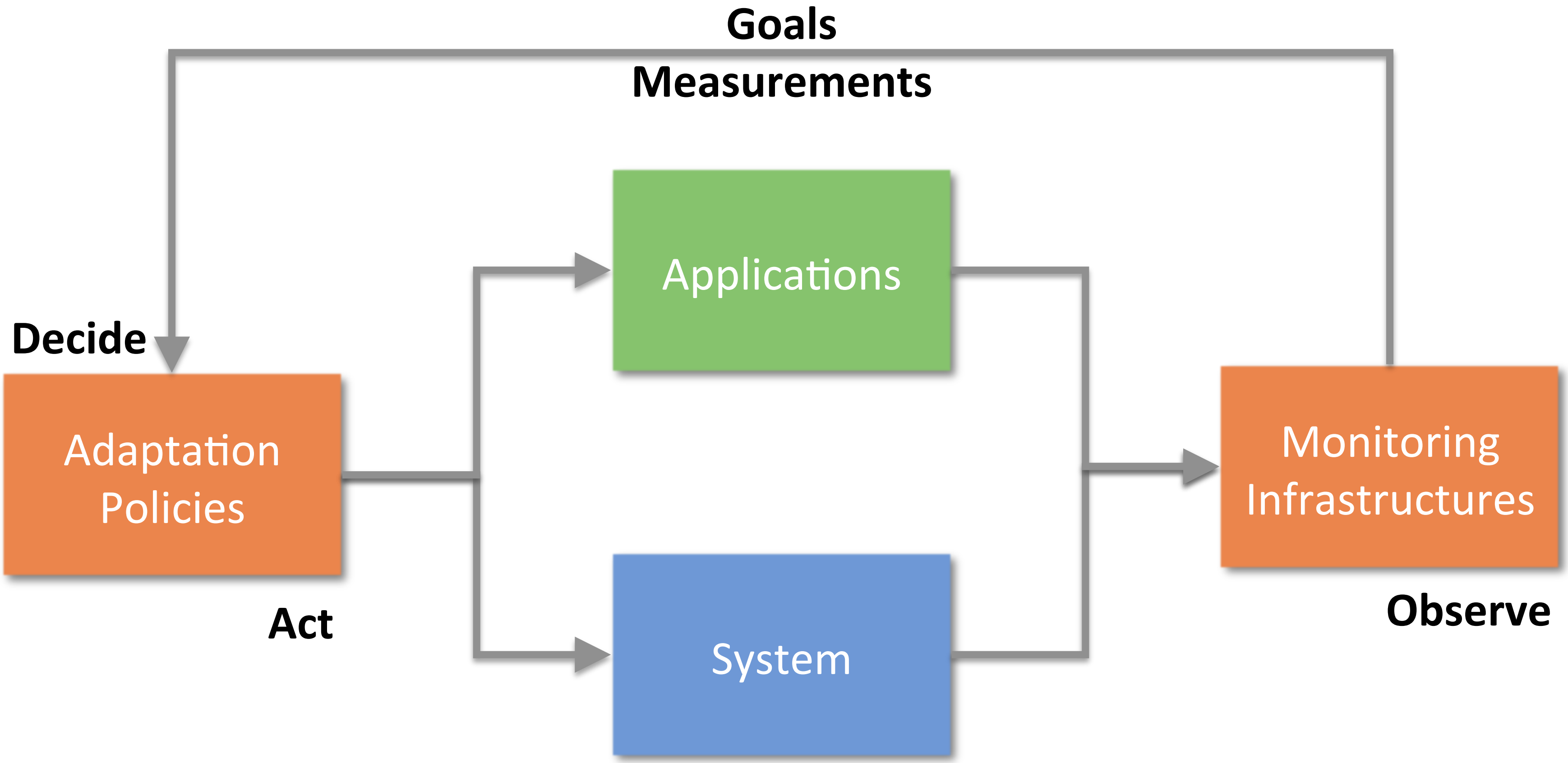
: Monitors

WHAAAAAAT?

Reconfiguration: Self-Aware



AcOS: via an intelligent ODA loop



The Heart Rate Monitor

- Heartbeats signal[1] either progresses or availability
  - video encoder: 1 heartbeat = 1 frame
  - web server: 1 heartbeat = 1 request
  - database server: 1 heartbeat = transaction
- Heart rate as a performance measure and goal
  - High-level, application-specific performance measurements and goals (e.g., video encoder: 30 heartbeats/s = 30 frames/s)
- Compact API, user/kernel-space partitioned implementation[2]
  - User-mode fast-path heartbeats issue
  - User and kernel-mode low-latency heart rate access

[1] Hoffmann et al., Application Heartbeats for Software Performance and Health,
Proc. of the 15th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming, 2010

[2] Sironi et al., *Metronome: Operating System Level Performance Management via Self-Adaptive Computing*,
Proc. of the 49th Annual Design Automation Conference 2012

The Heart Rate Monitor

The Heart Rate Monitor

- Set performance goal

The Heart Rate Monitor

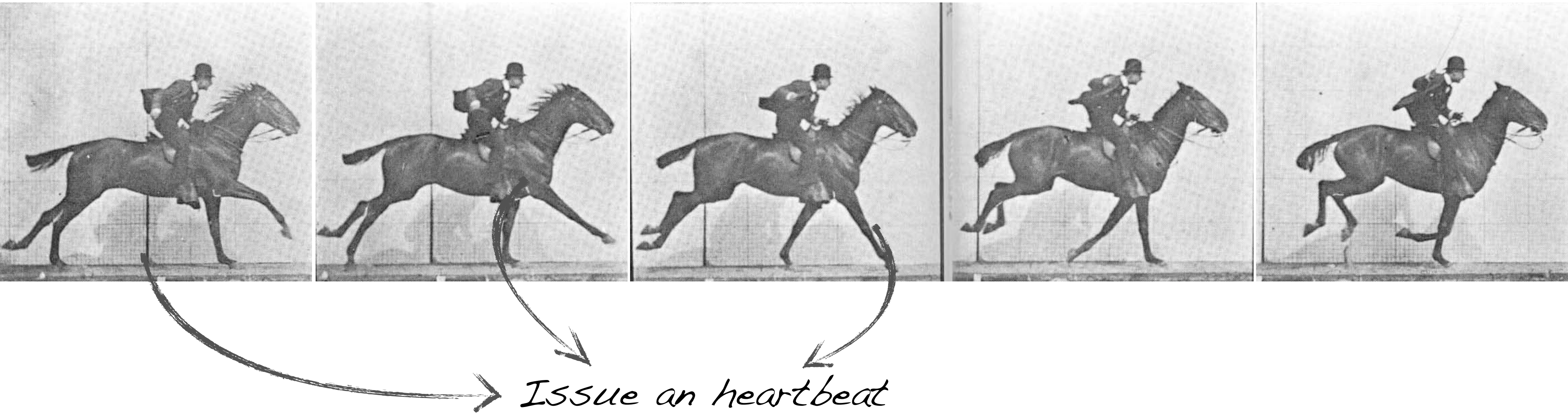
- Set performance goal ←

e.g.:
min: 25hb/sec
max: 35hb/sec

The Heart Rate Monitor

- Set performance goal
- Run the app and update progress

e.g.:
min: 25hb/sec
max: 35hb/sec

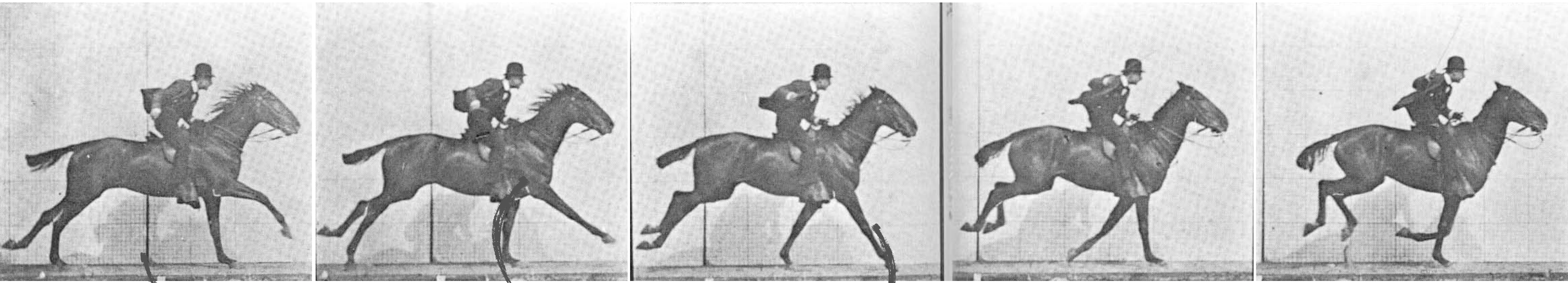


Issue an heartbeat

The Heart Rate Monitor

- Set performance goal
- Run the app and update progress

e.g.:
min: 25hb/sec
max: 35hb/sec



Issue an heartbeat

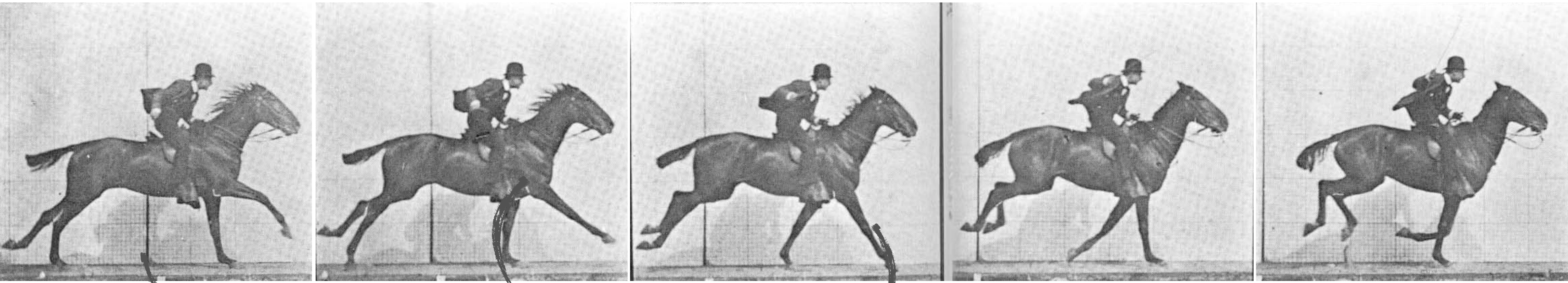
Statistics automatically updated

The Heart Rate Monitor

- Set performance goal

e.g.:
min: 25hb/sec
max: 35hb/sec

- Run the app and update progress



Issue an heartbeat → Statistics automatically updated

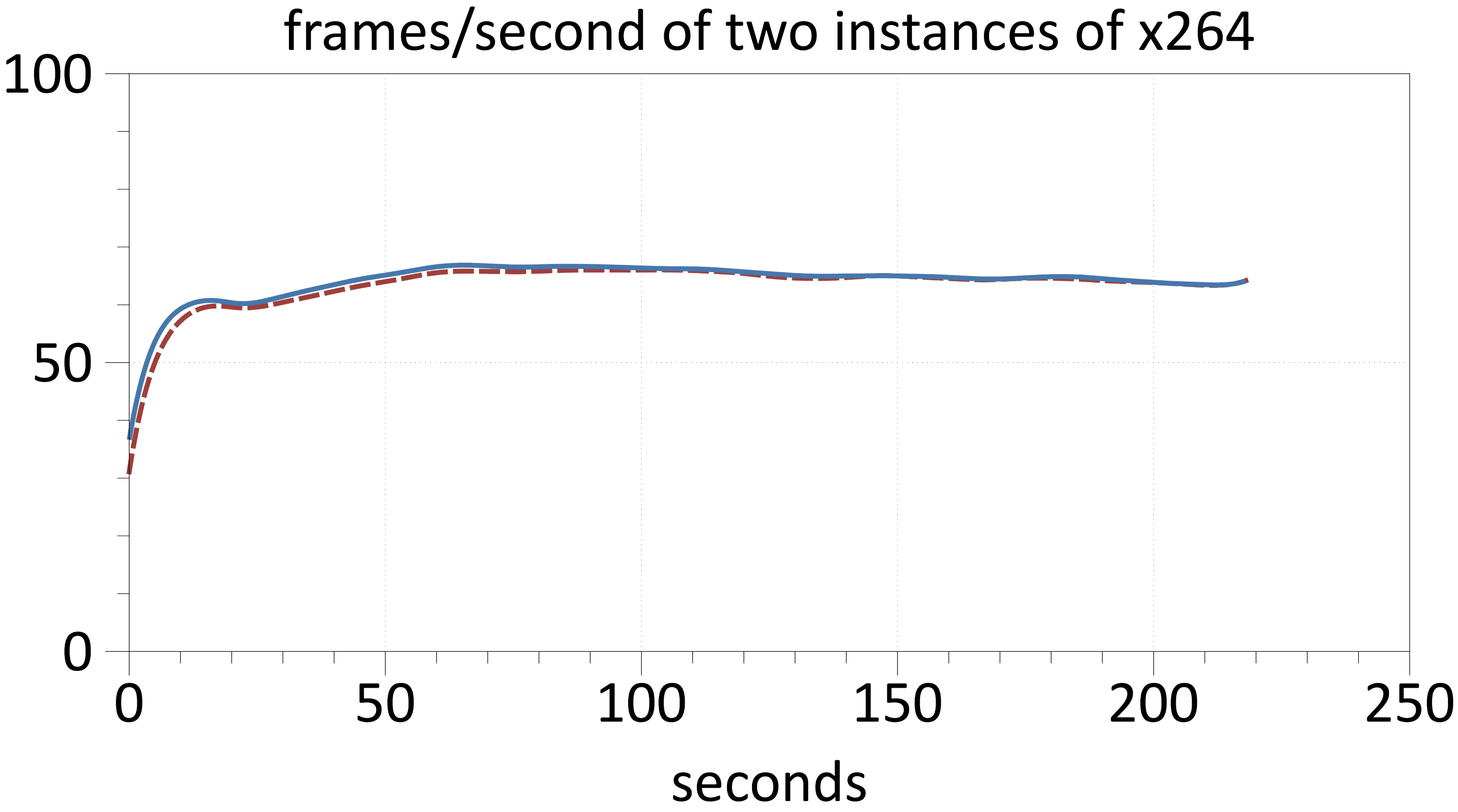
- Check heart rate and, if necessary, react

Autonomic Scheduling

- In a scenario were applications
  - are competing for the same set of resources
  - require predictable performance, expressed through high-level, application-specific metrics

- The scheduler has to become Performance-Aware to automatically allocate resources to match performance goals
  - With Metronome, we introduce performance-awareness by means of a non-invasive modifications to the Completely Fair Scheduler

Metronome


frames/second of two instances of x264
100
50
0
0
50
100
150
200
250
seconds

Metronome

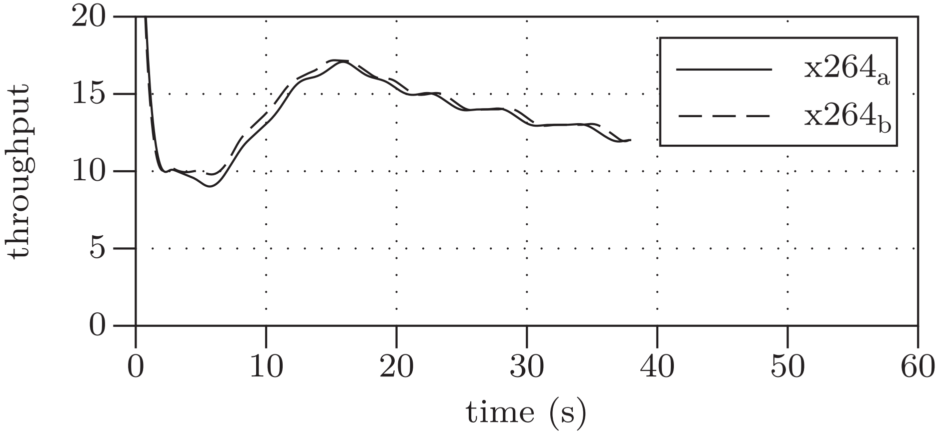
Metronome

Sironi et al., *Metronome: Operating System Level Performance Management via Self-Adaptive Computing*,
Proc. of the 49th Annual Design Automation Conference 2012

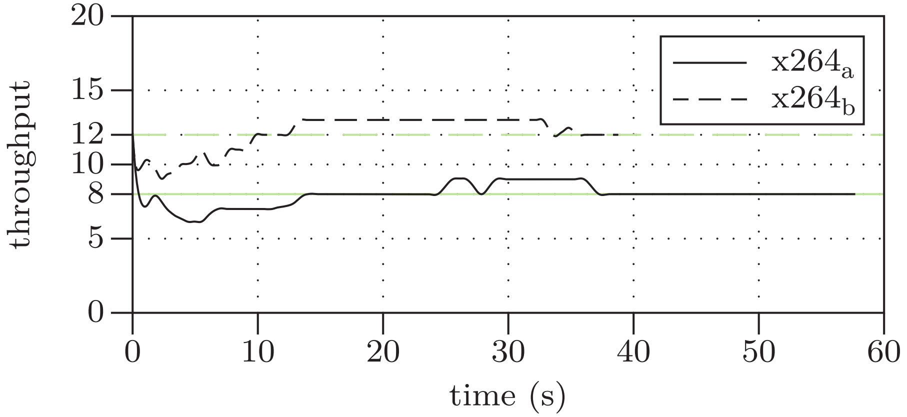
From Metronome to Metronome++

- Metronome demonstrates how a simple heuristic can be enough to enable goal-oriented resource allocation by exploiting runtime performance feedback

- Metronome++ has been designed with more advanced adaptation policy to dynamically allocate CPUs to SLO-bound
  - E.g. Through tasks migration among run queues



(1) Linux kernel vanilla



(2) Linux kernel enhanced with Metronome++

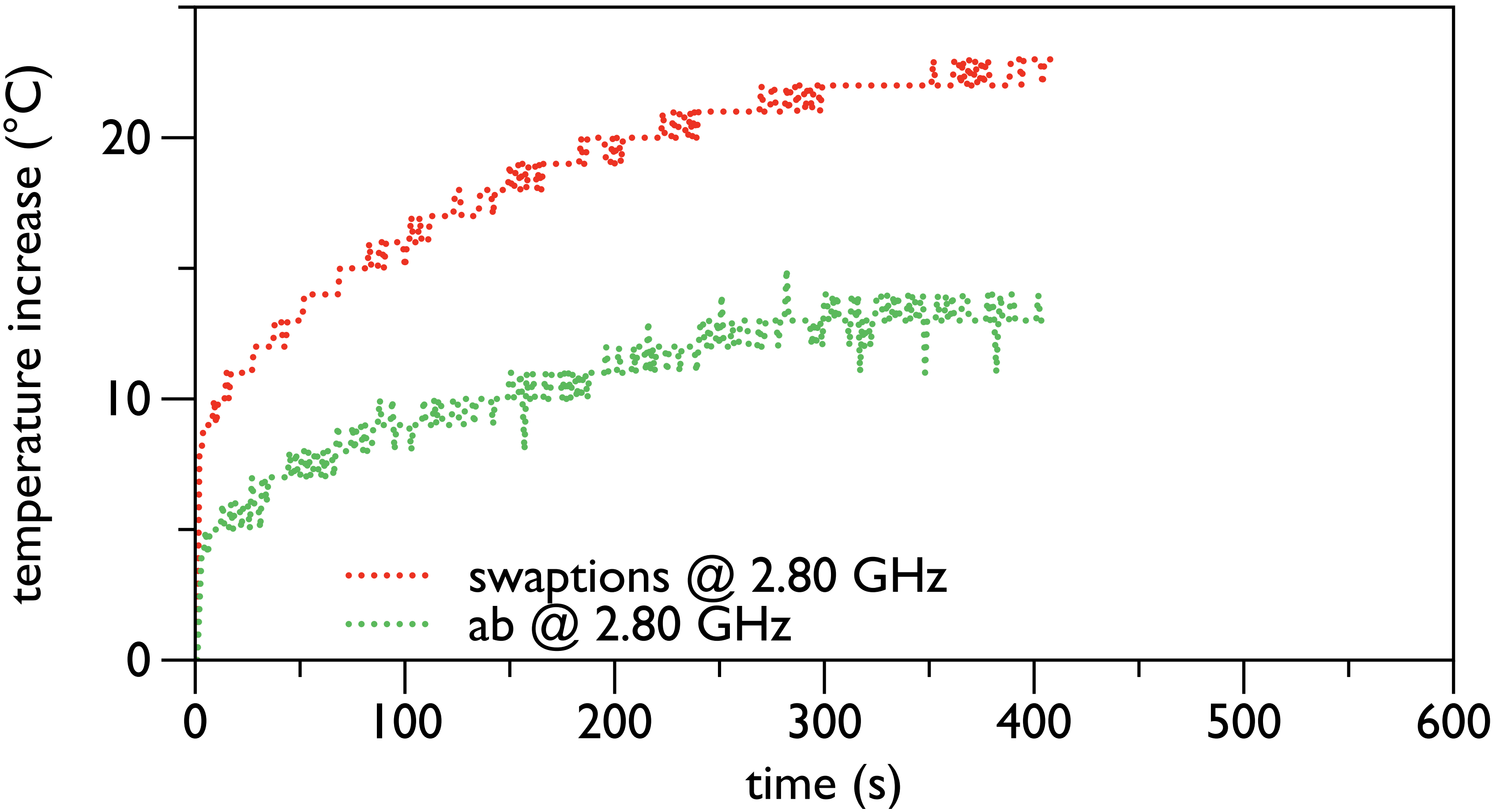
Trying to raise the bar (again)

- AcOS took into consideration performance…
- Any other *HOT* topic?

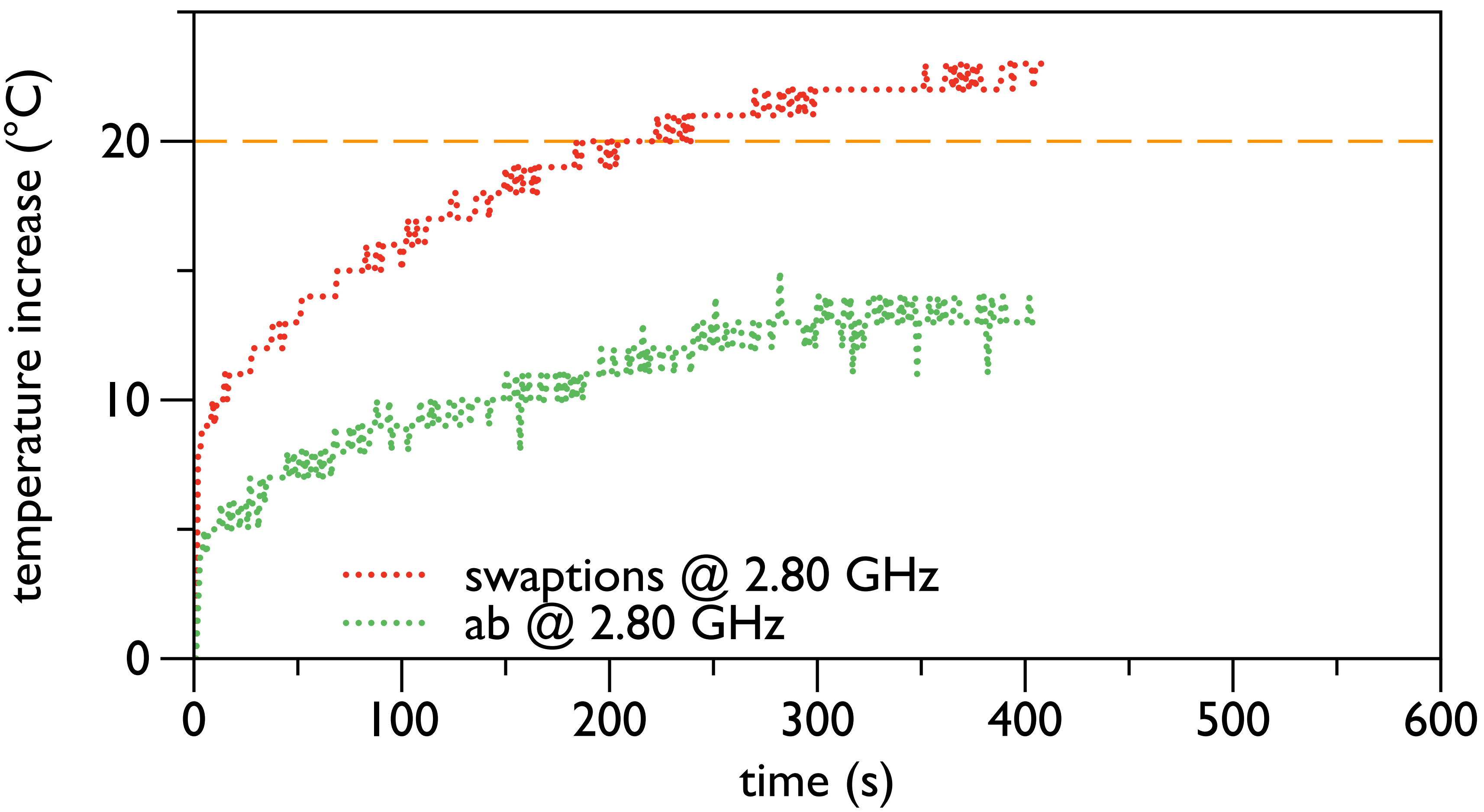
Trying to raise the bar (again)

- AcOS took into consideration performance…
- Any other *HOT* topic?
  - What about temperature Control/Management!

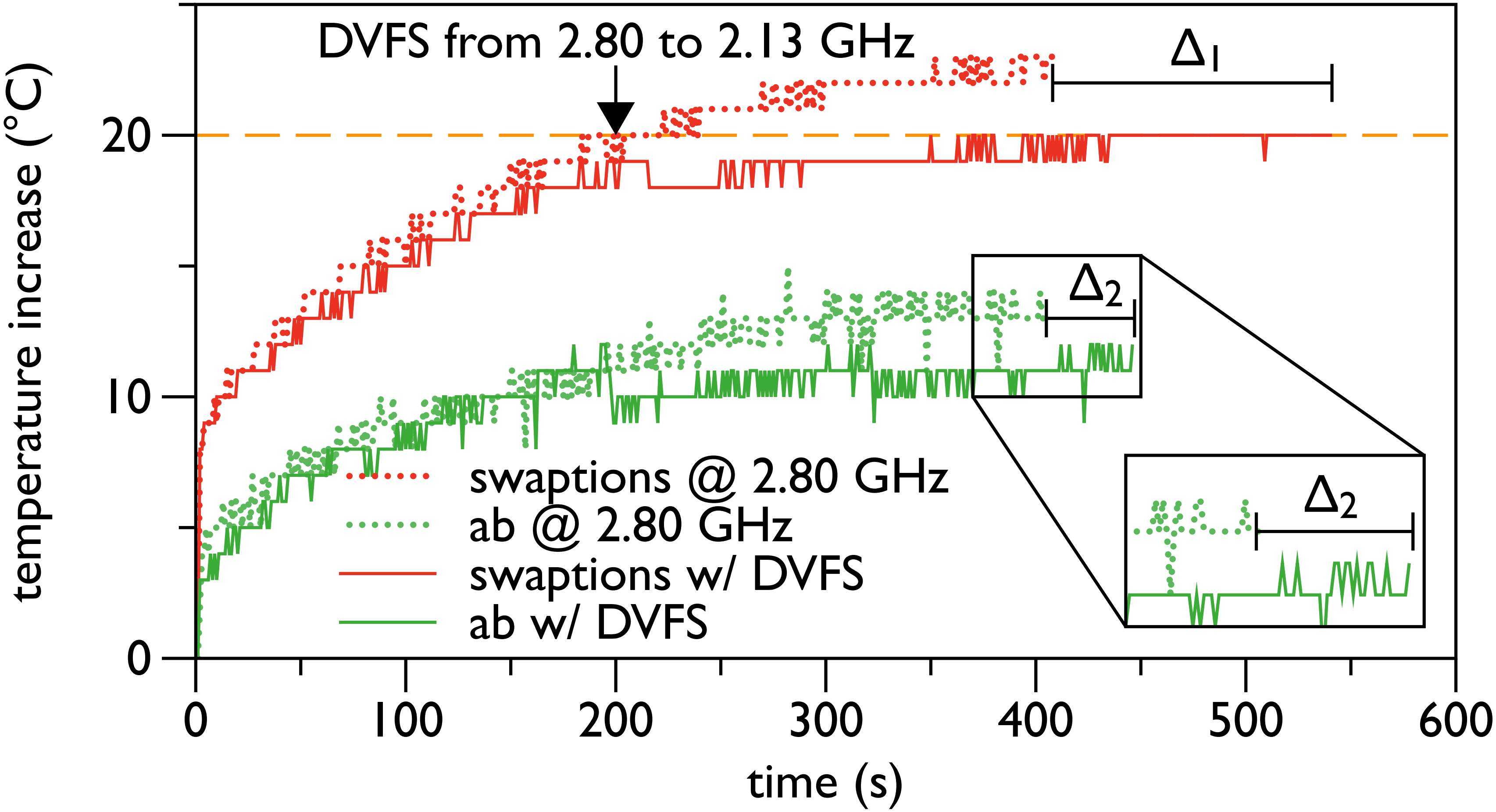
Temperature Control/Management starting scenario



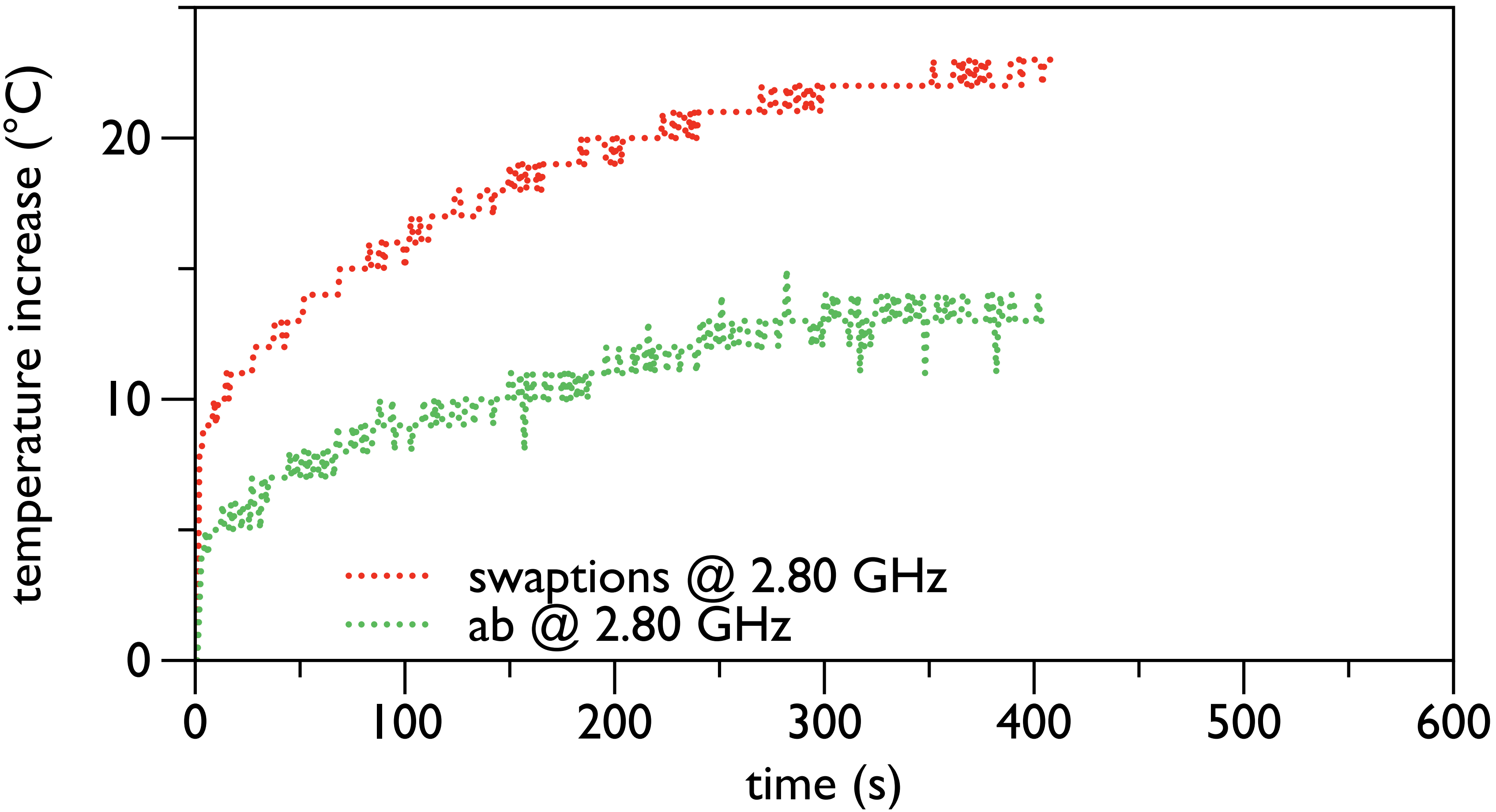
Temperature Control/Management
set a temperature cap



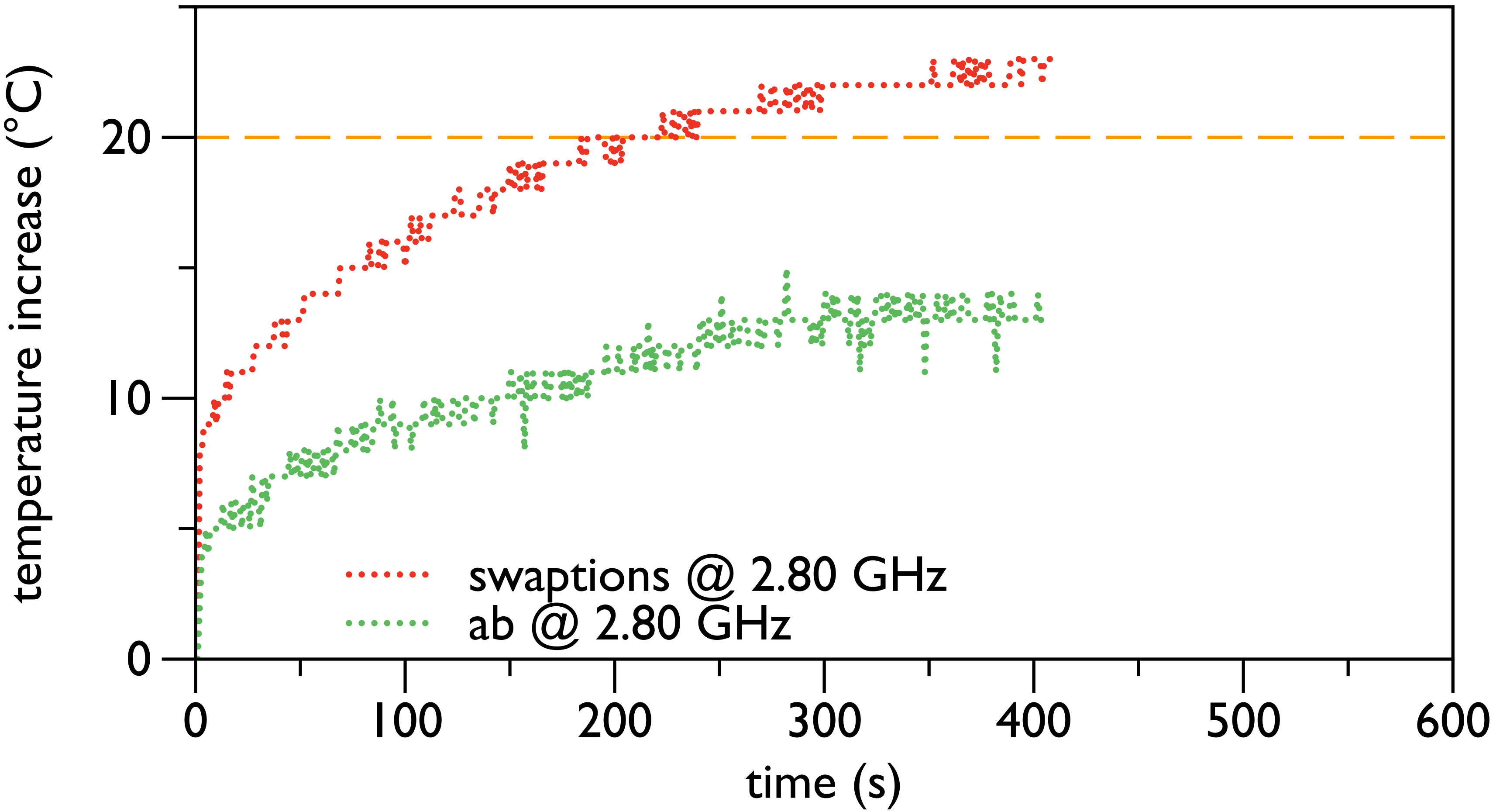
Temperature Control/Management DVFS is dangerous



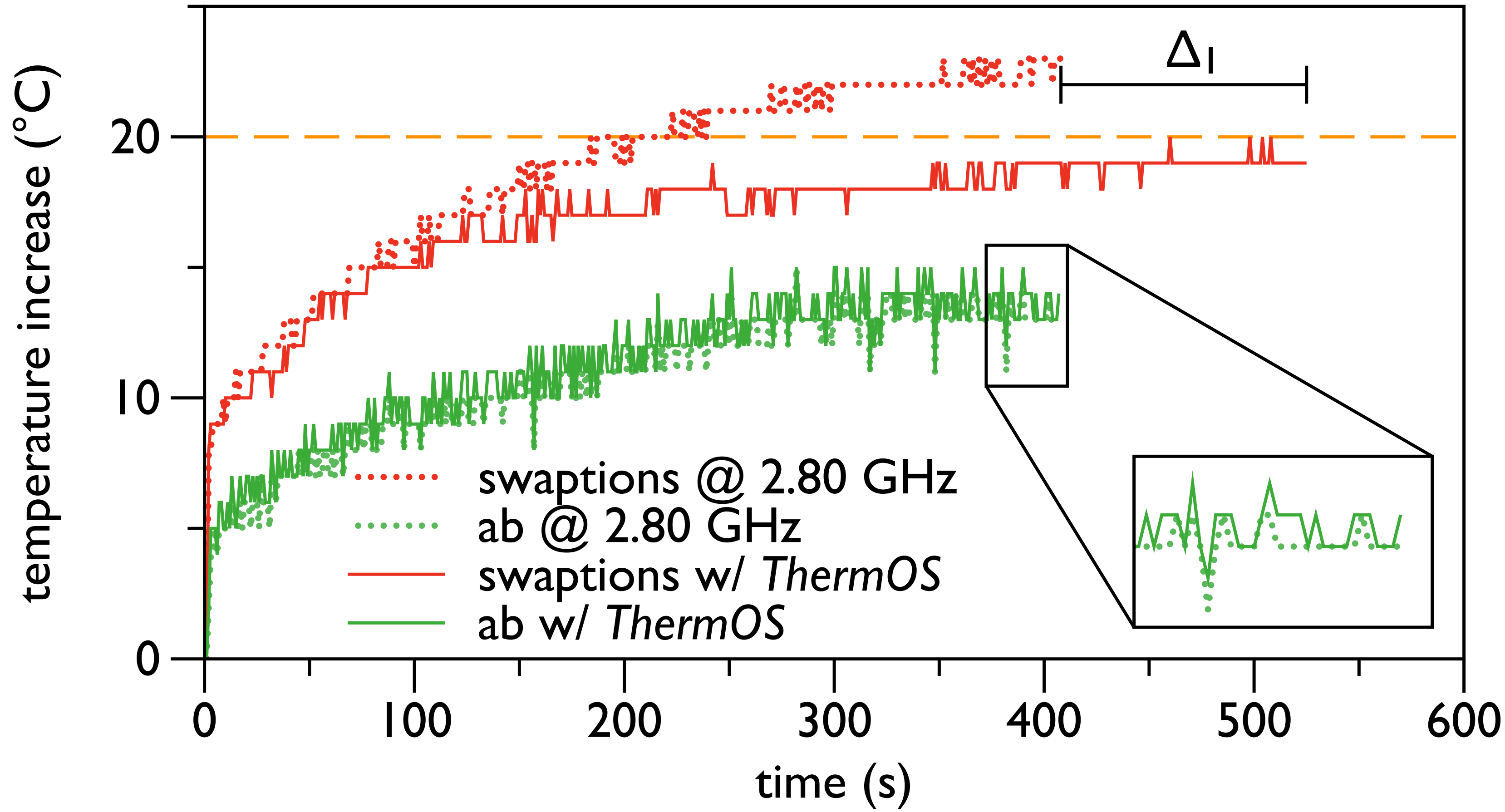
Temperature Control/Management back to the starting scenario



Temperature Control/Management set the same temperature cap



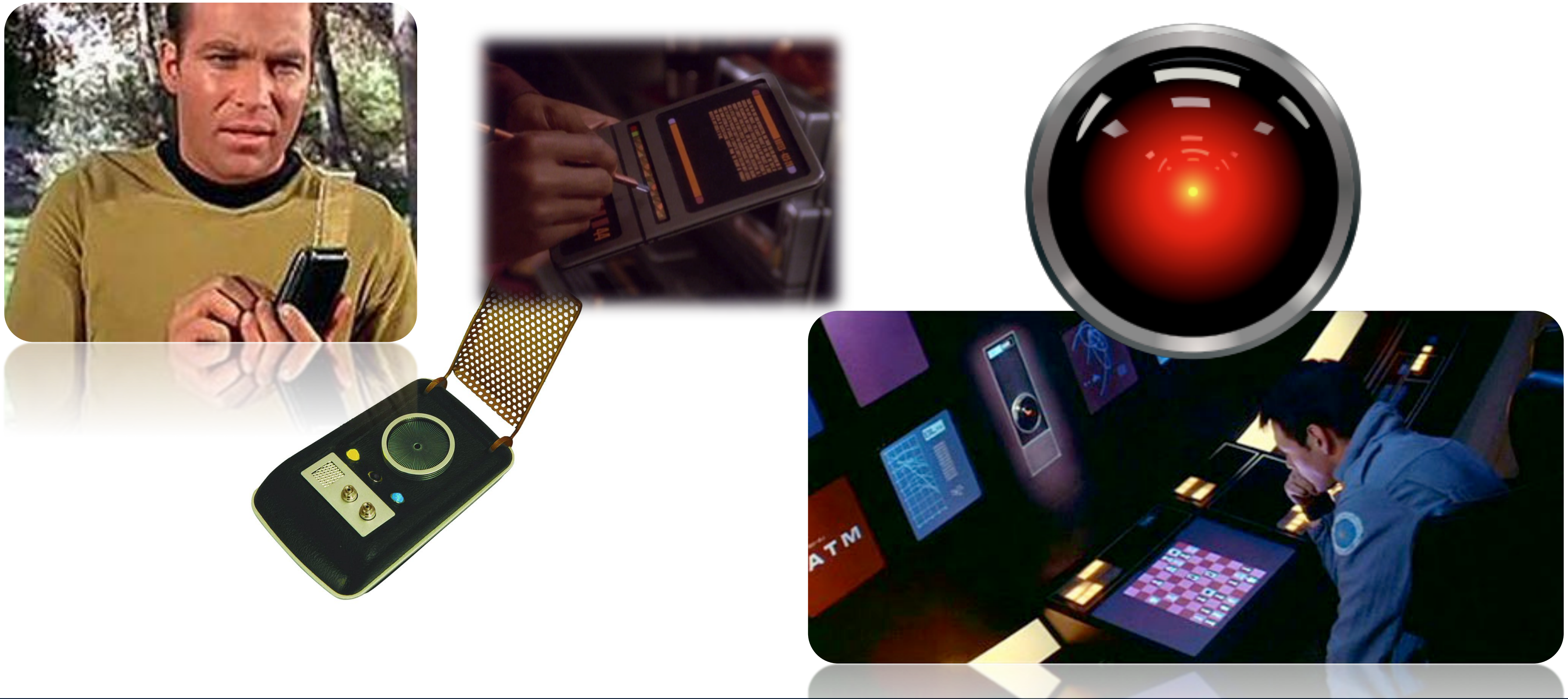
Temperature Control/Management
The AcOS refreshement: ThermOS



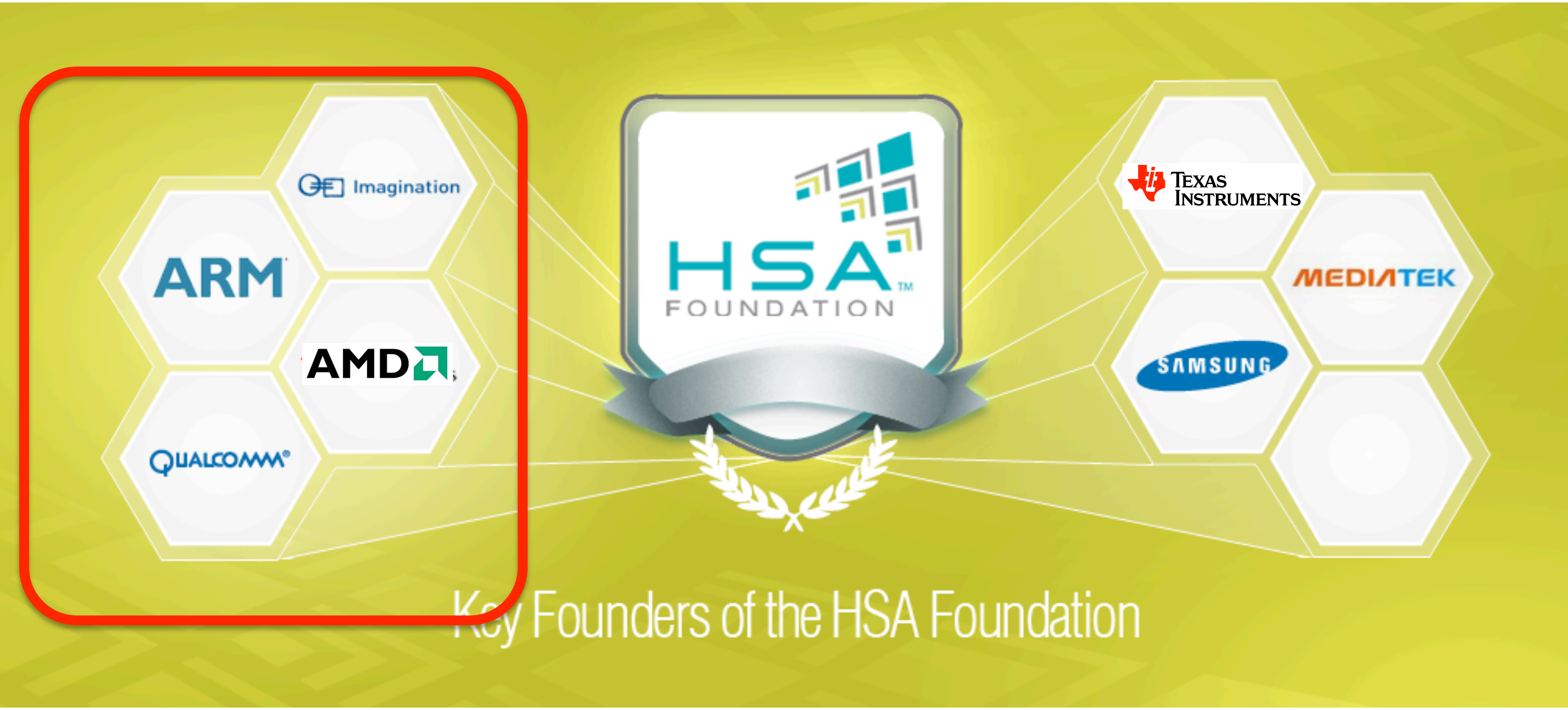
Where to go necst?

A vision

- To discover/understand the future, sometimes you have to look back at the past…



Heterogeneous System Architecture



More information available at: http://hsafoundation.com/

A global property

- Being able to adapt is not a specific domain property
  - Operating system, computer architecture, etc…
  - HPC
  - Exascale computing systems
  - Embedded systems
  - IoT
  - …

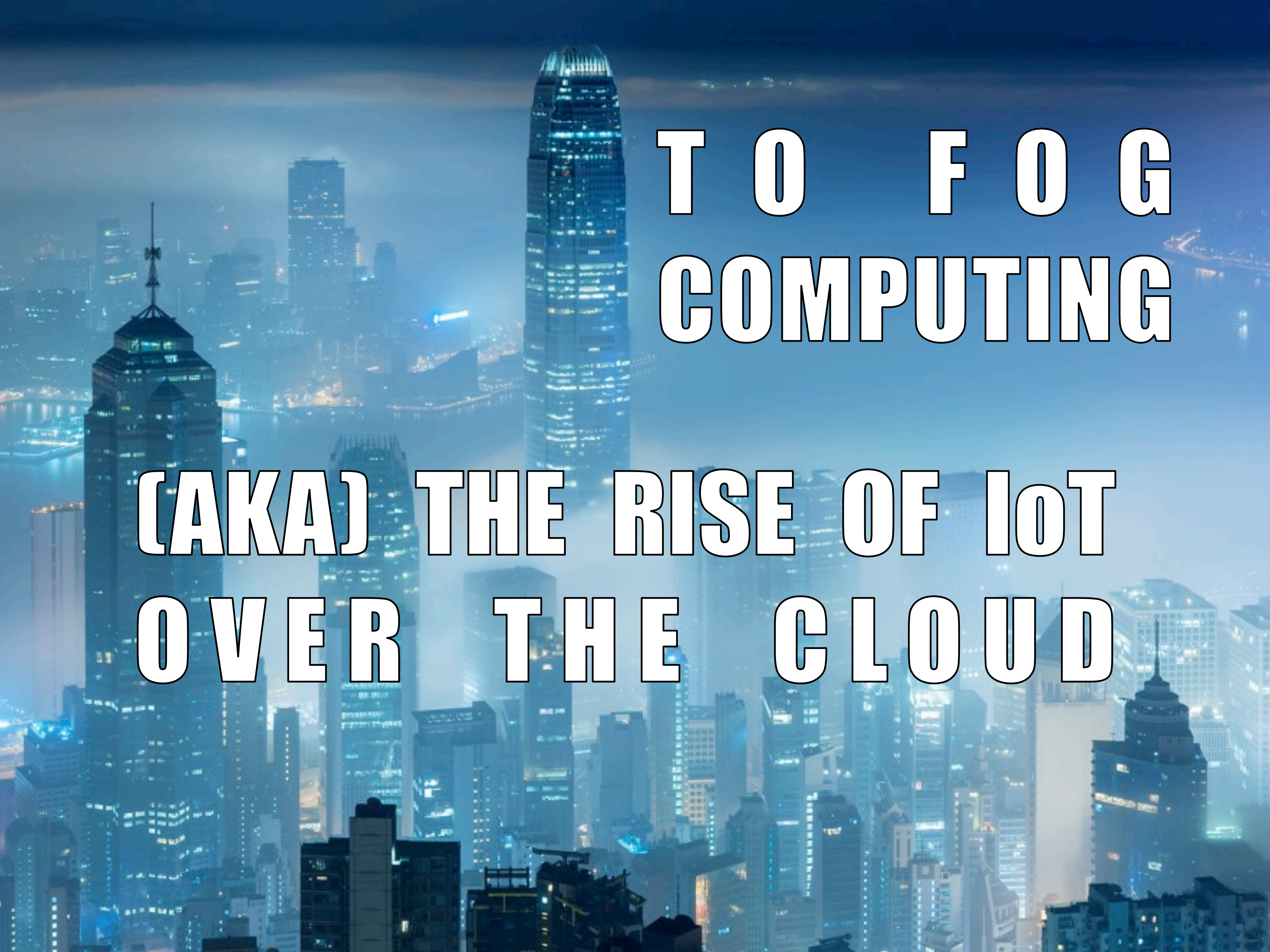
A global property

- Being able to adapt is not a specific domain property
  - Operating system, computer architecture, etc…
  - HPC
  - Exascale computing systems
  - Embedded systems
  - IoT

embedded/mobile devices and HPC/distributed/exascale computing

FROM CLOUD COMPUTING

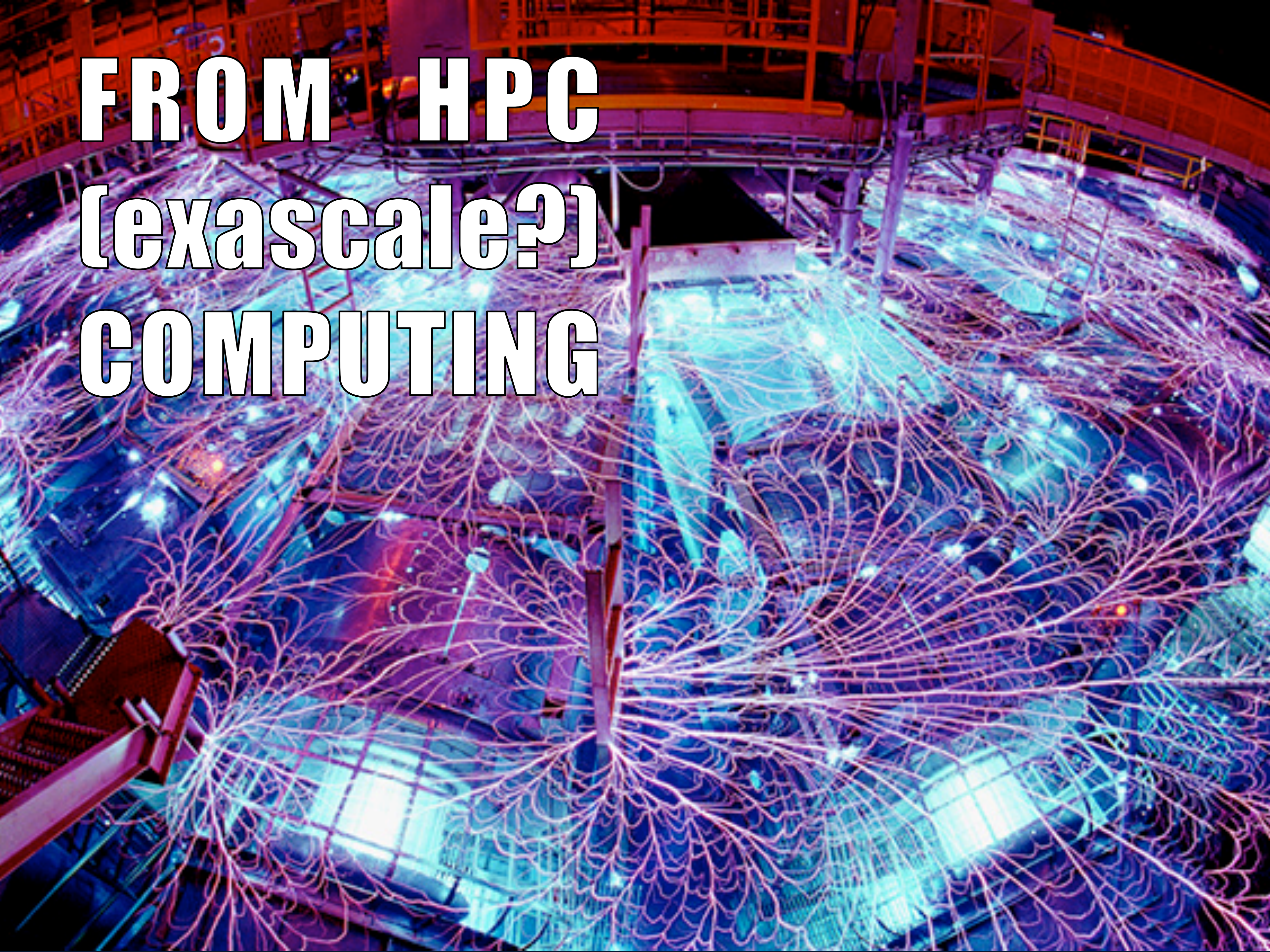
TO FOG COMPUTING

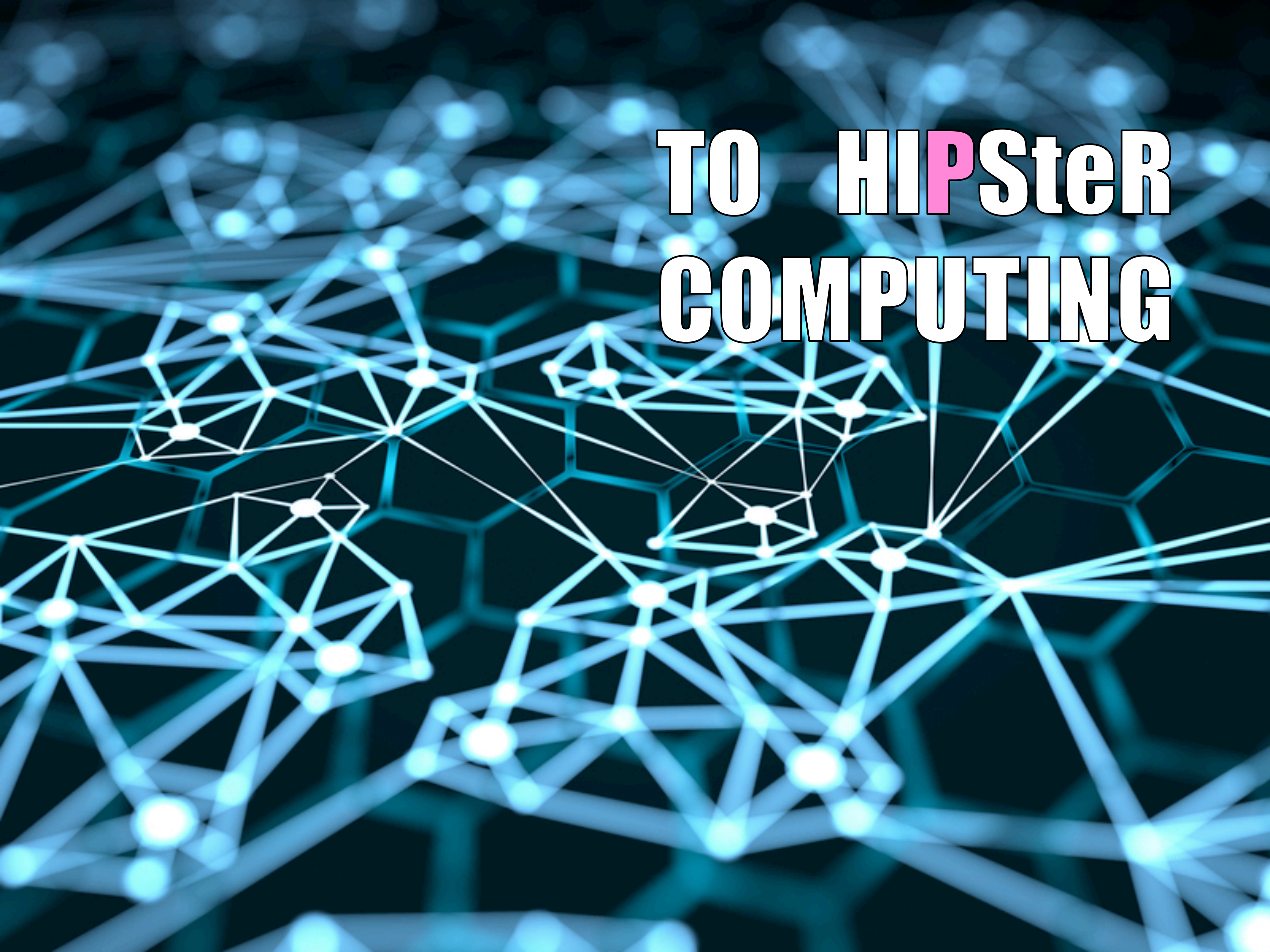


TO FOG COMPUTING

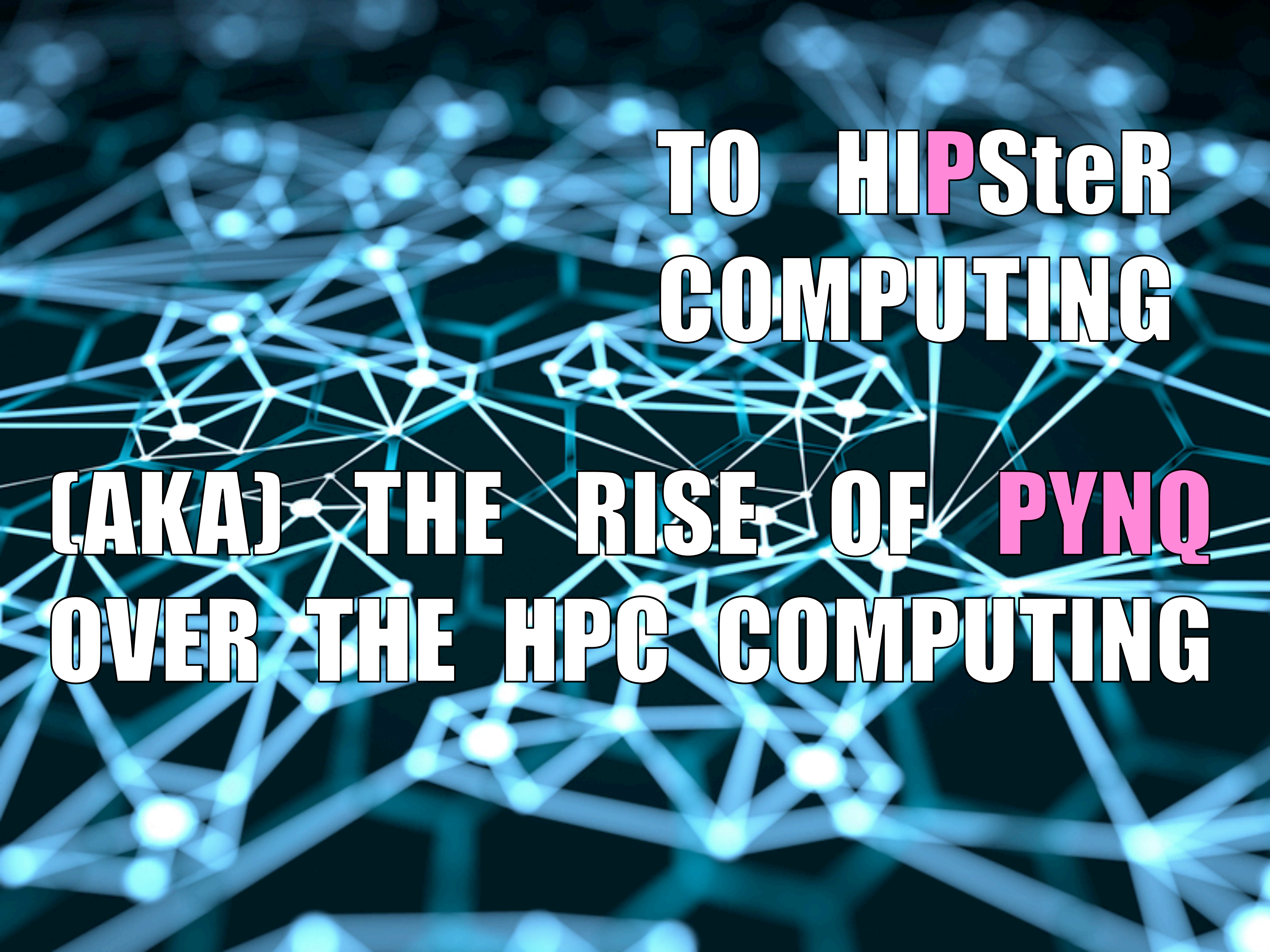
(AKA) THE RISE OF IoT OVER THE CLOUD

FROM HPC (exascale?) COMPUTING



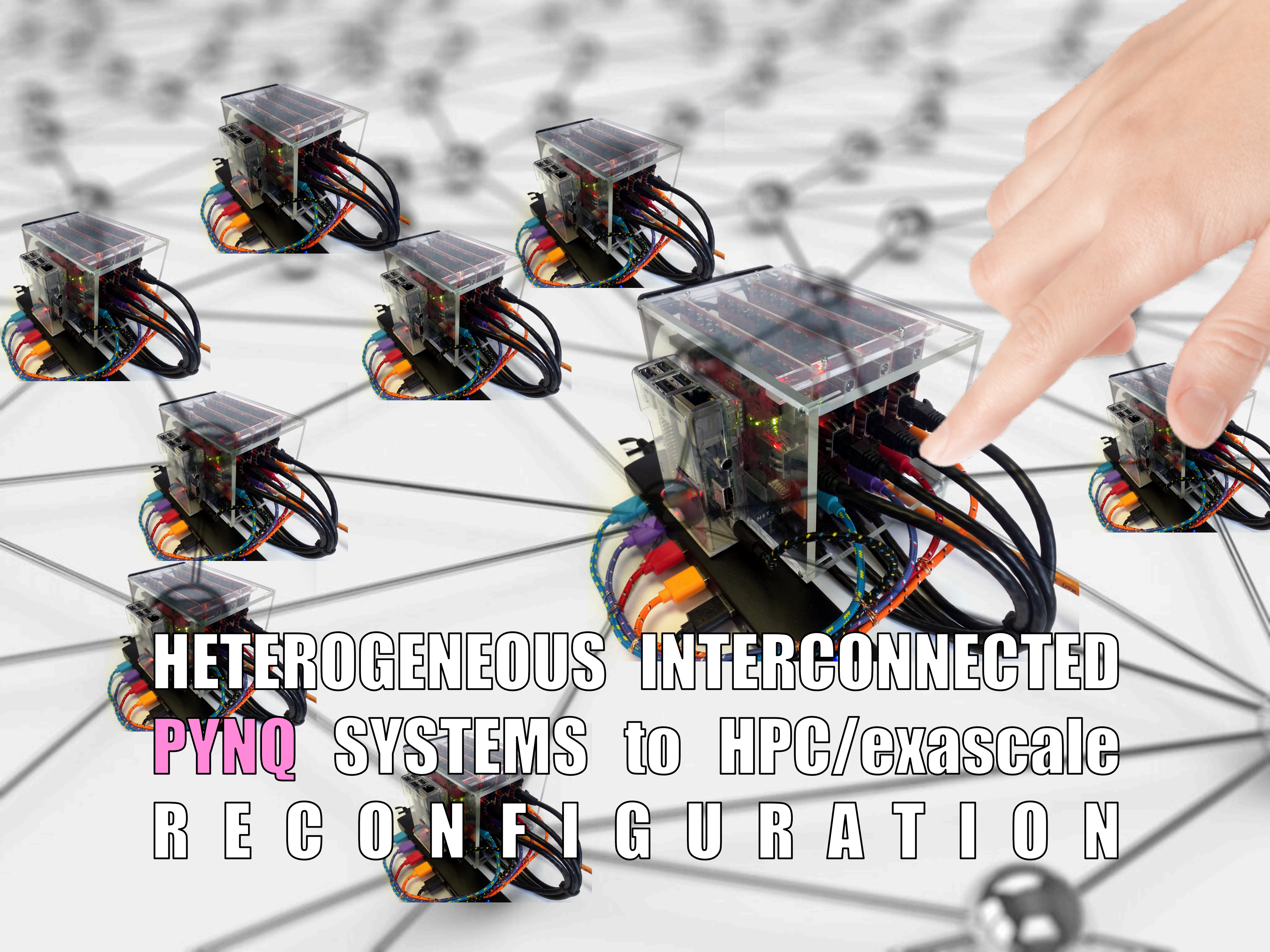


TO HIPSteR COMPUTING



TO HIPSteR COMPUTING

(AKA) THE RISE OF PYNQ OVER THE HPC COMPUTING


HETEROGENEOUS INTERCONNECTED
PYNQ SYSTEMS to HPC/exascale
RECONFIGURATION

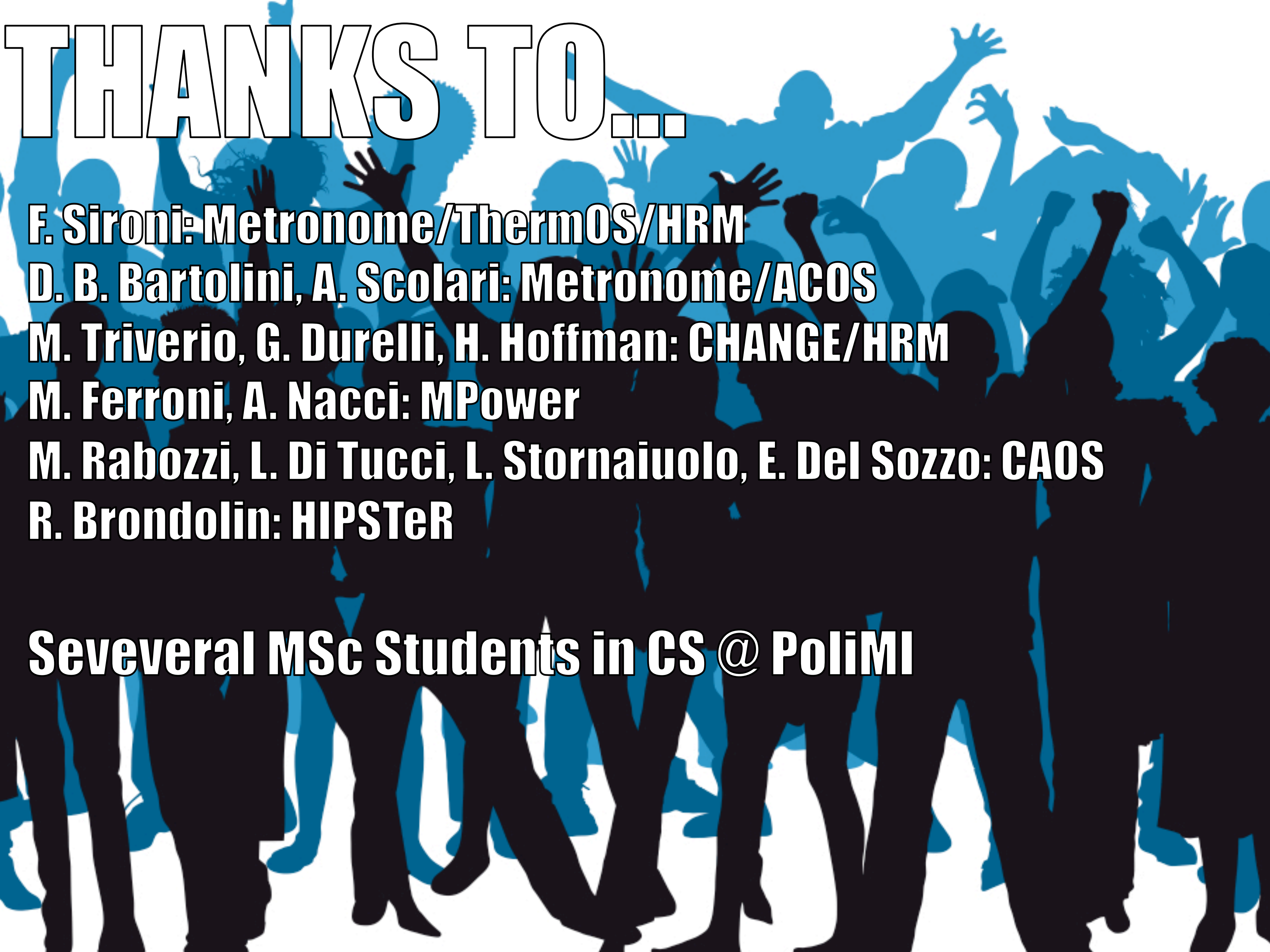
WAT.

THAT'S INSANE!

NOT COMPLETELY INSANE!

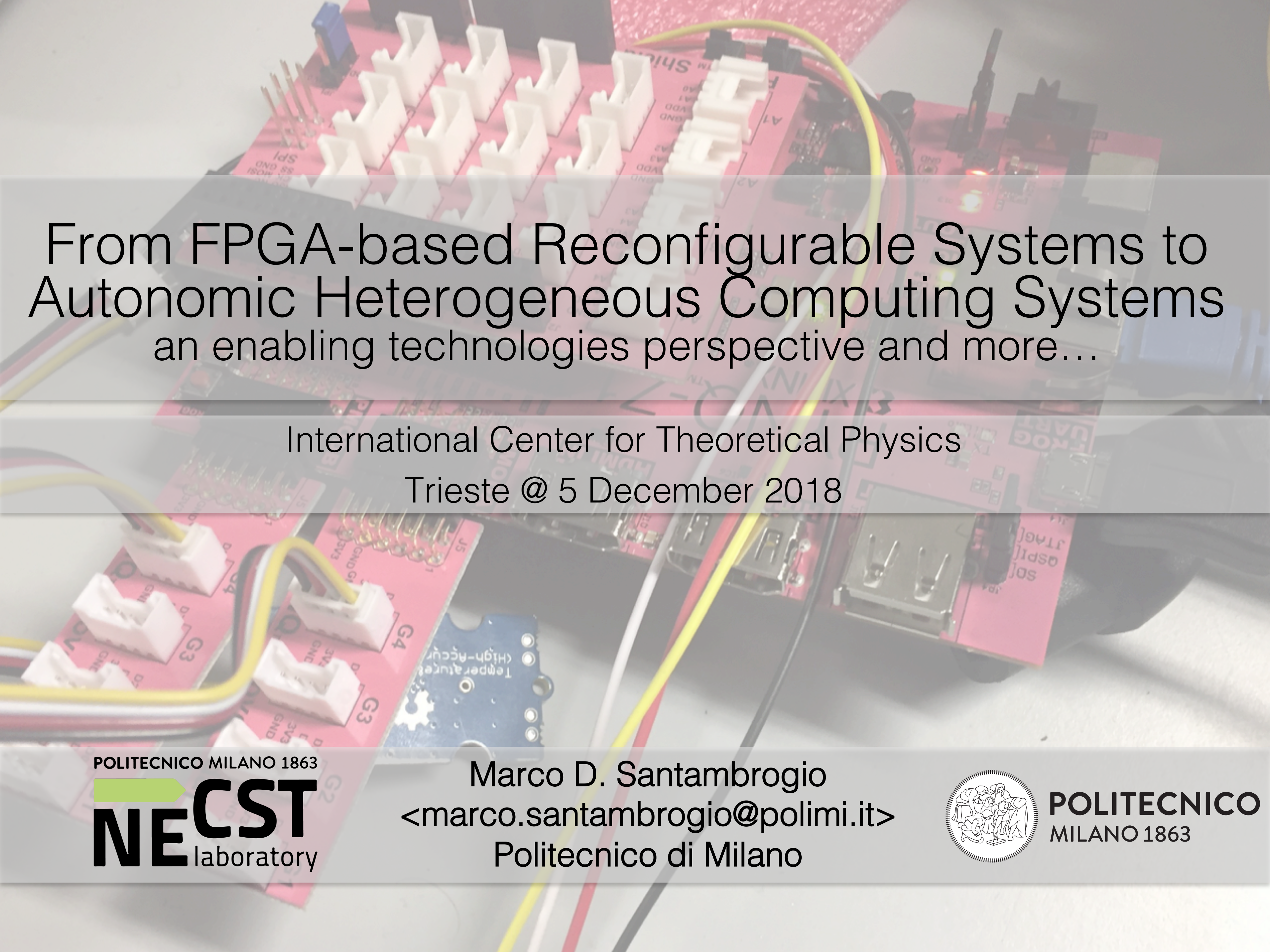
Device Name	Price	Look-up Tables	KLUT/$
snickerdoodle	$55	17,600	0.32
snickerdoodle-black	$149	53,200	0.36
PYNQ-Z1 (educational)	$65	53,200	0.81
PicoZed (7015)	$265	46,200	0.17
PicoZed (7030)	$375	78,600	0.20
Zynq MMP	$1295	218,600	0.17

THANKS TO...



F. Sironi: Metronome/ThermOS/HRM
D. B. Bartolini, A. Scolari: Metronome/ACOS
M. Triverio, G. Durelli, H. Hoffman: CHANGE/HRM
M. Ferroni, A. Nacci: MPower
M. Rabozzi, L. Di Tucci, L. Stornaiuolo, E. Del Sozzo: CAOS
R. Brondolin: HIPSTeR

Seveveral MSc Students in CS @ PoliMI



From FPGA-based Reconfigurable Systems to Autonomic Heterogeneous Computing Systems

an enabling technologies perspective and more…

International Center for Theoretical Physics

Trieste @ 5 December 2018

POLITECNICO MILANO 1863
NECST laboratory

Marco D. Santambrogio
<marco.santambrogio@polimi.it>
Politecnico di Milano

POLITECNICO
MILANO 1863