Runtime Support for Self-Awareness in Interconnected CPS Systems

Dionisios Pnevmatikatos

Dimitris Theodoropoulos

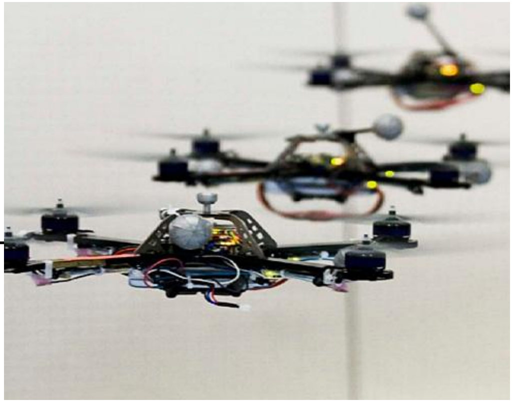
{pnevmati, dtheodor}@ics.forth.gr

Computing Systems Week 2015
May 7th, 2015, Oslo, Norway

Title Terms

- **Self-awareness**:
  - Adaptive, self-healing, knowledge of self & environment, goal-oriented
  - Deal with uncertainty => change behaviour
- **Runtime support**: System manager & optimizer
- **Interconnected CPSs:** System of linked collaborative computational elements

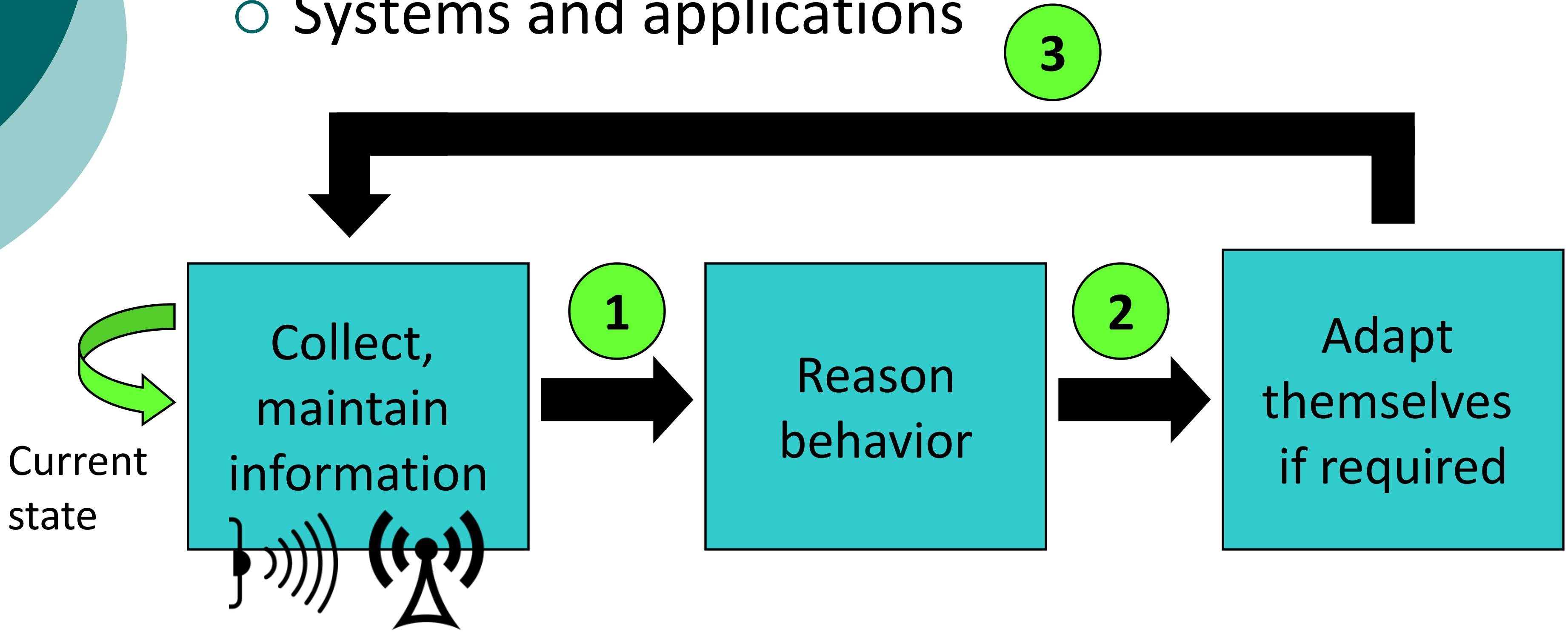
Self-aware Systems



- Domains:
  - Automotive
  - Aerospace
  - Robotics
  - Healthcare
  - Transportation
  - Energy saving

Self-aware Systems (cntd)

- Systems and applications



3

Current state

Collect, maintain information

1

Reason behavior

2

Adapt themselves if required

Surrounding environment evaluation

Self-aware Systems (cntd)

- System monitoring (self/local/global? state)
- Internal system adaptation
  - Available resources, current workload, energy budget
- External system adaptation
  - Information exchange, surrounding environmental conditions, application requirements, …
- (Self-)learning?

Self-aware Systems Projects

- AWARENESS Coordination Action (2010-2013)
- Autonomic Service-Component Ensembles (ASCENS)
- Engineering Proprioception in Computing Systems (EPICS)
- Organic computing
- Relevance and cognition for self-awareness in a content-centric Internet (RECOGNITION)
- Self-aware Pervasive Service Ecosystems (SAPERE)
- Symbiotic Evolutionary Robot Organisms(SYMBRION)
- Collective Cognitive Robots (CoCoRo)

Required Runtime support

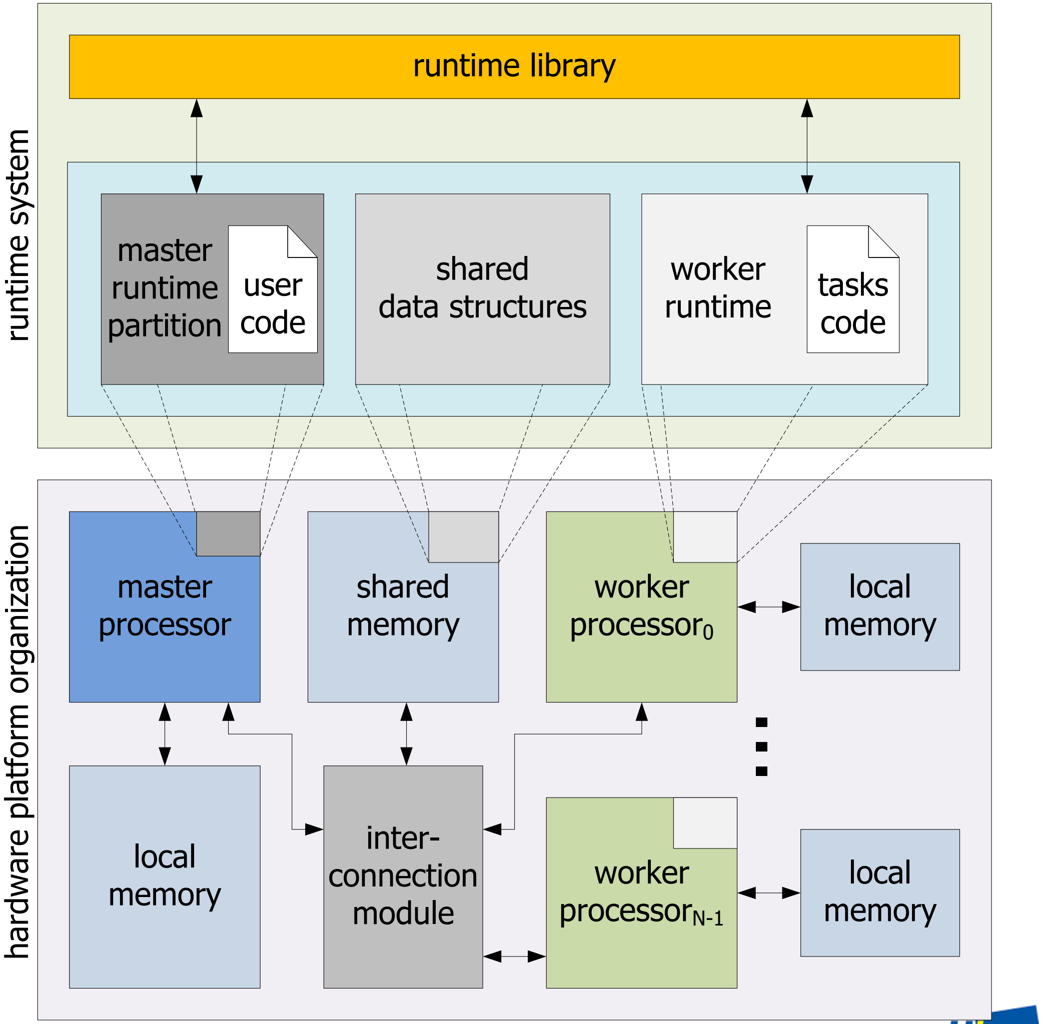
- Towards **RELIABLE** self-aware reconfigurable interconnected CPSs
  - Intelligent task scheduling
    - Software / Hardware tasks
    - Efficient tasks distribution to available nodes
  - Dynamic system reconfiguration (FPGA? Coarser-grain?)
  - Low-overhead runtime kernel
    - Evaluate current state
    - Assess surrounding environment
    - Switch operating modes when necessary
  - **Reliable functionality – Self healing**

The DeSyRe Runtime System

- Lightweight kernel for *reliable* reconfigurable MPSoCs
- Task-based programming model
  - Address faults at SW/HW, adapt QoS
- System level checkpoint / restore
- Transient errors support
  - Local task variables checkpoint task restart
- Heterogeneous, Adaptive, Supports QoS
  - Flexible expression of optimization goals

DeSyRe Runtime System (cntd)

- Master
  - Central, protected controller
- Workers
  - Do the work
  - Provide redundancy
- Opportunities for adaptation



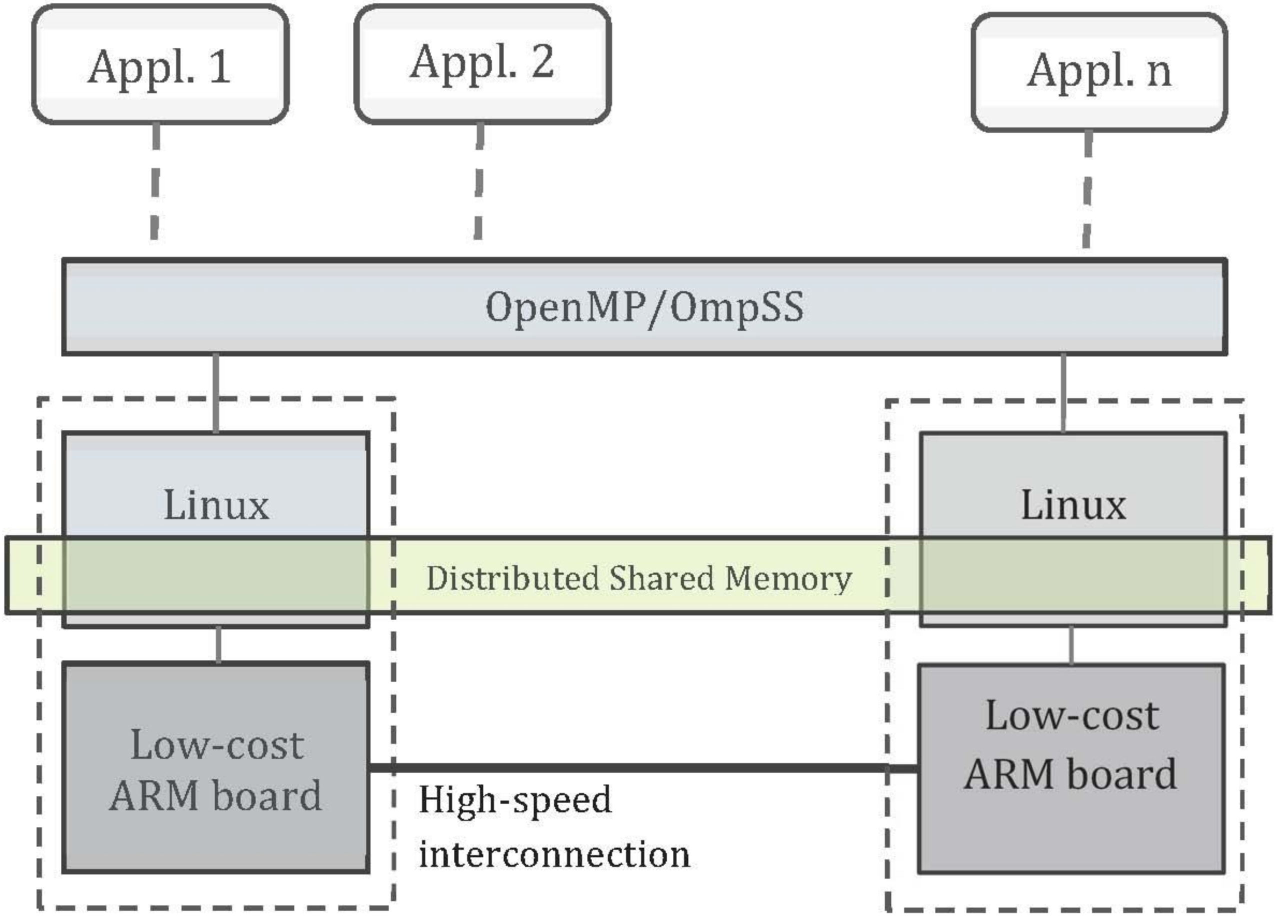
CPS: the AXIOM project

Agile, e**X**tensible, fast **I**/**O** **M**odule, started in Feb. 2015, Contract ID 645496

Aim: develop new HW/SW architectures for CPSs that heavily interact between them and humans

- Real-time system reaction
- Ample computational power
- Energy efficiency
- Modularity / scalability
- Easy programmability

AXIOM Approach



AXIOM Applications

Potential for Self-awareness

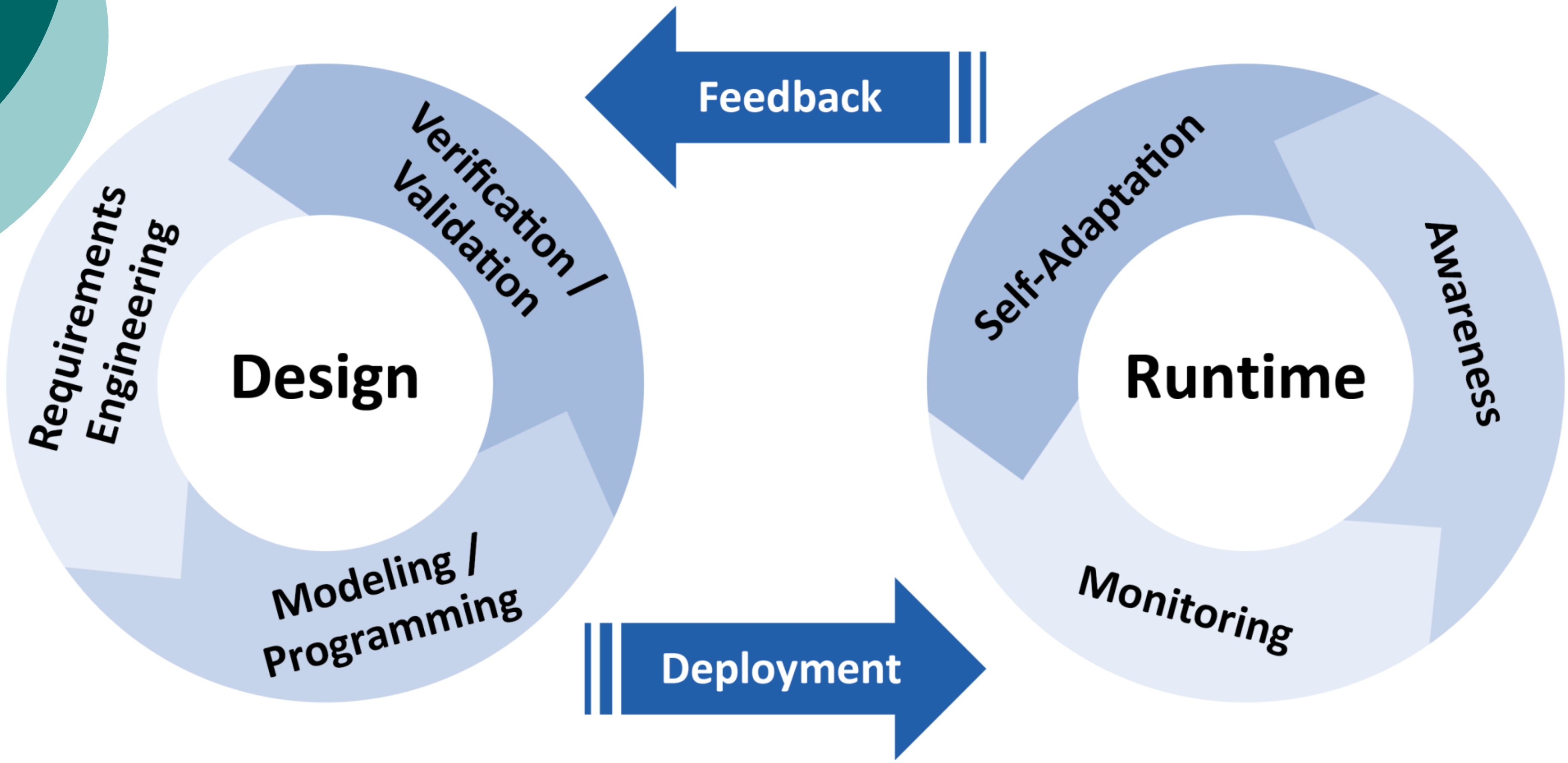
- Video-surveillance
- Smart home
- …

Light Switch
Power Plug
Remote Control
Temperature management
Preventive and interactive maintenance of infrastructures
Climate management

Extending AXIOM with self-awareness

- When adaptivity becomes awareness?
  - AI/machine learning/etc.
  - limited "palette" of options BUT intractable system level options?
- DeSyRe Runtime is a solid base for adaptation
- Flexible scheduling can accommodate learning
- “Smart” Decision making:
  - Criteria? Application Specific? Machine learning?
- How to design/debug/verify such a system?

Ascens: Ensemble Development Life Cycle



Where are we?

- Adaptable systems are hard to design
- Self-awareness support needed?
  - Provide flexibility/options
  - Provide metrics for estimating cost/benefit
  - Provide mechanisms for using chosen option
  - Debug the nodes and the entire system!
- We are not there yet!

Where are we?

- Q1: Where is this “awareness” in ES expressed? SW, HW, or else?
  - at all levels, parts of the global answer, but mostly: runtime (SW)
- Q2: achieved “self-awareness”?
  - no! But is it really necessary?
- Q3: What does “self-awareness” in ES really mean?
  - dynamic behavior control in large systems

Where are we?

- Q4: current solutions "self-aware" or just smartly designed?
  - majority: smart design
- Q5: "Self" refers to a single entity?
  - self behavior as part of system
- Q6: "Self-awareness is achieved"
  - It is not achieved
  - need tools to consider and support it in the design of complex systems

End of Presentation

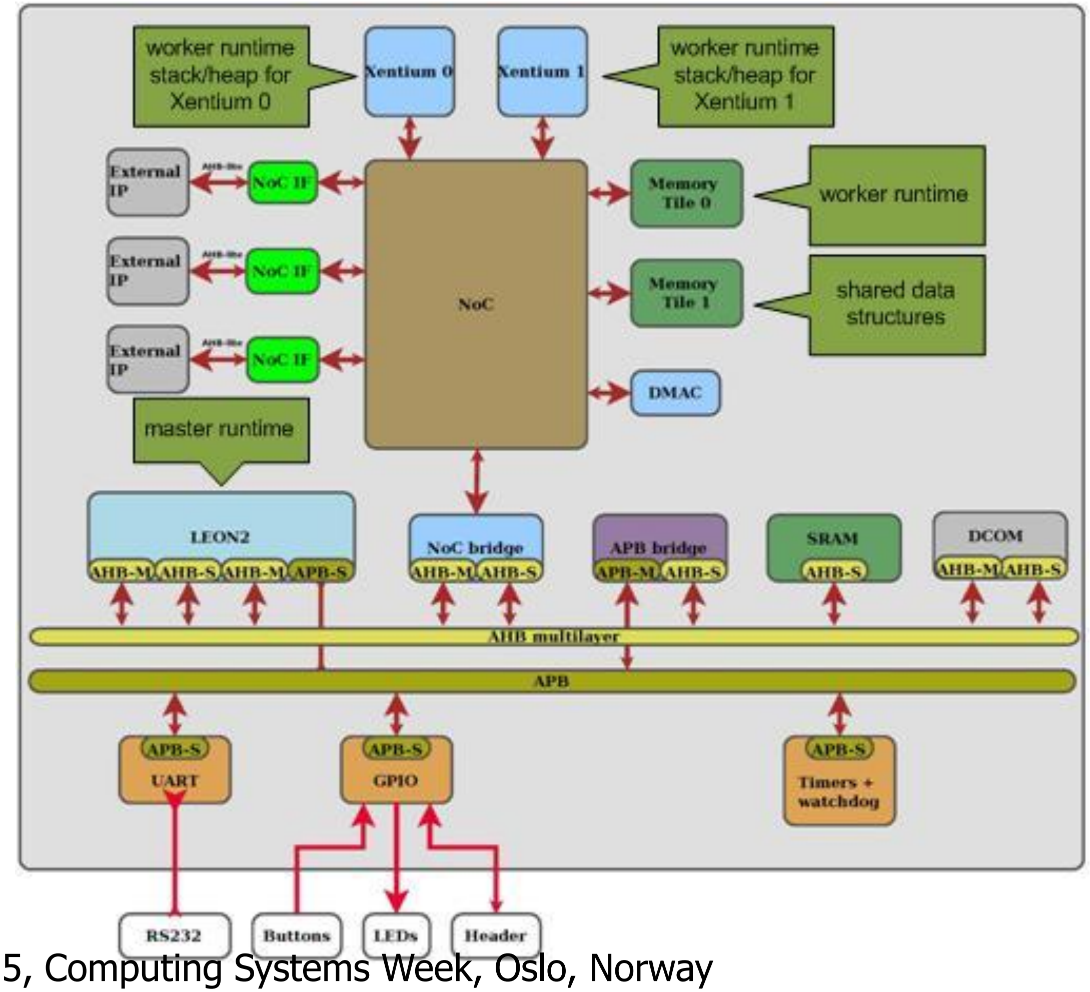
Questions / Discussion

DeSyRe Runtime System (cntd)

- Permanent faults recovery
  - Task migration to active workers
  - Alternative task implementations
- Generic structure, easily mapped to different shared-memory platforms
  - 8-core Microblaze-based implementation on ML605 FPGA board
  - Recore’s platform (Leon + 2 DSP Xentiums + 2 SiMS processors)

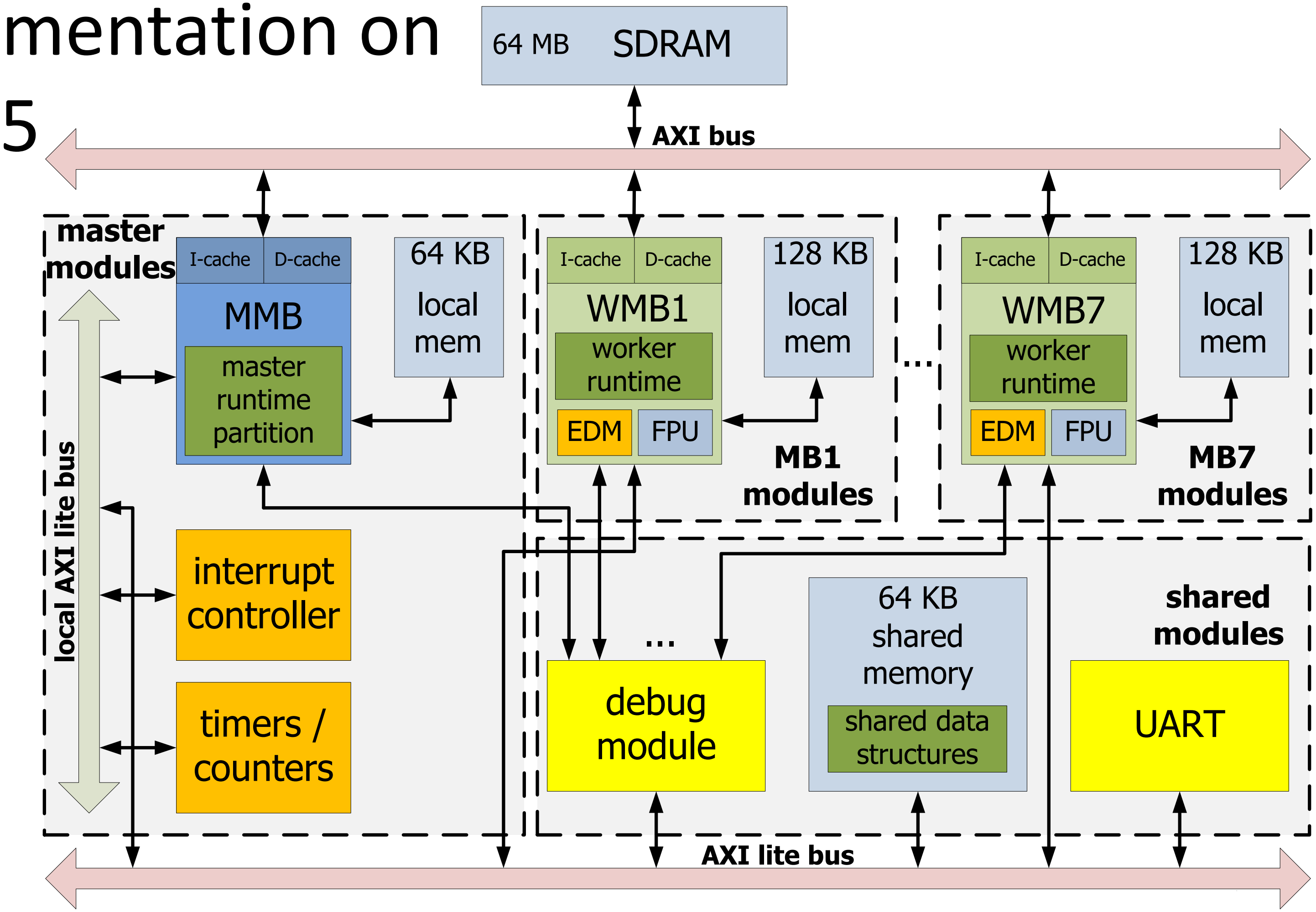
DeSyRe Runtime System (cntd)

- Implementation on Recore's platform



DeSyRe Runtime System (cntd)

- Implementation on ML605


64 MB SDRAM
AXI bus
master modules
I-cache
D-cache
MMB
master runtime partition
64 KB local mem
local AXI lite bus
interrupt controller
timers / counters
I-cache
D-cache
WMB1
worker runtime
EDM
FPU
128 KB local mem
MB1 modules
…
I-cache
D-cache
WMB7
worker runtime
EDM
FPU
128 KB local mem
MB7 modules
…
debug module
64 KB shared memory
shared data structures
shared modules
UART
AXI lite bus