Hardware / Software / Analog System Partitioning with SysML and SystemC-AMS

Daniela Genius and Ludovic Apvrille

daniela.genius@lip6.fr

Paris Sorbonne University

ERTS 2020

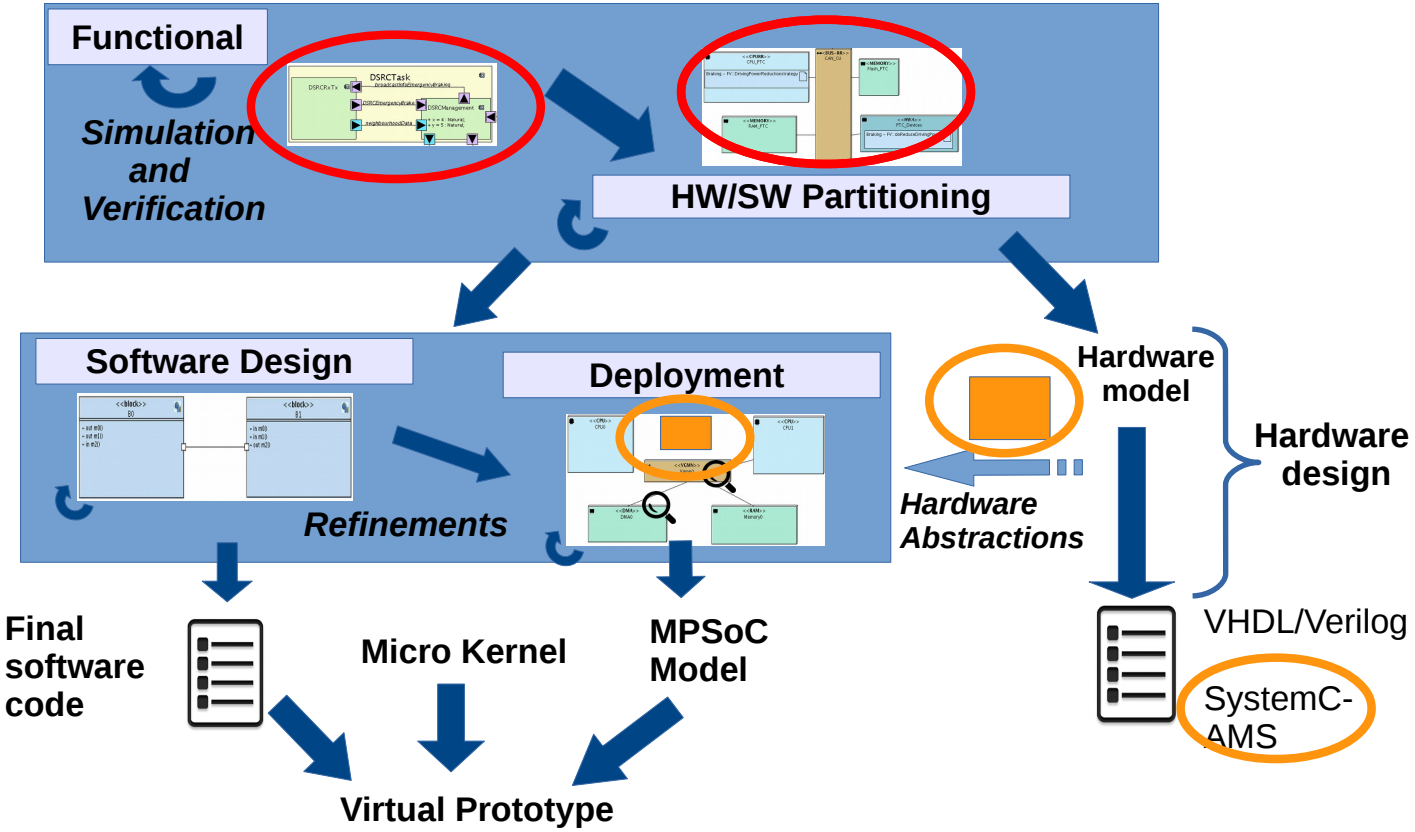
Limitations of MDE for analog/mixed-signal systems

- Model-driven approaches are generally limited to digital parts
- Virtual prototyping and co-simulation fo cyber-physical systems
- Rarely on high abstraction level, before partitioning

Related Work: Virtual Prototyping and Co-Simulation

- Modelica/Functional Mockup Interface (Blochwitz et al. 2011)
- Ptolemy II (Berkeley 2014)
- Metro II (Davare et al. 2007)
- Capella/Arcadia (Polarsys)
- MARTE with generation for Simics (Taha et al. 2010)
- MDGen (SODIUS)
- AADL (Feiler et al. 2012)

HW/SW partitioning and Code generation



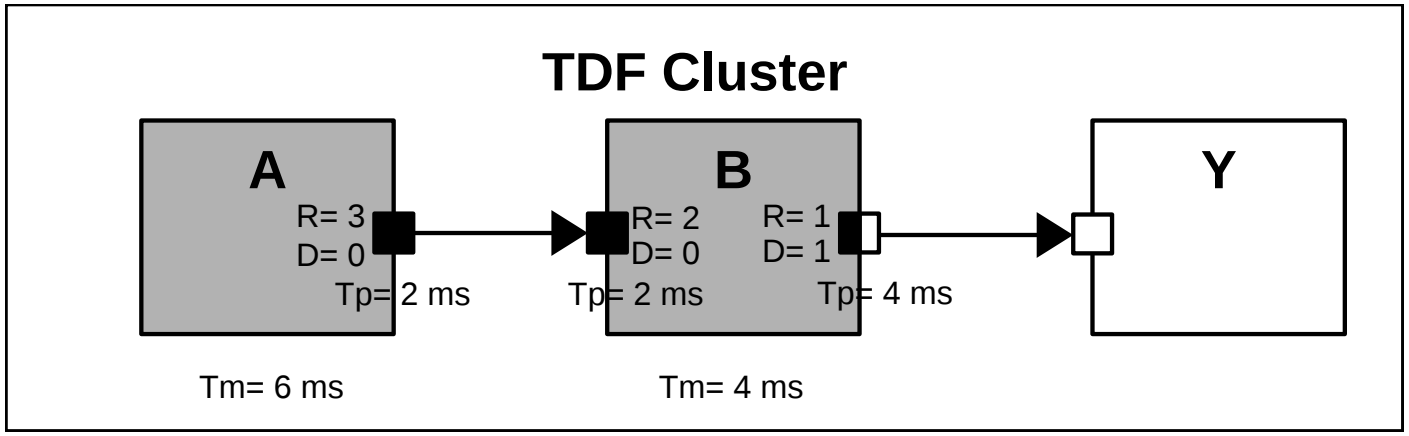
SystemC AMS

- SystemC-AMS extensions describes an extension of SystemC with AMS and RF features
- Different Models of Computation, among them Discrete Event (DE) and Timed Data Flow (TDF)
- About to become a standard, new User's Guide released early january 2020
- Commercialized by Coseda, spin-off of Fraunhofer IIS-EAS Dresden
- Co-simulaton between analog (SystemC AMS) and digital (SystemC) parts of the virtual prototype [RAPIDO 2019]

Timed Data Flow (TDF)

- Module Timestep (**Tm**): module's acitvation period
- Port Rate (**R**): read or write fixed number of data samples
- Port Timestep (**Tp**): period during which each port of a module is activated/time interval between two samples
- Port Delay (**D**): store a given number of samples on each activation, read or written on next activation

TDF Example



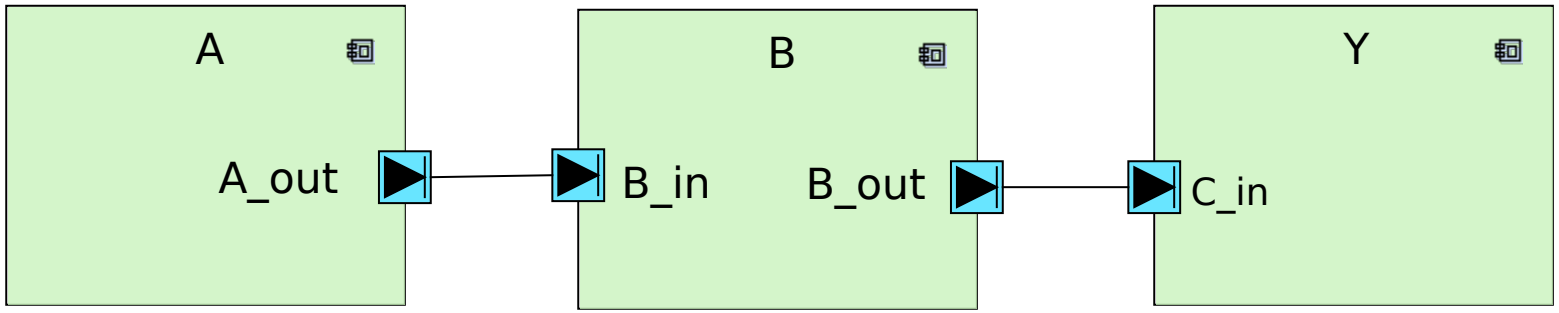
- DE modules: white boxes
- TDF modules: gray boxes
- Ports: black squares (TDF) white squares (DE)
- Converter ports: black and white squares

Modeling and verification approach

- Integration of analog aspects into HW/SW partitioning
  - Represent TDF modules in SysML-like style
  - DE modules can be directly translated (loops, channel read/write, duration estimation)
  - Capture behavior of each cluster with UML activity diagram
- Representation of analog components in architecture and mapping diagram
- Generation of abstract simulation code in C/C++

Modeling clusters: SysML block diagrams

- Blocks connected through ports featuring channels for exchanging data
- Control information: events and requests



Modeling cluster behavior: activity diagrams

- Choices directly translated into guarded branch control structures
- *Cluster Timestep*: complexity operator in activity diagram
  - Estimated or derived from schedule of existing TDF model
- Port *Rate*: number of data samples written to/read from a channel
- Behavior of cluster captured within a *loop forever*

Translation

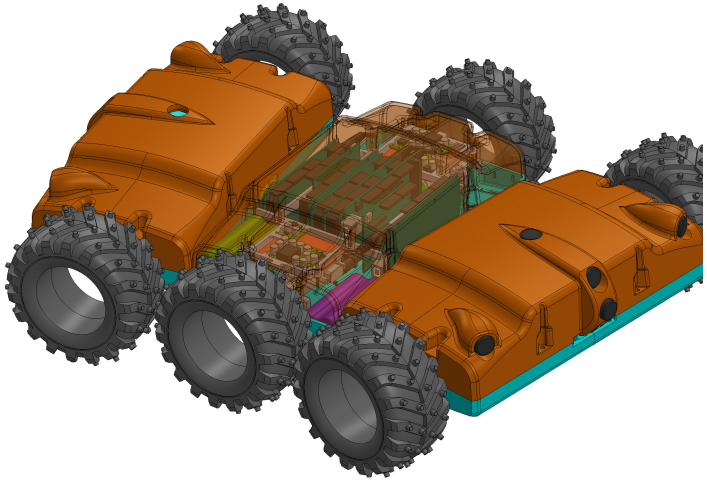
Activity Diagram		SystemC-AMS	TDF	DE
Control operators	Loop for ever	void processing(){...} void main_func(){...}	×	×
	for(i=0;i<N;i = i+1) inside loop exit loop	for(i=0; i<N; i=i+1){...}	×	×
	[guard0] [guard1] [guard2]	if(guard0){...} elsif(guard1){} else{...}	×	×
Complexity operator	N time units	module.set_timestep(N,unit);	×	
Communication operators	chan out(N)	out.write(); port.set_rate(N);	× ×	×
	chan in(N)	in.read(); port.set_rate(N);	× ×	×
	evt out()	out.notify();		×
	evt in()	in.wait();		×

Simulation and formal verification

- Mapping view: analog components are modeled as hardware accelerators
- Diagrams are converted into C++ before being simulated or formally verified
- Predictive Simulation engine: each processing element advances at its own pace until system event (data transfer, synchronization event, etc.) invalidates current transactions

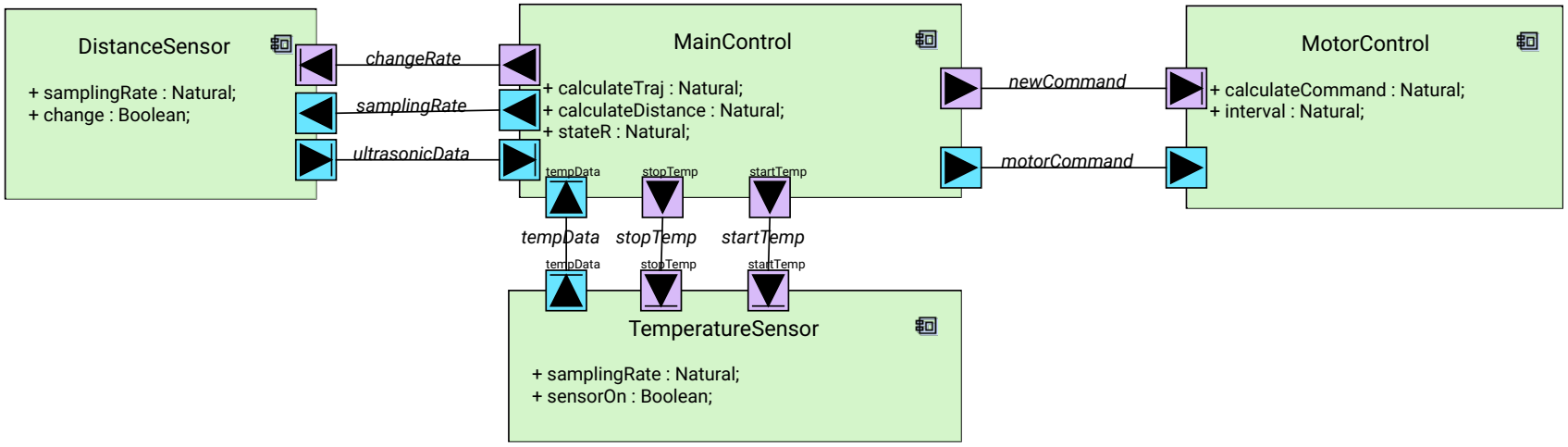
Case Study: Rover

Autonomous vehicle for disaster relief efforts (earthquake)

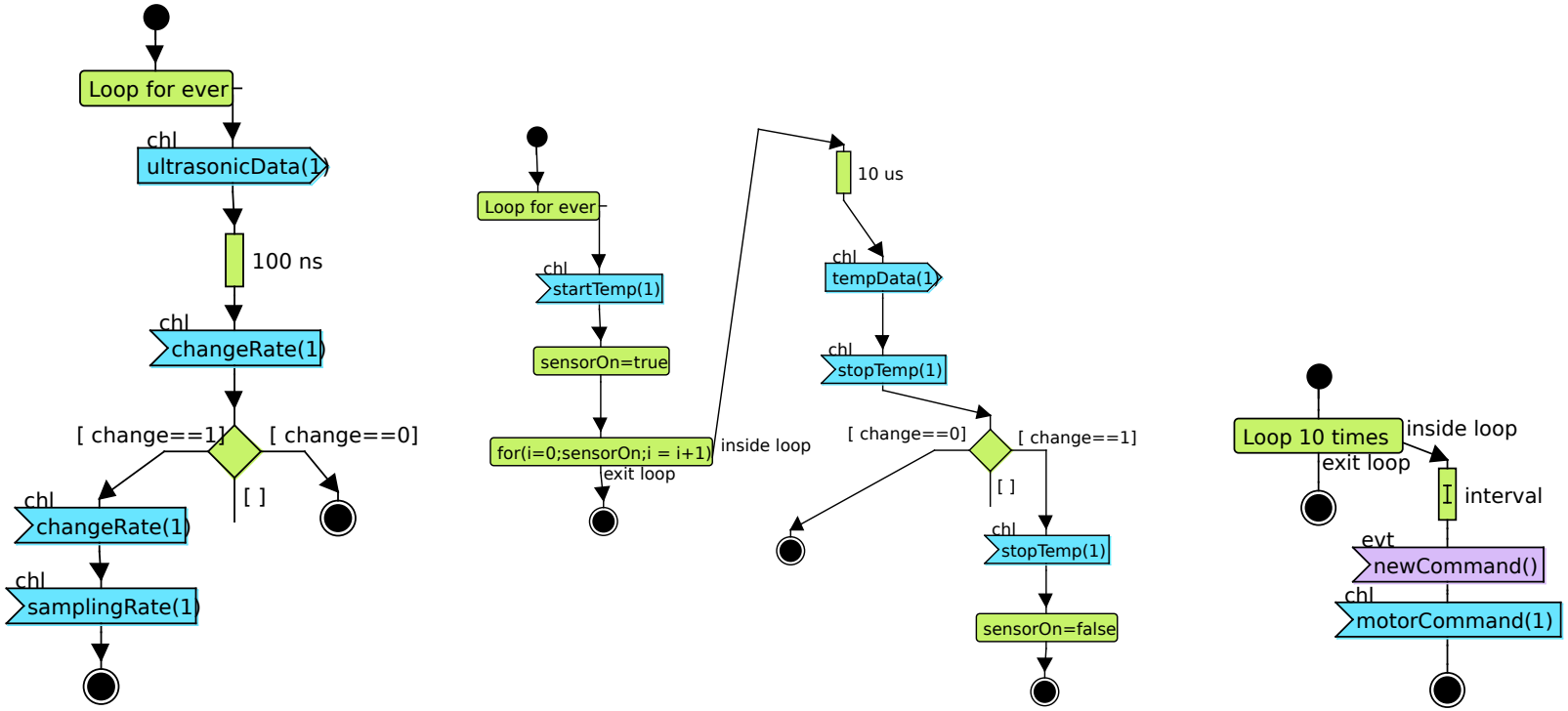


- Telemetric sensors to detect obstacles and navigate terrain autonomously
  - No obstacles in proximity → decrease sampling rate
  - Obstacle detected in close proximity → increase sampling rate
- Temperature and pressure sensors
- Avoid collisions → set time frame → impose maximal latency

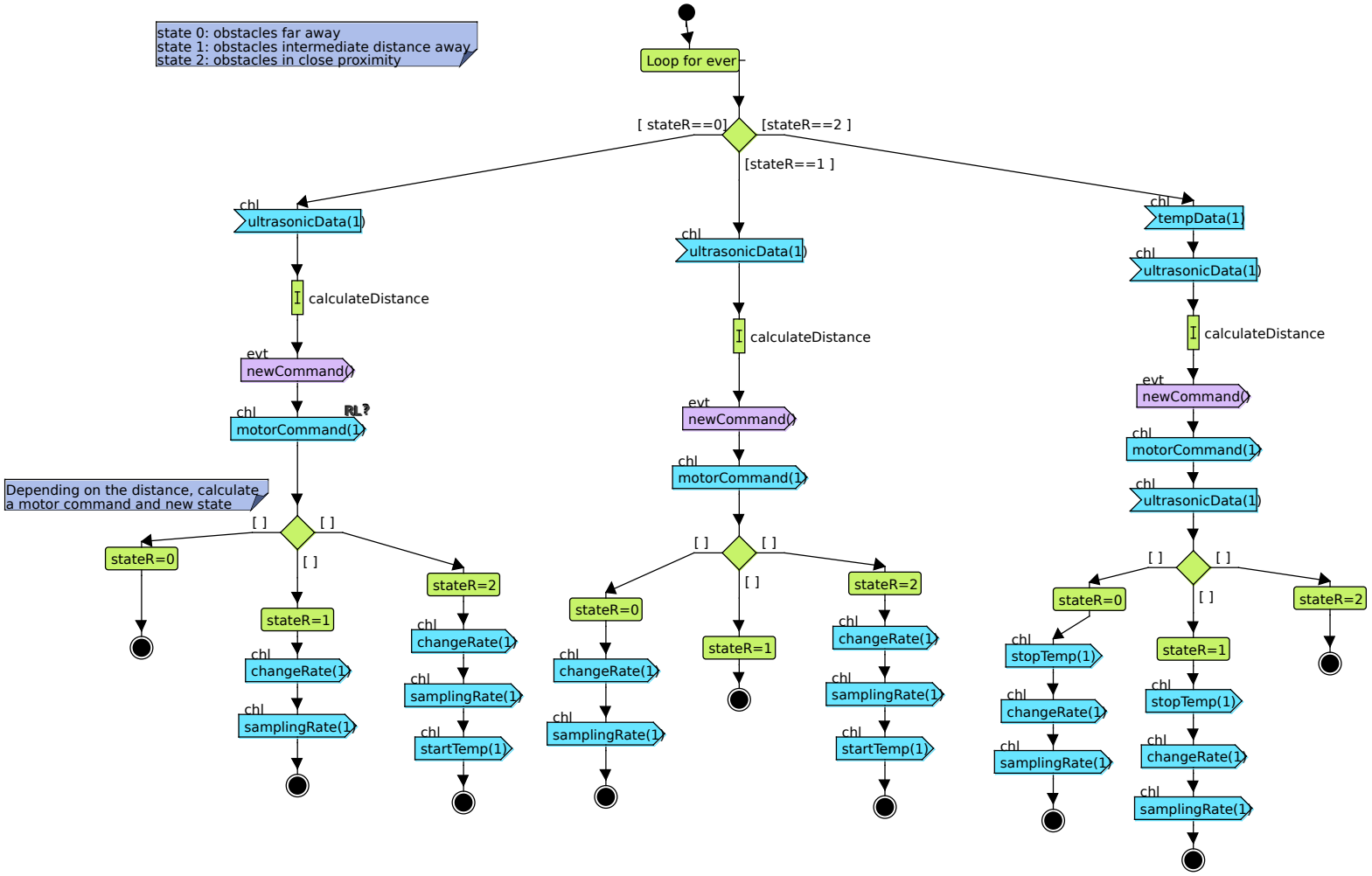
Functional view of the rover



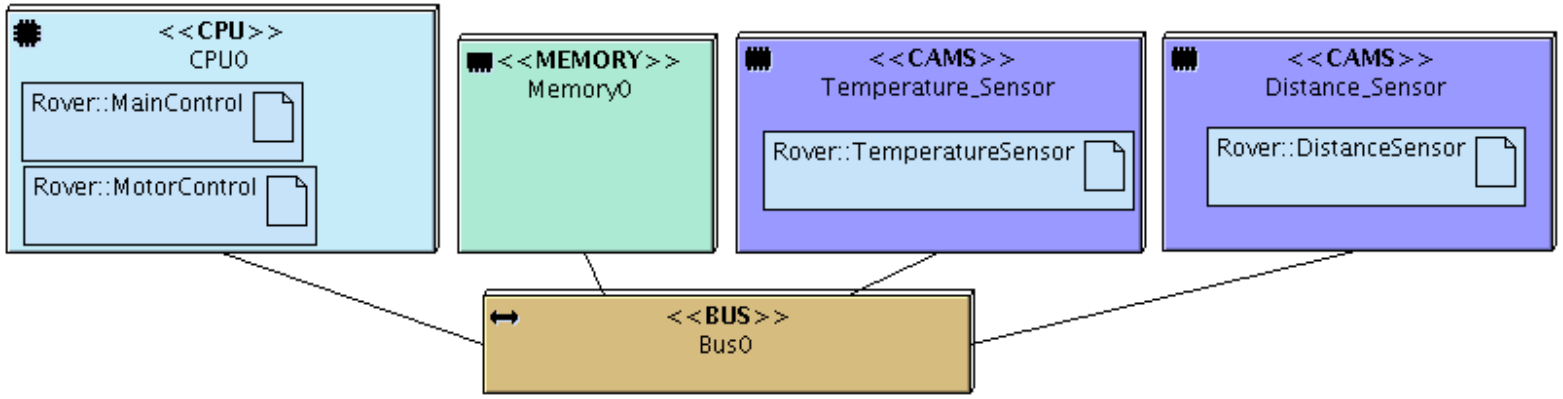
Activity Diagrams: Sensors and Motor Control



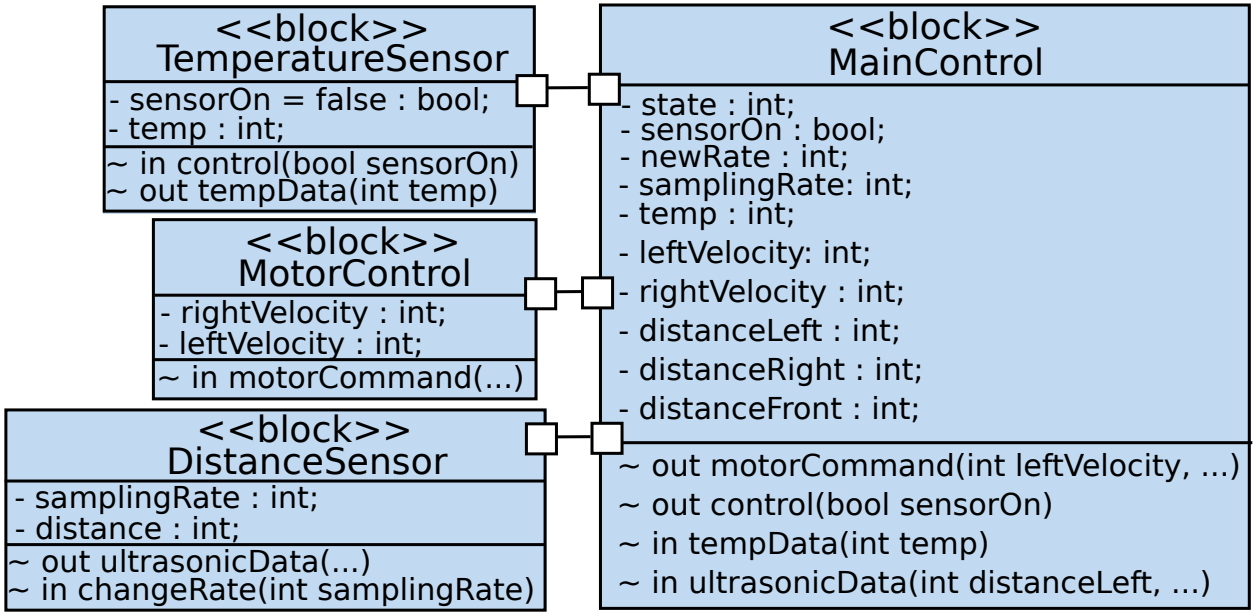
Activity Diagram: Main Control



Partitioning level architecture and mapping diagram

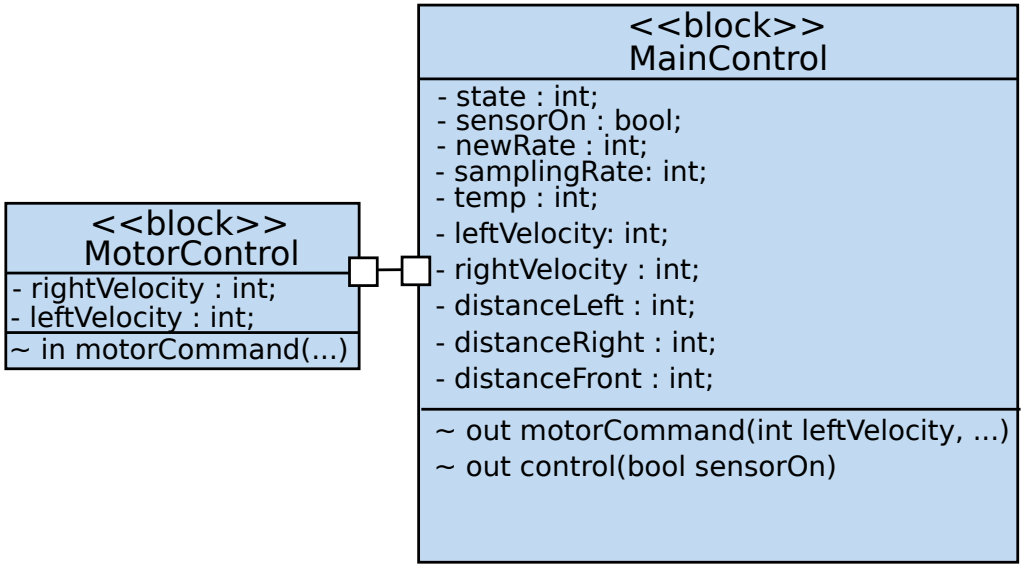


Software Block Diagram

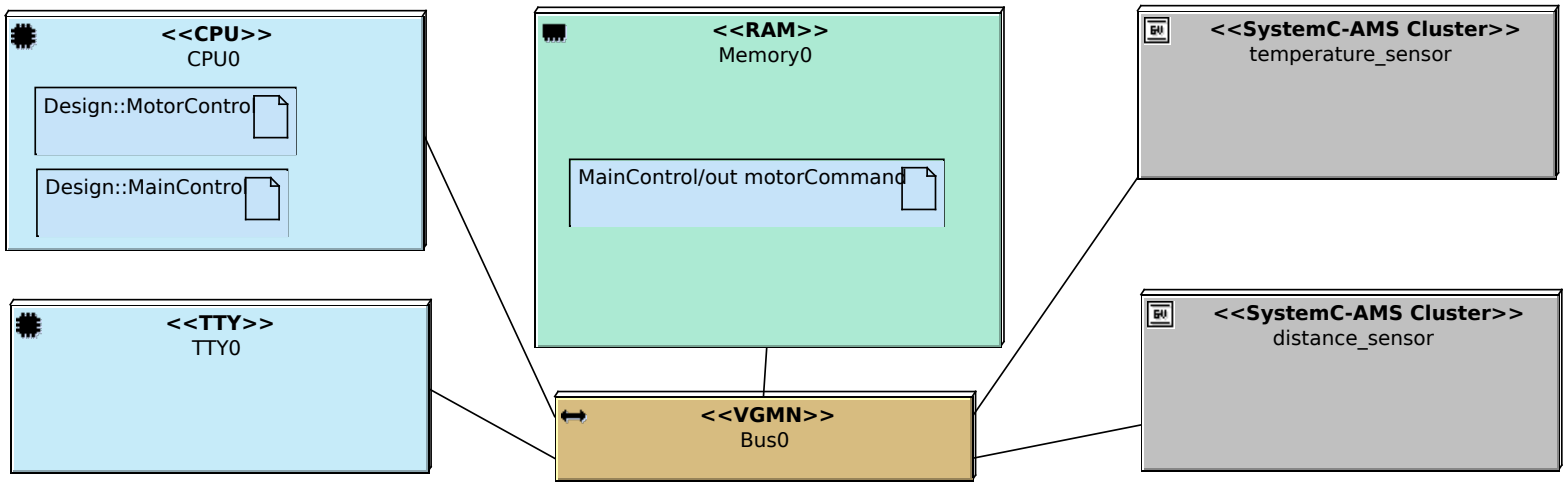


Software Block Diagram (AMS)

- AMS components (which are not software) modeled in separate SystemC-AMS panels
- Communication channels between digital and analog modules handled differently (co-simulation causality problems, Cortès Porto 2019)



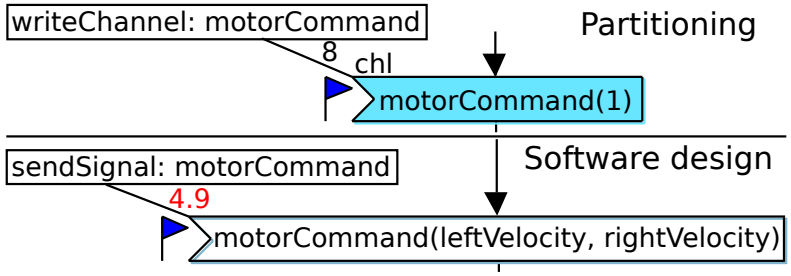
Extended deployment diagram



Feedback of results from virtual prototype

Requirements specified by designer

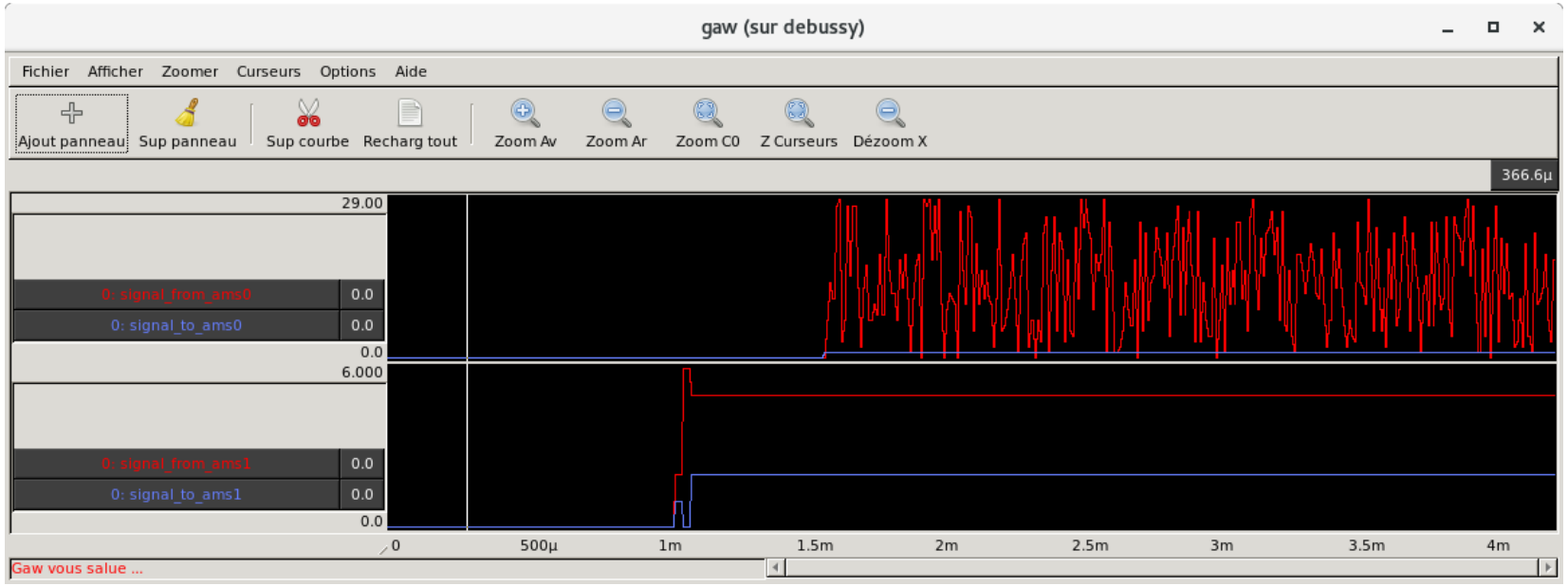
- Deviations more than percentage fixed beforehand, requirements not met: marked in red after simulation



Example: latency

- Deviation on the software design level *rightarrow* correct partitioning assumptions
- More realistic simulation of sensors → deviations reduced by factor 2 to 4 for case study

Analog traces



- One trace file per TDF cluster
- Trace with Gnu Analog WaveViewer (GAW, free software)

Conclusion and Future Work

Contributions

- Take into account digital and analog aspects of an embedded system from the first modeling phases onwards
- Cluster timestep used to obtain more accurate estimation of execution time for an operation

What's next?

- Formalization and refinement from cluster timestep (ModelsWard 2020)
- Larger industrial case studies (AQUAS H2020 projet)
- Application to medical apliances (EchOpen project 2019, master's projects with ICAN)
- Simulation speed: generate Transaction Level prototype

Thank You!

URLs

TTool: ttool.telecom-paris.fr

SoCLib: www.soclib.fr

SystemC AMS: www.accellera.org

Main Control State Machines