Design Refinement of Embedded Analog/Mixed-Signal Systems with SystemC AMS extensions

Prof. Dr. habil. Christoph Grimm

Chair Embedded Systems

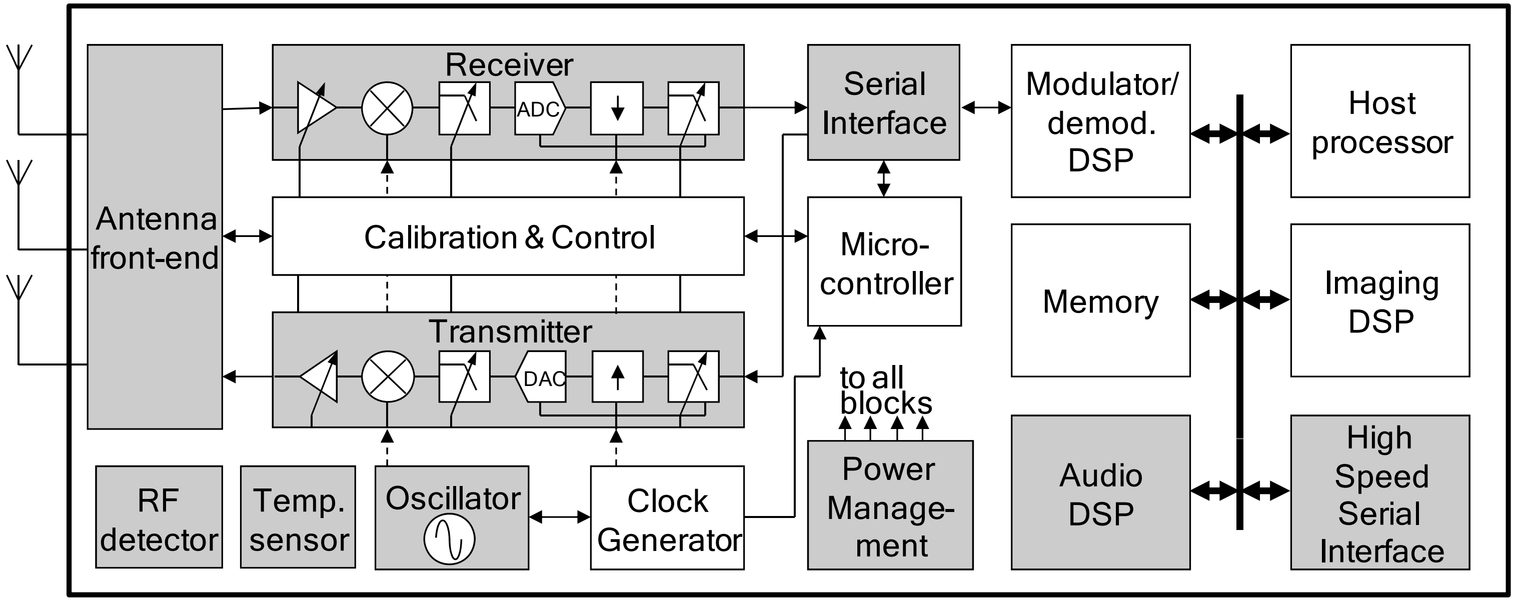
Vienna University of Technology

*partially supported by

EC FP 6 „ANDRES“, No. IST-5-033511

MEDEA+ Project GEODES/FI-IT SNOPS

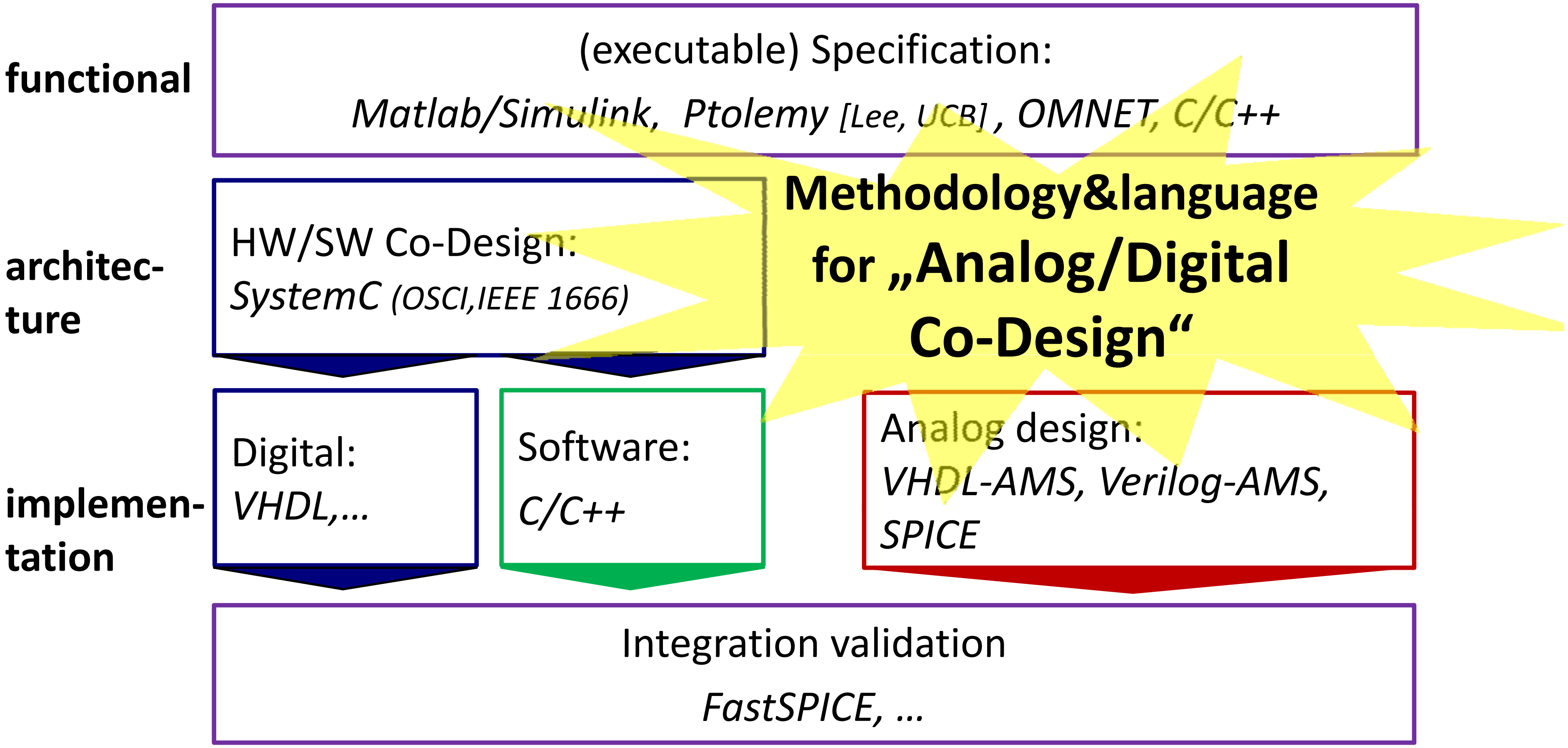
Beyond Moore: E-AMS



[Grimm/Barnasconi/Vachoux/Einwich: An Introduction to Modeling Embedded Analog/Mixed-Signal Systems using SystemC AMS Extensions. OSCI, June 2008]

- Tightly interwoven **DSP-software** and **analog circuit** functionality: **E**mbedded **A**nalog/**M**ixed-**S**ignal System (E-AMS)

Design of E-AMS


functional
(executable) Specification:
Matlab/Simulink, Ptolemy [Lee, UCB], OMNET, C/C++
architec-ture
HW/SW Co-Design:
SystemC (OSCI,IEEE 1666)
Methodology&language for „Analog/Digital Co-Design"
implemen-tation
Digital:
VHDL,…
Software:
C/C++
Analog design:
VHDL-AMS, Verilog-AMS, SPICE
Integration validation
FastSPICE, …

Design refinement of E-AMS systems

1.) SystemC AMS extensions

2.) Design refinement of E-AMS

3.) Outlook

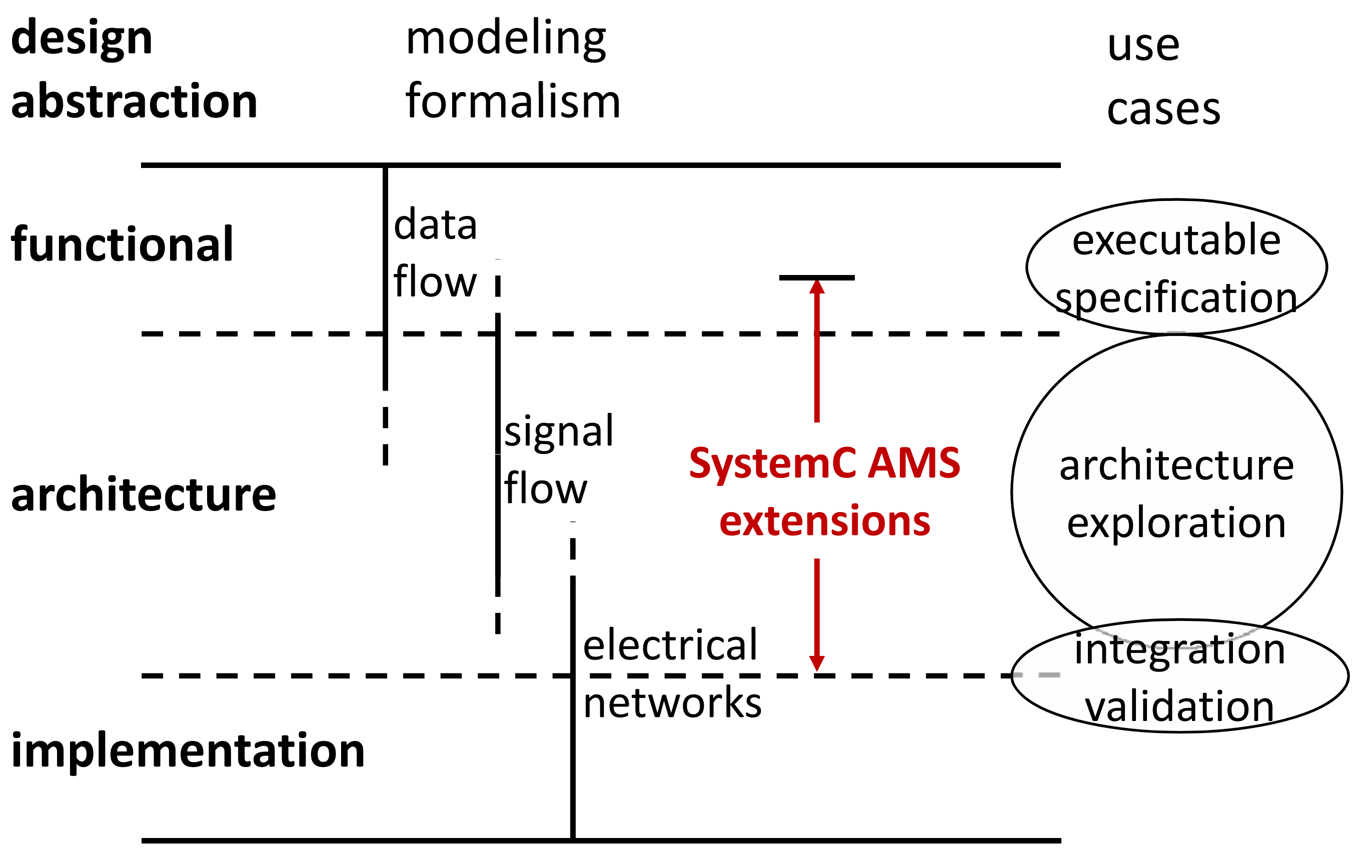
SystemC AMS extensions

- OSCI AMS WG driven by academia and industry

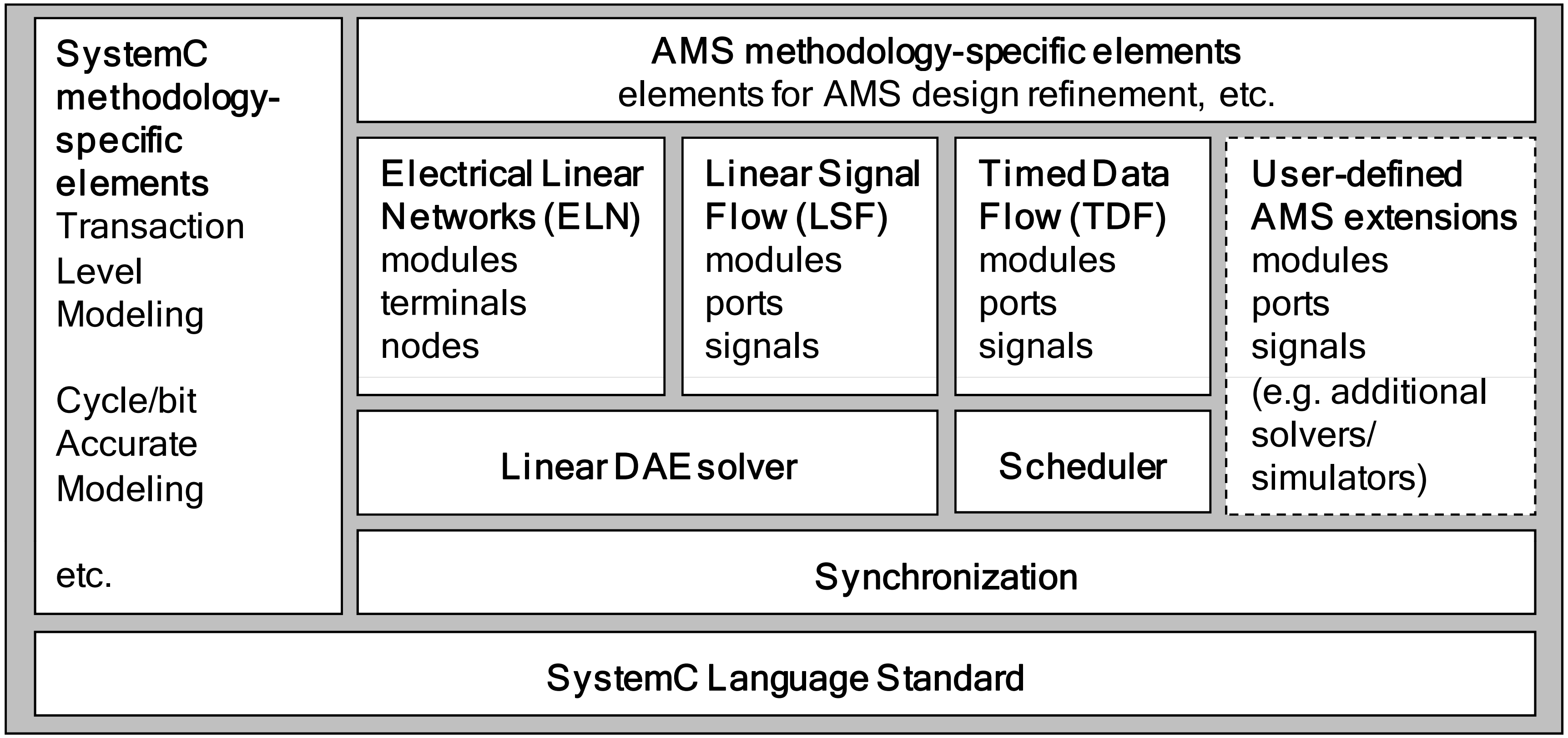
- Chair: Martin Barnasconi (NXP),

  Co-Chair: Christoph Grimm (TU Wien)

Modeling formalisms and use cases



SystemC AMS extensions

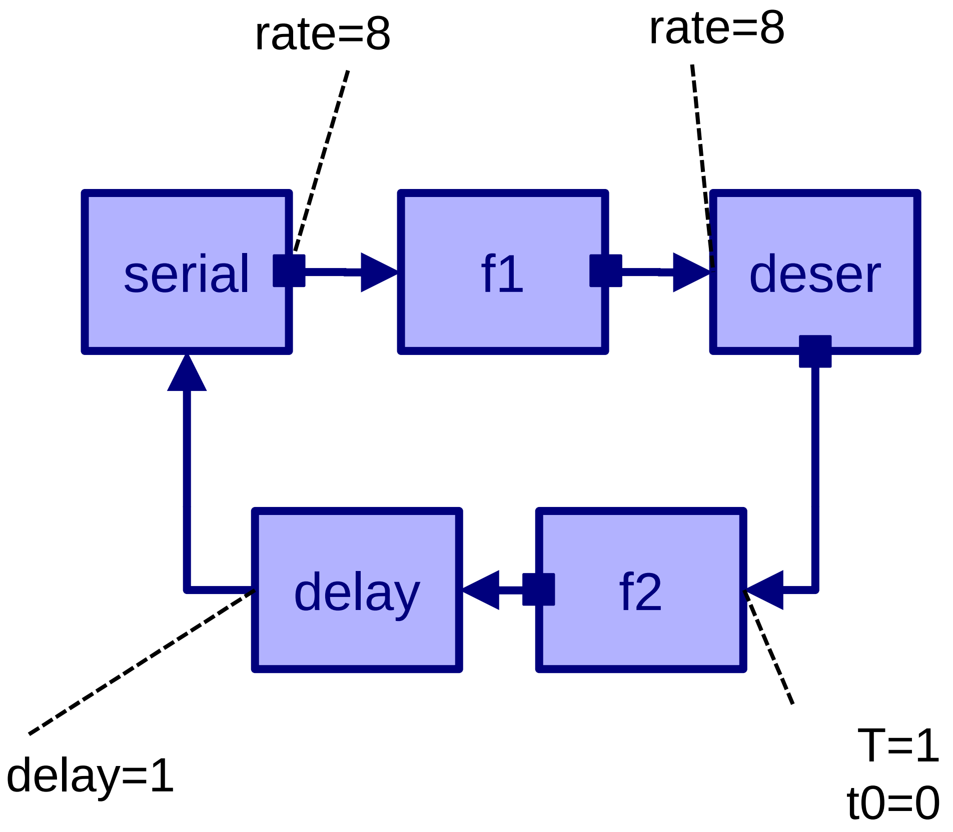


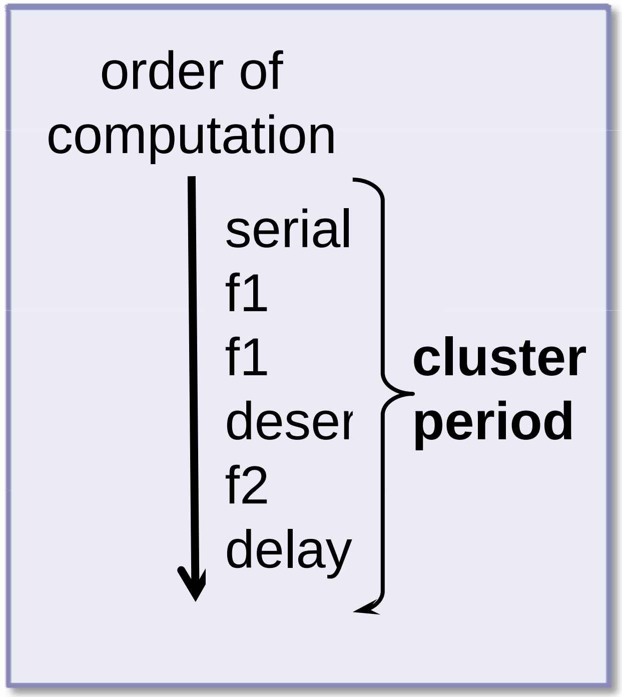
SystemC methodology-specific elements
Transaction Level Modeling
Cycle/bit Accurate Modeling
etc.
AMS methodology-specific elements
elements for AMS design refinement, etc.
Electrical Linear Networks (ELN)
modules
terminals
nodes
Linear Signal Flow (LSF)
modules
ports
signals
Timed Data Flow (TDF)
modules
ports
signals
User-defined AMS extensions
modules
ports
signals
(e.g. additional solvers/ simulators)
Linear DAE solver
Scheduler
Synchronization
SystemC Language Standard

[Grimm/Barnasconi/Vachoux/Einwich: An Introduction to Modeling Embedded Analog/Mixed-Signal Systems using SystemC AMS Extensions. OSCI, June 2008]

Timed Data Flow in SystemC AMS extensions

„cluster“ := set of connected TDF modules





Timed Data Flow Block: Serializer

TDF Module:

primitive module!

```
SCA_TDF_MODULE(serial)
{
  sca_tdf::sca_in<sc_bv<8> > in;
  sca_tdf::sca_out<bool>     out;
```

Attributes specify

timed semantics

```
  void set_attributes()
  { out.set_rate(8);
    //out.set_delay(1);
    //out.set_timestep(1, SC_MS);}
```

processing()

describes

computation

```
  void processing()
  {
    for (int i=7; i >= 0 ; i-- )
      out.write(in.get_bit(i), i);
  }
  SCA_CTOR(serial);
}
```

Interfacing Timed Data Flow and SystemC (DE)

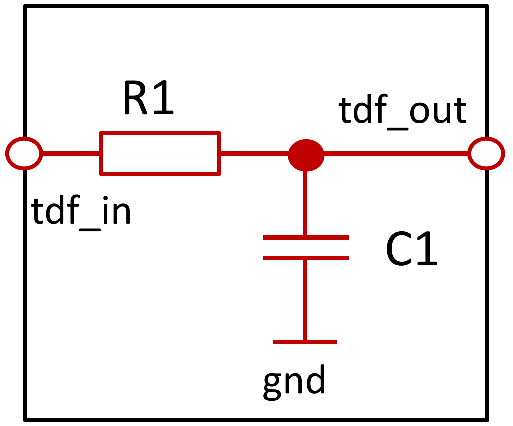
Converter ports towards discrete event domain

```
sca_tdf::sc_in < <type> >
sca_tdf::sc_out < <type> >
```

Note: Time in MR – TDF may run ahead DE time!

```
sc_time sca_get_time()
```

Linear Electrical Networks, Converters



```
SC_MODULE(lp_filter_eln)
{
  sca_tdf::sca_in<double>  in;
  sca_tdf::sca_out<double> out;
  sca_eln::sca_node in_node, out_node; // nodes
  sca_eln::sca_node_ref gnd;           // reference
  sca_eln::sca_r     *r1;              // resistor
  sca_eln::sca_c     *c1;              // capacitor
  sca_eln::sca_tdf2v *v_in;            // converter TDF->U
  sca_eln::sca_v2tdf *v_out;           // converter U->TDF
  SC_CTOR(lp_filter_eln) {
    v_in = new sca_eln::sca_tdf2v("v_in", 1.0);   // scale factor 1.0
      v_in->ctrl(in); v_in->p(in_node); v_in->n(gnd);
    r1 = new sca_eln::sca_r("r1", 10e3);          // 10kOhm resistor
      r1->p(in); r1->n(out_node);
    c1 = new sca_eln::sca_c("c1", 100e-6);        // 100uF capacitor
      c1->p(out_node); c1->n(gnd);
    v_out = new sca_eln::sca_v2tdf("v_out", 1.0); // scale factor 1.0
      v_out1->p(out_node); v_out1->n(gnd); v_out->ctrl(out);
  }
};
```

Design refinement of E-AMS systems

1.) SystemC AMS extensions

2.) Design refinement of E-AMS

3.) Outlook

„Design Refinement“ of E-AMS

Design Refinement is a top-down design methodology that stepwise and interactively augments an executable, functional specification with properties of intended implementation at architecture level.

- *Similar, but not the same:*
  - *Extreme programming, refactoring (SW Engineering)*
  - *Property refinement (Formal method to ensure safety properties)*

What is needed for „Design Refinement“?

1. A *single* language that combines functional level with architecture/implementation level:

   → **SystemC AMS extensions** (OSCI)

2. Design methodology specific support library

   → **HEAVEN** (TU Vienna‘s Heterogeneous Embedded Systems Analysis/Refinement Environment)

Design Refinement of E-AMS

(executable) Specification:

Matlab/Simulink, Ptolemy [Lee, UCB] , OMNET, C/C++

1.) Refinement of computation
Algorithms, data types, physical effects/accuracy

2.) Refinement of structure
Mapping functional blocks to concrete processors

3.) Refinement of interfaces
Signals, synchronization, bus protocols

Digital:
VHDL,…

Software:
C/C++

Analog design:
VHDL-AMS, Verilog-AMS, SPICE

System integration, Postlayout, Test

FastSPICE, …

1.) Refinement of computation

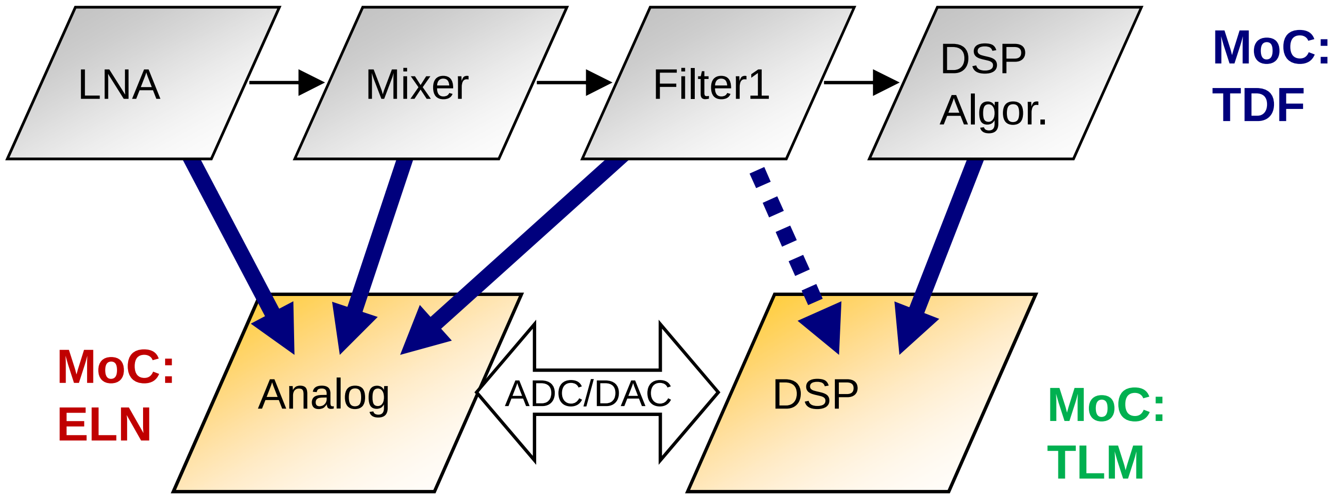
Objective: Evaluate impact of non-ideal behavior of assumed architecture/implementation using functional model

Method: Add non-ideal effects to executable specification by modifying block-behavior, e.g.:

```
void processing()  // Mixer with distortions and noise
{
  double rf = in1.read(); double lo = in2.read();
  double rf_dist = (alpha – gamma  * rf * rf ) * rf;
  double mix_dist = rf_dist * lo;
  if_out.write( mix_dist + my_noise() );
}
```

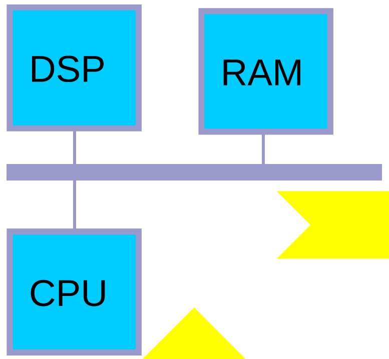
2.) Refinement of structure

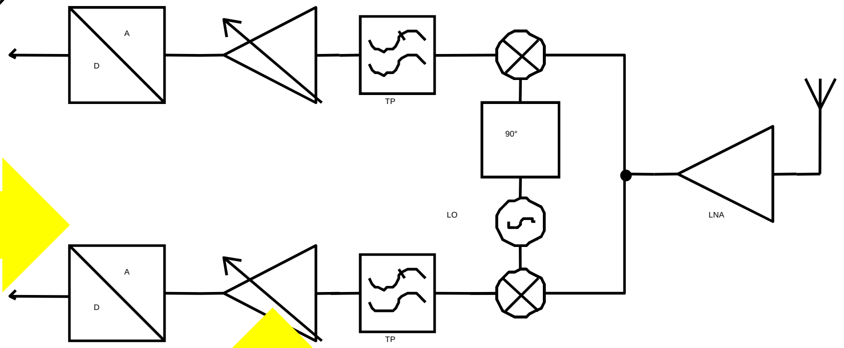
- **Objective:** Compare performance of different A/D/HW/SW partitionings more accurately
- Map functions to assumed or existing processor (=change structure, MoC)



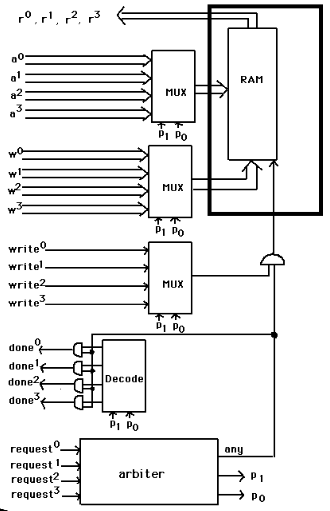
Integration in functional model „interactive“?

HW/SW:
messages (transactions)
Processors, …

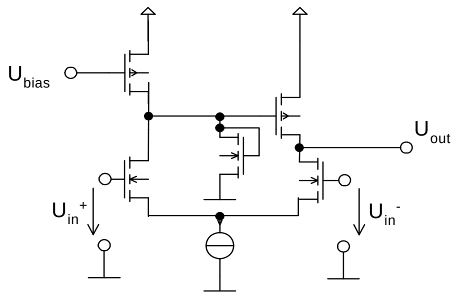




Signal/da flow: **Numbers**

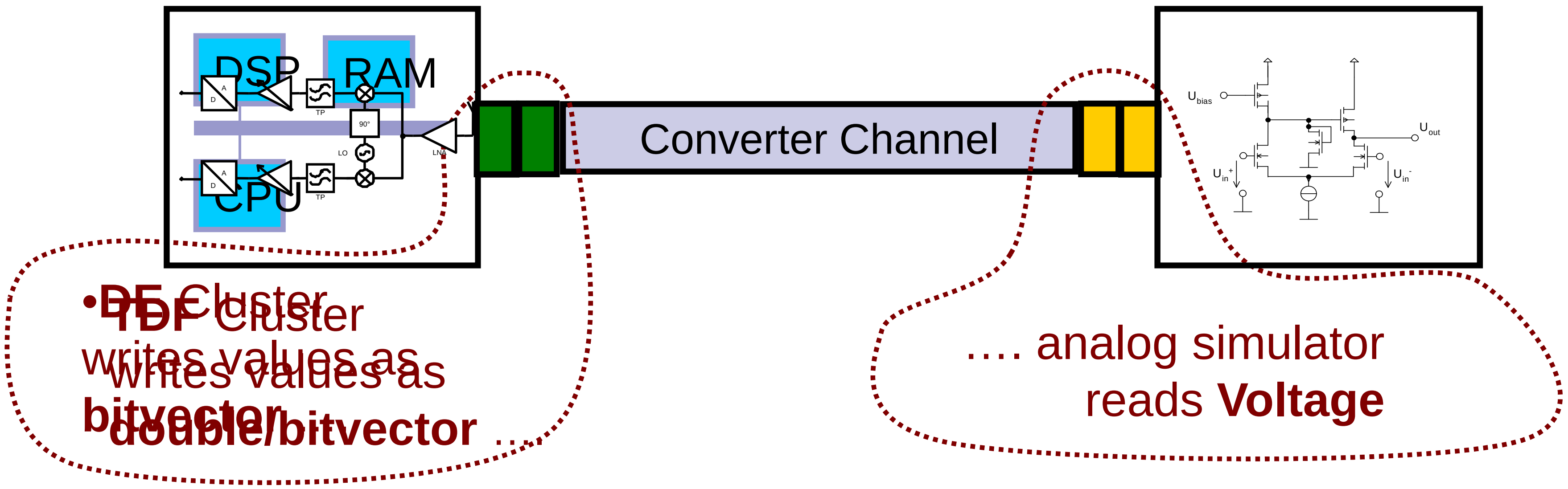


Digital circuit:
pinaccurate, single bits



Analog circuit:
Electrical nodes, physical sizes

2.) Refinement of structure


DSP
RAM
CPU
TP
90°
LO
LNA
A
D
Converter Channel
U_{bias}
U_{out}
U_{in}^{+}
U_{in}^{-}
•DE Cluster writes values as bitvector
TDF Cluster writes values as double/bitvector …
…. analog simulator reads Voltage

Converter channel from HEAVEN library :

- Automatically do conversion of signal types, ranges, …
- Enable interactive replacement of functional blocks by implementation

[Damm, Haase, Grimm, "Co-Simulation of mixed HW/SW and Analog/RF Systems at Architectural Level", *BMAS'08*, San José, Sept. 2008.]

Refinement of structure, re-partitioning

- **Method 2 (top-down):**

  Re-write functional part that is re-partitioned in new model of computation that matches intended realization (cumbersome …)
  - Analog: Signal flow, Network
  - Digital HW: TDF, DE (or TLM)
- *Slightly simplified by „static polymorphism“*

```
SCA_MoC_MODULE(par2ser)
{
  sca_MoC::sca_in<sc_bv<8> > in;
  sca_MoC::sca_out<bool>     out;
  …
  SCA_CTOR(par2ser)
}
```

3.) Refinement of interfaces

- **Intention:** Prepare validation of system integration -
  All ports accurately as in implementation
  (same types, same number, clocks, enable-signals, …)

- **Method :**
  Adapter classes translate between abstract data flow and (DE) pin-accurate protocols or analog nodes

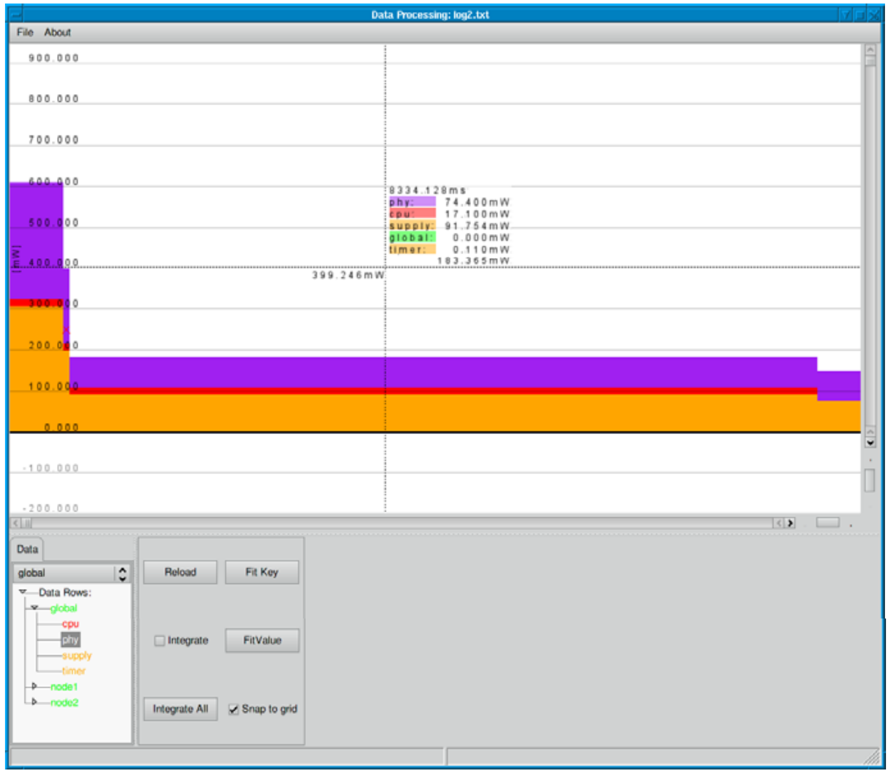
  (requires appropriate models inside, known from SystemC, TLM, …)

HEAVEN bundles support for refinement

- Converter channels
  - MoC supported: TDF, TLM, DE, Analog circuits
  - Data types: double, bitvector, logic_vector
  - Simulators: CADENCE, Matlab Simulink
- Adapter classes (at the moment: SSIO)
- Functional blocks & behavioral models for communication systems
- Analysis tools
  - Power estimation
  - Eye diagrams, Trellis diagrams, Constellation diagrams, …

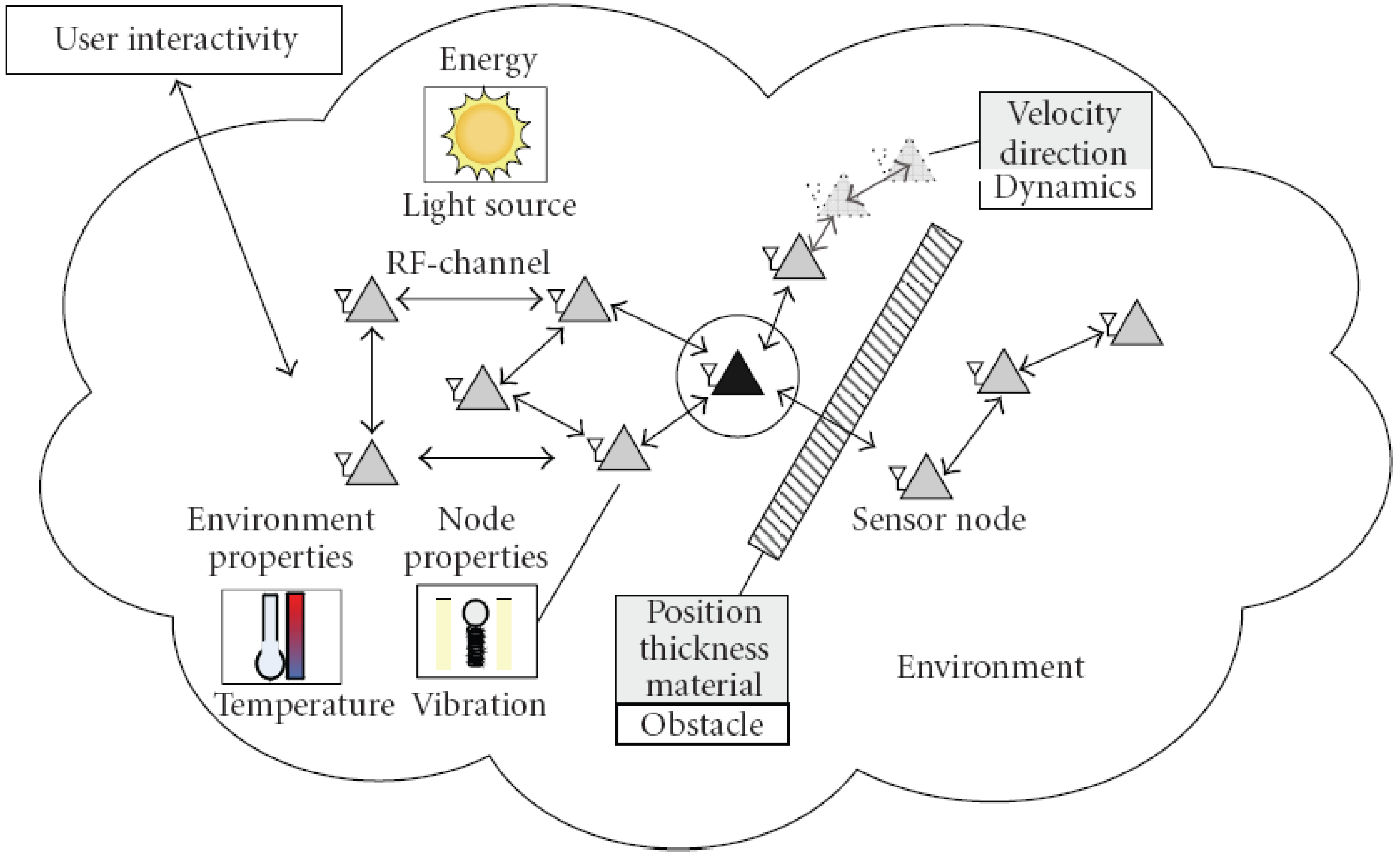
Analysis Tools: Estimation Power Consumption

- Estimation via
  - Usage of functionality
  - ISS
- “Calibration” by
  - Circuit simulation
  - Measurement
  - Data sheets
- “Usage profile” in log file
- Post Processing:
  Analysis, Visualization



```
8334.128ms
phy:       74.400mW
cpu:       17.100mW
supply:    91.754mW
global:     0.000mW
timer:      0.110mW
          183.365mW
399.246mW
```

Typical Simulation Scenario: Wireless Sensor Network



Results?

- Design Refinement aims at higher productivit
  - Hard to measure / compard …

- Statistics of industrial application
  - Executable spec, Integration validation w/ Verilog-AMS
    ➔ Today most SoC projects „first time right“
  - Architecture exploration by refinement (+SystemC AMS?)
    ➔ Target: 30% reduction of design time

Design refinement of E-AMS systems

1.) SystemC AMS extensions

2.) Design refinement of E-AMS

3.) Outlook

Summary, Future work

- **SystemC AMS extensions** provide appropriate means for architecture exploration of E-AMS systems

- Outlook SystemC AMS
  - 1st Draft of LRM available on www.systemc.org
  - public review – please comment!

Summary, Future work

- **Design refinement** could increase design productivity by
  a) Quickly available first models
  b) Immediate analysis/verification after changing/adding property

- *HEAVEN library:*
  - *Communication with CADENCE Design Framework*
  - *Application specific blockset for refinement of Ultra-Low power Wireless Sensor Networks incl. estimation of power at functional/architecture/implementation level.*

References

- www.systemc.org
  (OSCI members)
- www.systemc–ams.org

  (For information from former SystemC-AMS SG, provides some information for the public)
- *Ch. Grimm, M. Barnasconi, A. Vachoux, K. Einwich*: An Introduction to Modeling Embedded Analog/Mixed-Signal Systems using SystemC AMS Extensions. OSCI, June 2008
- *Ch. Grimm:* Modeling and Refinement of Mixed Signal Systems with SystemC. In: *SystemC: Methodologies and Applications*. Kluwer Academic Publisher (KAP), June 2003.