ESD Electronic Systems Design

Synthesis Verification Testing Power Communication

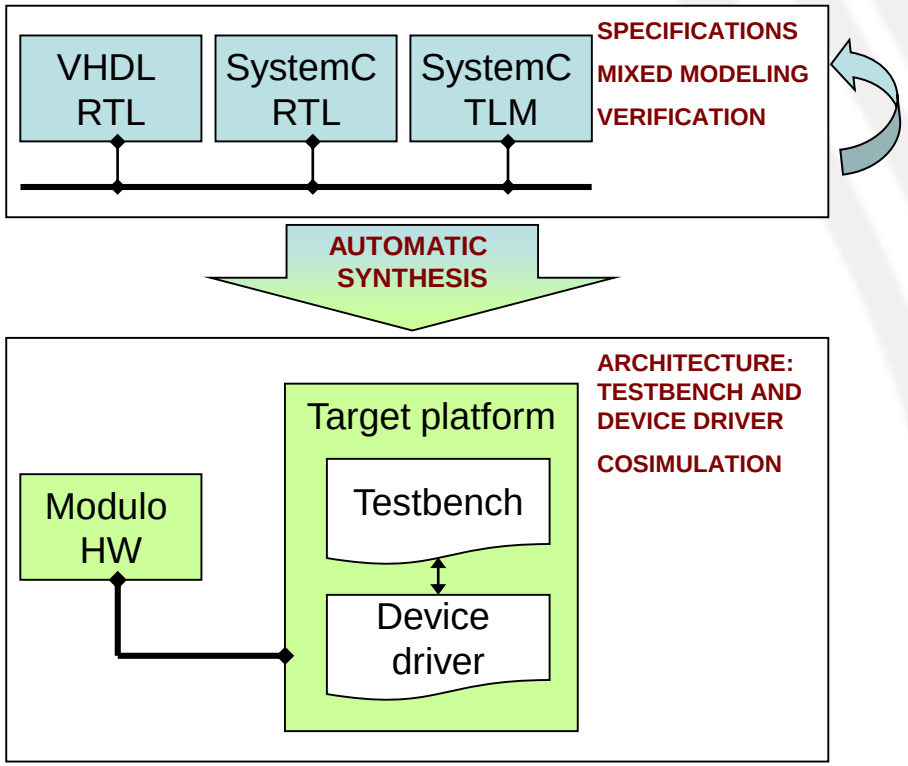
Mixed RTL/TLM/AMS modeling

Stefano Centomo

Course introduction: Lab topics

- **Emulate the design flow of a real embedded/cyber-physical system**

- **Specification**
  - Compilation/execution/debugging of SystemC code
  - Modeling of SystemC at RTL level
  - Modeling of SystemC at TLM level
  - Time evolution in SystemC
  - Modeling analog and continuous behaviors: SystemC-AMS
  - Components integration
  - Mixing SystemC RTL, TLM and AMS

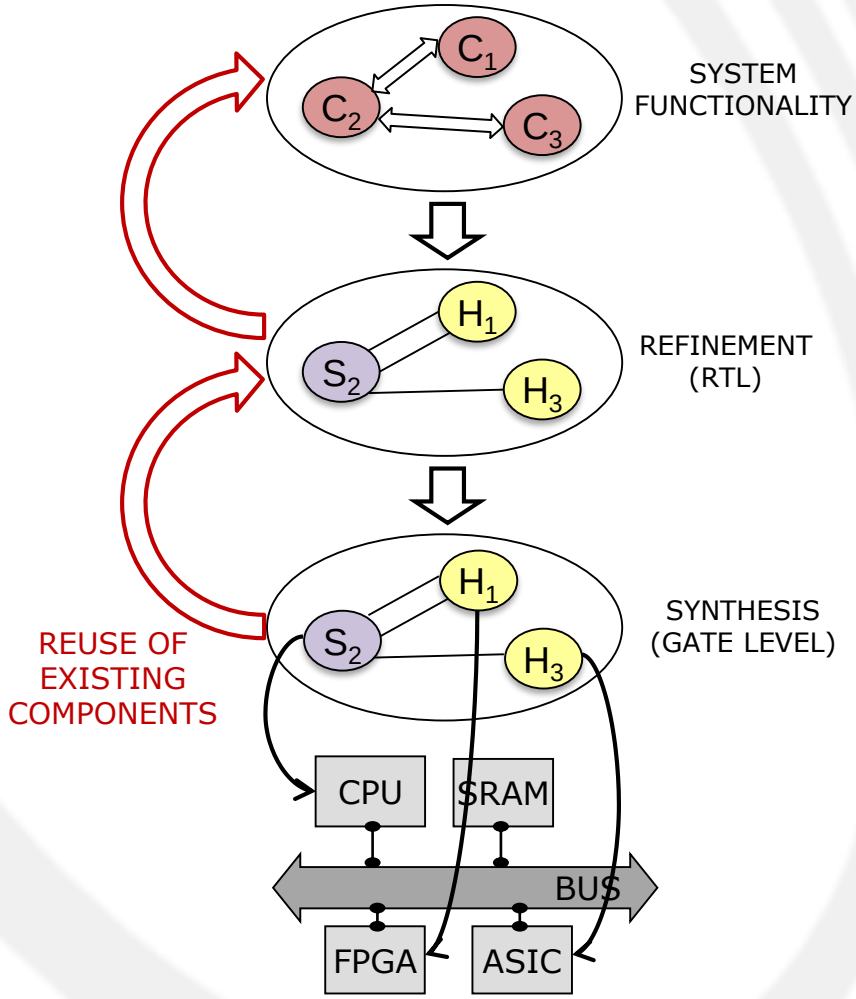


- **SW Synthesis**
  - Platforms, testbenches and drivers
  - Model Based Design

- **HW Synthesis**
  - VHDL modeling at RTL
  - Automatic Synthesis from TLM descriptions
  - Automatic Synthesis of RTL VHDL code

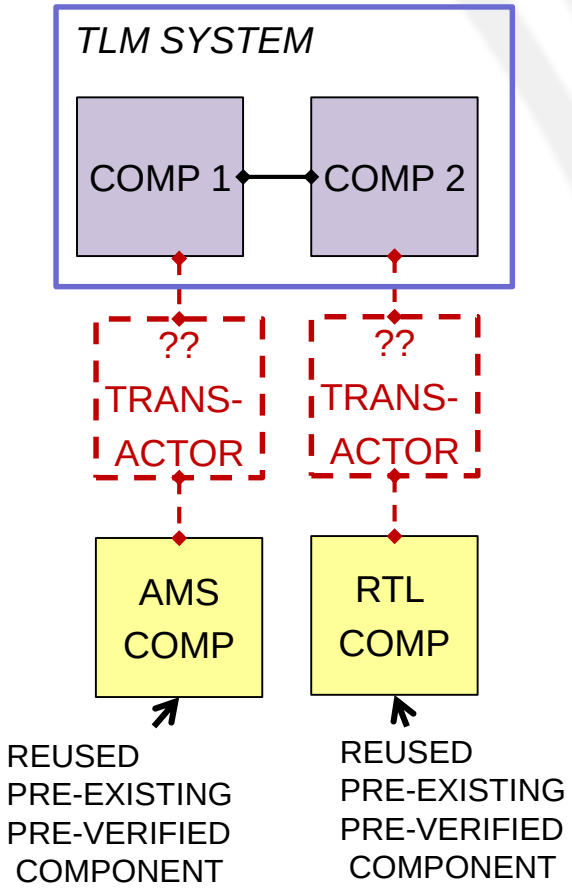
Design

- Design flow
  - In theory: top-down approach
    - Choose and describe the functionality
    - Implement it with abstract communication (TLM)
    - Implement proper communication protocol (RTL)
    - Gate-level synthesis
  - In practice: reuse
    - Fundamental to meet time to market constraints and to save money (design+verification)



Mixed Modeling

- Mixed modeling
  - Implied by reuse
    - Integrate components developed at different levels of abstractions
  - Transactor
    - Translator in the communication between modules implemented at different levels of abstraction
    - Allows mixed modeling and testbench reuse

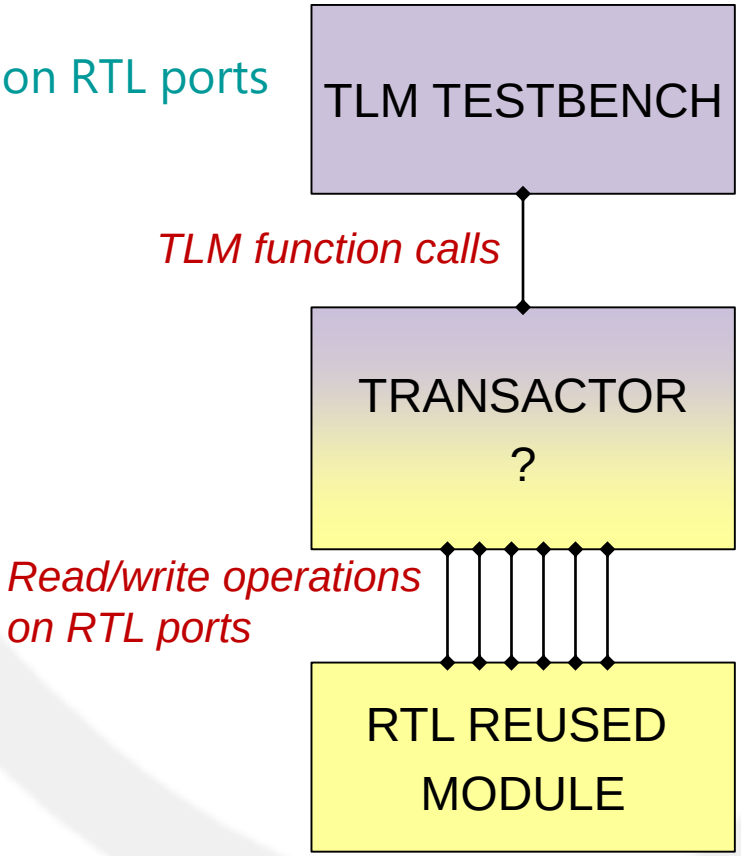


Connecting RTL and TLM

Basic Transactors

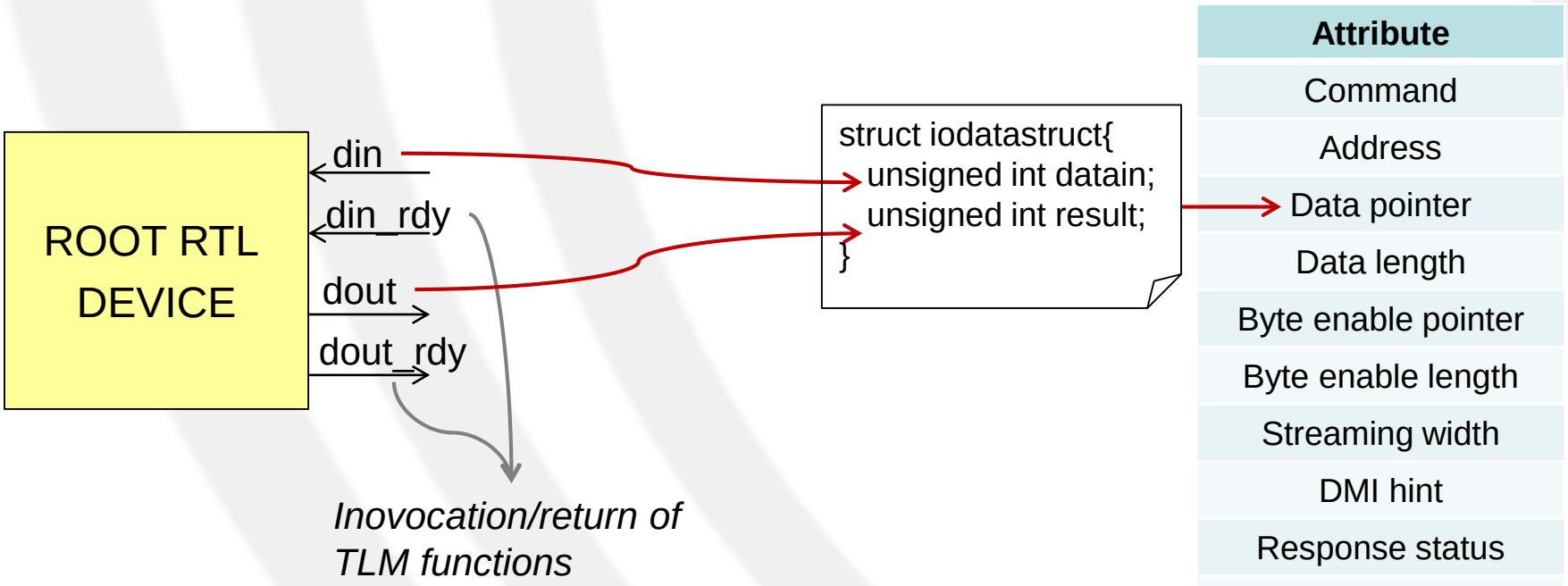
Transactor: main idea

- Two APIs
  - Higher level of abstraction (TLM)
    - TLM standard functions
  - Lower level of abstraction (RTL)
    - Sequence of read/write operations on RTL ports



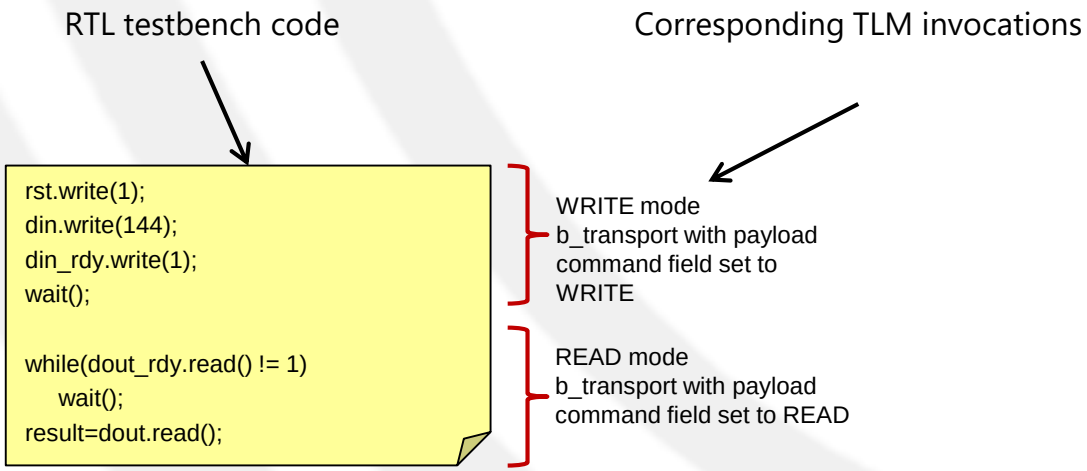
Transactor: mapping

- Define a mapping between RTL ports and TLM payload fields



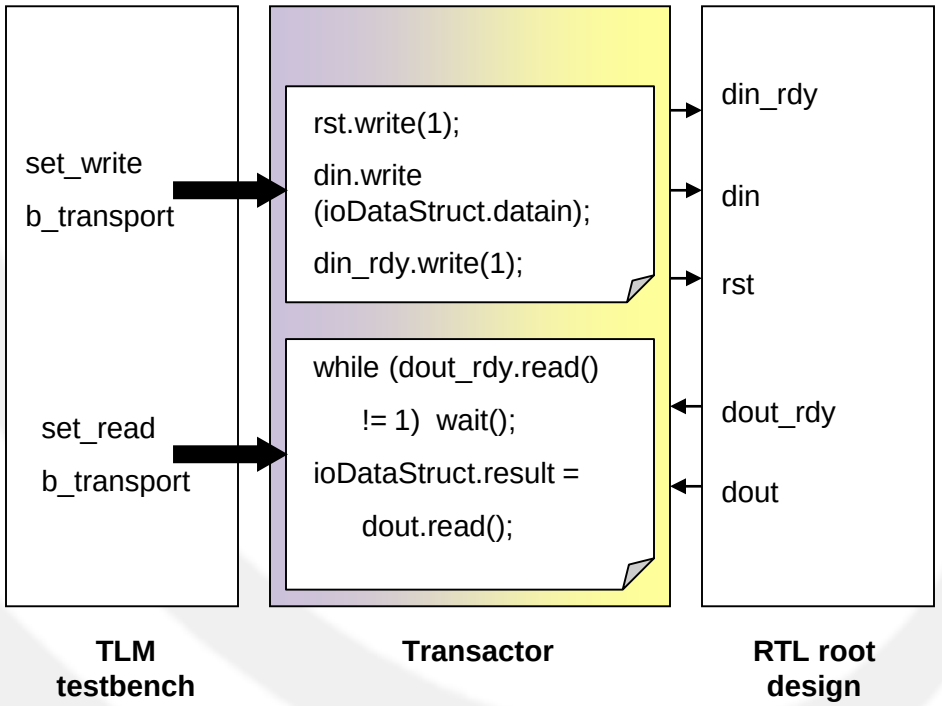
Transactor: communication

- Transaltion of TLM function calls to sequences of RTL signal operations
  - From blocking/non blocking primitives to cycle accurate temporization and handshaking
  - Root:
    - Two modes, read and write
    - Correspond to b_transport invocations with different command parameters



Transactor: structure

- Transactor structure
  - B_transport interface towards TLM
  - Depending on the command
    - Activates the WRITE process to set number_port and number_isready port
    - Activates the READ process to get the value of the result_port and result_isready port



Transactor: Example

```
void root_RTL_transactor::b_transport
        (tlm::tlm_generic_payload& trans, sc_time& t){
 tlm::tlm_command cmd = trans.get_command();
 switch (cmd) {
  case tlm::TLM_WRITE_COMMAND:
   ioDataStruct = *((iostruct*) trans.get_data_ptr());
   begin_write.notify();
   wait(end_write);
   break;
  case tlm::TLM_READ_COMMAND:
   ioDataStruct = *((iostruct*) trans.get_data_ptr());
   begin_read.notify();
   wait(end_read);
   break;
  default:
   break;
}
```

```
void root_RTL_transactor::WRITEPROCESS(){
 while (true) {
  wait(begin_write);
  rst.write(1);
  din.write(ioDataStruct.datain);
  din_rdy.write(1);
  end_write.notify();
  wait();
}}
```

```
void root_RTL_transactor::READPROCESS(){
 while (true) {
  wait(begin_read);
  while(dout_rdy.read() != 1)
  wait();
  ioDataStruct.result=dout.read();
  end_read.notify();
}}
```

Extending to the hybrid domain

TLM/AMS communication

Available interfaces

- SystemC-AMS provides interfaces for the discrete event environment
  - TDF "extra-cluster" ports
    - sca_tdf::sca_de::sca_in<T>
    - sca_tdf::sca_de::sca_out<T>
  - LSF specialized sources and sinks
    - sca_lsf::sca_de::sca_source
    - sca_lsf::sca_de::sca_sink
  - ELN voltage or current sources and sinks controlled by DE kernel
    - sca_eln::sca_de::sca_vsourse and sca_eln::sca_de::sca_isource
    - sca_eln::sca_de::sca_vsink and sca_eln::sca_de::sca_isink

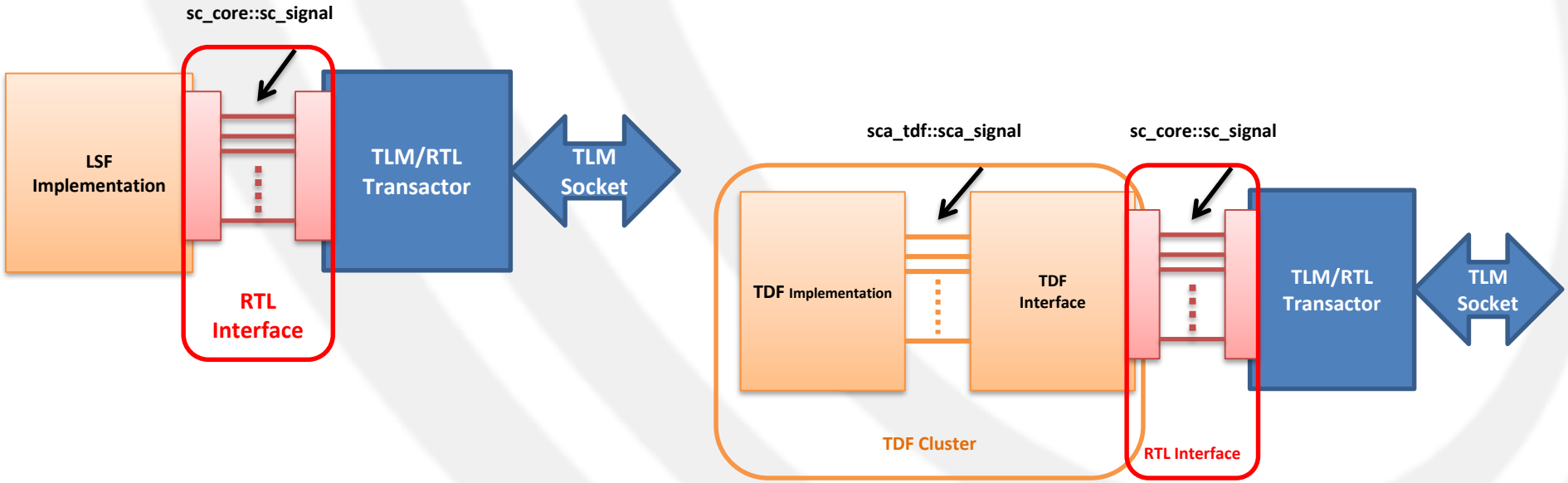
- All the provided interfaces requires a SystemC **signal** in the DE side
  - Templatic for TDF (any type can be used)
  - Double for ELN and LSF

- None of the MoC accepts TLM sockets!
  - **Only RTL models can be connected to AMS models**

TLM/AMS communication using TLM/RTL transactors

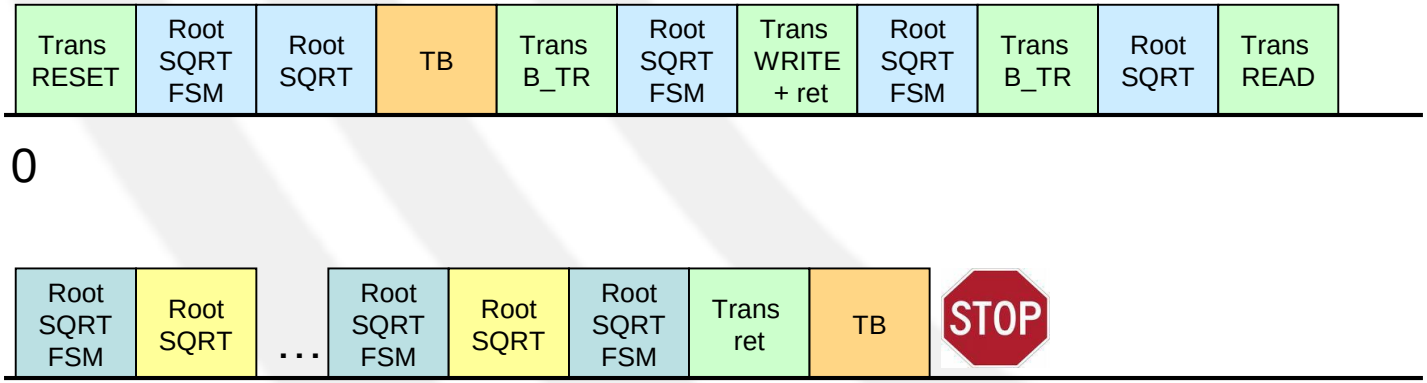
- **Remember:** when you assign a value to a signal the time does not advance, the delta count does!
  - Propagate values using asynchronous processes

- **Solution:** insert a RTL layer between TLM and AMS
  - Double interface:
    - TLM/RTL transactor to communicate with the TLM side
    - AMS with DE ports in the AMS side



SystemC into action

- Uncompress the archive
  - $ tar xjvf lesson_06_srcs.tar.bz2
  - $ cd lesson_06_srcs/
  - $ ls
    root_assertions root_transactor
    ams_transactor
  - $ cd root_transactor
  - $ ls
    - bin inc obj src
- Compile
  - $ make
  - $ ./bin/root_RTL.x



SystemC-AMS into action

- Uncompress the archive
  - $ tar xzvf 06_transactors_and_assertions.tar.gz
  - $ cd 06_transactors_and_assertions/
  - $ ls
    root_assertions root_transactor ams_transactor
  - $ cd ams_transactor
  - $ ls
    - bin inc obj src
- Compile
  - $ make
  - $ ./bin/feedback_system.x
- Plant (LSF) and Controller (TDF) system communicating through TLM primitives
  - Tracing the execution using gnuplot
    - Log files in the logs directory
    - A gnuplot script is provided
    - Generate a EPS file of the execution

Resources

- *F. Balarin, R. Passerone,* **"Specification, Synthesis, and Simulation of Transactor Processes"**, in IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems (Volume 26, Issue 10)
- *N. Bombieri, N. Deganello, F. Fummi,* **"Integrating RTL IPs into TLM designs through automatic transactor generation"**, in Proc. of IEEE/ACM DATE 2008
- *N. Bombieri, G. Pravadelli, F. Fummi,* **"On the evaluation of transactor-based verification for reusing TLM assertions and testbenches at RTL"**, in Proc. of IEEE/ACM DATE 2006
- *M. Lora, R. Muradore, R. Reffato, F. Fummi,* **"Simulation Alternatives for Modeling Networked Cyber-Physical Systems"**, in Proc. of EUROMICRO DSD 2014
  - Li, F., Dekneuvel, E., Jacquemod, G., Quaglia, D., Lora, M., Pecheux, F., & Butaud, R. "*Multi-level modeling of wireless embedded systems*". In Proc. Of IEEE/ECSI FDL 2014
  - Lora, M., Muradore, R., Quaglia, D., & Fummi, F. "*Simulation alternatives for the verification of networked cyber-physical systems*". in *Microprocessors and Microsystems*, *39*(8), 843-853.