

HW/SW Codesign II

Co-Simulation

Prof. Dr. Wolfram Hardt
Dipl.-Inf. Michael Nagler

Sommersemester 2015

Contents

- **Simulation**
- Co-Simulation
- Heterogeneous Co-Simulation
- Alternatives

Simulation (I)

- **problem: how to evaluate the systems behaviour?**

- ***remember:***

emulation = prototyping on homogenous hardware architectures

prototype = *implementation* with complete function coverage of the system specification and relaxed constraints for timing, ...

→ *emulation **not possible**, if hardware or a complete implementation is not available yet*

Simulation (II)

- **definition¹**

Simulation is the replication of a dynamic system by using _____. Aim is to get knowledge, which is transferable to the real system.

- **definition¹**

A _____ is a tool for simulation, which allows to execute or calculate the model, that is used to describe the dynamic system.

¹based on VDI directive 3633

Discussion

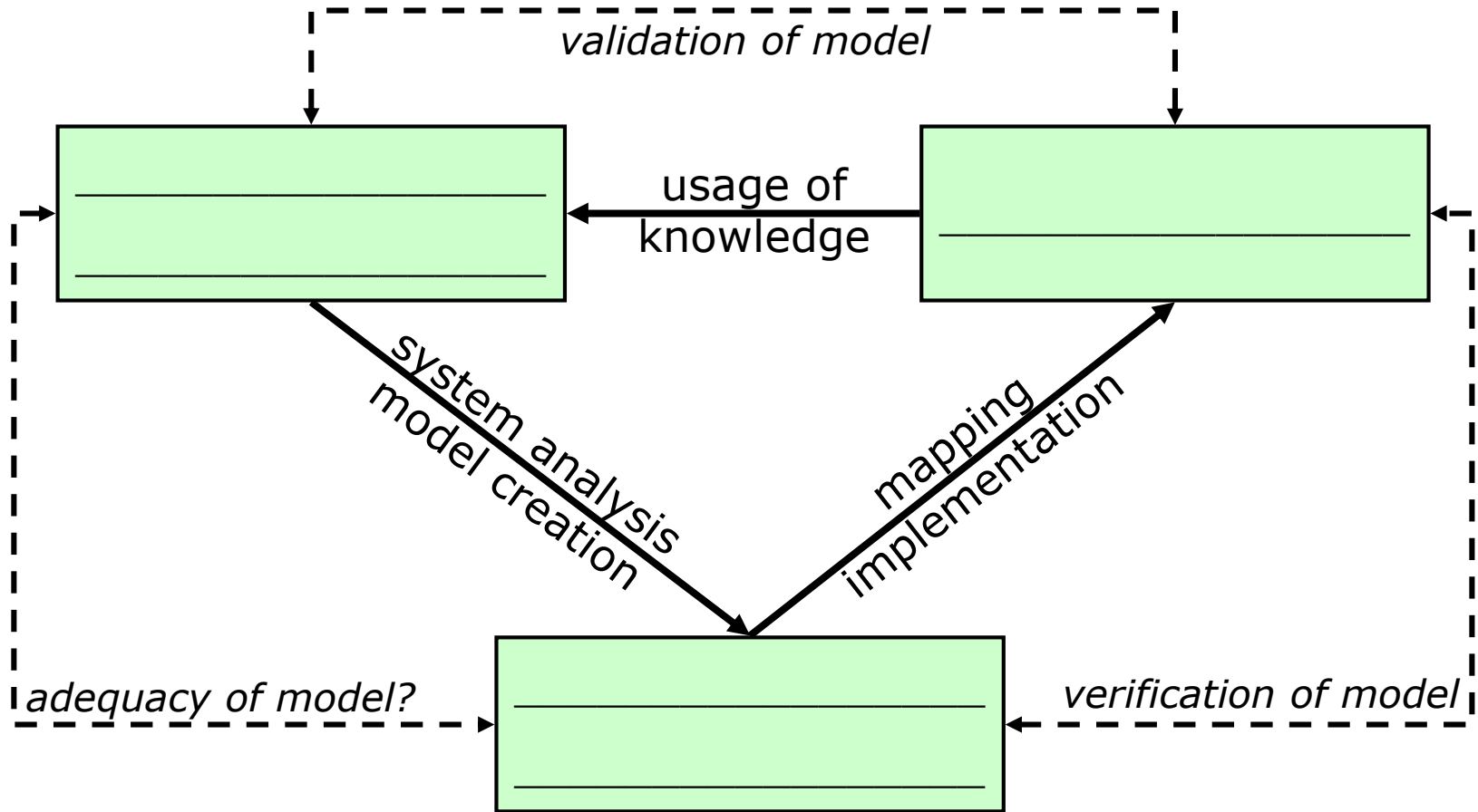
- what you simulate is what you get: simulation is important for bug free test of the product
- product schedule forces suitable strategies
- due to decrease in feature size and increase in chip size, more functionality are pushed to hardware
→ challenges for testing due to increased functionality
- knowledge about thoroughness of tests and system's environment necessary (e.g. if it involves with health and safety, then a detailed testing strategy is advisable) to define a suitable simulation strategy

Comparison

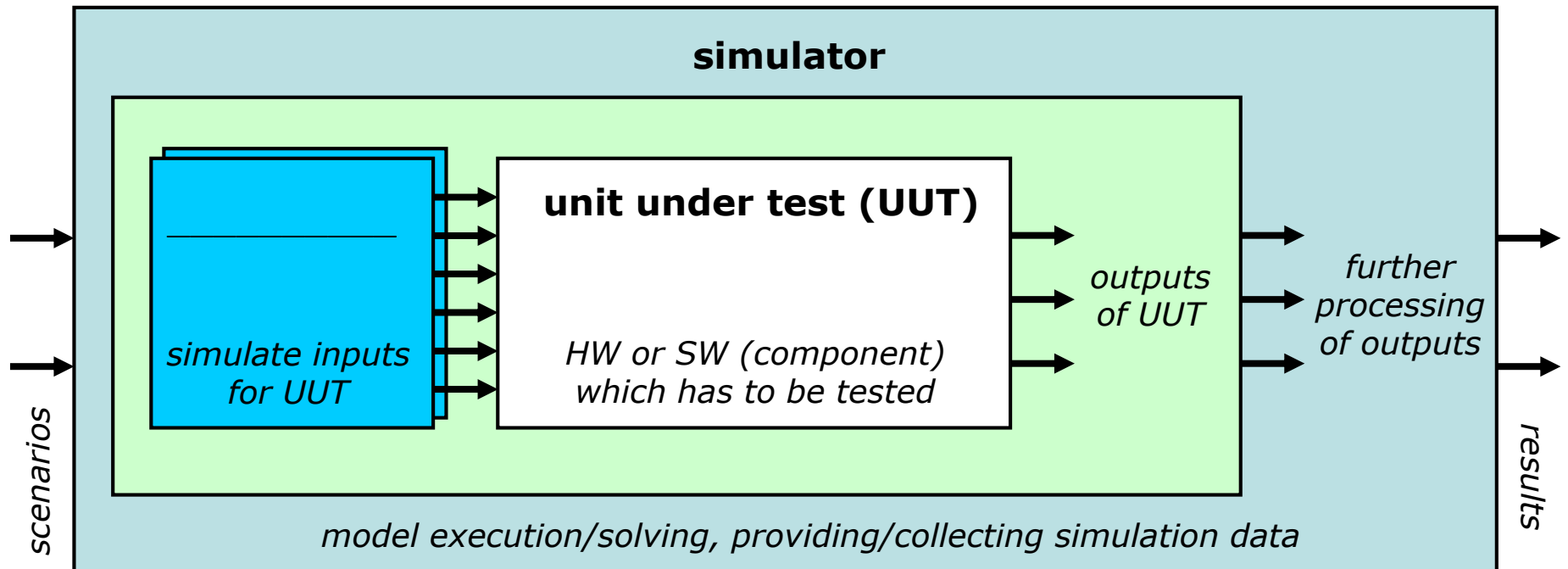
	simulation	emulation	real system
reproducibility	very easy	very easy	difficult to impossible
simplification	high degree of abstraction	low degree of abstraction	no abstraction
test execution	easy	easy	difficult
scalability	high	depends on system	depends on system

simulation \subset emulation \subset system

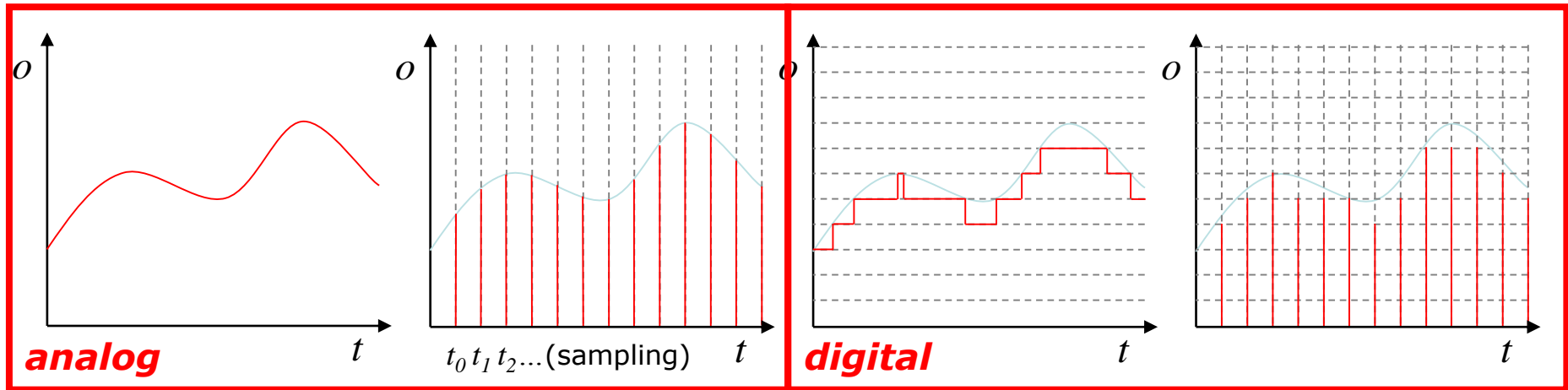
Simulation Process



Simulation Environment



Continuous vs. Discrete Systems



output o	continuous	continuous	discrete	discrete
$ o $	∞	∞	n	n
time t	continuous	discrete	continuous	discrete
simul. model	differential equations		algebras or state machines (Bool, FSM, UML,...)	

Simulation Time and Model States

- states of model change in time

- **definition**

An event is a point in time where a _____
of the simulation model takes place.

- continuous (analog) systems:
 - every point in time is an event
 - described by differential equations
 - time discretisation (sampling) possible
- discrete (digital) systems
 - volatile state transitions at a certain point of time
 - different concepts of model calculation (event-driven, ...)

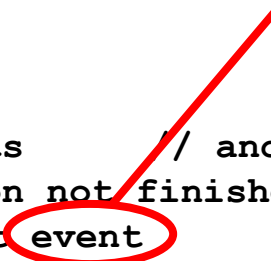
Event-Driven Simulation

- idea: only these parts of model, _____, are recalculated
- precondition: no state transition between two consecutive events
- **simulation time jumps from one event to next event**

- algorithm:

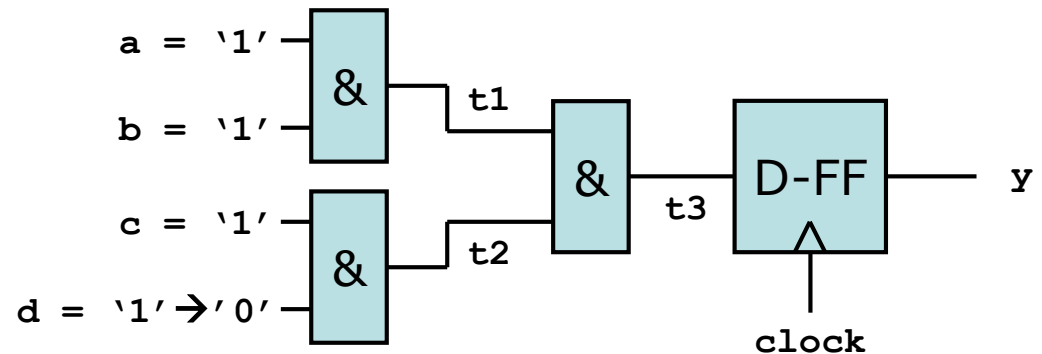
```
1. simTime = 0
2. initialise inputs // and generate first events this way
3. while (simulation not finished)
    1. E = get next event
    2. simTime = E.time
    3. remove E from event list
    4. execute E // recalculate affected parts or change
                  // inputs (depending on testbench) and store
                  // generated events
```

*'event' stores state transition
(location and value) and time
of transition*



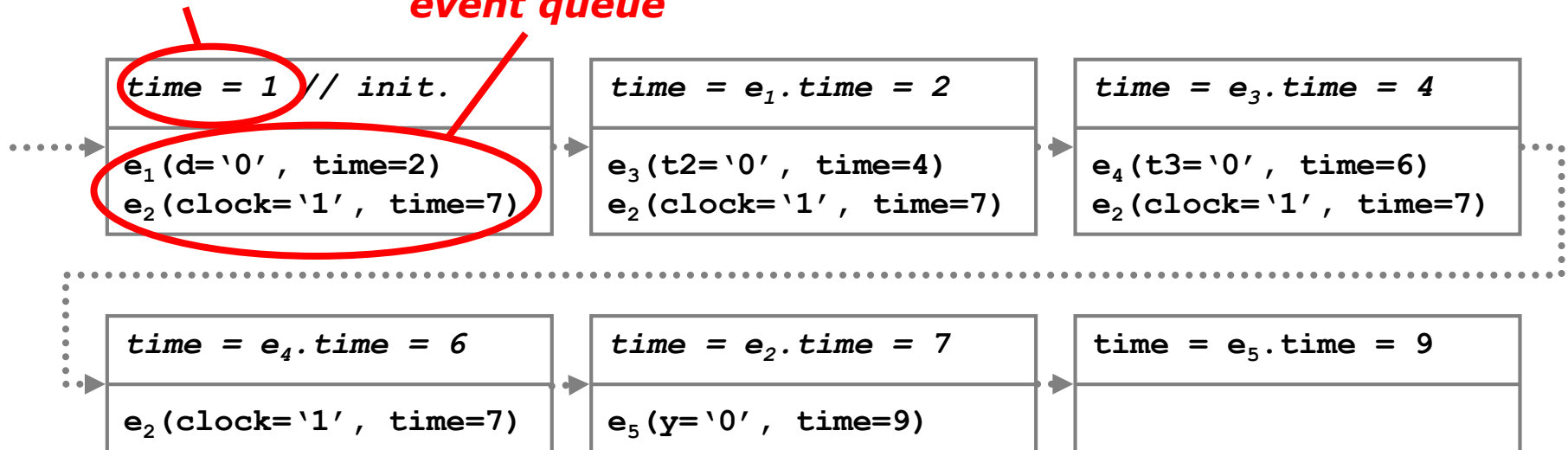
Example

- delay time of each element = 2
- system already initialised



simulation time

event queue

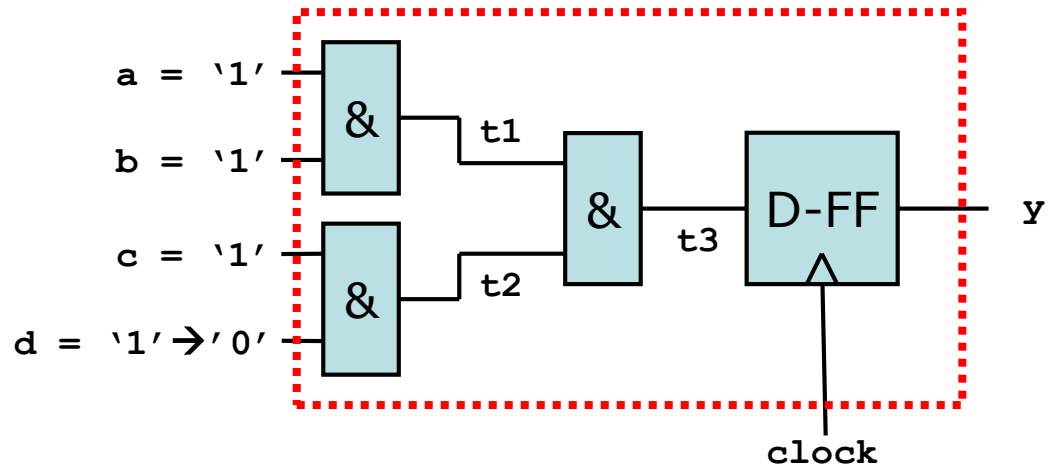
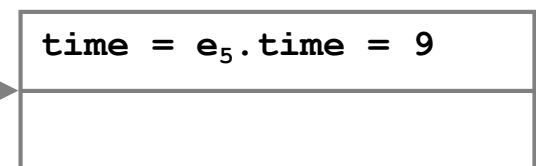
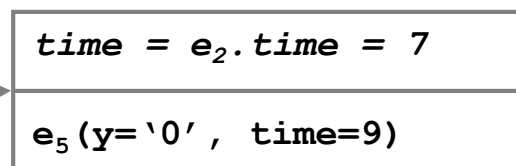
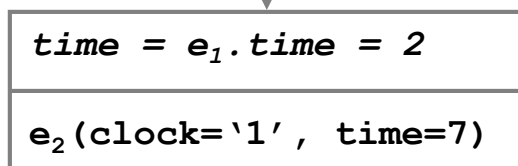
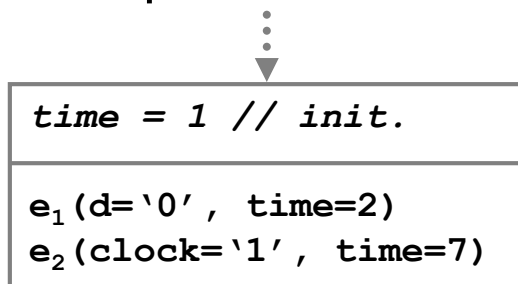


Cycle-Based Simulation

- idea: only _____ is evaluated
→ model is combined to one clocked component
- precondition: sequential system

→ **much more faster, but no timing/ delay information**

- example:

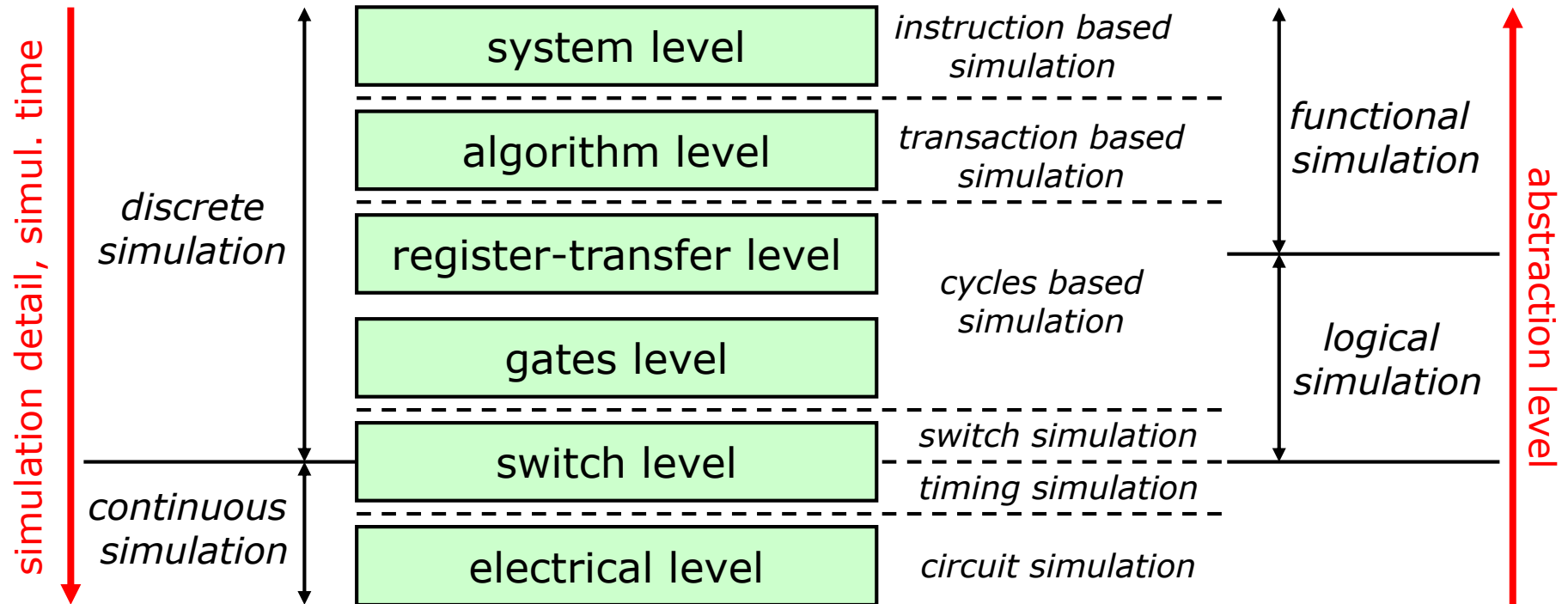


Data-Flow Simulation

- idea: signals are considered as data streams
no explicit evaluation of timing
 - blocks are activated, if all signals available
 - scheduling in the simulator for determination of execution order of blocks necessary
- **high level of abstraction, suitable for _____, typically used for determination of functionality of algorithms**

Simulation of Hardware

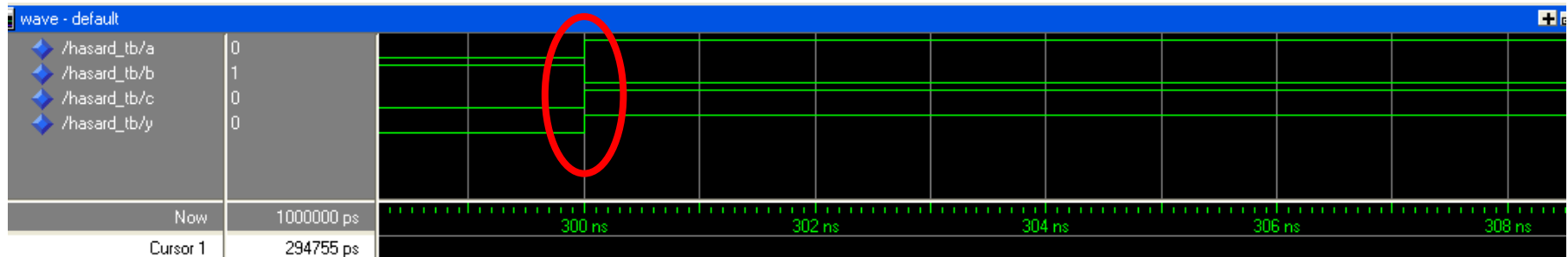
- depends on design level



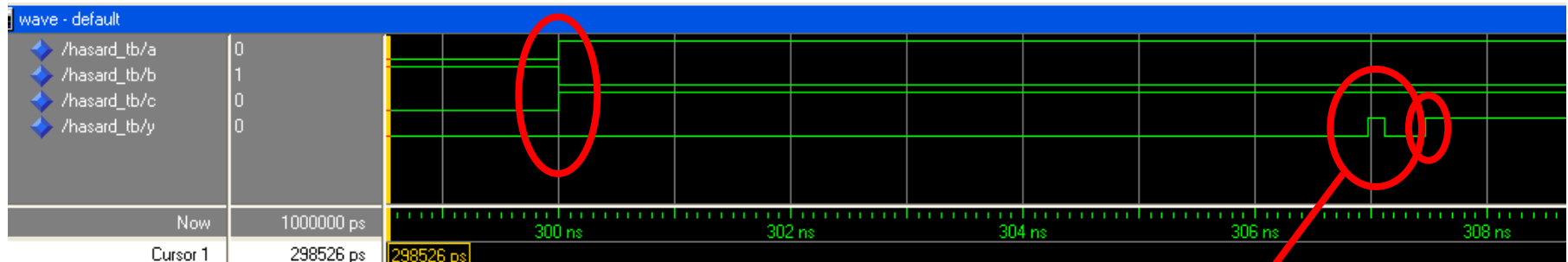
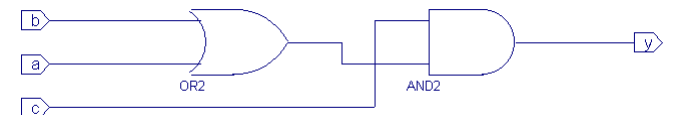
Example

- functional simulation (VHDL)

```
tmp <= a or b;  
y   <= tmp and c;
```



- technology mapped, logical simulation



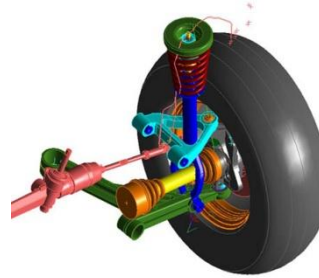
hazard – not detected in functional simulation

Simulation of Software

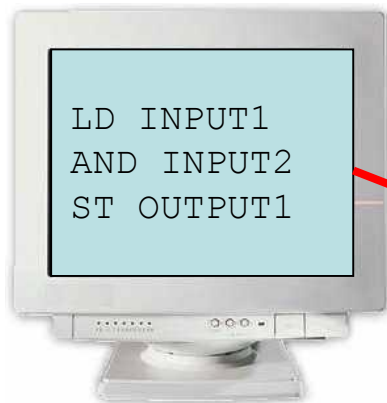
- if target architecture and associated (cross-)compiler exists
 - prototyping instead of simulation
- else runtime system has to be simulated
 - use an _____ (interprets assembly language at instruction level → no details about CPU available or used)
 - simulation environment
 - comfortable access to many design details
 - real-version mode
 - code without debug information
 - code without references to components of simulation environment
 - application example: simulation of 350 MHz Processors
 - initialization of 16 KByte table
 - simulation time > 5 min

Problem

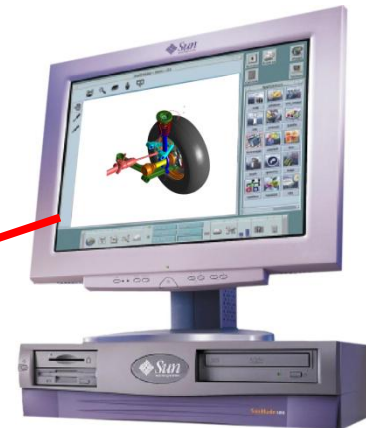
- example (mechatronics):
 - mechanical system controlled by a programmable logic controller



- development



step cascade

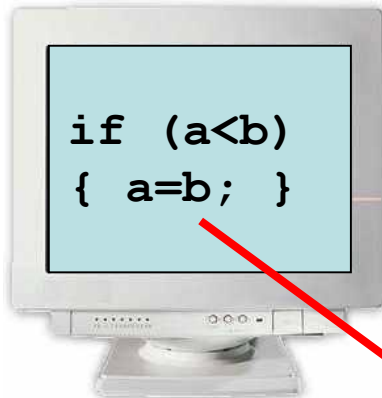


CAD model

**How to simulate
together?**

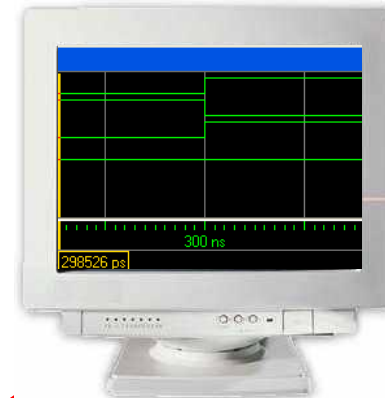
HW/SW Codesign Problem (I)

software program
(e.g. in C++)



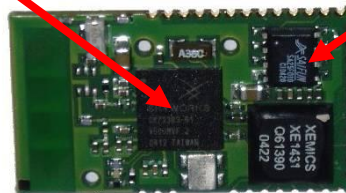
*should be
mapped to
microcontroller*

hardware description
(e.g. in VHDL)



*should be
realised as
ASIC*

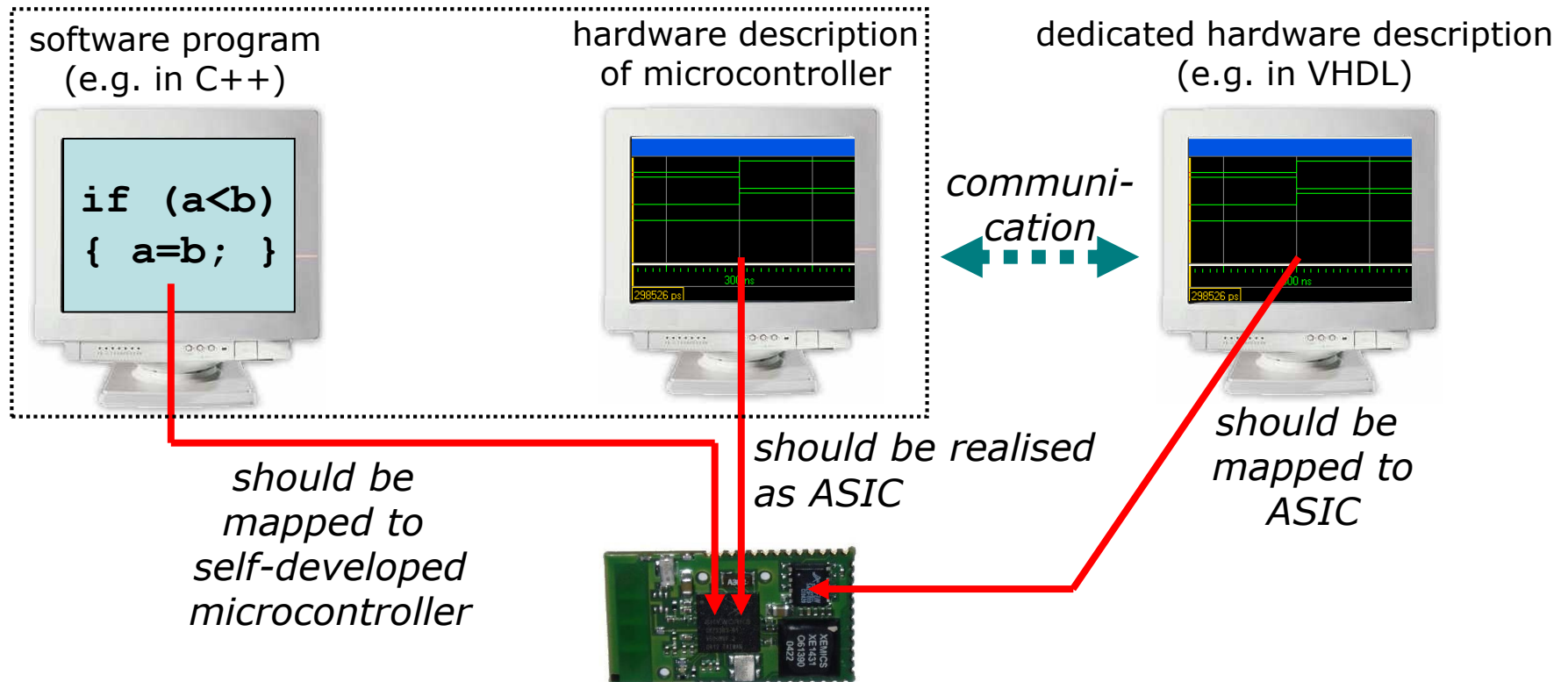
communication



- software cross-compiled to microcontroller
- hardware simulation on a PC

→ how to simulate the whole system (= interacting SW and HW)?

HW/SW Codesign Problem (II)



- microcontroller simulated on a PC
- hardware simulation on a PC
- **how to run/simulate the software in a _____?**
- **how to simulate the whole system?**

Contents

- Simulation
- Co-Simulation
- Heterogeneous Co-Simulation
- Alternatives

Co-Simulation?

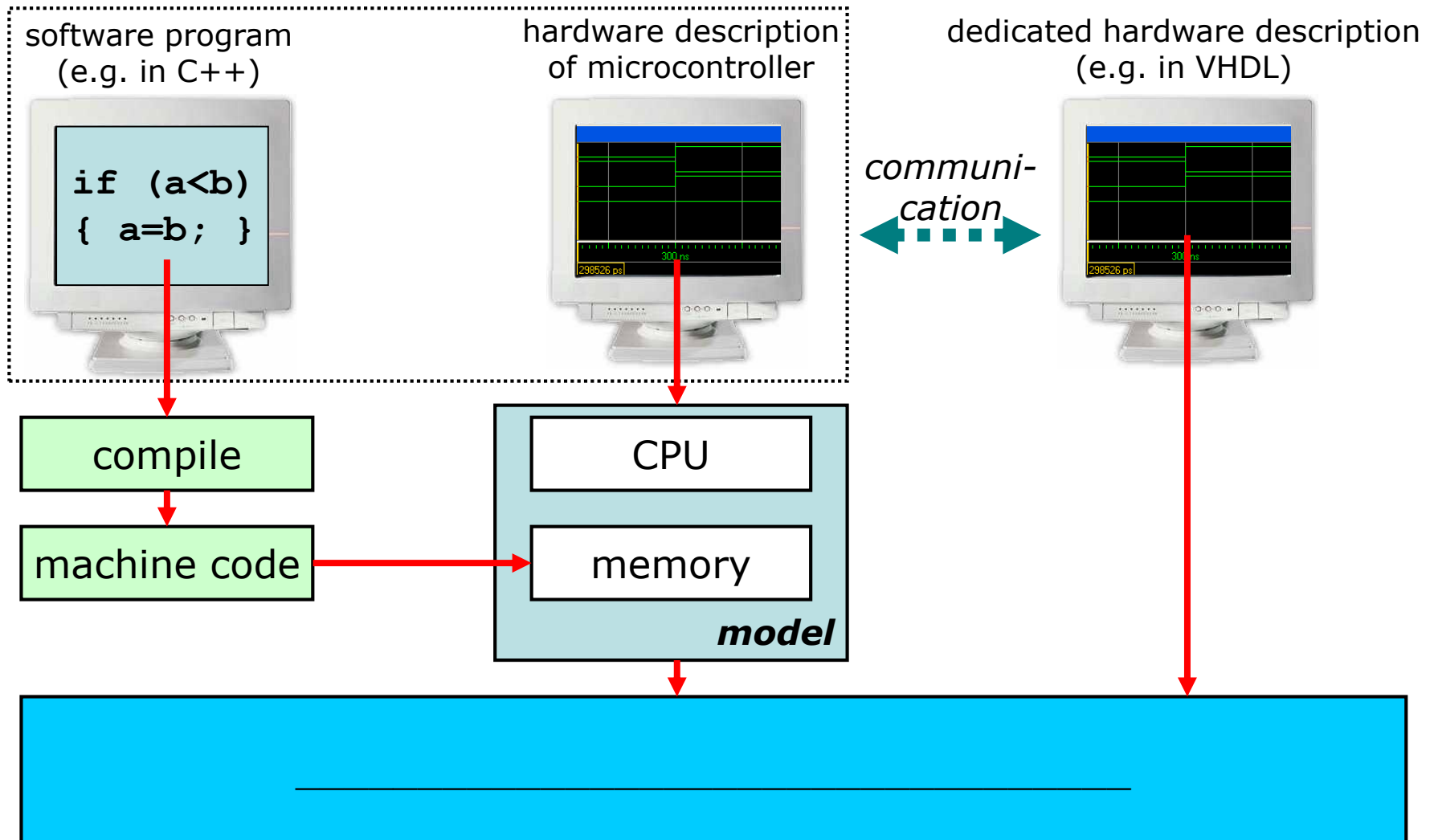
- **definition**

- goal is to verify as much of the whole systems functionality (hardware and software) as possible before fabricating parts of it (especially hardware)
- can also help to get rough parameters for HW/SW bi-partitioning decisions (runtime of different partitions, ...)

"Intuitive" Solution (I)

- put compiled software as memory content to the hardware description (including the CPU) and simulate the whole system in a logic simulation
- advantages:
 - usage of one simulator
 - timing details available
- disadvantages:
 - lot of simulation time wasted, if a already developed CPU is used
 - impossible, if no model of (bought) CPU available
 - impossible to debug software in a reasonable way

"Intuitive" Solution (II)



Contents

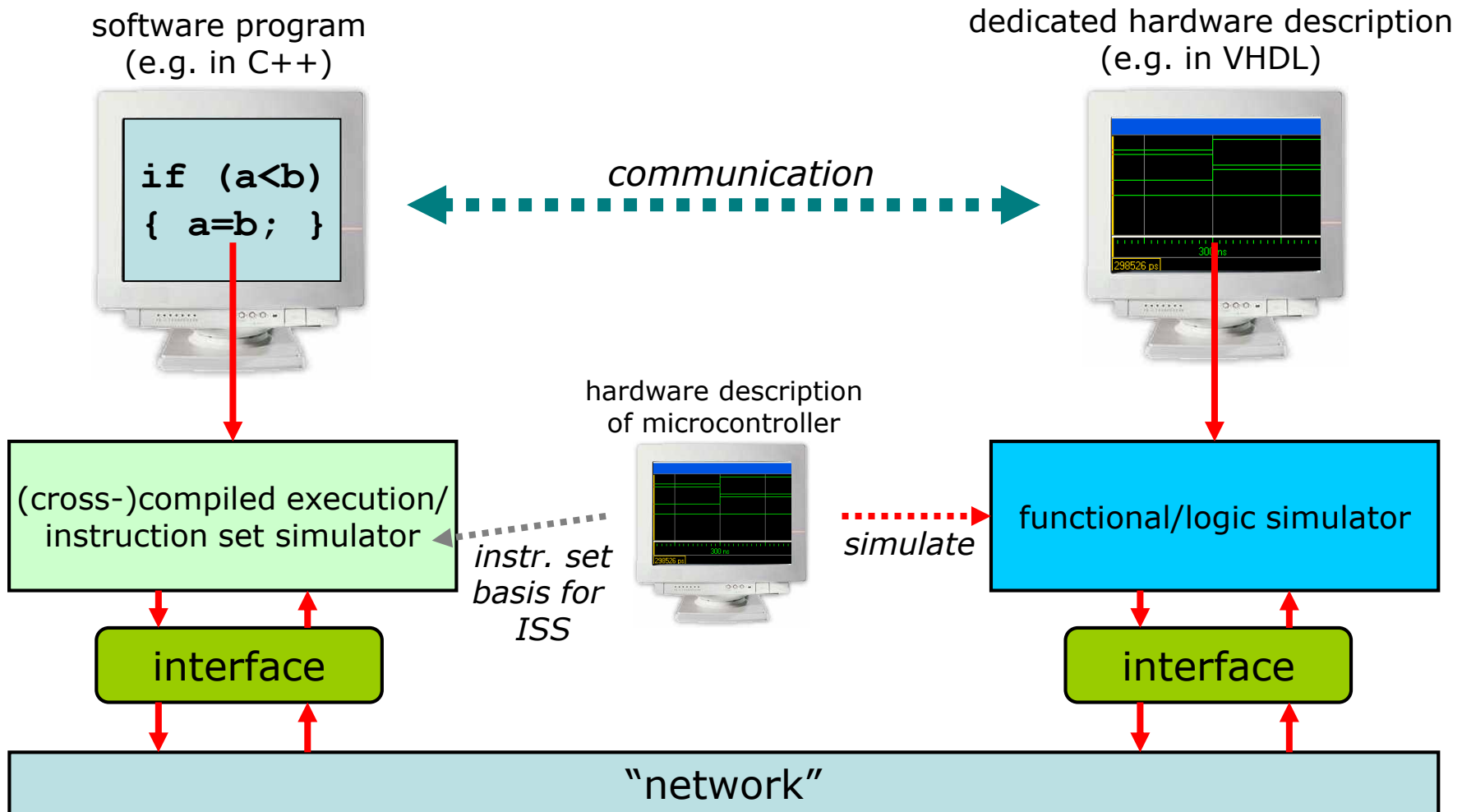
- Simulation
- Co-Simulation
- Heterogeneous Co-Simulation
- Alternatives

Heterogeneous Co-Simulation (I)

- use different simulators for _____

- advantages:
 - improve simulation speed
 - use adequate simulator with required level of simulation detail for different domains
- disadvantages:
 - common interface for simulators necessary
 - low timing details due to different simulation times on different domains

Heterogeneous Co-Simulation (II)



Runtime Problem

- what happens, when SW access HW?
 - SW sends request by using the interface to HW simulator and stops execution (→ idle)
 - simulator recognises request, loads corresponding inputs to simulation model and calculates the resulting state transitions
 - simulator answers the request by sending the corresponding outputs of simulation model
 - SW receives response and continues processing
- HW simulation runs a lot _____ than compiled/interpreted SW
- a lot of time is “wasted” in idle state
- “wasted” time depends on count of HW accesses by SW

Two Extremes

- interrupt controller in HW
 - interrupt controller “listens” on different ports and sends a signal to processor, if an event happened
 - **very low communication between SW and HW**
 - **very fast simulation using heterogeneous co-simulation**
- memory in HW
 - SW has to access simulated HW for every data, that is necessary
 - disastrous, if memory is used as program memory → every machine instruction has to be fetched from HW simulation
 - **a lot of communication between SW and HW**
 - **very slow simulation using heterogeneous co-simulation**

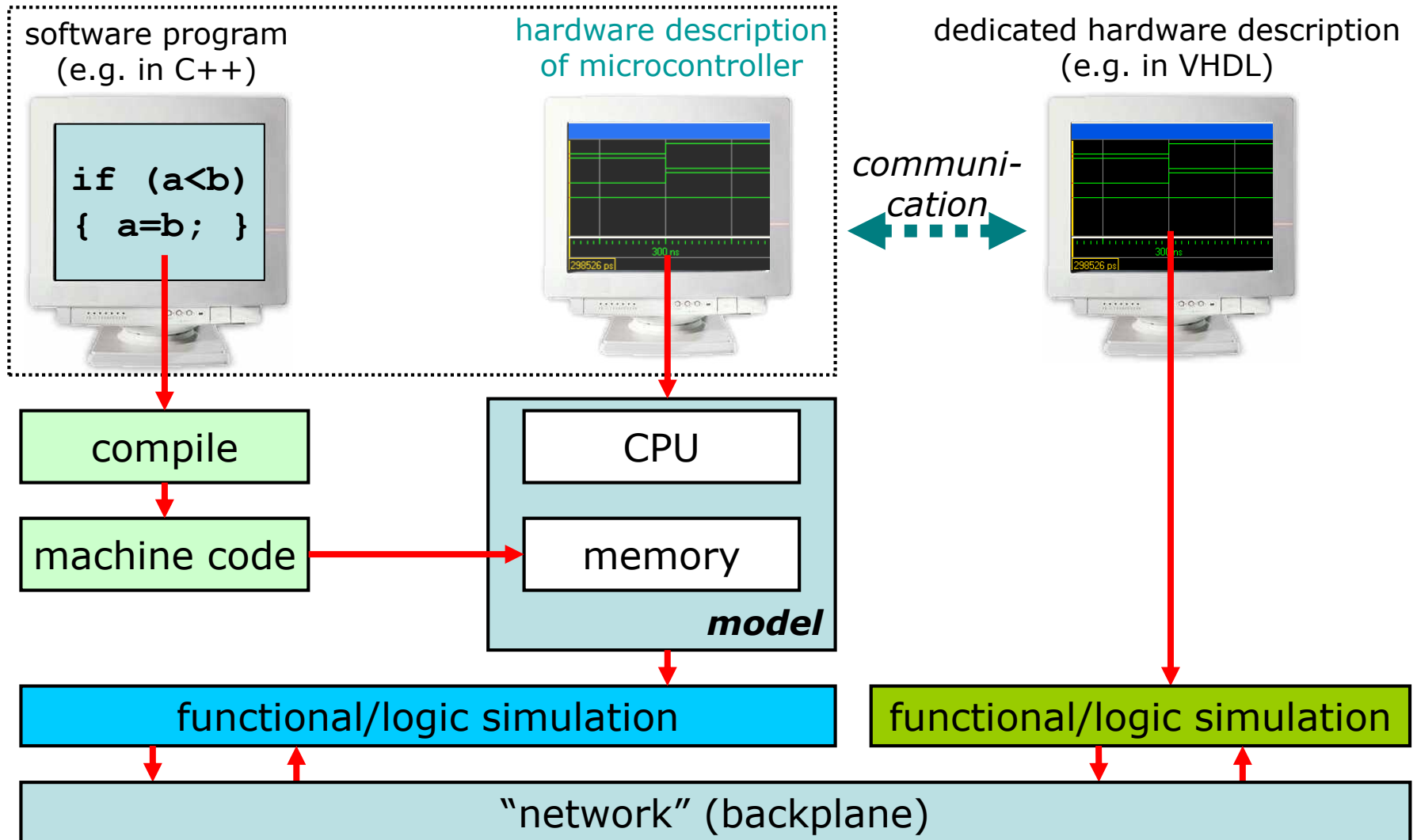
Possible Realisations

- depending on availability of target processor, related compiler, instruction set simulator, ... different concepts for distribution of simulation possible
- popular:
 - exact processor model
 - bus model
 - instruction set simulator model
 - compiled model
 - hardware model

Exact Processor Model (I)

- processor components (as memory, datapath, bus, instruction decoder, ...) are modelled as discrete event models
- modelled processor _____
(stored in the modelled instruction memory)
- interaction between processor and other hardware (running in another simulator) is captured using native event-driven simulation capability of simulator
- simulation speed depends on model
 - logic simulation is extremely slow (\sim tens of clock cycles/sec)
 - behavioral model is \sim hundred times faster

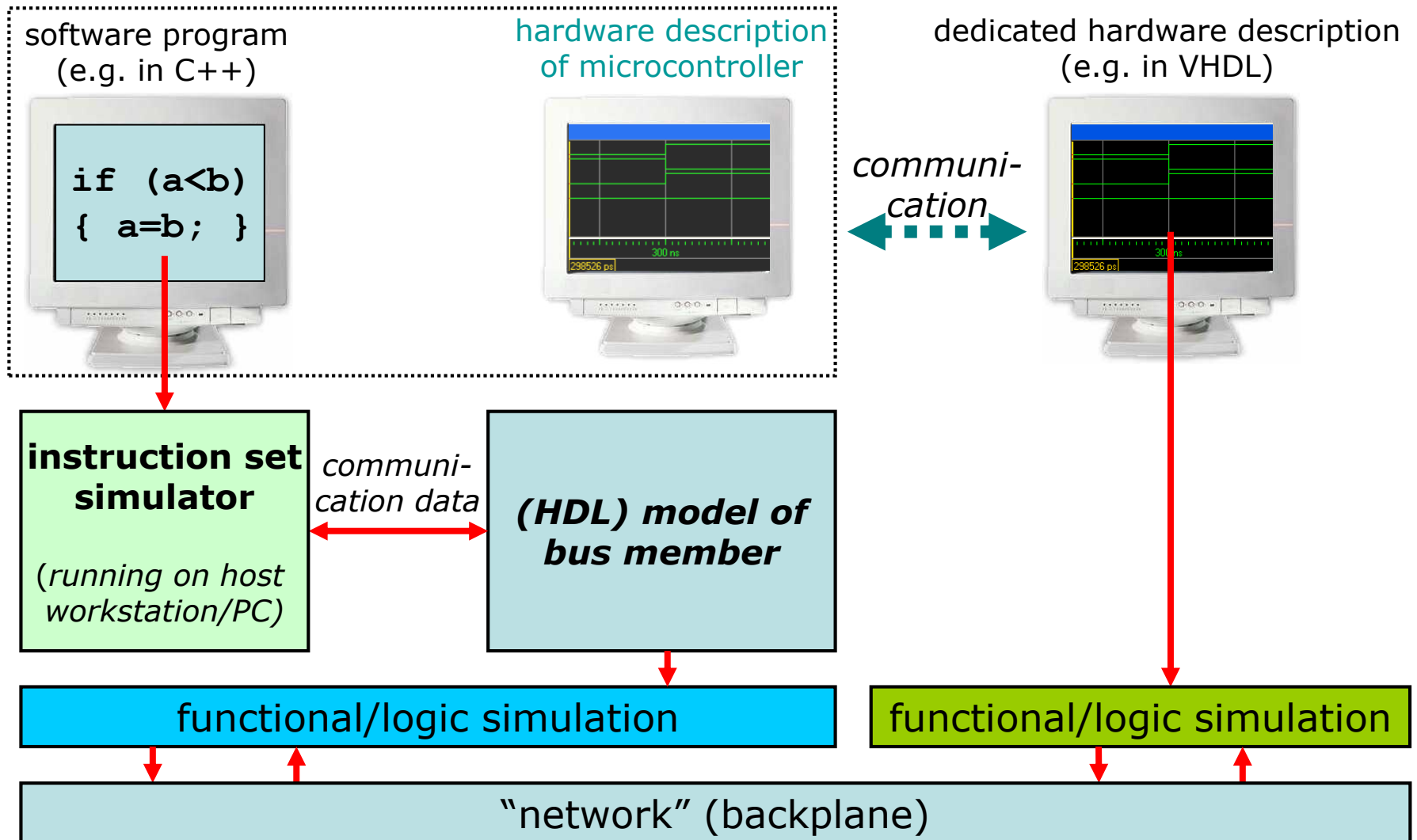
Exact Processor Model (II)



Bus Model (I)

- discrete-event model, that only simulate _____
_____ (bus connects processor and hardware of the developed system) without executing the software associated with the processor
- useful for low level interactions, such as bus and memory interaction
- software executed on instruction set simulator model and provide timing information in clock cycles for given sequence of instructions between pairs of I/O operations

Bus Model (II)

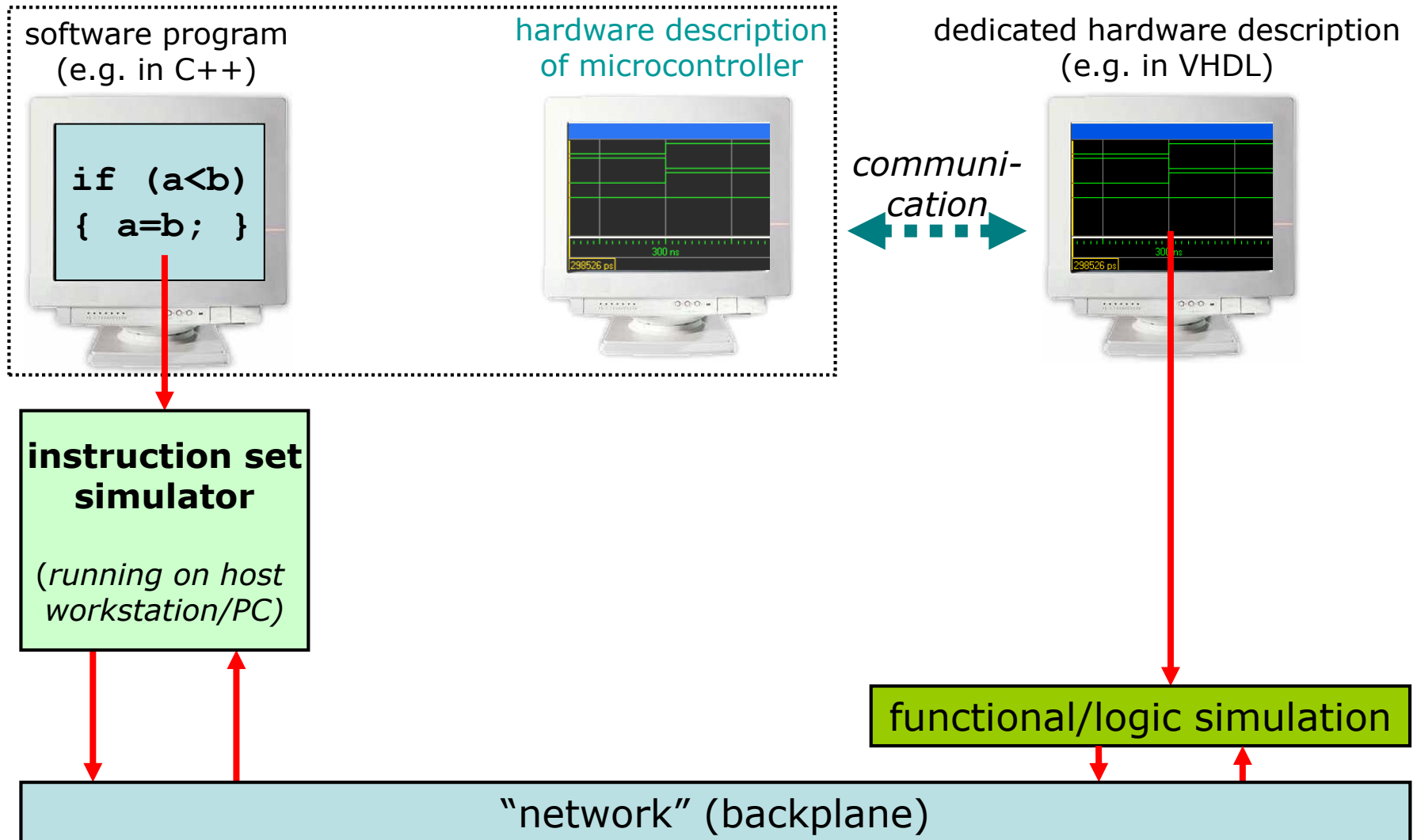


Instruction Set Simulator Model (I)

- instruction set of target processor can be simulated efficiently

- this C program is an interpreter for the embedded software
- embedded software executed on instruction set simulator program
- instruction set simulator provides timing (clock) details for the co-simulation.

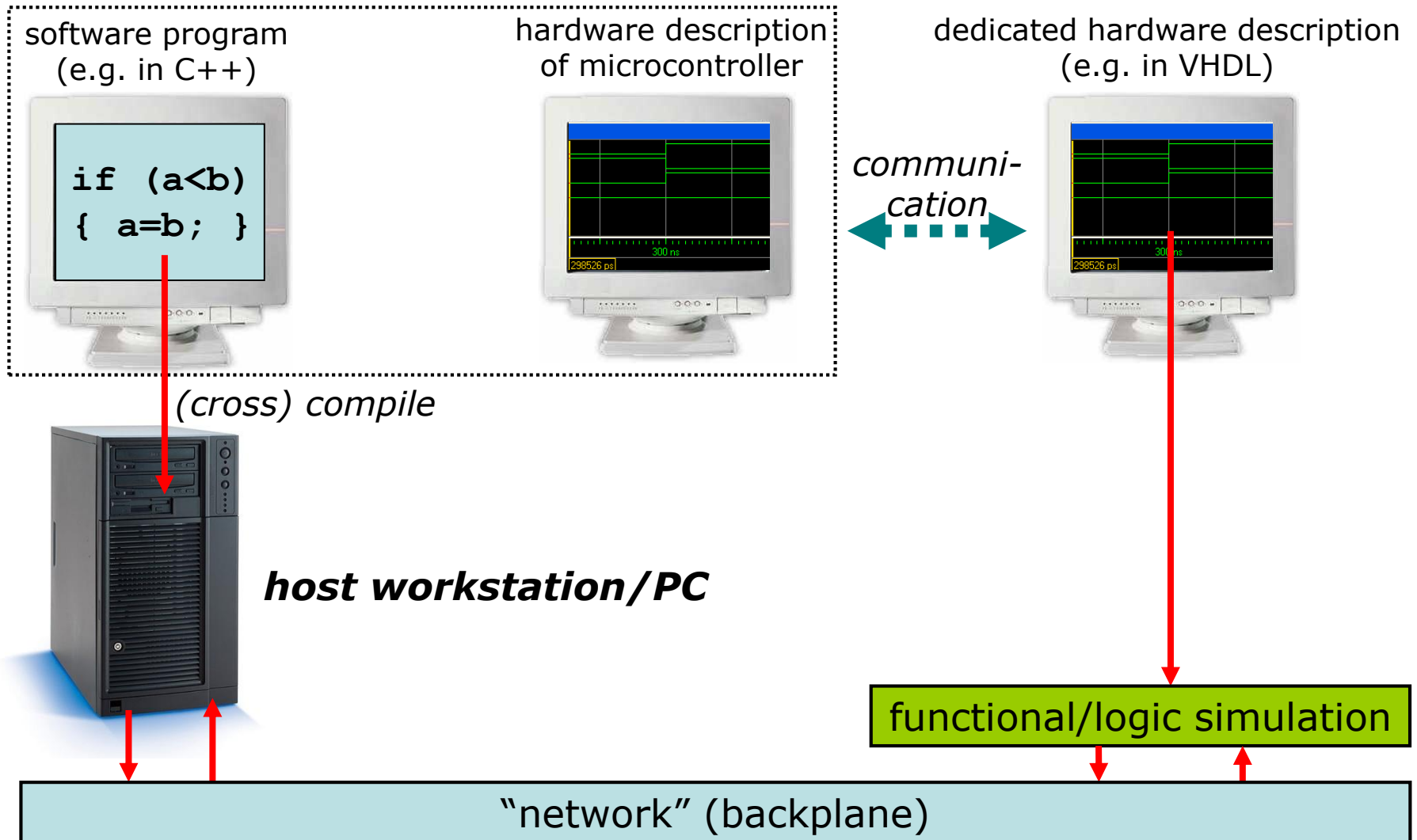
Instruction Set Simulator Model (II)



Compiled Model (I)

- very fast processor models are achievable in principle by translating the executable embedded software specification into _____ doing simulation
- e.g. code for programmable DSP can be translated into Sparc assembly code for execution on a workstation
- software execution provides timing details for co-simulation.

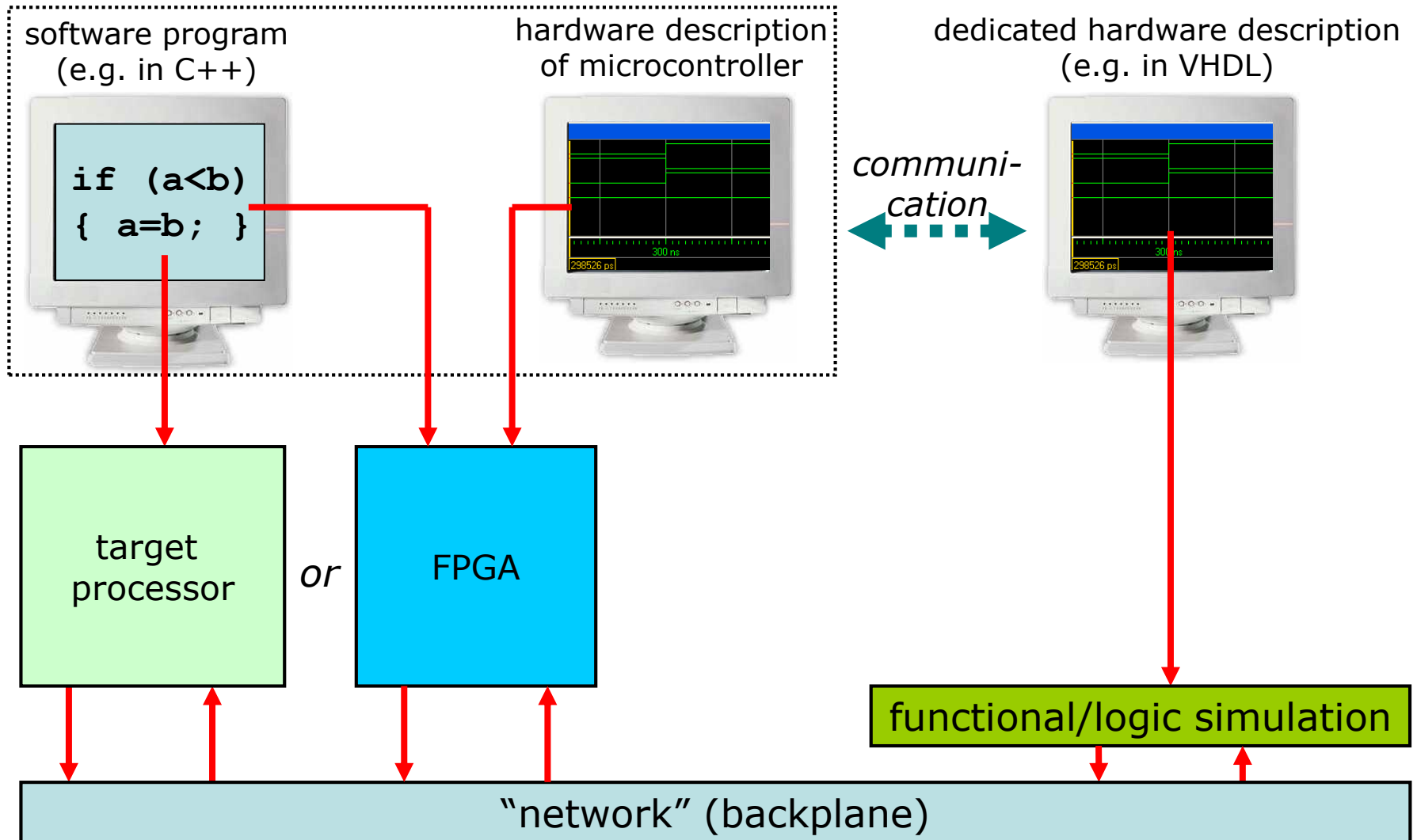
Compiled Model (II)



Hardware Model (I)

- if processor exists as hardware, it can be used to execute the embedded software in simulation
- alternatively, processor could be modelled _____
_____ (emulation/prototyping)
- interface to hardware simulator necessary
- difficult debugging during runtime

Hardware Model (II)



Comparison

- detailed processor model
 - most accurate model
 - slowest model
- bus model
 - less accurate but fast
- instruction set simulator model
 - more efficient than detailed processor model, if details of processor are not interesting
- compiled model
 - fastest model
- hardware model
 - high performance
 - processor hardware or emulation model necessary

Domain Coupling

- in three of the approaches, we have a host, that runs software
- interaction of host's software with hardware simulator necessary
- problems:
 - providing timing information _____
 - run two domains with different time models and execution times
properly synchronised

Synchronisation and Time

- in case of a single simulator, there is no problem for timing, as single _____ is managed for simulation
- several simulators and software programs in the domain (as usual in heterogeneous co-simulation)
 - hardware and software domain use a handshaking protocol to keep their time (clock) synchronised
 - signals from one domain to the other should have attached a time stamp
 - possible to use a loosely coupled strategy which allows the two domains to proceed more independently
 - run within own timing
 - if signal with a time stamp in past is received → roll back

Runtime Experiences of Heterogeneous Co-Simulation (I)

- in loosely coupled processor systems, the block responsible for hardware initializations has 30% instructions to access the hardware
- in general hardware density is important for simulation speed
- the simulation hardware and tools that communicate between the heterogeneous environment can contribute to the speed too
- if simulation is distributive (most often it happens these days), the network bandwidth, reliability and speed matters too

Runtime Experiences of Heterogeneous Co-Simulation (II)

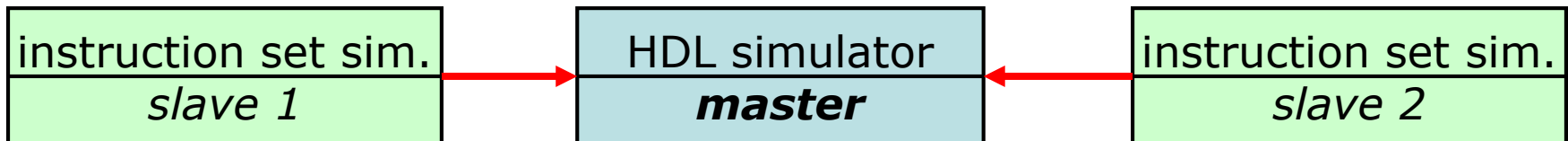
- if target processor is not PC, you may use cross compiler
- when software runs directly on target processor, it runs at “real speed”
- instruction set simulator usually runs at 20% of the speed of native processes

Contents

- Simulation
- Co-Simulation
- Heterogeneous Co-Simulation
- Alternatives

Master-Slave Co-Simulation

- one master simulator, several slave simulators
- slaves are invoked from master by _____
→ no concurrent simulation possible
- master defines global timing
- common interfaces for procedure calls necessary in all simulators



Distributed Co-Simulation

- simulate the system bus between processor (with embedded software) and hardware by software
- implementation of system bus _____
(e.g. Unix sockets, ...) → simulators may run on different workstations if system bus can be modelled using the network
- allows concurrency of simulators

