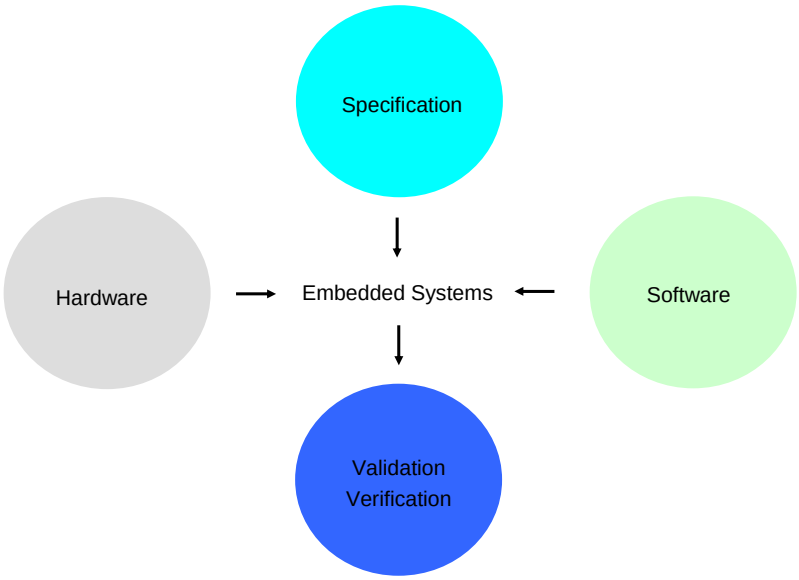
Embedded Systems

Bernd Finkbeiner
Calogero Zarba
Moritz Hahn

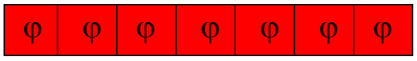
Sommersemester 2007

Course Recap



Linear-time Temporal Logic (LTL)

Specification

$\Diamond \varphi$	Eventually	
$\Box \varphi$	Henceforth	
$\varphi \mathcal{U} \psi$	Until	
$\bigcirc \varphi$	Next	

Safety vs. Reliability

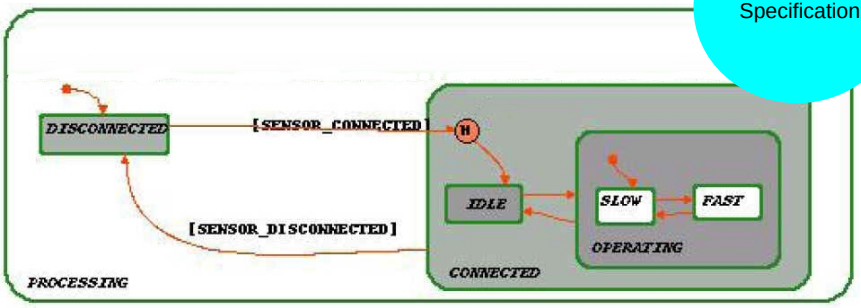
Specification

- Safe means *sufficiently low probability of serious harm caused by the system:*
  - e.g. ISO 8402: „State in which risk of harm (to persons) or damage is limited to an acceptable level.“
- *Reliable* means *sufficiently high probability of delivering intended service.*
  - Reliability is the probability of the system delivering the service it was designed for throughout the horizon, given
    - a defined temporal horizon
    - the operational conditions

Note: Safety and reliability are not synonyms!

StateCharts

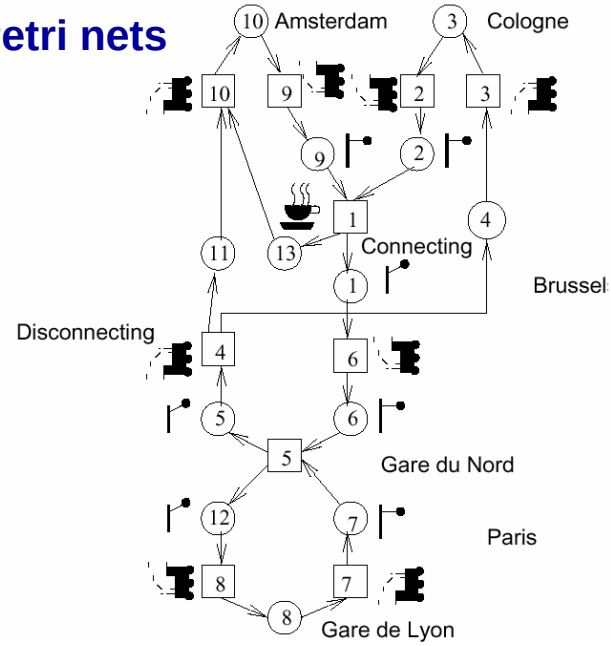
Specification



- Classical automata not concise enough for complex systems
  - ☞ Introduction of hierarchy
  - ☞ StateCharts [Harel, 1987]
- StateChart = *the only unused combination of „flow“ or „state“ with „diagram“ or „chart“*

Petri nets

Specification



VHDL

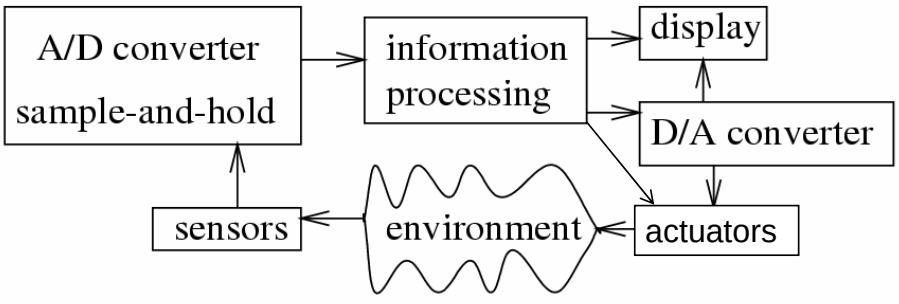
Specification

- Entities and (behavioral/structural) architectures
- Multiple-valued logic
  - General CSA approach
  - Application to IEEE 1164
- Modeling hardware parallelism by processes
  - Wait statements and sensivity lists
- VHDL semantics: the simulation cycle
  - δ cycles, deterministic simulation

Embedded System Hardware

Hardware

Embedded system hardware is frequently used in a loop (***„hardware in a loop“***):



Hardware

Design for Safety

Hardware

entails many facets, e.g.:

- Ensure safety of services:
  - Build a stringent safety case for each service, including
    - Functional requirements (state dynamics, request-answer-relation, …)
    - Non-functional requirements (power consumption, thermal impact, electromagnetic interference, resilience under radiation, …)
- Ensure correctness of the design:
  - if all components operate according to their design specifications then they satisfy the requirements
  - i.e., they deliver the desired service in accordance to safety requirements, unless unanticipated faults occur
- Enhance reliability:
  - Reduce risk of unanticipated faults
  - maintain safe operation under unanticipated faults

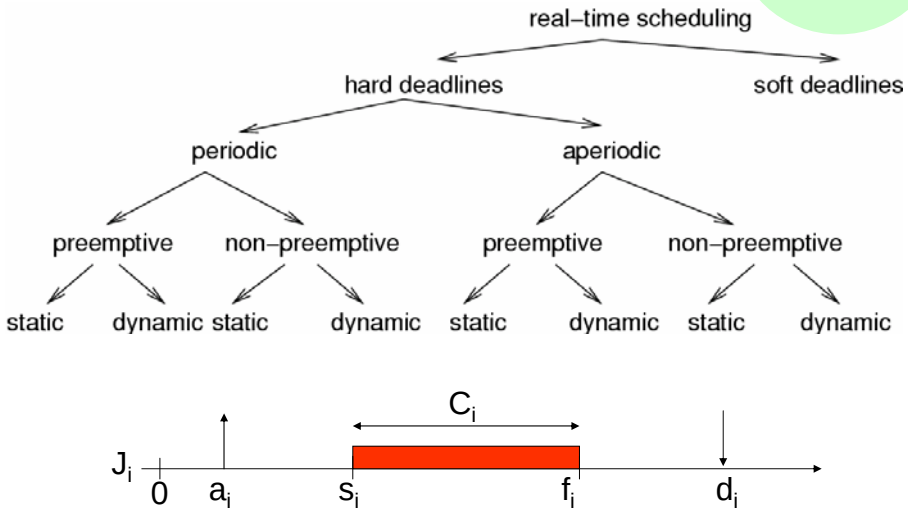
Counterfeiting unanticipated faults

Hardware

1. Fault avoidance
   - Tries to prevent faults from entering system during design stage (measures: high-level modeling, formal methods, …)
2. Fault removal
   - Find and remove faults before system enters service (testing, debugging, …)
3. Fault detection
   - Finds faults in the operational system. Is a prerequisite for
4. Fault tolerance
   - Allows the system to maintain correct service in presence of faults.

Scheduling

Software



Earliest Deadline First (EDF)

Software

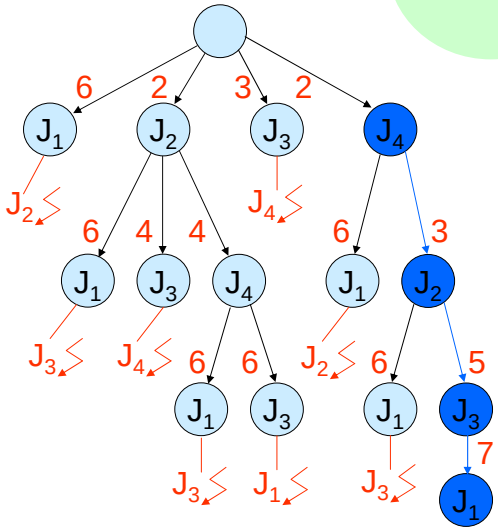
- **EDF** is a dynamic scheduling algorithm which **executes the current task with earliest deadline**.
- EDF is applicable to both periodic and a-periodic tasks.
- Preemptive: currently executing task is preempted by newly arrived task with earlier deadline.

Bratley's algorithm

Software

- Example:

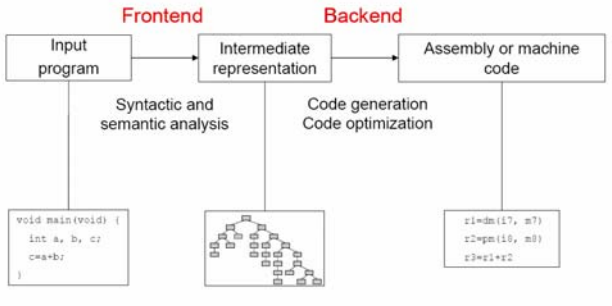
	J_1	J_2	J_3	J_4
a_i	4	1	1	0
C_i	2	1	2	2
d_i	7	5	6	4



Code Generation for Embedded Systems

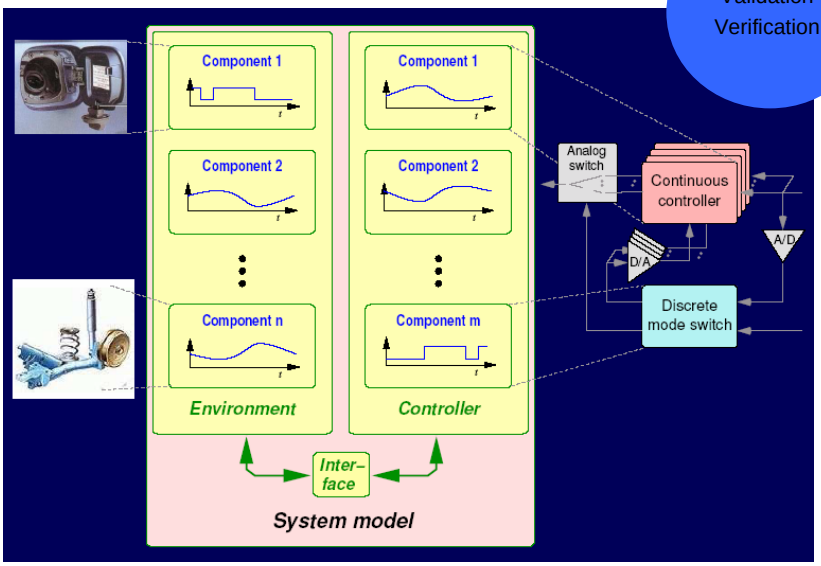
Software

Compiler Structure



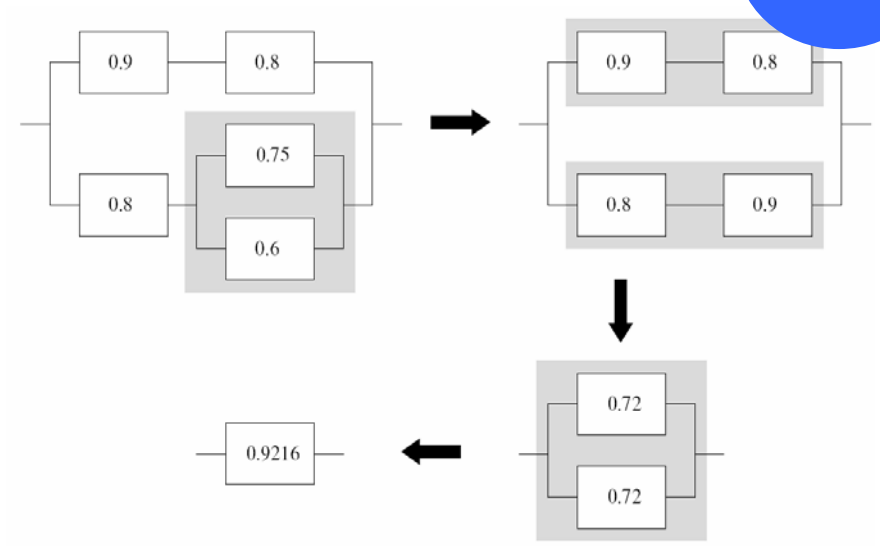
Simulation

Validation
Verification



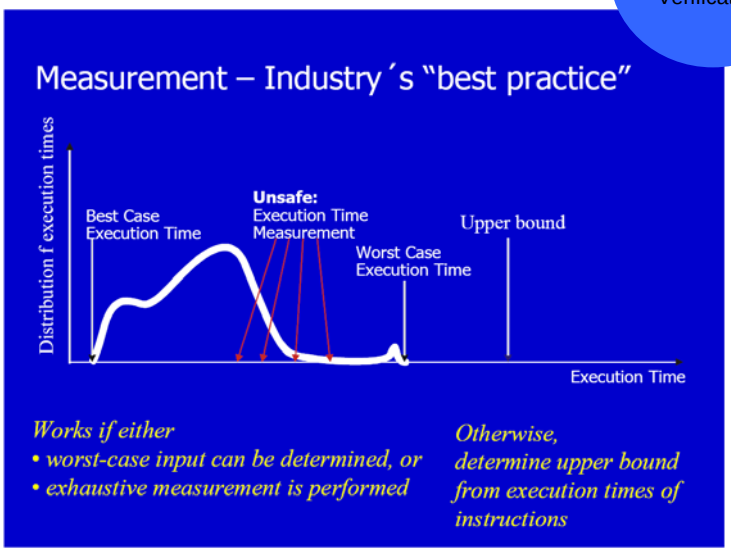
Reliability Analysis

Validation
Verification

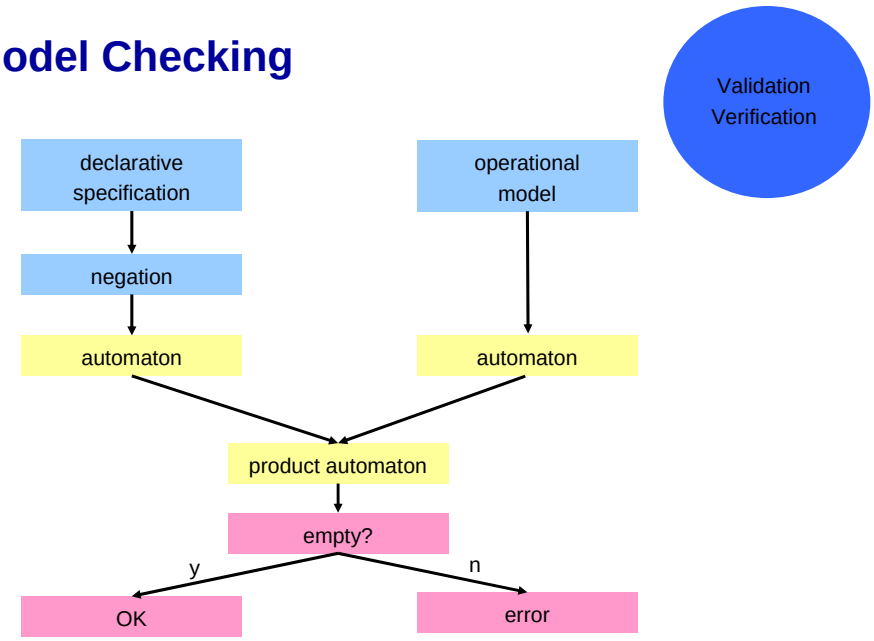


WCET Analysis

Validation
Verification



Model Checking



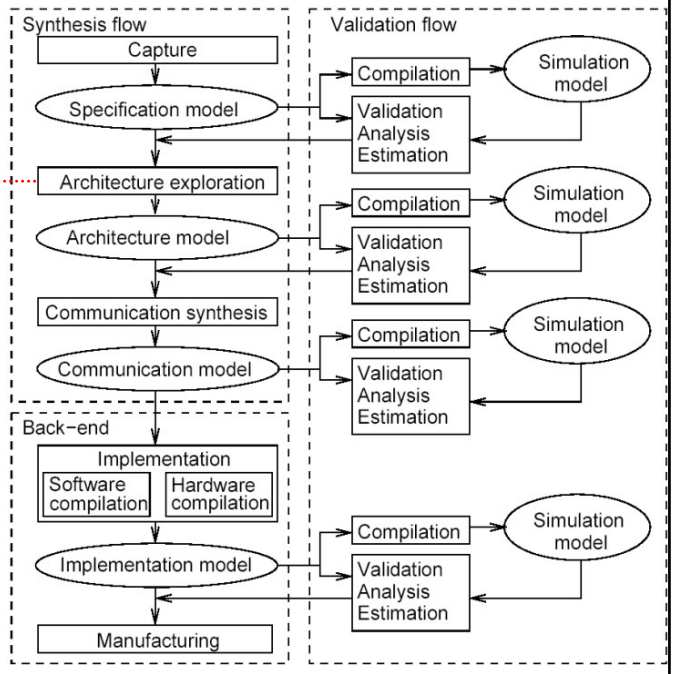
Actual design flows and tools

Design steps can be used in different combinations.
In the following, we will present some examples …

1. SpecC [Gajski, Dömer, Gerstlauer]
   Focus on intellectual property (IP) components

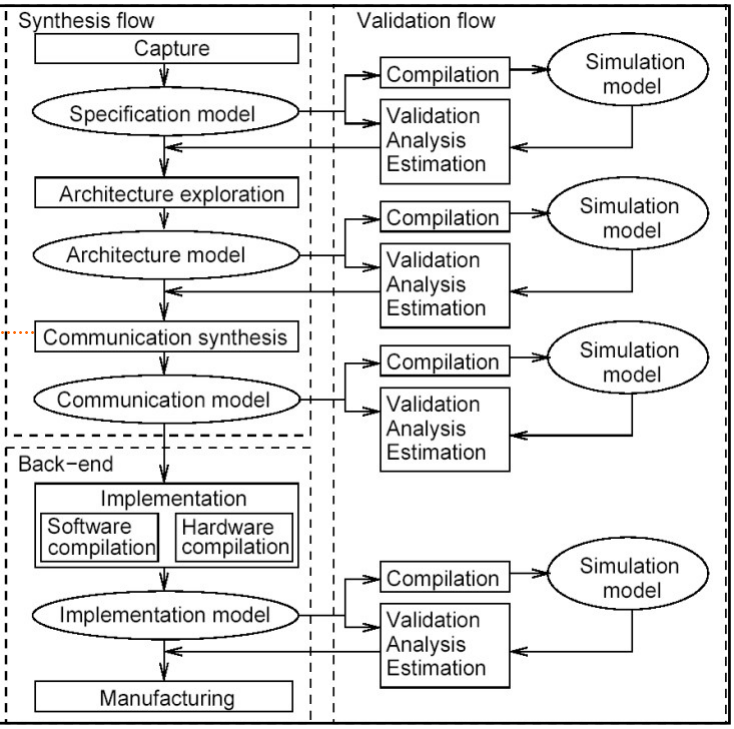
SpecC

Comprises allocation (selecting components from a library), partitioning (mapping of parts of the system specification onto the components) & scheduling (serialization of execution).

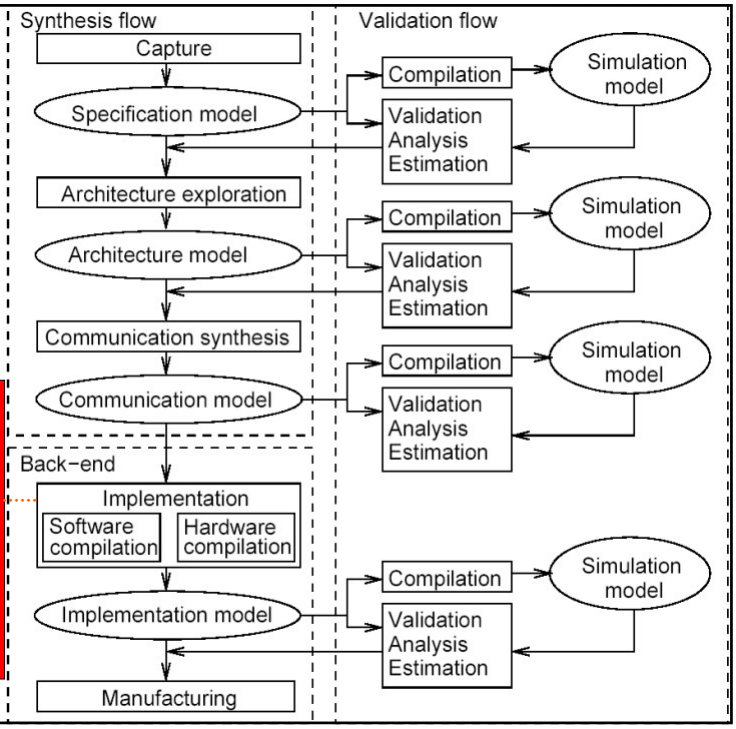


SpecC

abstract busses replaced by actual wires in a series of refinements



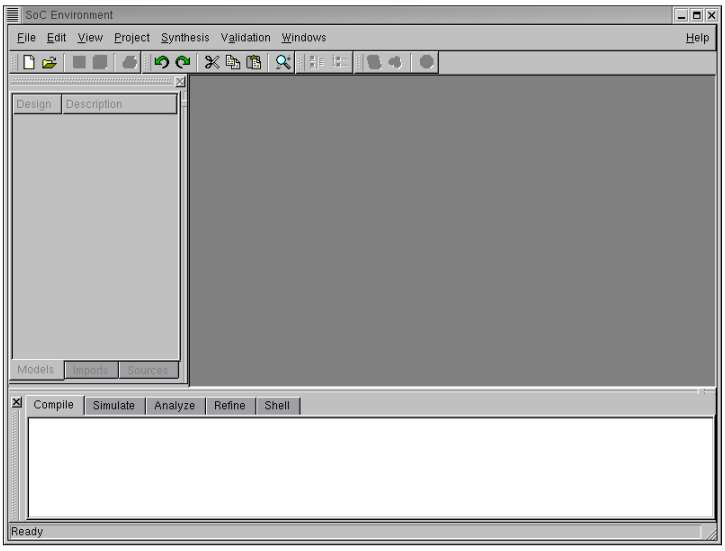
SpecC



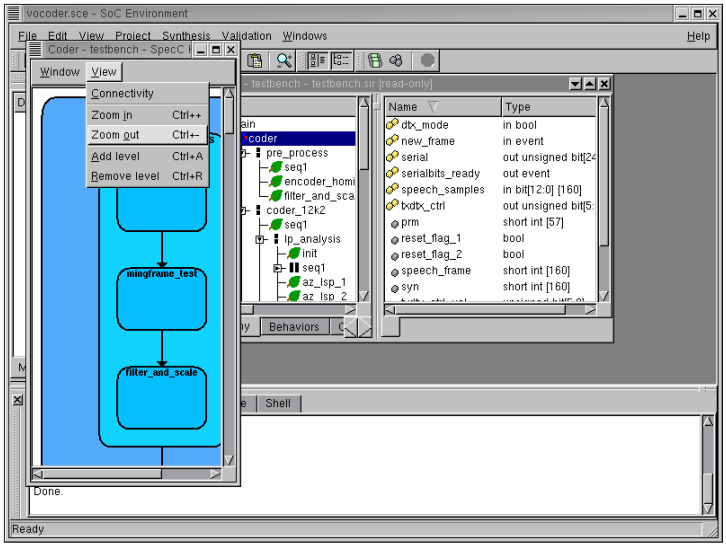
Compilers are used to generate binary machine code and hardware synthesis tools are used to generate custom hardware.

Bernd Finkbeiner

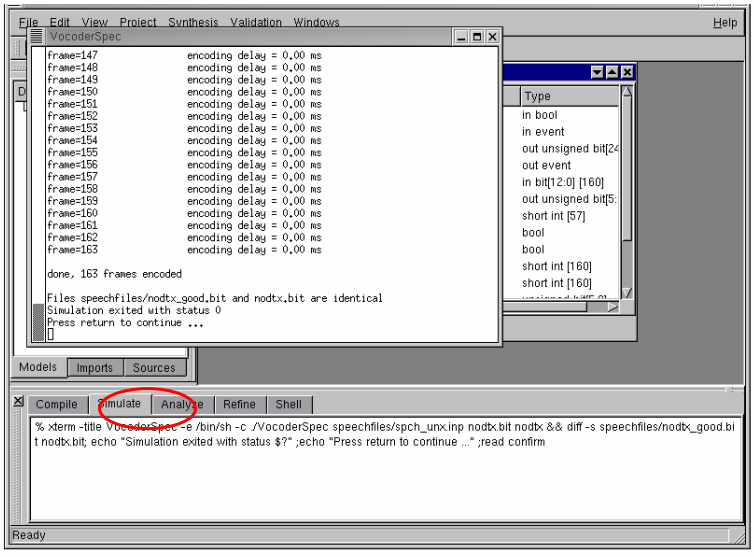
An actual example
- Getting started with the SCE window -



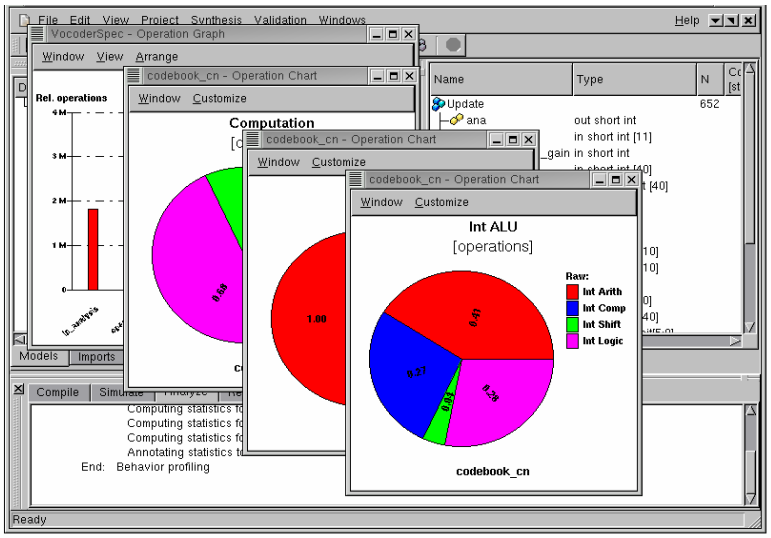
Browsing the specification



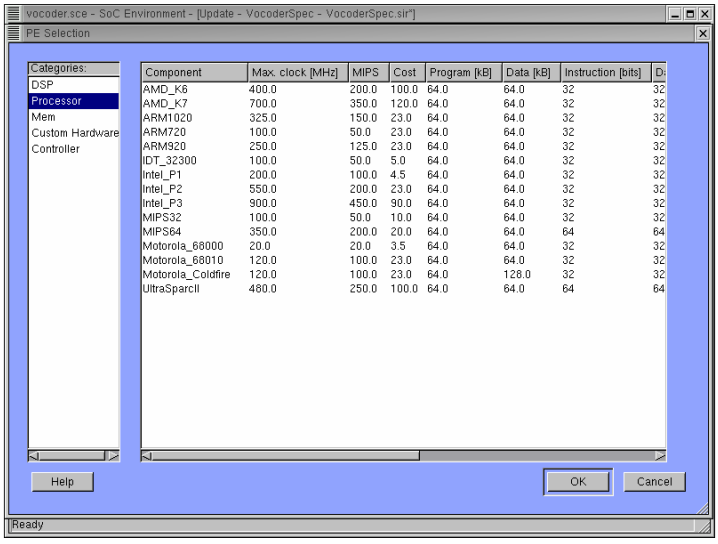
An actual example
- Validation by simulation -



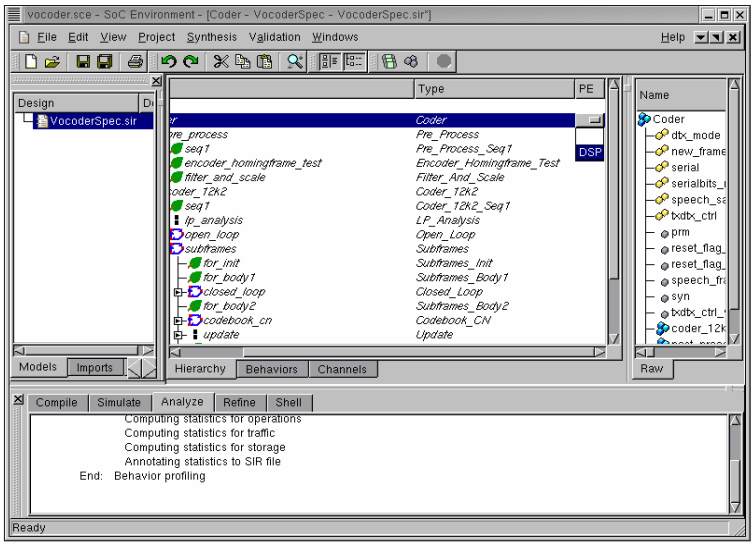
Analyze profiling results



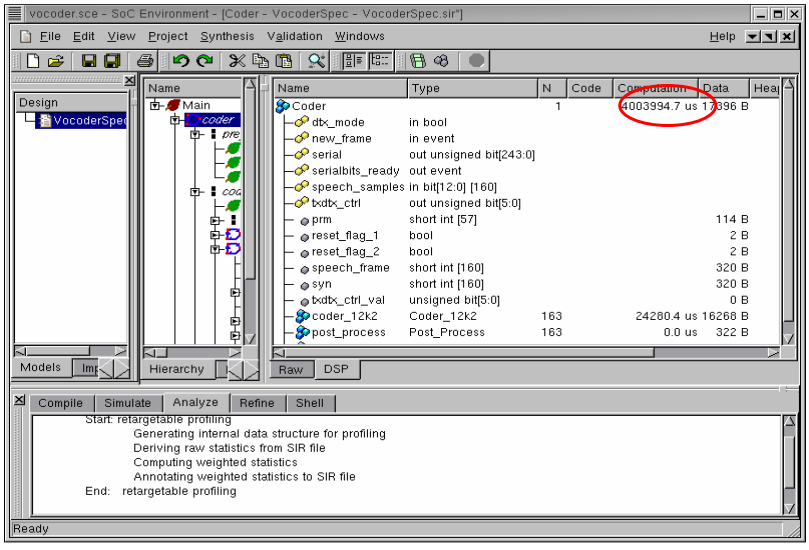
PE selection



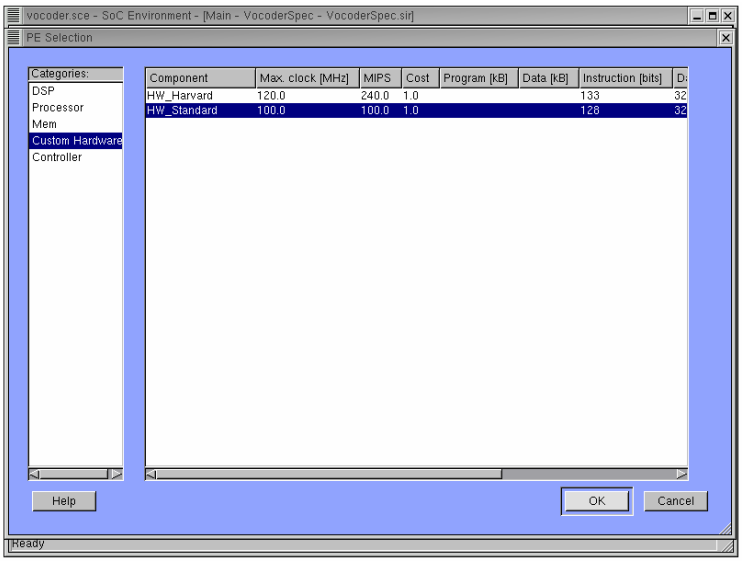
The top level behavior is mapped to the DSP



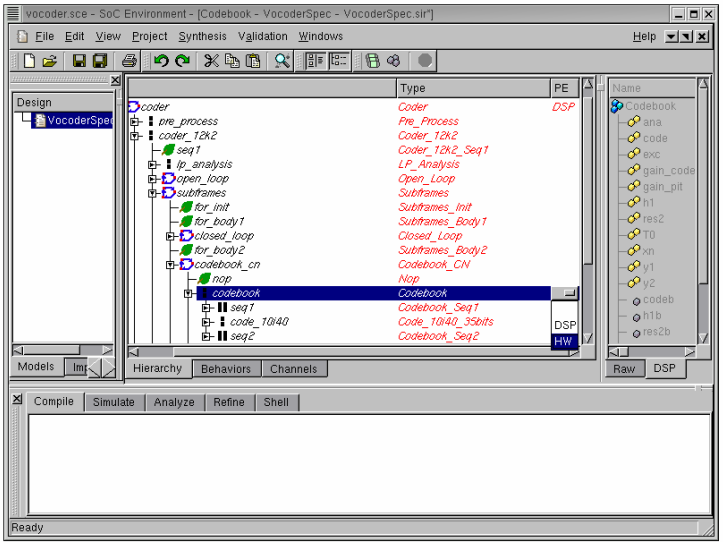
Estimate the performance (too slow)



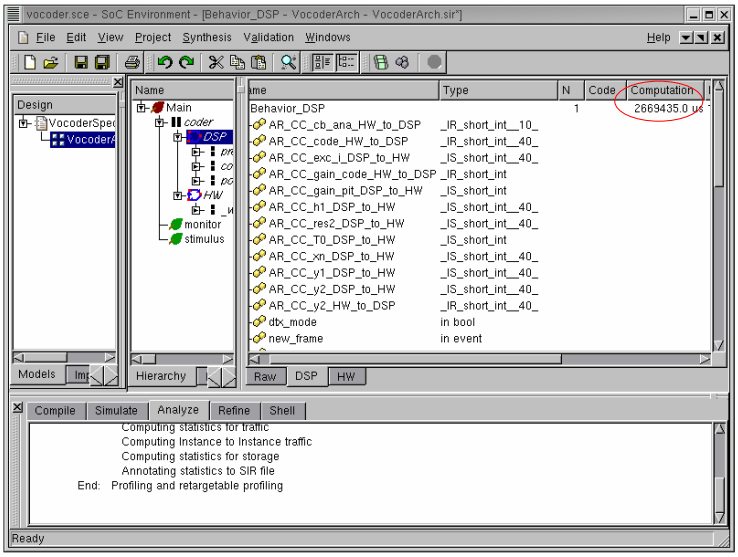
Selecting additional custom HW including datapath and controller



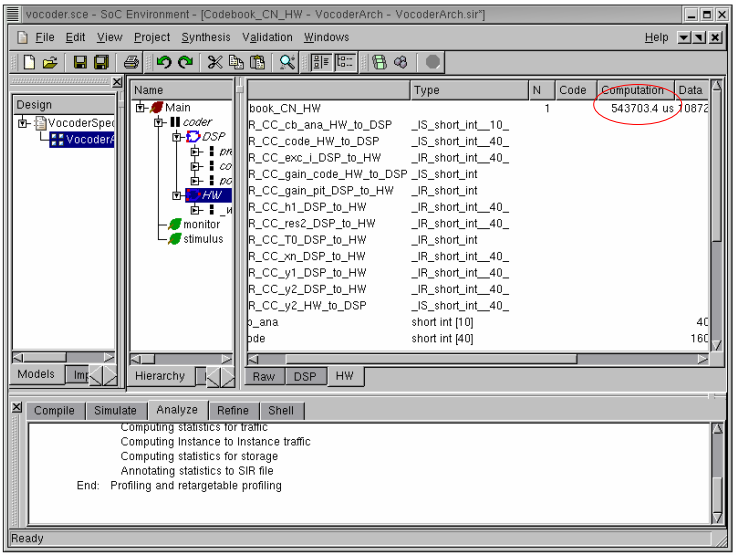
Binding „codebook“ to the custom datapath



Execution time of the DSP



Execution time for hardware

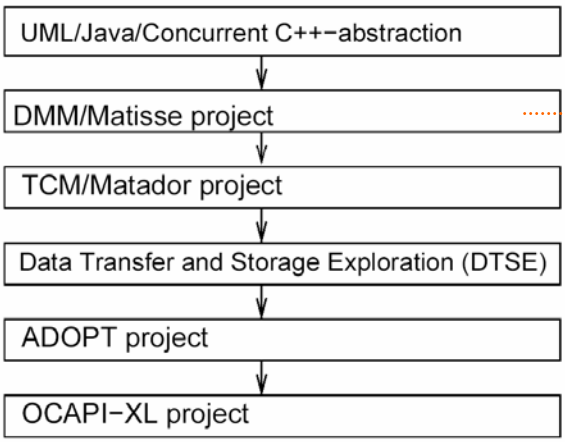


Assuming that 0.54+2.66 sec is acceptable

2. IMEC tool flow

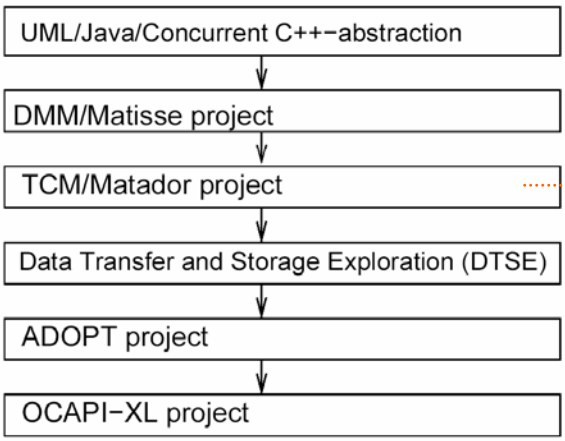
- IMEC = Interuniversitair Micro-Electronica Centrum,
  Leuven, Belgium
  (Large research facility)

Imec tool flow
- Global view -



Considers the system at the concurrent process level as a set of concurrent and dynamic processes, whose specification consists of algorithms, abstract data types, communication primitives, and real-time requirements. Tools can perform source code transformations on the dynamic data types & provide also a memory pool organization in the virtual memory space.

Imec tool flow
- Matador/TCM -



Again considering a system of concurrent processes. For these tools, the emphasis is on mapping tasks to processors. Different configurations of multi-processor systems are evaluated and curves of designs that are non-inferior to others are generated. These curves provide a view of the design space, and are the basis for final design decisions.

Matador/
Task Concurrency Management (TCM)

Wong et al. [Wong et al., 2001]: configurations for a personal MPEG-4 player: combination of StrongArm processors and custom accelerators. Found 4 configurations satisfying timing constraint of 30 ms.

Processor combination	1	2	3	4
Number of high speed processors	6	5	4	3
Number of low speed processors	0	3	5	7
Total number of processors	6	8	9	10

For combinations 1 and 4, only one allocation of tasks to processors meets the timing constraints. For combinations 2 and 3, different time budgets lead to different task to processor mappings and different energy consumptions.

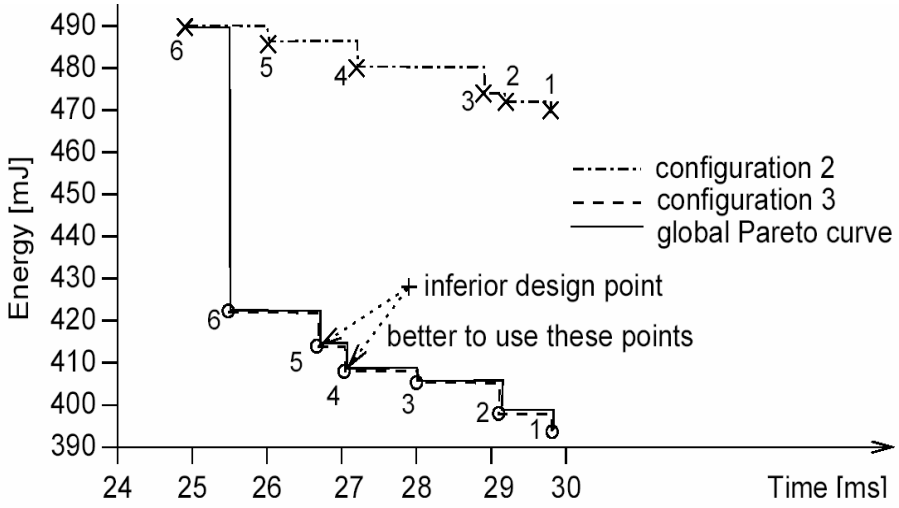
Pareto points

Multiobjective optimization:
– several conflicting criteria
- eg. performance vs. cost vs. power consumption

Definition: A (design) point J_i is **dominated** by point J_k, if J_k is equal or better than J_i in each criterion ($J_i \leq J_k$).

Definition: A (design) point is **Pareto-optimal** or a **Pareto point**, if it is not dominated by any other point.

Pareto curves

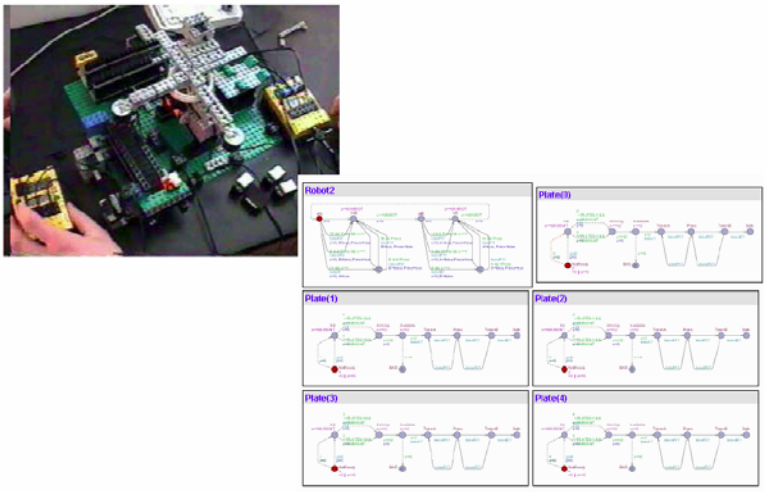


3. Ptolemy II (UC Berkeley)

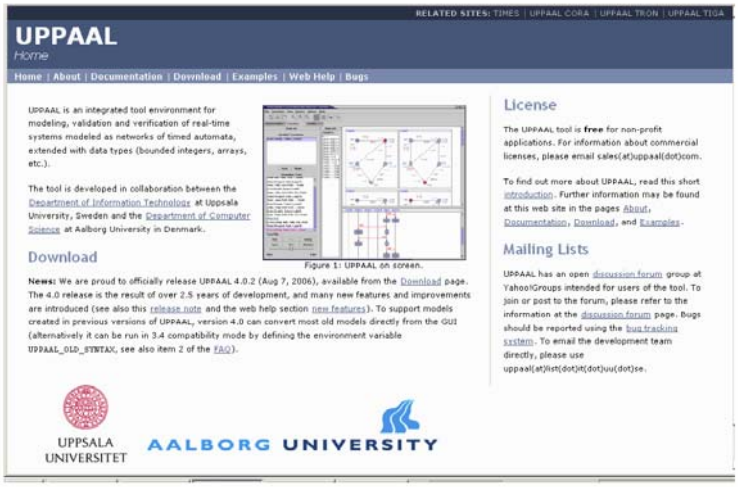
Ptolemy II supports specifications using different models of computation.

1. Communicating sequential processes (CSP).
2. Continuous time (CT):
   Supported by extensible differential equation solvers.
3. Discrete event model (DE):
   model used by many (e.g. VHDL) simulators.
4. Distributed discrete events (DDE).
5. Finite state machines (FSM).
6. Process networks (PN), using Kahn process networks
7. Synchronous dataflow (SDF)
8. Synchronous/reactive (SR) MoC. Discrete time, signals do not need to have a value at every clock tick. Esterel used.

4. Uppaal



Uppaal



Course Topics

1. StateCharts
2. Petri Nets
3. Kahn Process Networks, SDF

Midterm Exam

4. Dynamic Systems
5. Simulation
6. SDL
7. VHDL
8. Fault Tolerance
9. Reliability Analysis
10. Input Components
11. Communication
12. Processing Units
13. Scheduling
14. WCET Analysis
15. Code Generation for Embedded Processors

Final Exam

Backup Exam

16. Declarative specifications & Formal Verification

Exam Policy

- **Final:** July 26, 2007, 2:00pm
  in lecture hall 1 of the Math building.
  Length: 180 minutes.
- **Backup Exam:** September 28, 2007
- **Requirement for admission to final exam:**
  more than 50% points in homeworks
  (→ see list on web page, usual password)
- **Requirement for admission to backup exam:**
  more than 50% points in homeworks
  passing grade in either midterm or final (but not both)

Embedded Systems Development

- Daniel Kästner, Florian Martin, Marc Schlickling.
- Advanced course (6CP): Fr 12-14, Geb. E1.3, HS001. Übung, 2std.
- Goal: Working with industry tools for embedded systems development and understanding their theoretical background.
- Contents: Timed automata, model-based code generation, task scheduling and schedulability analysis, program analysis and worst-case execution time analysis, compilation for embedded processors.
- Tools used:
  - SCADE: CASE tool for safety-critical embedded systems (avionics)
  - SymTA/S: Task scheduling & schedulability analysis (automotive)
  - aiT WCET Analyzer: Worst-case execution time analysis (avionics & automotive)
- Practical project with LEGO Mindstorms.

Stammvorlesung Verification

- Bernd Finkbeiner, Anne Proetzsch, Lars Kuhtz
- 9 CP, Tuesdays and Thursdays 14-16
- Automata-theoretic verification
- Temporal logics
- Game logics
- Deductive verification
- CAV-Tools