Design Space Exploration of Systems-on-Chip: DIPLODOCUS

Ludovic Apvrille

Telecom ParisTech
ludovic.apvrille@telecom-paristech.fr

May, 2011

Outline

Outline

Systems-on-Chip

- A System-On-Chip = set of SW and HW components intended to perform a predefined set of functions for a given market
- Constraints
  - Right market window
  - Performance and costs

Design Challenges

Complexity

- Very high software complexity
- Very high hardware complexity

Problem

How to decide whether a function should be implemented in SW or in HW, or both?

Solution

Design Space Exploration!

Design Space Exploration

Design Space Exploration

- Analyzing various functionally equivalent implementation alternatives
- $\rightarrow$ Find an optimal solution

Important key design parameters

- Speed
- Power Consumption
- Silicon area
- Generation of heat
- Development effort

Level of Abstraction

Problematic

- Designers struggle with the complexity of today's circuits
- Cost of late re-engineering
  - Right decisions should be taken as soon as possible ...
  - And quickly (time to market issue), and so, simulations must be fast

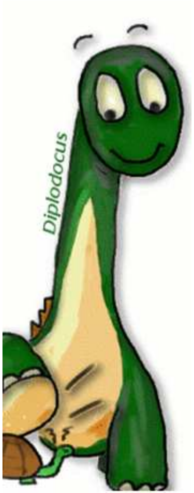
$\longrightarrow$ System Level Design Space Exploration

- Reusable models, fast simulations / formal analysis, prototyping can start without all functions to be implemented

But: high-level models must be closely defined so as to take the right decisions

Outline

DIPLODOCUS in a Nutshell



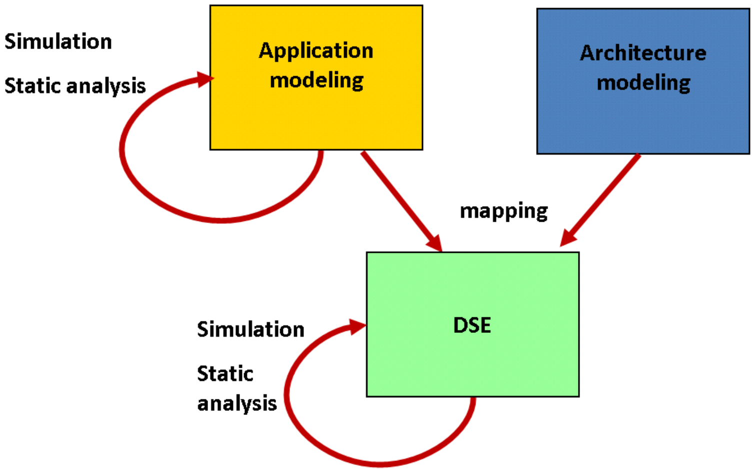
DIPLODOCUS = UML Profile

- System-level Design Space Exploration
- Y-Methodology
- MARTE compliant

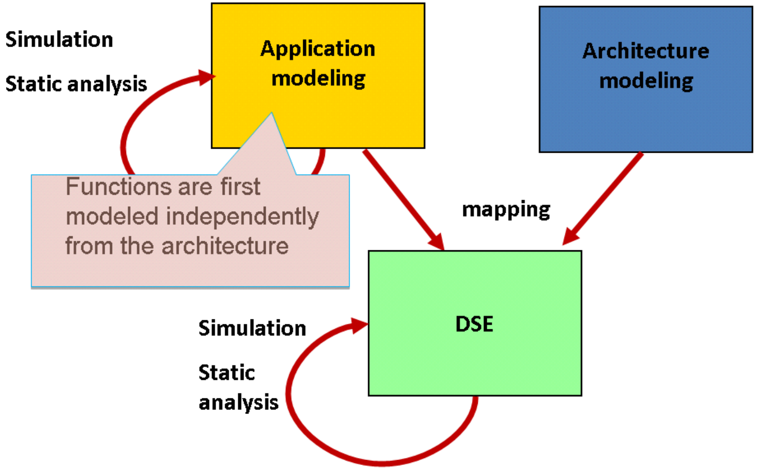
Main features

- Data are abstracted
- Formal semantics
- Very fast simulation support
- Fully supported by an open-source toolkit
  - TTool

The Y-Methododology

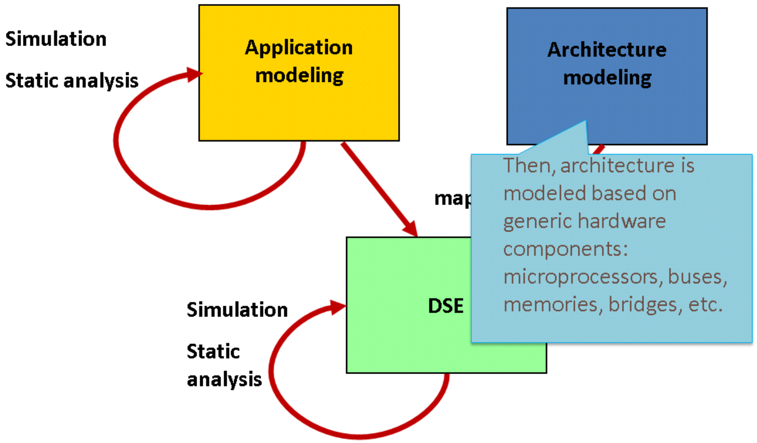


Application Modeling



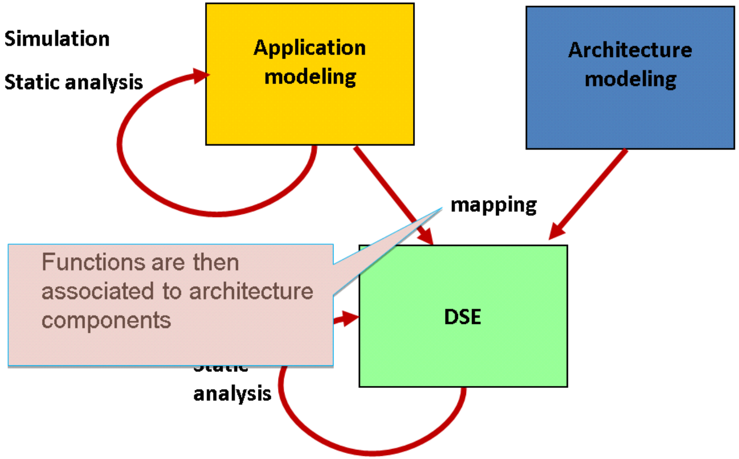
Functions are first modeled independently from the architecture

Architecture Modeling



Then, architecture is modeled based on generic hardware components: microprocessors, buses, memories, bridges, etc.

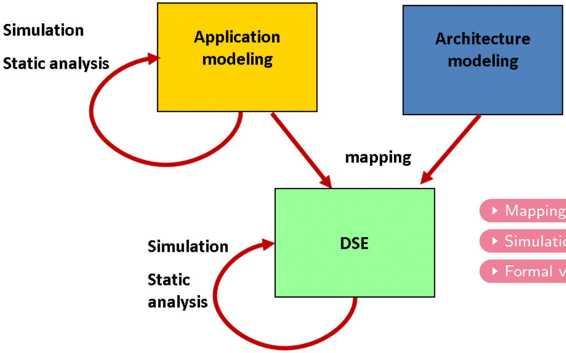
Mapping



Functions are then associated to architecture components

Browsing the DIPLODOCUS Methododology

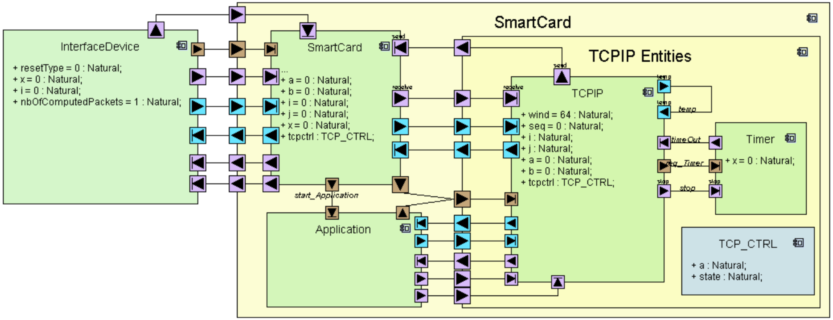
- Application structure
- Application behavior
- Formal verification



- Architecture model

- Mapping model
- Simulation
- Formal verification

Application Structure (Smart Card system)



Back to methodology

Application Behavior

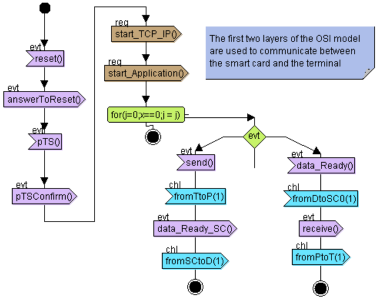
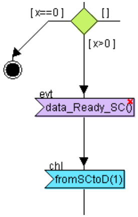


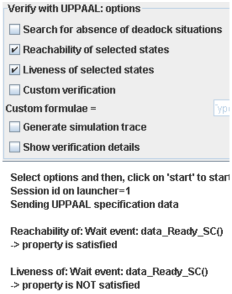
Figure: Activity Diagram of the SmartCard component

Back to methodology

Formal Verification at Application Level

- No assumption on the underlying architecture
- All possible interleavings between actions are considered
- Formal verification is based on LOTOS or UPPAAL
  - Press-button approach




Verify with UPPAAL: options

- [ ] Search for absence of deadock situations
- [x] Reachability of selected states
- [x] Liveness of selected states
- [ ] Custom verification

Custom formulae =

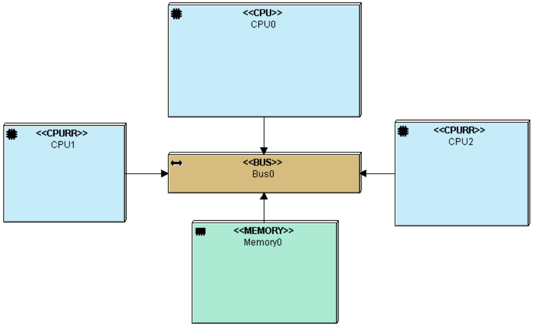
- [ ] Generate simulation trace
- [ ] Show verification details

```
Select options and then, click on 'start' to start
Session id on launcher=1
Sending UPPAAL specification data

Reachability of: Wait event: data_Ready_SC()
-> property is satisfied

Liveness of: Wait event: data_Ready_SC()
-> property is NOT satisfied
```

Back to methodology

Architecture

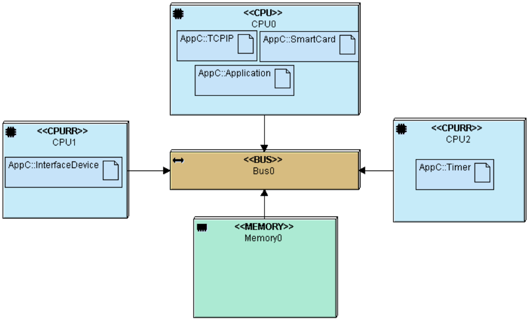
- Given in terms of parameterized nodes
- CPU, HWA, Bus, Memory, Bridge, etc.
- CPU parameters: scheduling policy, cache miss ratio, miss-branching prediction, pipeline size, etc.



Back to methodology

Mapping

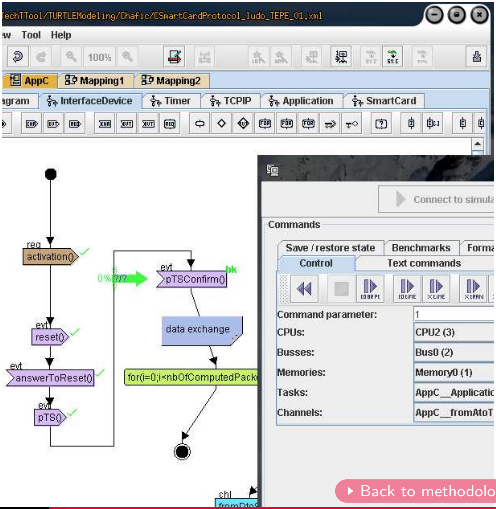
- Task are mapped on execution nodes (e.g., CPUs, HWAs)
- Channels are mapped on communication and storage nodes



Back to methodology

After-Mapping Simulation

- TTool Built-in simulator
- **Extremly fast**
- Diagram animation
- Step-by-step execution, breakpoints, etc.

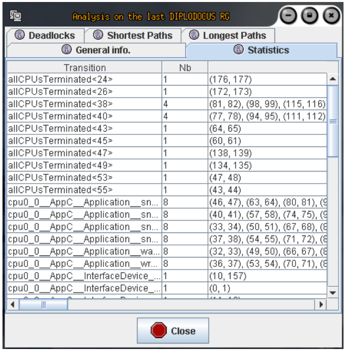


Command parameter: 1
CPUs: CPU2 (3)
Busses: Bus0 (2)
Memories: Memory0 (1)
Tasks: AppC__Applicatio
Channels: AppC__fromAtoT

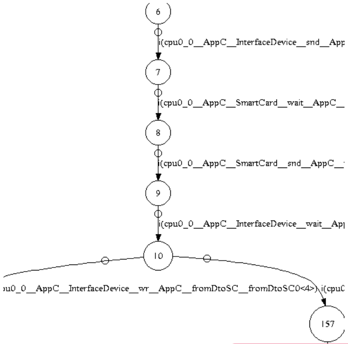
▸ Back to methodology

After-Mapping Formal Verification

- TTool built-in simulator can compute all possible execution paths
- Graph analysis and visualization



Transition	Nb	
allCPUsTerminated<24>	1	(176, 177)
allCPUsTerminated<26>	1	(172, 173)
allCPUsTerminated<38>	4	(81, 82), (98, 99), (115, 116)
allCPUsTerminated<40>	4	(77, 78), (94, 95), (111, 112)
allCPUsTerminated<43>	1	(64, 65)
allCPUsTerminated<45>	1	(60, 61)
allCPUsTerminated<47>	1	(138, 139)
allCPUsTerminated<49>	1	(134, 135)
allCPUsTerminated<53>	1	(47, 48)
allCPUsTerminated<55>	1	(43, 44)
cpu0_0__AppC__Application__sn...	8	(46, 47), (63, 64), (80, 81), (9
cpu0_0__AppC__Application__sn...	8	(40, 41), (57, 58), (74, 75), (9
cpu0_0__AppC__Application__sn...	8	(33, 34), (50, 51), (67, 68), (8
cpu0_0__AppC__Application__sn...	8	(37, 38), (54, 55), (71, 72), (8
cpu0_0__AppC__Application__wa...	8	(32, 33), (49, 50), (66, 67), (8
cpu0_0__AppC__Application__wr...	8	(36, 37), (53, 54), (70, 71), (8
cpu0_0__AppC__InterfaceDevice_...	1	(10, 157)
cpu0_0__AppC__InterfaceDevice_...	1	(0, 1)


6
i(cpu0_0__AppC__InterfaceDevice__snd__App
7
i(cpu0_0__AppC__SmartCard__wait__AppC__
8
i(cpu0_0__AppC__SmartCard__snd__AppC__
9
i(cpu0_0__AppC__InterfaceDevice__wait__Ap
10
pu0_0__AppC__InterfaceDevice__wr__AppC__fromDtoSC__fromDtoSC0<4>)
i(cpu0
157

▸ Back to methodology

TTool: Main Features

- Open-source UML toolkit
- Meant to support UML2 profiles
    - 8 UML2 profiles are currently supported
- Mostly programmed in Java
    - Editor, interfaces with external tools
    - Simulators and model-checkers are programmed in C++ or SystemC
- Formal verification and simulation features
    - Hides formal verification and simulation complexity to modelers
    - Press-button approach

Outline

Outline

Results

Fully integrated environment for the fast design of Systems-on-Chip

- Based on UML
- Open-source toolkit

Support

evita

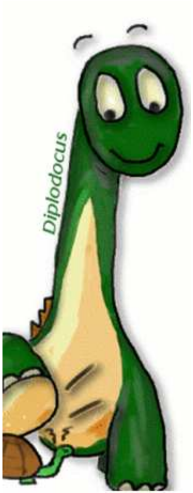
Texas Instruments

freescale semiconductor

A Few Case Studies ...

- MPEG coders and decoders (Texas Instruments)
- LTE SoC (Freescale)
- Partitioning in vehicle embedded systems (EVITA project)

To Go Further ...



TTool and DIPLODOCUS

- ttool.telecom-paristech.fr
  - Type *TTool UML* under *google*
  - And click on the *I am lucky* button!