System Specification

- **Context:** Systems are heterogeneous
  - Abstraction levels: clock-cycle accurate, system state, message and transaction level communication, ...
  - Execution models: native execution, model simulation, ISS, ...
  - Domains: HW, SW, RF, ..., environment.

- **Challenges:**
  - Specify interconnect for heterogeneous objects
  - Execution model for heterogeneous systems

- **Requirements**
  - Modular design, separation communication/computation
  - Multi and mixed level Co-simulation
Outline

1. Multiprocessors SoC (MP SoC)
   1.1. Multiprocessor SoC
   1.2. Multiprocessor SoC design
   1.3. This Course

2. Specification and validation of electronic systems
   2.1. Basic concepts
   2.2. Specification languages
   2.3. Heterogeneous systems modeling and validation

3. COLIF: A Design Model for MP SoCs
   3.1. COLIF: the Meta-model and the external syntax
   3.2. Mixed and multilevel model execution

4. A VDSL design example
   4.1. The application
   4.2. The design process

5. Summary
Multi-Processor System on Chip

- SoC: put on a chip what we used to put on one or several boards: 90% of ASICs in 2003
  - Heterogeneous components (CPU, memory, bus, ASICs, non digital)
  - Focus on architecture design to meet performances. Generic technology TTM, you don’t have to reinvent the O every day.

- Multi-processor
  - More than one instruction set processor on chip
  - Complex on-chip HW/SW communication network

- Multi-processor System on chip:
  - Tighter TTM constraints
  - Application-specific architecture
  - Design automation required
## Multi-processor SoC Examples (with Silicon share >> CPUs)

<table>
<thead>
<tr>
<th>Components</th>
<th>Data Computation</th>
<th>Control Computation</th>
<th>On Chip Memory</th>
<th>On Chip Communication</th>
<th>Specific Logic</th>
<th>Typical design</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wireless terminal, XDSL</td>
<td>1 DSP</td>
<td>1 MCU</td>
<td>&gt; MB</td>
<td>Bridge</td>
<td>&gt; M gates</td>
<td>STEP 1, VDSL (ST)</td>
</tr>
<tr>
<td>Multimedia</td>
<td>Few DSPs</td>
<td>1 MCU</td>
<td>&gt;&gt; MB</td>
<td>Network switch, cross bar</td>
<td>&lt; M gates</td>
<td>TRIMEDIA (Philips)</td>
</tr>
<tr>
<td>Network processor</td>
<td>Many DSPs</td>
<td>Few MCUs</td>
<td>&gt;&gt; MB</td>
<td>On chip network</td>
<td>&gt; M gates</td>
<td>IXPIZDE (INTEL)</td>
</tr>
<tr>
<td>Game processors</td>
<td>Few DSPs</td>
<td>Few MCUs</td>
<td>&gt;&gt; MB</td>
<td>On chip hierarchical network</td>
<td>&gt;&gt; M gates</td>
<td>Play station</td>
</tr>
</tbody>
</table>
System-on-Chip Architectures

- SoC Architecture:
  - HW Components
  - Application Software
  - HW-SW Communication
    - On Chip network (HW)
    - support Package (SW)
    - Programming Layer (SW)
- Multilevel APIs required
- Design Automation
  - HW component design: OK
  - SW Application design: OK
  - Architecture design: Still need to be invented

Application SW

- Resources management
- SW Communication (drivers, I/O, interrupts)
- On-chip HW Communication Network

HW Components
- CPUs, (DSP, MCU)
- IP, (ASICs, COTs)

SW
- OS

HW/SW trade-offs

Ahmed A. Jerraya
Outline

1. Multiprocessors SoC (MP SoC)
   1.1. Multiprocessor SoC
   1.2. Multiprocessor SoC design
   1.3. This Course

2. Specification and validation of electronic systems
   2.1. Basic concepts
   2.2. Specification languages
   2.3. Heterogeneous systems modeling and validation

3. COLIF: A Design Model for MP SoCs
   3.1. COLIF: the Meta-model and the external syntax
   3.2. Mixed and multilevel model execution

4. A VDSL design example
   4.1. The application
   4.2. The design process

5. Summary
Multi-Processor SoC Design

- Design: produce an RTL multi-processor architecture and the corresponding support software from system specification
- Key decisions:
  - Multiple vs. single OS
  - Static vs. dynamic allocation of tasks
  - Existing OS vs. Generated OS
  - Architecture platform vs. IP assembling
- System specification:
  - Abstract components/functions
  - Abstract interconnect model
  - Architecture design decisions
- Design steps:
  - Architecture definition & specilization
  - Application mapping
  - HW/SW communication design
  - Validation at different design steps

System Specification

- A
- B
- C

RTL Architecture

- μP
- DSP
- SW wr.
- HW wr.
- Comm. network

Ahmed A. Jerraya
Hardware/Software Communication Design

- Design of HW and SW communication layers to model abstract interaction between tasks/modules and with external world

Micro-architecture (e.g. Low-level C/VHDL)

Macro-architecture (e.g. SystemC)

#include <port.h>

void Task_A()
{
  …
  value: = FIFO_read (& port_Ain)
  HS_write (& port_Aout, value) … }

void Task_B()
{
  …
  value: = HS_read (& port_Bin) …
  FIFO_write (& port_Bout, value) …}

#include <port.h>

void Task_A()
{
  …
  IN_FIFO_B_16_INT (0xFC004) …
}

void Task_B()
{
  …
  Out_HS_B_24_INT (0xBA0F8, Value) …
}
Outline

1. Multiprocessors SoC (MP SoC)
   1.1. Multiprocessor SoC
   1.2. Multiprocessor SoC design
   1.3. This Course

2. Specification and validation of electronic systems
   2.1. Basic concepts
   2.2. Specification languages
   2.3. Heterogeneous systems modeling and validation

3. COLIF: A Design Model for MP SoCs
   3.1. COLIF: the Meta-model and the external syntax
   3.2. Mixed and multilevel model execution

4. A VDSL design example
   4.1. The application
   4.2. The design process

5. Summary
**Application-Specific Multi-Processor SoC**

*Summer school*

sponsored by IEEE Circuits and Systems Society and EDAA

<table>
<thead>
<tr>
<th>Monday 9th July</th>
<th>Tuesday 10th July</th>
<th>Wednesday 11th July</th>
<th>Thursday 12th July</th>
<th>Friday 13th July</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Welcome</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Specification and Validation for Heterogeneous Multiprocessor SoC</strong></td>
<td>Building Systems on a Chip with TriMedia technology</td>
<td>From a distributed embedded RTOS to a pragmatic framework for multi-core SoC</td>
<td>Challenges in Network Processor Architectures and Embedded S/W Tools</td>
<td>Communication architectures for deep-submicron VLSI Systems</td>
</tr>
<tr>
<td>A. Jerraya, TIMA</td>
<td>K. Vissers, TriMedia Technologies</td>
<td>E. Verhulst, Eonic Solutions</td>
<td>P. Paulin, STMicroelectronics</td>
<td>J. van Meerbergen, Philips</td>
</tr>
<tr>
<td><strong>Platform based design of embedded systems-on-chip</strong></td>
<td>From Applications to Multi-Processor DSP Architectures</td>
<td>Task-level run-time scheduling approach for dynamic multi-media systems.</td>
<td>F. Karim, STMicroelectronics</td>
<td>System Architectures: Hardware or Software dominant? A Case Study: xDSL modems</td>
</tr>
<tr>
<td><strong>Real-Time Operating Systems: Principles and a Case Study</strong></td>
<td>Energy-efficient design and manage-ment of SoCs</td>
<td>Modeling real-time systems</td>
<td>Adaptive EPIC Processors and Compilation Techniques</td>
<td></td>
</tr>
<tr>
<td><strong>RTOS for Embedded Systems and SoC</strong></td>
<td>The Architecture of Multiprocessor Systems on a Chip</td>
<td>Multiprocessor SoCs for Video Processing</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Real-Time Inter-Processor Synchronization Algorithms</strong></td>
<td>SOC Multiprocessor Architecture and Modeling</td>
<td>Configuring the Jazz VLIW-DSP Core for Application Specific Requirements</td>
<td></td>
<td>System on Chip: Embedded Test Strategies</td>
</tr>
<tr>
<td><strong>Architectural challenges and opportunities for systems on a chip</strong></td>
<td>Architectural challenges and opportunities for systems on a chip</td>
<td>Static scheduling for embedded systems</td>
<td></td>
<td>Architecture and Implementation of Application-Specific Multi-processor SOCs for Digital TV (DTV) and Media-Processing Applications, S. Dutta, Philips</td>
</tr>
</tbody>
</table>

Ahmed A. Jerraya

MP-SoC Summer School, July 2001, Aix-les-Bains, France - 11
Outline

1. Multiprocessors SoC (MP SoC)
   1.1. Multiprocessor SoC
   1.2. Multiprocessor SoC design
   1.3. This Course

2. Specification and validation of electronic systems
   2.1. Basic concepts
   2.2. Specification languages
   2.3. Heterogeneous systems modeling and validation

3. COLIF: A Design Model for MP SoCs
   3.1. COLIF: the Meta-model and the external syntax
   3.2. Mixed and multilevel model execution

4. A VDSL design example
   4.1. The application
   4.2. The design process

5. Summary
Electronic System Specification

- Basic Model: a set of hierarchically interconnected modules representing an abstract architecture

- Basic concepts:
  - Module
  - Interface to Port
  - Operation on ports
  - Content to Behavior
  - Behavior to Instances
  - Communication channels to Media, Behavior, Data unit

Different abstraction levels for behavior and communication
## Abstraction levels in behavior

<table>
<thead>
<tr>
<th>Abstraction level</th>
<th>Timing Unit</th>
<th>Typical language</th>
<th>Typical model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Untimed</td>
<td>Partial order</td>
<td>CSP, SDL</td>
<td>Communicating processes</td>
</tr>
<tr>
<td>Superstate</td>
<td>Computation/ control step</td>
<td>VHDL, SystemC, Esterel</td>
<td>Synchronous scheduling</td>
</tr>
<tr>
<td>RTL</td>
<td>C/K cycle</td>
<td>VHDL, SystemC, Verilog</td>
<td>Interconnected FSMs</td>
</tr>
<tr>
<td>Physical level</td>
<td>Physical Time (Delays)</td>
<td>VHDL/Verilog</td>
<td>Gates</td>
</tr>
</tbody>
</table>
## Abstraction levels in communication

<table>
<thead>
<tr>
<th>Abstraction level</th>
<th>Media</th>
<th>Typical Model</th>
<th>Typical communication primitive</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Client server</strong></td>
<td>Abstract Network</td>
<td>CORBA / UML</td>
<td>Print (file, network)</td>
</tr>
<tr>
<td><strong>Functional</strong></td>
<td>Active Channel</td>
<td>SDL</td>
<td>Send (file, disk)</td>
</tr>
<tr>
<td><strong>Macro architecture</strong></td>
<td>Logical connection</td>
<td>VHDL, SystemC</td>
<td>Write (Data, Port)</td>
</tr>
<tr>
<td><strong>RTL (Micro architecture)</strong></td>
<td>Physical connection</td>
<td>VHDL, SystemC, Verilog</td>
<td>Set (Value, Port)</td>
</tr>
</tbody>
</table>

- **Typical communication primitive**
  - Set (Value, Port)
  - Wait (clock)
  - Wait until $x = y$
  - Print (file, network)
  - Send (file, disk)
## Modeling Concepts through the Abstraction Level

<table>
<thead>
<tr>
<th>Abstraction level</th>
<th>C-S level</th>
<th>Functional</th>
<th>Macro-Architecture</th>
<th>RTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object</td>
<td>Module</td>
<td>Module</td>
<td>Module</td>
<td>Module</td>
</tr>
<tr>
<td>Interface</td>
<td>Port</td>
<td>Network Access (SAP)</td>
<td>Channel access</td>
<td>Logical port</td>
</tr>
<tr>
<td>Operation</td>
<td>Operation</td>
<td>Service request</td>
<td>Send/Receive to identified process</td>
<td>Read/Write Data</td>
</tr>
<tr>
<td>Behavior</td>
<td>Behavior</td>
<td>Concurrent Objects</td>
<td>Partially Ordered Transactions</td>
<td>Computation/ control steps</td>
</tr>
<tr>
<td>Instance</td>
<td>Instance</td>
<td>Instance</td>
<td>Instance</td>
<td>Instance</td>
</tr>
<tr>
<td>Content</td>
<td>Communication Channel</td>
<td>Media</td>
<td>Abstract Network</td>
<td>Active Channel</td>
</tr>
<tr>
<td></td>
<td>- Behavior</td>
<td>- Routing</td>
<td>- Protocol conversion</td>
<td>- Driver-level protocol</td>
</tr>
<tr>
<td></td>
<td>- DATA</td>
<td>- Request</td>
<td>- Generic data transmission</td>
<td>- Fixed data type</td>
</tr>
</tbody>
</table>
Register transfer level

```
port <= val
wait(clock)
val <= port
```

```
port <= val
wait(clock)
val <= port
```
Driver level

write(data,port)

wait until x=y
read(d,port)

Logic interconnections
Message level

send(data,f3)  get(data)

F1  F2  F3  F4
Service level

\[
\text{type } xmt = xmt_f1 / \ldots / xmt_f3 / \ldots
\]

request \((xmt_f3, \text{data})\)

\[
\text{request (xmt}_f3, \text{data)}
\]

\[
\text{service (d)}
\]

\[
\text{ORB}
\]

Ahmed A. Jerraya
Outline

1. Multiprocessors SoC (MP SoC)
   1.1. Multiprocessor SoC
   1.2. Multiprocessor SoC design
   1.3. This Course

2. Specification and validation of electronic systems
   2.1. Basic concepts
   2.2. Specification languages
   2.3. Heterogeneous systems modeling and validation

3. COLIF: A Design Model for MP SoCs
   3.1. COLIF: the Meta-model and the external syntax
   3.2. Mixed and multilevel model execution

4. A VDSL design example
   4.1. The application
   4.2. The design process

5. Summary
Welcome to the Jungle of specification languages

- Which purpose: Modeling, Simulation, Synthesis
- Which abstraction: Communication, Behavior
- Which computation model:
  - Task: Predictable execution time, Run to completion, Interactive with internal state (VHDL, SDL)
  - Control model: State-based or control-driven (SDL, ESTEREL) Data-driven (LUSTRE, COSSAP, Matlab)
  - Concurrency (Composition): Pure Synchronous (StateCharts, Esterel, Lustre), Synchronous Clocked (SystemC, VHDL, StateCharts), Asynchronous (SDL), Single thread (C, C++)
  - Communication behavior: Zero Delay (StateChart, VHDL), Non-zero delay (SDL, Corba)

- Which community
Description Languages: Different Communities

- **Methods**: UML, OMT, SART, ...
- **ADL**: WRITE, RAPID, IDL ...
- **RT systems**:
  - Synchronous: LUSTRE, ESTEREL, StateCharts, ...
  - Asynchronous: SDL, Objectime, ...
- **System modeling**:
  - Universal: Matlab, Matrixx, ...
  - DSP: Cossap, SPW
- **Semiconductor**:
  - HDL: VHDL, Verilog, e,
  - C/C++ extensions: SystemC, SpecC, ...

None of the existing languages covers all system-level concepts at all abstraction levels
## Missing Concepts in Specification Languages

<table>
<thead>
<tr>
<th>Language Concept</th>
<th>SystemC .9 HDL</th>
<th>SC 1.0 Cossap, SPW</th>
<th>Real Time Synchronous</th>
<th>Real Time, Asynchronous UML, ADLs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Module</strong></td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port</td>
<td>Abstract ports</td>
<td>Generic data port</td>
<td>Active ports</td>
<td>Physical ports</td>
</tr>
<tr>
<td>Operation</td>
<td>Protocol</td>
<td>Data Generic</td>
<td>Non fixed delay</td>
<td>- Regular data stream operations</td>
</tr>
<tr>
<td></td>
<td>independent</td>
<td>Operation</td>
<td>operation</td>
<td>- Detailed protocols</td>
</tr>
<tr>
<td>Process Task</td>
<td>Synchronized</td>
<td>Data dependent</td>
<td>Non regular cycle</td>
<td>- Cycle-true model</td>
</tr>
<tr>
<td></td>
<td>multiple task</td>
<td>computation</td>
<td>free computation</td>
<td>- Implementation related operation</td>
</tr>
<tr>
<td>Instance</td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td><strong>Content</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communication</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Media</td>
<td>Abstract</td>
<td>Active</td>
<td>Hierarchical &amp;</td>
<td>Physical signal buses</td>
</tr>
<tr>
<td></td>
<td>Channels</td>
<td>Channels</td>
<td>distributed</td>
<td></td>
</tr>
<tr>
<td>Behavior</td>
<td>Non transmission behavior</td>
<td>Protocol Conversion</td>
<td>Other than broadcasting</td>
<td>Regular fixed data streams</td>
</tr>
<tr>
<td>DATA</td>
<td>Generic data types</td>
<td>Generic data types</td>
<td>Generic data types</td>
<td>Fixed data representation</td>
</tr>
</tbody>
</table>
System-level specification trends

- extend an existing language: N2C, SpecC, SystemC
  + fast executable model for verification
  - task/comm. model is bound to the simulation model

- create a new language: Rosetta
  + formal verification
  - inflexible communication models

- Semantic models: OVI-SRM, VSIA
  + language neutral approach
  - peer-to-peer comm. and run-to-completion tasks
  - complex refinement process

- “Meta-models”: GSRC, COLIF “abstract syntax”
  - syntax and semantic independent
Outline

1. Multiprocessors SoC (MP SoC)
   1.1. Multiprocessor SoC
   1.2. Multiprocessor SoC design
   1.3. This Course

2. Specification and validation of electronic systems
   2.1. Basic concepts
   2.2. Specification languages
   2.3. Heterogeneous systems modeling and validation

3. COLIF: A Design Model for MP SoCs
   3.1. COLIF: the Meta-model and the external syntax
   3.2. Mixed and multilevel model execution

4. A VDSL design example
   4.1. The application
   4.2. The design process

5. Summary
Heterogeneous Models

- Components execution models
  - Hardware: Gates, RTL, Functional, Client Server,
  - Software: CA/ISS, IA/ISS, Native Bus Functional CA, Native with OS, Functional, Client Server
  - Functional Architecture, before partitioning
  - Environment, Multiple domains (Mechanical, Optical, …), Multiple computation models

- Full System Execution model
  - Different concepts ➔ co-simulation
  - Multiple computation models
    - Separation between computation and communication
    - Single or several languages
Example: C - VHDL Co-simulation

```c
main() {
    int a,b;
    exin(a);
    b=f(a);
    exout(b);
}
```

Each simulator communicates via a fragment of shared memory (or queues or ...)

The router secures the coherence between several memories via sockets.

VHDL entity

ROUTER (sockets, IPC, lightweight, ...)

Shared Memory

Interconnect

Shared memory

Ahmed A. Jerraya
Heterogeneous System approaches

- **SystemC**
  - Module or channel wrapper usage
  - Modules/channels with multiple abstraction levels
  - SystemC provides primitives (e.g. interface concept) and ideas (e.g. BCASH) to design wrappers.
- **VADel [Cesario]** Module wrapper generation
- **Bus Functional Model [Séméria et. al., ASPDAC2001]**
- **Modeling community: ADL, RAPID, CORBA based**
Summary of system specifications

- Specification languages
  - Few concepts
  - Many from different communities
  - Specific to application domain

- Key issues for MP SoC
  - Specify interconnect for heterogeneous objects
  - Execution model for heterogeneous systems

- Requirements
  - Modular design, separation communication/computation
  - Multi and mixed level Co-simulation
Outline

1. Multiprocessors SoC (MP SoC)
   1.1. Multiprocessor SoC
   1.2. Multiprocessor SoC design
   1.3. This Course

2. Specification and validation of electronic systems
   2.1. Basic concepts
   2.2. Specification languages
   2.3. Heterogeneous systems modeling and validation

3. COLIF: A Design Model for MP SoCs
   3.1. COLIF: the Meta-model and the external syntax
   3.2. Mixed and multilevel model execution

4. A VDSL design example
   4.1. The application
   4.2. The design process

5. Summary
COLIF: A Design Model for MP SoCs
Heterogeneous Specification

- Basic model: a set of hierarchically interconnected modules representing an abstract architecture
- Basic concepts: Module, Interface (internal, external), Content
- Accommodate both HW-SW models and Co-simulation
- Execution model through automatic wrapper generation
- External Syntax: VADeL, a SystemC extension
- Internal Syntax: XML
COLIF: An Internal Model for Mixed & Multi-level Refinement

- External syntax: e.g. SystemC, VADeL, ...
- COLIF
  - Hierarchical interconnect model
  - Design parameters

Simulation Model

Communication refinement

Automatic code generation

Implementation RTL: Synthesizable HW/Executable SW
Target Architecture

- Dissociate Components and communication network
  - HW: Requires communication co-processors (wrapper, bridge)
  - SW: Multiple Application specific RTOS

- Enable both Automatic Design and Co-simulation
- Cover both Hardware and Software
- Systematic Assembling of heterogeneous existing blocs
Heterogeneous System Specification

- Basic model: a set of hierarchically interconnected modules
- Basic concepts:
  - Virtual Module
  - Interface, set of ports (internal, external)
  - Content (Tasks / Instances + Communication channels)

Not executable due to the difference btw. internal and external ports
Heterogeneous System Specification

- Basic model: a set of hierarchically interconnected modules
- Basic concepts:
  - Virtual Module
  - Interface, set of ports (internal, external)
  - Content (Tasks / Instances + Communication channels)

Executable/implementable by the generated wrappers
Heterogeneous System Specification

- Basic model: a set of hierarchically interconnected modules
- Basic concepts:
  - Virtual Module
    - Interface, set of ports (internal, external)
    - Content (Tasks / Instances + Communication channels)

Abstraction levels of communication
- Service: client-server, e.g. CORBA
- System: e.g. send/receive in SDL
- Macro Architecture: e.g. FIFO
- Micro Architecture (RT level): e.g. AMBA
VADeL: a SystemC extension for COLIF execution

- VADeL: Virtual Architecture Description Language

- Embedded processor in VADeL
  - Virtual Module annotated with implementation parameters
  - SystemC modules for each task or IP
  - Virtual Ports for Abstraction-level /Language/Protocol adaptation
  - Virtual nets to group related SystemC signals

Ahmed A. Jerraya
VADeL - Mixed level specification

```c
#include <systemc.h>
#include <vadel.h>
#include <Vport1.h>

...#include <token2.h>

SC_VCOMPONENT (VCtoken) {
    sc_out_rtl<bool> VC_data1;
    sc_out_rtl<bool> req1;
    sc_out_rtl<bool> ack1;
    ...
    Vport1* Vp1;
    Vport2* Vp2;
    ...
    token* token2;
    SC_VCCTOR (VCtoken) {
        token1 = new token("token2");
        Vp1 = new Vport1("Vp1", &token2);
        Vp2 = new Vport2("Vp2", &token2);
        ...
    }
};
```

```c
#include <systemc.h>
#include <vadel.h>

SC_VPORT (Vport1) {
    sc_out_arch<bool>* VC_data;
    sc_in_rtl<bool>* ack;
    sc_out_rtl<bool>* data;
    sc_out_rtl<bool>* req;
    ...
    SC_VCCTOR (Vctoken, M) {
        VC_data = &M->data;
        data = &VC_data1;
        req = &req1;
        ...
    }
};
```

```c
#include <systemc.h>
#include <vadel.h>

SC_VCHANNEL (Vchannel1) {
    sc_ch_rtl<sc_lv<128)> ch1;
    sc_signal<bool> ch2;
    sc_signal<bool> ch3;
    SC_VCCTOR (VCtoken) {
    }
};
```
Executable models generation

System Specification

Co-simulation Library

- Module Adapt
- Channel Adapt.

Simulation Model

- Wrapper
- Wrapper

Colif

Simulation model generation

Wrapper for Co simulation

A

B

C

Comm Net (SystemC)
Outline

1. Multiprocessors SoC (MP SoC)
   1.1. Multiprocessor SoC
   1.2. Multiprocessor SoC design
   1.3. This Course

2. Specification and validation of electronic systems
   2.1. Basic concepts
   2.2. Specification languages
   2.3. Heterogeneous systems modeling and validation

3. COLIF: A Design Model for MP SoCs
   3.1. COLIF: the Meta-model and the external syntax
   3.2. Mixed and multilevel model execution

4. A VDSL design example
   4.1. The application
   4.2. The design process

5. Summary
Demo: VDSL Design Through Systematic HW/SW Assembly of IP

VDSL Application

Modified architecture
System Specification & co-simulation

- SystemC specification using virtual component concept
- Abstract communication
- Parameter annotations within specification
- Early co-simulation

config = *P1.Get();
TCS.Put(retrieve_init);
etat = *TXIN.Get();
while (etat != v_init)
etat = *TXIN.Get();
...
HW-SW Wrapper Generation Steps

1. System-Level Co-simulation
2. SW wrapper generation
3. HW Wrapper generation
4. RTL Generation
5. HW-SW RTL architecture
6. Validation
7. HW lib
8. SW lib
9. System Spec
10. SW Wrapper
11. HW Wrapper
12. RTL Co-simulation
13. Co-simulation
14. Co-simulation
15. Validation
16. HW-SW architecture
Interne: voc_available
parameatre value
SystemCType sc_in
SystemCDataType bool
SystemCDataBitWidth 1
SystemCPortBitWidth 0

Interne: en_voc_byte
parameatre value
SystemCType sc_out
SystemCDataType bool
SystemCDataBitWidth 1
SystemCPortBitWidth 0

Externe: putwocworld
SystemCType va_out_mac_pipe
SystemCDataType long int
SystemCDataBitWidth 32
SystemCPortBitWidth 0
SoftPortType WaitRegister
DATA_BIT_WIDTH 32
ADDRESS_DATA 0x000F0070
MASK_GET 2
MASK_PUT 1
C_DATA_TYPE long int
DATA_BIT_WIDTH 32
CHAN_PRIO 23
IT_Number 8
IT_Level 1

Interne: voc_byte
parameatre value
SystemCType sc_in
SystemCDataType bit8
SystemCDataBitWidth 8
SystemCPortBitWidth 0
Architecture generation through systematic HW-SW assembly

System Specification

SW library

Comm./Sys. Services

Device Drivers

HW library

APIs

send
recv
fifo
TS
wr
dr
SW Wrappergeneration

RTL Architecture

μP

DSP

Application

APIs
Comm./Sys. Services
Dev. Drivers

Processor

Proc. Adapter

CA

CA

HW Wrappergeneration

HW Wrappergeneration

Application

Comm./Sys. Services
Dev. Drivers

Proc. Adapter

CA

CA
HW-SW Wrapper Generation

ARM7 Core

M1

M2

ARM7 Bus Adpt.

Mem.

System Calls
ISR I/O SCHD D1 D2 D3 D4 D5

SW Wrapper

HW Wrapper

Env

clk rst

data_in

data_out

Env

M3

IP RTL
SystemCType va_in_mac_event
SystemCDataType bool
SystemCDataBitWidth 1
SystemCPortBitWidth 0
SoftPortType Interrupt
IT_NUMBER 1

SystemCType va_out_mac_pipe
SystemCDataType long int
SystemCDataBitWidth 32
SystemCPortBitWidth 0
SoftPortType GuardedRegister
ADDRESS_DATA 0x000F0008
ADDRESS_STATE 0x000F0004
MASK_GET 2
MASK_PUT 1
IT_LEVEL 1
IT_NUMBER 3
C_DATA_TYPE long int
DATA_BIT_WIDTH 32
CHAN_PRIO 3

SystemCType va_out_mac_pipe
SystemCDataType long int
SystemCDataBitWidth 32
SystemCPortBitWidth 0
SoftPortType GuardedRegister
ADDRESS_DATA 0x000F0010
ADDRESS_STATE 0x000F0014
MASK_GET 2
MASK_PUT 1
IT_LEVEL 1
IT_NUMBER 4
C_DATA_TYPE long int
DATA_BIT_WIDTH 32
CHAN_PRIO 1
VDSSL design summary

- **System specification/Validation**
  - VADeL model: Lines/Nets 150/21
  - Simulation model Lines/Nets 393/60
  - Simulation model generation time: 90s

- **Architecture design**
  - HW Wrapper generation
    - VM1: 3284 Gates, area 818 µm², maxF ck 168 MHz
    - VM2: 3795 Gates, area , maxF ck 162 MHz
    - Latence : Write 2 clk-cycles, Read 6 clk-cycles
    - VHDL code 2168 lines of VHDL RTL
  - SW Wrapper generation
    - VM1 : 1249 lines (281 ASM), Mem Code/data 1484/450 bytes
    - VM1 : 2153 lines (281 ASM), Mem Code/data 2624/1020 Bytes
Conclusions

- Multiprocessor SoC: already a reality and main future driver
- System Specifications: Few Concepts, Too many languages
- Key issues for MP SoC specification/validation:
  - On chip communication network abstraction,
  - Execution and refinement for multi and mixed level model
- COLIF: A Design Model for MP SoCs Specification
  - Abstract wrappers to connect heterogeneous components
  - Architecture generation through systematic HW-SW assembly
Reading about system specification


Reading about system co-simulation


Reading about COLIF and the Architecture design environment


