Integrating a Virtual Platform Framework for Smart Devices

V. Guarnieri\textsuperscript{1}, F. Stefanni\textsuperscript{1}, F. Fummi\textsuperscript{1}
M. Grosso\textsuperscript{2}, D. Lena\textsuperscript{2}
A. Ciccazzo\textsuperscript{3}, G. Gangemi\textsuperscript{3}, S. Rinaudo\textsuperscript{3}
Outline

• Introduction
• HIFSuite
• Testcases
• Abstraction results
• Conclusion
What is a smart system?

• Miniaturized self-sufficient device
  – incorporating functions of sensing, actuation and control
  – able to describe and analyze a situation
  – able to decide according to available data
  – Energy-autonomous and ubiquitously connected
Issues in smart system design (I)

• Extremely heterogeneous representations for the various components → co-simulation
  – Poor simulation performance
  – Manual translation required
Issues in smart system design (II)

- Too low design level to provide a global view of the entire system
  - Typical design levels: physical/device, structural
  - Global view should be at least functional or transactional
  - Manual abstraction required
Issues in smart system design (III)

- Analog and mixed-signal components
  - Difficult to integrate into higher-level systems, but ...
  - Absolutely required in smart systems
  - Multi-dimensional circuit sizing and verification problems
    → circuit analysis and automated optimization algorithms required
Possible solutions

• Heterogeneity of smart systems → homogeneous models
• Co-simulation techniques → simulation techniques
• Enhancing reuse through abstraction and systems aggregators
• Concurrent simulation of functional and extra-functional properties
Proposed solution

• Integration of heterogeneous components into a homogeneous virtual platform
  – Components written in different languages and belonging to different domains

• Optimization of the homogeneous description for simulation
  – Towards a full C++ model

• Design of each component with the most suited tool
  – Homogeneous platform to evaluate functional and extra-functional properties
HIFSuite

PARSERS → HIF API → GENERATORS

MANIPULATION TOOLS

VHDL
Verilog
Verilog AMS
IP-XACT

SystemC RTL
SystemC TLM
SystemC AMS
C++
IP-XACT

© Accellera Systems Initiative
Step 1

- **IPs HDL Descriptions**
- **Parsers**
- **HIFSuite**
- **hif2ipxact**
- **IP-XACT IPs descriptions**
- **IP-XACT Visual editor**
- **ipxact2hif**
- **Communication Information**
- **Platform IP-XACT model**
- **hif2sc**
- **Platform Top-Level – Bindings only (SystemC-RTL)**
Step 2

IPs HDL Descriptions

Platform IP-Xact model

HIFSuite

Components Library

Functionalities

Components Interfaces

Communication Information

Functional Implementation

Homogeneous Structural Implementation (SystemC-RTL + AMS)
Step 3

IPs HDL Descriptions

Platform IP-Xact model

Components Library
- Functionalities
  - ipxact2hif

Components Interfaces

Communication Information

Optimized Functionalities

Functional Implementation
- ddt
- a2t
- hif2sc

Functional Implementation (C++)
Testcases

• Two testcases provided by ST
  – Modular sensor node: monitoring of human gestures or movement
  – Enhanced LED driver engine: implementation of smart lightning modules

• Working prototypes, not marketed products

• Typical EDA challenges in
  – System-level design
  – MEMS-design
  – Analog-mixed signal-design
Modular sensor node (I)
Modular sensor node (II)

• Analog design domain
  – HIFSuite: VerilogA $\rightarrow$ SystemC-AMS
    • SystemC-AMS description to be used in system-level simulation
    • Avoiding co-simulation

• Digital design domain
  – HIFSuite: RTL $\rightarrow$ SystemC TLM or C++
    • Mitigating bottlenecks in system-level simulation due to complexity

• System-level model advantages
  – Performance evaluation under realistic workloads
Enhanced LED driver engine (I)

System (transactional)
- SystemC-AMS
- C++

Subsystem (structural level)
- SystemC-AMS
- C++

Component (device level)
- Mentor (Circuit simulator)
- Verilog A
- WiCked & RSM
- netlist (BCD)
- power analog

LED driver

HIFSuite
- SystemVue (DF Simulator)

Microcontroller
- SystemVue
- SystemC-AMS
- C++

Sensors
- SCNSL

Network interface
- data flow

Cadence (Circuit simulator)
- Verilog A
- WiCked & RSM
- netlist (CMOS)

SystemC
- C++

MATLAB
- m file

IP-XACT
- Verilog
- VHDL

WiCked
- RSM

Verilog A
- WiCked
- RSM

Verilog
- VHDL

Network interface
- data flow

Accellera Systems Initiative
Enhanced LED driver engine (II)

• Full-system simulation advantages
  – Better understanding of sub-systems interactions
  – Deep investigation of the microcontroller capabilities and peripherals usage
    • Early addressing performance issues and firmware optimization
  – Optimization of power modes
  – Correct interoperability between sub-systems
  – Anticipation of issues during design phase → reduction of development costs
Abstraction

• Process scheduler
  – Classical HDL process scheduler $\rightarrow$ more performing process scheduler

• Data types
  – Original HDL data types $\rightarrow$ C++ built-in data types

• Interface
  – RTL interface $\rightarrow$ TLM or C++ interface
## Abstraction results (I)

<table>
<thead>
<tr>
<th>Design</th>
<th>SystemC RTL (s)</th>
<th>ModelSim (s)</th>
<th>Abstracted C++ (s)</th>
<th>Speedup vs SystemC RTL (x)</th>
<th>Speedup vs ModelSim (x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camellia</td>
<td>26,974.8</td>
<td>1,074.7</td>
<td>3.4</td>
<td>7,933.8</td>
<td>316.1</td>
</tr>
<tr>
<td>DES56</td>
<td>7,112.2</td>
<td>790.4</td>
<td>4.3</td>
<td>1654.0</td>
<td>183.8</td>
</tr>
<tr>
<td>AES</td>
<td>850.9</td>
<td>67.5</td>
<td>7.1</td>
<td>119.8</td>
<td>9.5</td>
</tr>
<tr>
<td>SHA256</td>
<td>4,682.5</td>
<td>152.4</td>
<td>3.4</td>
<td>1,377.2</td>
<td>44.8</td>
</tr>
<tr>
<td>SHA512</td>
<td>6,302.1</td>
<td>175.6</td>
<td>5.0</td>
<td>1,260.4</td>
<td>35.1</td>
</tr>
<tr>
<td>XTEA</td>
<td>975.2</td>
<td>170.9</td>
<td>3.4</td>
<td>286.8</td>
<td>50.3</td>
</tr>
</tbody>
</table>
Abstraction results (II)

![Bar chart showing speedup vs SystemC RTL for Camellia, DES56, AES, SHA256, SHA512, XTEA.]
Abstraction results (III)

![Graph showing speedup vs ModelSim (x) for different algorithms: Camellia, DES56, AES, SHA256, SHA512, XTEA. The x-axis represents different algorithms, and the y-axis shows speedup. Camellia has the highest speedup, followed by DES56, AES, SHA256, SHA512, and XTEA.]
Conclusions

• HIFSuite assists designers of smart devices
  – Translation of heterogeneous descriptions of components into homogeneous SystemC description
    • No need for co-simulation
  – Abstraction to C++ to enhance simulation performance
  – Reuse of existing IPs and integration into virtual platforms
    • Design space exploration
    • Design validation
    • Performance evaluation
  – Reduction of time to market
Thanks for your attention!

Questions?