SystemC Module Observability for Observer-Controller Architectures

Johannes Zeppenfeld
Andreas Bernauer
Abdelmajid Bouajila

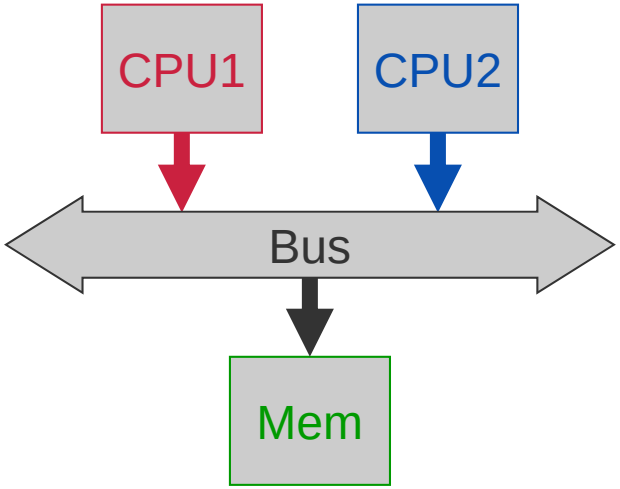
and other ASoC team members

Lehrstuhl für Integrierte Systeme
Theresienstr. 90
80333 München
www.lis.ei.tum.de

Outline

- Overview of ASoCsim
- Observer-Controller Architecture
- Queryable Module Parameters

ASoCsim Motivation



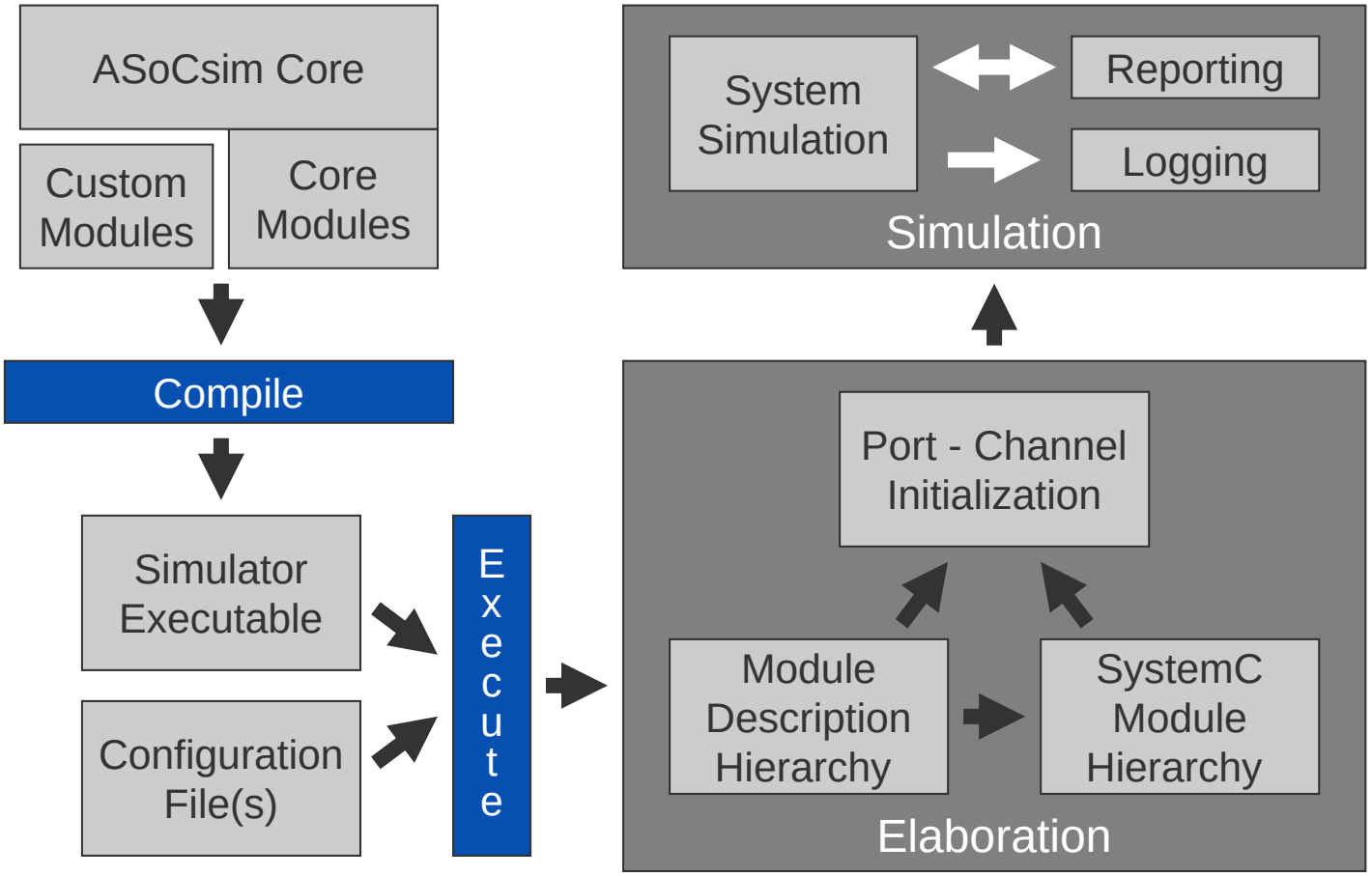

```
<ASoCsim name="MPSoC" version="primitive-1.0">

    <Module name="CPU1" type="Leon3">
        <param name="frequency" value="100MHz" />
        <param name="code">
            <param name="file" value="cpu.tr" type="path" />
        </param>
        <port name="bus" target=".Bus" />
    </Module>

    <Module name="CPU2" type="Leon3" />

    <Module name="Mem" type="SRAM" />

    <Module name="Bus" type="AMBA">
        <port name="mem" target=".Mem" />
    </Module>

</ASoCsim>
```

Fundamental Simulator Concepts

- Library of system components (modules) as building blocks
  - Selection, parameterization and interconnection of building blocks through configuration file (e.g. XML) during elaboration
  - Construction set principle
  - Only potential building blocks are known at compile time (CPU, Memory, Bus)
- Addition of custom modules without modification of library code
  - Any functionality not inherently provided by libasoc can easily be added by the user

- Transaction Level Modeling
  - Interested in the communication and collaboration between system components
  - Functionality internal to system components modeled as abstractly as possible

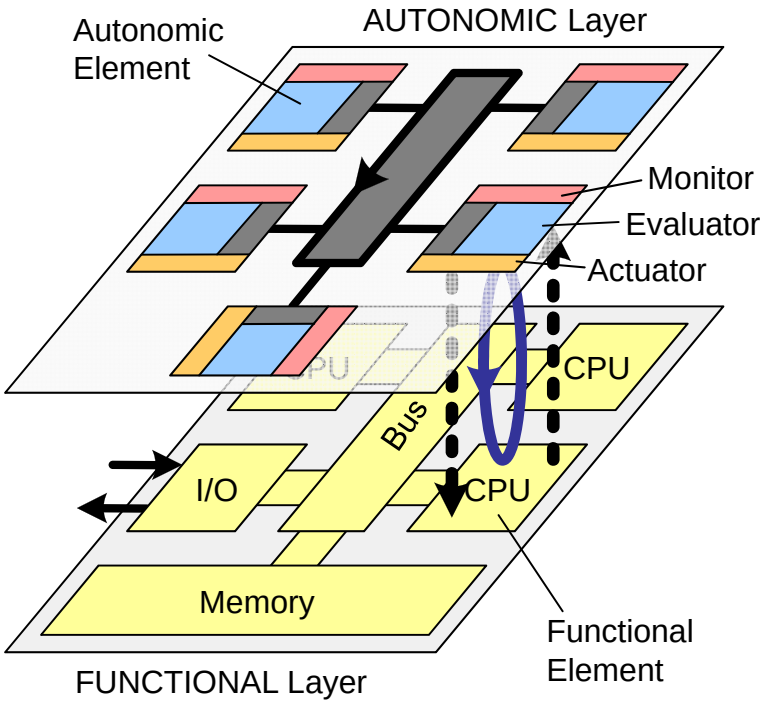
ASoCsim Workflow



Observer-Controller Architecture

- System is logically split into two layers
  - Functional layer containing traditional IP Modules (FEs)
  - Autonomic layer extending IP Modules with self-x properties
- Monitors and actuators require access to FE's parameters
- Adding or modifying an AE should not require changes in the FE

- **Requires transparent and configurable mechanism for accessing FE parameters**



Queryable: Accessing Members by String Identifier

Similar to Key-Value Coding in Objective-C

```
class Leon3 : public Module
{
    … Various Public and Private Methods …

    QUERYABLE( double, m_util, "utilization" );
    MUTABLE( frequency, m_freq, "frequency" ) { … }
};

void AE::observe_control( Module &mod )
{
    double util = query( mod, "utilization" );

    accessor<frequency> freq
                    = query( mod, "frequency" );
    freq->setFrequency( 100, frequency::MHz );
};
```


Queryable: Accessing Members by String Identifier

- Transparent:
  - No knowledge of monitor or actuator required by FE
  - Only knowledge of queried id and type needed by monitor
  - Access of unknown type possible with accessor<void>
- Configurable:
  - String identifiers well suited for configuration files
- Safe:
  - Full type checking
  - Unambiguous access to any number of members of any type
- Fast:
  - Accessor initialization can be done once during elaboration
  - Single virtual function call per access through accessor

Conclusion

- ASoCsim: SystemC-based simulator using construction set principle
  - Library of modules that can be assembled during elaboration
- Queryables allow transparent and configurable access to module members
  - Separating functional interface from observer-controller interface
  - Avoiding massive multiple inheritance for abstract base class interfaces
  - Allowing unambiguous access to any queryable or mutable member
- Available under the GNU General Public License
- Build environments included for GCC Make and MSVC 8.0