This Publication's Reference:

[1] Andreas Raabe, Andreas Nett, and Andreas Niers. A Refinement Case-Study of a Dynamically Reconfigurable Intersection Test Hardware. In *ReCoSoc'08*, July 2008.

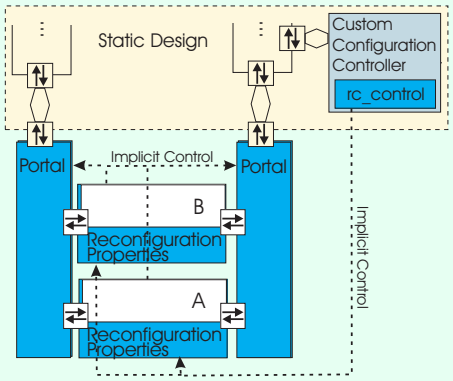
ReChannel - Simulating Reconfiguration in SystemC

Andreas Raabe, Armin Felke, Andreas Niers
Dept. of Computer Science, University of Bonn {raabe,felke,niers}@cs.uni-bonn.de

The presented ReChannel library provides a **language extension for SystemC for describing and simulating dynamically reconfigurable systems** on all levels of abstraction. It provides maximum freedom of description of reconfiguration behaviour and its control, while featuring simulation of run-time configuration, removal and exchange of custom modules as well as third-party IP-cores during the complete architecture refinement process. It effectively **combines IP reuse and high-level modelling with reconfiguration**. When designing on RT-level the resulting hardware description can easily be synthesised with standard synthesis tools.

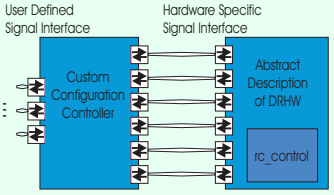
The ReChannel Approach

Configuration Control

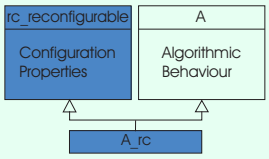


The configuration controller (CCC) is modelled as a normal (custom) module. And thus the CCC ...

- ... needs no special treatment, and thus no irregularities are introduced into the language.
- ... can be described on all levels of abstraction.
- ... can be implemented as a software or hardware module.
- ... can even be described in a synthesisable fashion.

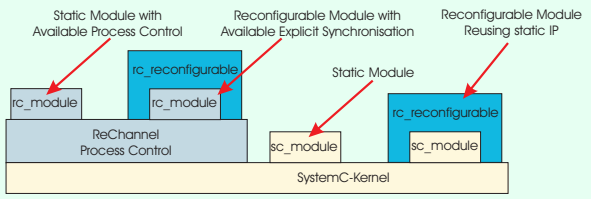


IP Reuse



- A reconfigurable module can be derived from a static one and `rc_reconfigurable`.
- Its implementation is not necessarily known to the designer (closed source).
- The module can be extended with reconfiguration properties after derivation, such as (re-)configuration delay, synchronisation, etc.

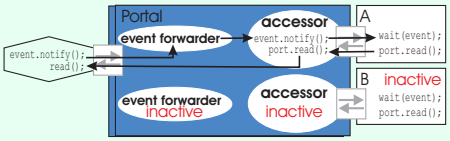
Process Control Layer



ReChannel layers its process control functionality on top of the SystemC kernel. This enables ...

- ... killing and resetting of reconfigurable objects and static components that were designed using ReChannel primitives.
- ... co-simulation with arbitrary SystemC modules without any further effort.
- ... compliance to the IEEE SystemC language standard.
- ... extension of standard SystemC modules with process control after derivation.

The Switch Concept

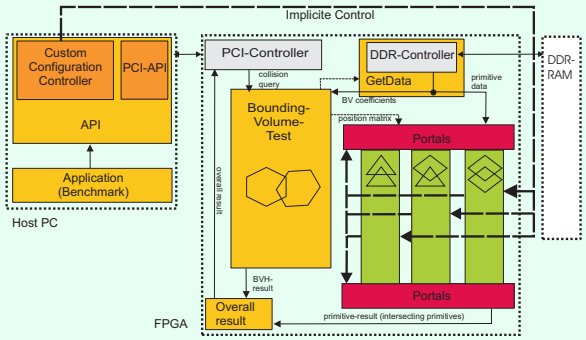


ReChannel [2] uses the switch concept to decouple binding and communication. This allows to rechannel data streams and thus to simulate reconfiguration.

- Switches come in two flavours: As portals and exportals.
- Events are forwarded only from/to static modules and active reconfigurable modules.
- Only static modules and active reconfigurable modules may access channels.
- Only exports / interfaces of static modules and active reconfigurable modules may be accessed from outside.

Case Studies

CollisionChip



The CollisionChip project [1, 3, 4] provides an optimal framework to evaluate ReChannel's applicability. It consists of over 17,000 lines of SystemC code. The CollisionChip checks virtual 3D objects for intersection and was founded by the DFG (German Research Council).

- Different primitive tests are exchanged against each other during the system's runtime. The primitive intersection modules were treated like closed-source components and hence were not manipulated to render them reconfigurable.
- Using ReChannel it took a single developer only 2 days to introduce reconfiguration into the (RTL) design.
- Reconfiguration was used on all levels of abstraction featured by SystemC.
- The impact of reconfiguration on the system's performance was evaluated using ReChannel features.
- The RTL simulation's run-time increased only by about 11% compared to the static architecture.

References

[1] A. Raabe, B. Bartyzel, J. K. Anlauf, and G. Zachmann. Hardware Accelerated Collision Detection — An Architecture and Simulation Results. In *Design Automation and Test (DATE)*, pages 130–135, Munich, Germany, Mar.7–11 2005. IEEE Computer Society.

[2] A. Raabe, P. A. Hartmann, and J. K. Anlauf. Rechannel: Describing and Simulating Reconfigurable Hardware in SystemC. *ACM Transactions on Design Automation of Electronic Systems (accepted)*.

[3] A. Raabe, S. Hochgürtel, G. Zachmann, and J. K. Anlauf. Space-Efficient FPGA-Accelerated Collision Detection for Virtual Prototyping. In *Design Automation and Test (DATE)*, pages 206–211, Munich, Germany, 2006.

[4] A. Raabe and F. Zavelberg. Defying the Memory Bottleneck in Hardware Accelerated Collision Detection. In *WSCG '2008*, University of West Bohemia, Plzen, Czech Republic, 2008.