Transactor-based debugging of massively parallel processor array architectures

Markus Blocherer, Srinivas Boppu, Vahid Lari, Frank Hannig, Jürgen Teich
Hardware/Software Co-Design
University of Erlangen-Nuremberg

1st International Workshop on
Multicore Application Debugging (MAD 2013),

November 14-15, 2013

Germany

Agenda

Motivation

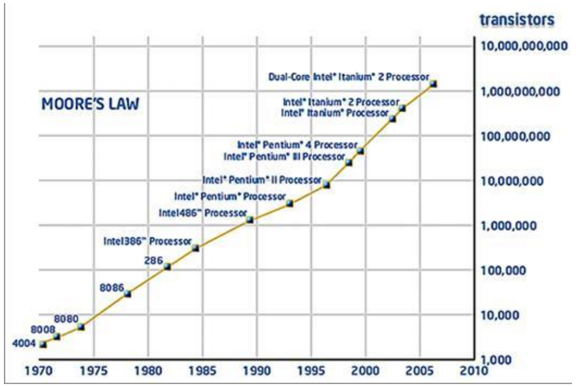
Invasive Computing

Hardware Debugging

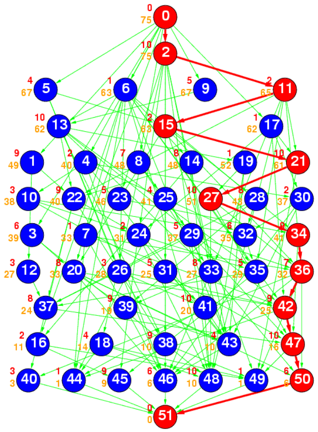
Transactor-based Prototyping

Conclusions

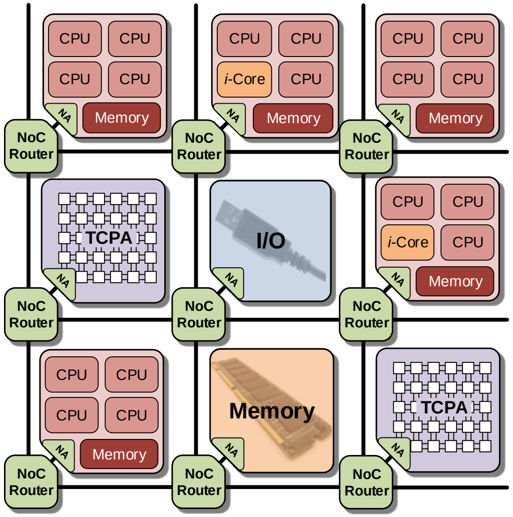
Motivation



Steady increase in the number of cores on a chip



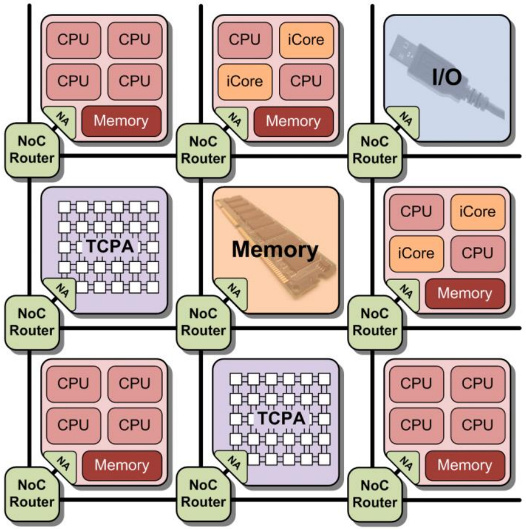
Steady increase in the application complexity



Customization and heterogeneity are the key success for future performance gains

Invasive Computing

Challenge: Simultaneous development of different architecture and software parts as well as their integration and validation

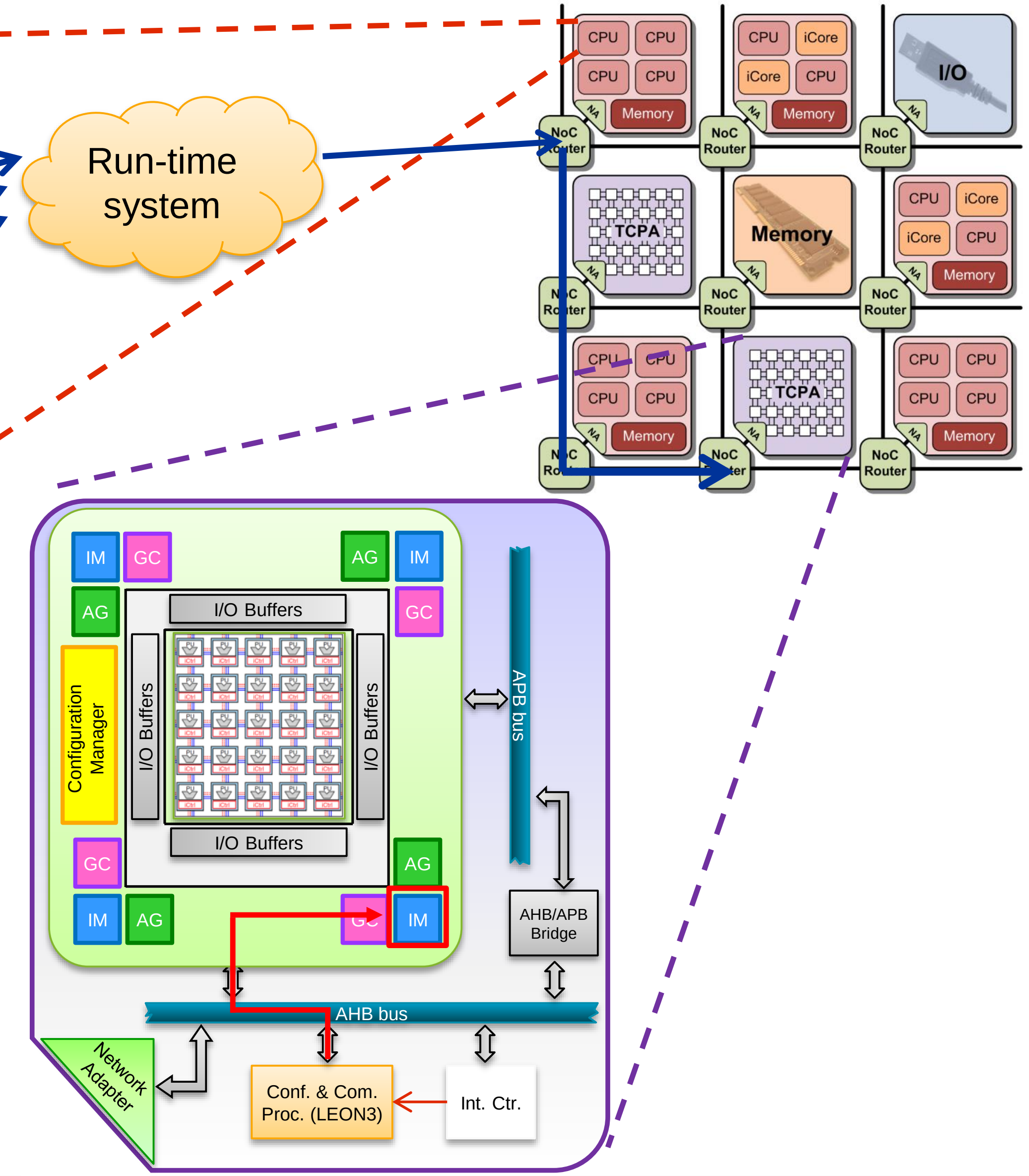


tiled architecture

- Architecture consists of different compute tiles
  - RISC CPU tiles
  - RISC CPUs with reconfigurable fabrics
  - Programmable accelerators (TCPA)

Invasion on TCPAs

```
 /* code to be executed sequentially*/
...
val constraints = new AND();
constraints.add(new TypeConstraint(PEType.TCPA));
constraints.add(new PEquantity(4));
constraints.add(new Layout(LIN));
val claim = Claim.invade( constraints );
val ilet = (id:IncarnationID) => {
  /* code to be executed in parallel */
  ...
};
claim.infect( ilet );
…
claim.retreat();
```

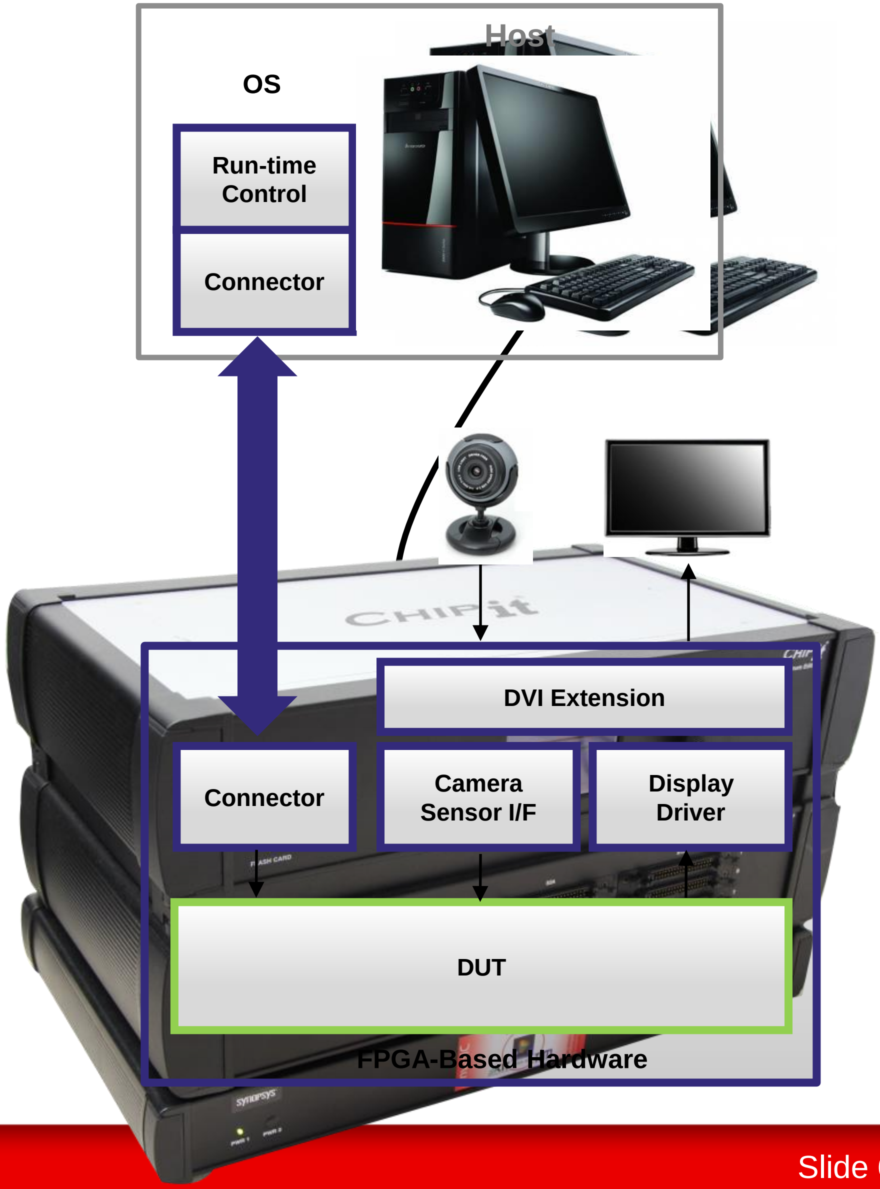


- Run-time system interaction with TCPAs
  - Resource requests and releases
  - Application configuration
  - Input/output data streams

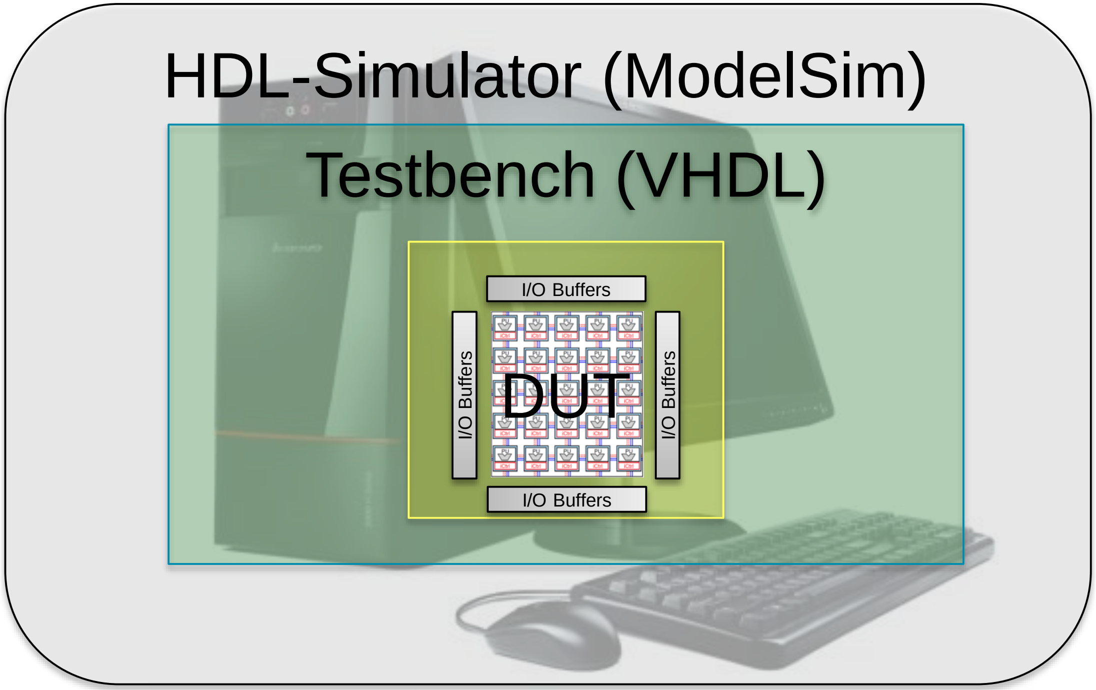
How do we prototype TCPAs with tight software/hardware interactions?

InvasIC Prototyping Platform

- Synopsys FPGA-based prototyping platform
  - Up to 12 million ASIC gates of capacity
  - Tools for multi-FPGA prototypes (Certify) and RTL debug (Identify)
  - UMRBus interface kit for host workstation
  - Transactor library for AMBA to support bus-protocol communication
  - Portable hardware



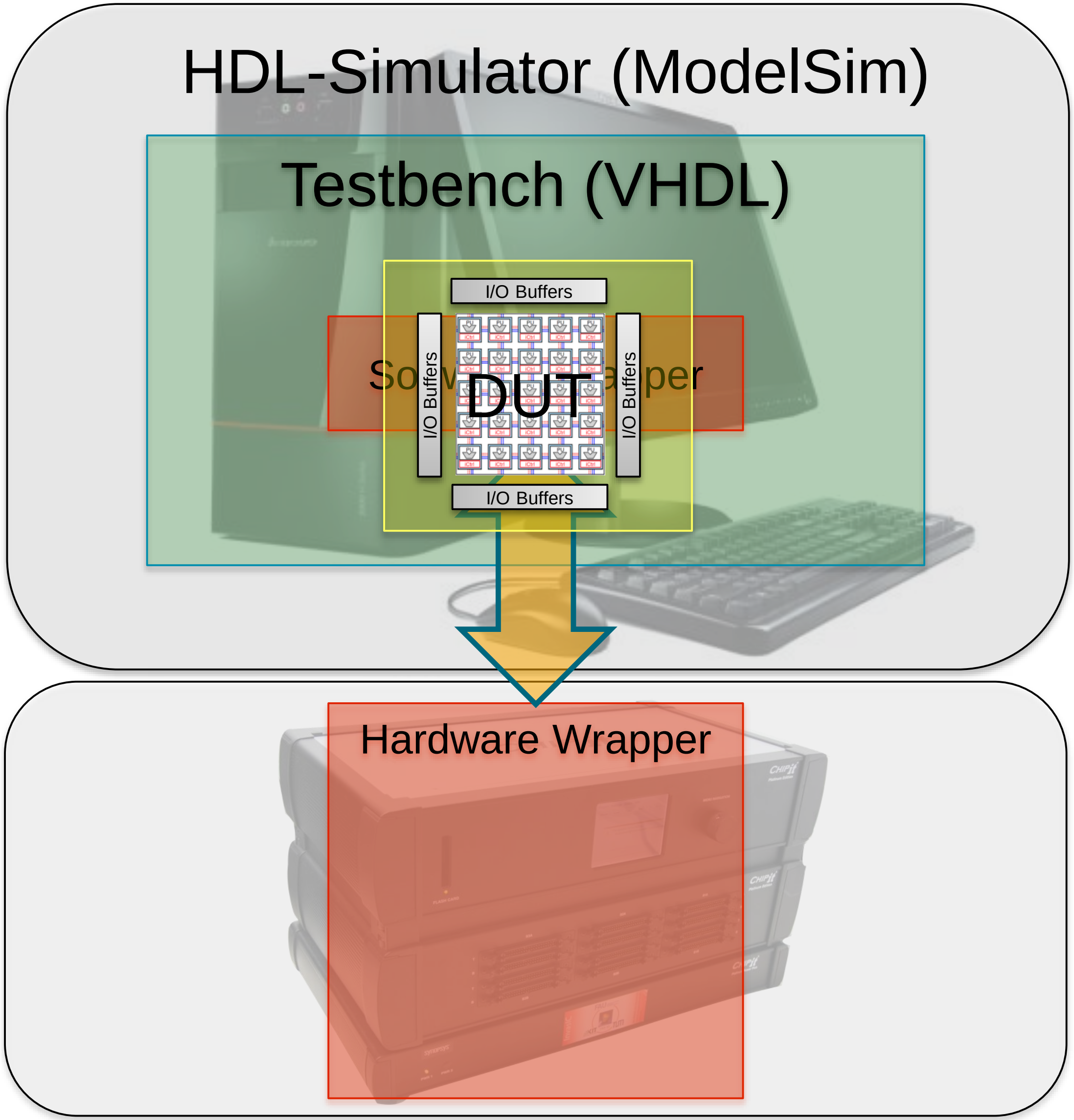
Typical HDL-based Development



HDL-Simulator (ModelSim)

Testbench (VHDL)

HDL-Bridge-based Debugging



HDL-Simulator (ModelSim)

Testbench (VHDL)

I/O Buffers

I/O Buffers

DUT

I/O Buffers

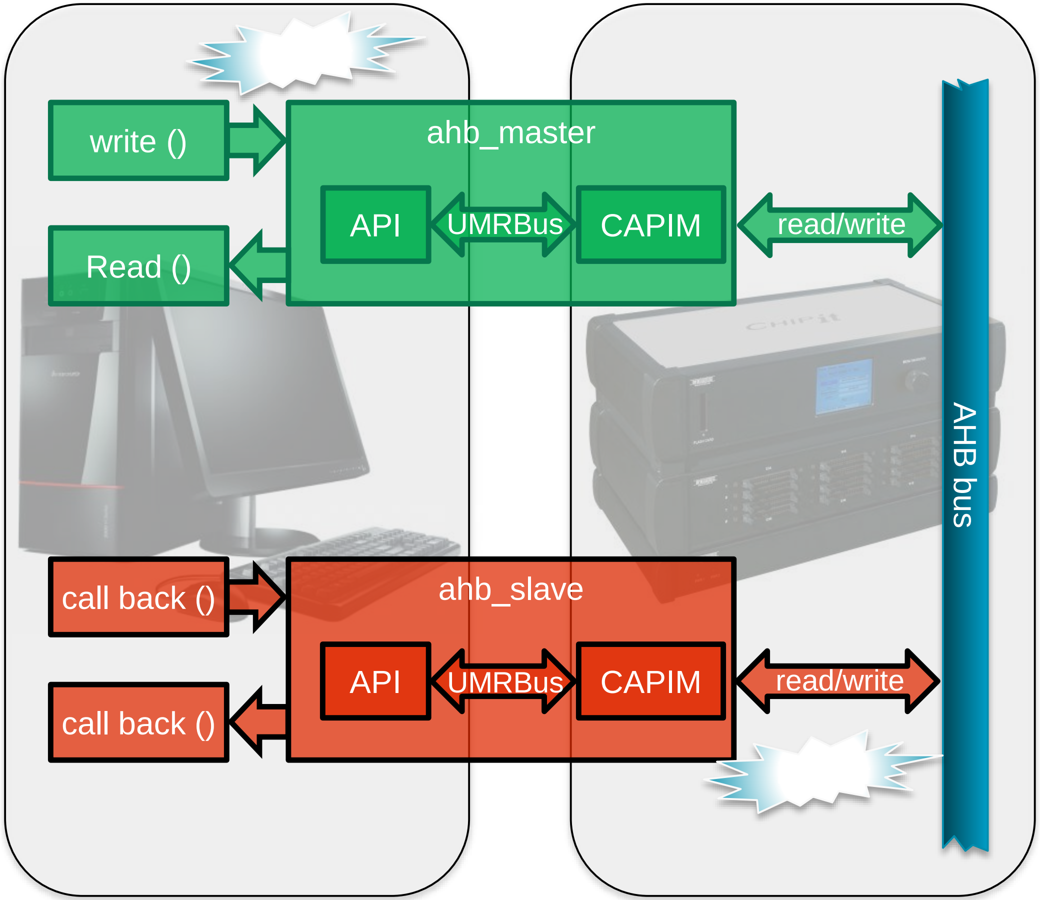
I/O Buffers

Software Wrapper

Hardware Wrapper

Synopsys Transactor Library

- Library offers UMRBus-based transactors
  - AMBA
  - UART
  - GPIO
  - …
- C++ and Tcl API
- Easy to integrate into existing RTL designs



Evaluation

- Hardware developing and debugging requires cycle accuracy and highly flexible possibilities to observe individual signals
- For software developing and testing, the performance is a key feature beside observability of registers

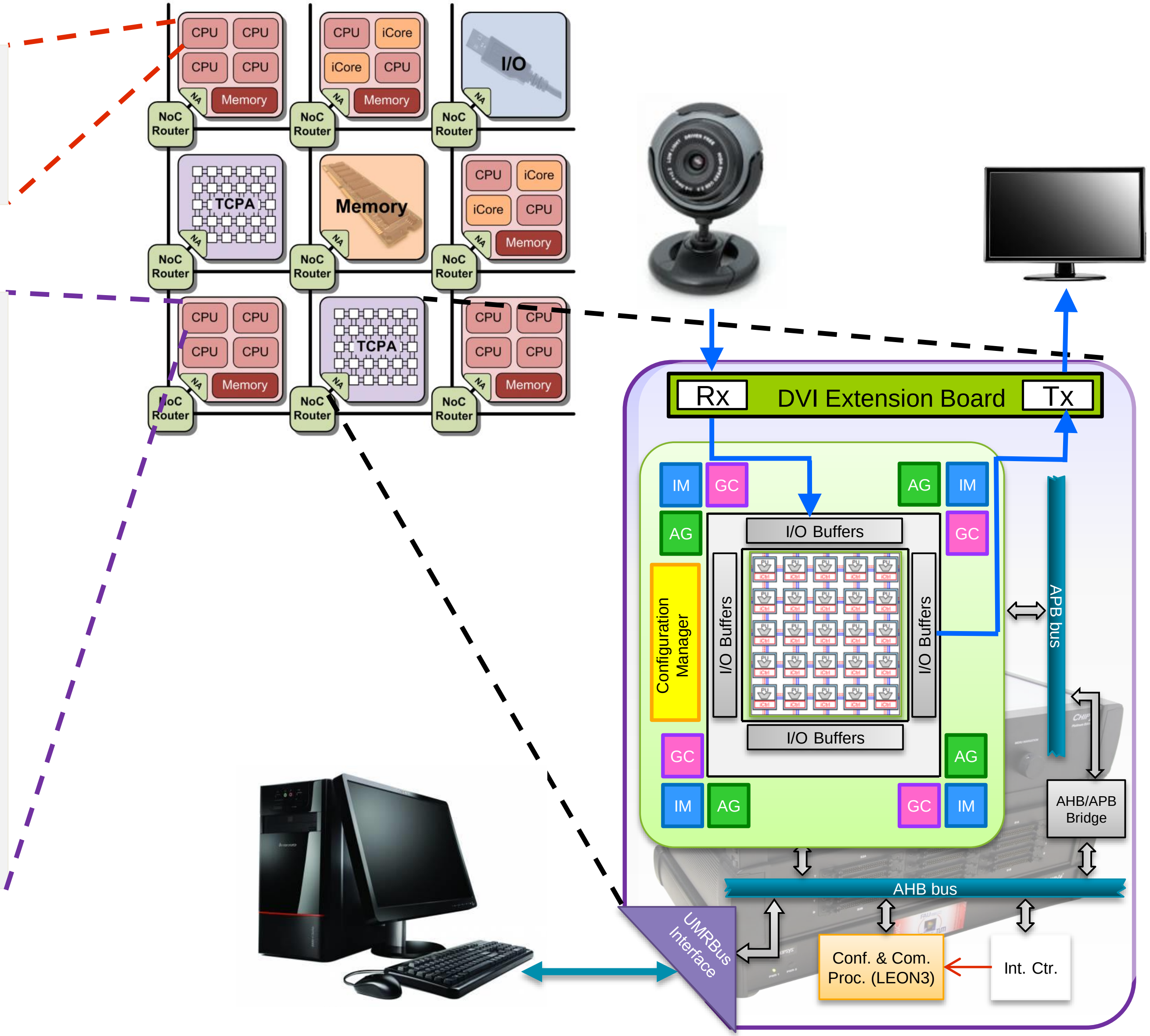
	Performance	Cycle accuracy	Signal observability	Intended use
HDL-Simulation	slowest	yes	high	hardware development
HDL-Bridge	slow	yes	medium	hardware debugging
AMBA-Transactor	high	no	low	integration and extended testing

Test Application

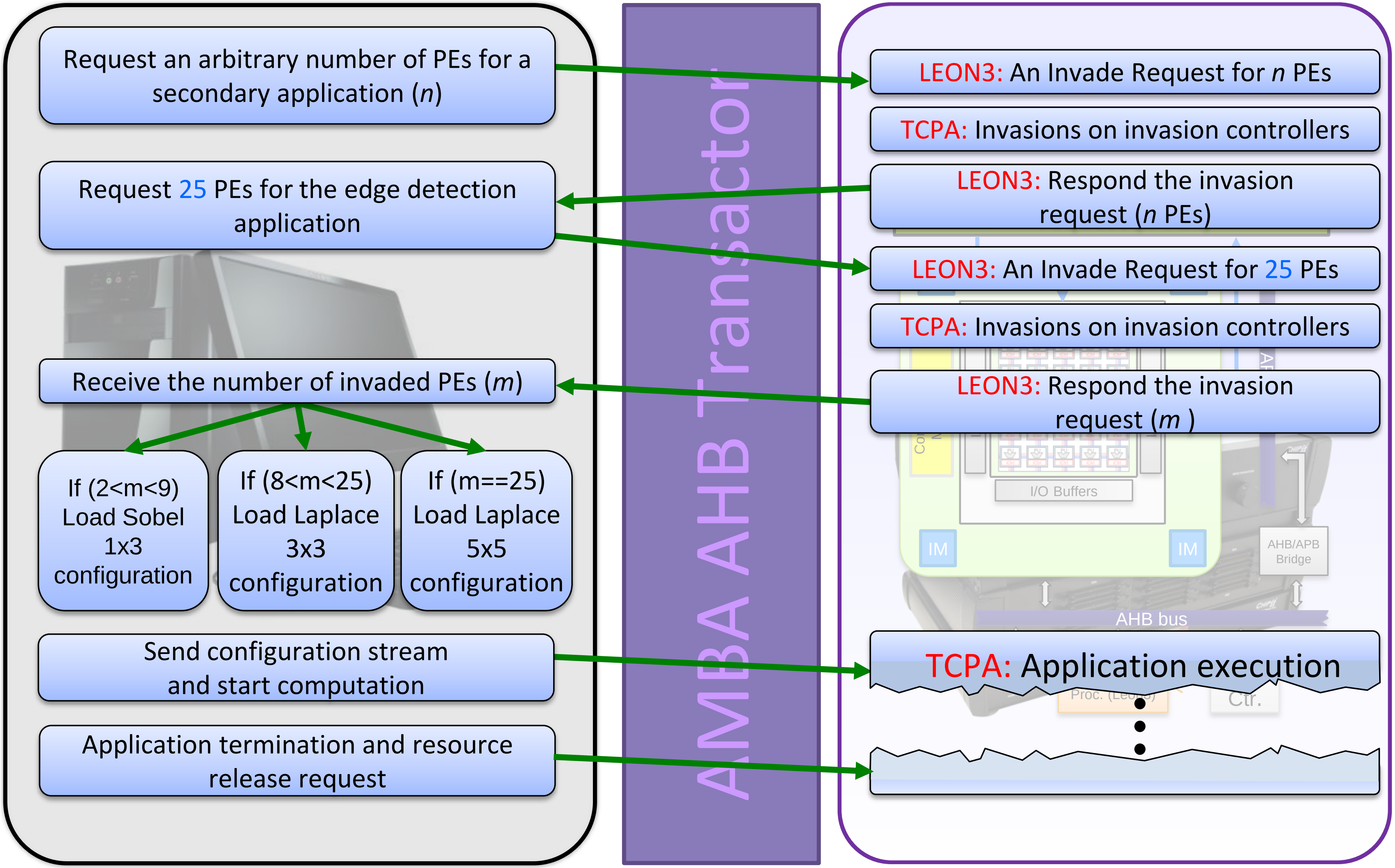
A secondary application pre-occupies a number of PEs on the target TCPA-tile

Now, the main video based application (**Edge detection**) tries to capture the remaining PEs on the TCPA tile, while satisfying the following properties:

- Guaranteed constant throughput for a 1024x768 frame resolution
- Dynamic adaptation of quality of service (Laplace or Sobel)



Hardware/Software Interactions



Request an arbitrary number of PEs for a secondary application (*n*)

Request 25 PEs for the edge detection application

Receive the number of invaded PEs (*m*)

If (2<m<9) Load Sobel 1x3 configuration

If (8<m<25) Load Laplace 3x3 configuration

If (m==25) Load Laplace 5x5 configuration

Send configuration stream and start computation

Application termination and resource release request

LEON3: An Invade Request for *n* PEs

TCPA: Invasions on invasion controllers

LEON3: Respond the invasion request (*n* PEs)

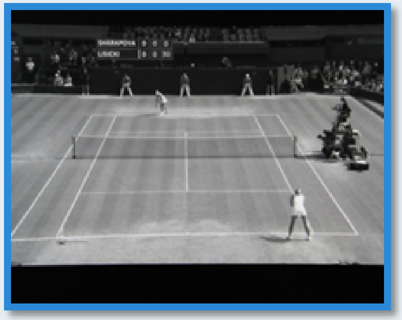
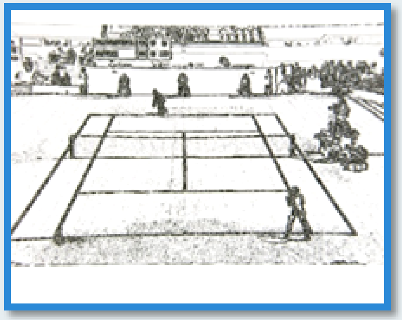
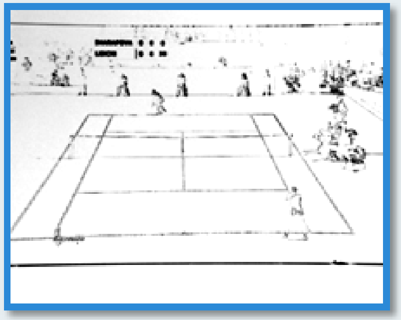
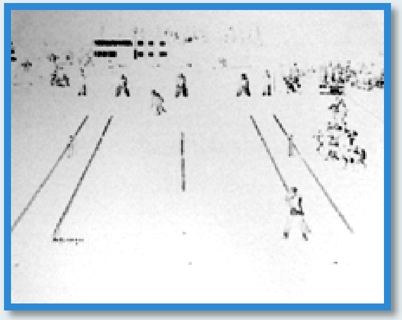
LEON3: An Invade Request for 25 PEs

TCPA: Invasions on invasion controllers

LEON3: Respond the invasion request (*m*)

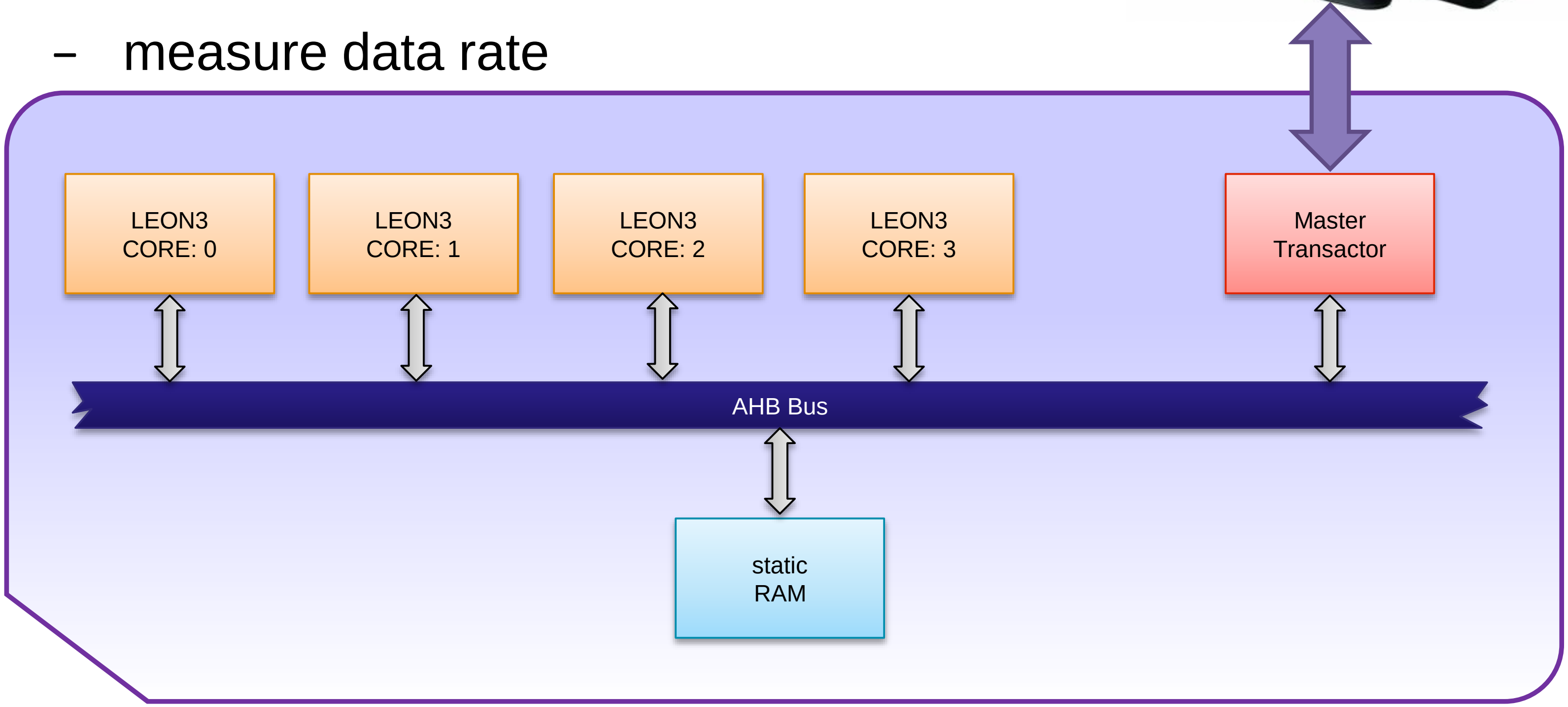
TCPA: Application execution

Application Scenarios / Results

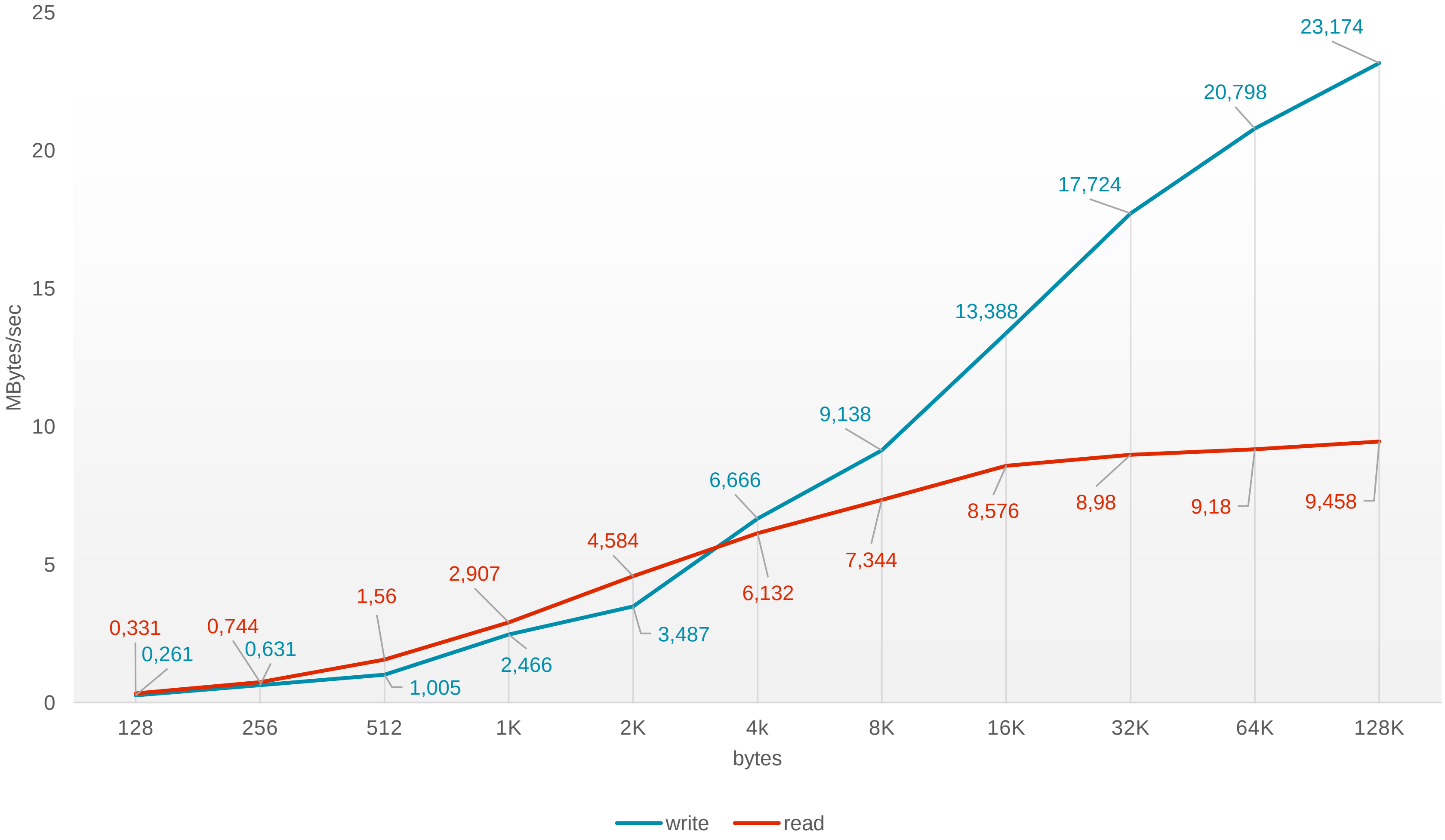
TCPA array Utilization	Scenario 1	Scenario 2	Scenario 3
🟩 Claimed PEs 🟧 Pre-occupied PEs 🟦 Free PEs	🟩🟩🟩🟩🟩 🟩🟩🟩🟩🟩 🟩🟩🟩🟩🟩 🟩🟩🟩🟩🟩 🟩🟩🟩🟩🟩	🟦🟦🟩🟩🟩 🟦🟦🟩🟩🟩 🟦🟦🟩🟩🟩 🟧🟧🟧🟧🟧 🟧🟧🟧🟧🟧	🟦🟦🟩🟩🟩 🟧🟧🟧🟧🟧 🟧🟧🟧🟧🟧 🟧🟧🟧🟧🟧 🟧🟧🟧🟧🟧
Application: Edge Detection	Laplace 5x5: $H(i,j)=\begin{pmatrix} -4 & -1 & 0 & -1 & -4 \\ -1 & 2 & 3 & 2 & -1 \\ 0 & 3 & 4 & 3 & 0 \\ -1 & 2 & 3 & 2 & -1 \\ -4 & -1 & 0 & -1 & -4 \end{pmatrix}$	Laplace 3x3: $H(i,j)=\begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}$	Sobel 1x3: $H(i,j)=\begin{pmatrix} -1 & 2 & -1 \end{pmatrix}$
Input Stream 	**Output Stream** 	**Output Stream** 	**Output Stream** 

Experimental Setup

- 1. Step
  - Write data to the RAM
  - measure data rate
- 2. Step
  - Read data from RAM
  - measure data rate



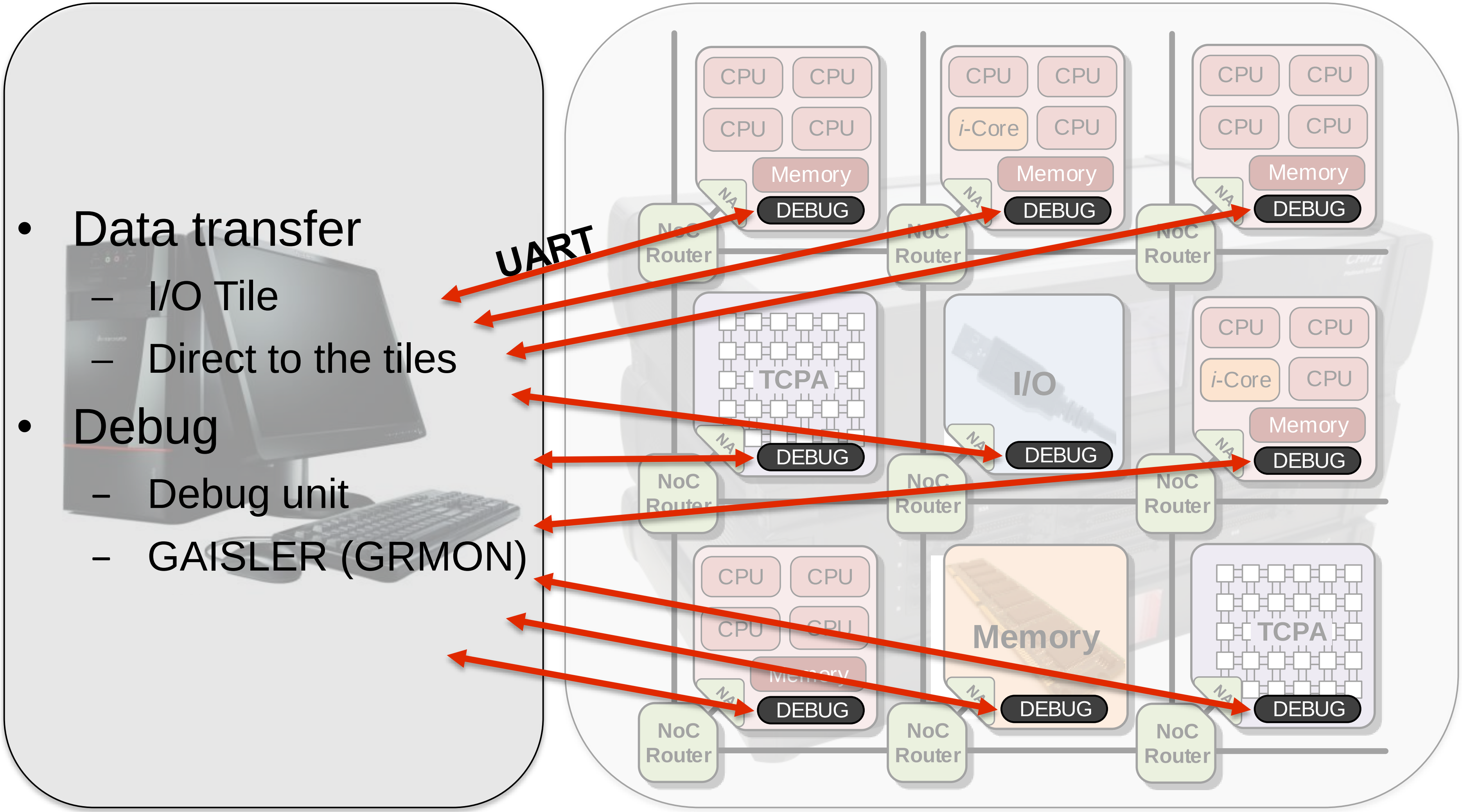
Master Transactor Data Rate



Software Development

- GRMON
  - General debug monitor for the LEON3 processor
    - Read/write access to all system registers and memory
    - Built-in disassembler and trace buffer management
    - Downloading and execution of LEON applications
    - Breakpoint and watchpoint management
    - Support for USB, JTAG, RS232, PCI, and Ethernet debug links
    - Tcl interface (scripts, procedures, variables, loops etc.)
- Challenges
  - Initial situation offered by GAISLER
    - Bus-based MPSoC with up to 16 cores and only one GRMON instance
  - But, we need a GRMON instance to each tile
    - Each instance needs a separate connection medium to CHIPit
    - Synchronization between the tiles

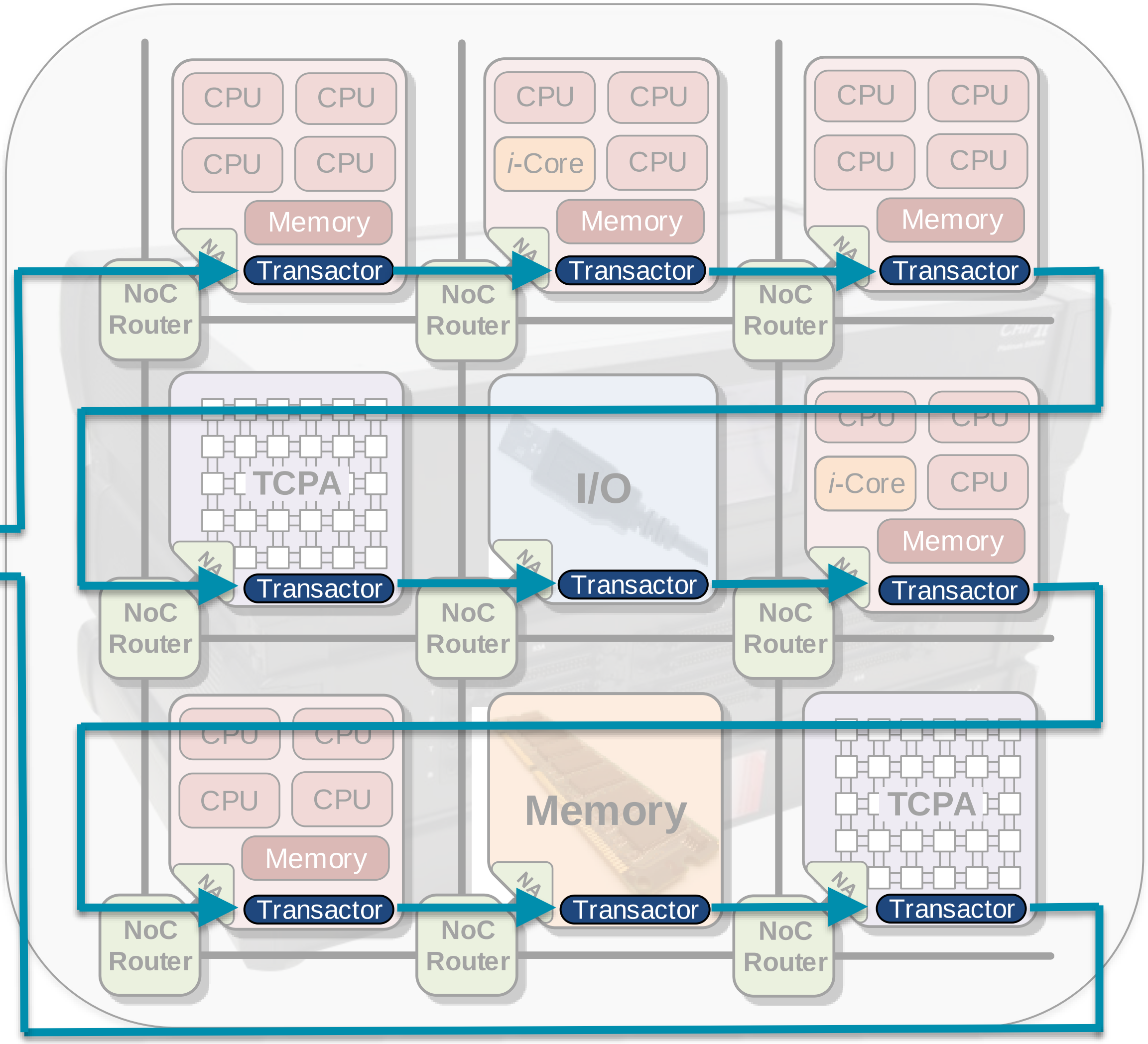
GRMON Debugging



- Data transfer
  - I/O Tile
  - Direct to the tiles
- Debug
  - Debug unit
  - GAISLER (GRMON)

Multiple Transactor-based Debugging

- Data transfer
  - I/O Tile
  - Direct to the tiles
- Debug
  - AMBA Transactor
  - GAISLER (GRMON)

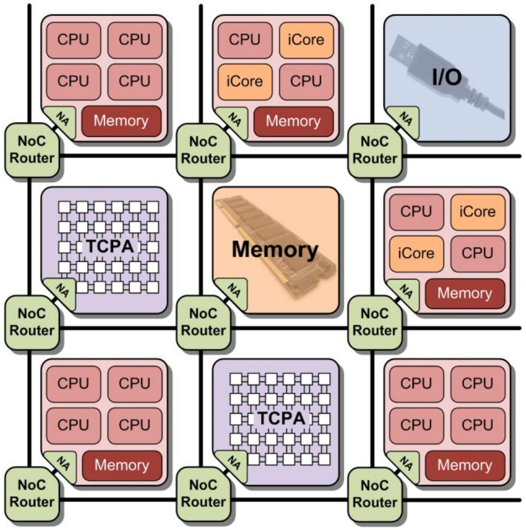


Conclusions

- HDL-Bridge-based debugging enables efficient and precise hardware development on multiple FPGAs
- AHB transactor interface eased connectivity and control over FPGA-based prototype
- Transactor-based debugging offers fast and scalable hardware-software interaction of heterogeneous MPSoC
- Our FPGA-based prototyping approach is feasible for MPSoC validation and demonstration

Thank you for your attention!

Transactor-based debugging of massively parallel processor array architectures



Contact

Markus Blocherer
Hardware/Software Co-Design
Universität Erlangen-Nürnberg
Cauerstraße 11, 91058 Erlangen, Germany
Email: markus.blocherer@fau.de

www.invasive-computing.org