MPSoC Design: *The Tomahawk Approach*

Oliver Arnold

Vodafone Chair Mobile Communications Systems
Technical University of Dresden

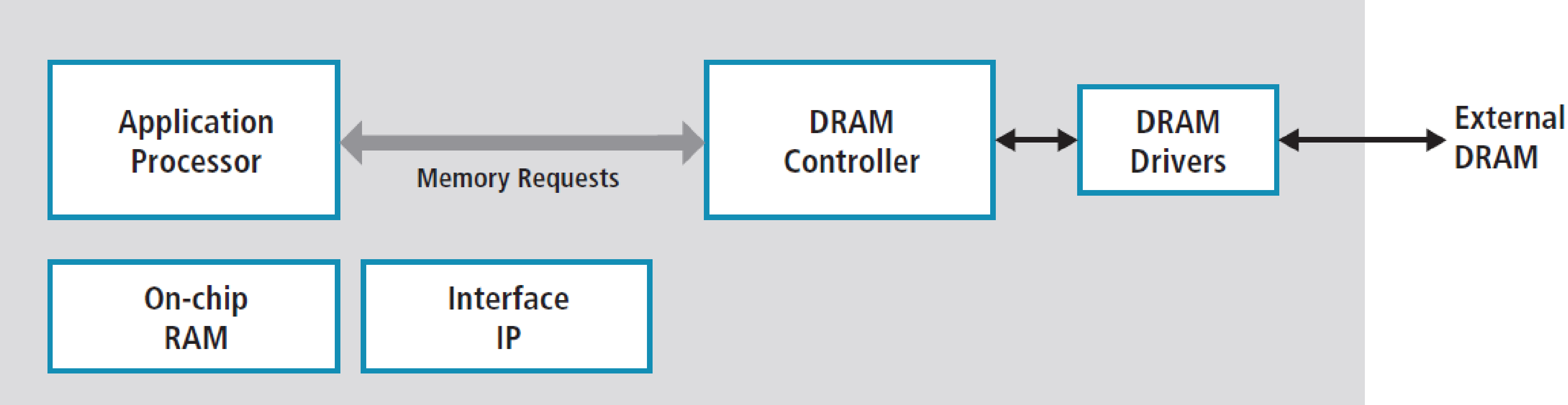
Winter school on "Design and Applications of Multi-Processors System on Chip", 24-28 November 2014, Tunis, Tunisia

Agenda

- Motivation
- MPSoC Design and Challenges
- Tomahawk Architecture Framework
    - Programming Model
    - Runtime Management
    - Network-on-Chip
    - PE Integration Framework
- Tools
- Tomahawk2 heterogeneous MPSoC
    - Silicon Prototype
    - System Design
    - Results

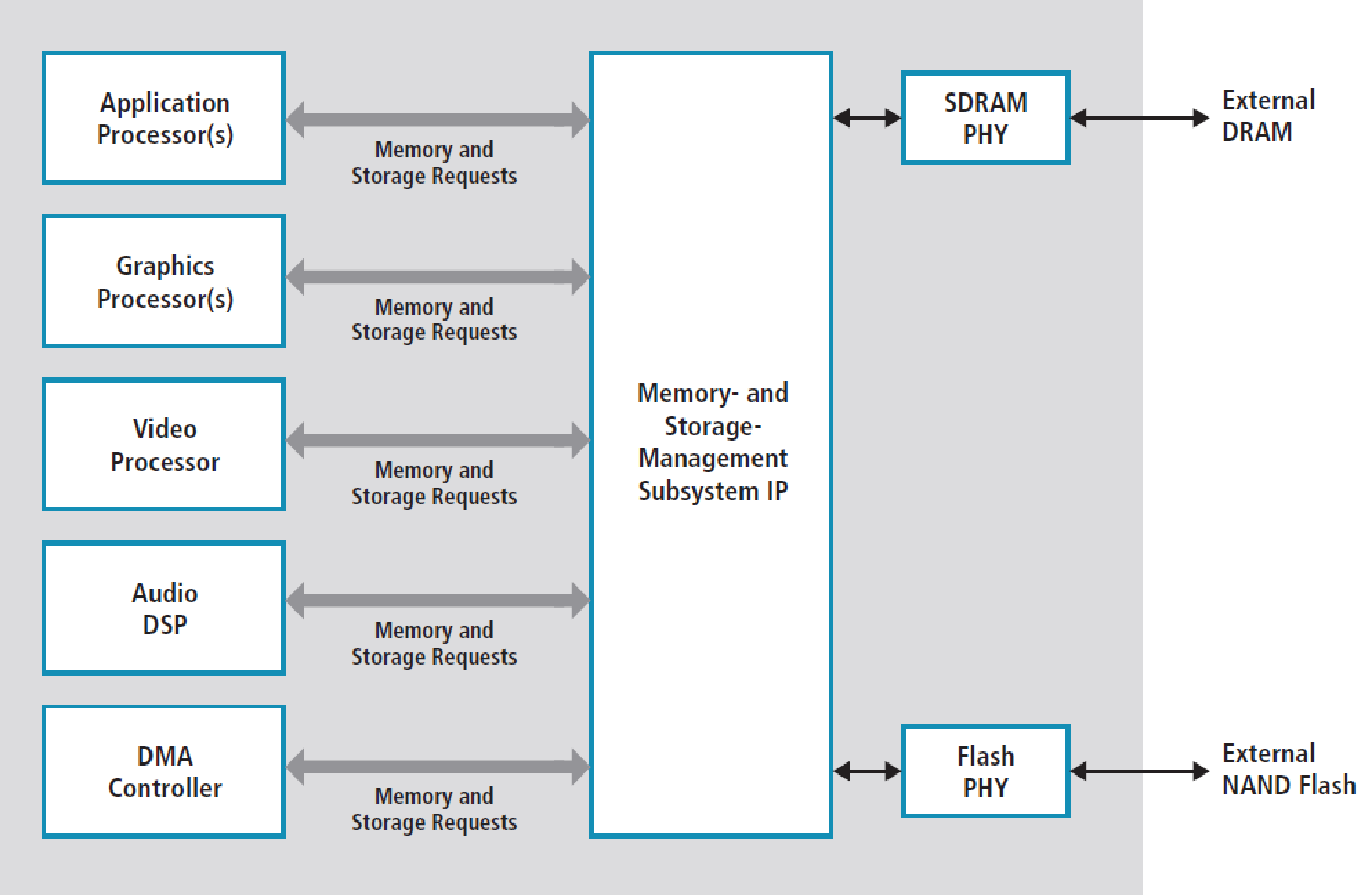
MOTIVATION

Before Year 2000: SoC



[Cadance, MPSoC Designs: Driving Memory and Storage Management IP to Critical Importance, 2011]

After Year 2010: MPSoC


Application Processor(s)
Memory and Storage Requests
Graphics Processor(s)
Memory and Storage Requests
Video Processor
Memory and Storage Requests
Audio DSP
Memory and Storage Requests
DMA Controller
Memory and Storage Requests
Memory- and Storage-Management Subsystem IP
SDRAM PHY
External DRAM
Flash PHY
External NAND Flash

[Cadance, MPSoC Designs: Driving Memory and Storage Management IP to Critical Importance, 2011]

MPSoC Design

- **How many processors?**
  How should they be configured?
  Should they be homogeneous, heterogeneous or a combination of the two?

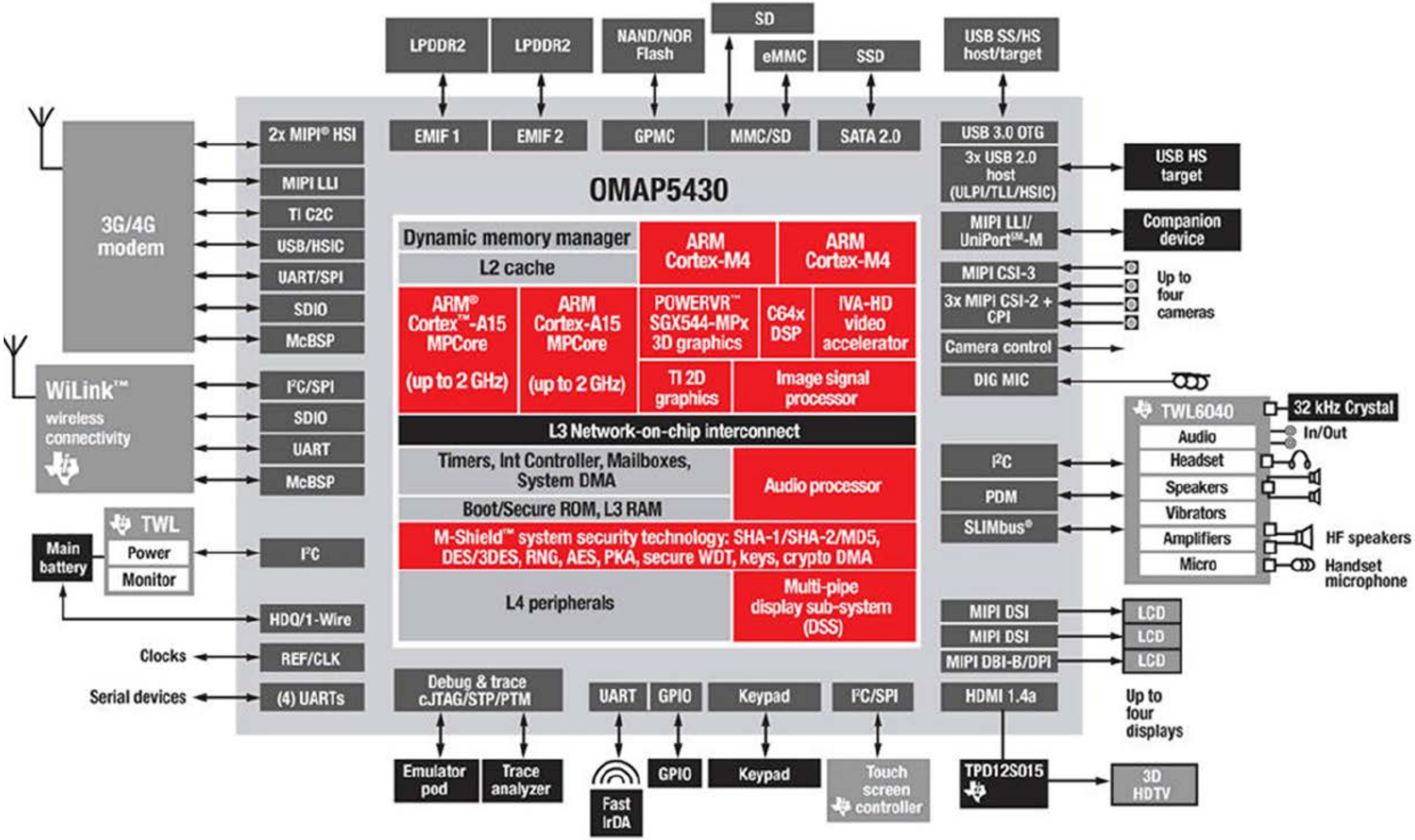
- **How do blocks communicate?**
  Standard hierarchical buses, shared memory, point to point, network-on-chip (NoC), or a combination?

- **What is the memory hierarchy?**
  How much local instruction and data memory for each processor?
  How much system memory and how is it organized?

- What is the **concurrency, synchronization** and **control model** for the applications, and what programming model should be used?

- How do you control **energy consumption** and manage the system for low power?

OMAP5430 MPSoC



[Cadance, MPSoC Designs: Driving Memory and Storage Management IP to Critical Importance, 2011]

Examples of MPSoCs

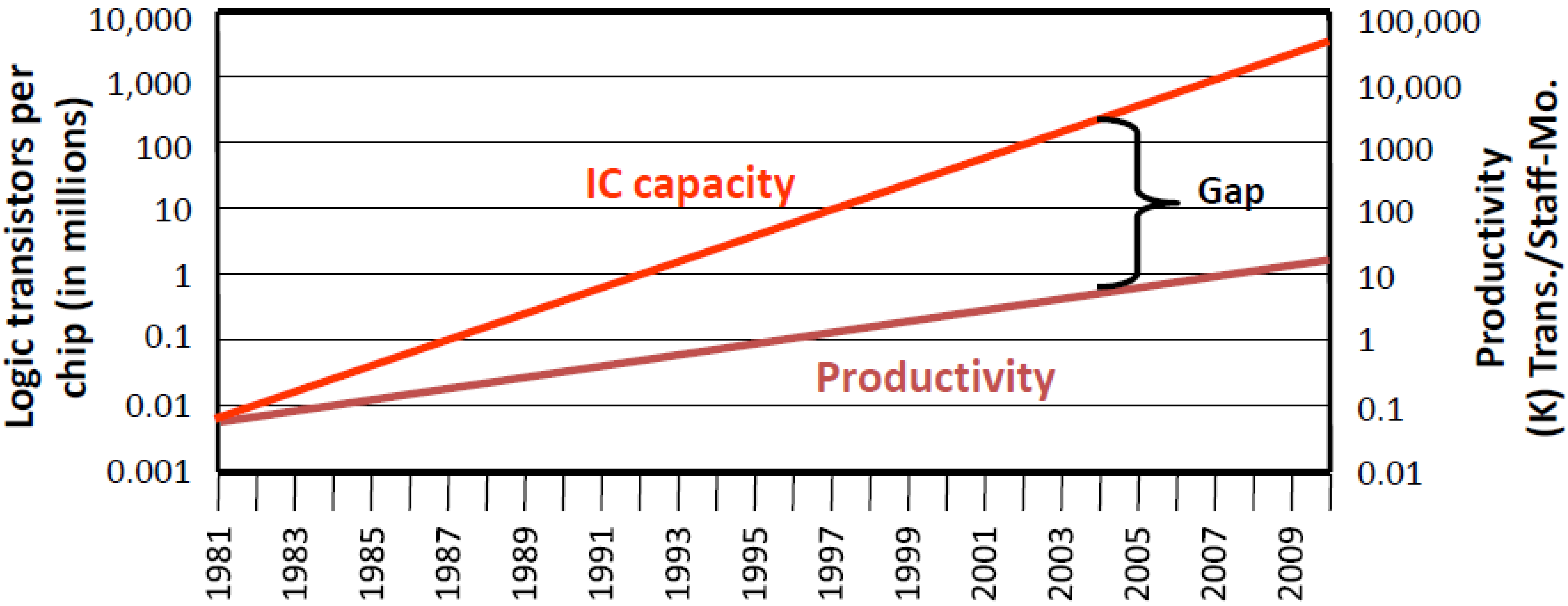
▪ Nomadik (ST)	- multimedia
▪ Cell (IBM / Sony / Toshiba)	- multimedia/games
▪ ARM11 MPCore (ARM)	- multimedia
▪ MeP (Toshiba)	- multimedia
▪ MP-211 (NEC)	- multimedia/graphics
▪ UniPhier (Panasonic)	- multimedia
▪ OMAP (TI)	- multimedia
▪ Nexperia (NXP)	- multimedia
▪ Emotion Engine (Sony)	- games (Playstation)
▪ CRS-1 (CISCO)	- networking
▪ OnDSP ➔ EVP (NXP)	- baseband WLAN, DVB-H
▪ MUSIC (Infineon)	- baseband 3G (SDR)
▪ Sandblaster (Sandbridge)	- baseband SDR
▪ SODA (UNI Michigen)	- baseband SDR
▪ CoreWerk (Dresden Silicon)	- multimedia
▪ …	

MPSOC DESIGN AND CHALLENGES

MPSoCs

- (Massively) Parallel Systems on a single chip!
- Application-specific: automotive electronics, avionics, multimedia, consumer electronics, etc.
- E.g., media and signal processing:
  - ❑ Support multiple applications and various standards
  - ❑ Provide real-time performance
  - ❑ Heterogeneous system architectures
  - ❑ Short time to market (cell phones)
- Homogeneous vs. heterogeneous
- Goal: high-performance and low power
- IP reuse

IC capacity vs. Productivity



T. Givargis, F. Vahid. "Embedded System Design", Wiley 2002

MPSoC Design

- Tradeoffs:
  - Cost (silicon area, design time)
  - Performance
  - Power consumption
  - Flexibility
  - Time-to-market
- Resource management
  - Highly variable resource demand vs. highly variable resource availability
  - Event triggered ⇔ time triggered systems
  - Synchronization
  - Scheduling approach / algorithm

Embedded Systems Software Design

- Software: Major challenge in MPSoC
  - Portability etc.

- Goals:
  - High performance
  - Real time
  - Low power

- Algorithm partitioning: Task-level parallelism
  - Task abstraction
  - Low-level behavior characteristics for system-level analysis

[Jerraya, Wolf: The What, Why, and How of MPSoCs]

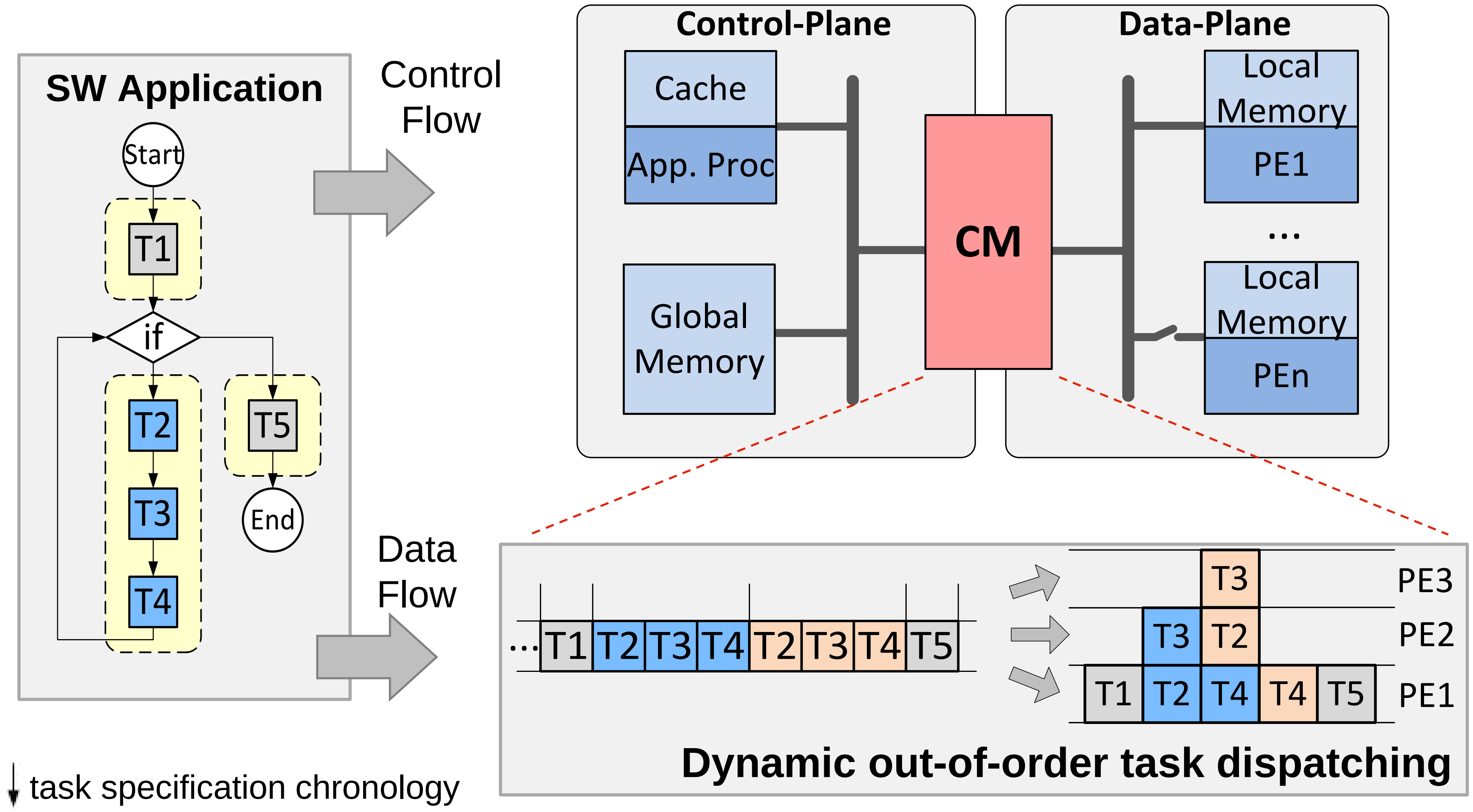
Data Locality Principle

- Eliminate the use of registers, where possible
  - ➔ Direct communication between functional unit, …
- Eliminate the memory read/write operations, where possible
  - ➔ Compiler, …
- Eliminate the processor-to-processor data transfer, where possible
  - ➔ Runtime management, …

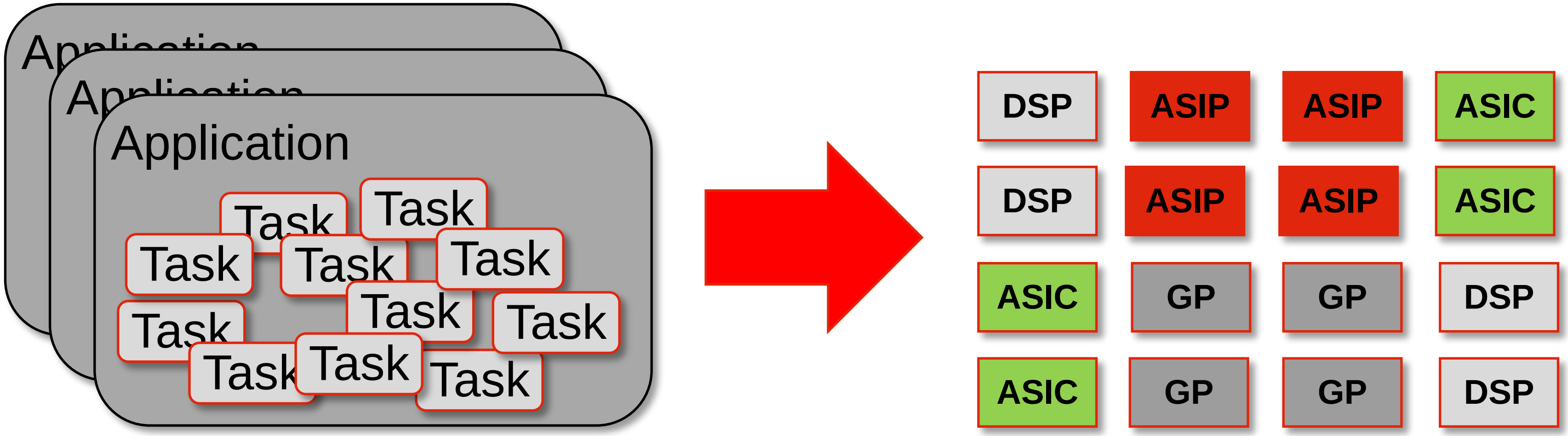
➔ **Data management is very important**

TOMAHAWK ARCHITECTURE FRAMEWORK

Tomahawk Framework



Scheduling Approach



DSP	ASIP	ASIP	ASIC
DSP	ASIP	ASIP	ASIC
ASIC	GP	GP	DSP
ASIC	GP	GP	DSP

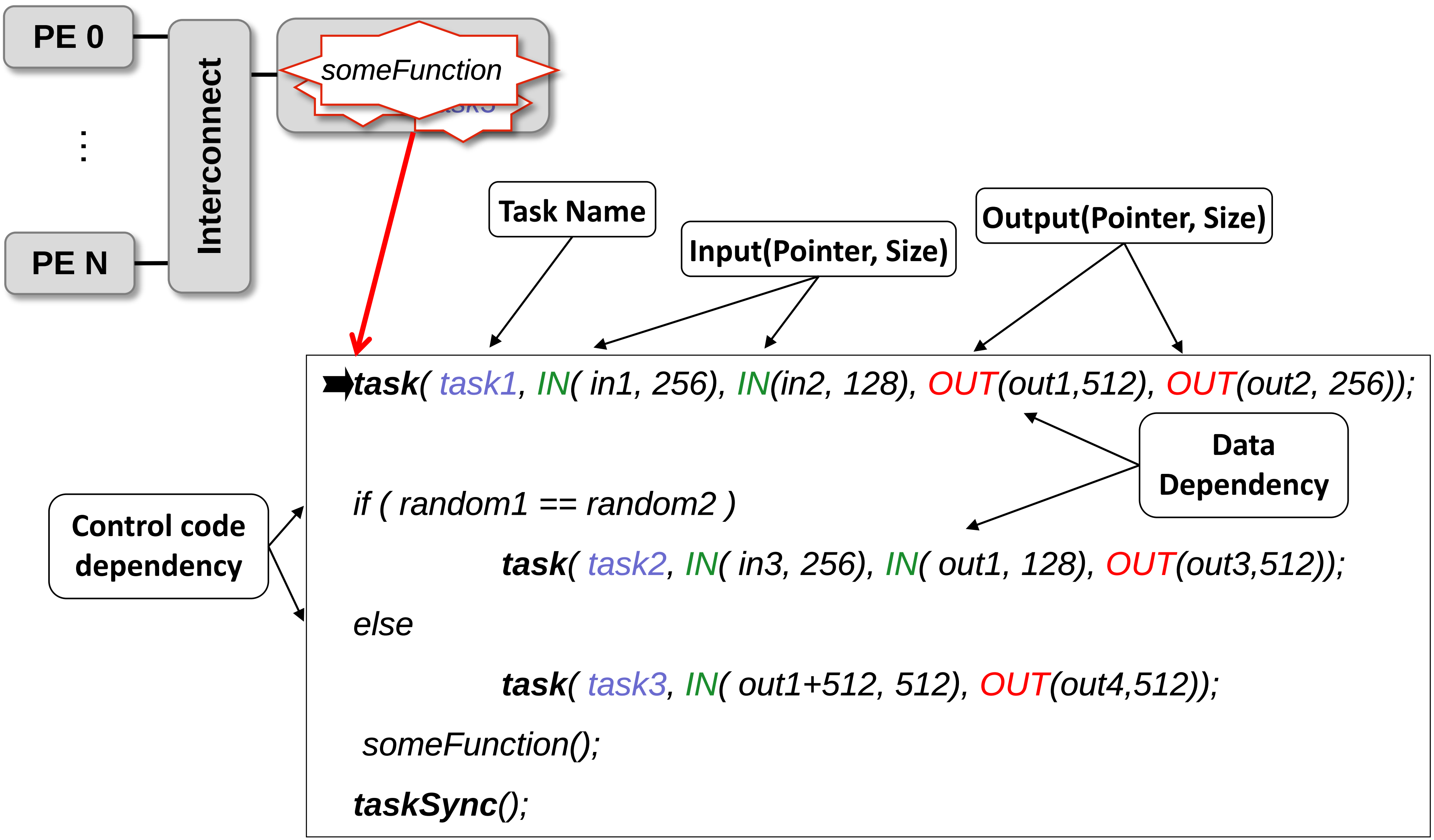
- **Challenges:**
  - Several types and numbers of applications concurrently running on the same hardware resources
  - Non-predictable start times of applications
  - Dynamic priorities on application level and task level
  - Variable workload of applications due to unknown input parameters
  - Non-deterministic task execution times

➡ **Dynamic scheduling unit**

PROGRAMMING MODEL

TASKC

TaskC Programming Model



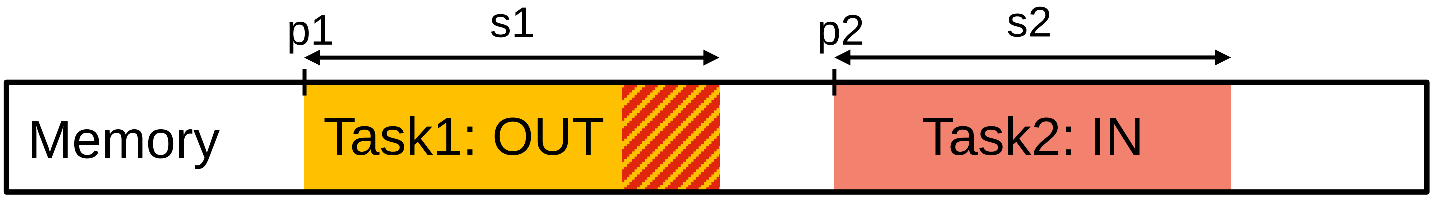
```
task( task1, IN( in1, 256), IN(in2, 128), OUT(out1,512), OUT(out2, 256));


if ( random1 == random2 )
            task( task2, IN( in3, 256), IN( out1, 128), OUT(out3,512));
else
            task( task3, IN( out1+512, 512), OUT(out4,512));
 someFunction();
taskSync();
```

RUNTIME MANAGEMENT

COREMANAGER

Overview

- **CoreManager**
  - ❑ Dedicated central task scheduling unit
  - ❑ Data plane control
  - ❑ Dynamic data dependency checking



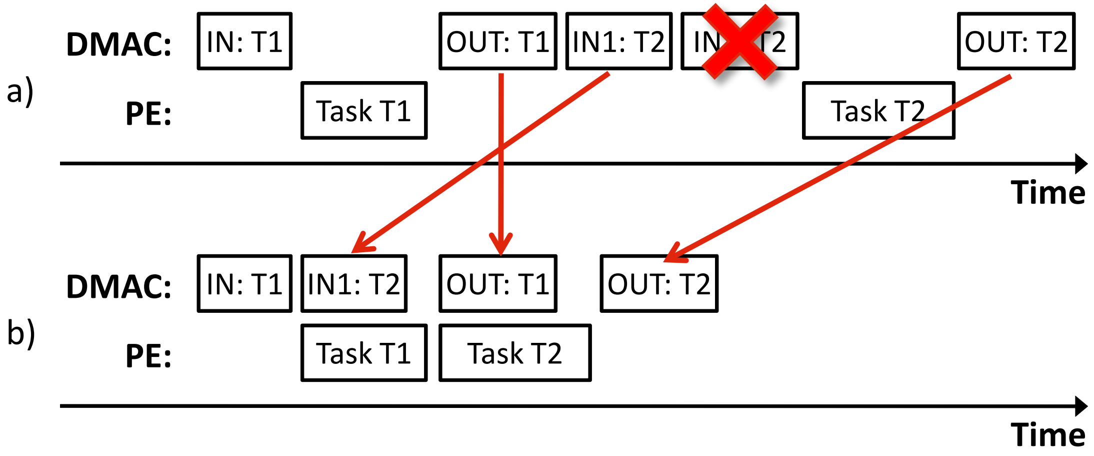
- **Processing Elements**
  - ❑ Solely work on local on-chip memories
    - ➔ no cache misses, deterministic execution time
  - ❑ Concurrent task execution and data transfers
  - ❑ Own address space
  - ❑ State-of the art debugger

CoreManager Approach

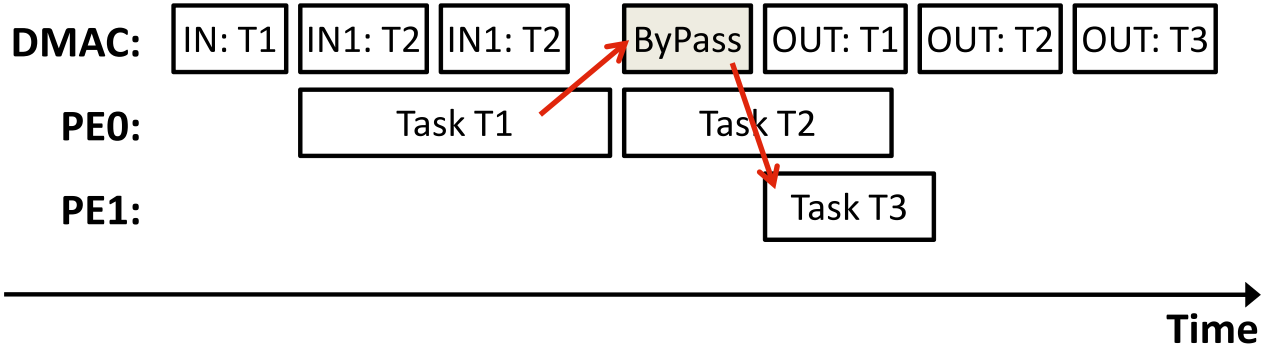
- **Scheduler**
  - ❑ Flexible
  - ❑ E.g.: list based ASAP, EDF
  - ❑ Task window of 4 / 8 / 16 / 32 … tasks
- **PE local memory allocation**
  - ❑ Explicit memory management for increased data locality
  - ❑ Allocation strategies: single space, top-down, block based
- **Power Management**
  - ❑ Frequency scaling for each PE
  - ❑ Power domain for each PE ➔ shut-off / power-on
- **Soft real time**
  - ❑ Priority levels
  - ❑ Annotations on application or task level ➔ static or dynamic
  - ❑ CoreManager prioritizes task execution **and** data transfers

Increased Data Locality

- Data Locality
  - ❑ Task1 ➔ IN, OUT
  - ❑ Task2 ➔ IN, IN, OUT

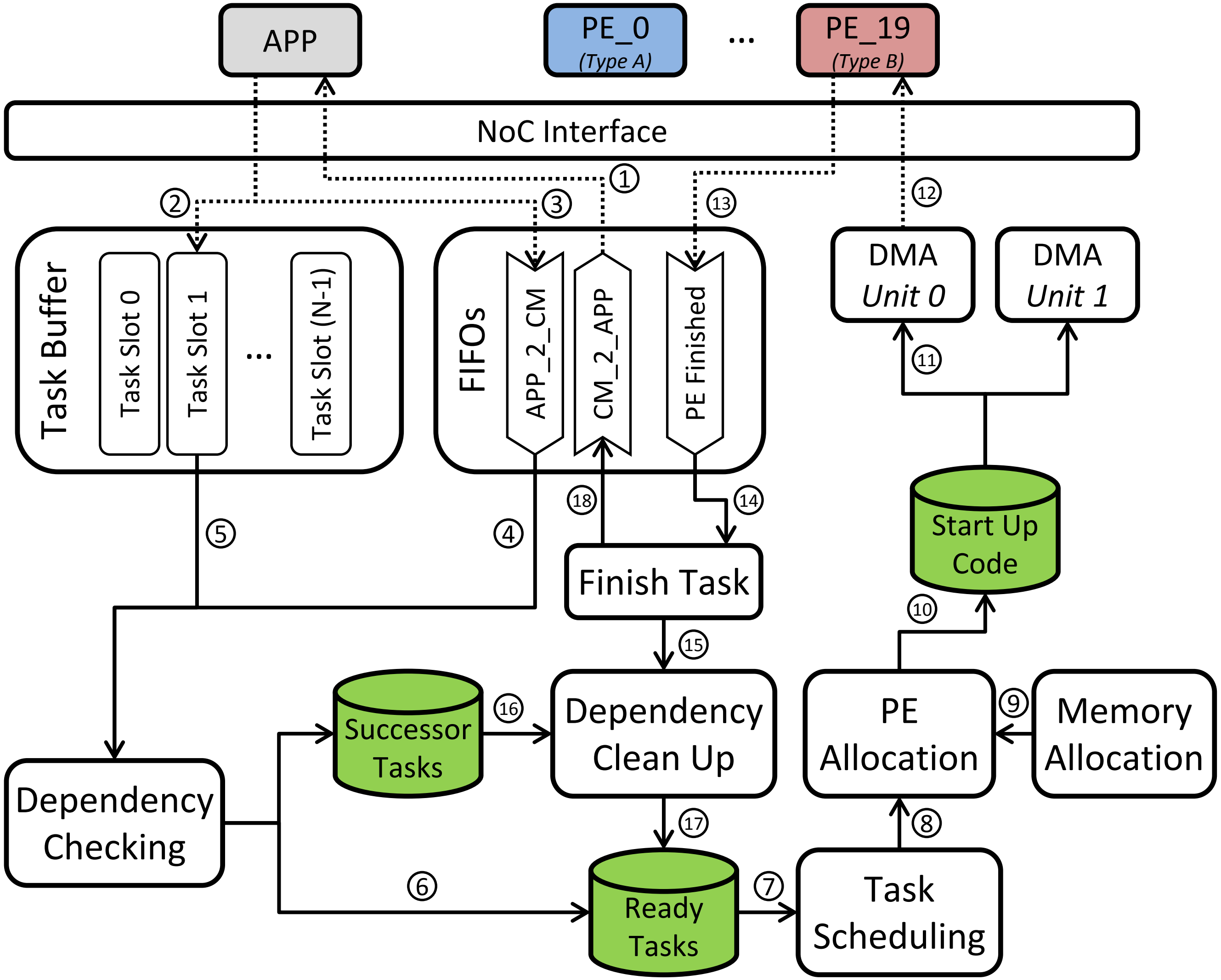


- Data Bypassing

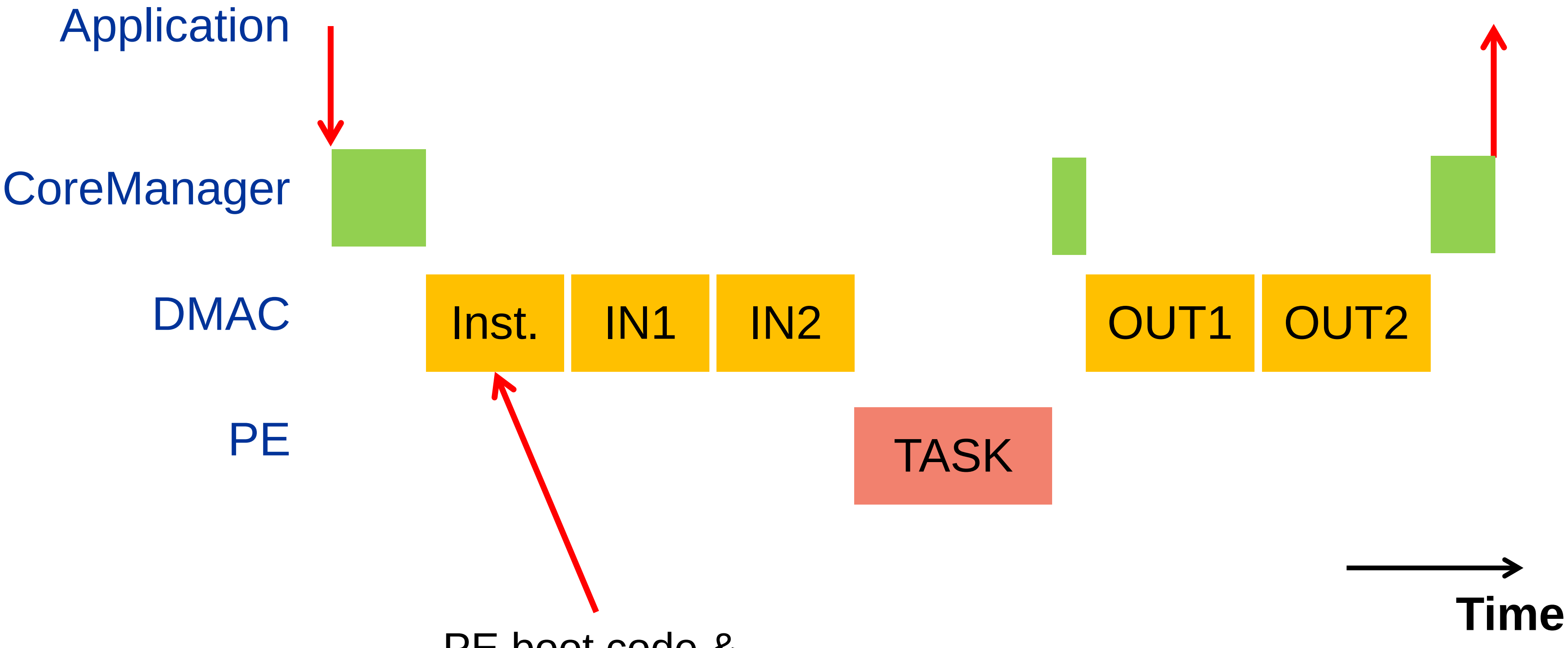


➔ **Intelligent memory management to reduce communication overhead**

Basic CoreManager Behavior



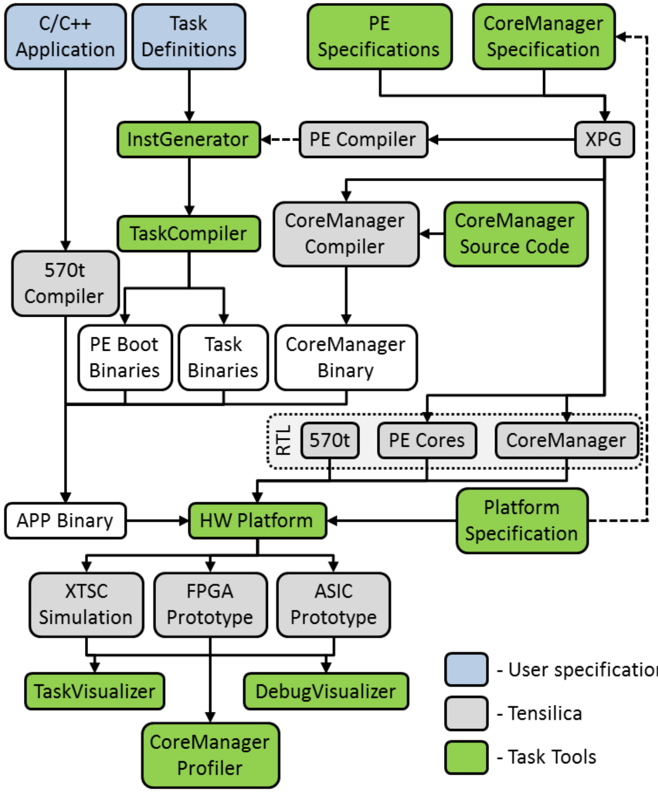
CoreManager Overhead



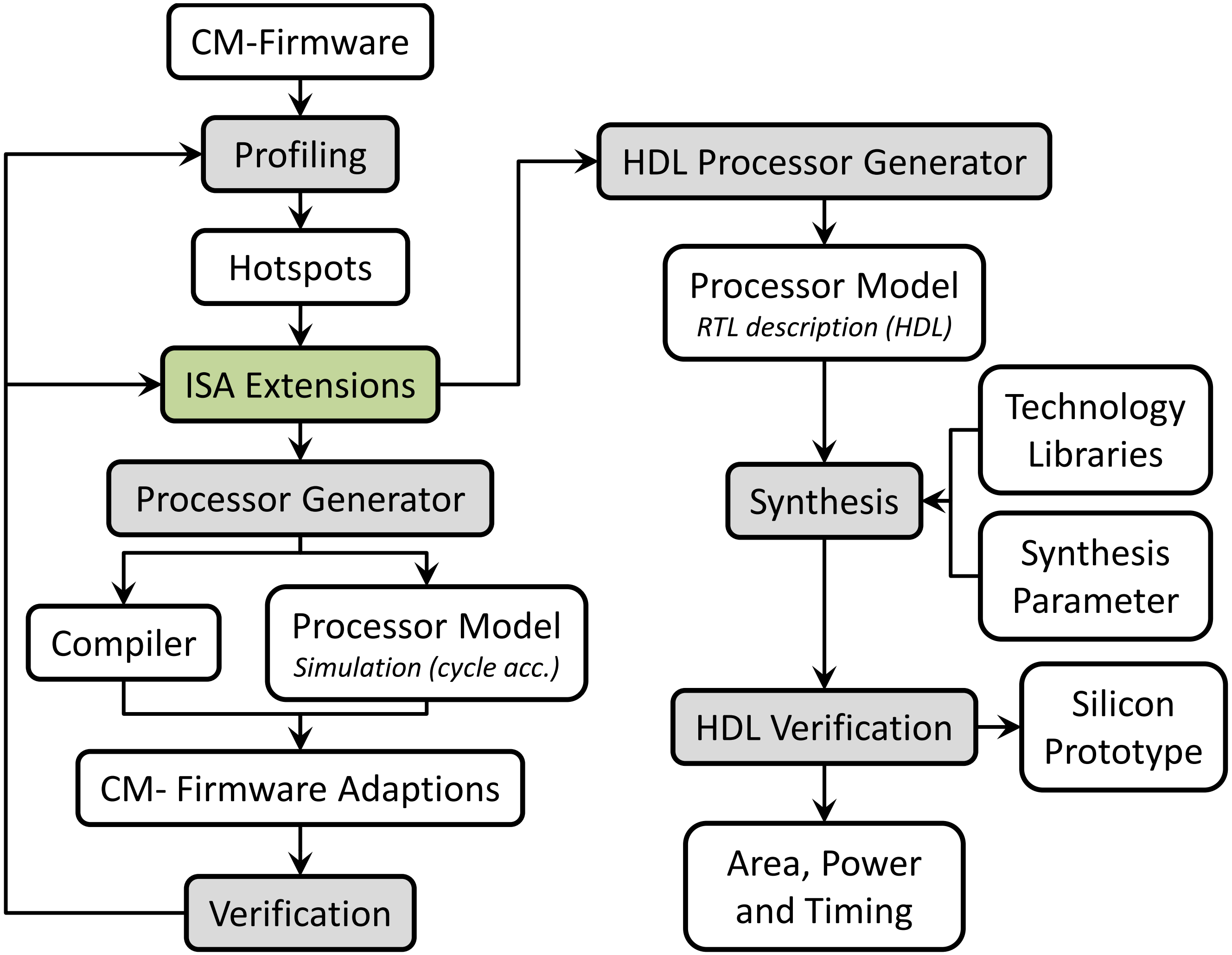
PE boot code &
task inst.

CoreManager Implementations

- Standard RISC Core
  - ❑ **CM-LX4**
  - ❑ Tensilica LX4
- RISC Core + Very Long Instruction Word
  - ❑ **CM-VLIW**
  - ❑ New 64-bit instruction word: composed of several RISC instructions
- RISC Core + Extended Instruction Set
  - ❑ **CM-EIS / CM-ASIP**
  - ❑ Hardware-Software-Co-Design
  - ❑ Application specific instruction set



Tool Flow: ISA Extensions



Dynamic Data Dependency Checking

$$\left((unsigned)(p0 - p1) < s1)\right) \,||\, \left((unsigned)(p1 - p0) < s0\right)$$

↓ **+ Merge Instructions**

$$asm\_depCheck(p0, s0, p1, s1)$$

↓ **+ 64-bit Regs (reg64_X={pX,sX})**

$$asm\_depCheck(reg64\_0, reg64\_1)$$

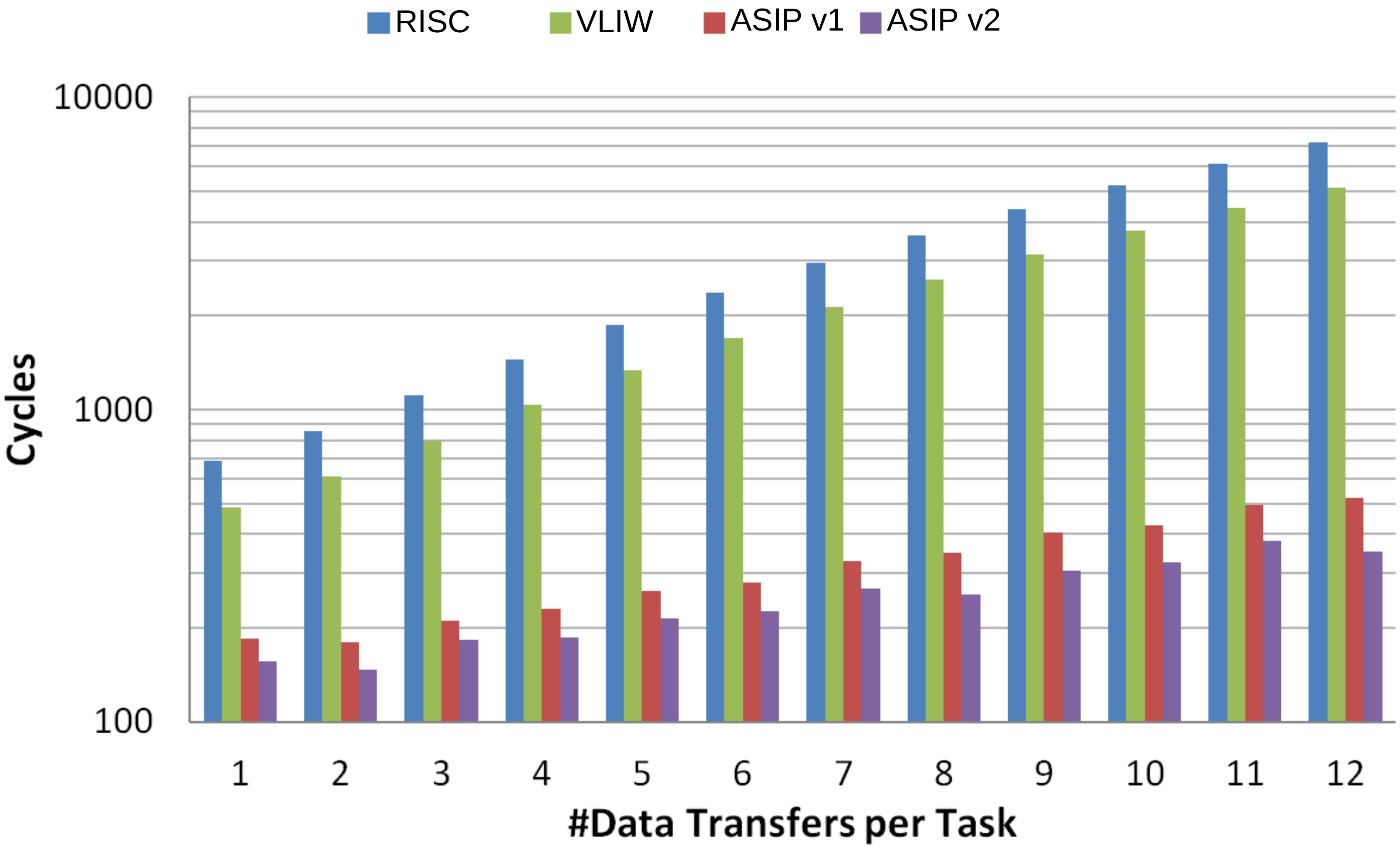
↓ **+ SIMD (4 comparisons)**

$$asm\_depCheckSIMD4(reg64\_0, reg64\_1, reg64\_2, reg64\_3)$$

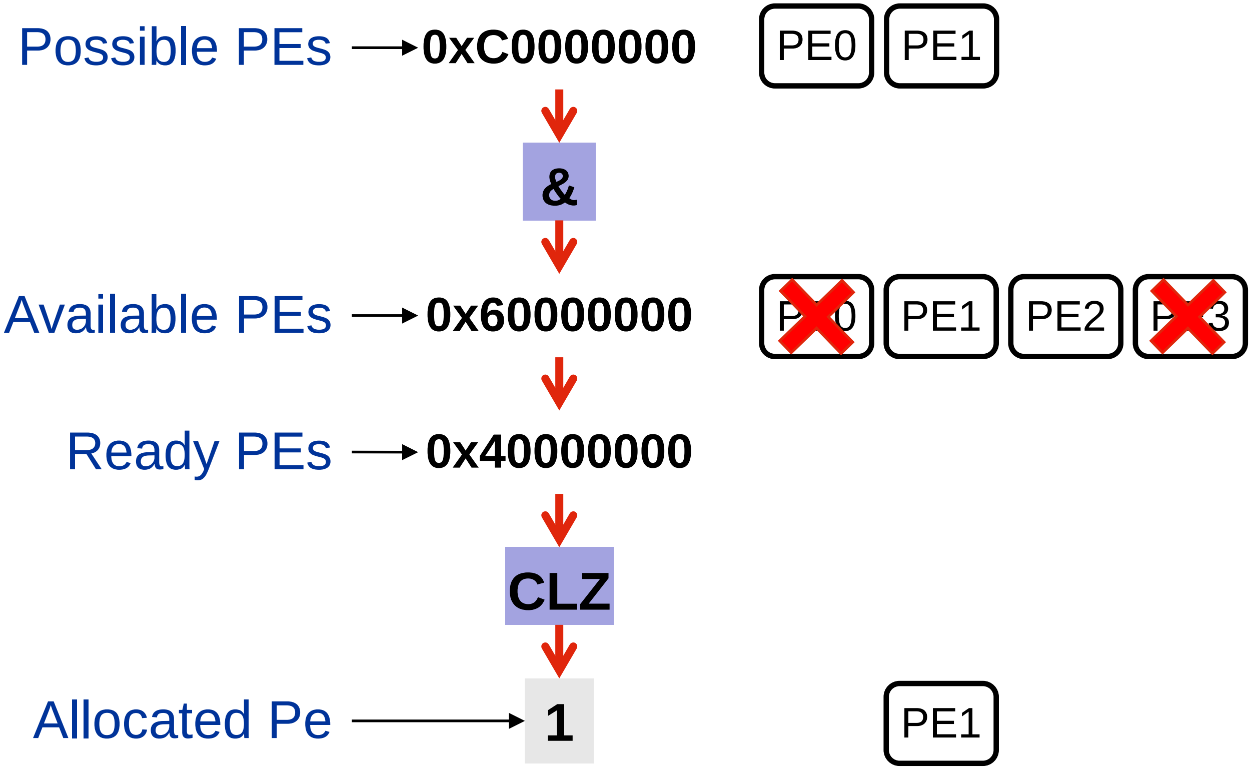
↓ **+ Explicit Load Instructions**

$$asm\_depCheckSIMD\_LD4(reg64\_2, reg64\_3)$$

Results


RISC
VLIW
ASIP v1
ASIP v2
10000
1000
100
Cycles
1
2
3
4
5
6
7
8
9
10
11
12
#Data Transfers per Task

PE Allocation: 1 bit per PE

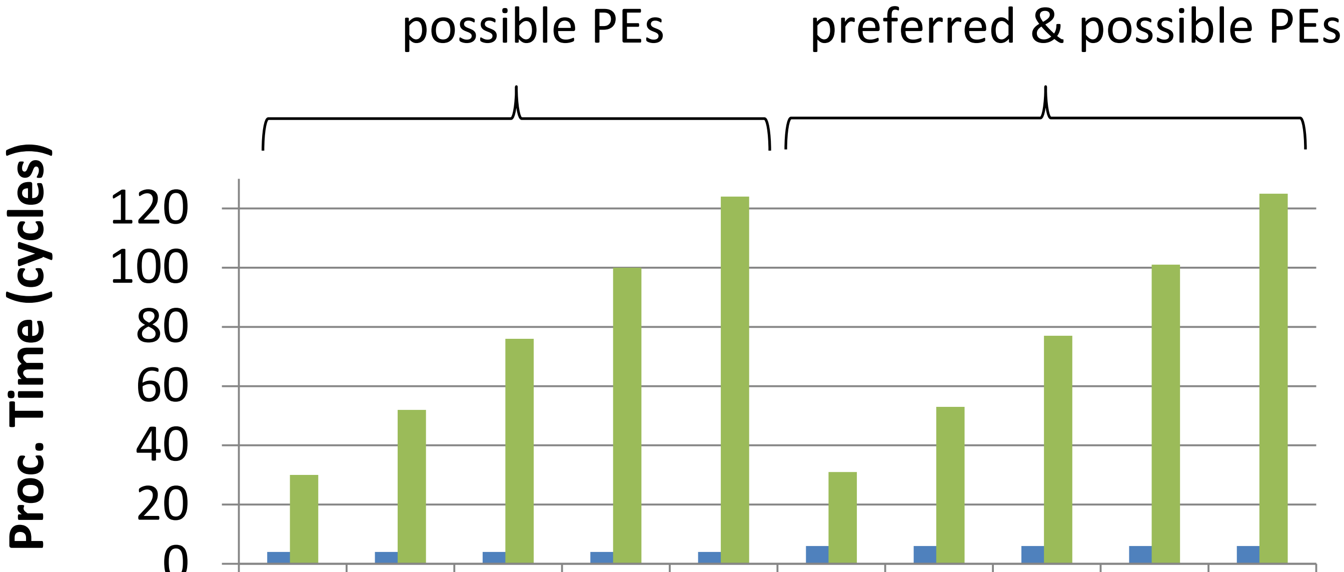


- ➔ Execution time is independent of task type and PE type
- ➔ New ISA: **asm_getPE(taskTypePEs)**

| taskTypePEs = {possiblePEs, preferredPEs}

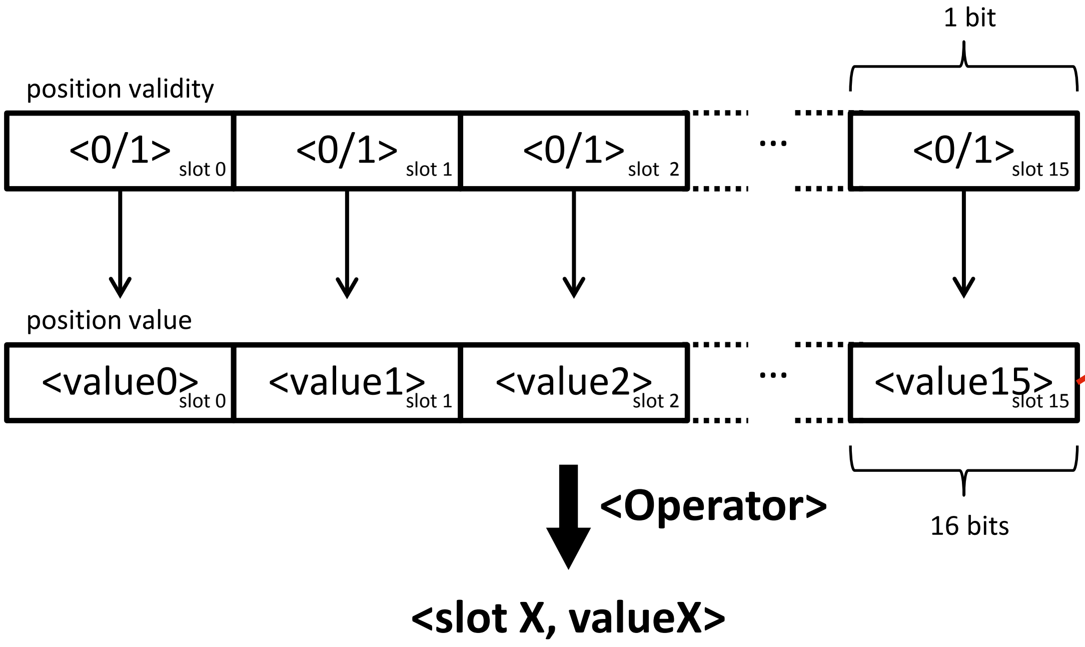
CoreManager: Dynamic PE Allocation

- **Heterogeneous MPSoC ➔ several types of cores**



PE-ID	0	7	15	23	31	0	7	15	23	31
🟦 CM-ASIP	4	4	4	4	4	6	6	6	6	6
🟩 CM-RISC	30	52	76	100	124	31	53	77	101	125

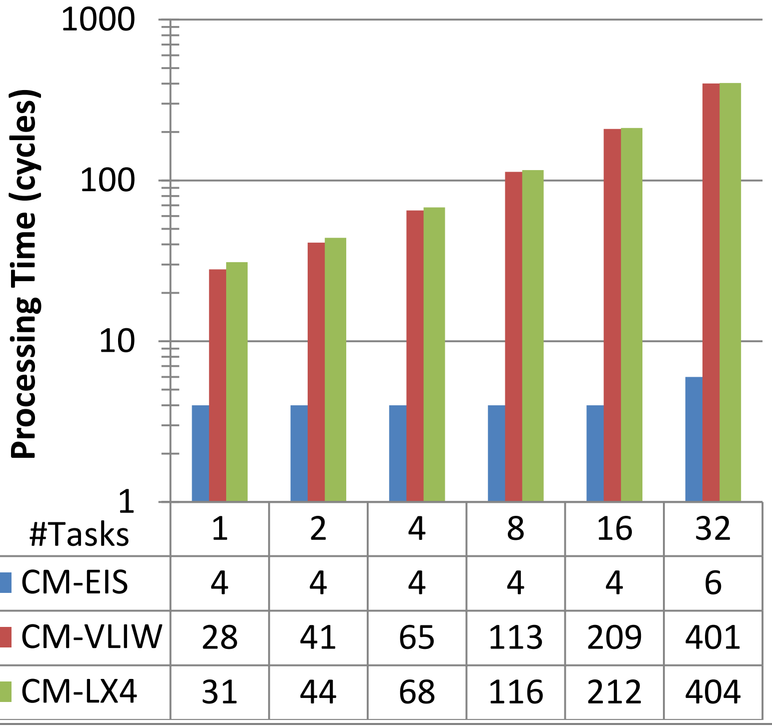
Task Scheduling



<Operator>

<slot X, valueX>

Deadline, Priority Level etc.

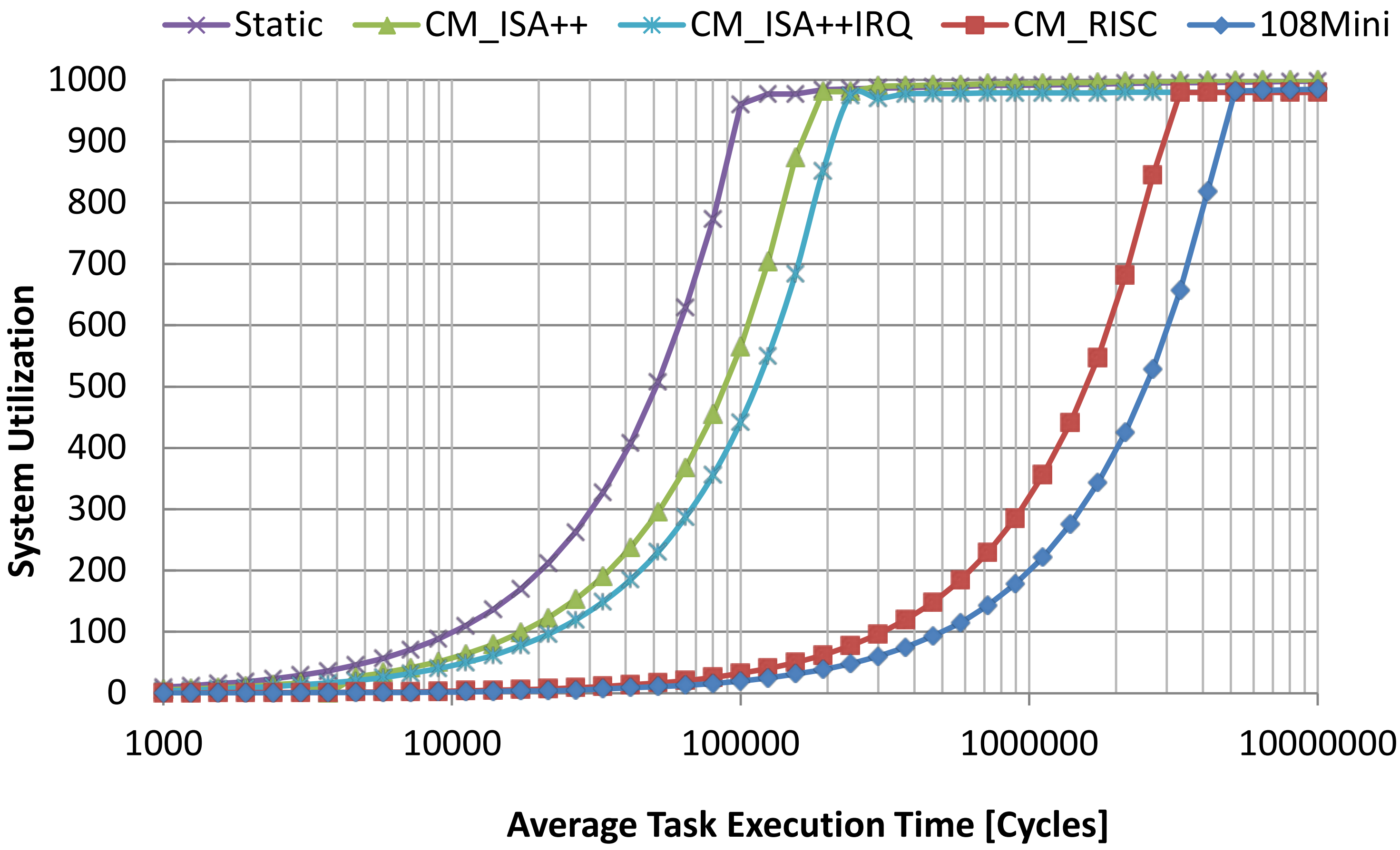


#Tasks	1	2	4	8	16	32
🟦 CM-EIS	4	4	4	4	4	6
🟥 CM-VLIW	28	41	65	113	209	401
🟩 CM-LX4	31	44	68	116	212	404

System Utilization

TECHNISCHE UNIVERSITÄT DRESDEN

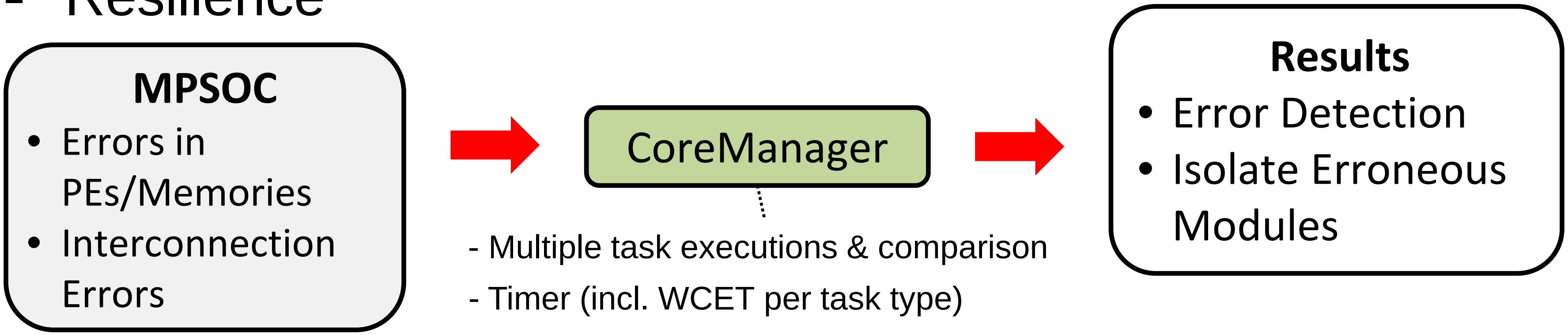
vodafone chair



$$\eta_S = \sum_{pe=0}^{\#PEs-1} \eta_{pe} \qquad \eta_{pe} = \frac{\sum_{task=0}^{\#Tasks(pe)-1} t_{pe,tasks,Task-End} - t_{pe,task,Task-Start}}{t_{System-End} - t_{System-Start}}$$

CoreManager Adaptions

- Resilience



MPSOC

- Errors in PEs/Memories
- Interconnection Errors

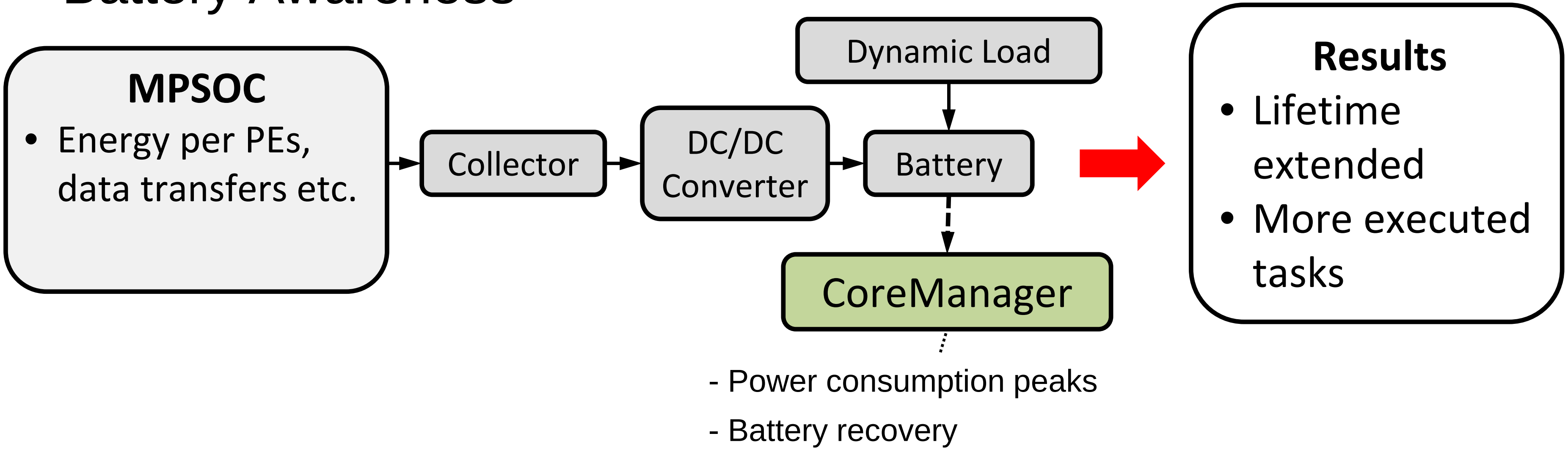
CoreManager

- Multiple task executions & comparison
- Timer (incl. WCET per task type)

Results

- Error Detection
- Isolate Erroneous Modules

- Battery Awareness



MPSOC

- Energy per PEs, data transfers etc.

Collector

DC/DC Converter

Dynamic Load

Battery

CoreManager

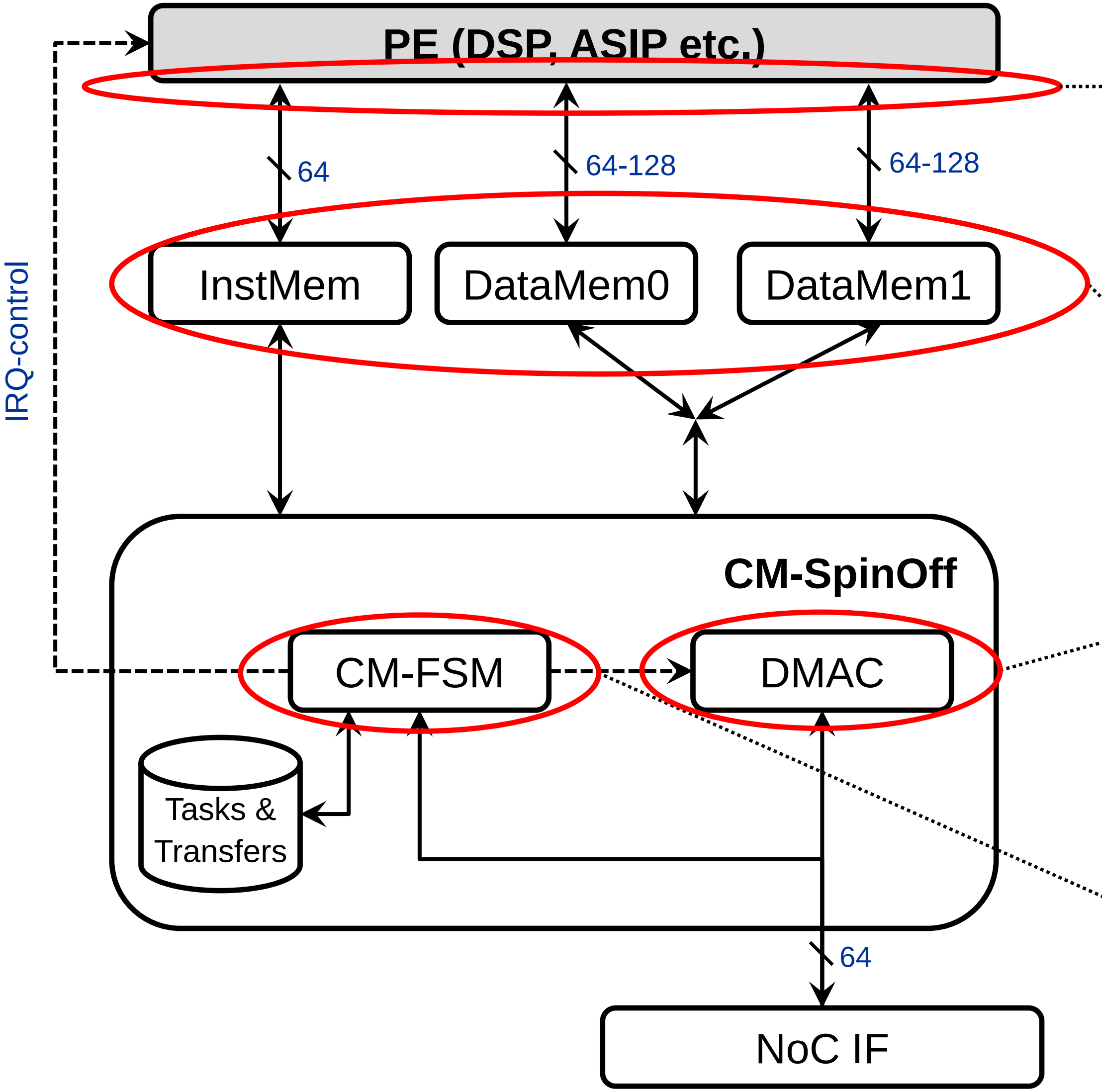
- Power consumption peaks
- Battery recovery

Results

- Lifetime extended
- More executed tasks

PROCESSING ELEMENT INTEGRATION

PE Integration Framework



- 1 Cycle latency!
  - ➔ no caches misses
  - ➔ deterministic execution time (WCET ☺)

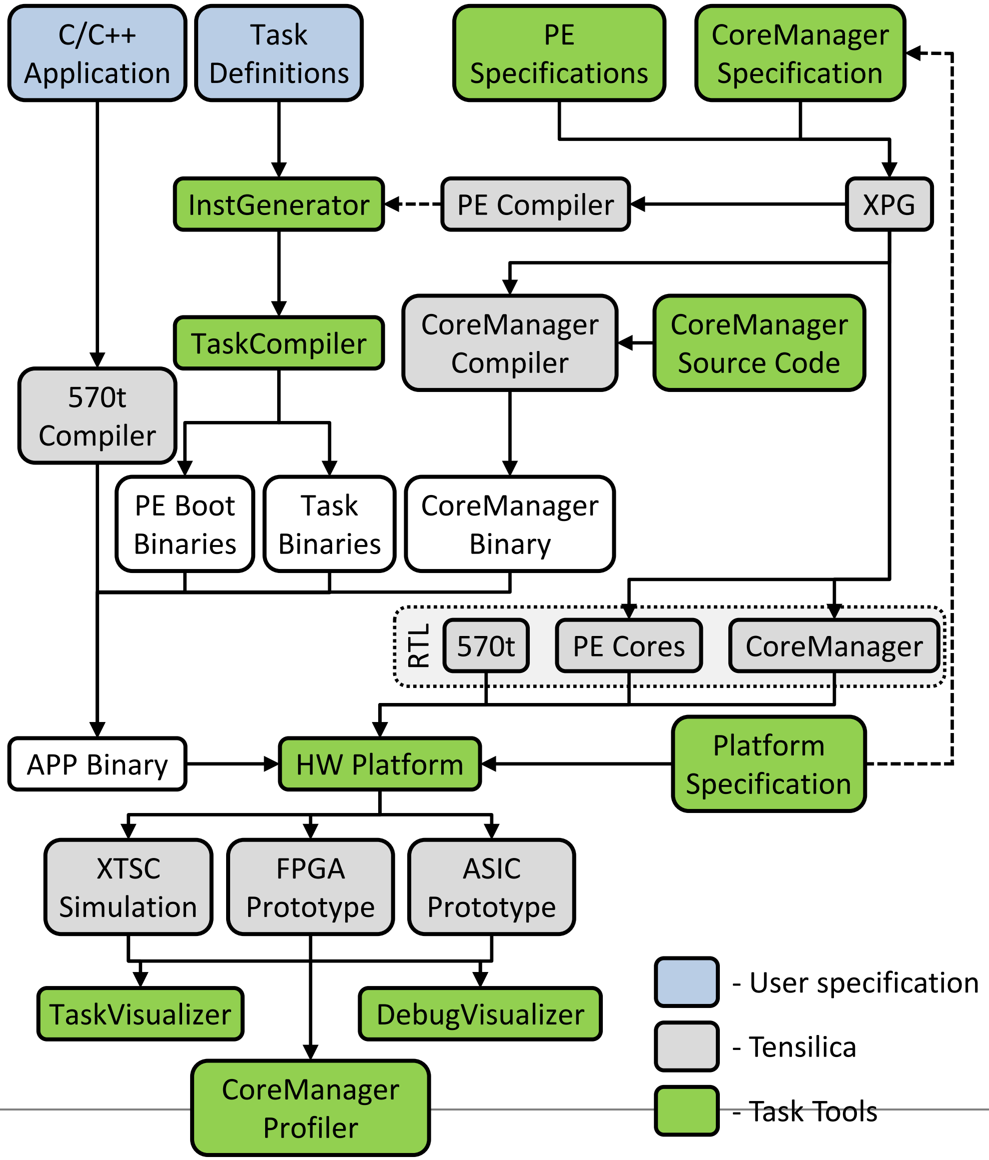
- Local Memories
- Dual-Port memories
- Optional: DataMem1

- Burst Mode
- Explicit data prefetch

- PE & DMAC Control
- Pre-defined by the CoreManager (event triggered)
- Task-FIFO for up to 4 tasks incl. data transfers

TOOLS

Basic Tool Flow

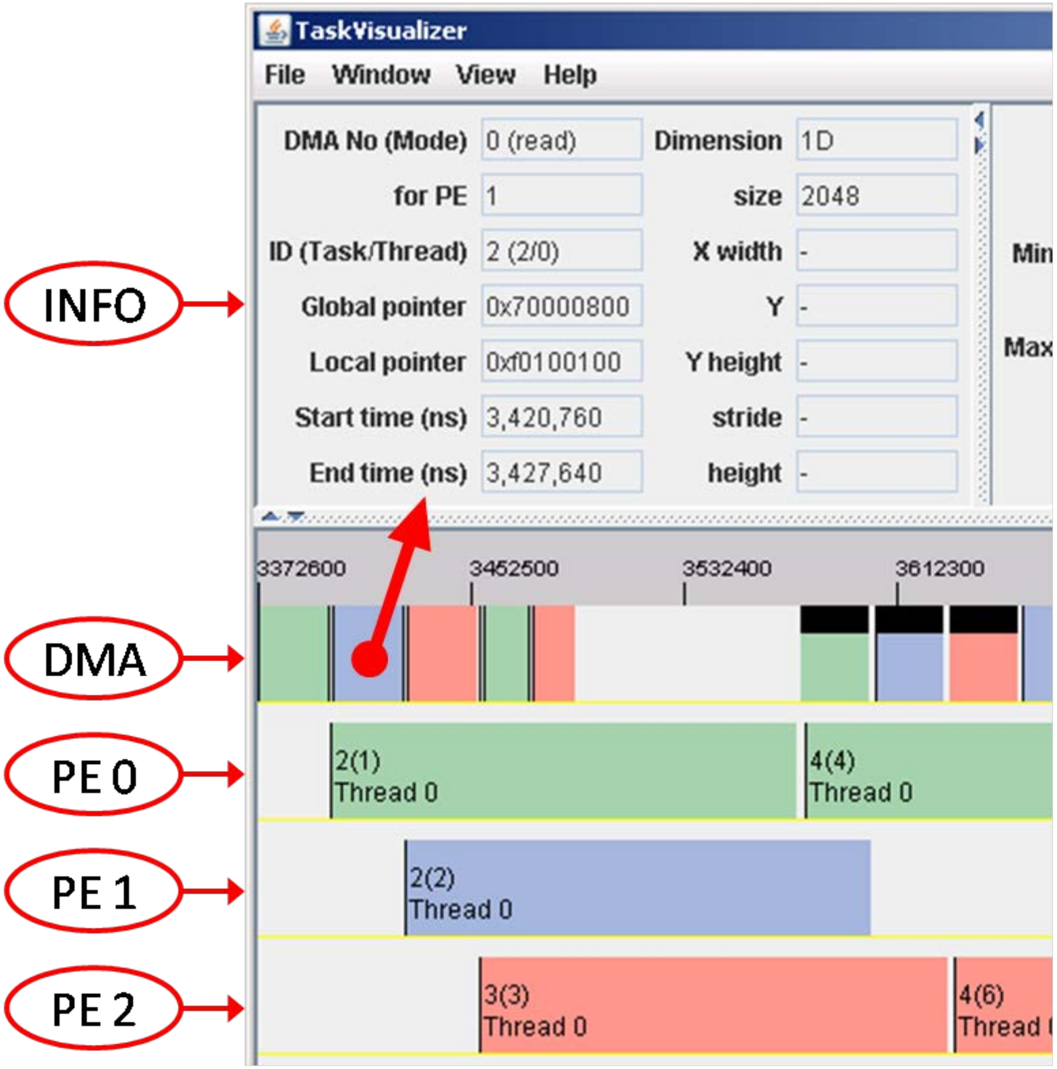


🟦 - User specification

⬜ - Tensilica

🟩 - Task Tools

TaskVisualizer

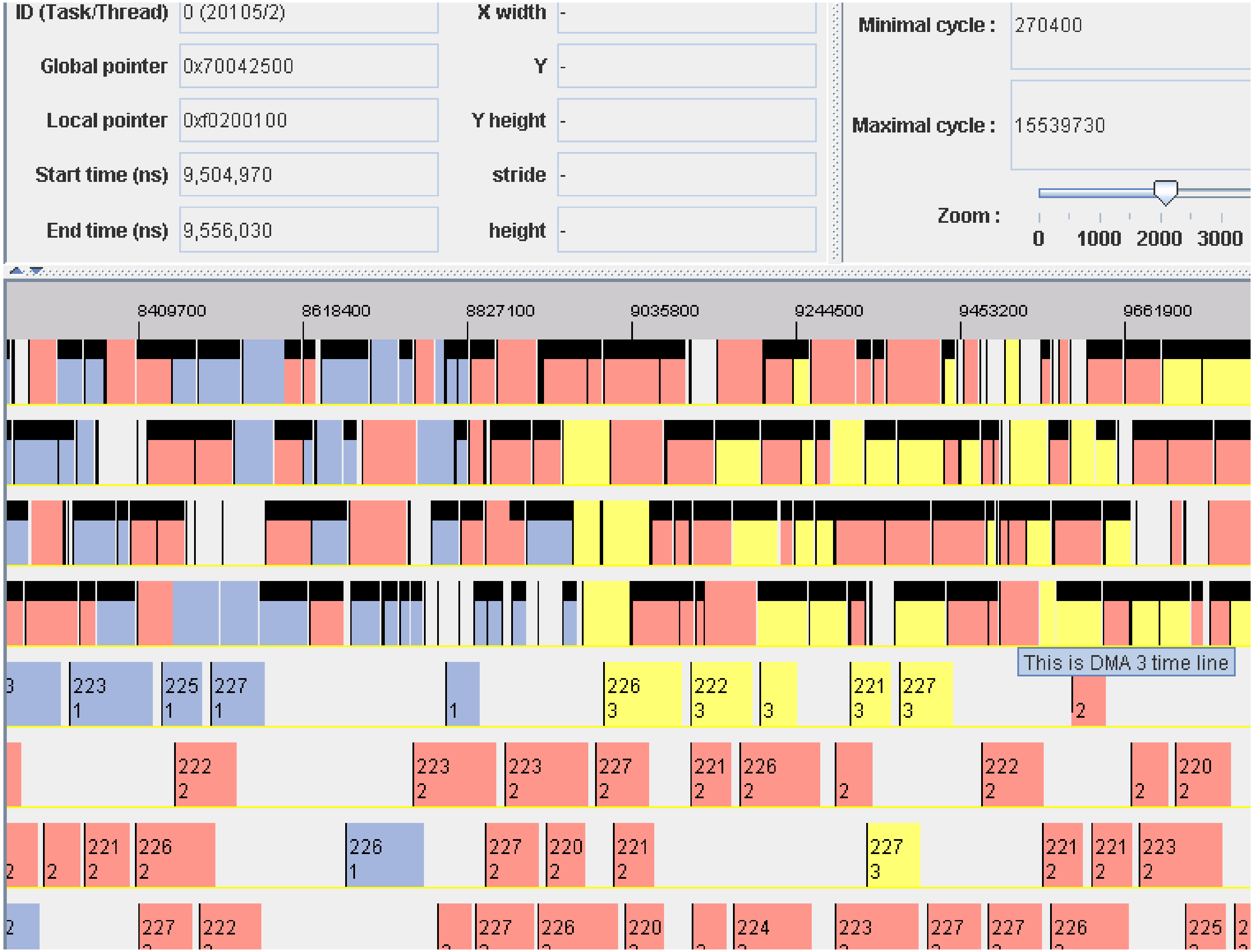


DMA No (Mode)	0 (read)	Dimension	1D
for PE	1	size	2048
ID (Task/Thread)	2 (2/0)	X width	-
Global pointer	0x70000800	Y	-
Local pointer	0xf0100100	Y height	-
Start time (ns)	3,420,760	stride	-
End time (ns)	3,427,640	height	-

Available Backends:

- TLM
- Cycle accurate Sim.
- FPGA
- Silicon prototypes

TaskVisualizer: Thread View



ID (Task/Thread)	0 (20105/2)	X width	-
Global pointer	0x70042500	Y	-
Local pointer	0xf0200100	Y height	-
Start time (ns)	9,504,970	stride	-
End time (ns)	9,556,030	height	-

Minimal cycle : 270400

Maximal cycle : 15539730

Zoom :

MPSoC Debugging ➔ Tomahawk PEs

- Off-the-shelf debugger ➔ single core debugger
- Processing elements
  - ❑ Explicit control of
    - Instructions
    - Input & Output data
  - ❑ Local address space
- No race conditions! (>1 thread is in the system)
- No deadlocks! (>1 thread is in the system)

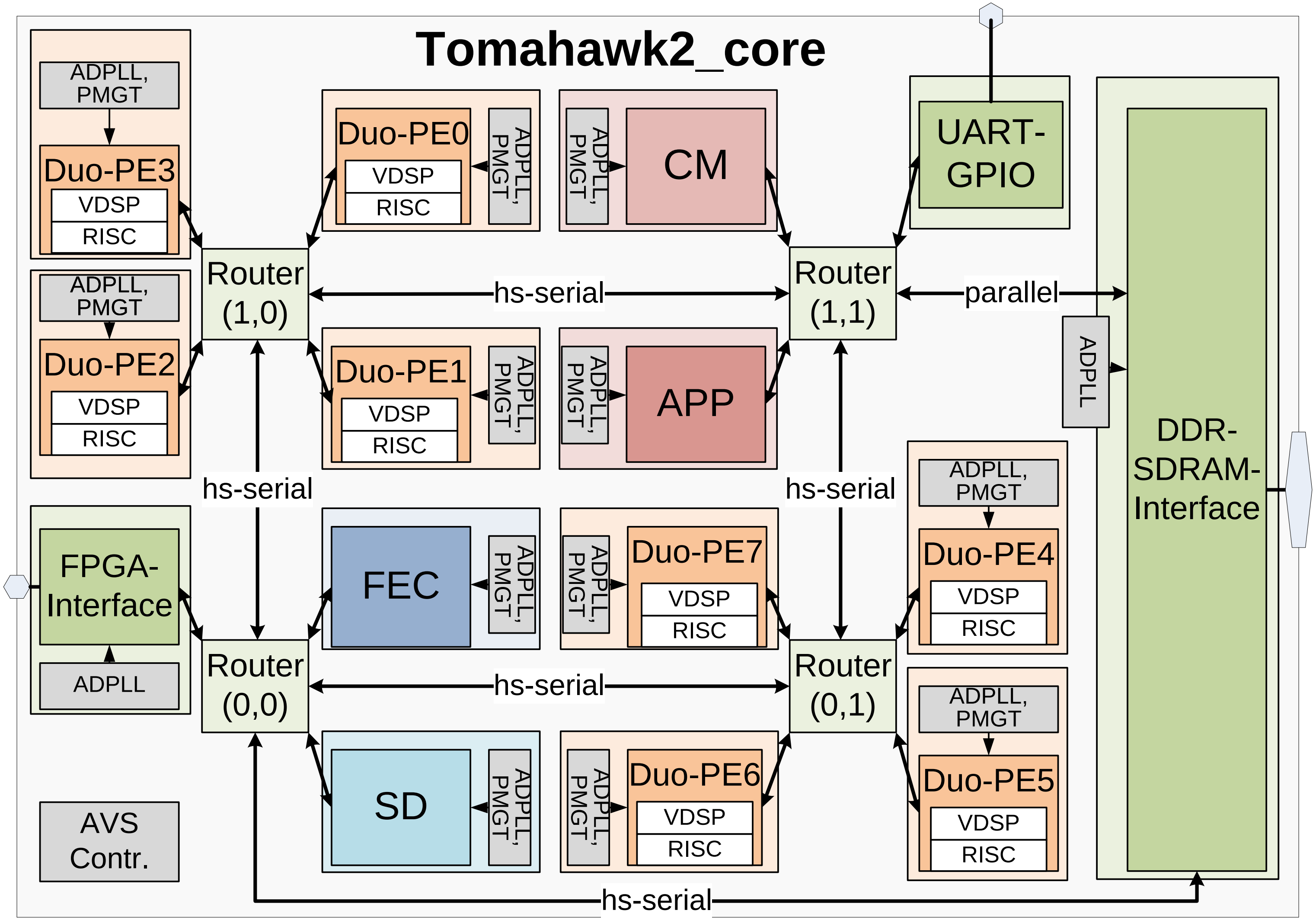
Task model: ➔ Less verification/validation effort
➔ Off-the-shelf debugger
➔ Faster application development

TOMAHAWK2 HETEROGONOUS MPSOC

Tomahawk2 Silicon Prototype

- Chip Facts
  - ❑ Globally asynchronous, locally synchronous heterogeneous 20 core MPSoC
  - ❑ Hierarchical packed switched Network-on-Chip (star-mesh topology)
  - ❑ Fine granular DVFS/AVFS on PE level
  - ❑ Voltage adaptable: between 0.7V-1.3V (typical: 1.2V)
- Dynamic Data plane control: **CoreManager**
  - ❑ Area: 1,36 mm^2 (incl. 0,87 mm^2 memory)
  - ❑ Max. frequency: > 500 MHz
  - ❑ Power Consumption: 14.1 mW @ 200 MHz, 0.9V
- ISSCC'14
  - ❑ *"A 105 GOPS 36mm^2 heterogeneous SDR MPSoC with energy-aware dynamic scheduling and iterative detection-decoding for 4G in 65nm CMOS"*
  - ❑ February 9-13 2013, San Fransisco
  - ❑ Paper presentation and Live-Demo

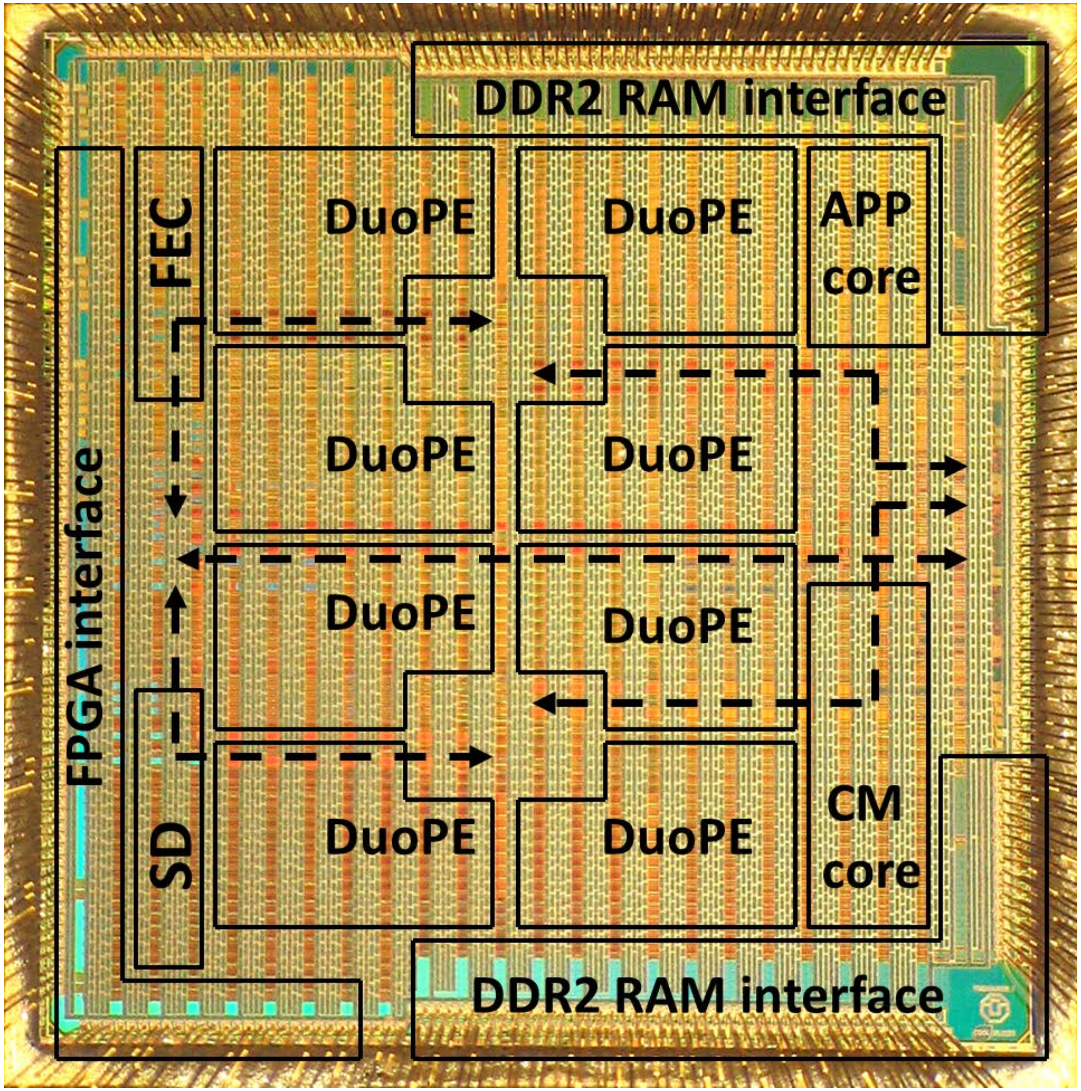
Block Diagram



Tomahawk2 Die Photo

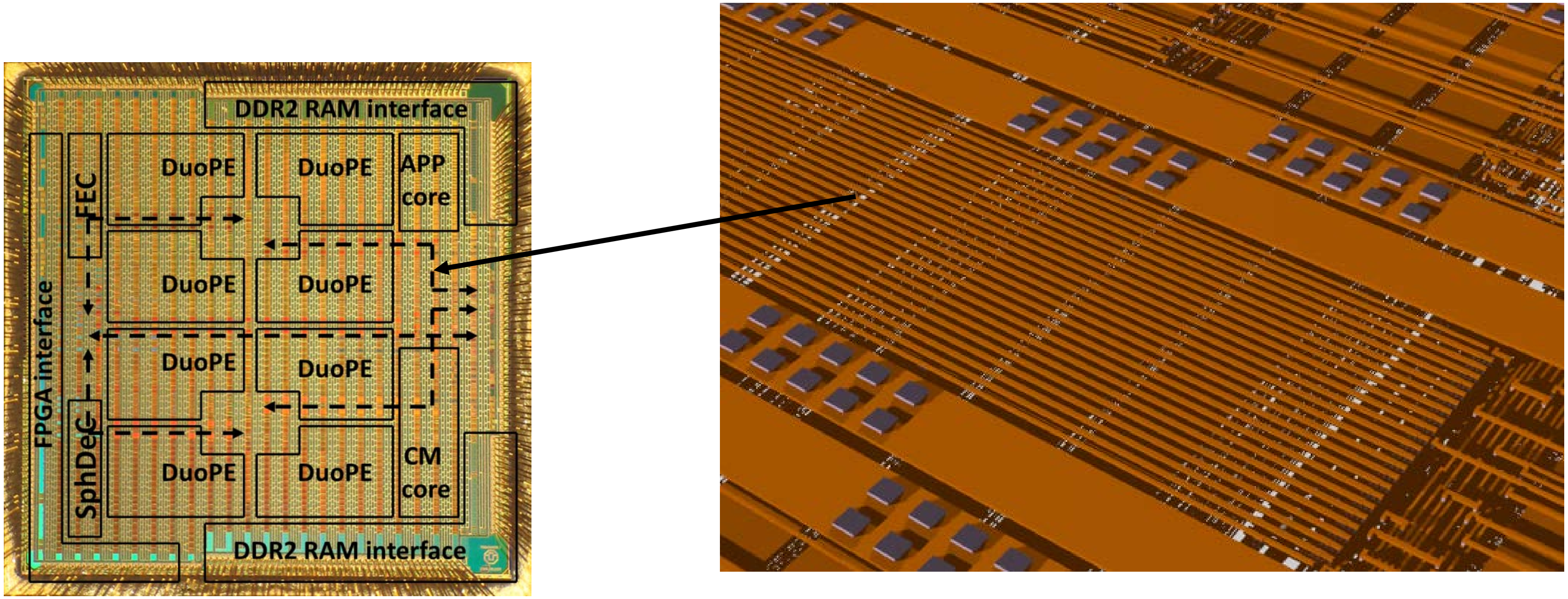
- 65 nm TSMC LP
- 6x6 mm^2
- 20 cores, 6 types

– – serial NoC link



High-speed Serial NoC Links

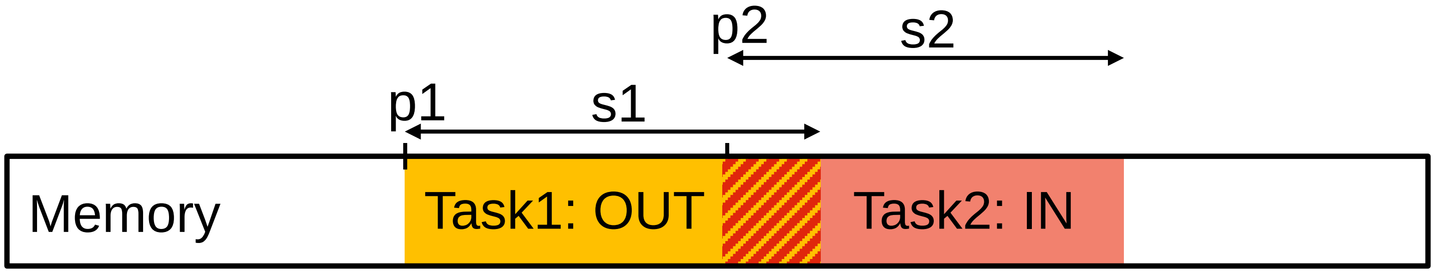
- Point-to-point connections within star-mesh NoC
- 80 Gbit/s (8GBit/s/lane) serial links [1] at <150fJ/Bit/mm
- Compact floorplan realization by bridging of core macros



D. Walter et al. , A Source-Synchronous 90Gb/s Capacitively Driven Serial On-Chip Link Over 6mm in 65nm CMOS, ISSCC 2012

Platform Control

- **CoreManager ➔ ASIP**
  - ❑ Central task scheduling unit
  - ❑ Data plane control (8xVDSP, 8xRISC, 1xFEC, 1xSD)
  - ❑ Scheduling-specific instruction set ➔ fully flexible in contrast to ASIC
  - ❑ Optional: dynamic data dependency checking



Tomahawk2 Measurements Results

- CoreManager: CM-TIE
- Local Memory
  - ❑ 64 KByte Data
  - ❑ 32 KByte Instruction

Area

Komponente	Fläche [mm²]	
CoreManager Core	0.269	**19.8%**
Memory	0.870	**64.0%**
CoreManager Transfer Unit	0.221	**16.2%**
All	1.360	

Power Consumption (CoreManager Power Domain)

Frequency/Voltage	Dep_Check_2_4 [mW]	PE_ALLOC [mW]	øS1 [mW]
200 MHz @ 0.90 V	15.7	14.5	14.8
286 MHz @ 1.00 V	28.4	25.9	26.3
333 MHz @ 1.08 V	38.5	36.2	36.8
445 MHz @ 1.25 V	74.6	68.0	68.9
500 MHz @ 1.32 V	91.7	81.5	83.4

2.5x

5.6x

CoreManager Comparison

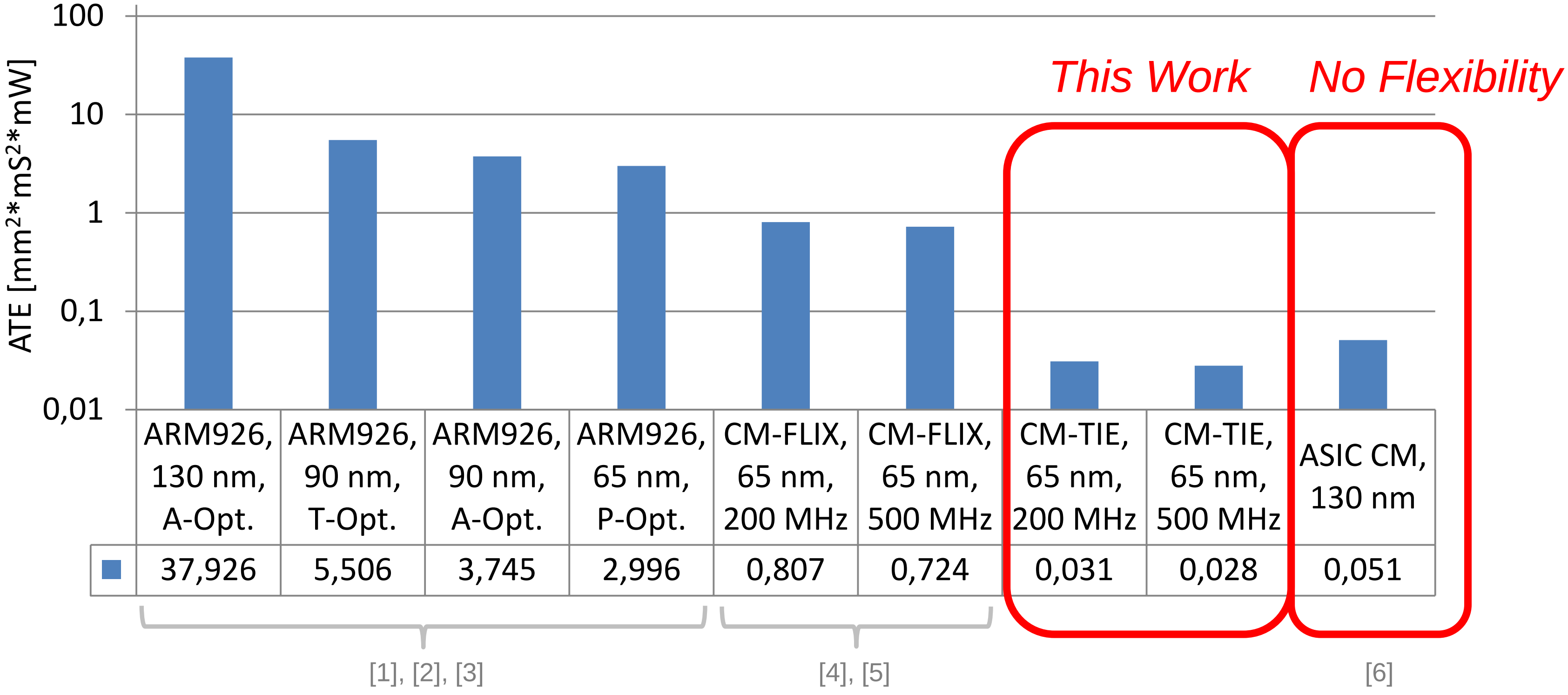
	ASIC [3]	RISC	ASIP (this work)
Scheduling Configurability	Fixed	Flexible	Flexible
Task Queue size	16	16-256	16-256
Max. Frequency [MHz]	175	445	445
Task scheduling [us]	0.4	16.9	0.9
Technology [nm]	130	65	65
Supply Voltage [V]	1.3	1.2	1.2
Power [mW] @fmax	282	68	74.6
Energy per Task [nJ] @fmax	113 (27*)	1149	67
Area (logic) [mm²]	4.51 (1.13*)	0.34	0.49
ATE product [mm²*us*nJ]	204 (12*)	6602	29

T. Limberg et al., A Fully Programmable 40 GOPS SDR Single Chip Baseband for LTE/WiMAX Terminals,
in Proceedings of the 34th European Solid-State Circuit Conference (ESSCIRC'08), Edinburgh, UK, 15.9. - 19.9.2008

CoreManager Comparison: ATE Product

$$C = A * T * E = A * T^2 * P$$

Scenario: 16 tasks, w/ dynamic data dependency checking



	ARM926, 130 nm, A-Opt.	ARM926, 90 nm, T-Opt.	ARM926, 90 nm, A-Opt.	ARM926, 65 nm, P-Opt.	CM-FLIX, 65 nm, 200 MHz	CM-FLIX, 65 nm, 500 MHz	CM-TIE, 65 nm, 200 MHz	CM-TIE, 65 nm, 500 MHz	ASIC CM, 130 nm
■	37,926	5,506	3,745	2,996	0,807	0,724	0,031	0,028	0,051

[1] http://www.arm.com/products/processors/classic/arm9/arm926.php/

[2] Keating, M. ; Flynn, D. ; Aitken, R. ; Shi, K.: Low power methodology manual: for system-on-chip design. Springer, 2007

[3] Same source code as in this work, simulated with Synopsys/VaST CoMET 6, -o3

[4] Same source code as in this work, simulated with Tensilica Xtensa Xplorer, -o3

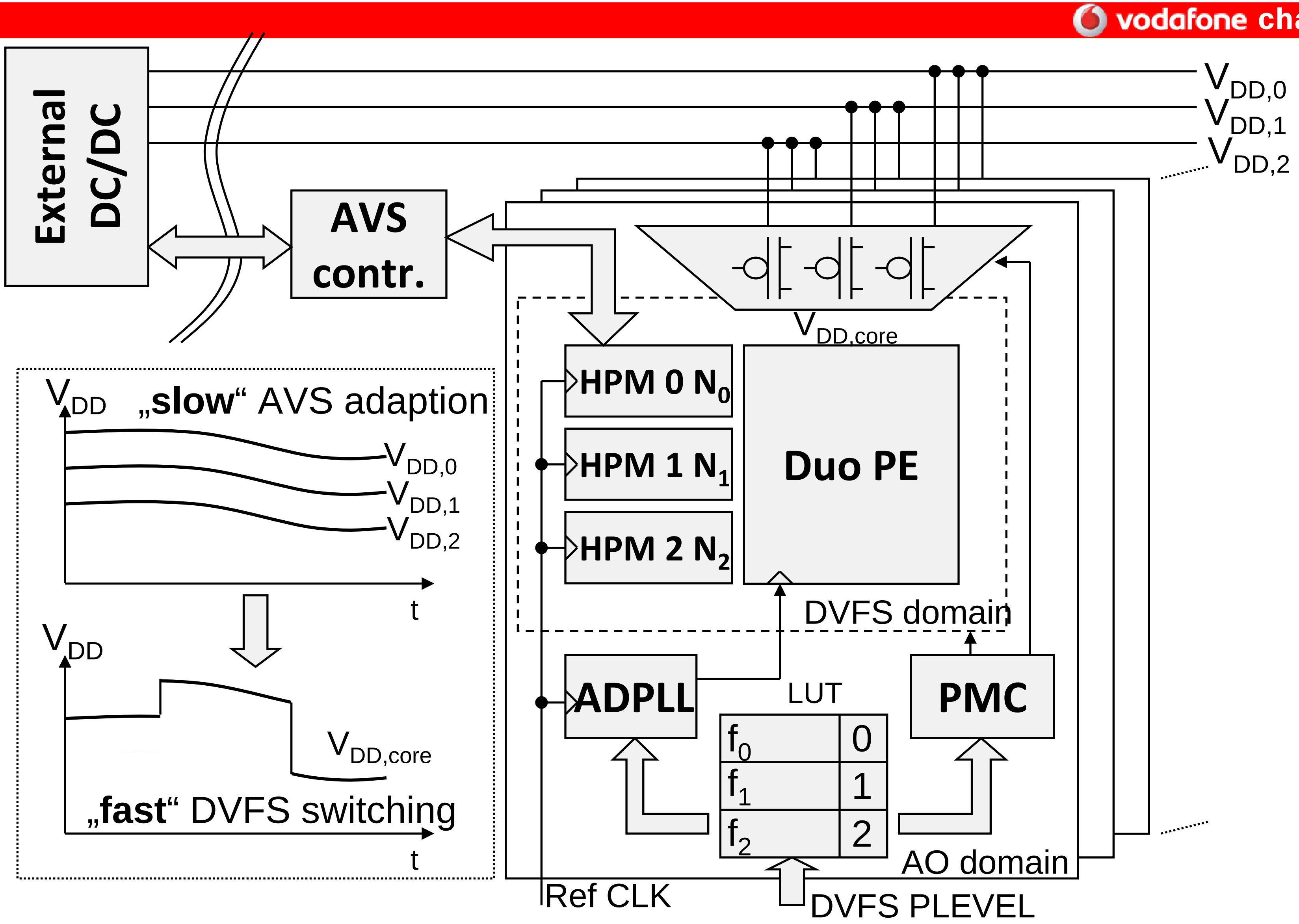
[5] Synopsys Design Compiler → Mentor Questa → Synopsys PrimeTime

[6] Limberg et. al: A fully programmable 40 gops sdr single chip baseband for lte/wimax terminals. In: *ESSCIRC 2008.*

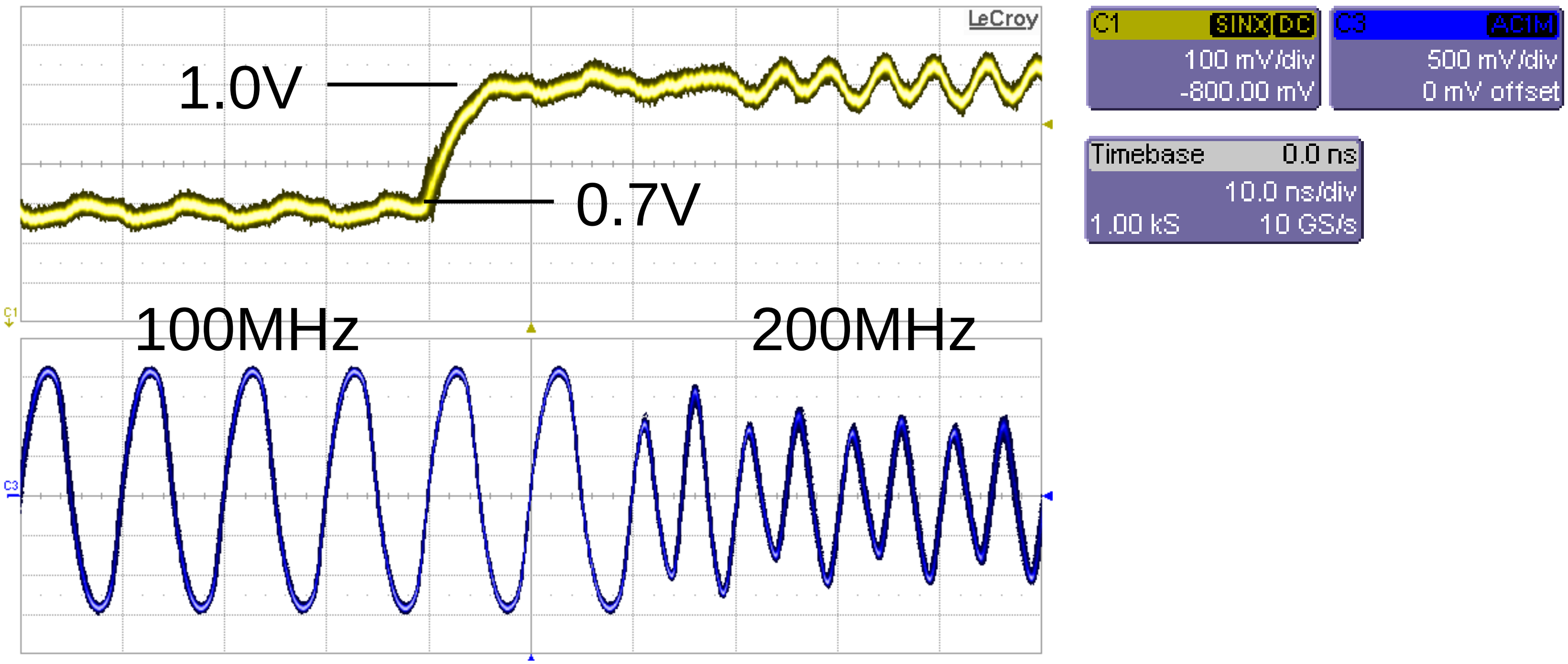
Power Management

- **Fine-grained: CoreManager**
  - Based on application requirements (e.g., start times and deadlines) and system status (e.g., allocated PEs)
  - Result: target frequency and voltage level for each PE for each task
  - Enabled by fine grained **fast** DVFS on PE level
- **Coarse-grained**
  - Global AVS ➔ voltage control of DVFS levels
  - Result: **slow** voltage adaption for temperature and process for DVFS rails

Power Management Architecture



Fast DVFS Measurement Result



C1 SINX DC
100 mV/div
-800.00 mV

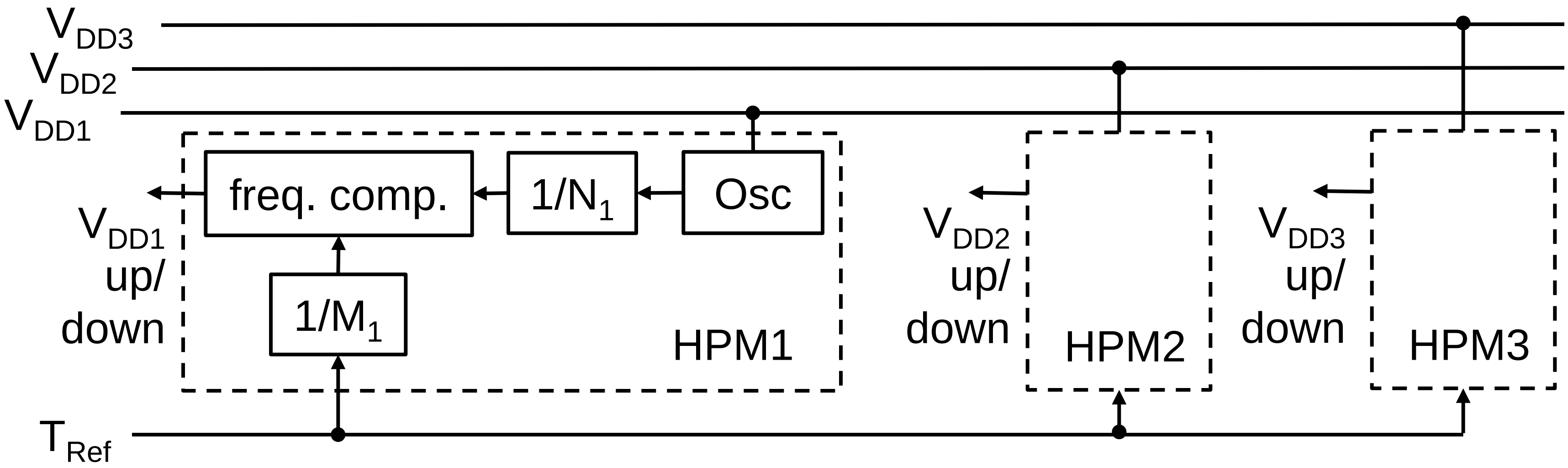
C3 AC1M
500 mV/div
0 mV offset

Timebase 0.0 ns
10.0 ns/div
1.00 kS 10 GS/s

➔ Ultra fast DVFS level change of Duo-PE in <20ns

3-Level DVFS with combined AVS

- 3 performance levels (V_{DD1},f_1) (V_{DD2},f_2), (V_{DD3},f_3)
  - **Fast** DVFS switching between rails
- Hardware Performance Monitor (HPM) with virtual frequency multiplication capability
  - Configurable ring oscillator replicates critical timing
  - **Slow** AVS adaption of V_{DD} such that $f_{Osc,i}=f_{Ref}\cdot N_i/M_i=f_i$

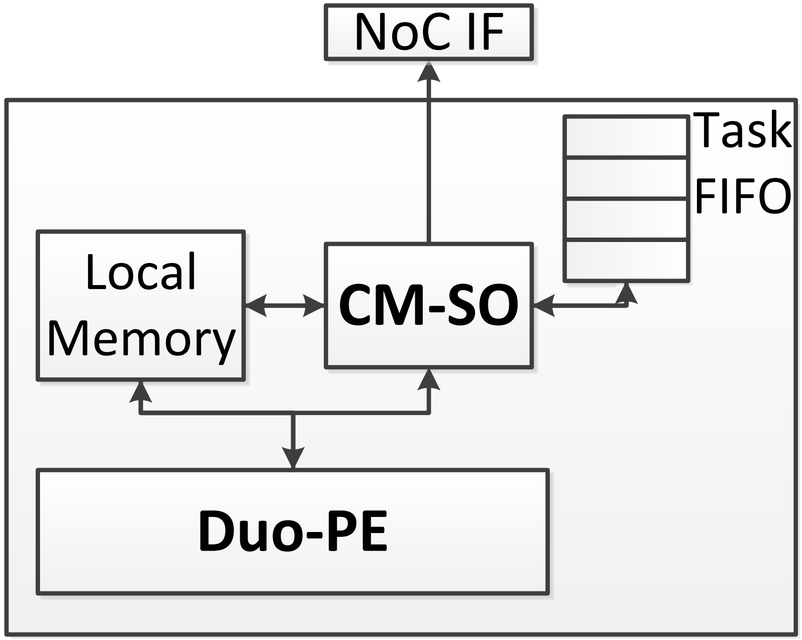


Duo-PE Concept

- Motivation:
  - Data transfer among heterogeneous cores increases power consumption, network congestions and latencies
- Heterogeneous cores connected to local Memory:
  - 4-fold SIMD Vector DSP → 16-bit fixed point comp.
  - General Purpose RISC Core → high precision FP comp.
- Advantage:
  - Further increase of data locality and area efficiency
  - Dynamic core selection during runtime
  - Use of existing compilers

Duo-PE

- **Duo-PE controlled by the CM-Spinoff**
  - DMA controller programmed by task descriptions
  - Allows concurrent data prefetching during task execution
  - Activates the clock for the requested core



Tomahawk2 Components

		Area [mm2]		f_{max} [MHz] @VDD=1.2 V	Throughput $@f_{max}$	$P_{Application-Scenario}$* [mW]
		total	*mem*			
APP		0.582	0.245	445	890 MOPS	off
CM		1.360	0.870	445	1.1 MTasks/s	14.1 @200MHz, 0.9 V
Duo-PE	RISC	1.357	0.800	445	890 MOPS	off
	VDSP			500	10 GOPS	35.0 @282 MHz, 0.9 V
SD		0.522	0.260	445	396 Mb/s	36.5 @200 MHz, 1.15 V
FEC		1.154	0.618	500	155 Mb/s	132.2 @200 MHz, 1.15 V
FPGA-IF		0.602	-	500	10 Gb/s	-
DDR-IF		4.552	-	400	12.8 Gb/s	-
NoC		3.417	-	500	80 Gb/s/link	18.0 @286 MHz, 1.15 V

*4×4 MIMO 3GPP-LTE baseband application

Demonstrator

Summary

- Tomahawk Archticture Framework
  - ❑ TaskC programming model
  - ❑ CoreManager (dynamic task scheduling)
  - ❑ PE integration
  - ❑ Network-on-Chip
- Tomahawk2
  - ❑ Heterogeneous MPSoC with dynamic task-scheduling
  - ❑ Several types of PEs: VDSPs, GP-cores and ASIPs
  - ❑ Data-plane control: CoreManager with scheduling-specific instruction set
  - ❑ Star-Mesh NoC with serial links
  - ❑ Dynamic and Adaptive Power Management

Future Work

- Further optimizations of the architecture and the algorithms ➔ especially for signal processing
- CoreManager for >1000 cores
- Next silicon prototypes:
  - ❑ 4/2014 - test chip (28 nm Globalfoundries)
  - ❑ 10/2014 - small 5-core MPSoC (28 nm Globalfoundries)
  - ❑ 3/2015 - MPSoC (28 nm Globalfoundries)

References

[1] O. Arnold, E. Matus, B. Nöthen, M. Winter, T. Limberg and G. Fettweis, “**Tomahawk - Parallelism and Heterogeneity in Communications Signal Processing MPSoCs**”, ACM Transactions on Embedded Computing Systems (TECS), Volume 13, 2014.

[2] O. Arnold and G. Fettweis, "**Power Aware Heterogeneous MPSoC with Dynamic Task Scheduling and Increased Data Locality for Multiple Applications**," SAMOS’10, July 2010, pp.110-117.

[3] O. Arnold and G. Fettweis, “On the Impact of Dynamic Task Scheduling in Heterogeneous MPSoCs”, SAMOS'11, July 2011, pp. 17-24.

[4] O. Arnold, B. Nöthen, and G. Fettweis, **“Instruction Set Architecture Extensions for a Dynamic Task Scheduling Unit”** in Proceedings of the IEEE Annual Symposium on VLSI (ISVLSI'12), 2012, pp. 249-254.

[5] O. Arnold, B. Noethen, and G. Fettweis, “**A Flexible Analytic Model for a Dynamic Task-Scheduling Unit for Heterogeneous MPSoCs**,” SIMUL’13, Venice, Italy, 2013.

[6] O. Arnold and G. Fettweis, “**Resilient dynamic task scheduling for unreliable heterogeneous MPSoCs**”, IEEE SCD, 2011, pp. 1-4.

[7] O. Arnold and G. Fettweis, “**Self-aware heterogeneous MPSoC with dynamic task scheduling for battery lifetime extension**”, ASPLOS’11 Workshop: CHANGE, pp. 1-7, March 2011.

[8] B. Noethen et al., “**A 105GOPS 36mm 2 heterogeneous SDR MPSoC with energy-aware dynamic scheduling and iterative detection-decoding for 4G in 65nm CMOS**”, Solid-State Circuits Conference (ISSCC’14), San Francisco, USA, pp. 188 - 189, 2014.

[9] O. Arnold, B. Nöthen, and G. Fettweis, “**CM_ISA++: An Instruction Set for Dynamic Task Scheduling Units for More Than 1000 Cores**”, IEEE SOC Conference (SOCC'14), Las Vegas, USA, September 2014.

THANK YOU