ICTOPEN

# Liquid Computing

Stephan Wong
Computer Engineering Lab
Faculty of EEMCS, TU Delft

Netherlands Organisation for Scientific Research
Physical Sciences

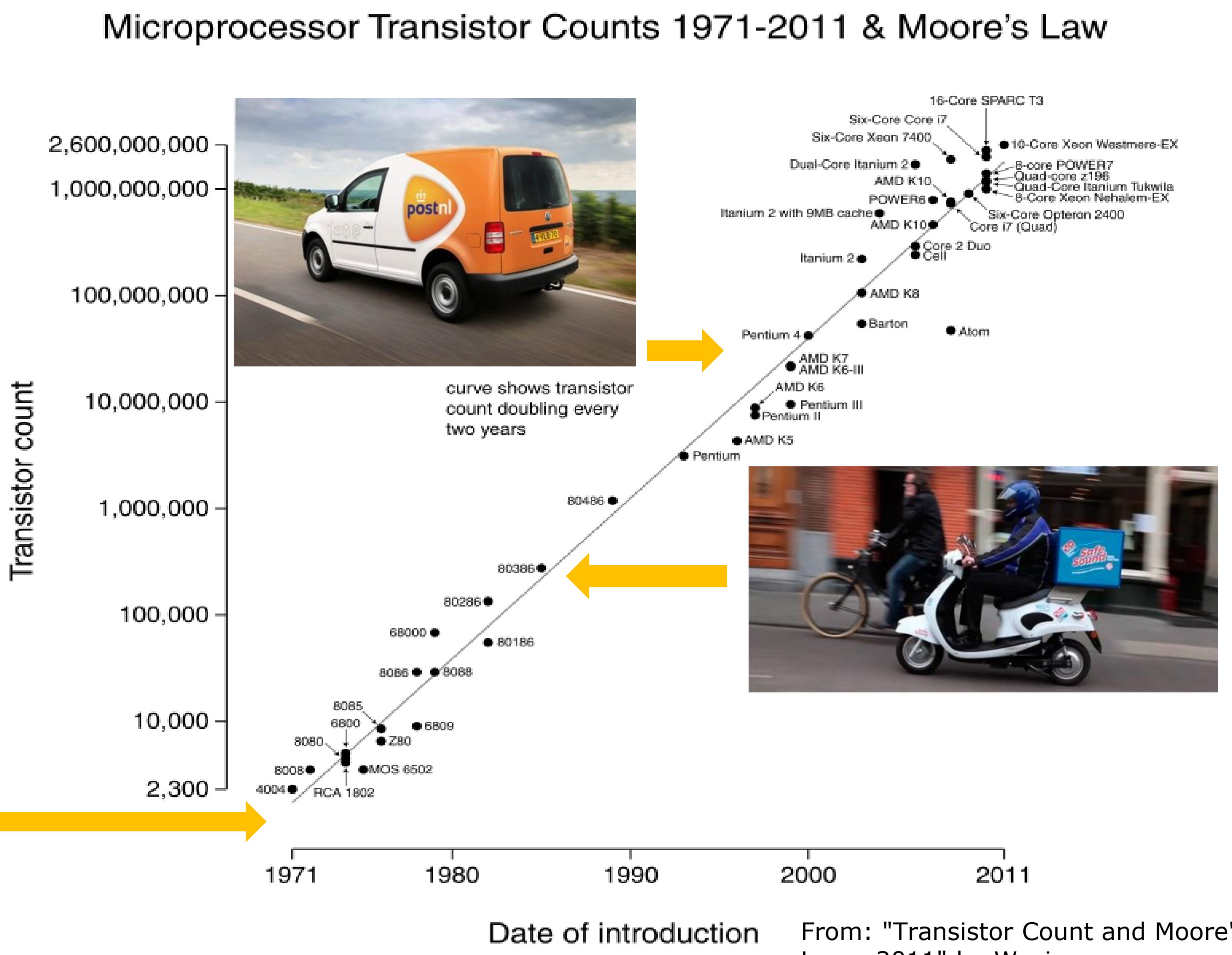
ICT-onderzoek Platform Nederland

Enabling new technology

## Analogy: Package Delivery vs. Processor Design

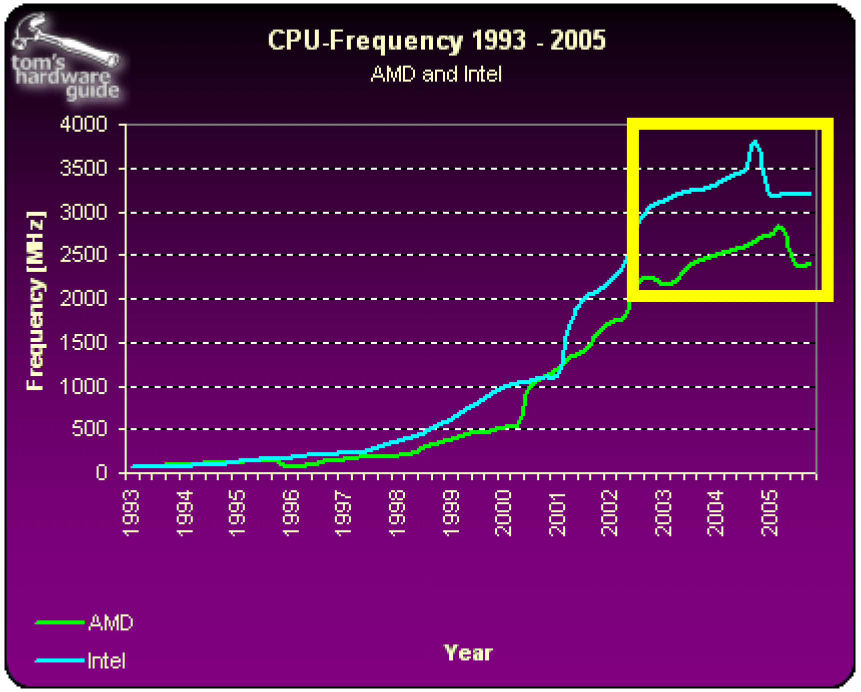
*Use the increasing amount of "transistors" to build better package delivery systems:*

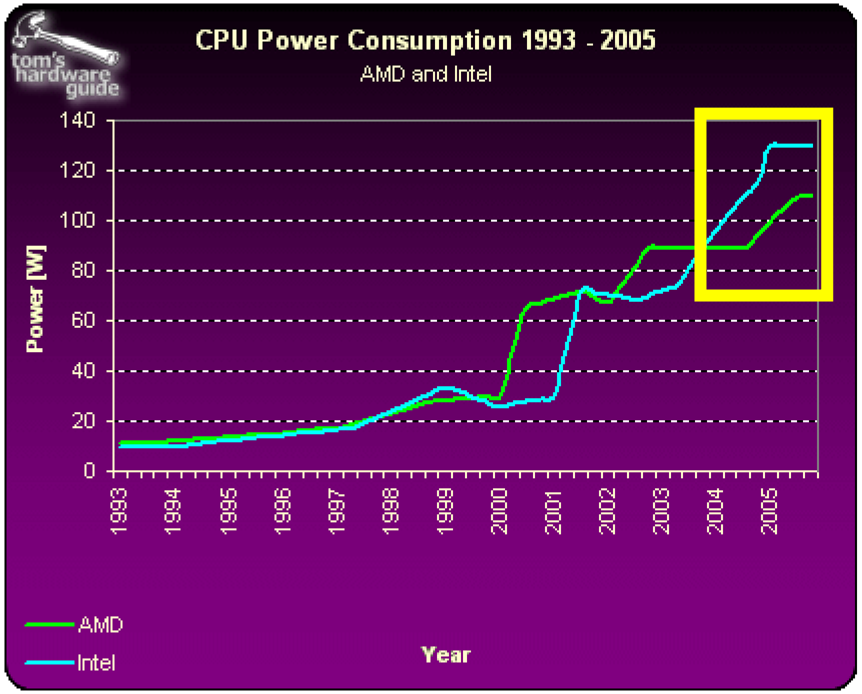
- *Larger packages*
- *Faster delivery*
- *More energy-efficient*



From: "Transistor Count and Moore's Law - 2011" by Wgsimon

## Around 2005: Frequency & Power leveling off





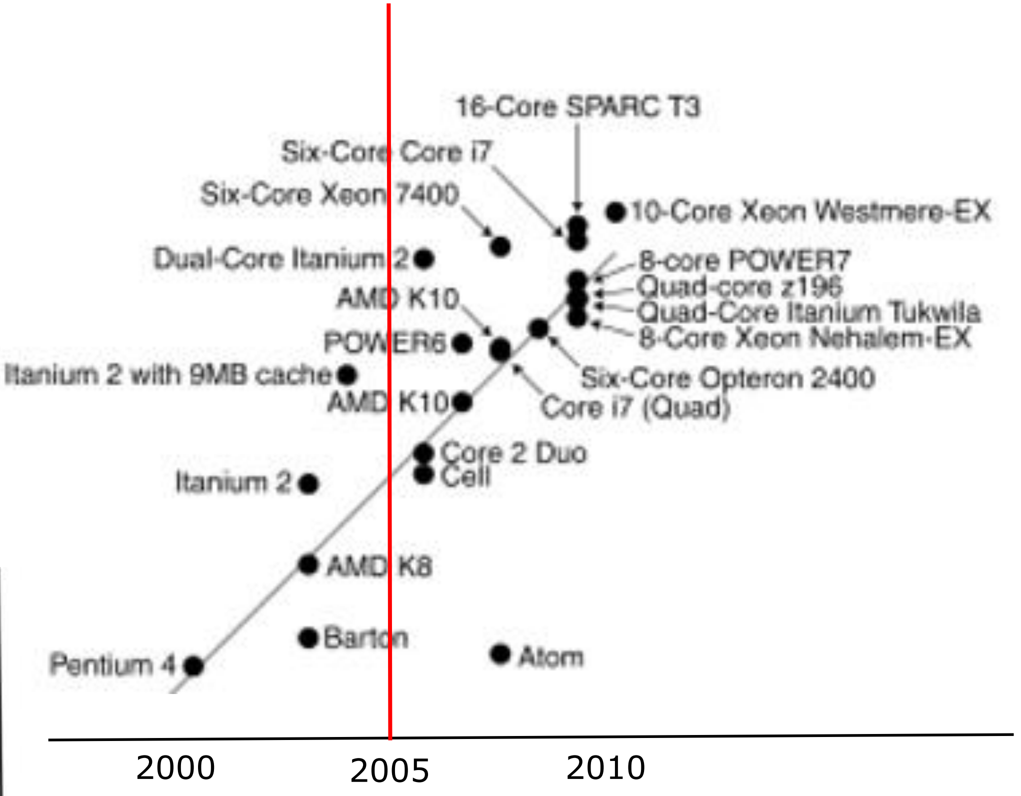
From: www.tomshardware.com

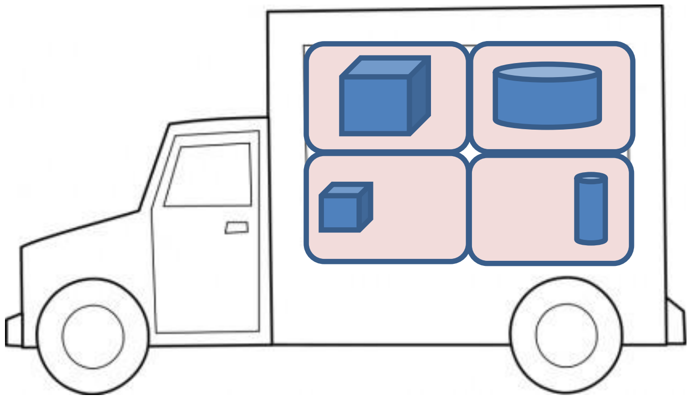
- Dennard Scaling (power density remains constant) ended 2005-2007
- However, Moore's Law (#transistors doubles every ~2 yrs) continued
- What was the effect??

## Homogeneous Computing

Terminology (after 2005):

- Dual-core
- Quad-core
- Six-core
- 8-core
- 10-core
- 16-core
- "Just" more the same core

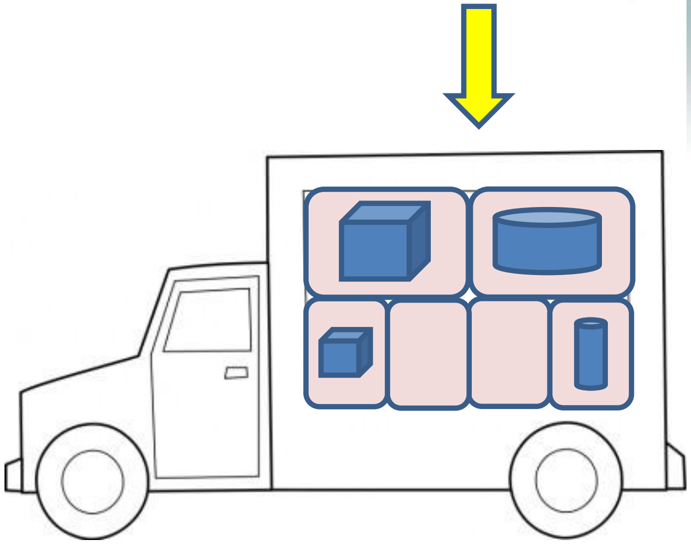


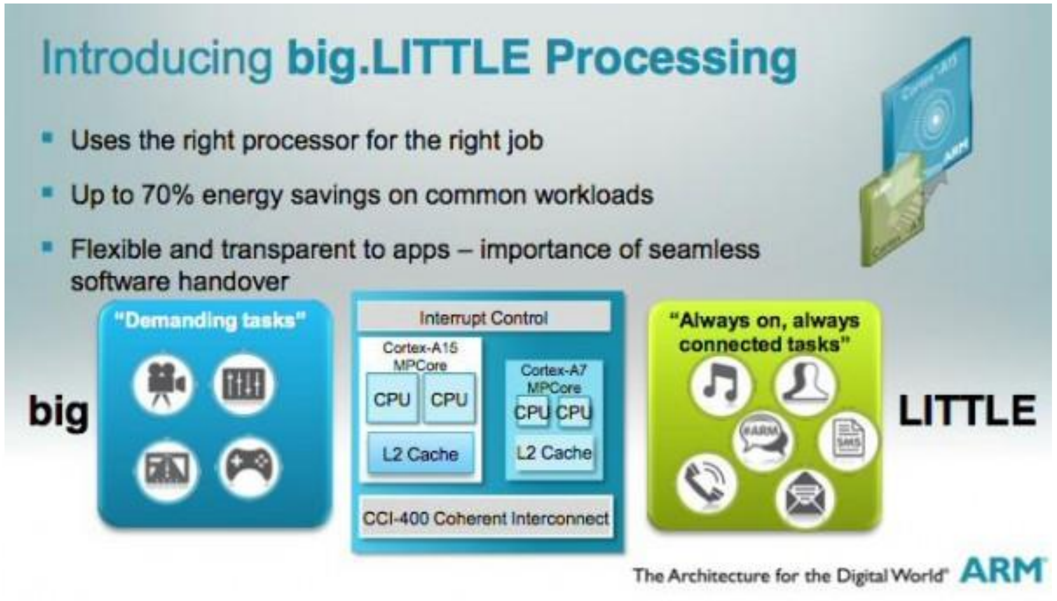


Analogy: Use the same box (=processor) to transport various sizes of packages (=applications)

## Heterogeneous Computing

- Example: in 2011 ARM Introduces big.LITTLE





Introducing **big.LITTLE Processing**

- Uses the right processor for the right job
- Up to 70% energy savings on common workloads
- Flexible and transparent to apps – importance of seamless software handover

big
"Demanding tasks"
Interrupt Control
Cortex-A15 MPCore
CPU
CPU
L2 Cache
Cortex-A7 MPCore
CPU
CPU
L2 Cache
CCI-400 Coherent Interconnect
"Always on, always connected tasks"
LITTLE
The Architecture for the Digital World ARM

From: www.arm.com

Analogy: Use the different boxes (=heterogeneous processors) to transport various sizes of packages (=applications)

## Now What?

- How can we even more efficiently use the transistors, minimize waste, and reduce energy consumption?

- → Liquid Computing

- First two intermezzo's before I give definition

# Intermezzo: 3D Printing

**MakerBot Replicator**

Desktop 3D Printer

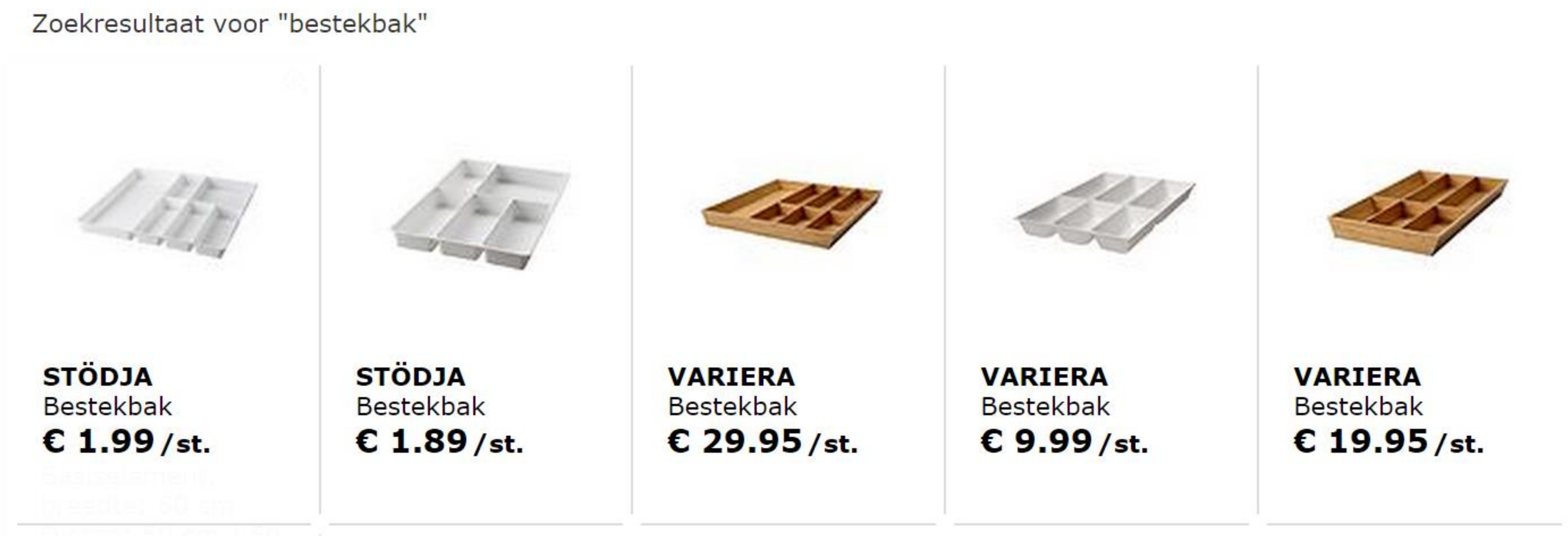
- Easy to use use with simple connectivity for all your 3D printing needs
- Makes true-to-life objects quickly and easily
- Fun and engaging to use, putting modeling and 3D design in your students' hands

Cityscape by MakerBot

From: http://www.makerbot.com/uses/for-professionals

- Continuing analogy → build the *best container* for each package
- This means: build the *best processor* for your application
- → Field-Programmable Gate Arrays (FPGAs) is now best candidate

## Intermezzo: IKEA



Zoekresultaat voor "bestekbak"

| | | | | |
|---|---|---|---|---|
| **STÖDJA** Bestekbak **€ 1.99** /st. | **STÖDJA** Bestekbak **€ 1.89** /st. | **VARIERA** Bestekbak **€ 29.95** /st. | **VARIERA** Bestekbak **€ 9.99** /st. | **VARIERA** Bestekbak **€ 19.95** /st. |

From: www.ikea.com

- Continuing analogy: reconfigure a kitchen drawer (=reconfigurable processor) for different kitchen utensils (=applications)
- → Parameterized reconfigurable processors
- *(NOTE: One single design and not necessarily using FPGAs)*

## Liquid Computing: A definition

- Run-time adaptivity of computing systems (processors, memories, network-on-chips) to meet changing requirements of applications being executed in different environments
- Analogy: Versatile and flexible package delivery system that can cope with any type and size of packages to be transported in all (weather) conditions at any time

- The predecessor to LC was the ERA project

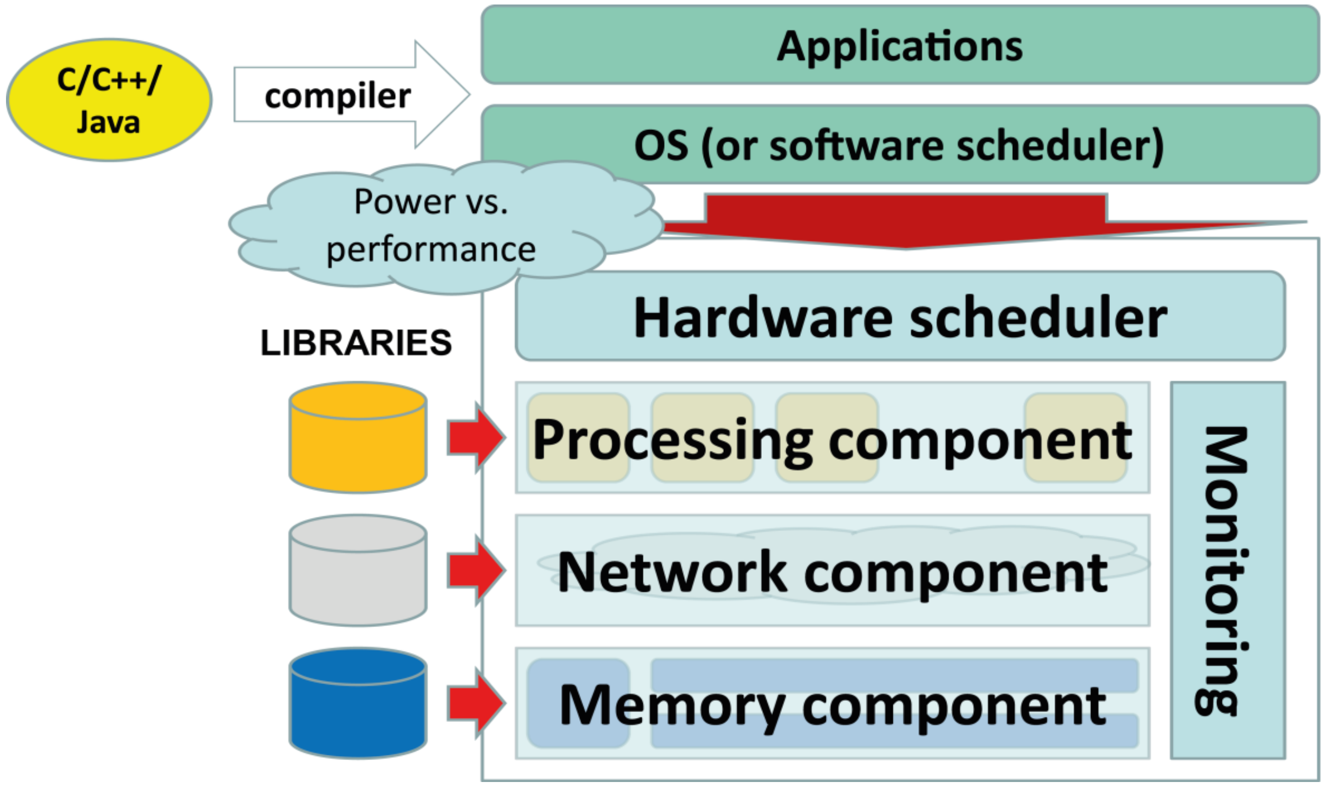
ERA

# Embedded Reconfigurable Architectures

**Dynamic adaptation** to *software requirements & operating environment*

**Dynamic adaptation** in *performance, power / energy, and resources*

**Dynamic reconfiguration** of *processor cores, caches, and NoCs*



**Partners**

**Technische Universiteit Delft** *(TUD) – NL (Coordinator)*

**Industrial Systems Institute** *(ISI) – GR*

**Universita' degli Studi di Siena** *(UNISI) – IT*

**Chalmers University** *(CHALMERS) – SE*

**University of Edinburgh** *(UEDIN) – UK*

**Evidence** *(EVI) – IT*

**STMICRO** *(STMICRO) – IT*

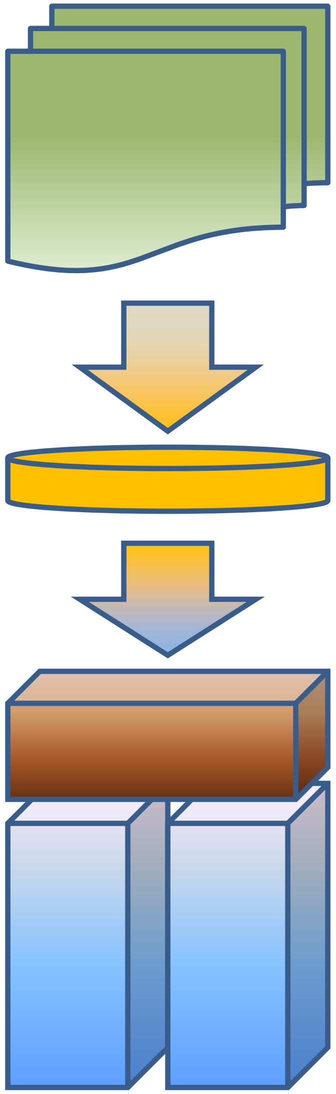
**IBM** *(IBM) – IL*

**Universidade do Rio Grande do Sul** *(UFRGS) – BR*

**Uppsala University** *(UU) – SE*

Contract: **INFSO-ICT-249059**
EU Funding: **2.8 MEuro**
Start - End: **01-2010 – 03-2013**

## Mainstream processors



**Programs:**

- General-purpose programs (think: desktop, office)
- Domain-specific programs (think: embedded)
- Different characteristics (e.g., parallelism)

Will programs run efficiently on the processor in most cases?

- **for general-purpose computing: YES**
- **for domain-specific computing: NO**

Why not?

- fixed nature of processor - not tuned for applications

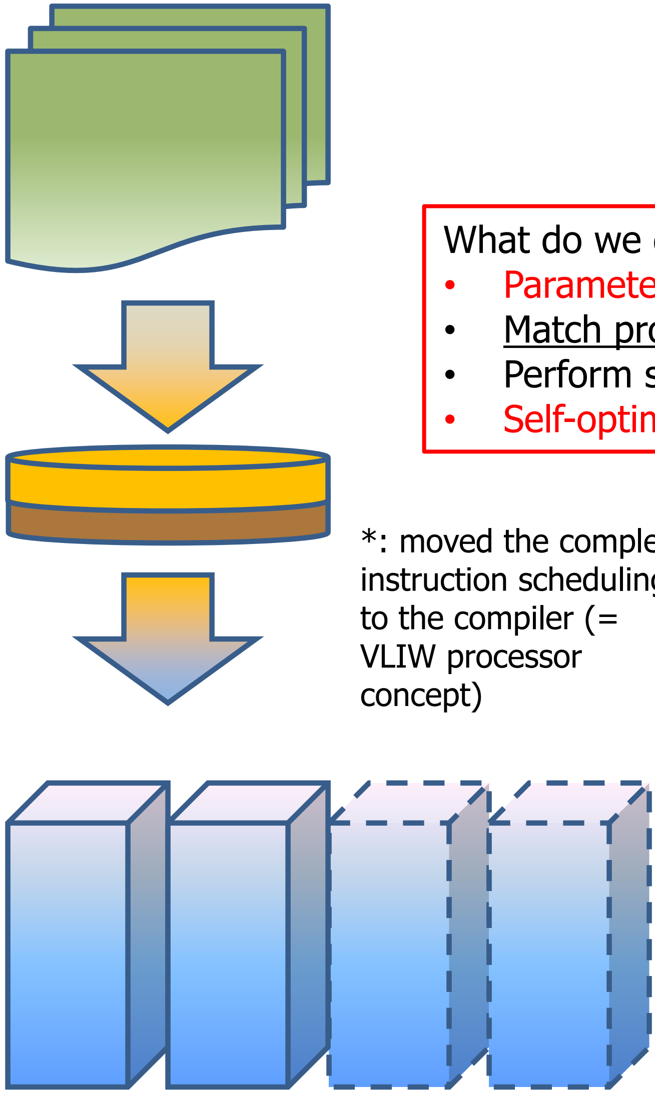
**Compiler:**

- Targets fixed processor
- Match characteristics with processor capabilities

**Single processor:**

- General-purpose
- Fixed (parallel) functionality
- Complex hardware to fully utilize parallel hardware (= power hungry)

## Embedded Reconf. Arch.



**Programs:**
- General-purpose programs (think: desktop, office)
- Domain-specific programs (think: embedded)
- Different characteristics (e.g., parallelism)

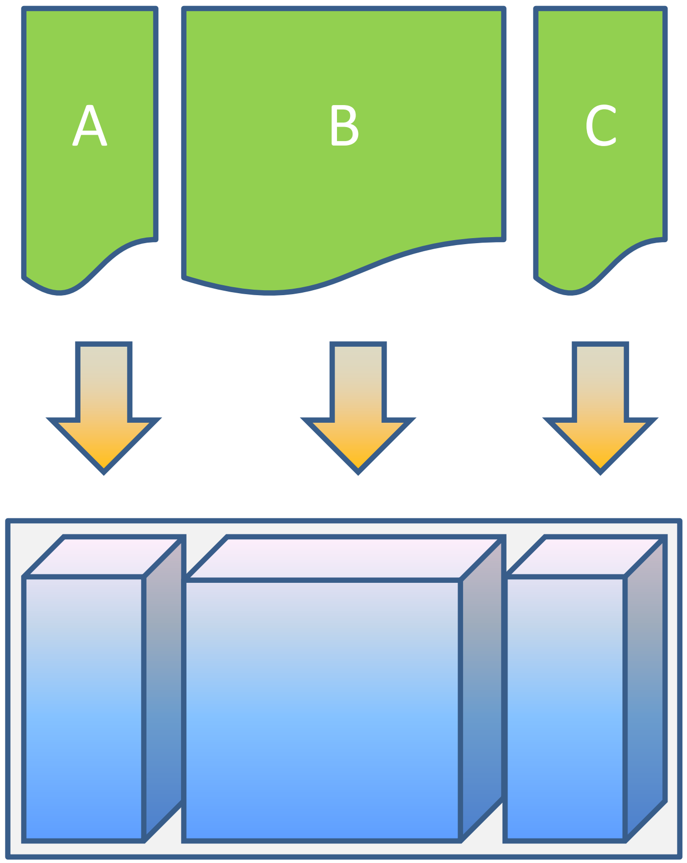
What do we do in the ERA project?
- Parameterization of processor designs
- Match processor designs to the applications (through parameters)
- Perform switching of processor cores dynamically (at run-time)
- Self-optimize based on available resources and power budget

*: moved the complex instruction scheduling to the compiler (= VLIW processor concept)

**Compiler:**
- Targets reconfigurable & parameterized processor
- Match characteristics with processor capabilities

**Many datapaths:**
- Can be combined to form different processors
- Reconfigurable & parameterized processor(s)
- Adaptive functionality
- Adaptive behavior based on resources, power budget, and target performance (self-optimization)

## An Example (1/3)



Program A wants to run on the ERA platform

Instantiate a core capable of running program A

Run program A on the new core

Program B wants to run on the ERA platform

Instantiate a core capable of running program B

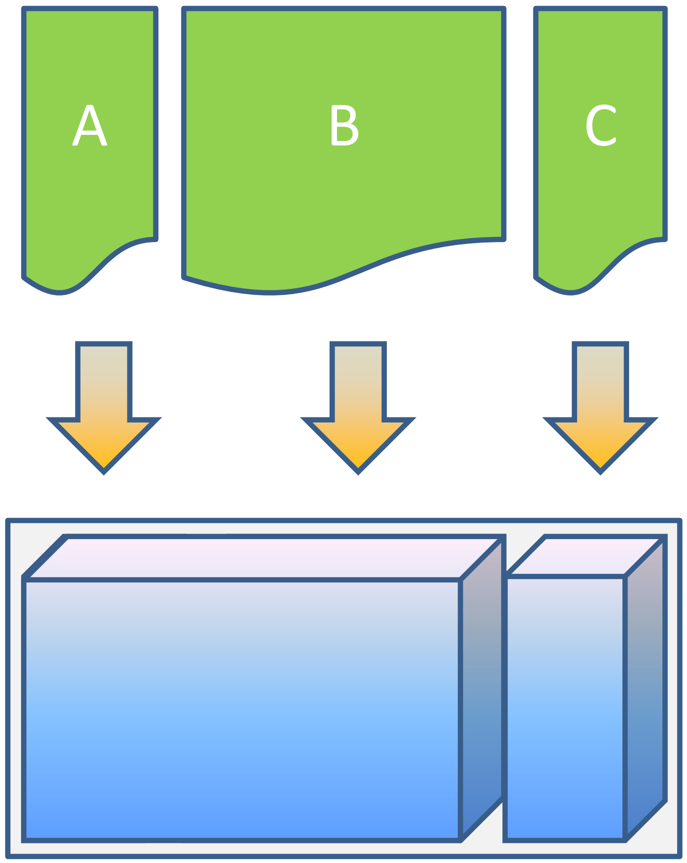
Run program B on the new core

Program C wants to run on the ERA platform

Instantiate a core capable of running program C

Run program C on the new core

## An Example (2/3)



Program A finishes

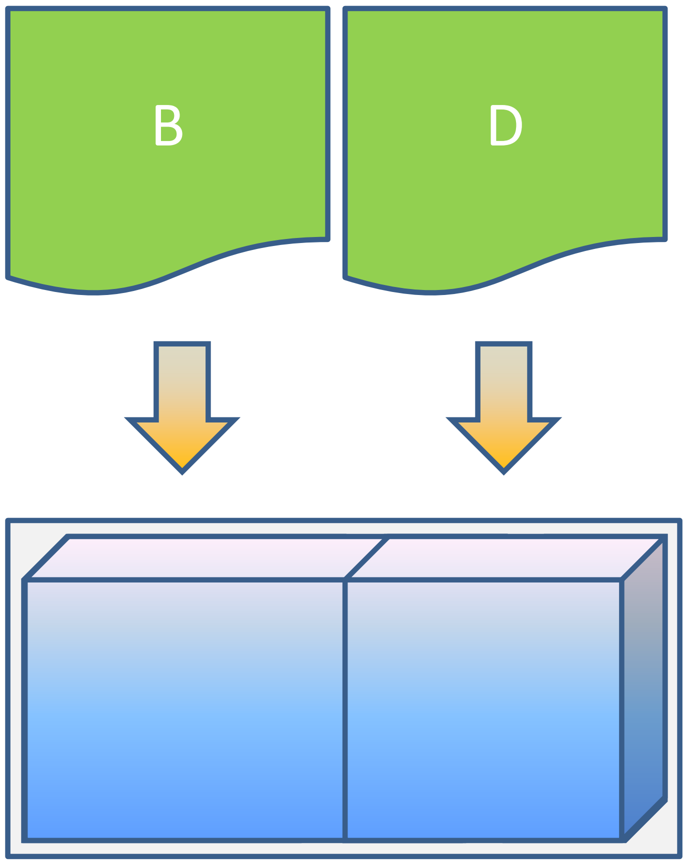
The related core is gated off to save power

Program B utilizes more resources to improve performance

Program C finishes

The related core is gated off to save power

## An Example (3/3)



Program D wants to run on the ERA platform

Program D's preferred core size is not available

Instantiate a non-preferred core on the remaining resources and execute program D on that core; OR

Reduce the core size executing program B

Instantiate a core capable of running program D

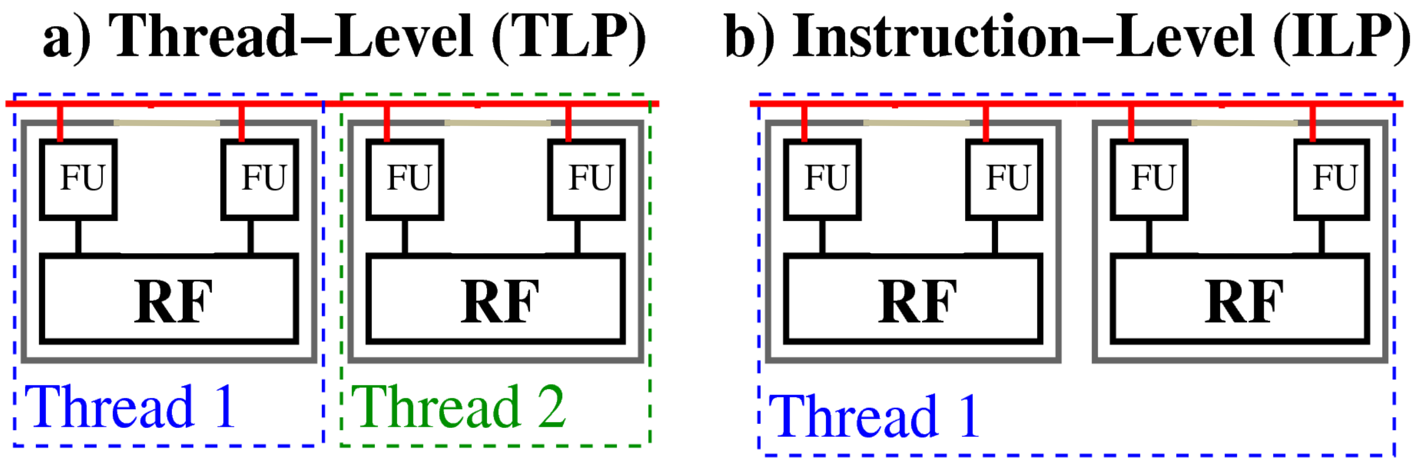
Run program D on the new core

How about the network-on-chip (NoC) and memory hierarchy?

We apply the same concepts illustrated by the processor example to the NoC and memories!!

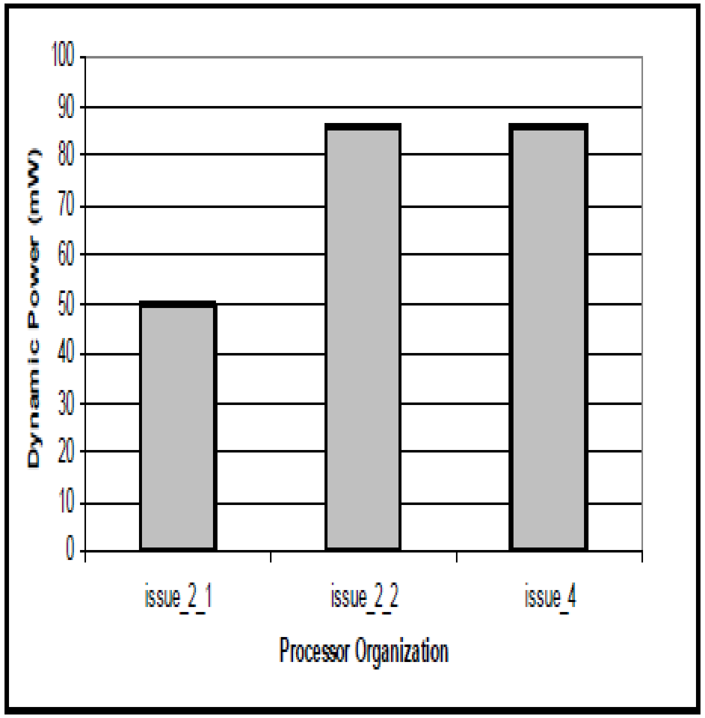
## TLP vs. ILP

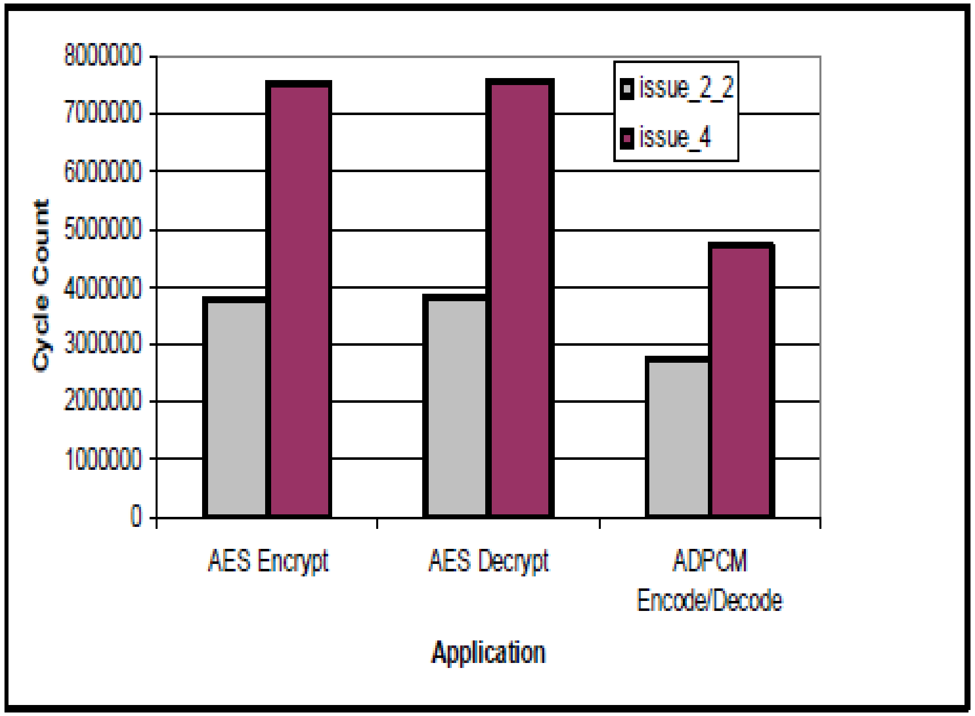
- Leverage reconfigurable multi-core to adapt resources from TLP to ILP or fault-tolerance
- Key ideas:
  - Add direct pair-wise fine-grain communication support to interconnect and ISA
  - Compiler manages ILP through advanced clustering techniques



## Energy/Performance Trade-Offs

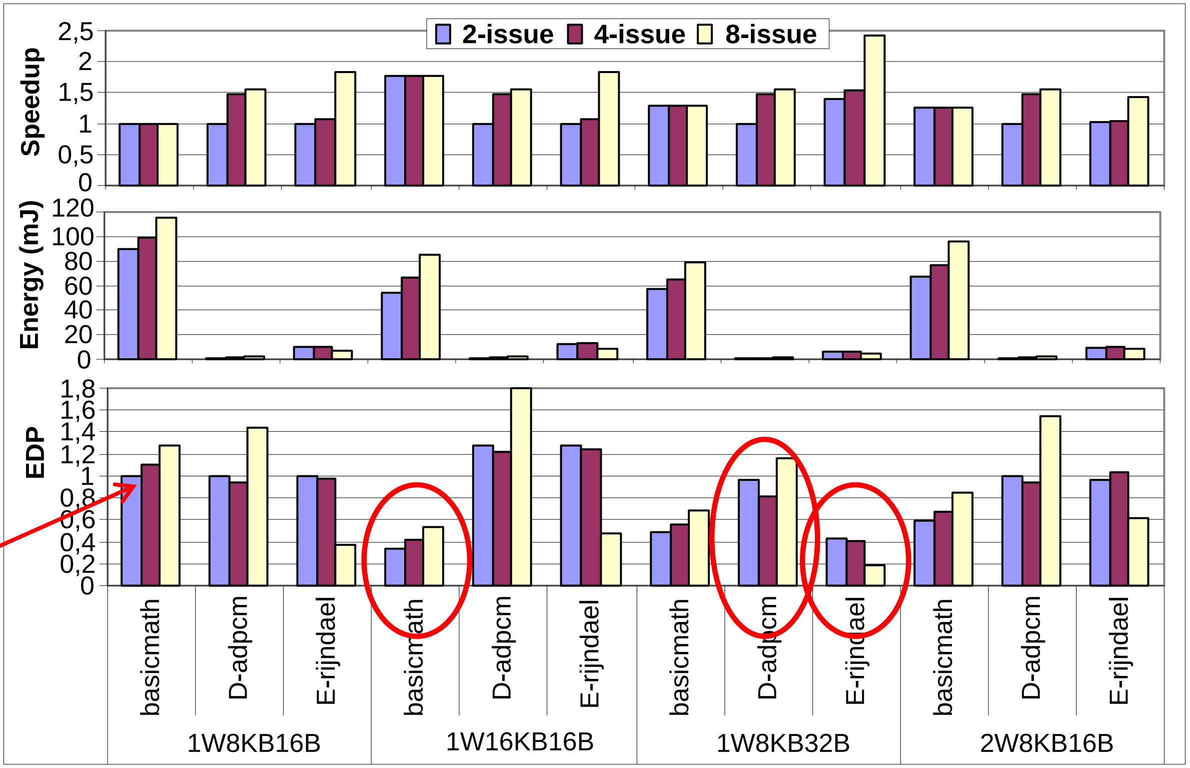
- Higher ILP applications favor wider issue cores
- Higher TLP favors "more & smaller" cores





## Processor-Memory Reconfiguration (1)

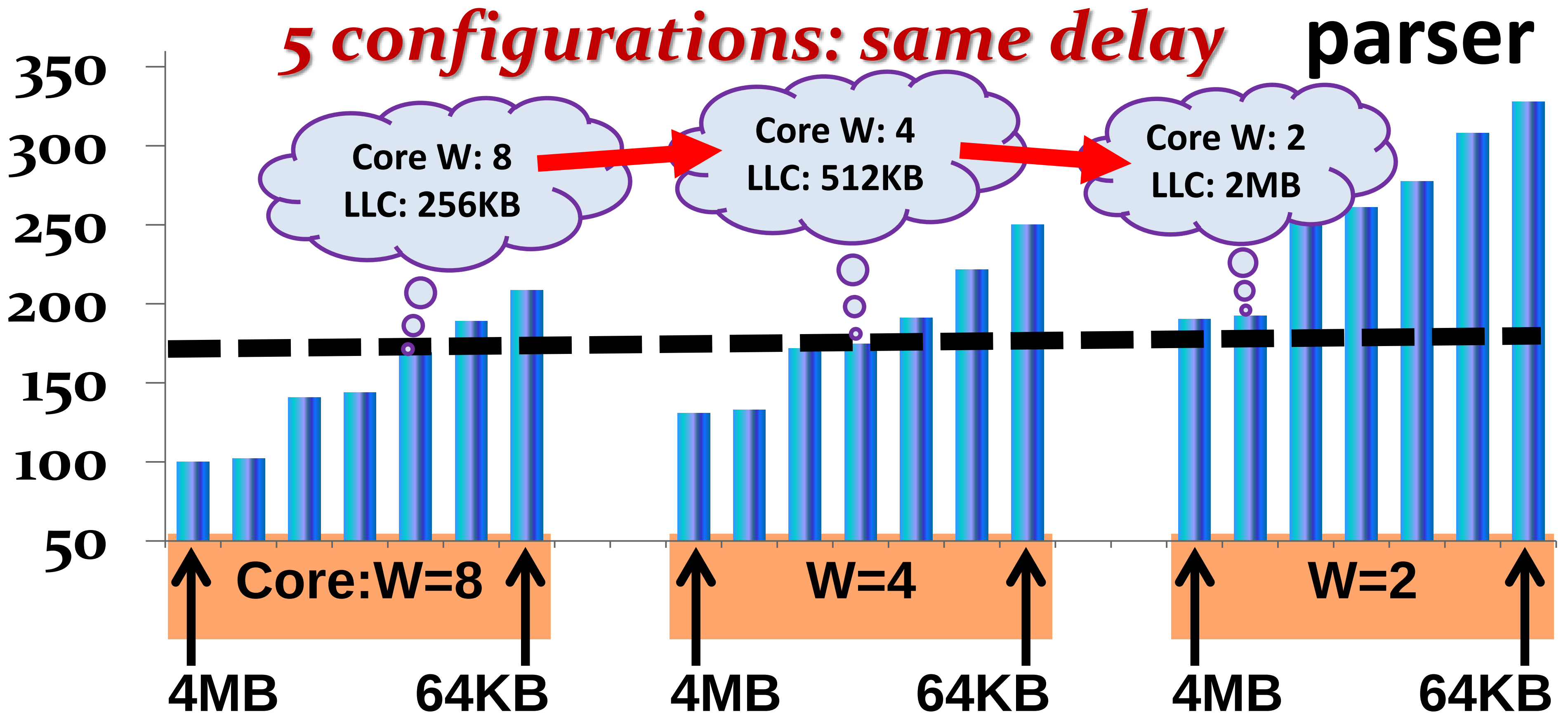
- Different **processor configurations** require different **I-cache configurations** for **different applications**
- Therefore, we configure the core and memory according to application requirements



Normalized to 2-issue

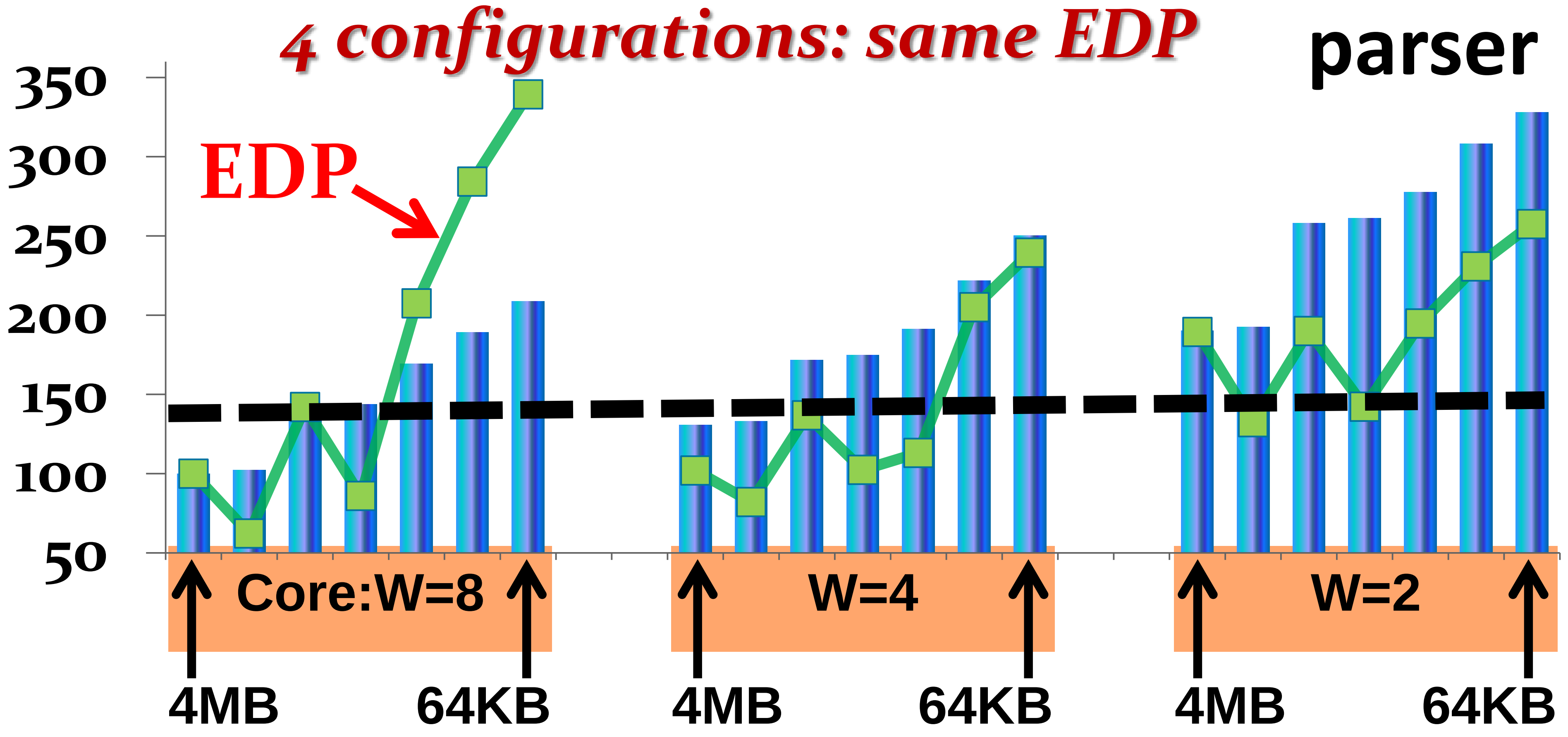
## Processor-Memory Reconfiguration (2)

- **Performance** varying **Core Parallelism** and **LLC Size**



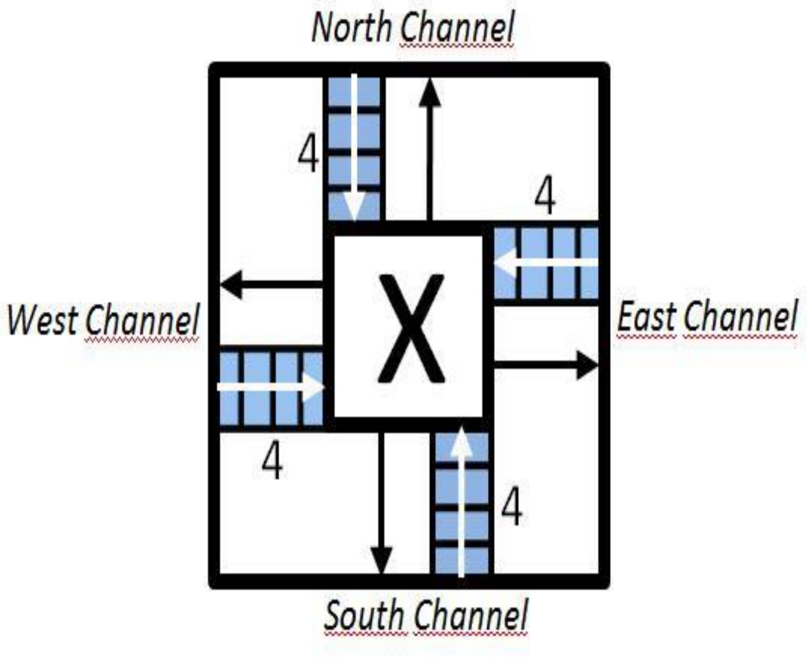
## Processor-Memory Reconfiguration (3)

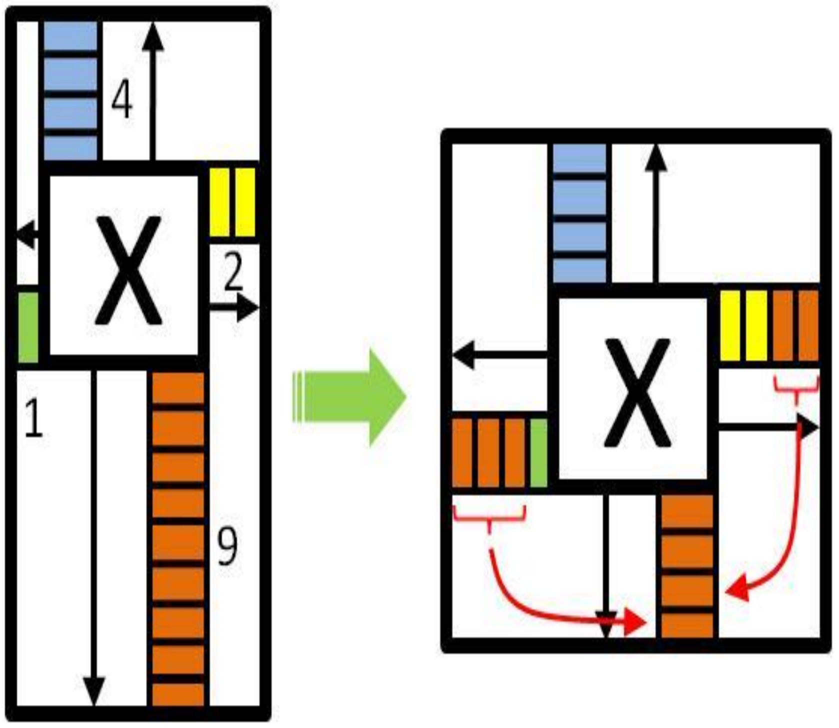
- **EDP** varying **Core Parallelism** and **LLC Size**



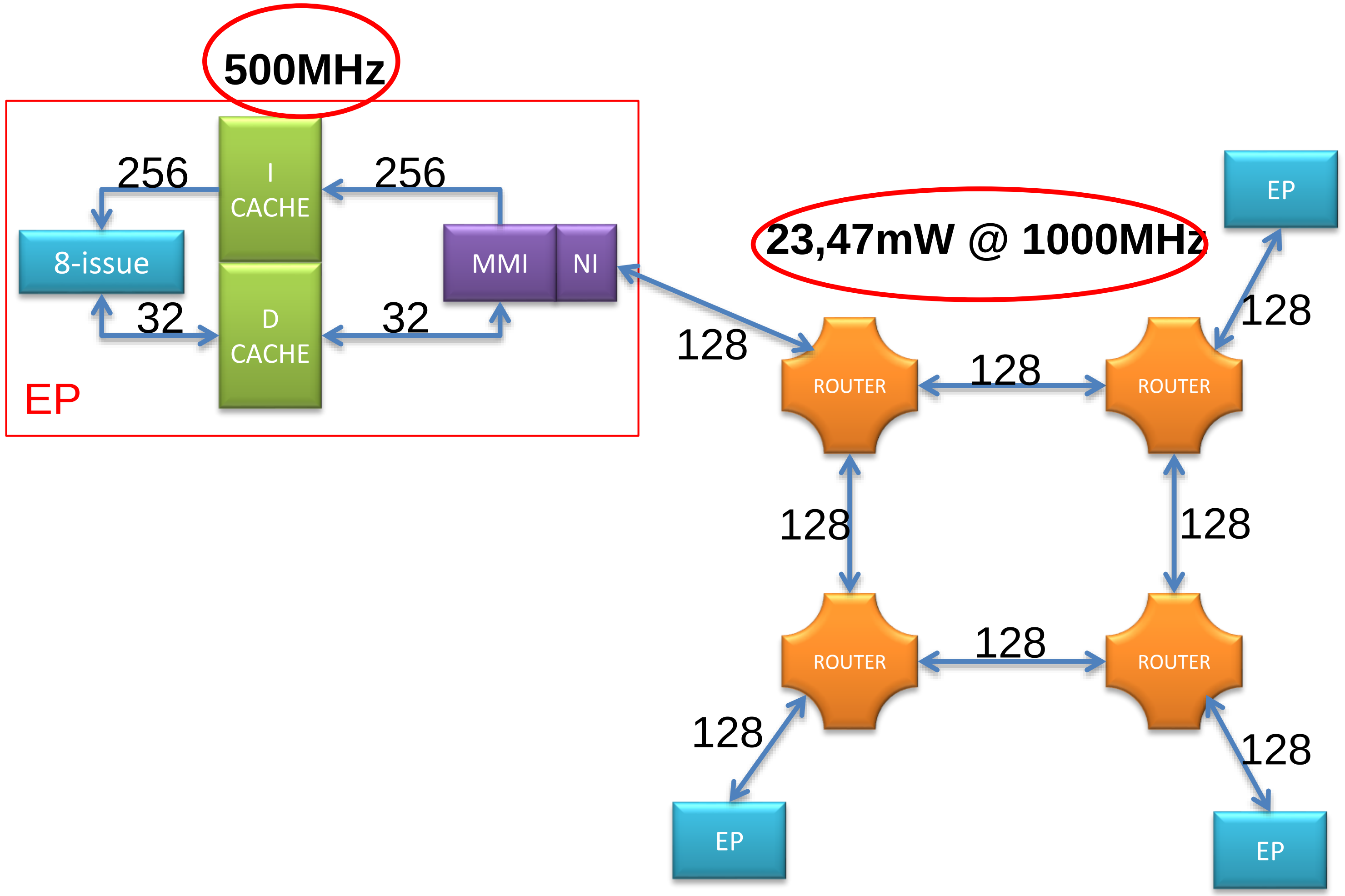
## Reconfigurable NoCs

- Different traffic characteristics require different dimensioning of the NoC
- Even the same application can have phases generating different traffic

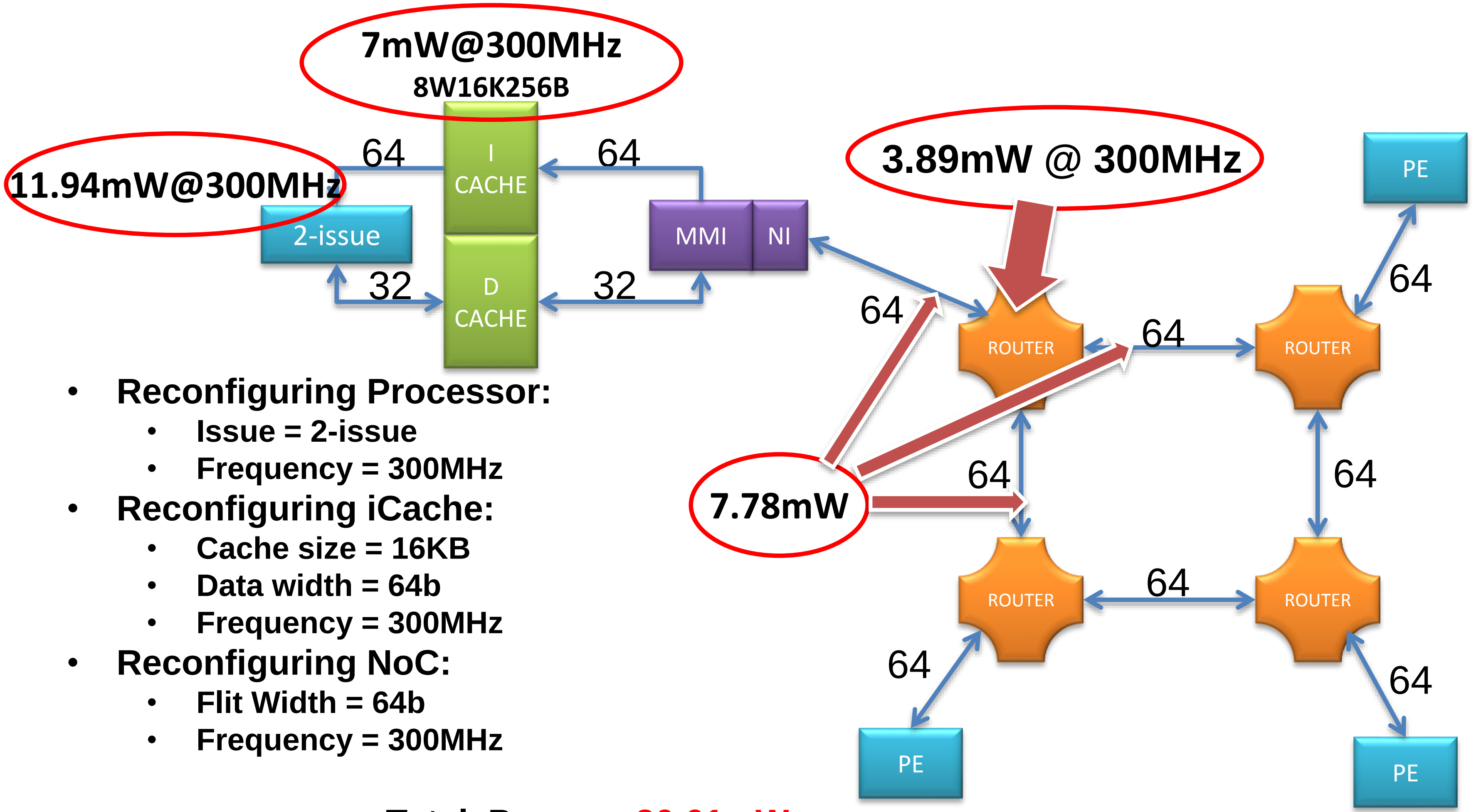




## MPSoC: Default Configuration



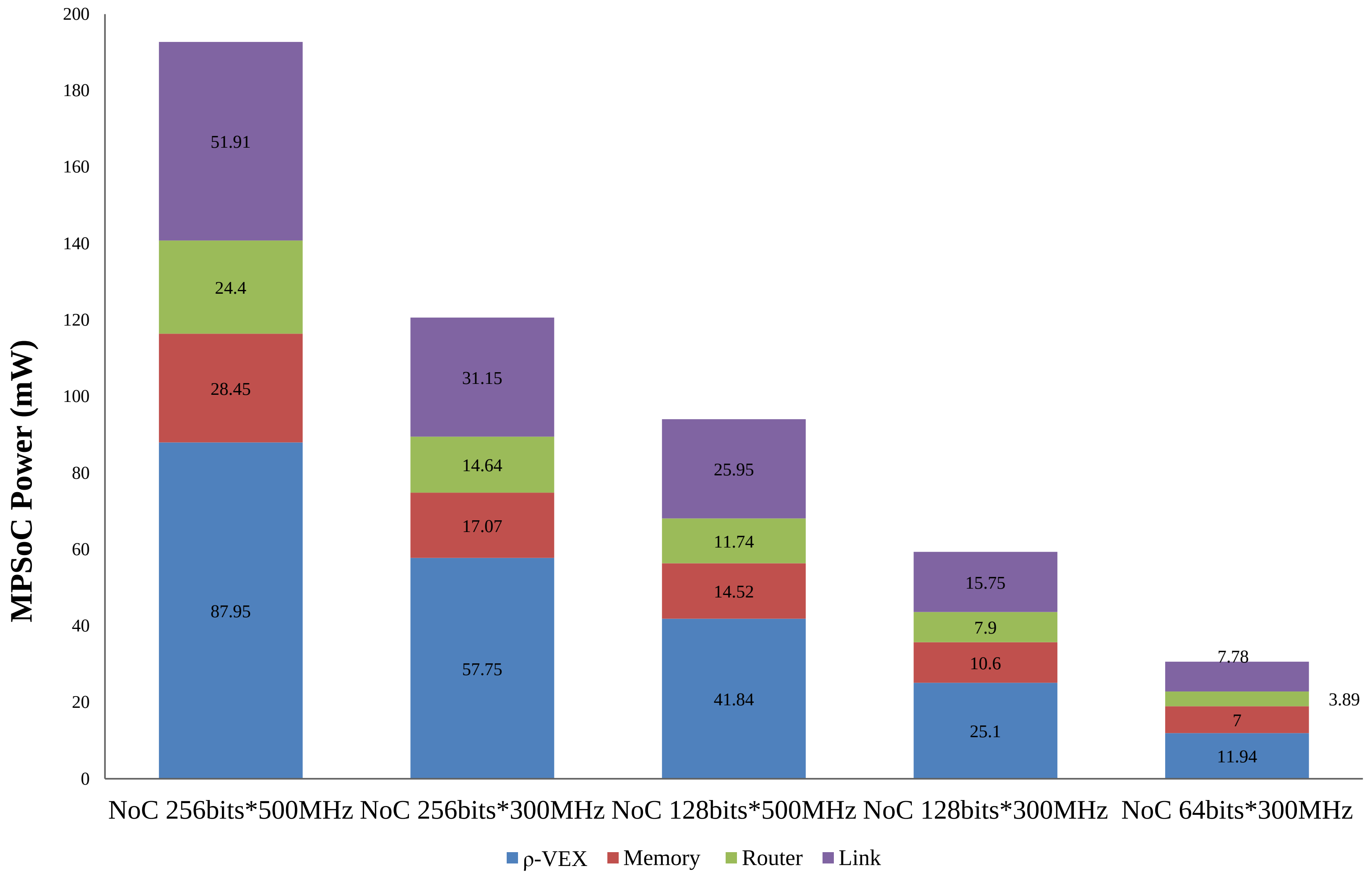
# MPSoC: Reconfiguring processor, memory, & NoC



- **Reconfiguring Processor:**
  - **Issue = 2-issue**
  - **Frequency = 300MHz**
- **Reconfiguring iCache:**
  - **Cache size = 16KB**
  - **Data width = 64b**
  - **Frequency = 300MHz**
- **Reconfiguring NoC:**
  - **Flit Width = 64b**
  - **Frequency = 300MHz**

**Total Power = 30.61mW**

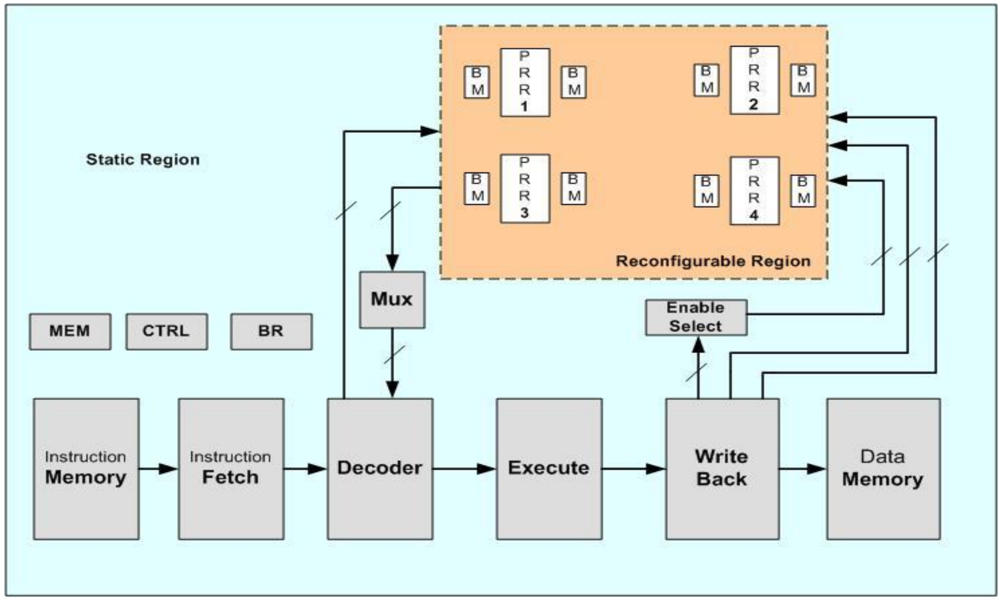
# Fully Adaptable System



## Recent Developments for the ρVEX

### Dynamically Reconfigurable Register File for ρ-VEX:

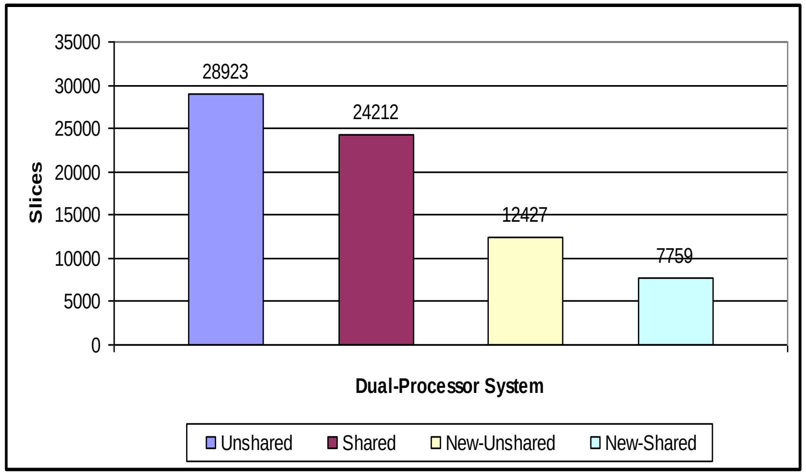
- Implemented dynamically reconfigurable register file for ρ-VEX [*presented at DATE 2010*]
- Registers can be configured as per requirement of application at runtime
- Extra area (slices) can be saved
- Registers are configured in a chunk of 8-registers (924 slices)
- Utilized the Xilinx EAPR methodology

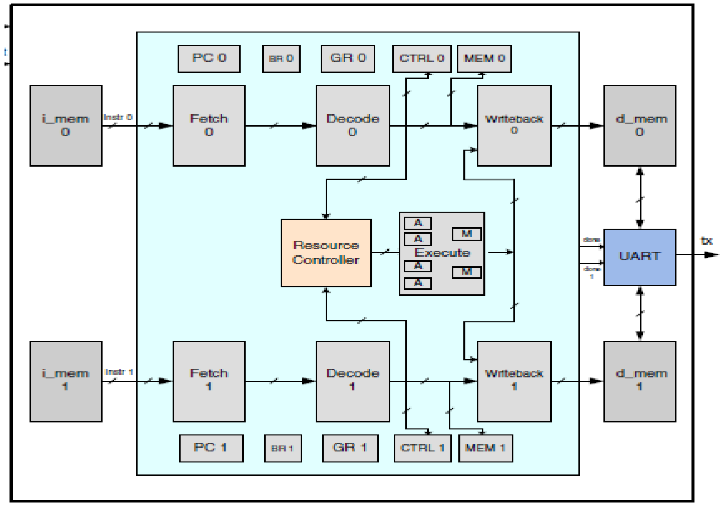


# Recent Developments for the ρVEX

## A Shared Reconfigurable VLIW Multiprocessor System:

- Implemented a dual-core processor system sharing the resources [*presented at RAW 2010*]
- Based on ρ-VEX (v1.0) on Virtex-II Pro
- Execute unit and register file consume 34% and 58% resources
- Two contributions:
  - Execute unit is shared between two cores, and
  - Register file is implemented using BRAMs (New-unshared and New-shared)





ICTOPEN
THE INTERFACE FOR DUTCH ICT-RESEARCH
24 - 25 MARCH 2015

## Recent Developments for the ρVEX

ρVEX V1.0 [*presented at FPT 2008*]

Dynamically Reconfigurable Register File for ρ-VEX [*presented at DATE 2010*]

Multi-ported register file design using BRAMs [*presented at FPT 2010*]

ρVEX V2.0 & extensions: [*presented at WRC 2012 (2 papers)*]
- Paper 1: Pipelined, forwarding logic, Paper 2: Support for traps (interrupts, exceptions)

Run-time task migration [*presented at ARC 2012*]

Dynamic issue-width reconfiguration: [*presented at FPT 2010, DATE 2011*]
- Dynamic adaptation of issue slots

Dynamic issue-width and 1st level I-cache reconfiguration: [*pres. at SAMOS 2012*]
- Simultaneous reconfiguration of core issue width and I-cache parameters

Binary compatibility for dynamic issue-width adaptivity [*presented at DATE 2013*]

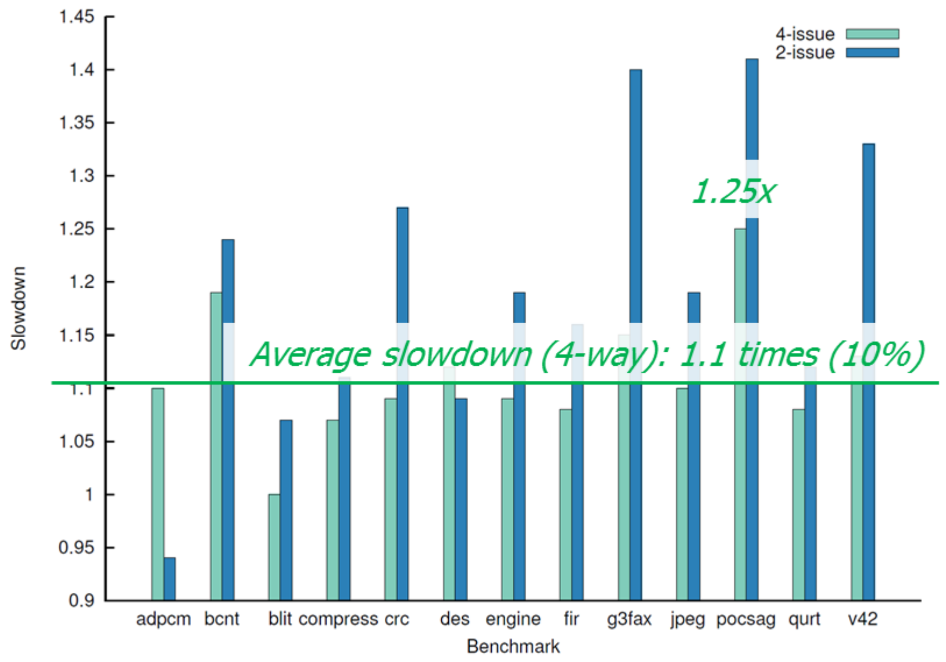
Dynamic support for fault-tolerance [*presented at ARC 2013*]

ICTOPEN
THE INTERFACE FOR DUTCH ICT-RESEARCH
24 - 25 MARCH 2015

## Recent Developments for the ρVEX

Binary compatibility for dynamic issue-width adaptivity:

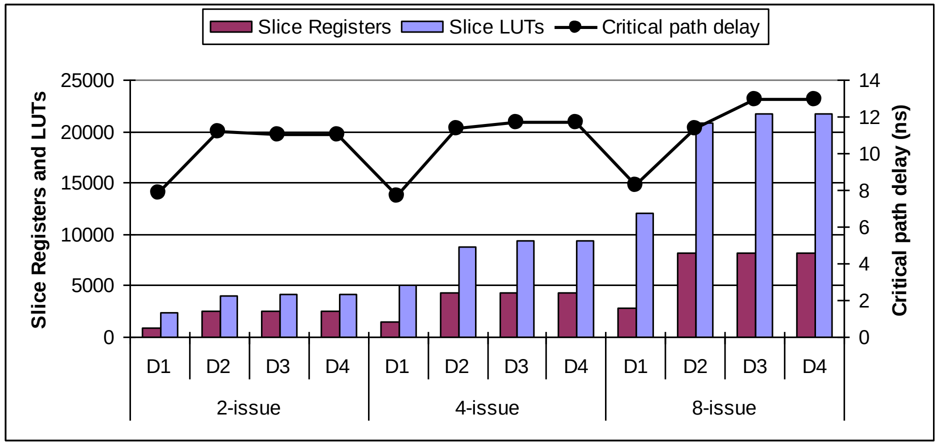
- Definition of the "generic binary" [*presented at DATE 2013*]
- Approach:
  - Compile for 8-issue and address them as 2-issue bundles
  - Fix false dependencies, skip NOPs-only bundles
  - Simple hardware change in "update PC" & "skip NOPs"
- Advantages (over code versioning):
  - Interruptability
  - Dynamic switching of issue width (controlled by application designer, compiler, hardware scheduler, and/or OS scheduler)
- Disadvantage → performance loss (measured: avg. 30%, projected: 10%)
- NEW RESULTS: avg. 5%



# Recent Developments for the ρVEX

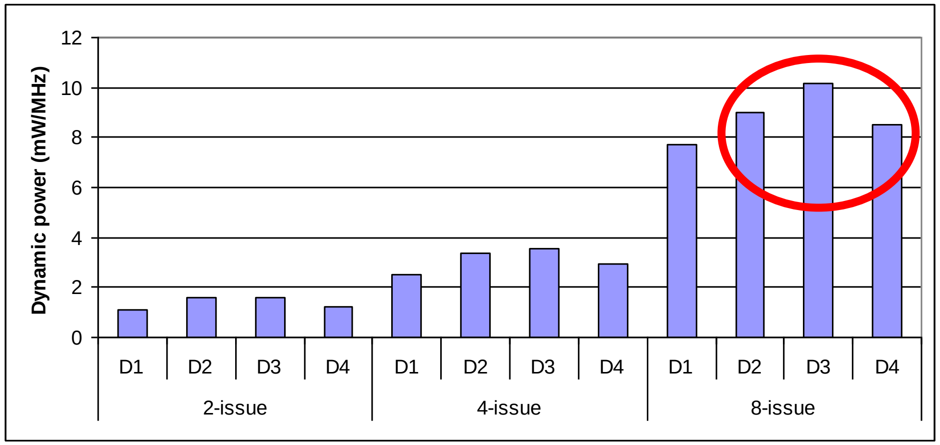
Dynamic support for fault-tolerance (presented at ARC 2013):

- [*presented at ARC 2013*]
- Parity bits for Imem, Dmem, and register file
- TMR for all FFs within design (targeted both FPGA and ASIC (not shown))



- Similar critical path for designs
- Similar resource utilization for designs *(unused BRAM bits & LUTs)*
- Dynamic power consumption *(assuming 10% switching activity)*

D1 – Base non fault-tolerant design
D2 – Permanently enabled fault-tolerant design
D3 – Run-time configurable fault-tolerant design with the fault tolerance enabled
D4 – Run-time configurable fault-tolerant design with the fault tolerance disabled



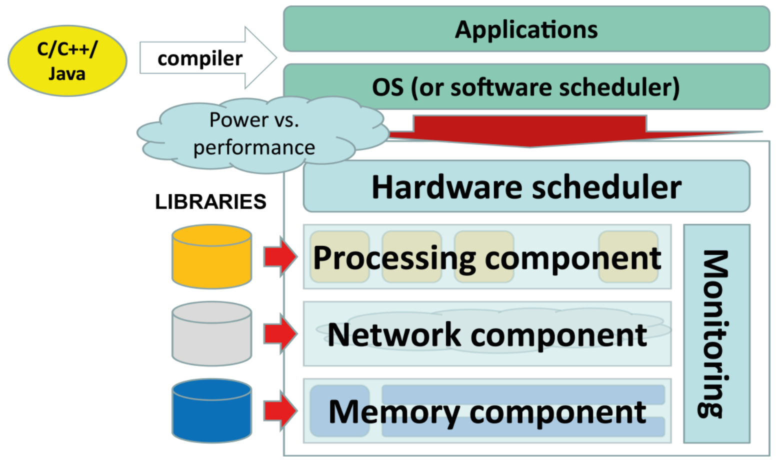
ICTOPEN

THE INTERFACE FOR DUTCH ICT-RESEARCH

24 - 25 MARCH 2015

ERA

# Embedded Reconfigurable Architectures



**Dynamic adaptation**
to *software requirements & operating environment*

**Dynamic adaptation**
in *performance, power / energy, and resources*

**Dynamic reconfiguration**
of *processor cores, caches, and NoCs*

## Achievements at end of project:

- **Fully functional design of the ρVEX (v2.0)* processor:**
  1. Tested on many FPGA chips (Xilinx Virtex4-5-6, Altera Stratix)
  2. Instance on Virtex-6 running at 150 MHz
- **3 fully functional platforms:**
  1. Operating System (OS) controlling the ρVEX cores (*many-core*)
  2. Stand-alone ρVEX with adaptive memory systems (*multi-core*)
  3. Reconfigurable NoC with several ρVEX cores
- **2 cycle-accurate simulators:**
  1. VEX simulator from HP (*open-source*)
  2. xSTsim simulator from STMicro (*binary*)
- **2 fully functional toolchains:**
  1. HP compiler based on Multiflow (*binary*)
  2. GCC (*open-source*)

Release in:

***V2.0 – July 2013***
***V2.1 – September 2013***
***V3.0 – November 2014***
***(dynamically load programs)***

***NOW:***
- ***Lab (incl. manual)***
- ***Hands-on tutorial***
- ***Linux OS on ρVEX***
- ***LLVM port for ρVEX***
- ***Open64 port for ρVEX***
- ***CoSy port for ρVEX***

*: developed in Delft

**Partners**

**Technische Universiteit Delft**
*(TUD) – NL (Coordinator)*

**Industrial Systems Institute**
*(ISI) – GR*

**Universita' degli Studi di Siena**
*(UNISI) – IT*

**Chalmers University**
*(CHALMERS) – SE*

**University of Edinburgh**
*(UEDIN) – UK*

**Evidence**
*(EVI) – IT*

**STMICRO**
*(STMICRO) – IT*

**IBM**
*(IBM) – IL*

**Universidade do Rio Grande do Sul**
*(UFRGS) – BR*

**Uppsala University**
*(UU) – SE*

Contract: **INFSO-ICT-249059**
EU Funding: **2.8 MEuro**
Start - End: **01-2010 – 03-2013**

## Current team and beyond (March 2015)

**PhD students**:

1. Porting Linux (v2.0)
2. Scheduling and compiler algorithms (finishing)
3. Run-time task scheduling (1 year)
4. Cache coherence (1 year)

**MSc students**:

1. Multiple context support & MMU design
2. Hardware/Software Fault-Tolerance (2x)
3. Compiler support for reconfiguration
4. Compilation support for Linux without virtual memory
5. *(Just started)*

**Active collaborations**:

1. University of Torino, Italy (visiting professor & visiting MSc student)
2. UFRGS, Brazil (2 visiting professors & 2 visiting PhD students)
3. UFV, Brazil (1-year sabbatical of professor)
4. Ruhr University Bochum, Germany (uses our processor design)
5. TU Darmstadt, Germany (uses our processor design)
6. Brno University of Technology, Czech Republic (compiler work using LLVM)

**Others:**

- At least 10 signed licenses to use V2.x of ρVEX release.
- Platform now being used by two courses in Computer Engineering MSc Programme.

## Analogy cont'd: An Army of Delivery Drones

*Liquid Computing:*

- *Run-time adaptivity*
- *Efficient computing*
- *Fault-tolerance*
- *Efficient HW utilization*
- *Energy-efficient computing*
- *Many exciting ideas to explore!!*

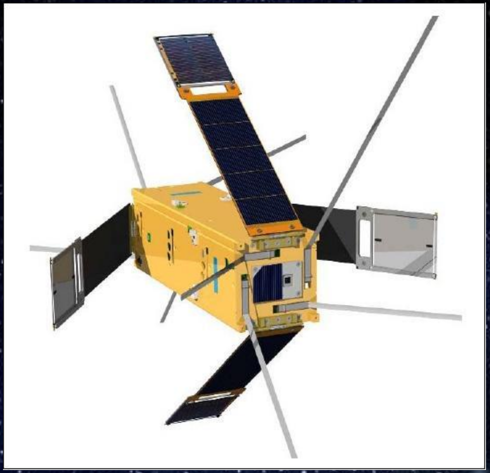
*Like having an army of delivery drones of all sizes flowing through our campus!*

ICTOPEN

THE INTERFACE FOR DUTCH ICT-RESEARCH

24 - 25 MARCH 2015

## Liquid Computing in Space


Delfi-C3 from TU Delft

- Harsh environment requires run-time adaptability, e.g., fault-tolerance
- Certain control systems need responsiveness

We are working to bring Liquid Computing into SPACE!