Session 4b: Design Requirements for Multicore Reconfigurable Platforms

The future of architectural design for embedded platforms

13.10.14 PSP FPGA Symposium

Monday October 14th 2013
Congrescentrum Het Pand,
Onderbergen 1, 9000

psp//fpga www.psp-fpga.eu

Carlos Valderrama carlos.valderrama@fpms.ac.be
Electronics & Microelectronics Department [SEMi]
Mons University [UMons]
Mons, Belgium

…

Moore’s Law Evolution

Continuously increasing number of transistors per chip

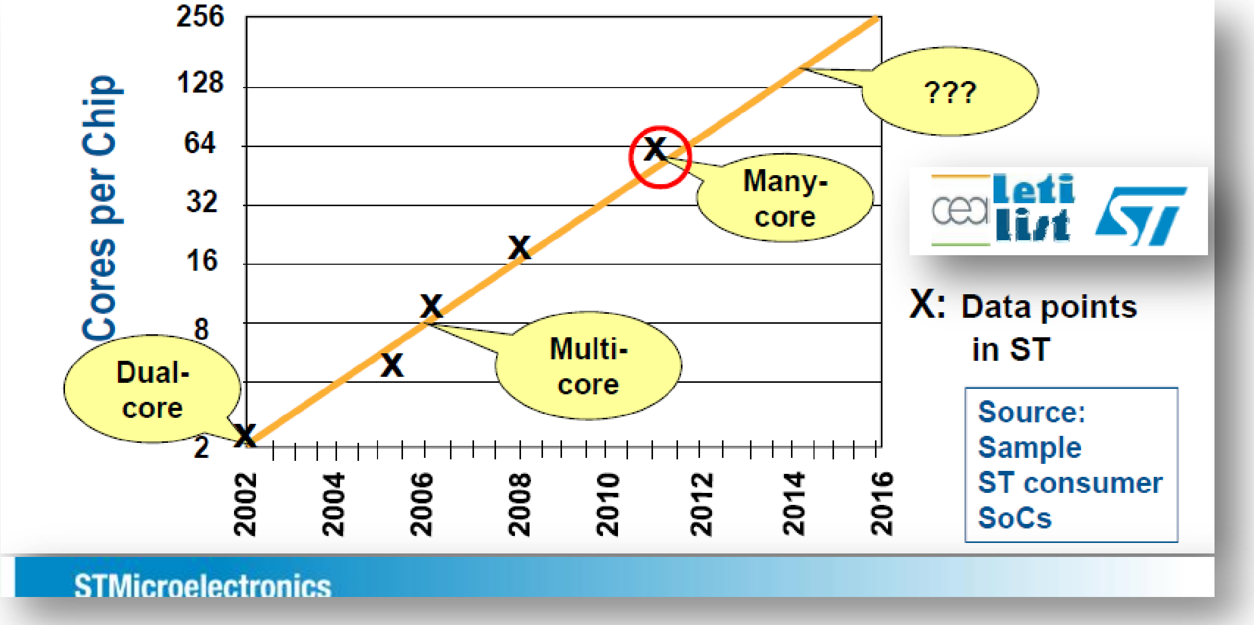
- **Scaling problems with ASICs**[3]: the best cost/performance in high-volume products but increasing development /verification costs
- **Static power increases with process scaling**: Smaller silicon geometries result in **increased leakage** currents resulting in **higher static power**
- **Dynamic Power increase**: voltage scaling flattening around 1Volt
- **Increasing demand of bandwidth requirements while reducing cost and power**: FPGA vendors have progressed from previous silicon processing technologies to **28 nm** to provide the benefits of Moore’s Law
- **FPGAs move to latest process very early**: Foundries use FPGAs as **process drivers**. **Regular structures** help shake out systematic FAB issues.
- **MPSoC**[2] : from fat/complex cores to **multi-core architectures** and multi-core programming. **SMP Symmetric Multiprocessing** multithreaded architectures, **SIMD**s and **MPPAs Massively Parallel Processor Arrays**.
- **Era of GPU computing**[1]: thin cores in large quantity. CPU has exhausted its potential.

[1] Chien-Ping Lu, nVidia, 2010
[2] P. Paulin, Towards Multicore Programming, 2011
[3] LaPedus, 2007

Core's Law

Cores in SoC products: 2x every 2 years (embedded SoC)

- **Parallelism**: from **sequential** programming to **parallel** (a huge leap forward[1])
- **Derivative**: from processing elements to **clusters** of configurable cores & **customizable tiles**. Domain oriented.
- **Scalability**: From **bus interconnection** to **NoC**. More resources and performance.

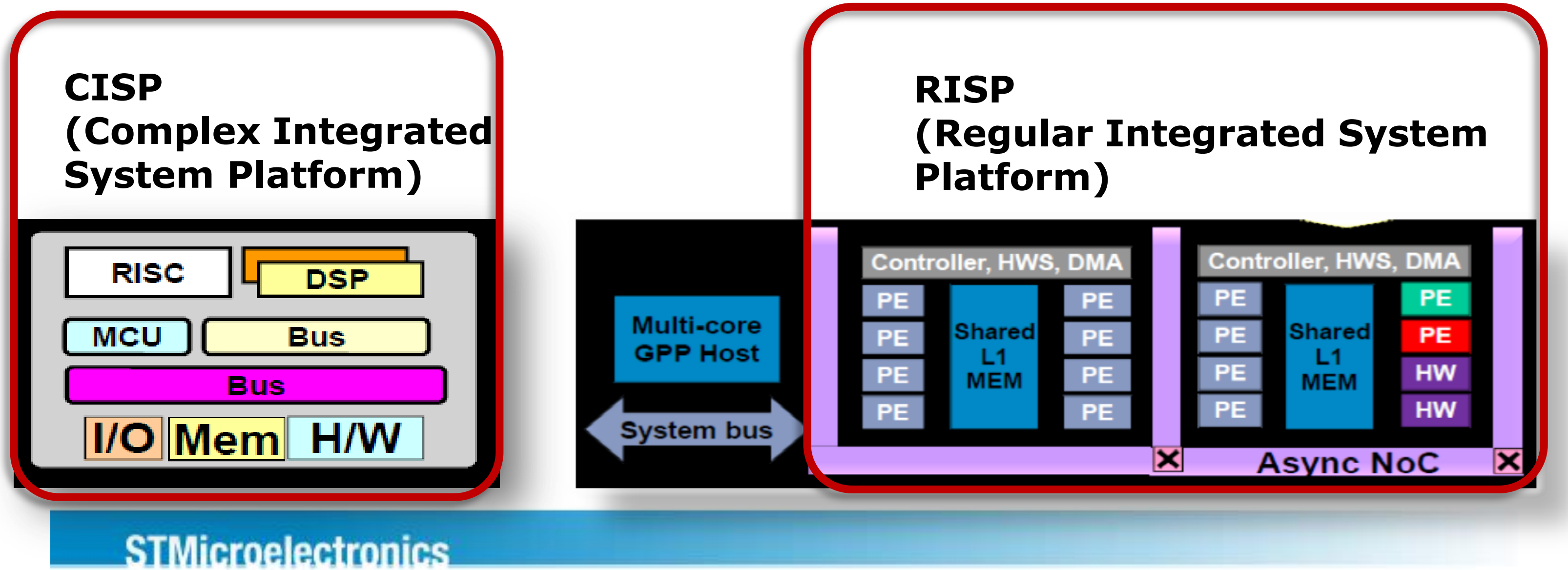


[1] P. Paulin, Towards Multicore Programming, 2011

Core's Law Evolution

Multi-processor SoC: Scalable Tile

- **Multi-core fabric**: **regular** and **heterogeneous** platforms (**CISP** & **RISP**)
- **Component-based Programming Models**: Higher productivity & **platform independence**



[1] P. Paulin, Towards Multicore Programming, 2011

ARTEMIS in 2006
Multicore Processing

ARTEMIS: **A**dvanced **R**esearch & **T**echnology for **Em**bedded **I**ntelligence and **S**ystems

MPSOC: on-chip clusters of **concurrent heterogeneous distributed** functionality

- **Roadmap**:
  - Reference designs and architectures,
  - Seamless connectivity and middleware and
  - System design methods and tools
- **Transparency**: platform **abstraction** for SW designers (from application to HW **layers**)
- **Parallel programming languages**: evolution of C-based **programming paradigms**
- **Portability:** addressed by **middleware layers**

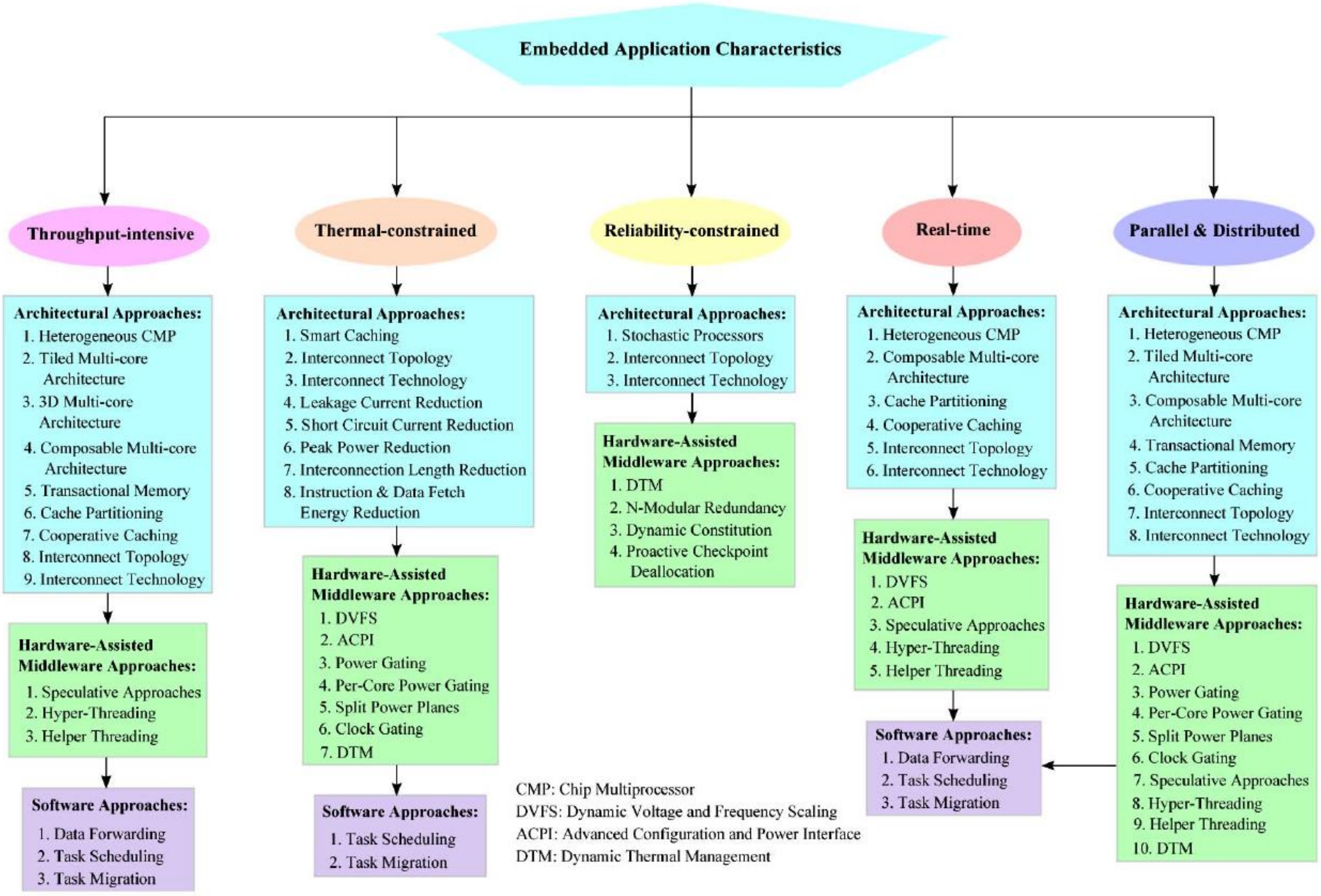
Multicore Processing and ARTEMIS - An incentive to develop the European Multiprocessor research, 2006

ARTEMIS in 2006 Requirements

- **Application: unified models** that capture **concurrency** will enable efficient and flexible **mapping** onto various platforms. **Decomposition** into **tasks** and concurrent **composition** at **run-time**. **Performance constraints**: throughput, power, size, etc.
- **System**: **self-organization**. **Self-awareness** or **context-awareness** (user demands or environment). **Versatility, adaptability, responsiveness**.
- **SW**: **concurrent paradigms** evolution towards the **multiprocessor** approach. **Compiler technologies** covering **firmware layers** (adaptability, static/dynamic, **recompilation**, **reconfiguration**, replacement, repairing).
- **HW**: provision of **abstraction** (transparency, re-utilization, adaptation at application requirements), **granularity**, **dynamic reconfiguration** and **standardized access** to resources
- **Tools**: systematic support for **concurrency**, **synchronization**, communication, global time representation, consistency of **shared data structures**

Multicore Processing and ARTEMIS - An incentive to develop the European Multiprocessor research, 2006

Embedded Requirements



Embedded Application Characteristics

Throughput-intensive

Architectural Approaches:
1. Heterogeneous CMP
2. Tiled Multi-core Architecture
3. 3D Multi-core Architecture
4. Composable Multi-core Architecture
5. Transactional Memory
6. Cache Partitioning
7. Cooperative Caching
8. Interconnect Topology
9. Interconnect Technology

Hardware-Assisted Middleware Approaches:
1. Speculative Approaches
2. Hyper-Threading
3. Helper Threading

Software Approaches:
1. Data Forwarding
2. Task Scheduling
3. Task Migration

Thermal-constrained

Architectural Approaches:
1. Smart Caching
2. Interconnect Topology
3. Interconnect Technology
4. Leakage Current Reduction
5. Short Circuit Current Reduction
6. Peak Power Reduction
7. Interconnection Length Reduction
8. Instruction & Data Fetch Energy Reduction

Hardware-Assisted Middleware Approaches:
1. DVFS
2. ACPI
3. Power Gating
4. Per-Core Power Gating
5. Split Power Planes
6. Clock Gating
7. DTM

Software Approaches:
1. Task Scheduling
2. Task Migration

Reliability-constrained

Architectural Approaches:
1. Stochastic Processors
2. Interconnect Topology
3. Interconnect Technology

Hardware-Assisted Middleware Approaches:
1. DTM
2. N-Modular Redundancy
3. Dynamic Constitution
4. Proactive Checkpoint Deallocation

Real-time

Architectural Approaches:
1. Heterogeneous CMP
2. Composable Multi-core Architecture
3. Cache Partitioning
4. Cooperative Caching
5. Interconnect Topology
6. Interconnect Technology

Hardware-Assisted Middleware Approaches:
1. DVFS
2. ACPI
3. Speculative Approaches
4. Hyper-Threading
5. Helper Threading

Software Approaches:
1. Data Forwarding
2. Task Scheduling
3. Task Migration

Parallel & Distributed

Architectural Approaches:
1. Heterogeneous CMP
2. Tiled Multi-core Architecture
3. Composable Multi-core Architecture
4. Transactional Memory
5. Cache Partitioning
6. Cooperative Caching
7. Interconnect Topology
8. Interconnect Technology

Hardware-Assisted Middleware Approaches:
1. DVFS
2. ACPI
3. Power Gating
4. Per-Core Power Gating
5. Split Power Planes
6. Clock Gating
7. Speculative Approaches
8. Hyper-Threading
9. Helper Threading
10. DTM

CMP: Chip Multiprocessor
DVFS: Dynamic Voltage and Frequency Scaling
ACPI: Advanced Configuration and Power Interface
DTM: Dynamic Thermal Management

TOPADS “high performance energy efficient multicore embedded computing”, 2012

Embedded Requirements

Architecture

- ❑ Heterogeneous CMP, Tiled multi-core, Composable multicore
- ❑ Transactional memory, Cache partitioning, cooperative caching
- ❑ Interconnect topology

Middleware

- ❑ Hyper-threading, Helper threading

Software

- ❑ Data forwarding, task Scheduling, task Migration

TOPADS “high performance energy efficient multicore embedded computing”, 2012

ARTEMIS in 2006 Requirements

System developers (application)

- Algorithm modeling
- Concurrency extraction
- Execution scenarios
- benchmarks

SW Frameworks

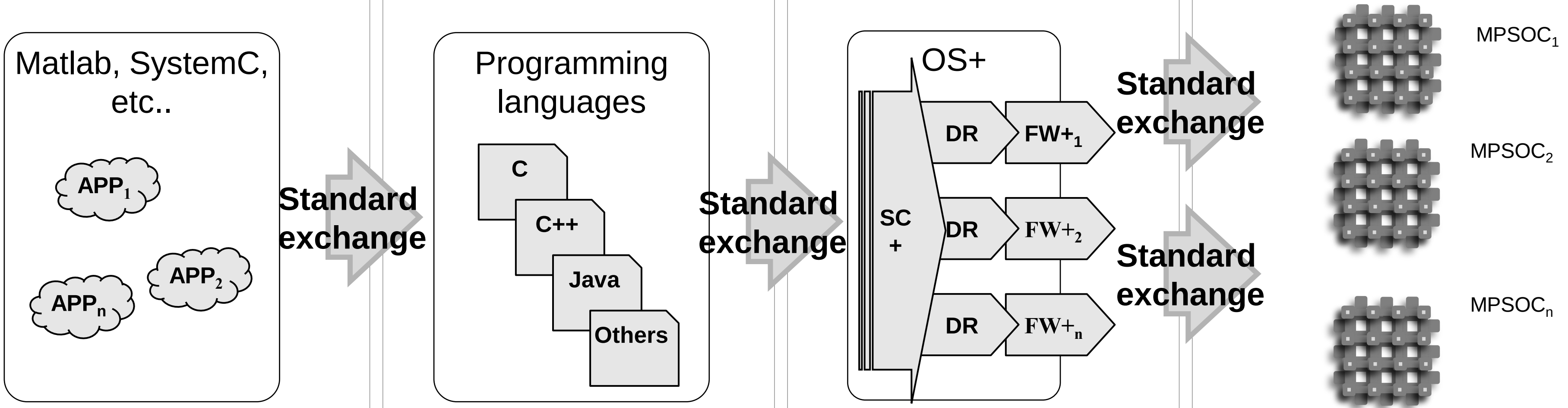
- OS
- Compilers
- Scheduling

SW Firmware FW+

- OS+ (platform aware)
- Compiler+ (platform aware) SC+
- Reconfig policies
- Middleware
- Drivers DR
- Scheduling (time+space)

HW technology providers

- Platform frameworks
- Placement
- Reconfiguration
- Scaling
- Communication



Matlab, SystemC, etc..

APP_1

APP_2

APP_n

Standard exchange

Programming languages

C

C++

Java

Others

Standard exchange

OS+

SC +

DR

DR

DR

$FW+_1$

$FW+_2$

$FW+_n$

Standard exchange

Standard exchange

$MPSOC_1$

$MPSOC_2$

$MPSOC_n$

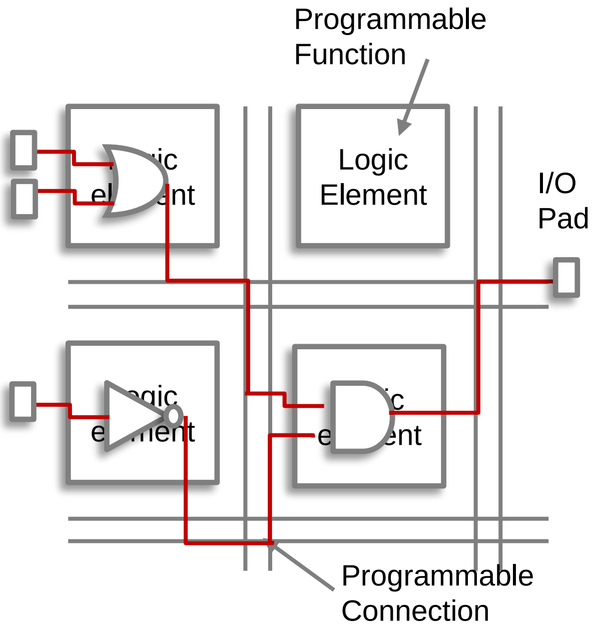
Multicore Processing and ARTEMIS - An incentive to develop the European Multiprocessor research, 2006

FPGA Status

Field Programmable Gate Array

- Two-dimensional **array** of **programmable logic** elements connected with **programmable switches**

  (connections)
- Latest IC technology: always in synch with **latest available technology** (from 130 nm to 40 nm, 28 nm). **High volume production**.
- Prototype oriented: **re-programmable hardware** (one FPGA can serve every customer). **Path to ASIC** (RTL programming, IP providers). **Lower project cost** (No NRE).
- FPGAs/ASIC ratio of silicon area[1] : on average **20-40x area** and **9-12x power** (circuits with only combinational logic and flipflops). Otherwise **Structured ASICs** (one time mask programmable (HardCopy, EasyPath, eASIC).



[1] I. Kuon, Measuring the Gap Between FPGAs and ASICs, 2006

FPGA Evolution

Customization/Specialization/ Increased granularity/Area-efficient and lowest system latency

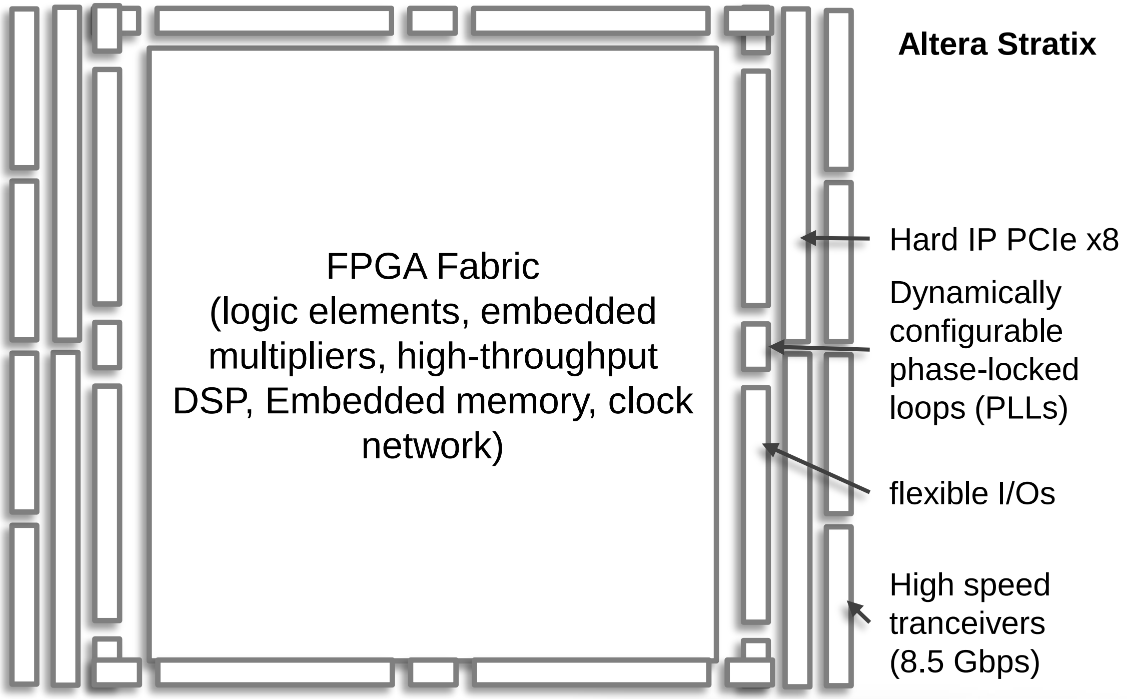
- **Complex and specialized new features**: logic elements, **RAM**. Embedded **multipliers**. High throughput **DSP**.
- **Domain-specific architectures**: **high-speed serial IO** and memory interfaces. **Hard IP**. **Increased performance**: PCIe Gen3x8. 40G/100G Ethernet hard IP variants.
- **Reconfigurable interconnections**: **NoC**
- **Additional processing capabilities**
- **Software in the flow**: **Processor + FPGA**. Preserves processor integrity (both concentrate on their respective product differentiators)
- **Partial Reconfiguration**
- **Lowering cost and power**: Embedded **HardCopy** Blocks (65% reduction in power and 2x performance improvement versus soft logic), **Programmable Power**, Physical Synthesis for Timing

[1] I. Kuon, Measuring the Gap Between FPGAs and ASICs, 2006

FPGA Evolution

FPGA Fabric

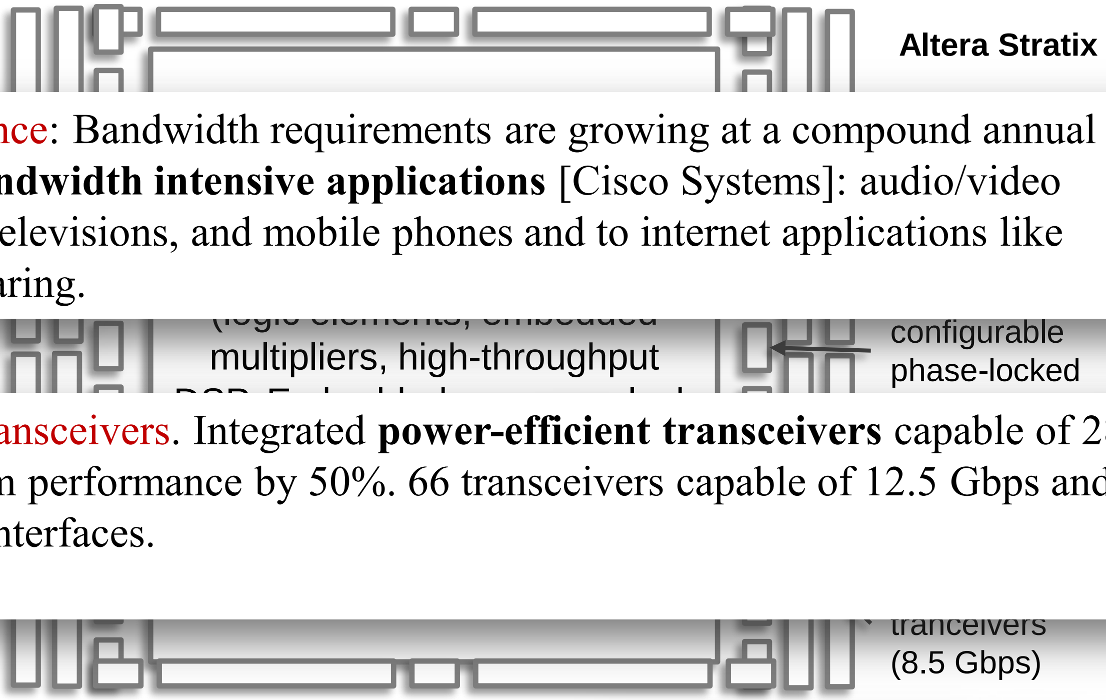
- Embedded **multipliers** & high throughput **DSP**
- Programmable regions
- I/O & clock distribution
- **Distributed memory**


Altera Stratix
FPGA Fabric
(logic elements, embedded multipliers, high-throughput DSP, Embedded memory, clock network)
Hard IP PCIe x8
Dynamically configurable phase-locked loops (PLLs)
flexible I/Os
High speed tranceivers (8.5 Gbps)

FPGA Evolution

FPGA Fabric

- Embedded multipliers & high throughput DSP
- Programmable regions
- I/O & clock distribution
- Distributed memory



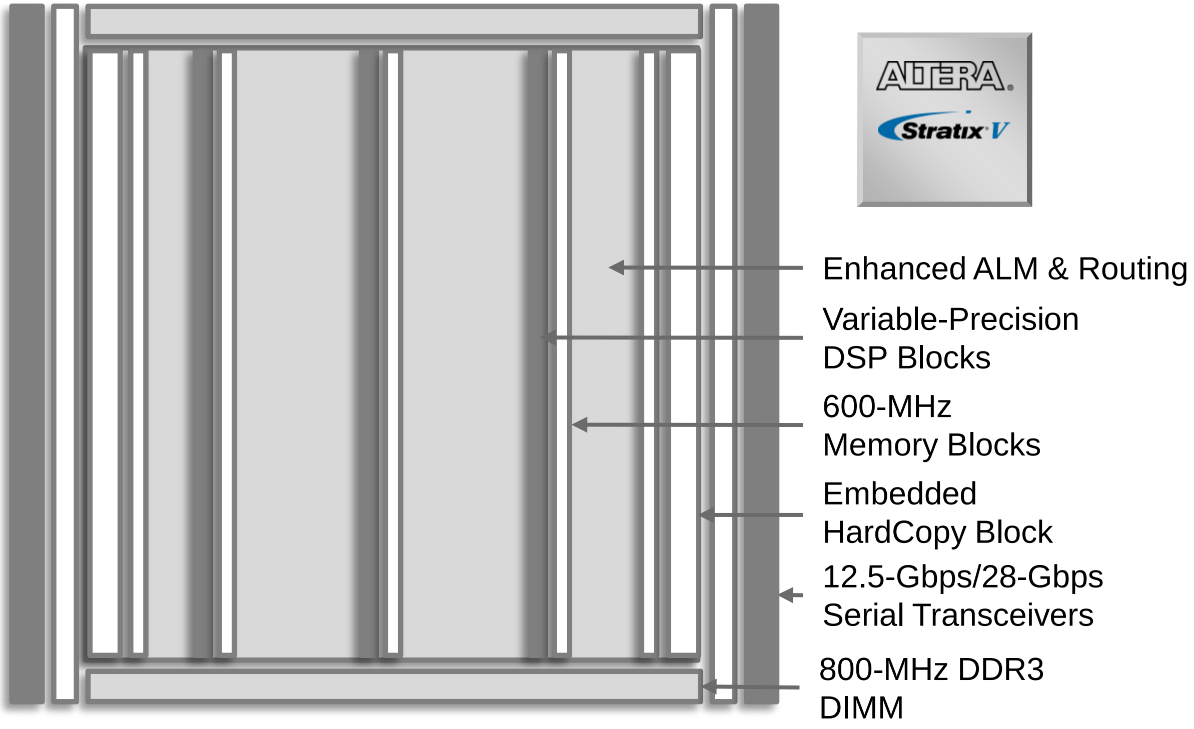
- Bandwidth and performance: Bandwidth requirements are growing at a compound annual growth of 40% due to **bandwidth intensive applications** [Cisco Systems]: audio/video streaming to computers, televisions, and mobile phones and to internet applications like email, games, and file sharing.

- Devices with 28-Gbps transceivers. Integrated **power-efficient transceivers** capable of 28 Gbps and increase system performance by 50%. 66 transceivers capable of 12.5 Gbps and 6 x72 800-MHz DDR3 interfaces.

FPGA Evolution

FPGA Fabric

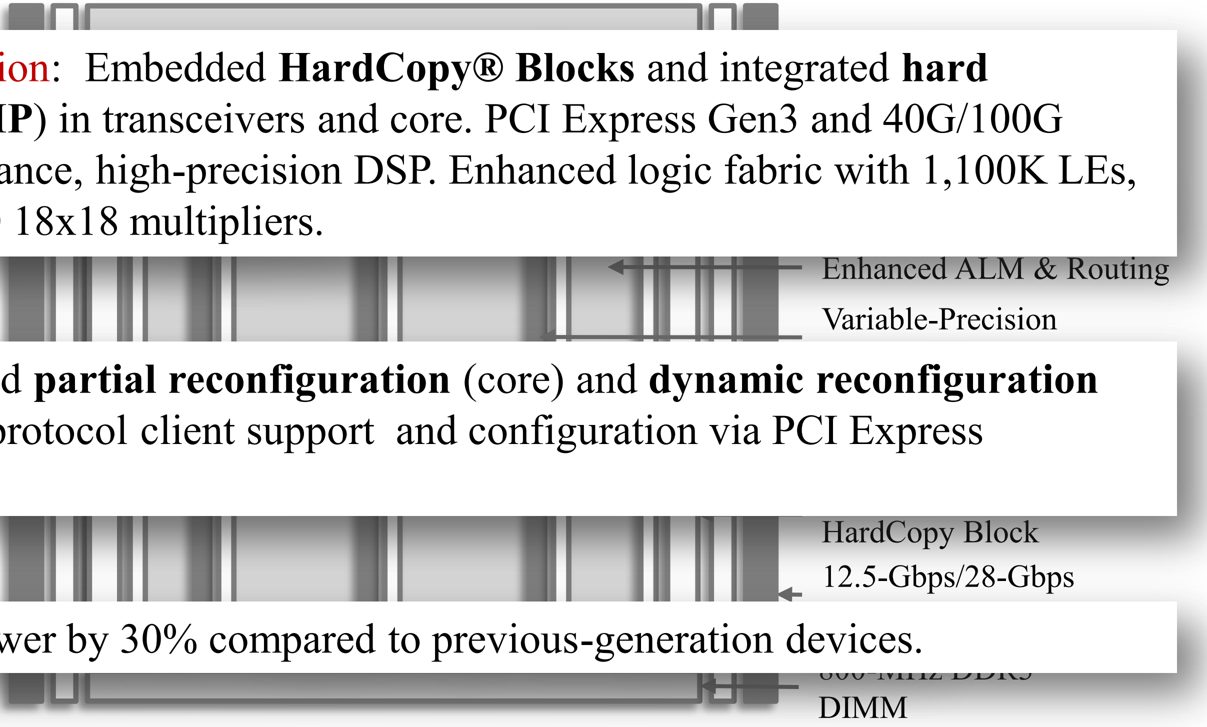
- Altera's Stratix V® : **Bandwidth and performance**. Higher bandwidth requirements while meeting cost and power budgets.



FPGA Evolution

FPGA Fabric

- Altera's Stratix V® : **Bandwidth and performance**. Higher bandwidth requirements while meeting cost and power budgets.



- Highest system integration: Embedded **HardCopy® Blocks** and integrated **hard intellectual property (IP)** in transceivers and core. PCI Express Gen3 and 40G/100G Ethernet. High-performance, high-precision DSP. Enhanced logic fabric with 1,100K LEs, 53 Mb RAM, and 3,680 18x18 multipliers.

- Flexibility: Fine-grained **partial reconfiguration** (core) and **dynamic reconfiguration** (transceivers) for multiprotocol client support and configuration via PCI Express (CvPCIe).

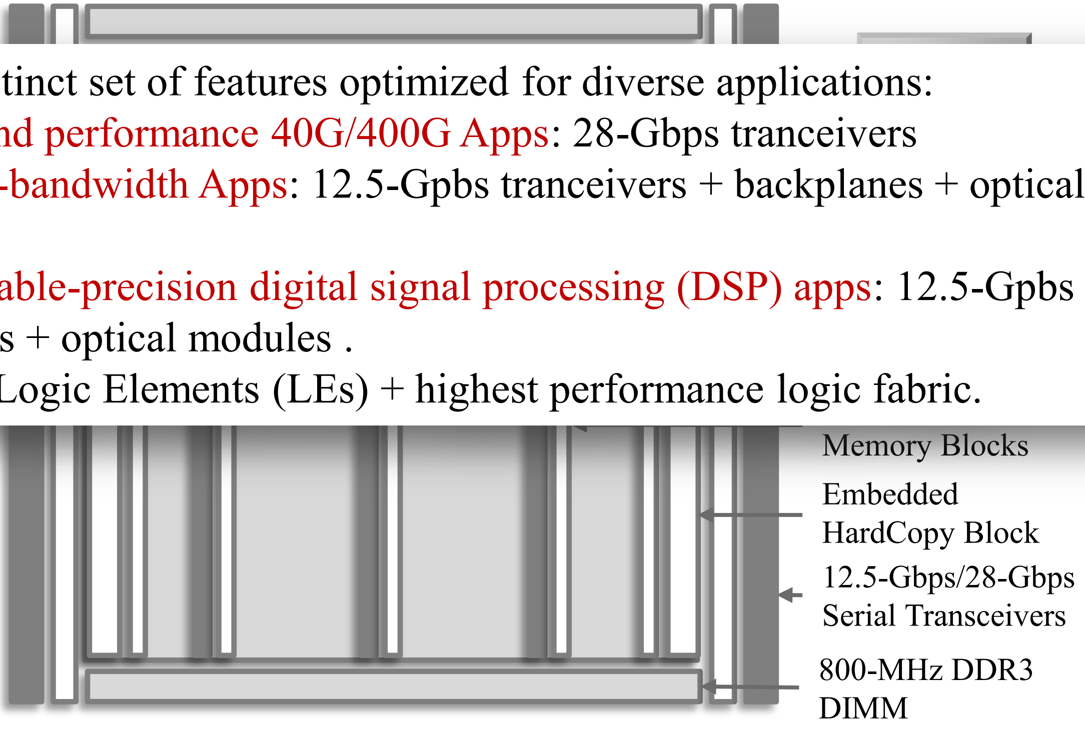
- Power: Reduce total power by 30% compared to previous-generation devices.

FPGA Evolution

FPGA Fabric

- Altera's Stratix V® : **domain/application oriented architectures**.

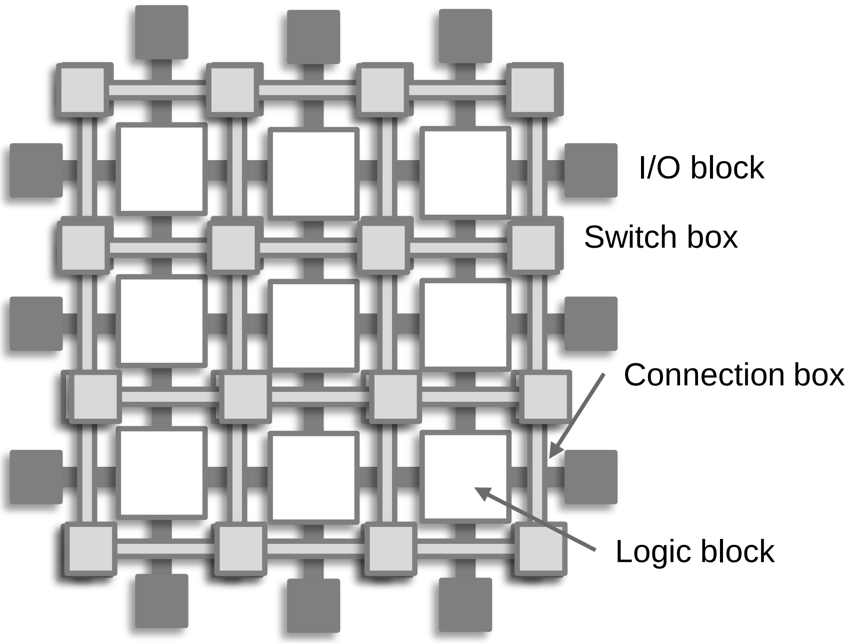
- Variants/Families: A distinct set of features optimized for diverse applications:
- Ultra-high bandwidth and performance 40G/400G Apps: 28-Gbps tranceivers
- High-performance high-bandwidth Apps: 12.5-Gpbs tranceivers + backplanes + optical modules).
- High-performance, variable-precision digital signal processing (DSP) apps: 12.5-Gpbs tranceivers + backplanes + optical modules .
- ASIC prototyping: 1M Logic Elements (LEs) + highest performance logic fabric.



FPGA Evolution

Interconnection architecture

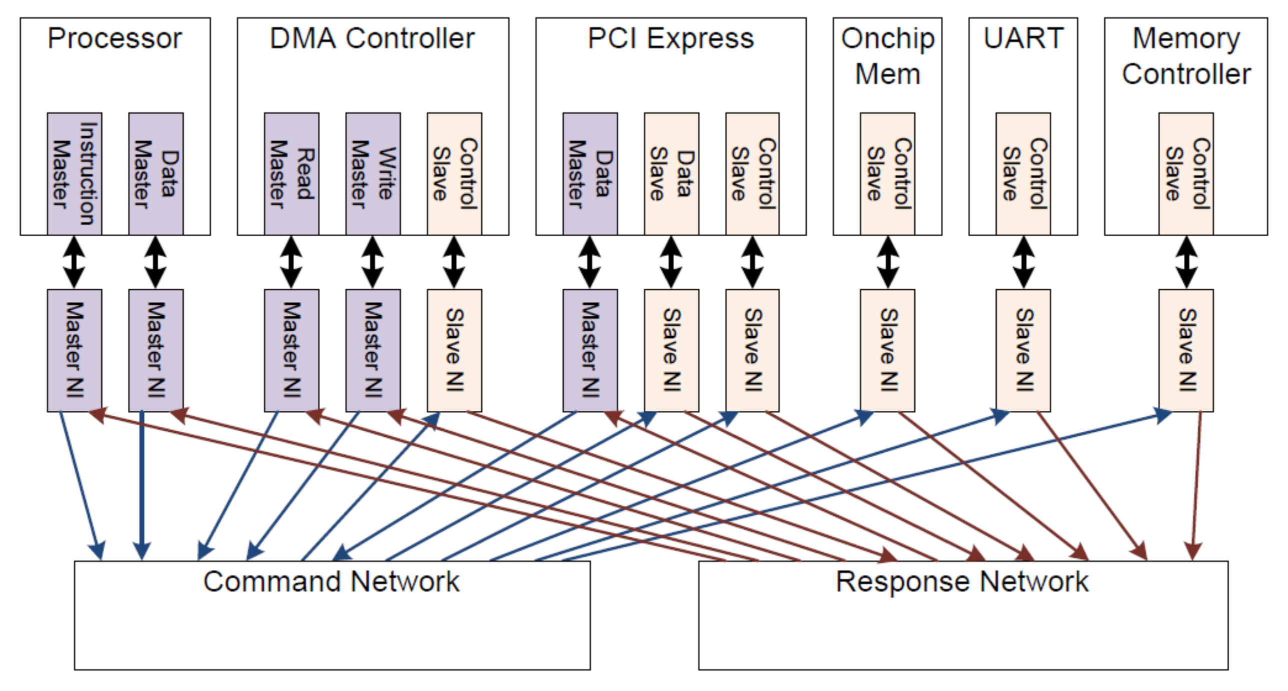
- Internal data & logic connections
- **Global** & **local** interconnections (**Asynchronous** & **Synchronous**)
- **Data transport lines**: wire segment + connection box
- **Programmable Switch boxes**: data/logic connection



FPGA Evolution

Interconnection architecture

- **Network topology** with separated networks: command/service and response/transport.
- Higher **concurrency** and lower resource utilisation.

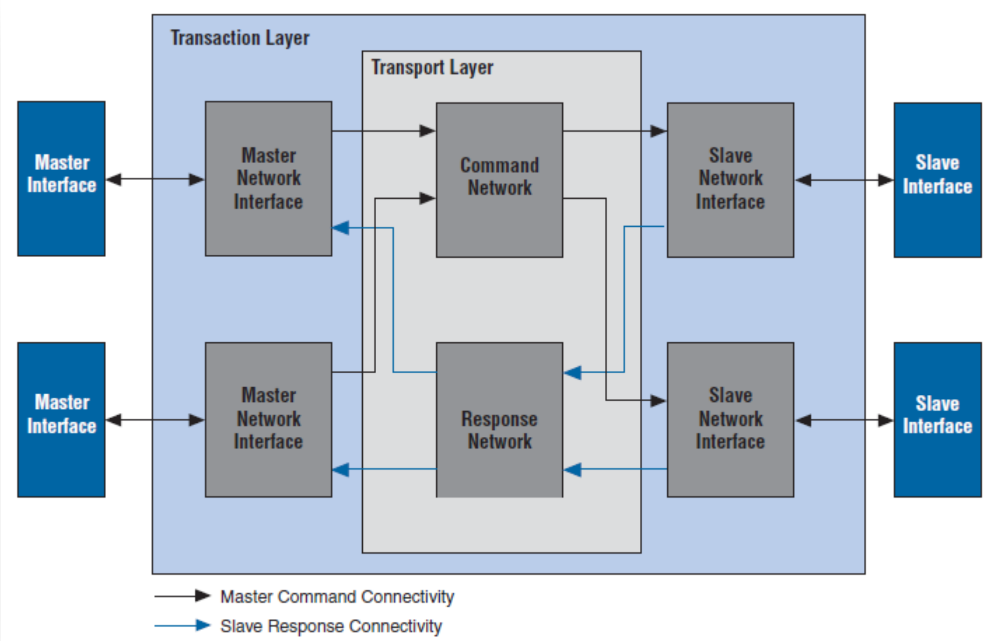


[1] Altera, Qsys, 2011

FPGA Evolution

Interconnection architecture

- **Protocol stack**: transaction/transport layer & packet encapsulation.
- **Independent layer** design and optimization: Altera's **Qsys interconnect** [1]

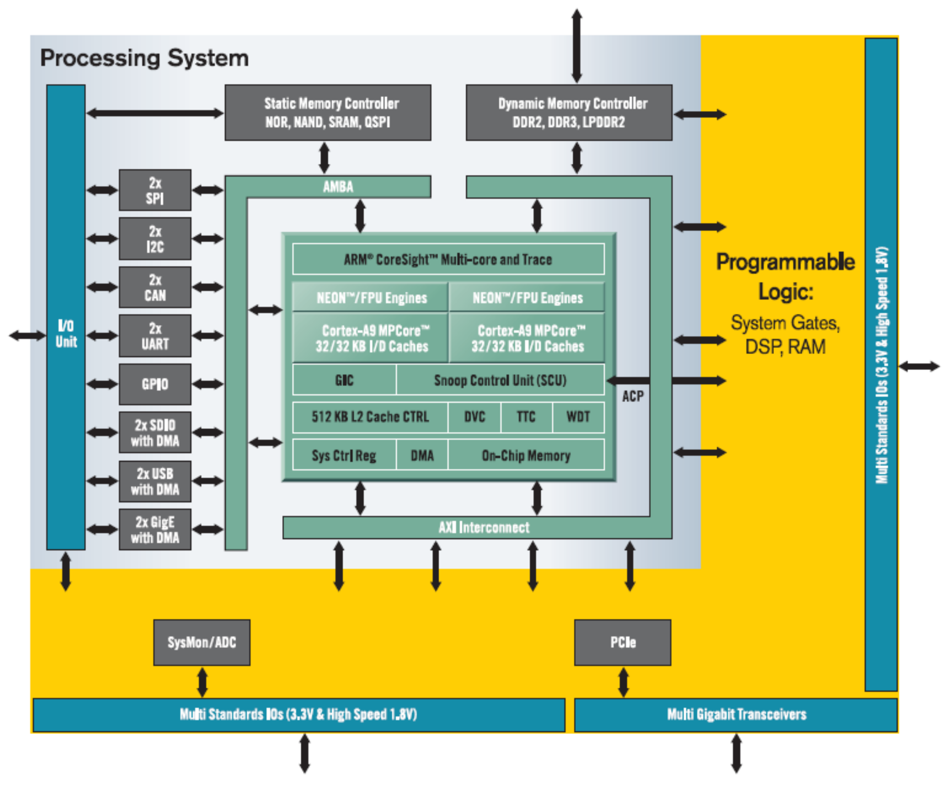


[1] Altera, Qsys, 2011

FPGA Evolution

Design Platforms

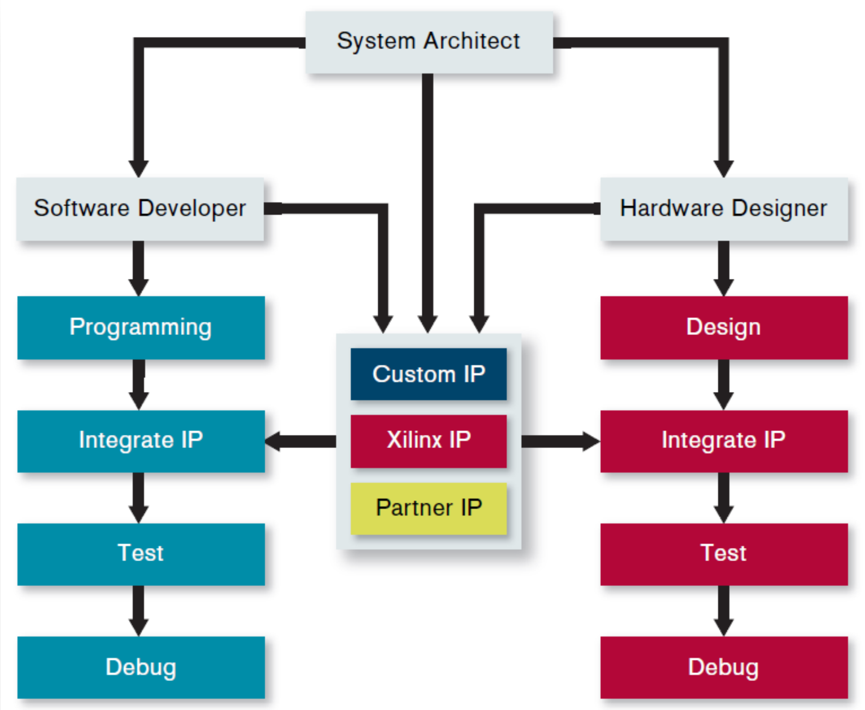
- **Xilinx Extensible Processing Platform Zynq™-7000**
- Increasingly complex functions and exploding demand for signal-processing performance.
- Reducing costs through hardware and software design reuse across a common platform for multiple products.
- Size, power, throughput and cost advantages in a flexible single-chip architecture.
- Xilinx 7 series programmable Logic.
- A defined SW programming Model.



FPGA Evolution

Unified Design Flow

- **Xilinx Extensible Architecture**
- Software-centric development flow with parallel hardware design.
- Operating Systems: Linux, VxWorks + uC-OSII MiCrium.
- Processor system: caches, memory controllers, and peripherals (dynamically reconfigurable) available at power-up.



AMBA (advanced microprocessor bus architecture)
http://www.xilinx.com/support/documentation/white_papers/wp369_Extensible_Processing_Platform_Overview.pdf

ASIC Evolution

The highest performance available for embedded systems

- “Without further improvements from architecture, **performance increases have dropped sharply** to about 20% per year.” [1]

The best cost/performance in high-volume products

- “Since 2003, **development cost has been the limiting factor** to actual ASIC design size, not available gates, causing a **Hardware Design Productivity Gap**.” [2]

Verification prior tape-out is mandatory

- “**Verification cost goes up much faster than design size**. Doubling the number of gates doubles the work per cycle, but also as much as doubles the number of verification cycles needed to cover all the possible gate-to-gate relationships.” [3]

Serial software model

- “No one knows how to design a 15-GHz processor, so the other option is to **retrain all the software developers**.” [4]

[1] M. Butts, Multicore and Massively Parallel Platforms and Moore's Law Scalability, 2008.

[2] G. Smith, The Crisis of Complexity, Dataquest DAC Briefing, 2003

[3] J. Hennessy & D. Patterson, Computer Architecture: A Quantitative Approach, 2006.

[4] P. Magarshack, Improving SoC Design Quality through a Reproducible Design Flow, 2002

ASIC Evolution

Productivity increase

- **Abstraction level**: RTL (transistors and gates) + ESL (system-level)
- **High Level Languages**: hardware (SystemC and HDLs) and software (serial software model)
- **EDA/ESL tools**: verification, validation, synthesis (partition, synchronization)

Power consumption

- **Power management**: frequency, voltage, technology.

Flexibility

- **Regularity**: communication, interconnection, components. Data very densely structured in irregular and unpredictable ways.
- **Software degree of freedom**: SDR, evolution of standards

High-performance embedded systems aimed at streaming media

- **Data flowing continuously at gigabytes per second processed in real time**. Streaming applications such as video codecs, software-defined radio (SDR), real-time sensor processing or image recognition.

1

ASIC Evolution

ASSP (Application Specific Standard Product) [1]

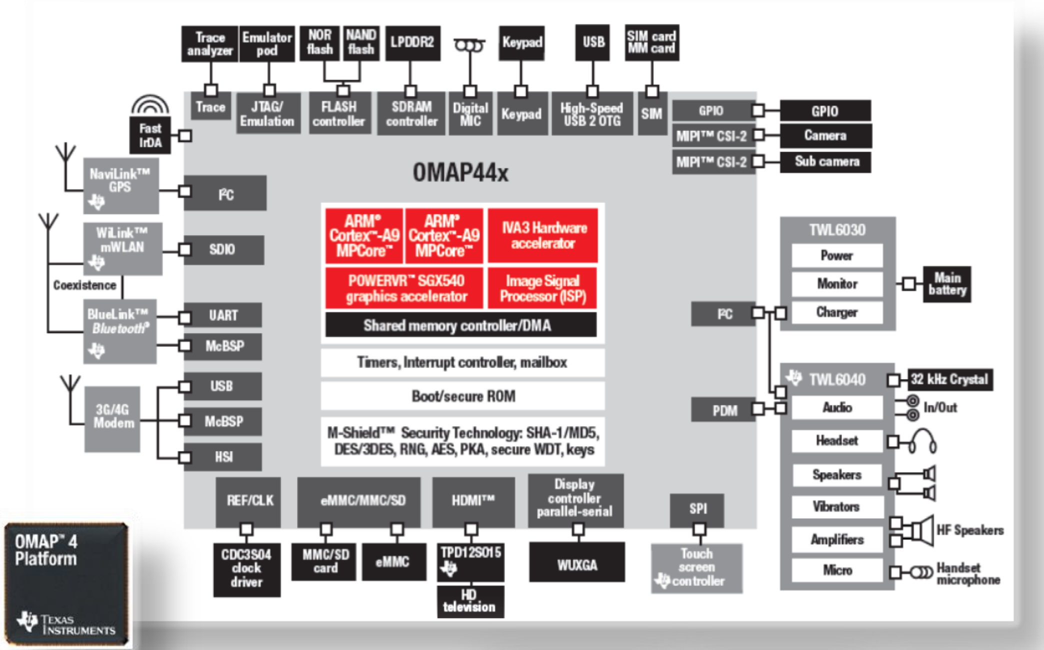
- **TI OMAP 4**. Targets smart phones and mobile internet devices. Low leakage 45 nm process technology. Supports full 1080p multi-standard HD video.
- **Dual SMP processor**: 2 ARM Cortex-A9 cores (720 MHz and 1 GHz) + NEON multimedia extensions.
- **Programmable video engine** (TI IVA3) programmable C64x DSP core + video codec accelerators.
- **Graphics engine**:

  Imagination Technologies'

  POWERVR SGX540.
- **Image signal processing**:

  TI ISP.
- **Audio backend engine**:

  TI ABE.

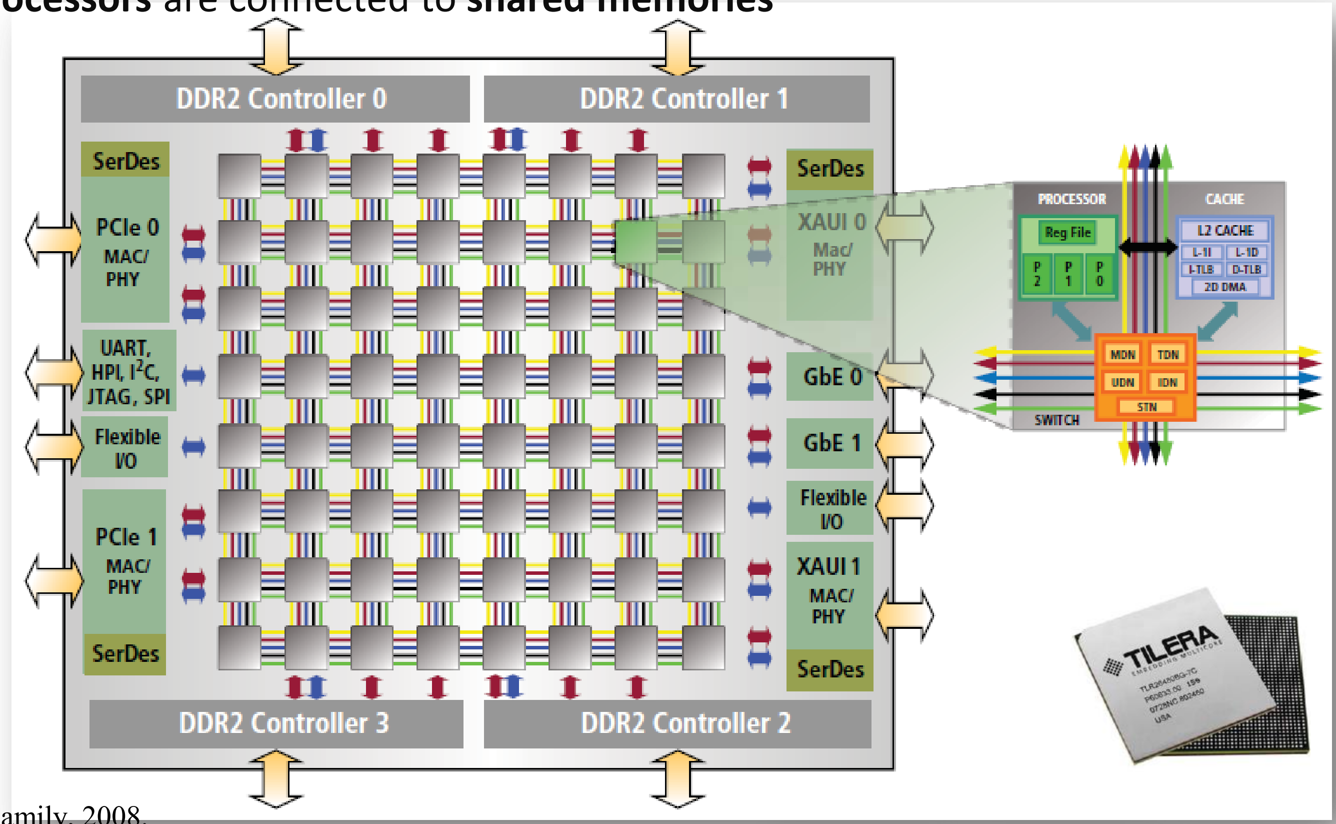


[1] Texas Instruments OMAP 44x family, 2009.

ASIC Evolution

SMP (Symmetric Multiprocessing)[1]

- **Symmetric multiprocessor**: each processor has similar access to a **common memory space**
- **Multiple processors** are connected to **shared memories**

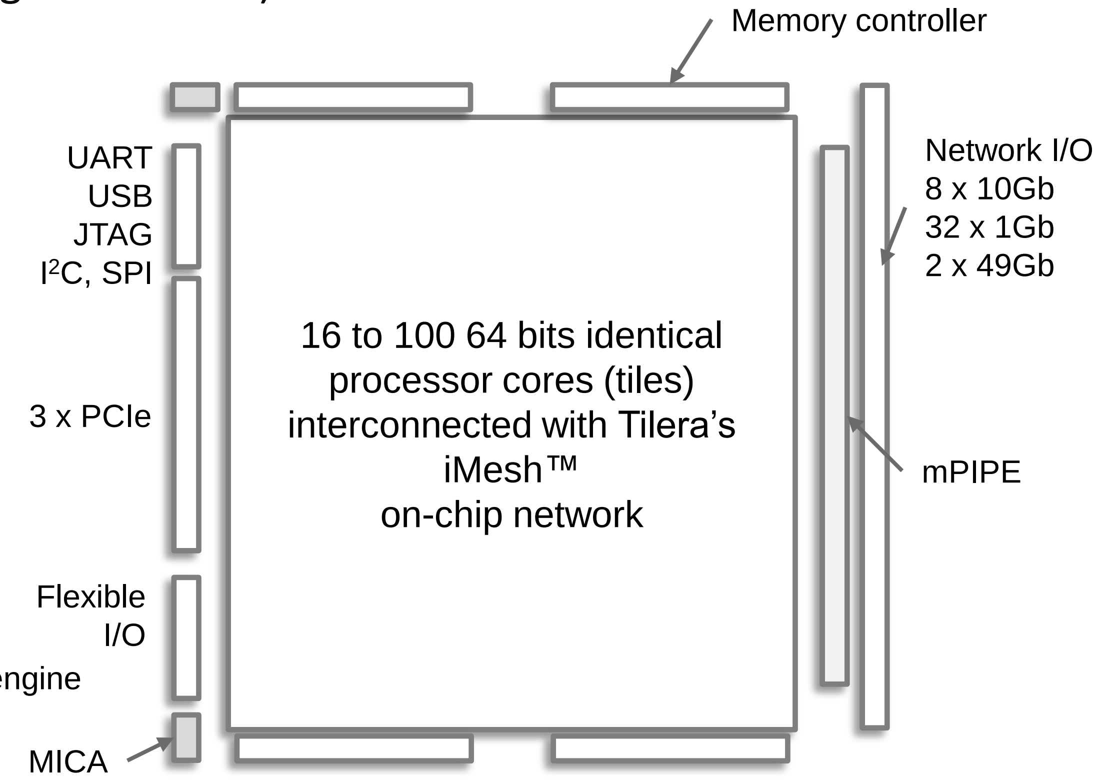


[1] Tilera, Inc., Tile64 Processor Family, 2008.

ASIC Evolution

SMP (Symmetric Multiprocessing)[1]

- The **SMP programming model** has **task-level** and **thread-level** parallelism (separate threads of execution, and explicit manage of data sharing and synchronization among the threads).


Memory controller
UART
USB
JTAG
I²C, SPI
3 x PCIe
Flexible I/O
MICA
16 to 100 64 bits identical processor cores (tiles) interconnected with Tilera's iMesh™ on-chip network
Network I/O
8 x 10Gb
32 x 1Gb
2 x 49Gb
mPIPE

MICA: Multistream iMesh Crypto Accelerator engine
mPIPE: Multicore Programmable Intelligent Packet Engine (Wire-speed packet engine)

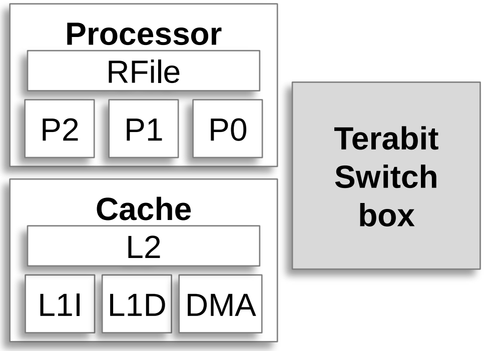
[1] Tilera, Inc., Tile64 Processor Family, 2008.

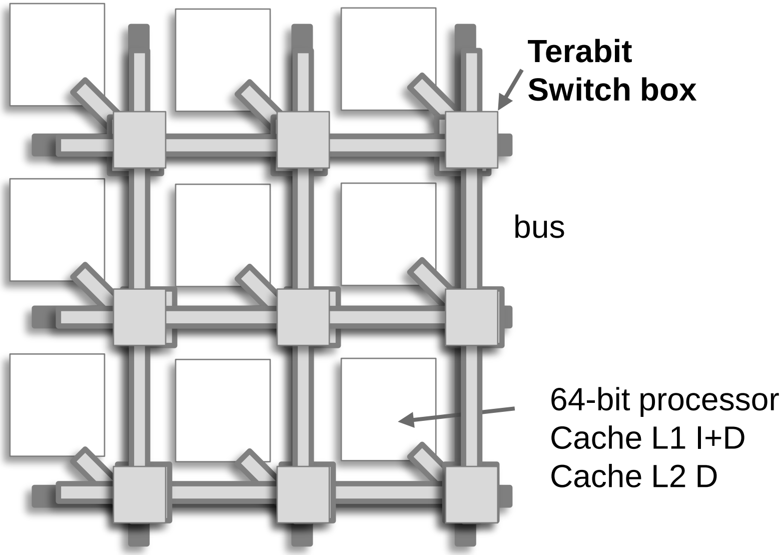
ASIC Evolution

SMP (Symmetric Multiprocessing)[1]

- Each processor has its own **single or multi-level cache**, connected through an interconnect to other processor caches, main memory and external devices.
- An on-chip packet-switched network (i.e. **switch fabric**) provides direct communication paths. Communication paths and switches are shared.

- 3-way VLIW pipeline (instruction level parallelism)
- 31 Tbps on-chip mesh interconnect
- 50 Gbps I/O bandwidth
- 5MB on-chip cache
- 15-22W@ 700MHz



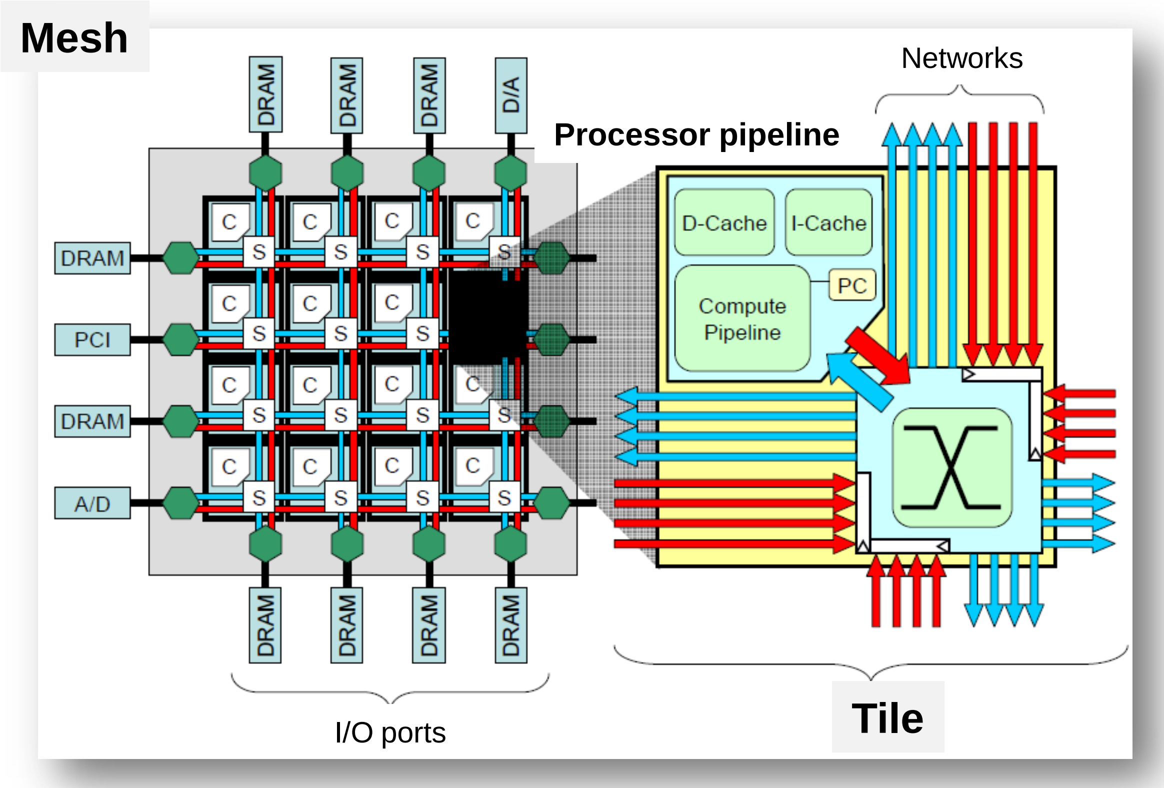


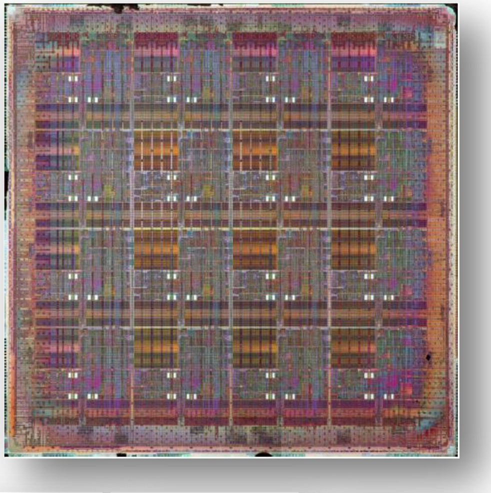
[1] Tilera, Inc., Tile64 Processor Family, 2008.

ASIC Evolution

SMP (Symmetric Multiprocessing)[1]

- **Raw Tiled Processor**: 16 scalar cores. 4 **mesh networks** (2 static + 2 dynamic I/O devices wormhole communication). **Message passing** communication.



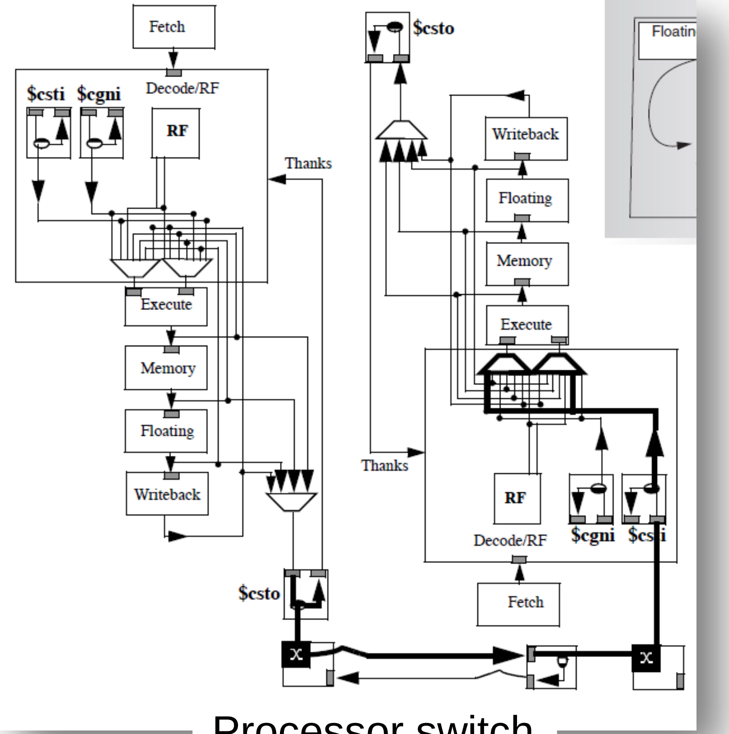


[1] MIT RAW, 2007.

ASIC Evolution

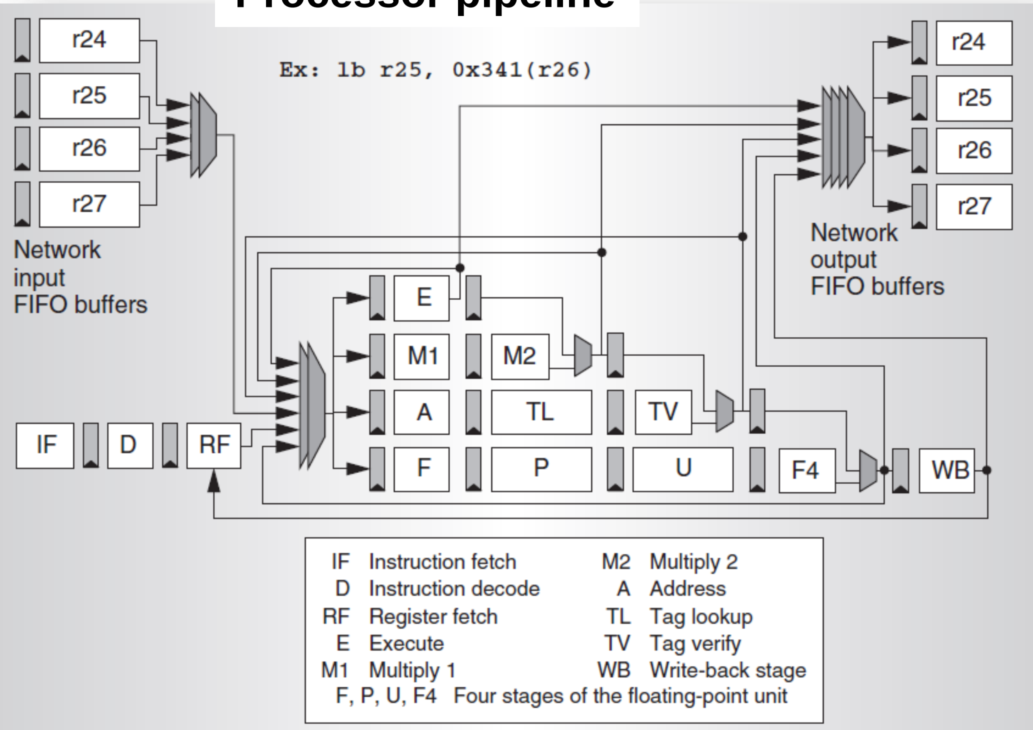
SMP (Symmetric Multiprocessing)[1]

- **Raw Tiled Processor**: **Processor pipeline**. 8-stage pipeline. 2-cycle integer mul. 4-cycle FPU. A RISC processor per tile.



Processor switch

Processor pipeline



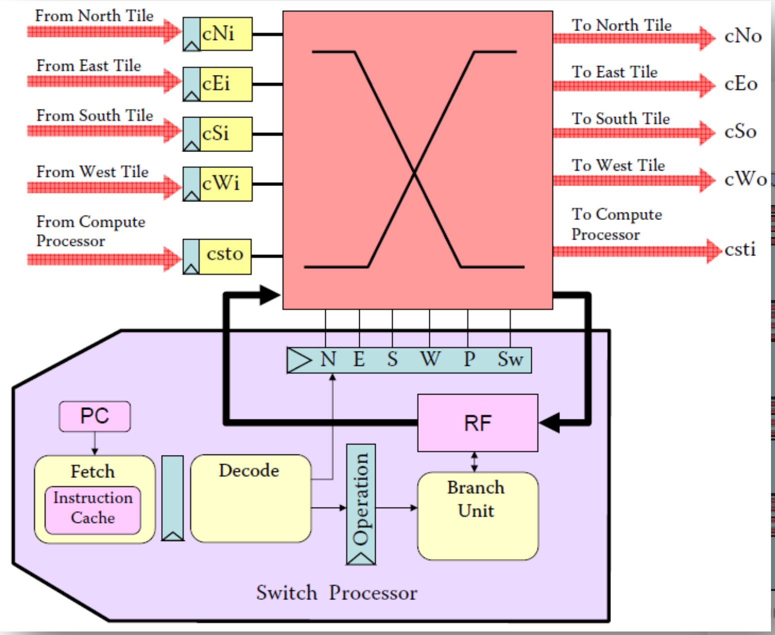
IF	Instruction fetch	M2	Multiply 2
D	Instruction decode	A	Address
RF	Register fetch	TL	Tag lookup
E	Execute	TV	Tag verify
M1	Multiply 1	WB	Write-back stage
F, P, U, F4	Four stages of the floating-point unit		

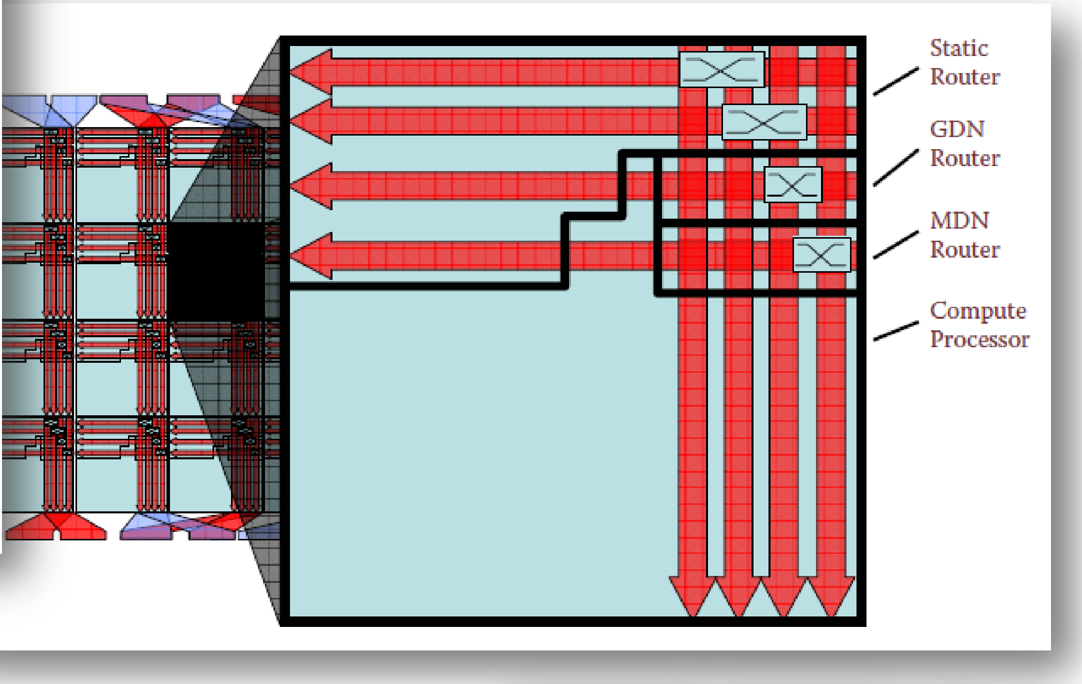
[1] MIT RAW, 2007.

ASIC Evolution

SMP (Symmetric Multiprocessing)[1]

- **Raw Tiled Processor**: **Static router**. Single cycle latency. Mapping fine-grain ILP across tiles .





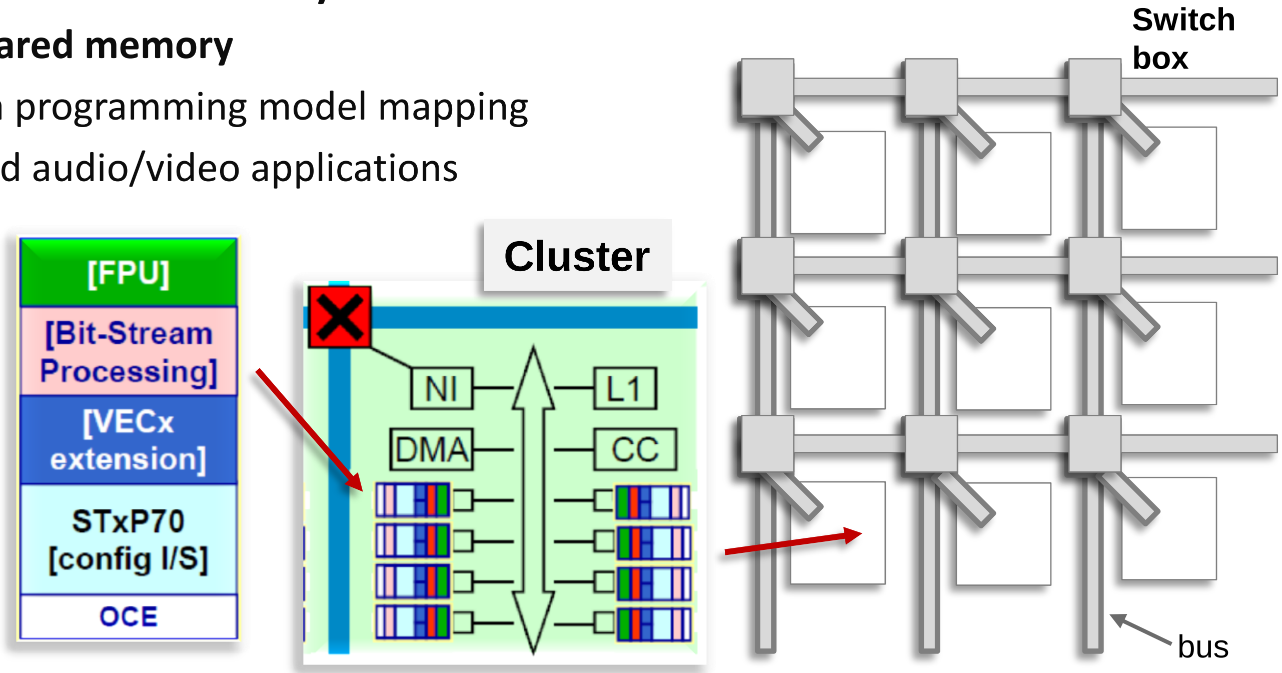
[1] MIT RAW, 2007.

ASIC Evolution

Multicore Fabric - Platform 2012 ST RISP

- **Massively parallel array of clusters** (configurable cores + optional Hw processing elements) and memories
- **GALS interconnection** (Globally Asynchronous – Locally Synchronous)
- **Globally distributed memory**
- **Locally shared memory**
- Requires a programming model mapping
- control and audio/video applications

- Up to 32 Clusters
- Async NoC (GALS)
- 1-16 configurable cores
- Optional Hw PEs
- 100 GOPs to several TOPS



[1] P. Paulin, Towards Multicore Programming, 2011

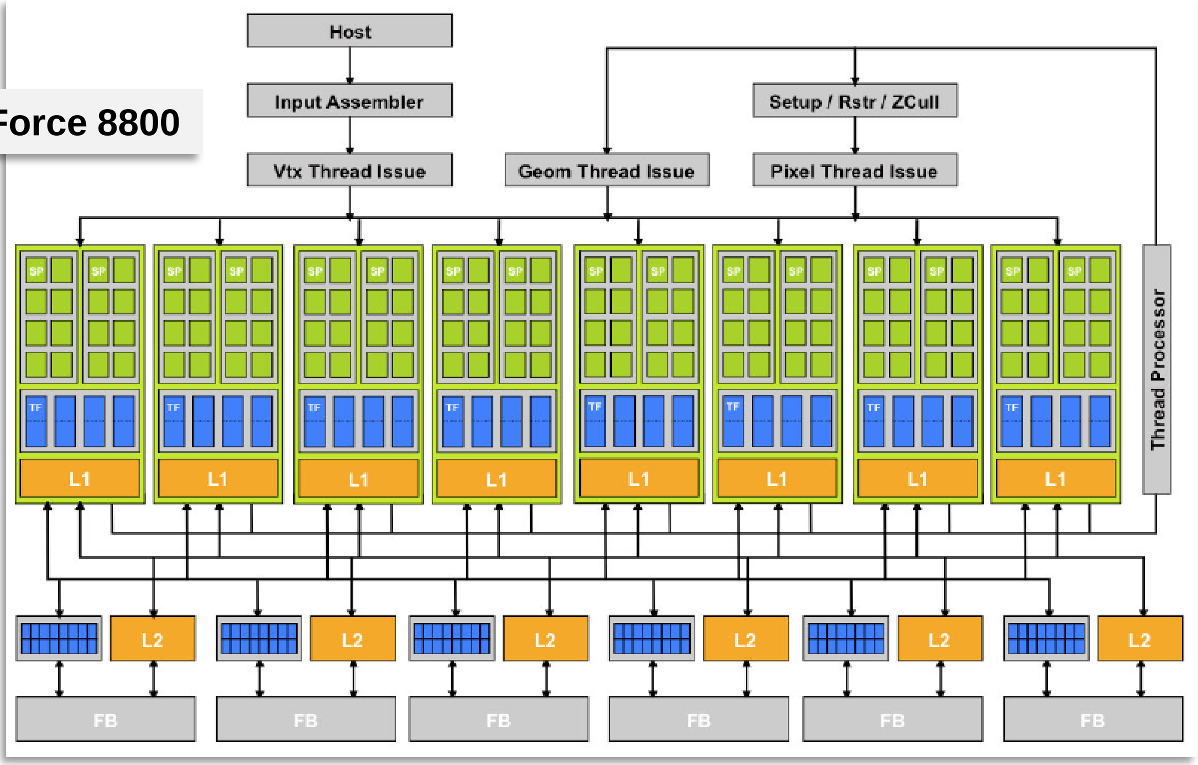
ASIC Evolution

Single Instruction Multiple Data (SIMD)[1]

- Multiple data paths controlled by a **single instruction stream**
- Single-chip forms: graphics processing units (GPU), IBM's Cell processor, and Stream Processing, Inc.'s Storm DSP.

NVIDIA GeForce 8800

- 128 PEs – 8 shader cores

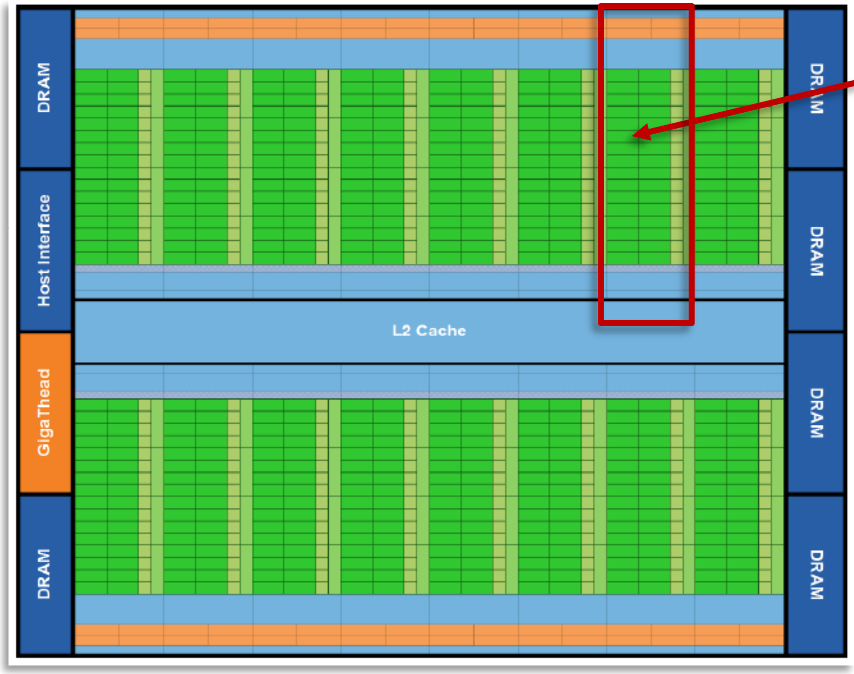


[1] NVIDIA GeForce 8800 unified shader architecture, 2006

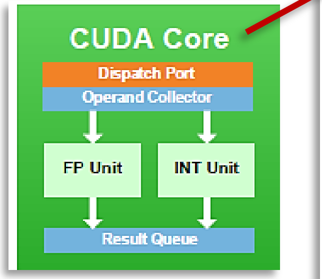
ASIC Evolution

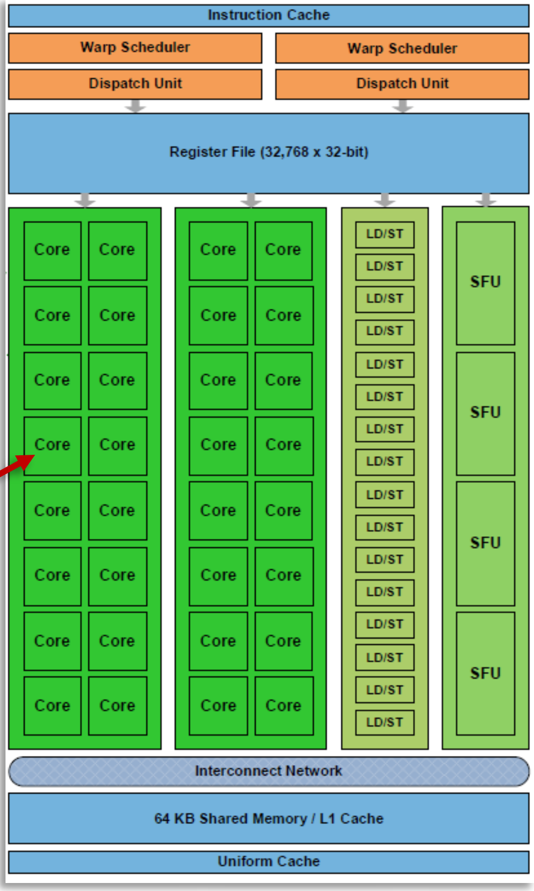
Single Instruction Multiple Data (SIMD)[1]

- **NVIDIA CUDA** (Compute Unified Device Architecture)
- General purpose graphic processing units functioning as parallel processing engines (stream processors - over 120 handling single tasks).



NVIDIA Fermi



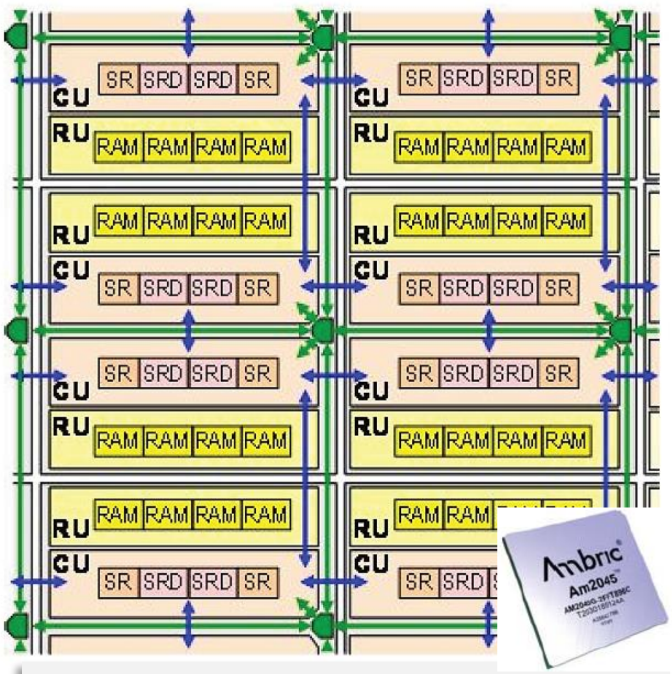


[1] NVIDIA Fermi, 2009 [multiple streaming multiprocessors SMs of 32 cores each]
SFU Special Function Unit

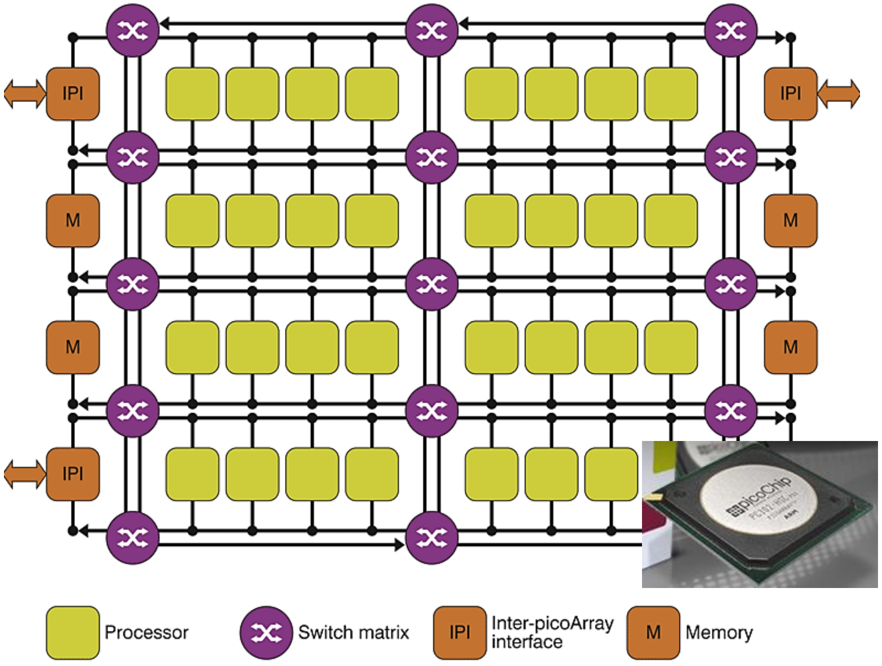
ASIC Evolution

MPPA (Massively Parallel Processor Array) [2,3]

- **MIMD** (Multiple Instruction streams, Multiple Data) architecture, **distributed memory** accessed locally (not shared globally)



Ambric: Brics & interconnect



PicoChip: PicoArray concept

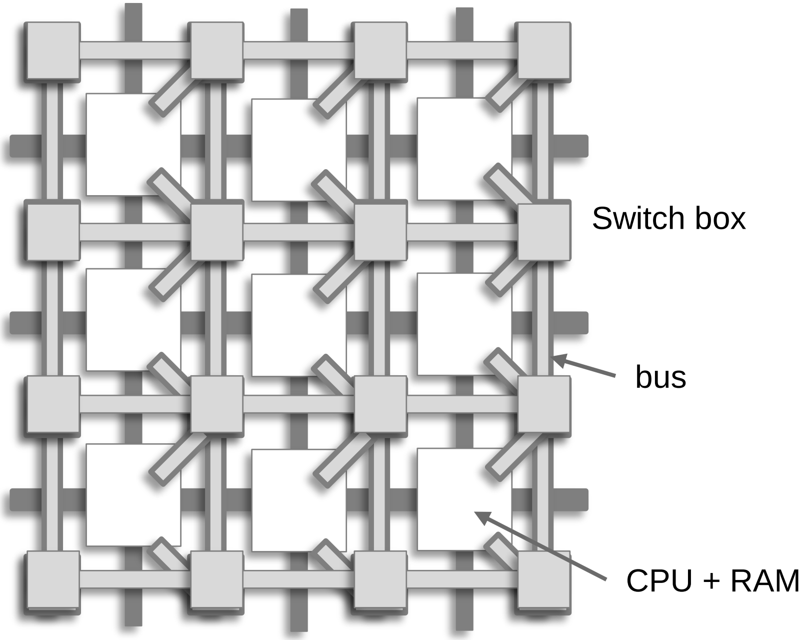
[2] www.ambric.info *MPPAs Technology Overview 2008*

[3] www.picochip.com *multi-core DSP architecture 2010*

ASIC Evolution

MPPA (Massively Parallel Processor Array)[1]

- A **massively parallel array** of **CPUs** and **memories**, interconnected by a **2D Mesh** configurable interconnect of word-wide buses
- High throughput embedded system with low cost, power efficiency, and deterministic behavior



[1] M. Butts, *Multicore and Massively Parallel Platforms and Moore's Law Scalability, 2008*

ASIC Evolution

Parallel Platform Characteristics for Embedded Applications[1]

- **Suitability**: How well-suited is its architecture for the full range of **high-performance embedded computing applications**?
- **Efficiency**: processors' potential **performance** achieved? **Energy** and **cost**?
- **Development Effort**: **architecture**, **programming** and **debugging** effort?

Key elements in parallel computing are how the programming model handles inter-processor communication and synchronization:

- **Communication**: how easily can processors pass **data and control** from stage to stage, correctly and without interfering with each other?
- **Synchronization**: how do processors **coordinate actions** with one another, to maintain the correct workflow?
- **Scalability**: as Moore's Law continues its march, will the hardware architecture and application development effort **scale up to a massively parallel** system of hundreds or thousands of processors?

[1] M. Butts, *Multicore and Massively Parallel Platforms and Moore's Law Scalability, 2008*

ASIC Evolution

Parallel Platform Characteristics for Embedded Applications[1]

	SMP	SIMD	MPPA
Suitability	Limited (GP)	Limited (scientific)	Good (Embedded)
Efficiency	Poor (expensive comm.)	Poor (only regularity)	Good (very flexible)
Development	Poor (multithreading)	Poor (only regularity)	Good (testable & reliable)
Communication	Poor (cache)	Good	Good (direct)
Syncronization	Poor	Good (single thread)	Good (built-in)
Scalability	Poor	Poor (only data flow)	Good (both Hw/Sw)

[1] M. Butts, *Multicore and Massively Parallel Platforms and Moore's Law Scalability, 2008*

ASIC Evolution

Massively parallel processors: Who's still alive?[1]

Company	Product	Comments
Ambric	AM2000 family-344 Processors	Was shipping video chips, but backing dried up
Aspex Semiconductor (U.K.)	Linedancer 4,096 Processors	Silicon killed; now marketing IP for video
BrightScale	BA 1024 Video Processor	Was Connex Technology; now marketing IP
ClearSpeed Technologies	Multithread 96-element array processor	Now selling as subsystems
Coherent Logix, Inc.	hx2100 HyperX-based DSP	Still in development
CPU Technology, Inc.	Acalis7 Field-Programmable MultiCores	Actively marketing
Element CXI	Reconfigurable Array	Showing silicon-actively marketing
Elixent/Panasonic (U.K.)	D-Fabrix Array	Now Captive; status unknown
IMEC (Belgium)	ADRES: Coarse Grain Array for VLIW's	Licensable IP, actively marketing
Intellasys	Scalable Embedded Array 24 processors	Actively marketing
IP Flex (Japan)	DAP/DNA-955 16-bit processors	Now shipping for video
MathStar	Field Programmable Object Array	Out of business
Motorola Labs	Reconfig. Streaming Vector Processor	Development ceased
NEC (Japan)	Dynamically Reconfigurable Logic Engine	Development appears to have ceased
PACT XPP Technologies	XPP 3C-64 processors	Actively marketing IP
PicoChip Designs (U.K.)	picoArray Massively Parallel Array	**Shipping in volume**
Plurality (Israel)	Hypercore Processor: 16-256 cores	In Development
Rapport Inc.	Kilocore KC-256 with 256 processors	Appears to have folded
Recore (Netherlands)	Montium Tile Processors	Licensable IP-status unknown
Silicon Hive (Netherlands)	Moustique Block Accelerators	**Licensable IP-actively marketing**
Stream Processors Inc.	Storm-1 Family-80 32-bit ALUs	Surveillance Video-actively marketing
Tabula	Unannounced	**In Stealth Mode**
Tilera	TilePro36 & 64	**1st generation now shipping**

[1] K.Williston Massively parallel processors: Who's still alive?, 2009

Architectures	Computing Performance	Flexibility	Power Consumption	Compilation Time	Application Development Time

General research question:

Can we improve the existent solutions in order to obtain a flexible architecture for image processing that can meet real-time performance with low power and low compilation & development times?

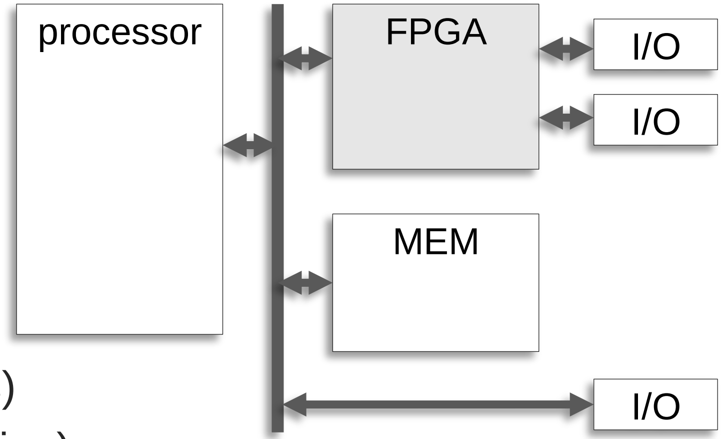
Architecture Considerations

- ❑ Platform-based design: generic infrastructure (flexible) + dedicated co-processors or accelerators
- ❑ Derivative design: remove and keep only what is required
- ❑ Multi-core regular (+ shared memory): flexible + easy to use
- ❑ Multi-core irregular (heterogeneous or hybrid): dedicated + performance
- ❑ Parallel Pipeline: FPGA alternative
- ❑ Offload Code: SW alternative
- ❑ Shared Memory: slow but flexible
- ❑ Tools: from sequential to multithread + ESL + HLS + LS

Platform-based design

Single Core: standard processor + FPGA (accelerator)

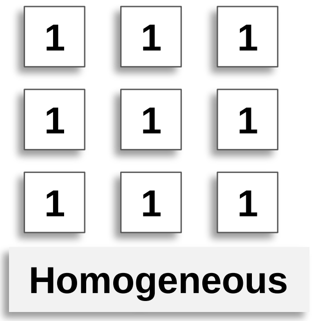
- ❑ **Standardization** due to High level tools & languages (gains in design workflow)
- ❑ **ILP & loop-level parallelism** (CISC & accelerators), VLIW (bundle), TLP (RISC)
- ❑ **SW flexibility**: reuse existing code + libraries
- ❑ **SW common bus** & standard I/O
- ❑ **Extendibility** due to dedicated HW
- ❑ SW to **HW code migration**
- ❑ HW **determinism & reliability**
- ❑ HW **performance** + **energy efficiency**
- ❑ SW **instruction set extended** (custom algorithms)
- ❑ HW **basic interface** (protocol, FIFO, synchronization)
- ❑ **Not parallel nor scalable** (go for modular tile-based architectures)



Multi-core

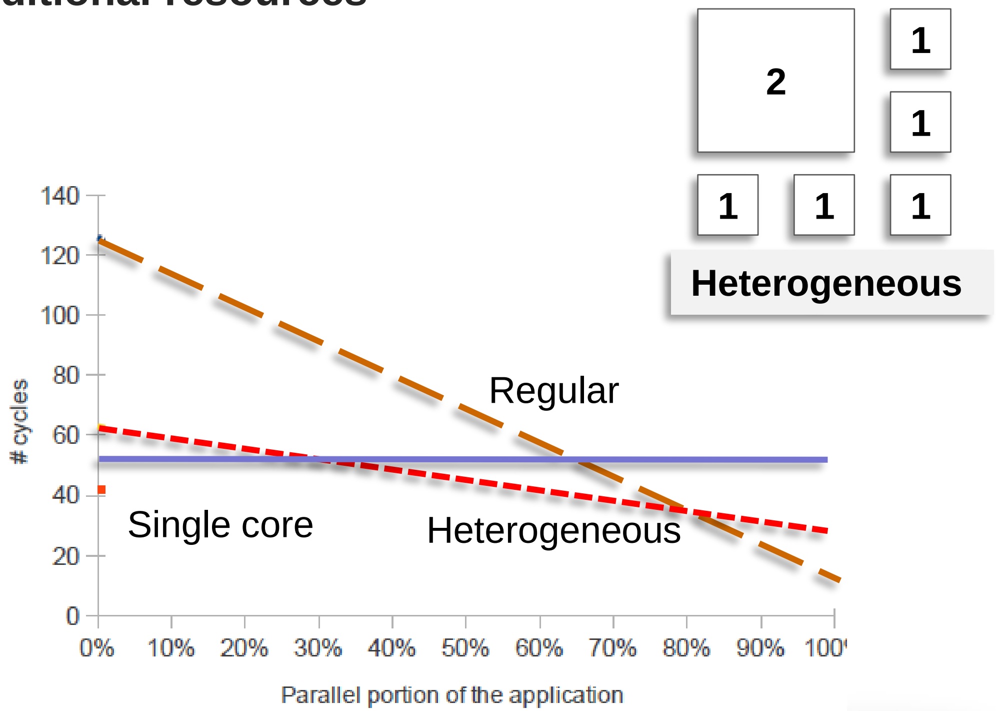
Regular

- □ **Scalability**
- □ Equal processing power & complexity (Amdahl+Gustafson)
- □ Processors optimal execution reducing program's critical path
- □ **Sequential parts slow down overall performance**
- □ **An intensive task will not receive additional resources**

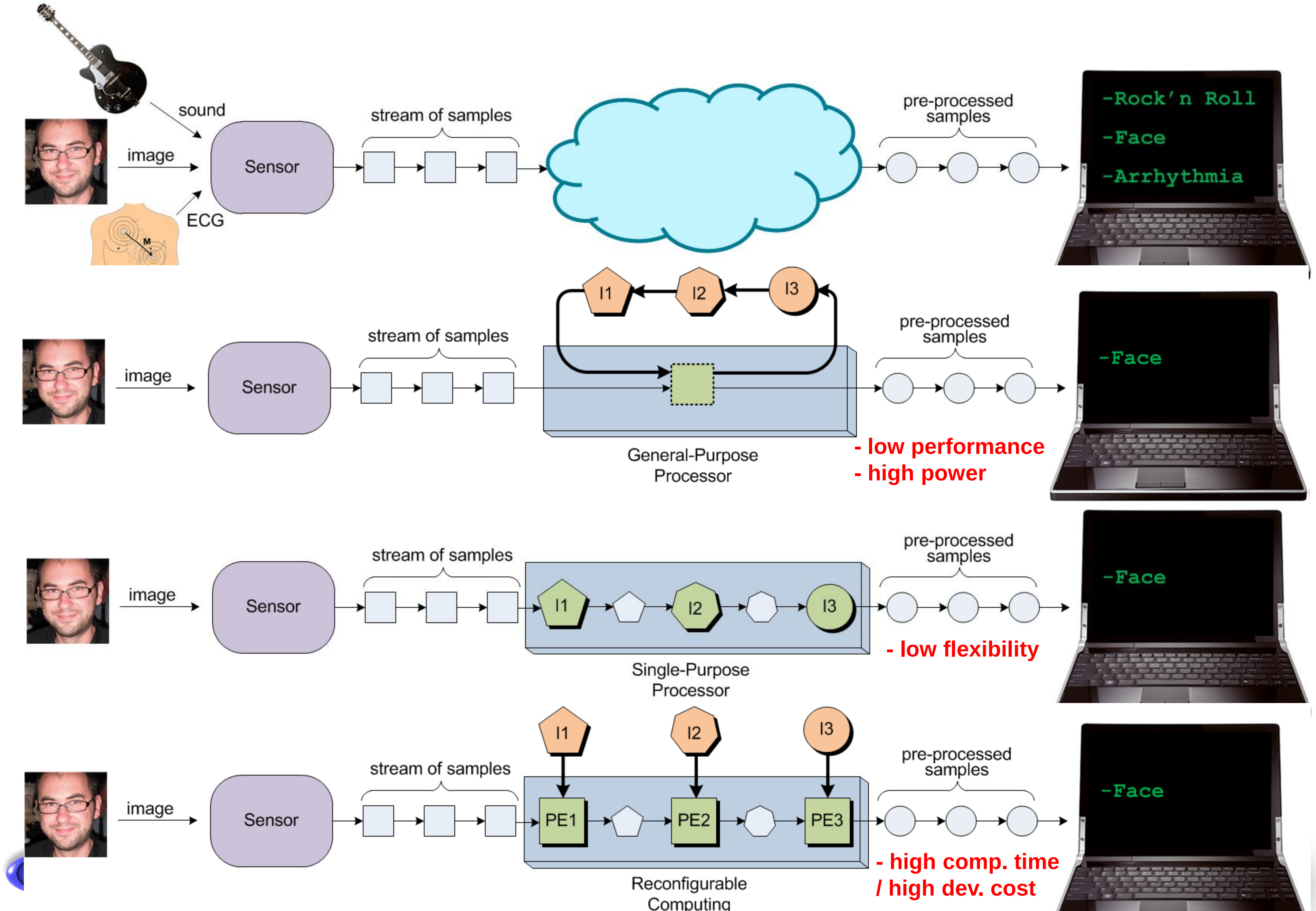


Irregular

- □ **Heterogeneous** processing power & complexity (Pollack: performance roughly proportional to the square root of complexity increase)
- □ **Higher performance for sequential tasks**
- □ **Balanced critical path**



Pipeline



- low performance
- high power

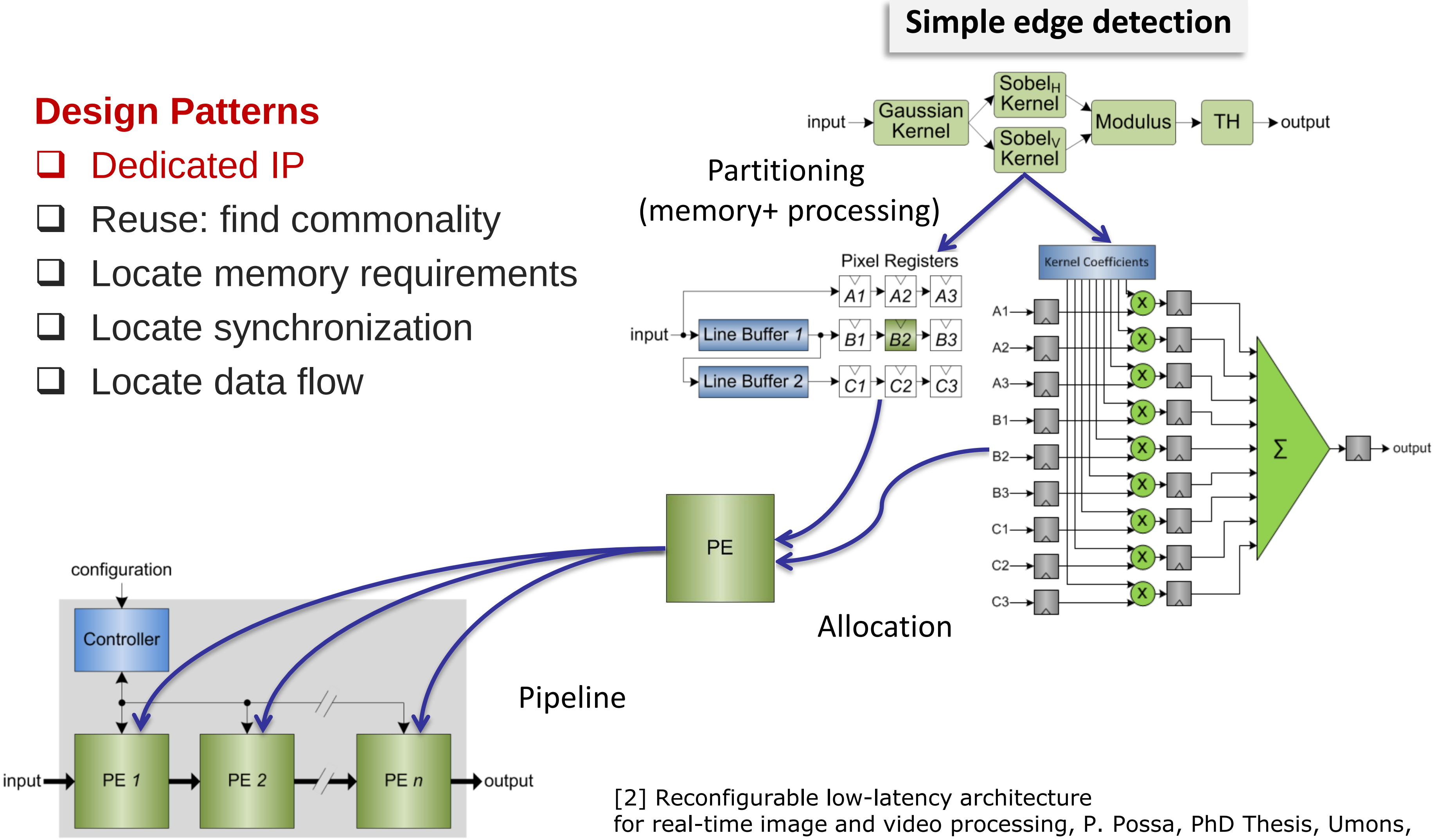
- low flexibility

- high comp. time / high dev. cost

Pipeline

Design Patterns

- ❑ Dedicated IP
- ❑ Reuse: find commonality
- ❑ Locate memory requirements
- ❑ Locate synchronization
- ❑ Locate data flow



Simple edge detection

Partitioning (memory+ processing)

Allocation

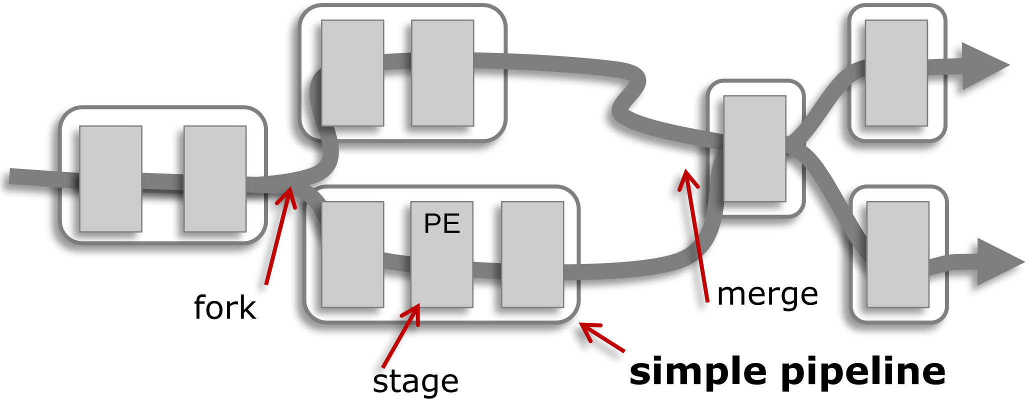
Pipeline

[2] Reconfigurable low-latency architecture for real-time image and video processing, P. Possa, PhD Thesis, Umons, Belgium, 20013

Parallel Pipeline

A large program partitioned and each piece assigned to a stage in the pipeline

- ❑ Each stage does its portion and then hands the packet on to the next stage
- ❑ A stage will have as many parallel engines as required to meet throughput requirements
- ❑ Each engine in the stage will execute an identical program on a different packet
- ❑ **Particularly well suited to packet-processing data-plane applications**
- ❑ **Load-balance: programmable & reconfigurable PEs (Processing Elements)**

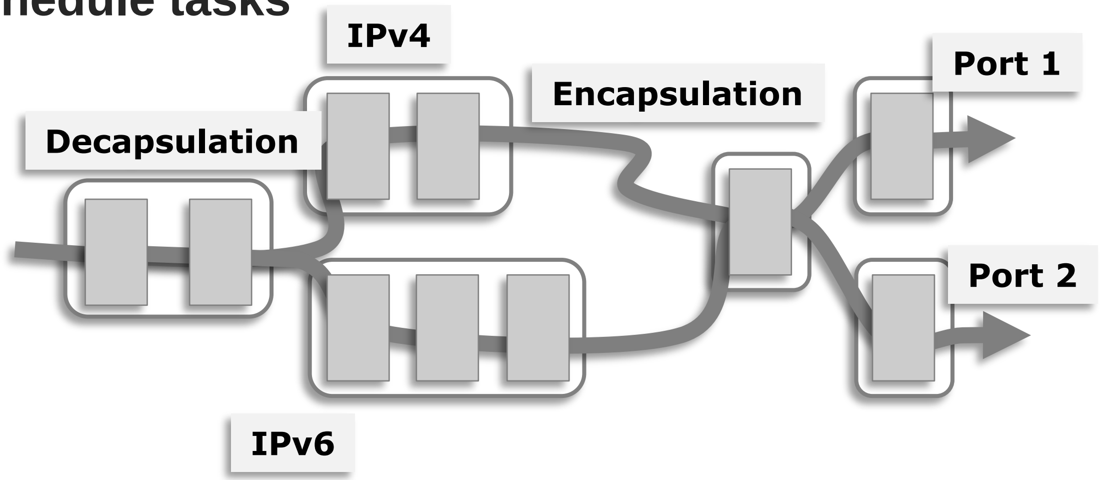


Parallel Pipeline

Fork & Merge: pipelines may need to fork and merge, which allows multiple pipelines to take on different tasks

- ❑ **Routing**: many applications will require more than a simple pipeline (fork & merge)
- ❑ **Load-balancing considerations**: when a packet needs to be sent to one of two ports
- ❑ **Reconfigurable processing elements**: stages are fed by a queue of tasks to be executed

- ❑ **An OS is not needed to schedule tasks**
- ❑ **Private/local memory**

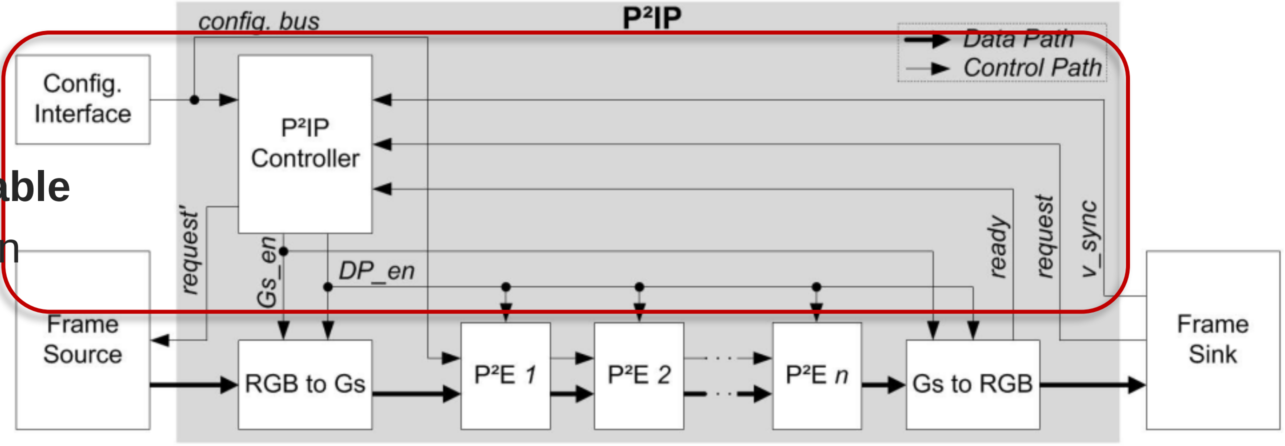


[1] Building Blocks Simplify Multicore in FPGAs, B. Moyer, Chip Design Magazine, 2007

Parallel Pipeline

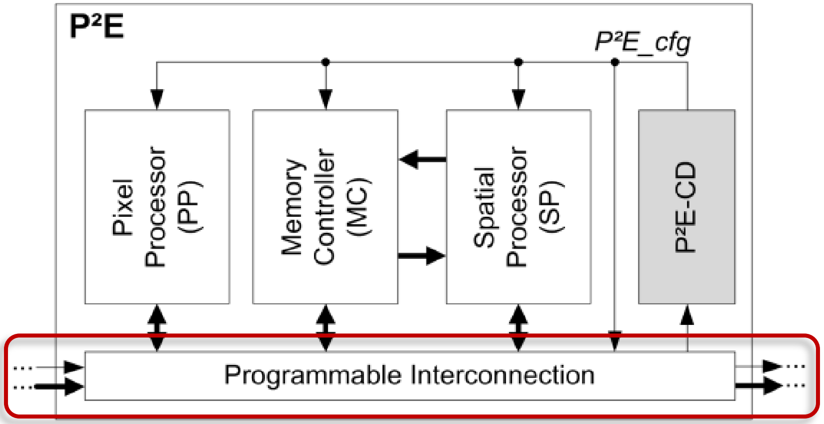
P^{2}IP Architecture

- ❑ **Flexible: programmable**
- ❑ Global synchronization
- ❑ Global clock
- ❑ Global memory
- ❑ Regular pipeline



P^{2}E Architecture

- ❑ Custom IP
- ❑ Dedicated local memory
- ❑ **Reconfigurable interconnection**
- ❑ Irregular pipeline

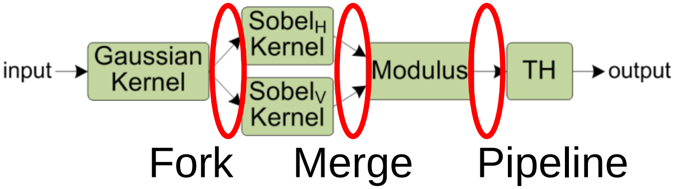


[2] Reconfigurable low-latency architecture for real-time image and video processing, P. Possa, PhD Thesis, Umons, Belgium, 20013

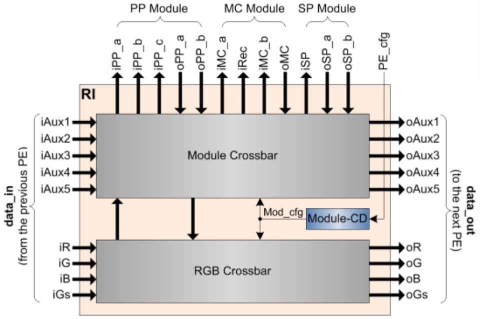
Parallel Pipeline

Programming Architecture

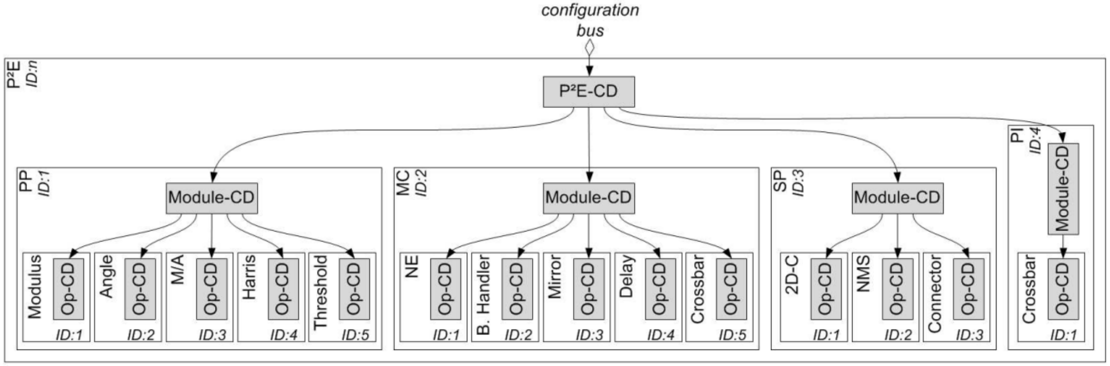
Simple edge detection



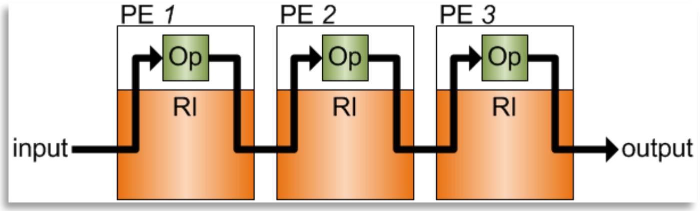
❑ **Routing**: programmable



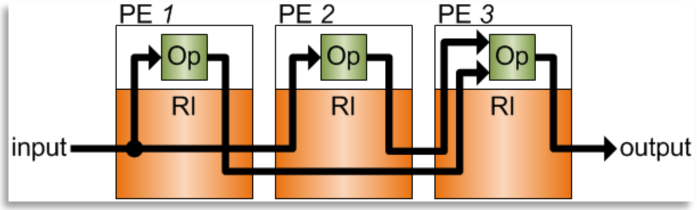
❑ **Mapping**: programmable



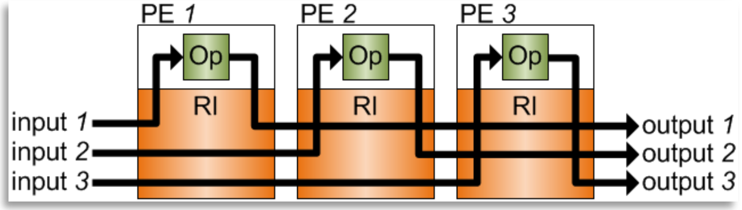
Reconfigurable interconnection



Regular pipeline



fork + merge



Parallel pipelines

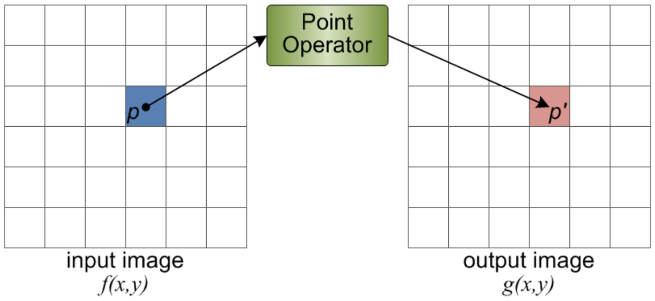
[2] Reconfigurable low-latency architecture for real-time image and video processing, P. Possa, PhD Thesis, Umons, Belgium, 20013

Shared Memory

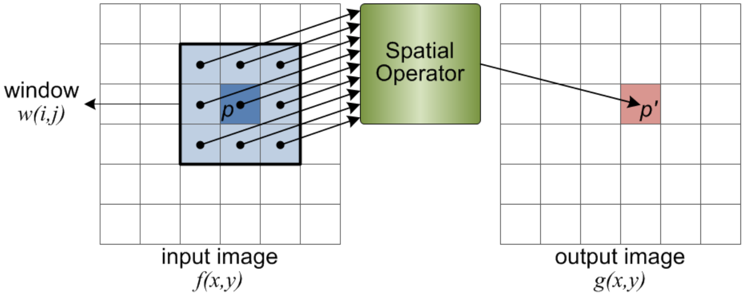
- ❑ **Available memory**: If shared memory is required, it can be built out of the **internal** (raw RAM) or **external memory**
- ❑ **Required memory**: the application determines **whether shared memory** is needed as well as which engines need to access it
- ❑ **Bus arbitration**: access to **shared memory** is a critical element for ensuring high performance
- ❑ **Latency**: because **memory latency** has a huge impact on throughput, each access must be as fast as possible

Shared Memory

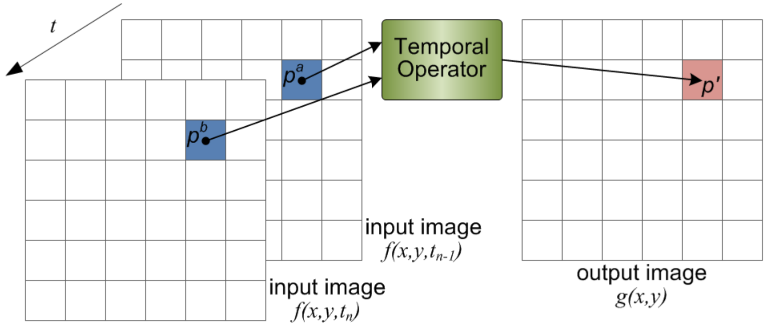
Memory access design patterns



- Arithmetic Functions
  - Multiplication, addition…
- Logic Functions
  - AND, OR…
- Testing Functions
  - Threshold, less-than, grater-than…



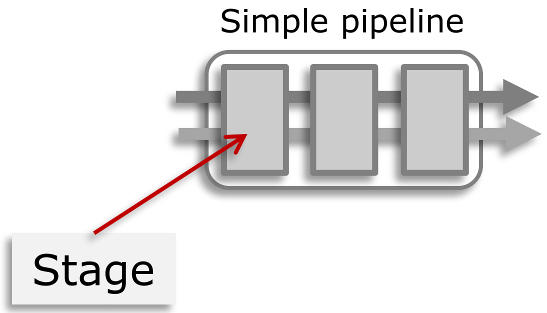
- Linear Filters
  - Sobel, Gaussian, mean…
- Rank Filters
  - Max, Min, Median.
- Mathematical Morphology
  - Erosion, Dilation…



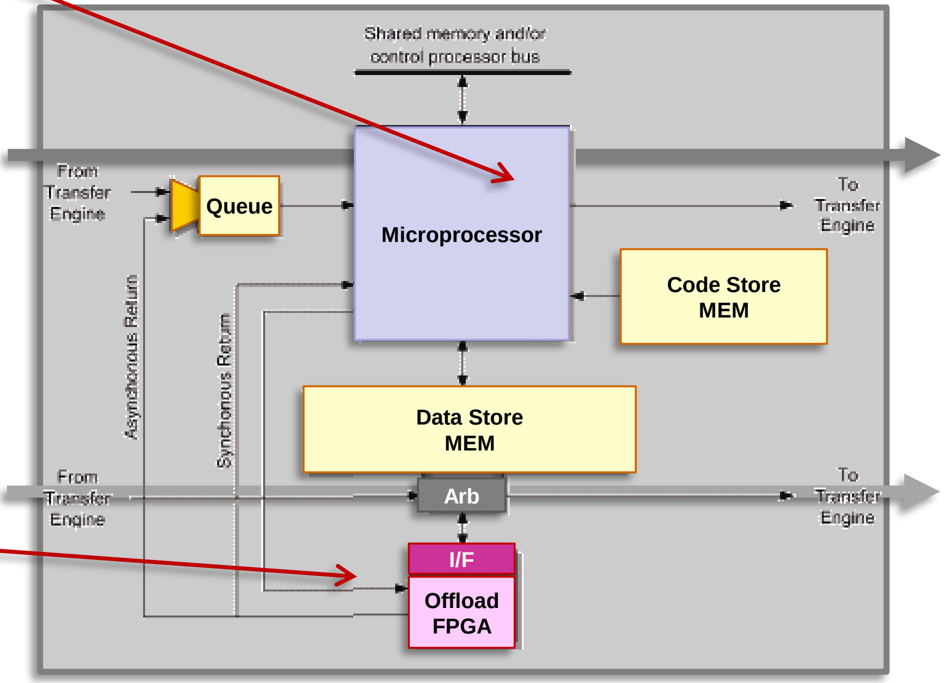
- Background Subtraction
- Motion Detection
- Optical Flow

Offload Code

Tradeoffs can be made between performance, flexibility and the number of processors required



- ❑ Software (SW) provides flexibility
  - Portions of the application that are susceptible to **change or extension** should be left in software
- ❑ Hardware (HW) provides processing power
  - Well-established portions of the code (especially compute-intensive ones) can be **offloaded** to an **accelerator (HW)**



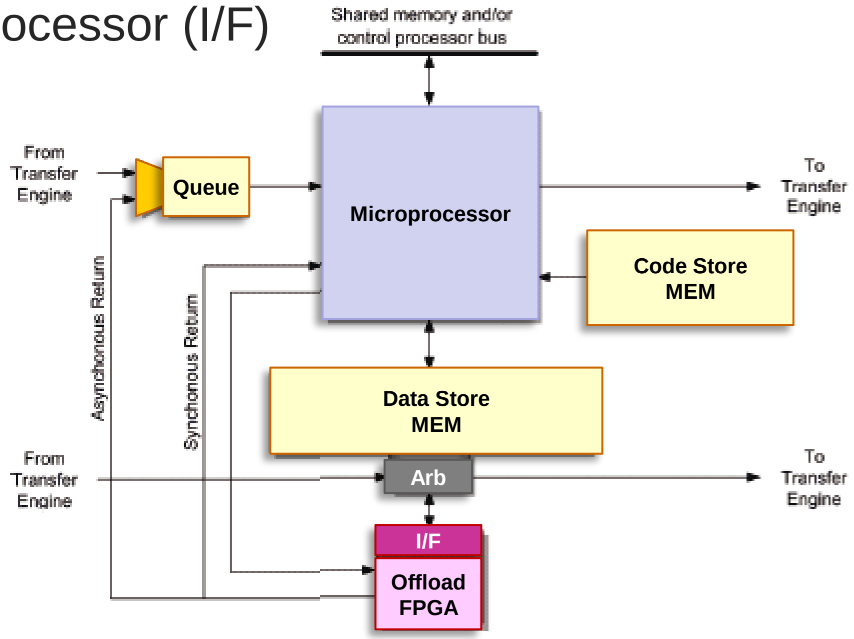
[1] Building Blocks Simplify Multicore in FPGAs, B. Moyer, Chip Design Magazine, 2007

Offload Code

By adjusting the amount of code reduced to hardware (FPGA or IP), tradeoffs can be made between performance and flexibility

But you still need ...

- ❑ Accelerators (HW Offload FPGA) can be created using any register-transfer-level (RTL) logic
- ❑ Wrappers required to interface the processor (I/F)
- ❑ Synchronization of Operation/Data is required (wait, hands off...)



System-level Tools

The enormous range of possibilities of heterogeneous multicore systems makes it difficult to join the easiness to program of homogeneous systems and the performance and power savings of heterogeneous multicores

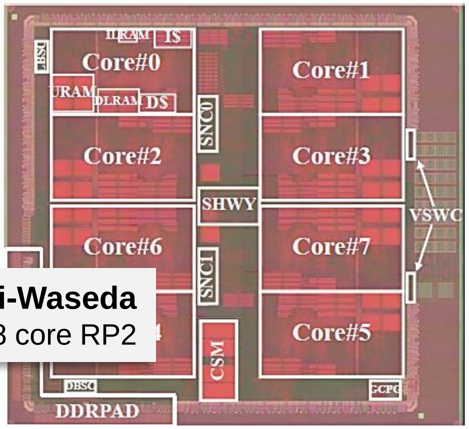
[2013 Automatic Design Exploration Framework for Multicores with Reconfigurable Accelerators]

- ❑ **General purpose computing**: profit from **standard** tools and languages for application development
- ❑ **SW High-level tools**: a standard architecture profit from mature and optimized high-level tools (POCC, Par4All, openMP, MPI, CUDA)
- ❑ **ESL (Electronics System Level Design Tools)**: Xilinx AutoESL, Altera Qsys, ROCCC, CatapultC, NI LabView
- ❑ **HW Higher-level tools**: not just sequential but multicore and distributed architecture design

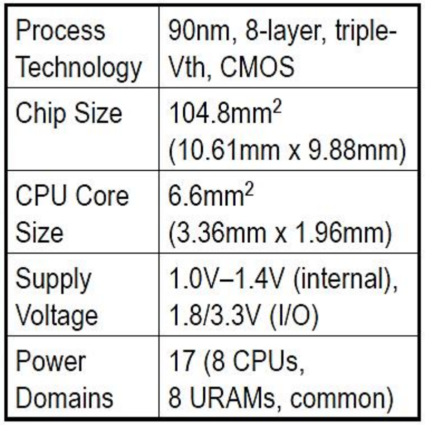
System-level Tools

OSCAR: Optimally SCheduled Advanced multiprocessoR

- ❑ **Multicore architecture**:
- ❑ **GPP** (OpenSPARC-based) cores with
- ❑ **HW** tightly-coupled reconfigurable accelerators (run-time reconfigurable FU application-specific)

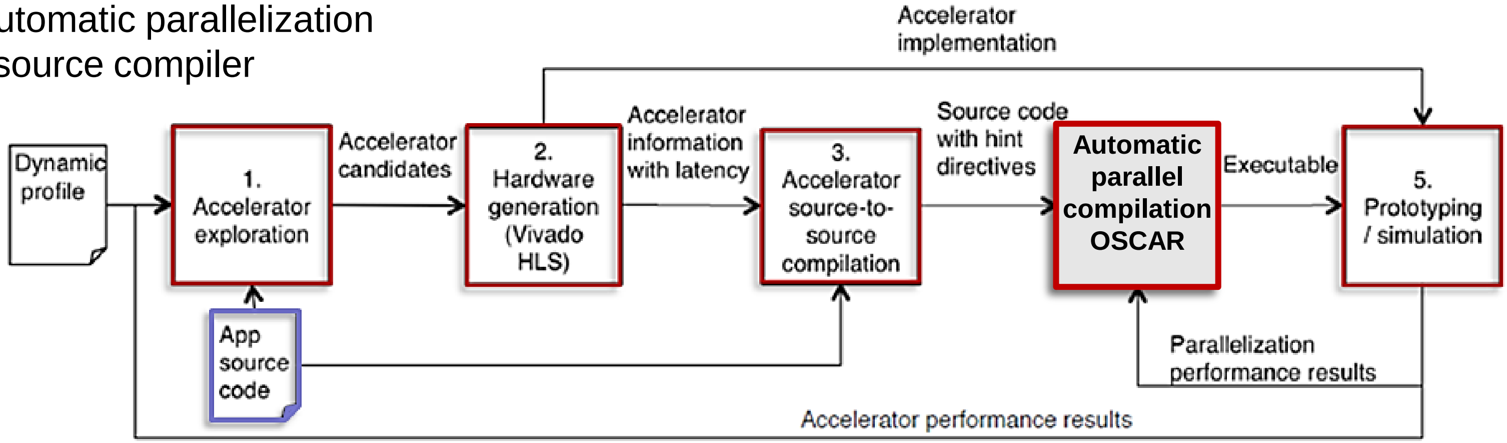


Renesas-Hitachi-Waseda
Low Power 8 core RP2



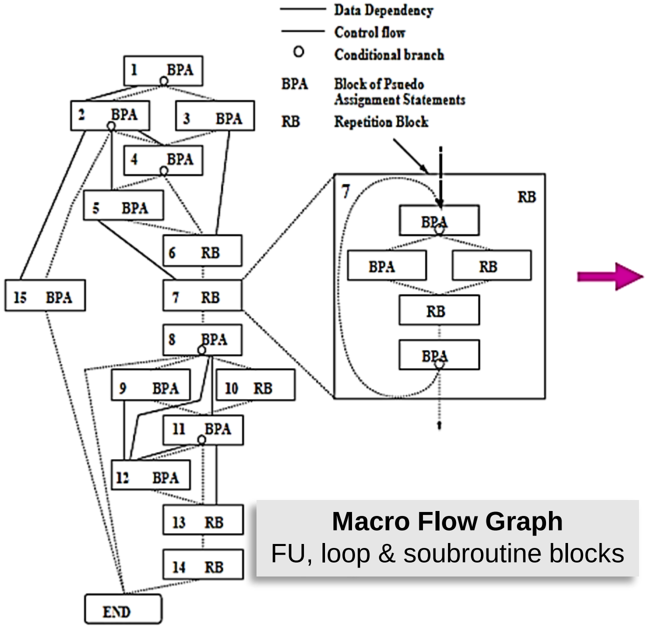
Process Technology	90nm, 8-layer, triple-Vth, CMOS
Chip Size	104.8mm^2 (10.61mm x 9.88mm)
CPU Core Size	6.6mm^2 (3.36mm x 1.96mm)
Supply Voltage	1.0V–1.4V (internal), 1.8/3.3V (I/O)
Power Domains	17 (8 CPUs, 8 URAMs, common)

OSCAR Automatic parallelization source-to-source compiler

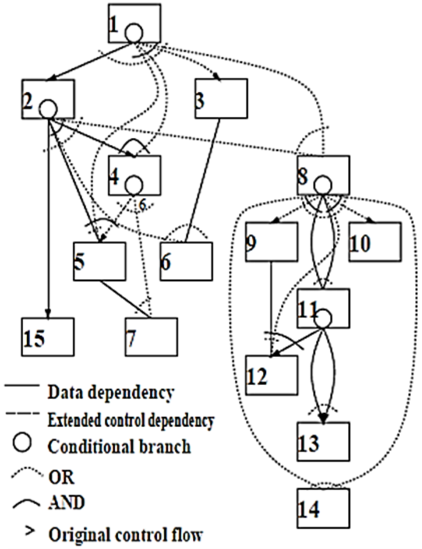


Accelerator implementation
Dynamic profile
1. Accelerator exploration
Accelerator candidates
2. Hardware generation (Vivado HLS)
Accelerator information with latency
3. Accelerator source-to-source compilation
Source code with hint directives
Automatic parallel compilation OSCAR
Executable
5. Prototyping / simulation
App source code
Parallelization performance results
Accelerator performance results

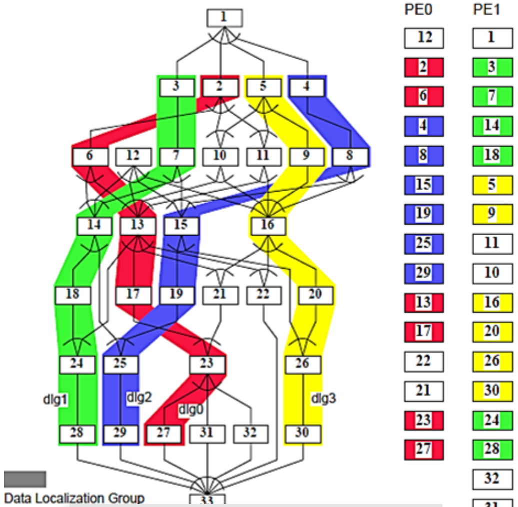
System-level Tools



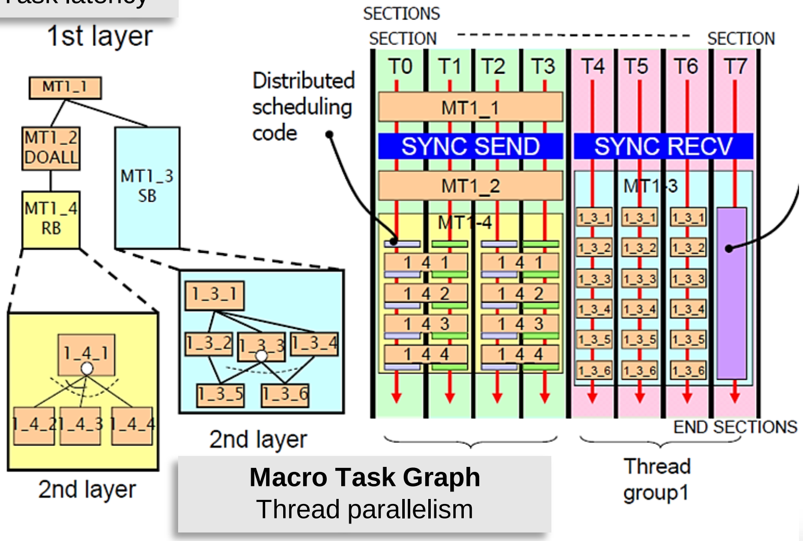
Macro Flow Graph
FU, loop & soubroutine blocks



Macro Task Graph
Data depencencies, Task latency



Macro Task Graph
Scheduling & Mapping



Macro Task Graph
Thread parallelism

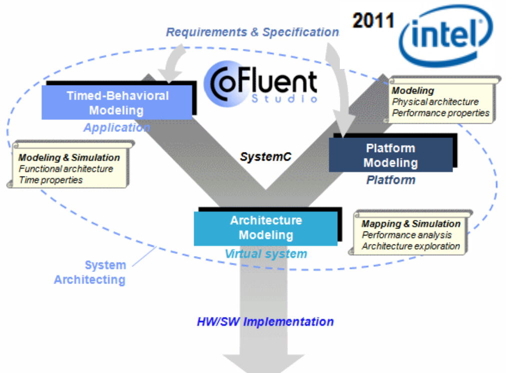
OSCAR Automatic parallelization

[2013 Automatic Design Exploration Framework for Multicores with Reconfigurable Accelerators]

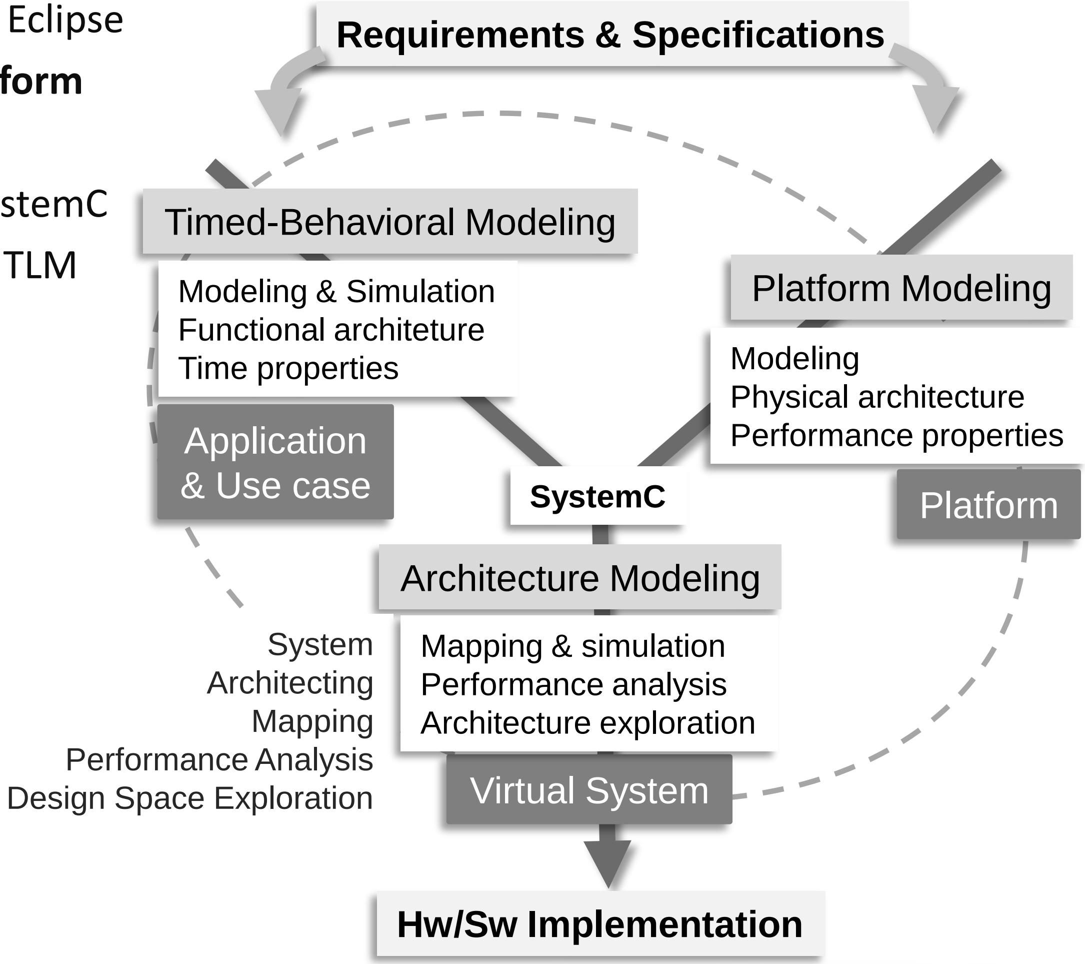
System-level Tools

Cofluent Studio: SystemC transactional models from graphics to standard C multi-OS, multi-core systems

- **Graphical modeling**: UML + Eclipse
- **System = Application + Platform**
- **Application**: ANSI C/C++
- **System Implementation**: SystemC
- **Timing properties analysis**: TLM



http://www.cofluentdesign.com



Requirements & Specifications

Timed-Behavioral Modeling

Modeling & Simulation
Functional architeture
Time properties

Application
& Use case

Platform Modeling

Modeling
Physical architecture
Performance properties

Platform

SystemC

Architecture Modeling

Mapping & simulation
Performance analysis
Architecture exploration

Virtual System

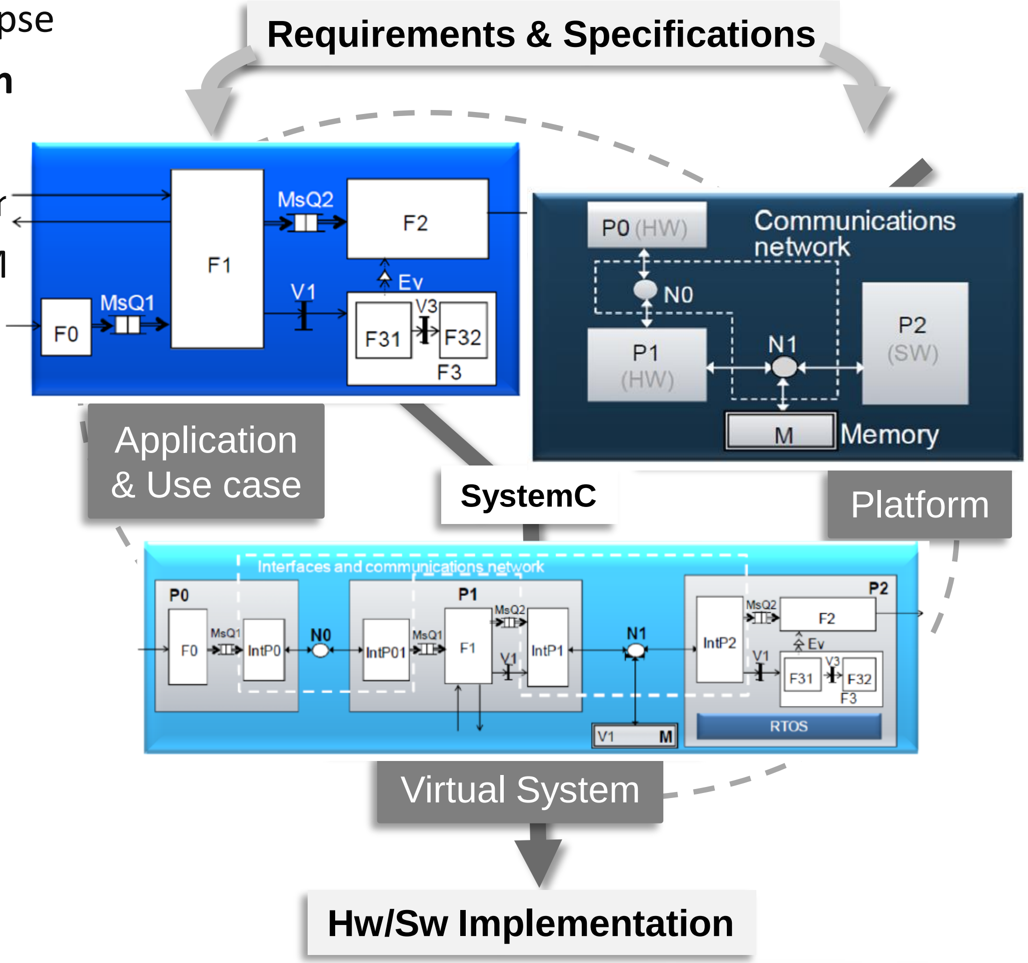
System
Architecting
Mapping
Performance Analysis
Design Space Exploration

Hw/Sw Implementation

System-level Tools

Cofluent Studio: SystemC transactional models from graphics to standard C multi-OS, multi-core systems

- **Graphical modeling**: UML + Eclipse
- **System = Application + Platform**
- **Application**: ANSI C/C++
- **System Implementation**: System
- **Timing properties analysis**: TLM



Requirements & Specifications

Application & Use case

SystemC

Platform

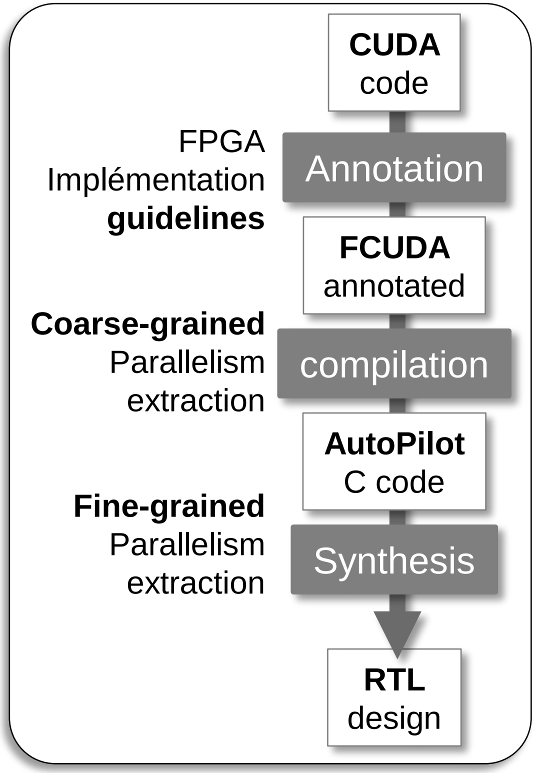
Virtual System

Hw/Sw Implementation

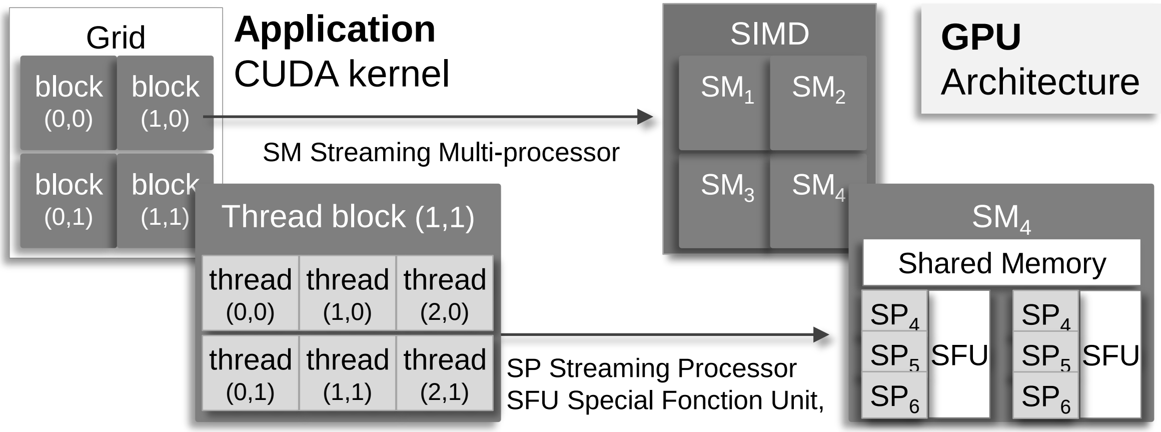
System-level Tools

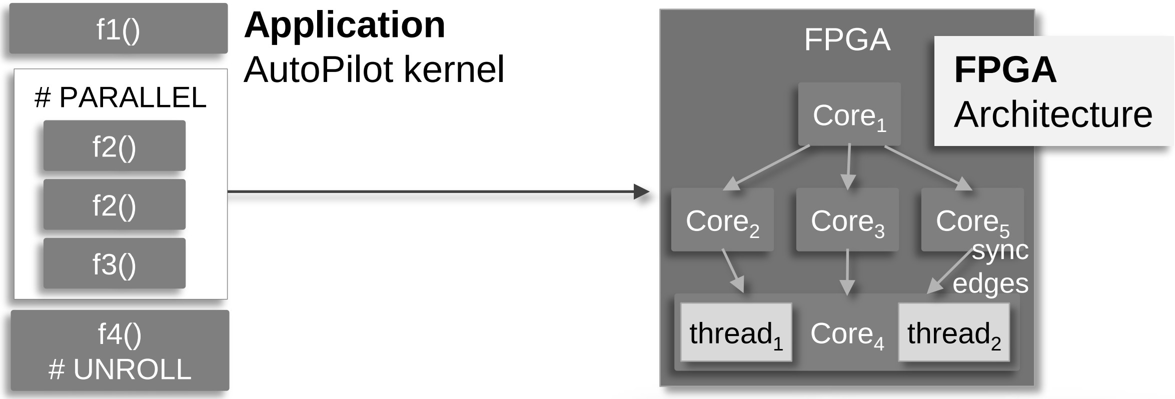
FCUDA: Compilation of CUDA kernels onto FPGA [1]

AutoPilot & AutoESL (Xilinx)



CUDA code
FPGA Implémentation **guidelines**
Annotation
FCUDA annotated
Coarse-grained Parallelism extraction
compilation
AutoPilot C code
Fine-grained Parallelism extraction
Synthesis
RTL design





[1] Papakonstantinou, 2009

System-level Tools

ST Platform 2012 Tools stack

Programming Models | **Application Capture** | **Programming Environment** | **Parallel Programming Pattern** | **System Infrastructure & Runtime** | **Platform Models**

Standard

OpenMP

Streaming

OpenCL

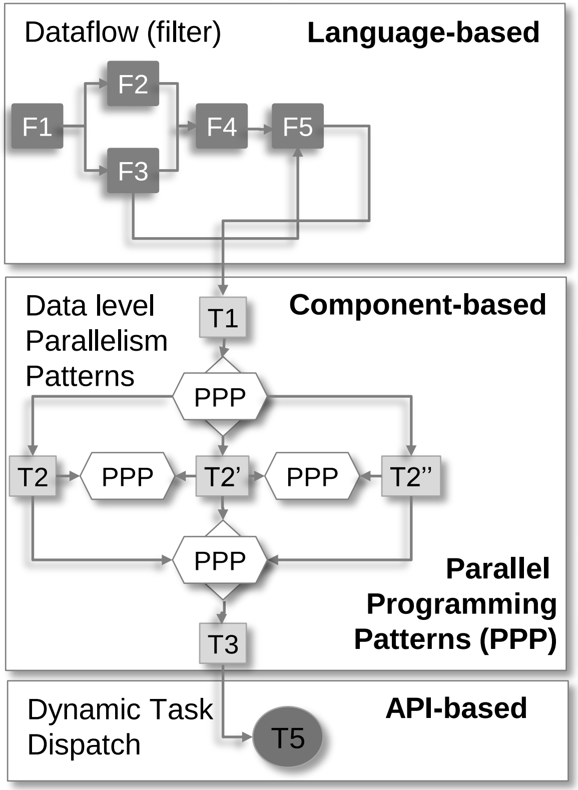
Domain specific

PEDF DDF → TLP DLP

HL Threading

Native Programming layer

NPL API → HAL



PEDF Predicated Exécution Data Flow
DLP Data Level Parallelism

Execution Model

PEDF, RunToCompl, DDF, ThreadPool, SDF, Thread

Communication/Sync

Exchanger, FIFO, QueueIterator, SyncBuffer

Memory Access

Prefetch, Asynch Prefetch

Dynamic Deployment

QoS

Power Management

Execution Engines

Monitoring

Fault-tolerance

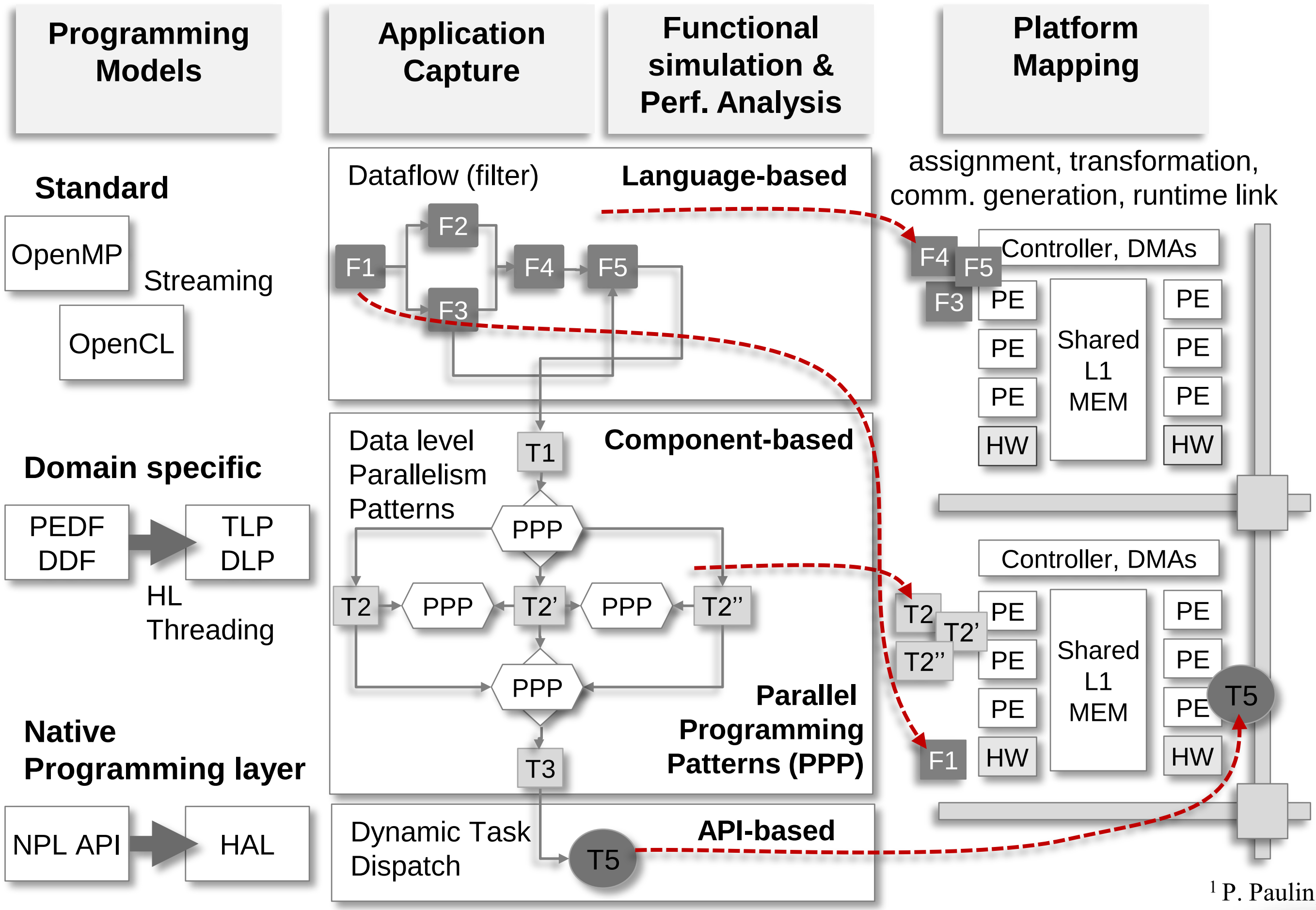
SystemC

TLM

1 P. Paulin, Towards Multicore Programming, 2011

System-level Tools

ST Platform 2012 Tools stack



Programming Models

Application Capture

Functional simulation & Perf. Analysis

Platform Mapping

Standard

OpenMP

Streaming

OpenCL

Dataflow (filter)

Language-based

assignment, transformation, comm. generation, runtime link

Controller, DMAs

Domain specific

PEDF DDF

TLP DLP

HL Threading

Data level Parallelism Patterns

Component-based

Parallel Programming Patterns (PPP)

Controller, DMAs

Native Programming layer

NPL API

HAL

Dynamic Task Dispatch

API-based

[1] P. Paulin, Towards Multicore Programming, 2011

System-level Tools

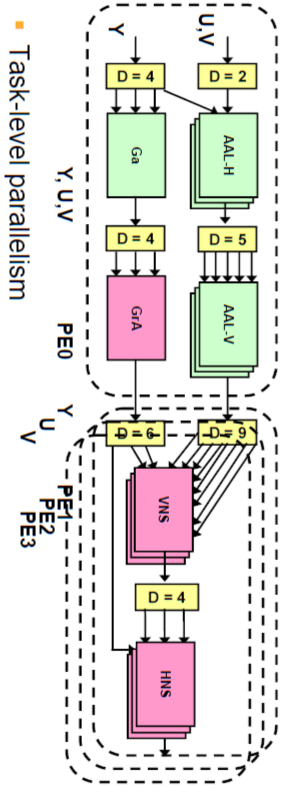
ST Platform 2012 Tools stack

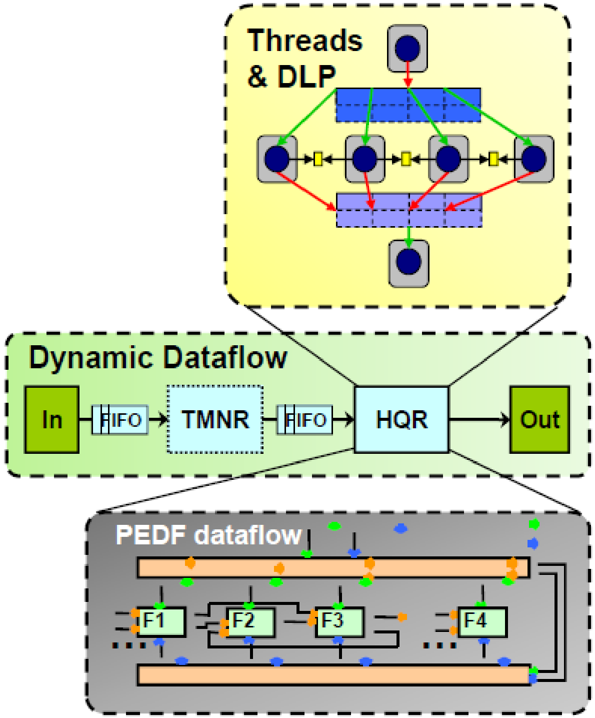
Programming Models

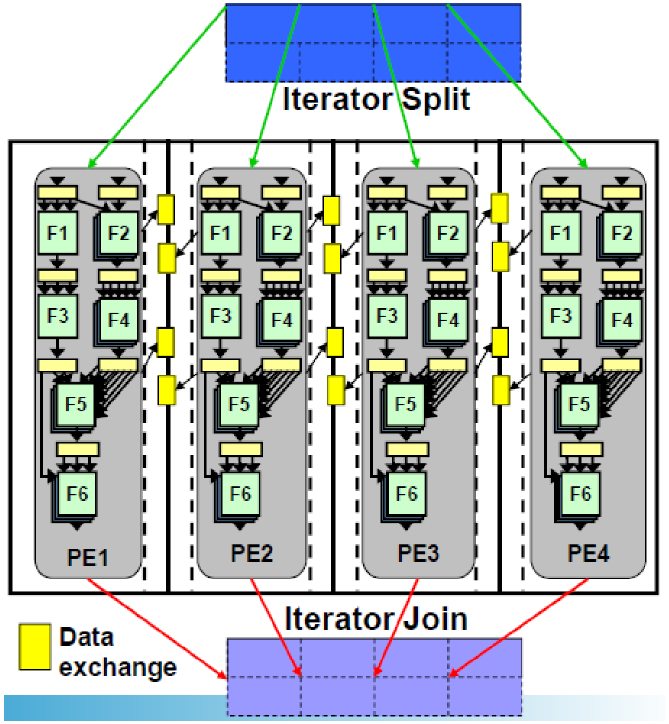
Application Capture

Platform Mapping

SW Mapping







[1] P. Paulin, Towards Multicore Programming, 2011

Conclusion

Efficient Architectures Evolution

- **Portable and scalable implementations**. **Heterogeneous** & **homogeneous** architectures. **Multicore** + **Reconfigurable** interconnections.
- **Power consumption ahead of devices concern**: Energy costs and green requirements for **portable**, **telecom** and **home devices**.
- **More Hard blocks**: **power**, **area** and **speed** improvement. FPGA dominates reconfigurable flexibility but at a high silicon price (prevent broad adoption).
- **Generic**: processors, switch fabric, datapath operators. **Dedicated/Specific**: RAM, PLL, DSP. **Power favors**: power reduction vs. soft logic and lower static power (when not used)

Programmable platforms

- **FPGA strategy + multi-core approach adoption**: FPGA concepts, trends and roadmaps driving new technology and raising competition
- **Standards**: keep market interest, evolution and interoperability (even with unconventional technology)

Conclusion

Multimedia and telecommunication applications demand software performance

- **Sequential software migration**. Need for **parallel programming** to harness **multi-core** architectures. Parallel programming **model** and **common patterns** to capture **resources/architecture requirements**. Compelling parallel applications. Program efficiency.

Raising Design Abstraction

- **Programmable + Reconfigurable concepts**: **FPGAs use is mainly restricted to hardware engineers** (**productivity issue**: HDL centric design).
- **Software-like design flow**: including **API**s, **GPP**s, **drivers** and high performance **libraries**.
- **Streaming programming languages** (including parallel programming concepts).
- **Programming models**: higher productivity, increased **platform independence**, multiple objectives