Advanced Computer Architecture
System-on-Chip

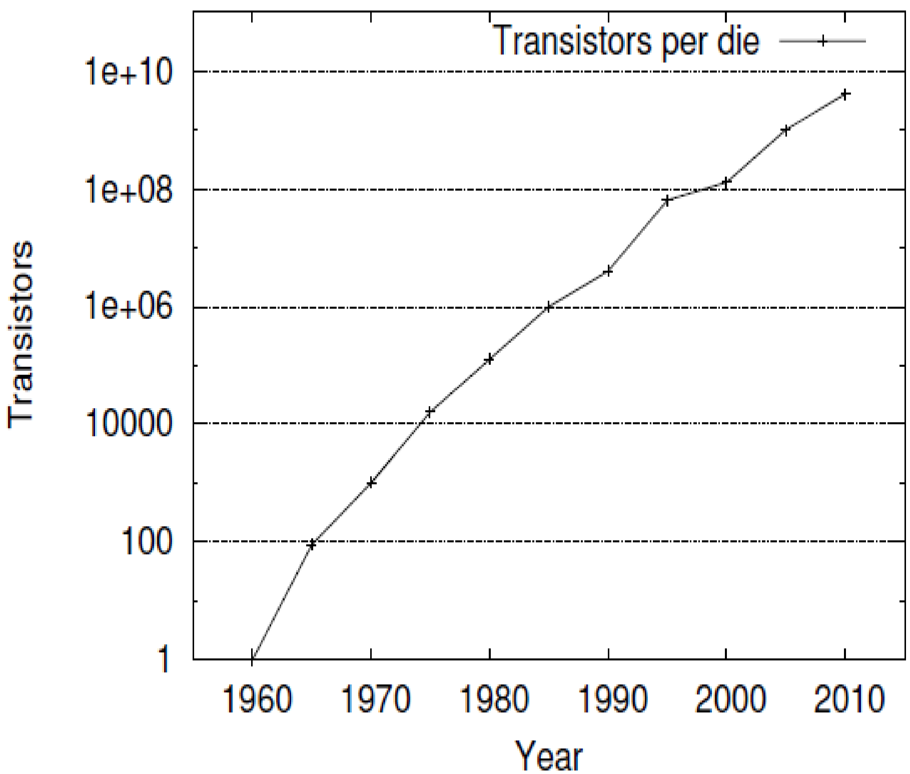
By Amirreza Hosseini

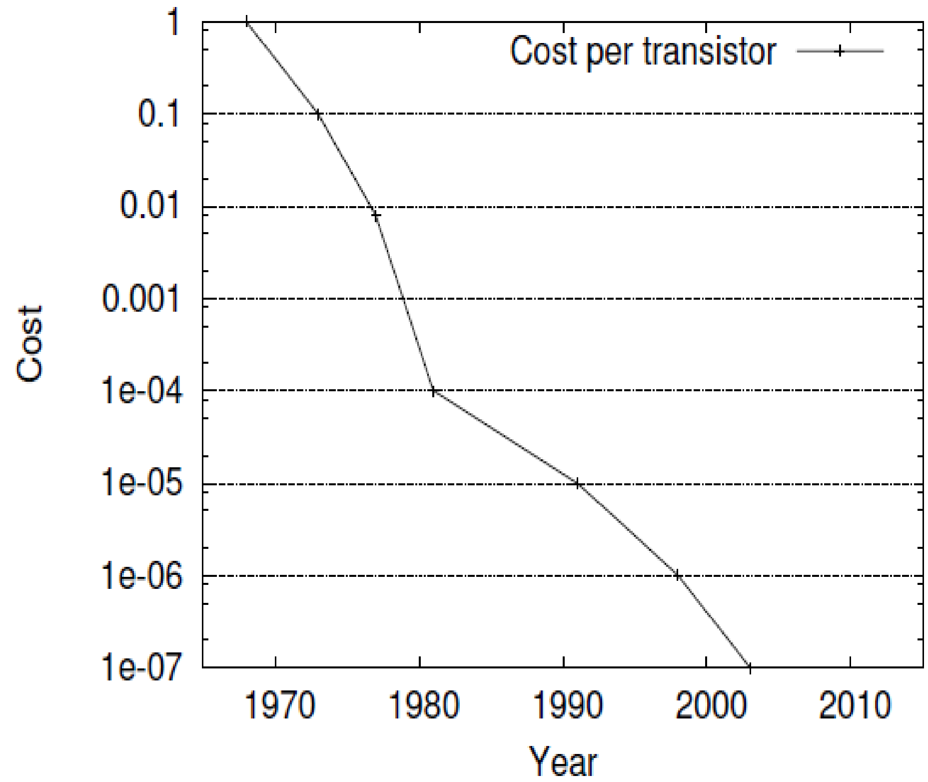
Dr. Farbeh

SOC architecture and design

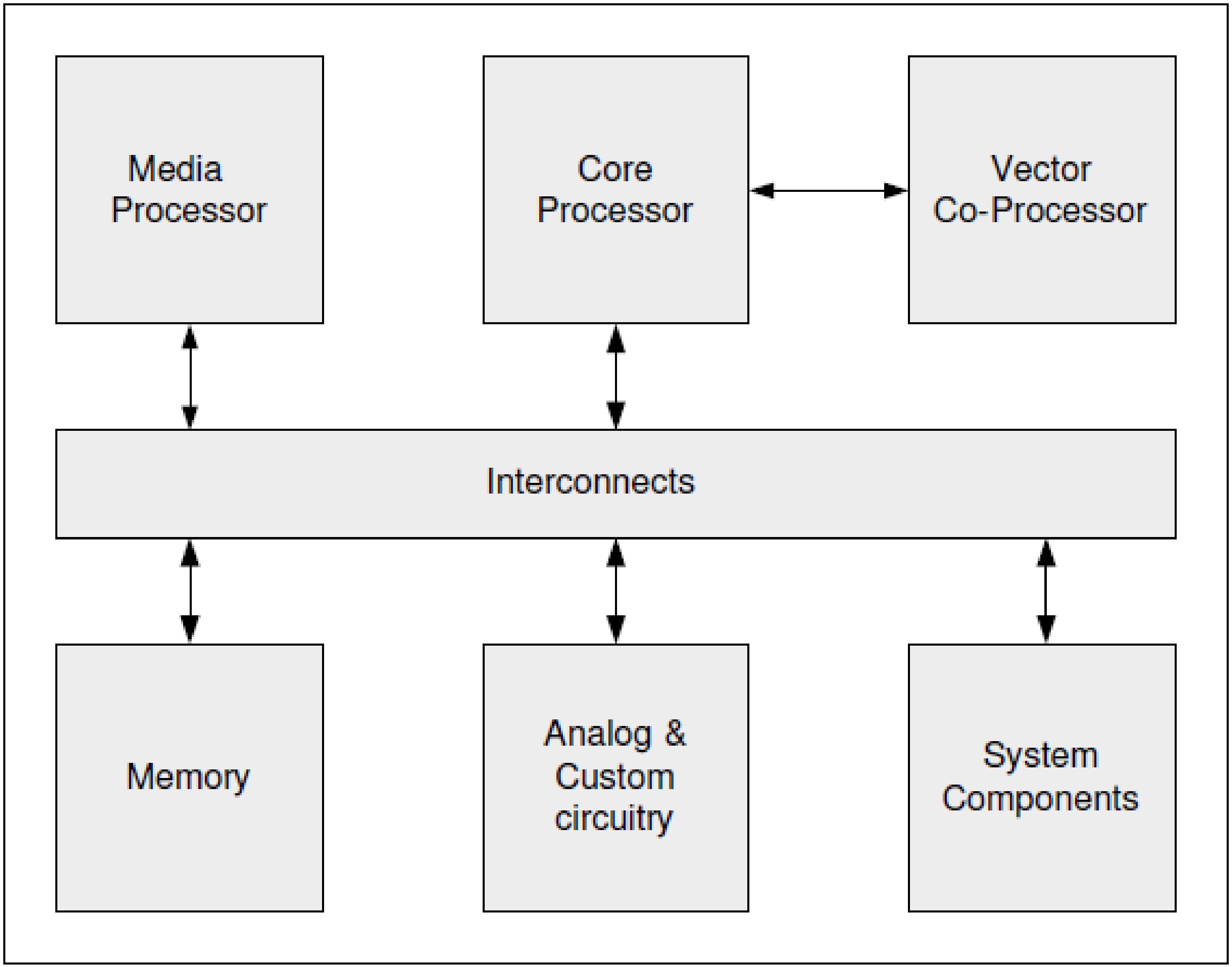
- system-on-chip (SOC)
  - processors: become components in a system
- SOC covers many topics
  - processors, cache, memory, interconnect, design tools
- need to know
  - user view: variety of processors
  - basic information: technology and tools
  - processor internals: effect on performance
  - storage: cache, embedded and external memory
  - interconnect: buses, network-on-chip
  - evaluation: processor, cache, memory, interconnect
  - advanced: specialized processors, reconfiguration
  - design productivity: system modelling, design exploration

System on a Chip: driven by semiconductor advances





Basic system-on-chip model

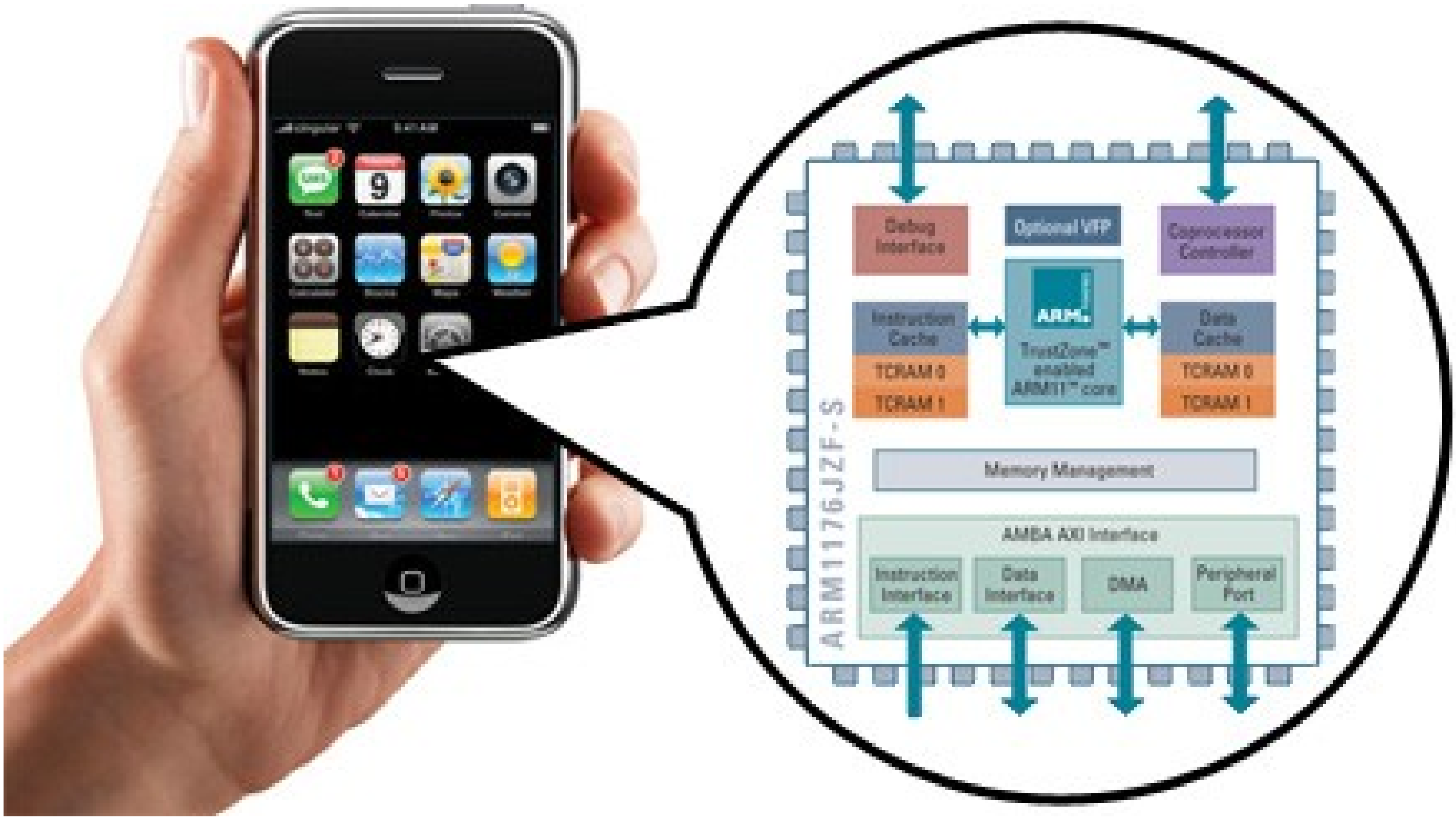


SOC vs processors on chip

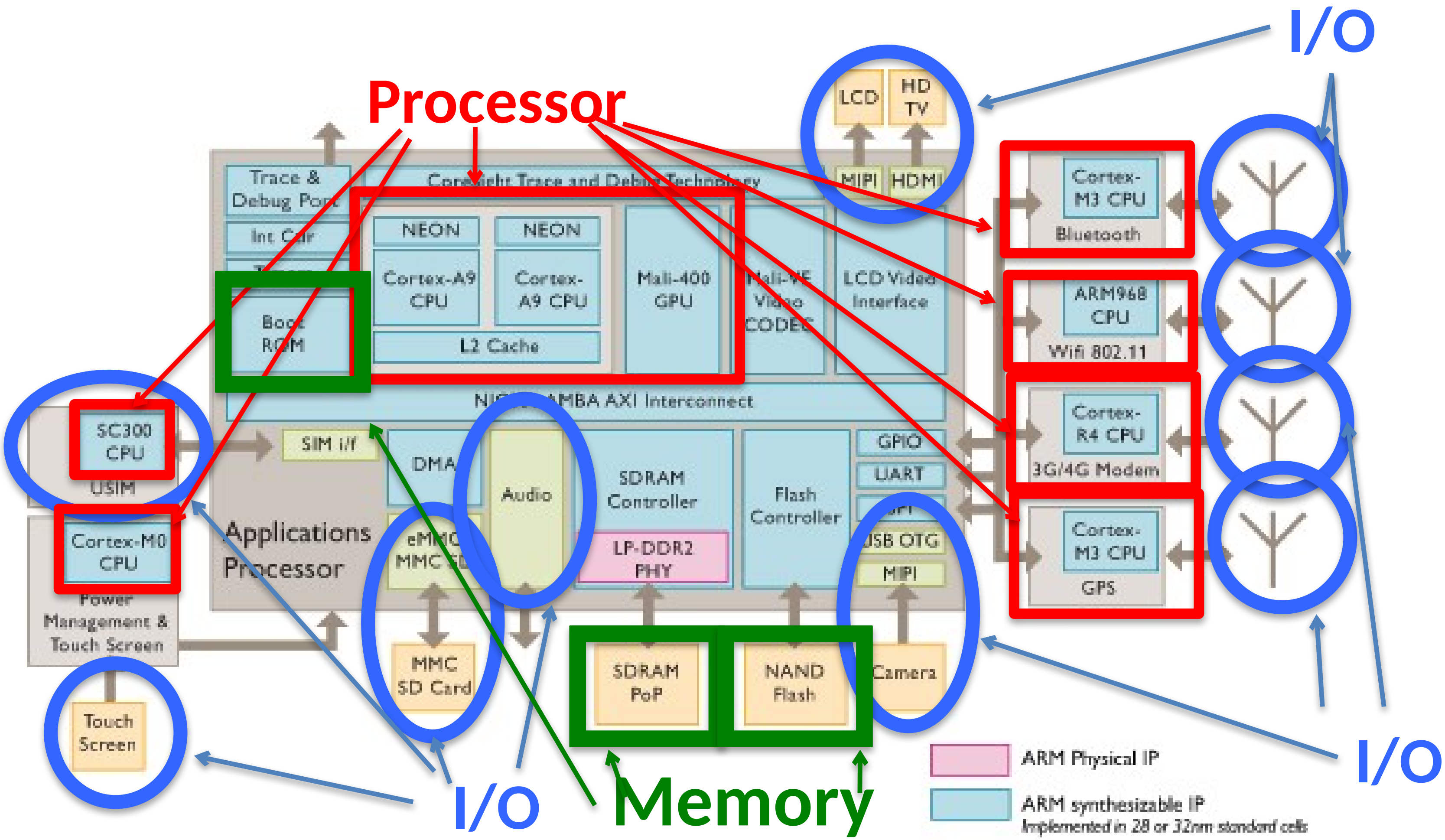
- with lots of transistors, designs move in 2 ways:
  - complete system on a chip
  - multi-core processors with lots of cache

	System on chip	*Processors on chip*
processor	multiple, simple, heterogeneous	few, complex, homogeneous
cache	one level, small	2-3 levels, extensive
memory	embedded, on chip	very large, off chip
functionality	special purpose	general purpose
interconnect	wide, high bandwidth	often through cache
power, cost	both low	both high
operation	largely stand-alone	need other chips

iPhone: has System-on-Chip

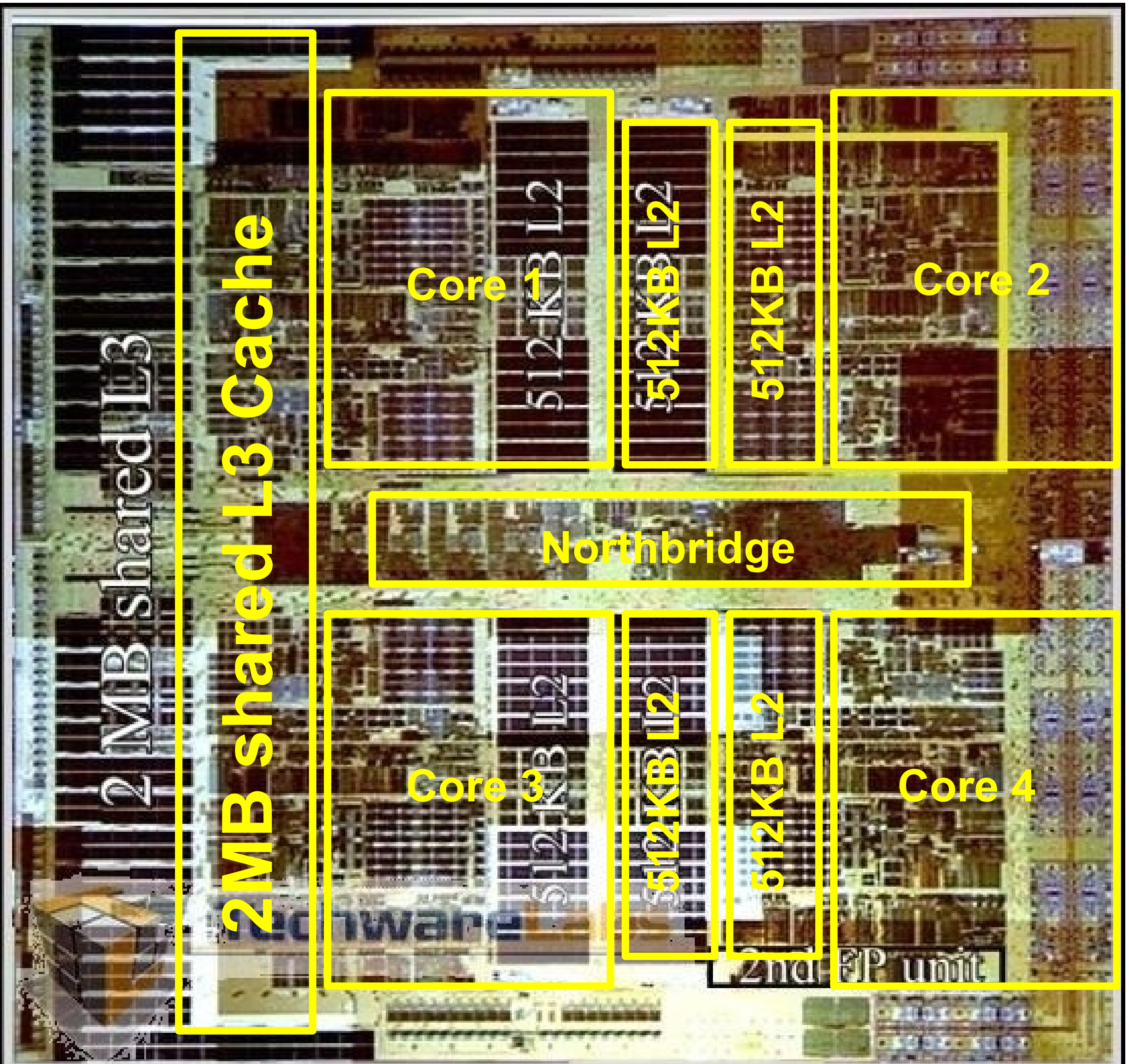


iPhone SOC

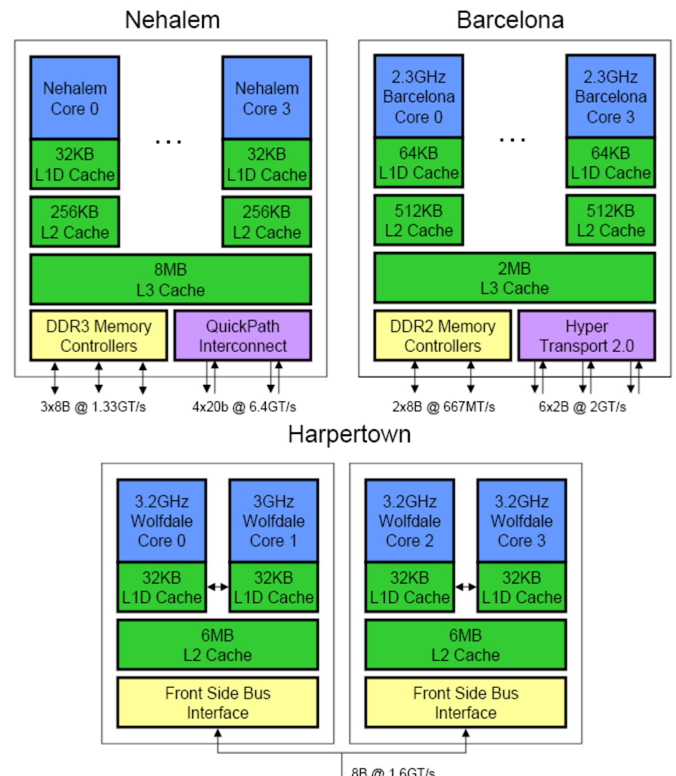


I/O
Processor
LCD
HD TV
MIPI
HDMI
Trace & Debug Port
CoreSight Trace and Debug Technology
NEON
NEON
Cortex-A9 CPU
Cortex-A9 CPU
L2 Cache
Mali-400 GPU
Mali-VE Video CODEC
LCD Video Interface
Boot ROM
Cortex-M3 CPU
Bluetooth
ARM968 CPU
Wifi 802.11
Cortex-R4 CPU
3G/4G Modem
Cortex-M3 CPU
GPS
SC300 CPU
USIM
Cortex-M0 CPU
Power Management & Touch Screen
SIM i/f
DMA
Audio
SDRAM Controller
LP-DDR2 PHY
Flash Controller
GPIO
UART
USB OTG
MIPI
Applications Processor
MMC SD Card
SDRAM PoP
NAND Flash
Camera
Touch Screen
I/O
Memory
I/O
ARM Physical IP
ARM synthesizable IP
Implemented in 28 or 32nm standard cells

AMD's Barcelona Multicore



- 4 out-of-order cores
- 1.9 GHz clock rate
- 65nm technology
- 3 levels of caches
- integrated Northbridge



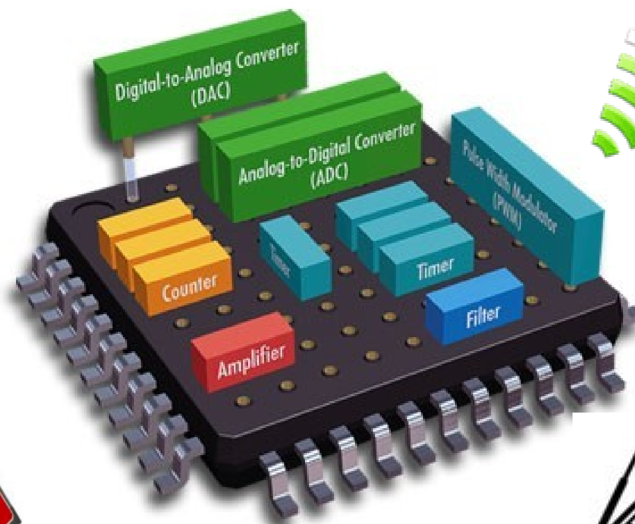
http://www.techwarelabs.com/reviews/processors/barcelona/

Why System on Chip

LCD, TFT display

GPS

Video camera



802.11 LAN

Keypad

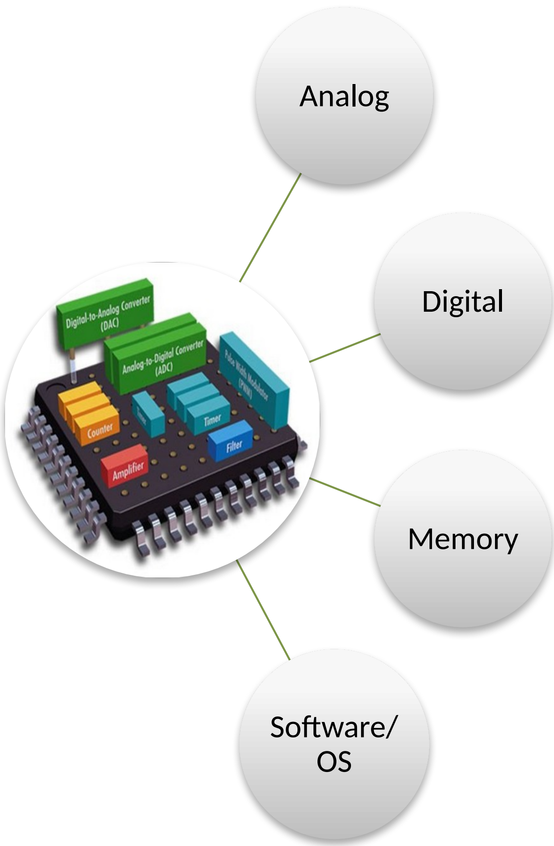
Camera

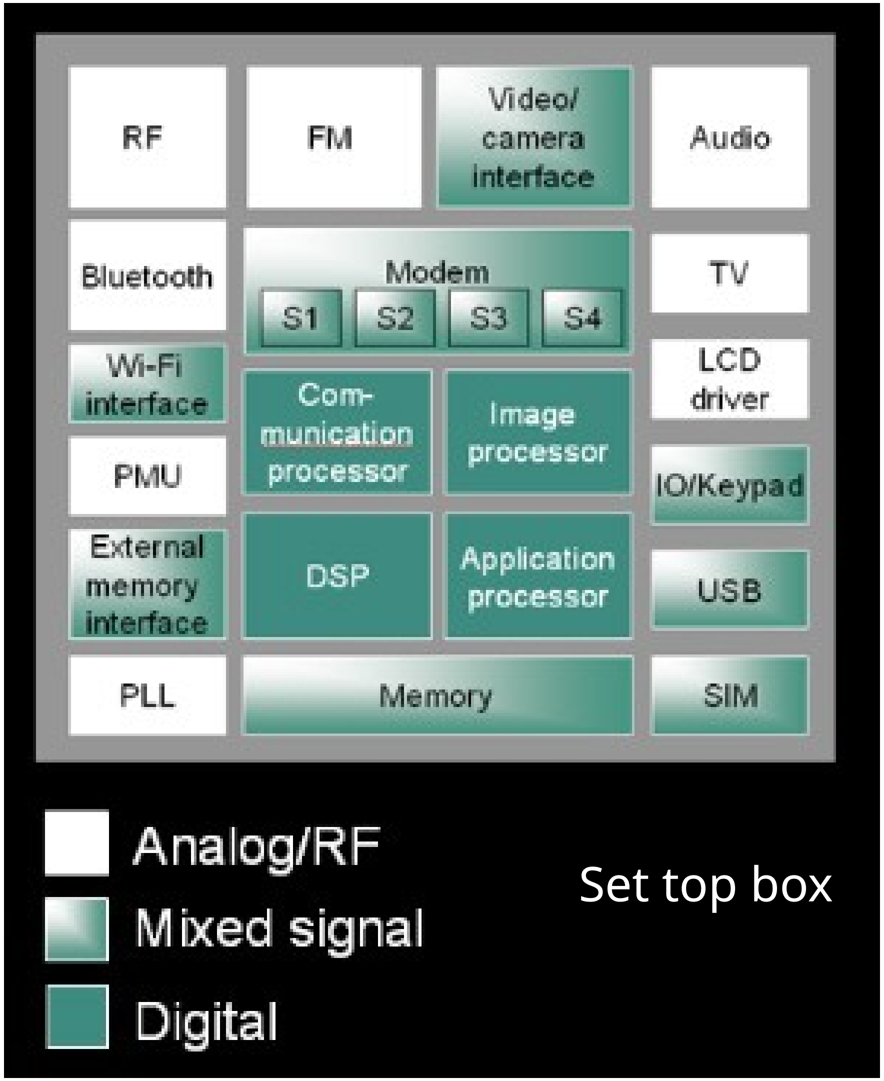
Compact Flash

Radio

http://www.c-p-e.be/cypros/

System of Chip Components



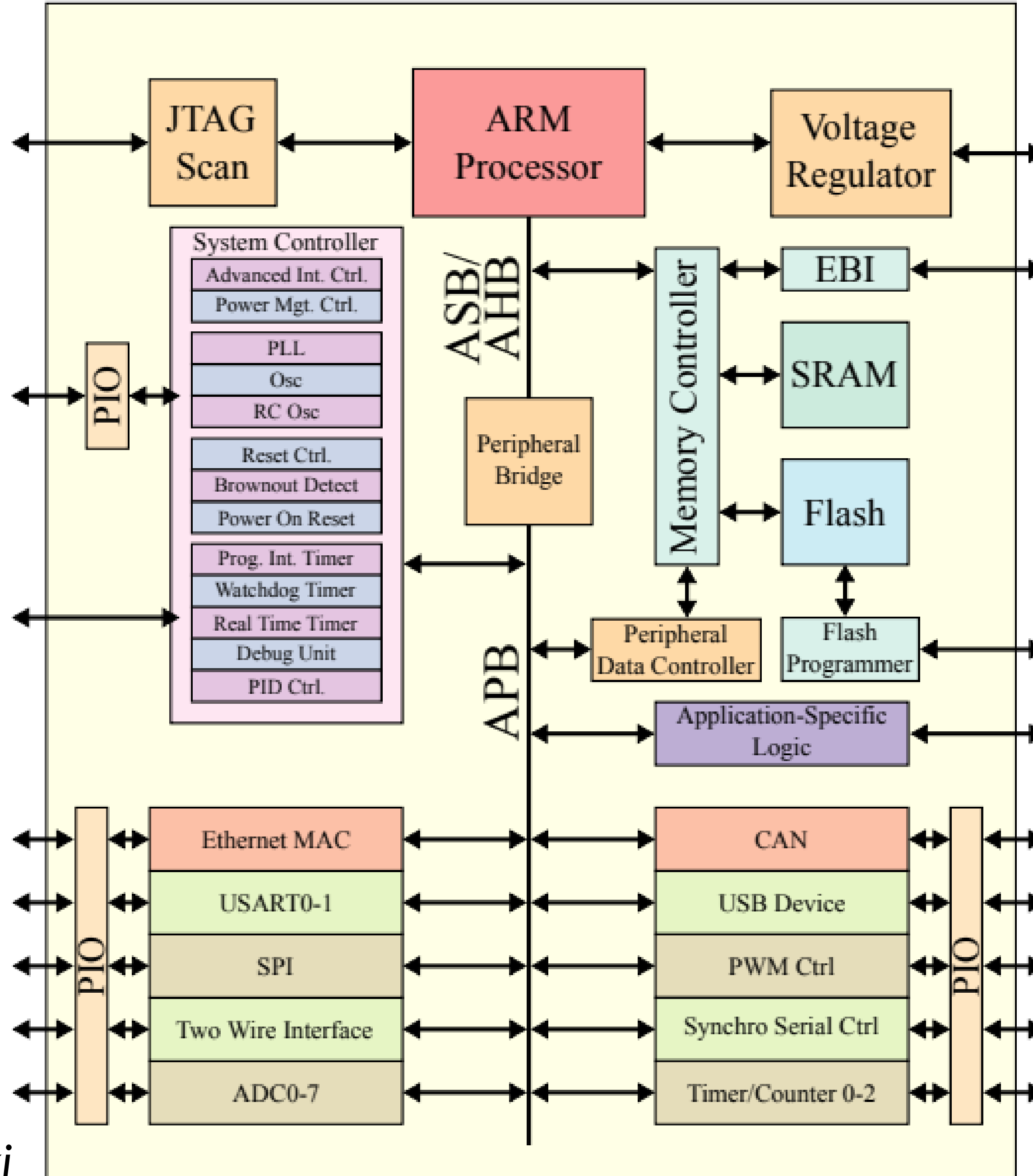


http://low-powerdesign.com/cadence_karnane.htm

SoC Hardware Architecture

Often used in embedded application.

How to implement an application on a HW platform executing some SW programs?



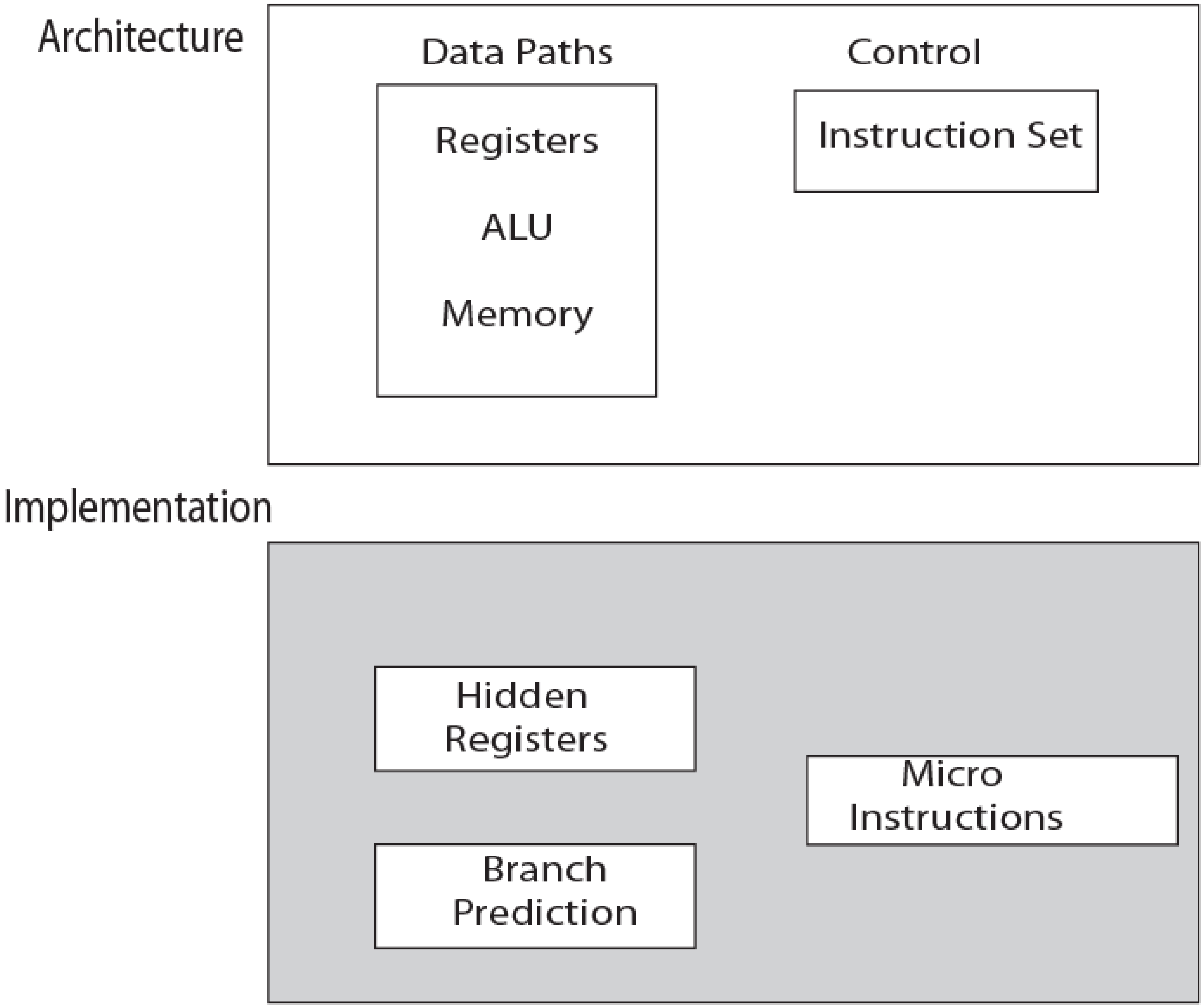
source: wiki

SOC design: key ideas

- to design and evaluate an SOC, designers need to understand:
  - its components: processors, memory, interconnect
  - applications that it targets
- SOC economics heavily dependent on:
  - costs: initial design, marginal production
  - volume: applicability, lifetime
- reducing design complexity
  - Intellectual Property (IP)
  - reconfigurable technology

SOC processors

- usually a mix of special and general purpose (GP)
  - can be proprietary design or purchased IP
- commonly GP processor is purchased IP
  - includes OS and compiler support
- GP processor optimized for an application
  - additional instructions
  - vector units



Architecture
Data Paths
Registers
ALU
Memory
Control
Instruction Set
Implementation
Hidden Registers
Branch Prediction
Micro Instructions

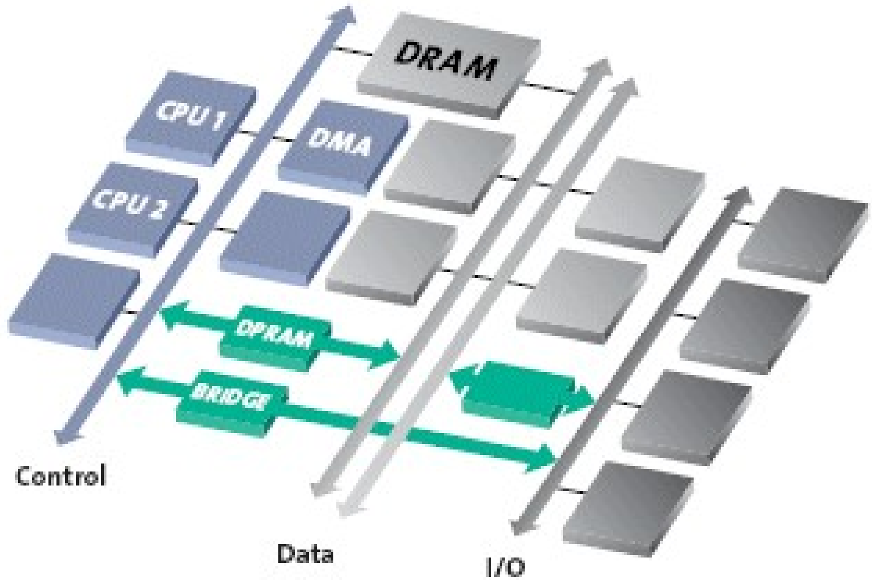
Research

- Interconnect topologies
  - e.g. bus
- Hybrid modules, Analog, Digital, ASIP, i.e. integration of mixed signal elements.

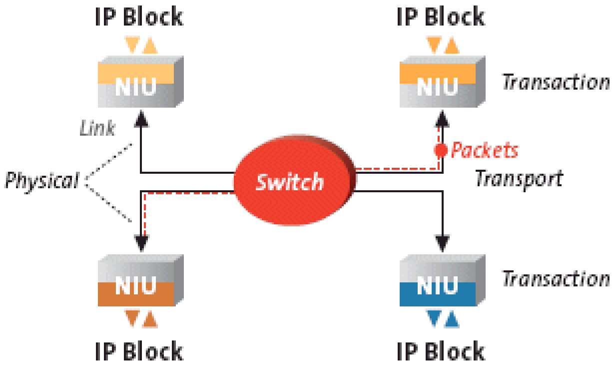
Mixed Signal (integrated analog blocks such as A to D and D to A converters, phase-locked loops, and adaptive filters.)

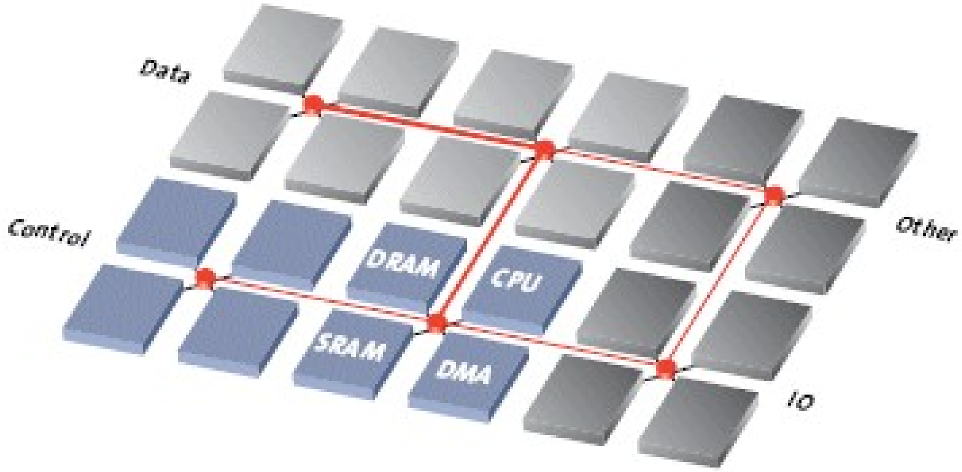
- On chip communication, power, switching delay, performance such as throughput.
- Hardware software co design for reuse.

Traditional architecture



- Current NOC (arteris.com)





Interconnect topologies

- Bus
- Star
- Cube
- Tree
- Spin

Some processors for SOCs

SOC	*Basic ISA*	*Processor description*
Freescale c600: signal processing	PowerPC	Superscalar with vector extension
ClearSpeed CSX600: general	Proprietary	Array processor with 96 processing elements
PlayStation 2: gaming	MIPS	Pipelined with 2 vector coprocessors
ARM VFP11: general	ARM	Configurable vector coprocessor

Processor types: overview

Processor type	*Architecture / Implementation approach*
SIMD	Single instruction applied to multiple functional units
Vector	Single instruction applied to multiple pipelined registers
VLIW	Multiple instructions issued each cycle under *compiler* control
Superscalar	Multiple instructions issued each cycle under *hardware* control

Adding instructions

- additional instructions to support specialized resources
  - exception: superscalar, with hardware control
- instructions can be added to base processor for coprocessor control
  - VLIW: Very Large Instruction Word
  - Array
  - Vector

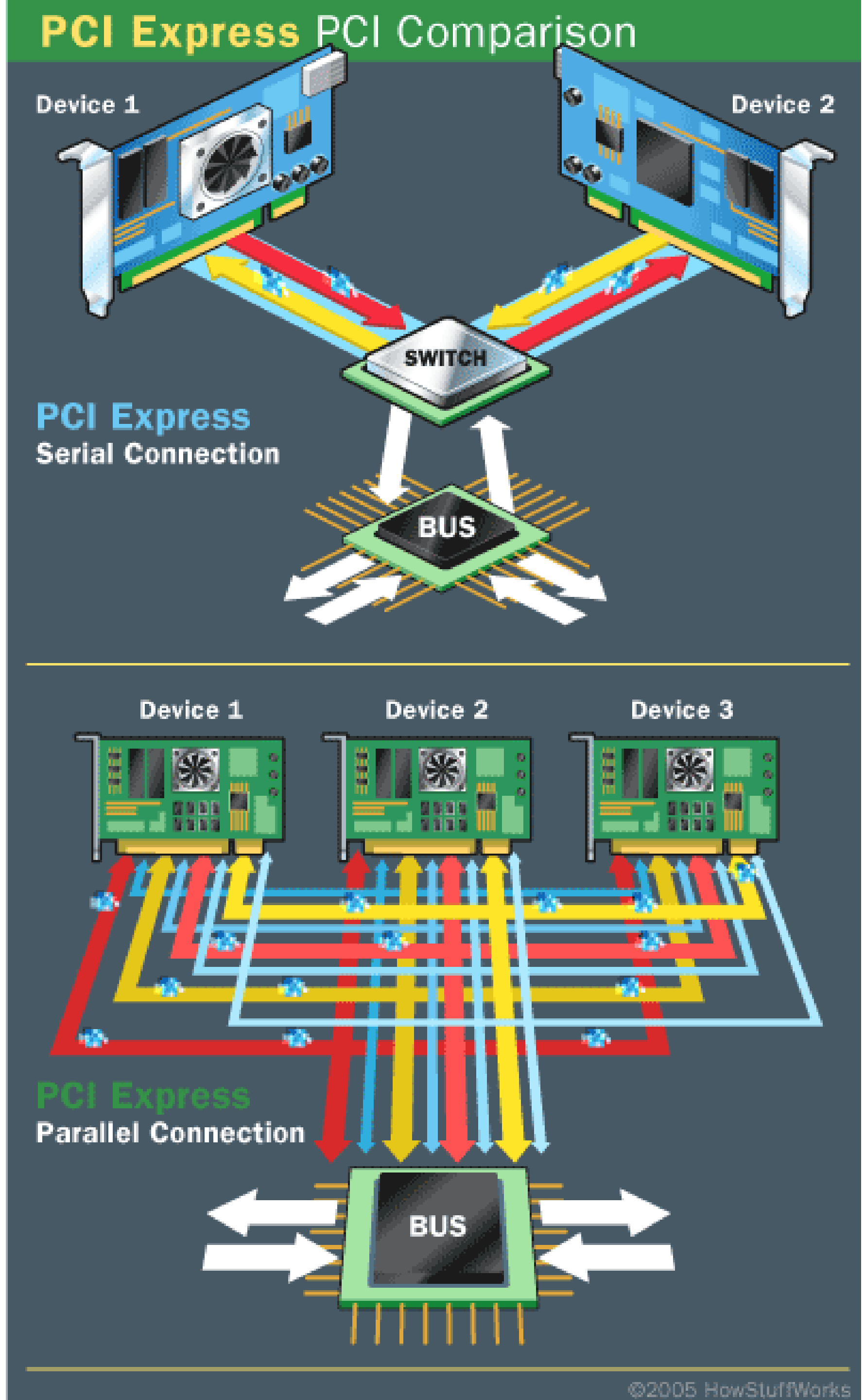
Sequential and parallel machines

- basic single stream processors
  - pipelined: basic sequential
  - superscalar: transparently concurrent
  - VLIW: compiler generated concurrency
- multiple stream
  - array processors
  - vector processors
- multiprocessors

Sequential processors

- operation
  - generally transparent to sequential programmer
  - appear as in order instruction execution
- pipeline processor
  - execution in order
  - limited to one instruction execution / cycle
- superscalar processor
  - multi instructions / cycle, managed by hardware
- VLIW
  - multi op execution / cycle, managed by compiler

PCI serial and parallel



http://computer.howstuffworks.com/pci-express1.htm

System Chip – Advantages and Issues

- Advantages
  - Easily reprogrammable, reuse of Intellectual property.
  - Lower chip area, reduce in silicon
  - Low cost

- Issues
  - On chip routing and timing.
  - Limitations of On chip resources

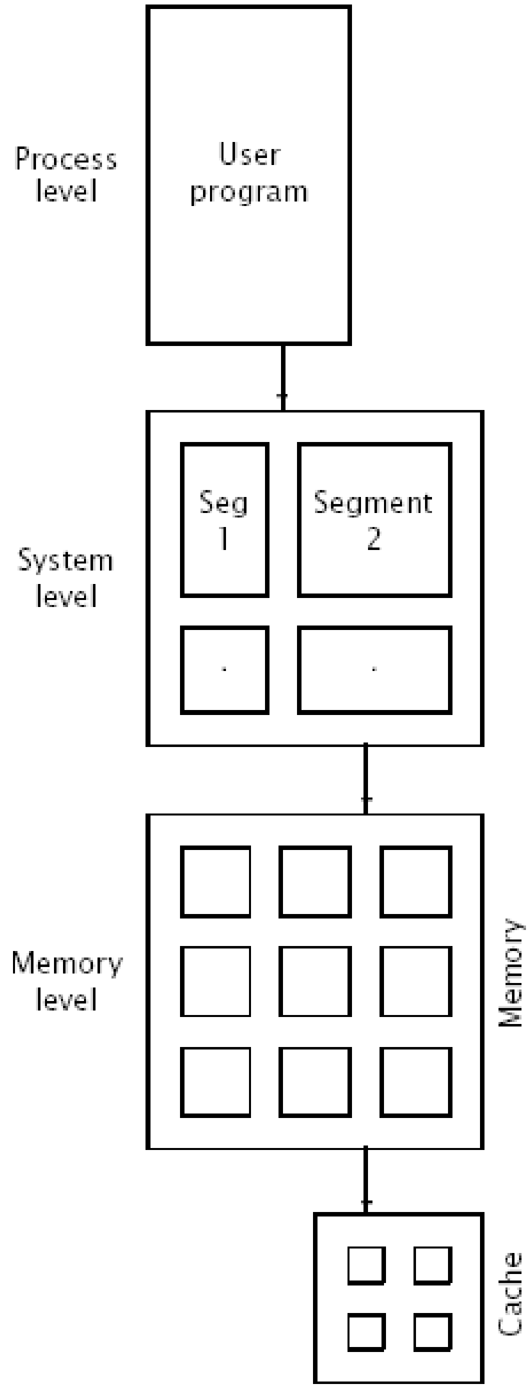
Parallel processors

- execution managed by programmer
- array processors
  - single instruction stream, multiple data streams: SIMD
- vector processors
  - SIMD
- multiprocessors
  - multiple instruction streams, multiple data streams: MIMD

Memory and addressing

- many SOC memory designs use simple embedded memory
  - a single level cache
  - real (rather than virtual) addressing
- as SOC become more complex
  - their designs are expected to use more complex memory and addressing configurations

Three levels of addressing



User writes program assuming user is the only occupier of memory.

System relocates and protects multiple user programs in a common memory system space.

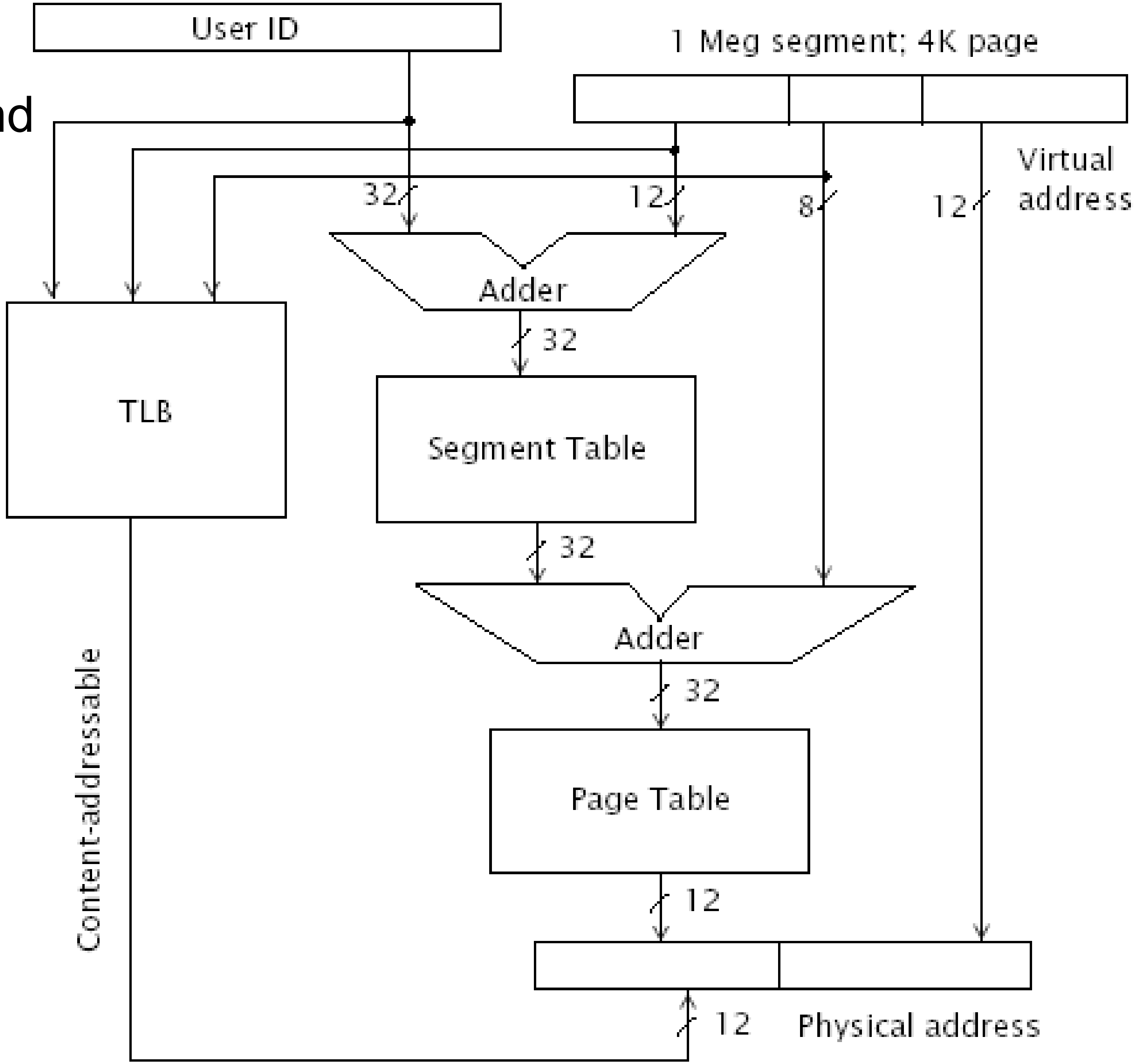
Memory system loads only those "localities," or pages, of the system space that are expected to be referenced shortly. The most active localities are loaded into the cache by hardware.

User view of memory: addressing

- a program: process address (offset + base + index)
  - ***virtual address:*** process address + process id
- a process: assigned a segment base and bound
  - ***system address:*** segment base + process address
- pages: active localities in main/real memory
  - virtual address: translated by table lookup to ***real address***
  - page miss: virtual pages not in page table
- **TLB** (translation look-aside buffer): recent translations
  - TLB entry: corresponding real and (virtual, id) address
- a few hashed virtual address bits address **TLB** entries

  - if virtual, id = **TLB** (virtual, id) then use translation

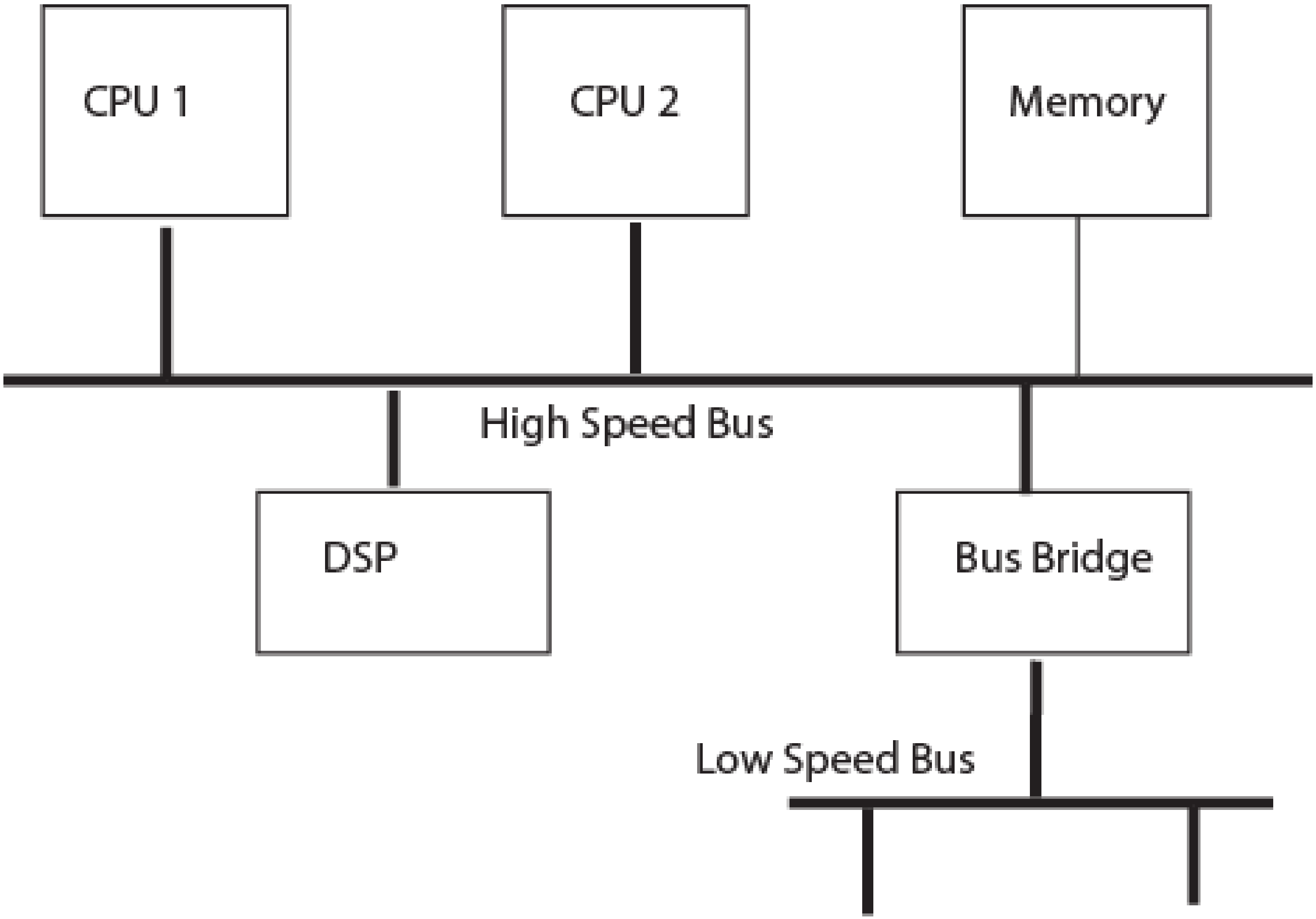
The TLB and The MMU



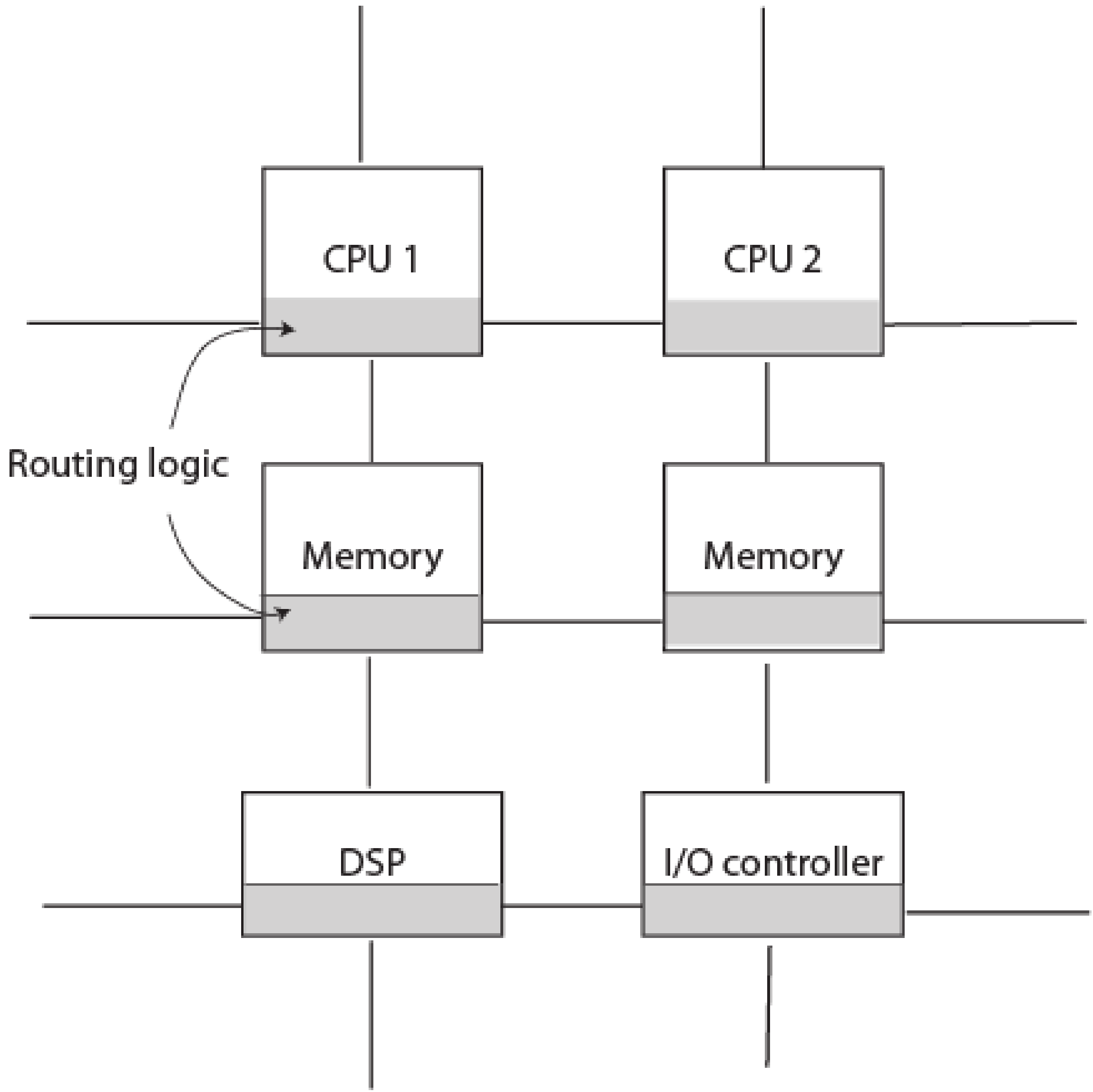
SOC interconnect

- interconnecting multiple active agents requires
  - bandwidth: capacity to transmit information (bps)
  - protocol: logic for non-interfering message transmission
- bus
  - AMBA (adv. Microcontroller bus architecture) from ARM, widely used for SOC
  - bus performance: can determine system performance
- network on chip
  - array of switches
  - statically switched: eg mesh
  - dynamically switched: eg crossbar

Bus based SOC



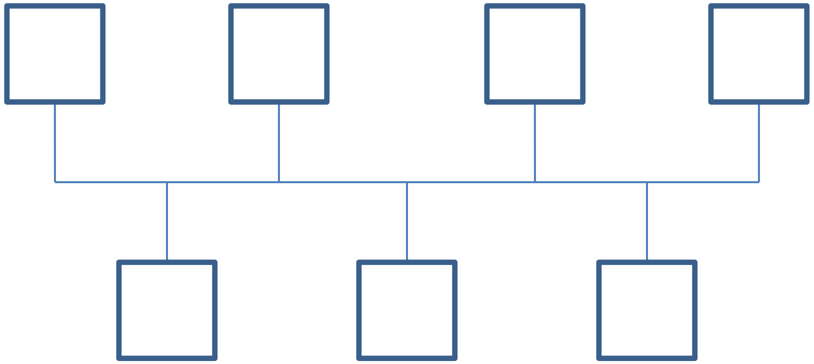
Network on a Chip



Networks-on-Chip (NoCs)

Motivation

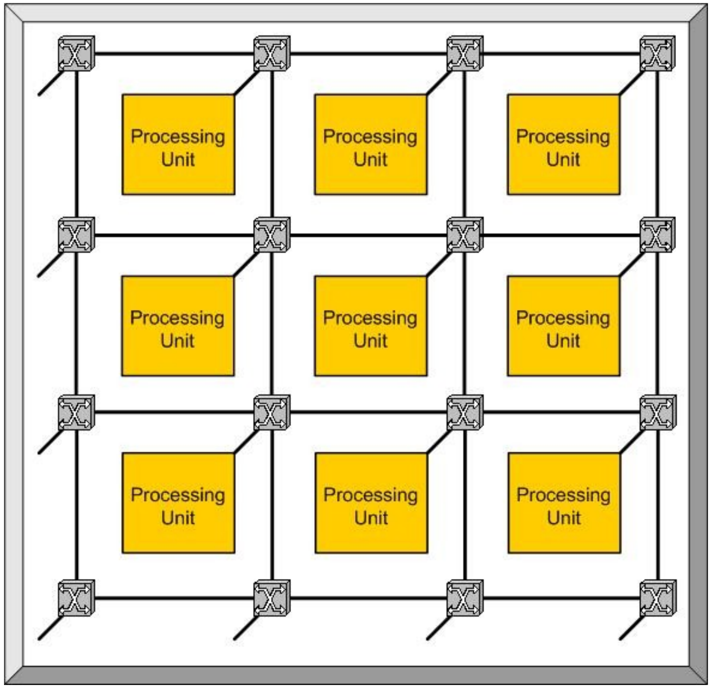
- Bus has been the most popular interconnect for multiprocessor systems
- When scaling feature sizes and frequency, wire delays remain larger than clock cycle
- When expanding to a many core-system, contention decreases throughput



- Need for interconnect with deterministic delays and scalability

What is a Network-on-Chip (NoC)?

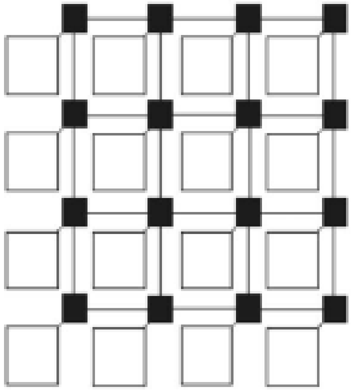
- Leveraging existing computer networking principles to improve inter-component intra-chip communications
- Each on-chip component connected by an intelligent switch to particular communication wire(s)
- Improvement over standard bus based interconnections for SoC architectures in terms of throughput



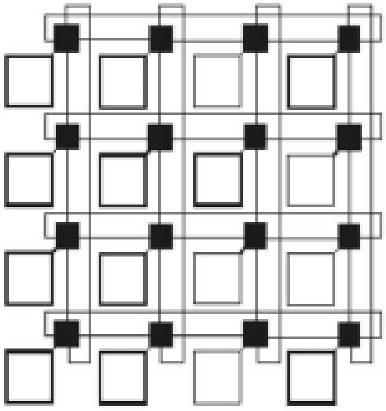
Networks on chip: advantages

- differentiated services
  - offer different kinds of communication with one network
- scalable
  - add routers and network interfaces for extra bandwidth (at the cost of additional latency)
- compositional
  - add routers/NIs without changing existing components e.g. timing, buffers
- efficient use of wires
  - statistical multiplexing/sharing (average vs. worst-case) $\Rightarrow$ fewer wires $\Rightarrow$ less wire congestion
  - point to point wires at high speed
- communication becomes re-usable, configurable IP

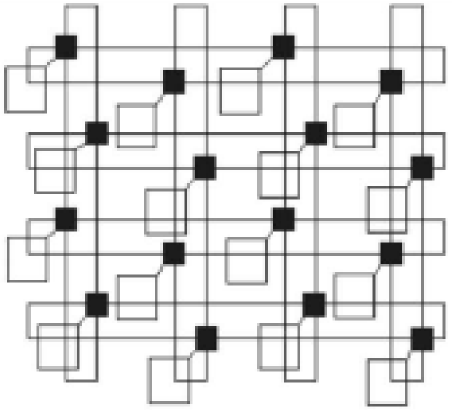
Topologies



CLICHÉ

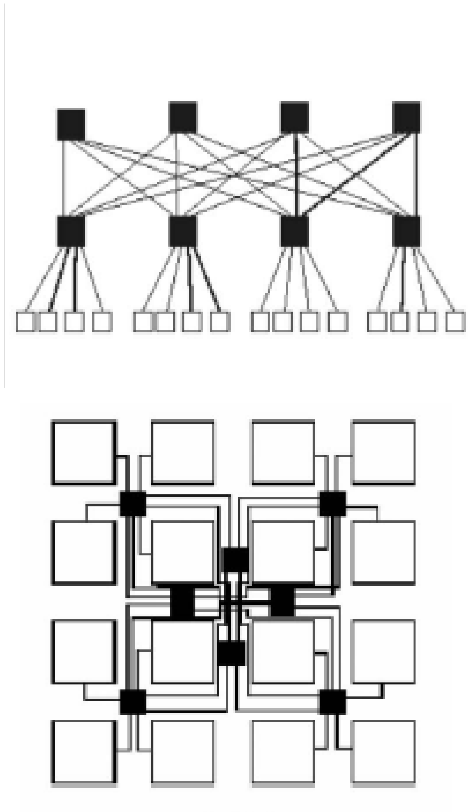


Torus

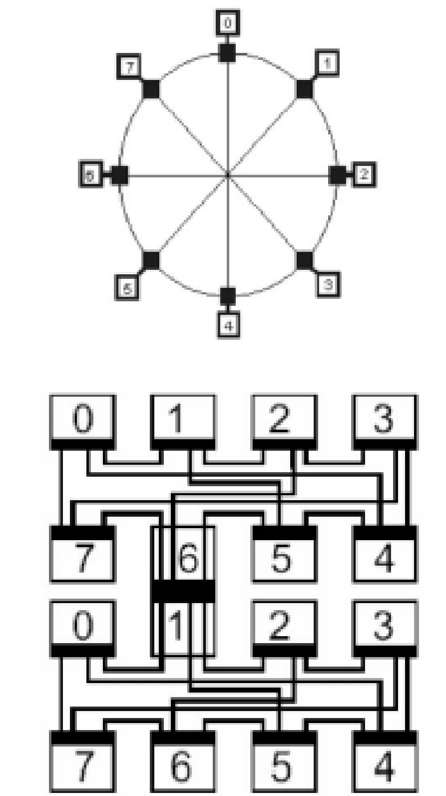


Folded torus

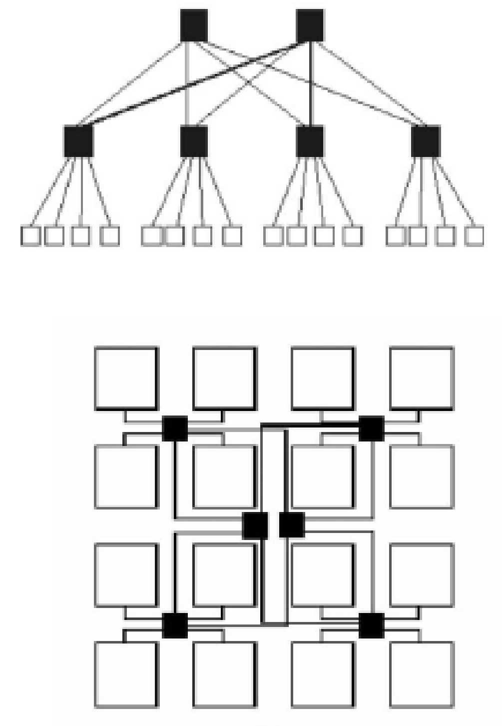
Topologies contd.



SPIN



Octagon



BFT

Dedicated wires vs. Network

Dedicated Wiring	On-Chip Network
Spaghetti wiring	Ordered wiring
Variation makes it hard to model crosstalk, returns, length, R & C.	No variation, so easy to exactly model XT, returns, R and C.
Drivers sized for 'wire model' – 99% too large, 1% too small	Driver sized exactly for wire
Hard to use advanced signaling	Easy to use advanced signaling
Low duty factor	High duty factor
No protocol overhead	Small protocol overhead

Switching

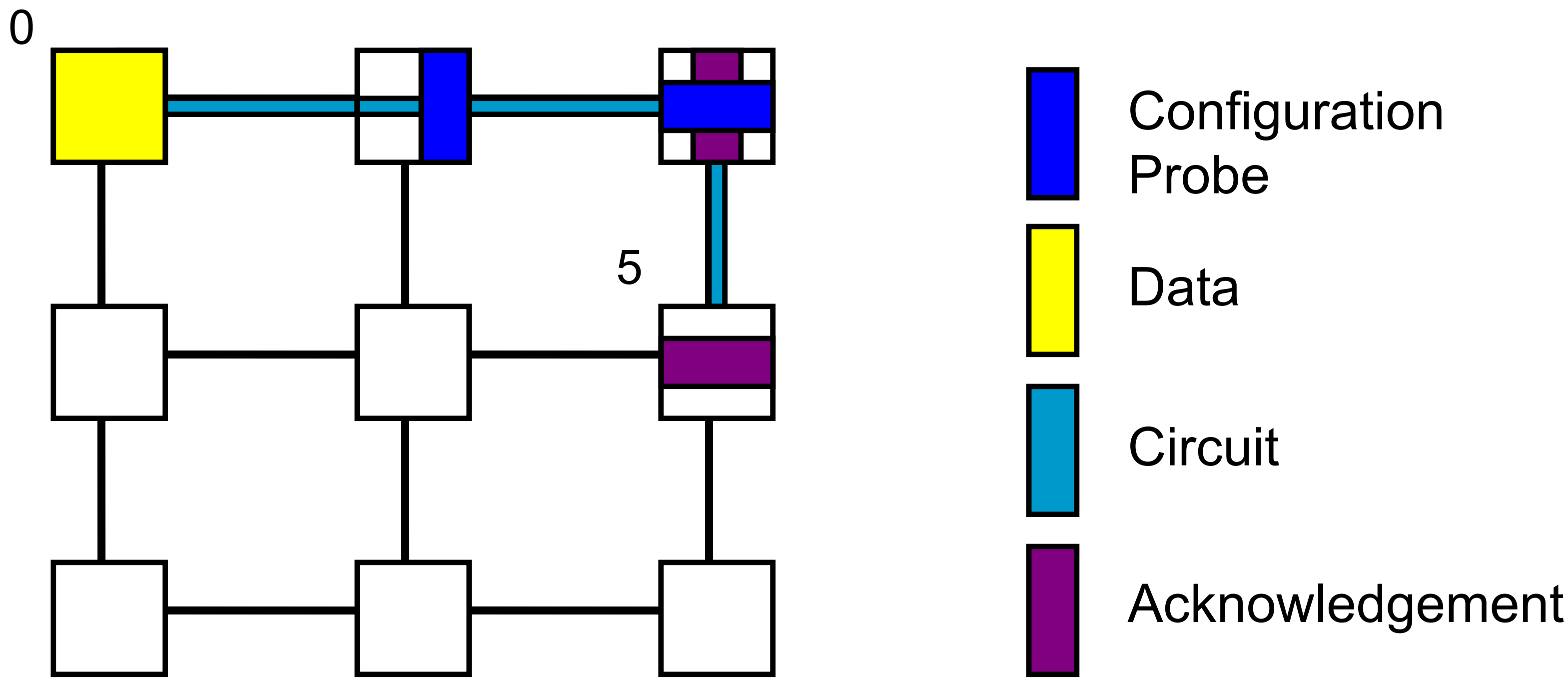
Circuit Switching

- Dedicated path, or circuit, is established over which data packets will travel
- Naturally lends itself to time-sensitive guaranteed service due to resource allocation
- Reservation of bandwidth decreases overall throughput and increases average delays

Packet Switching

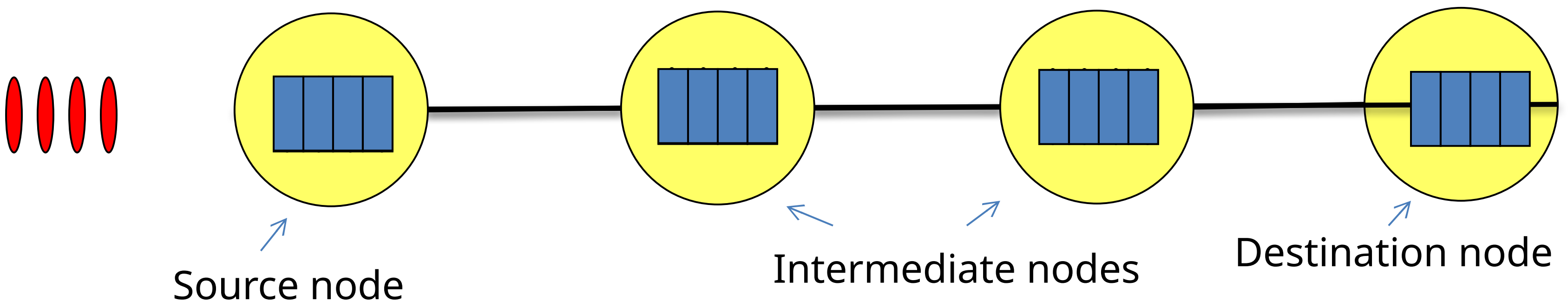
- Intermediate routers are now responsible for the routing of individual packets through the network, rather than following a single path
- Provides for so-called best-effort services
- Sharing of resources allows for higher throughput

Circuit Switching Example



- Significant latency overhead prior to data transfer
- Other requests forced to wait for resources

Store & Forward Switching

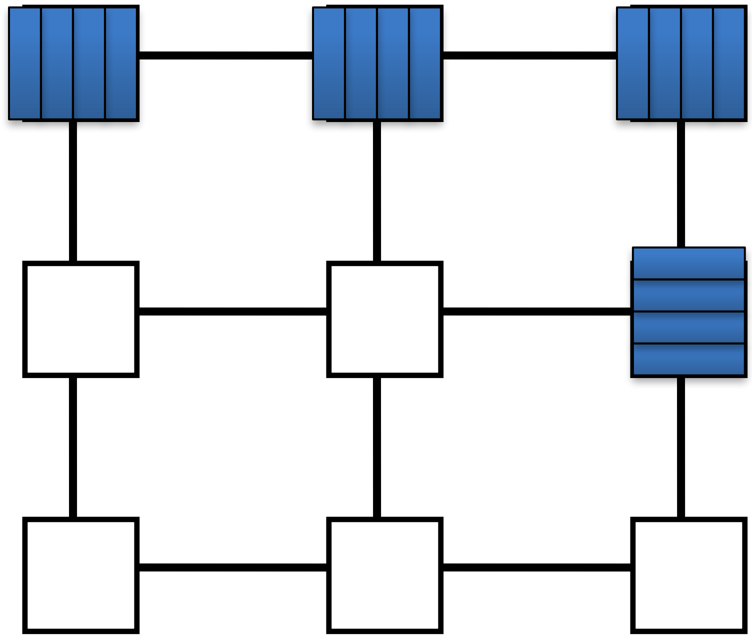


- Each node along a route waits until a packet is completely received (stored) and then the packet is forwarded to the next node
- Two resources are needed
  - Packet-sized buffer in the switch
  - Exclusive use of the outgoing channel

Store & Forward Switching

- S&F switching forwards a packet only when there is enough space available in the receiving buffer to hold the entire packet.
- Thus, there is no need for dividing a packet into flits. This reduces the overhead, as it does not require circuits such as a flit builder, a flit decoder, a flit stripper and a flit sequencer.
- Nevertheless, such a switching technique requires a large amount of buffer space at each node. Thus, it may not be a feasible solution for embedded applications

Store & Forward Switching Example



- High per-hop latency
- Larger buffering required

Store & Forward Switching

- Finer grained sharing of the link bandwidth
- Routing, arbitration, switching overheads experienced for each packet
- Increased storage requirements at the nodes
- Packetization and in-order delivery requirements
- Alternative buffering schemes
  – Use of local processor memory
  – Central (to the switch) queues

Switching contd.

<u>Wormhole Switching</u>

- Message is divided up into smaller, fixed length flow units called flits
- Only first flit contains routing information, subsequent flits follow
- Buffer size is significantly reduced due to the limitation on the number of flits needed to be buffered at any given time

<u>Virtual Channels</u>

- Allows for several instances of wormhole switching
- Additional buffers are added, which increases overall switch size, but significantly increases throughput

Performance Metrics

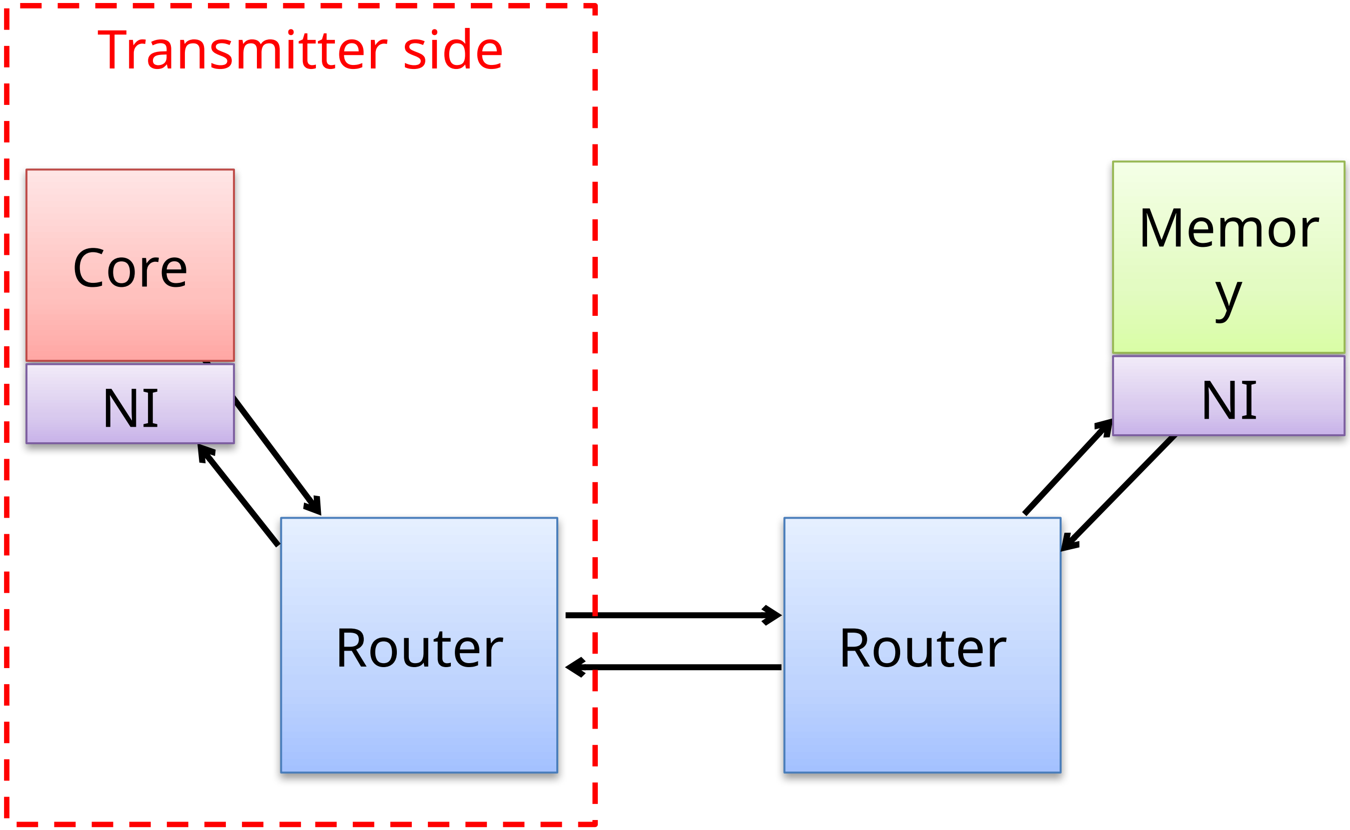
Simulator developed to measure:

- Throughput (in flits)
- Latency (of flits)
- Energy (per packet)

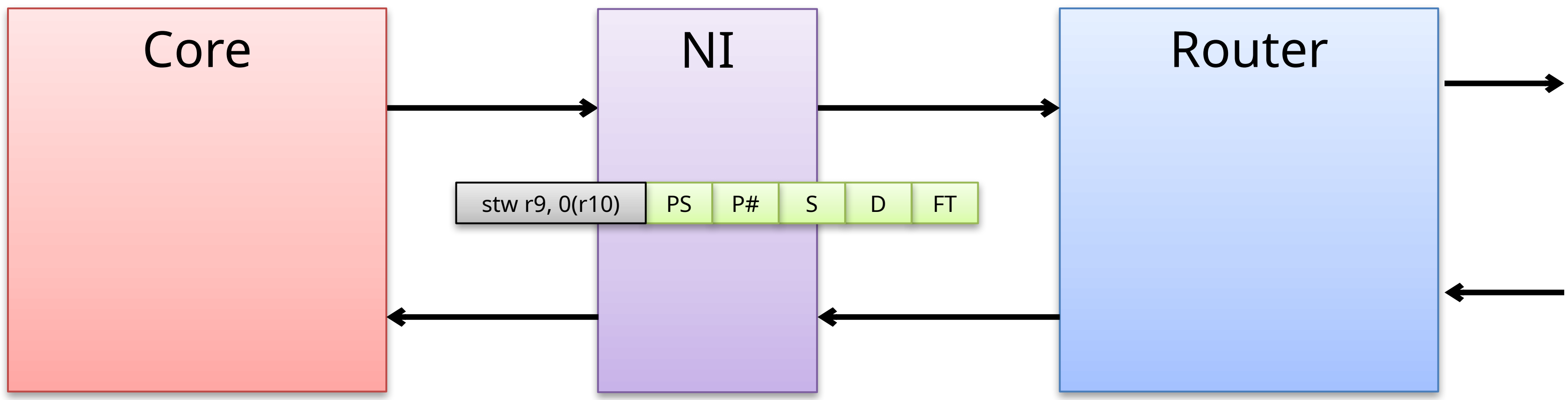
Hardware model developed to estimate:

- Area (router and link overhead)

Network Interface



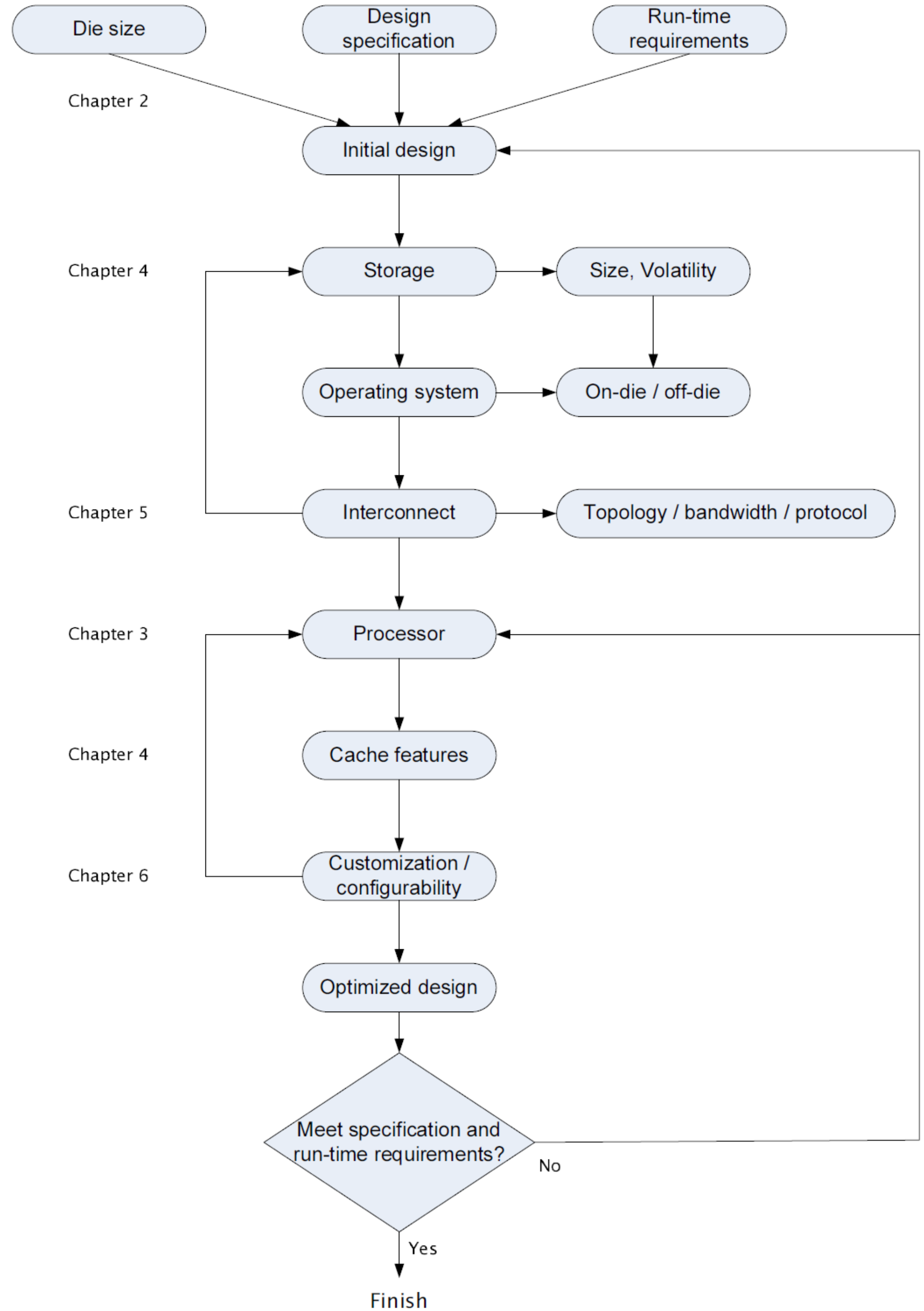
Network Interface



SOC design approach

- understand application (compiler, OS, memory and real time constrains)
- select initial die area, power, performance targets; select initial processors, memory, interconnect
- assume target processor and interconnect performance, design and evaluate memory
- evaluate and redesign processors with memory
- design interconnect to support processors and memory
- repeat and iterate to optimize

SOC design approach



Processor optimization example

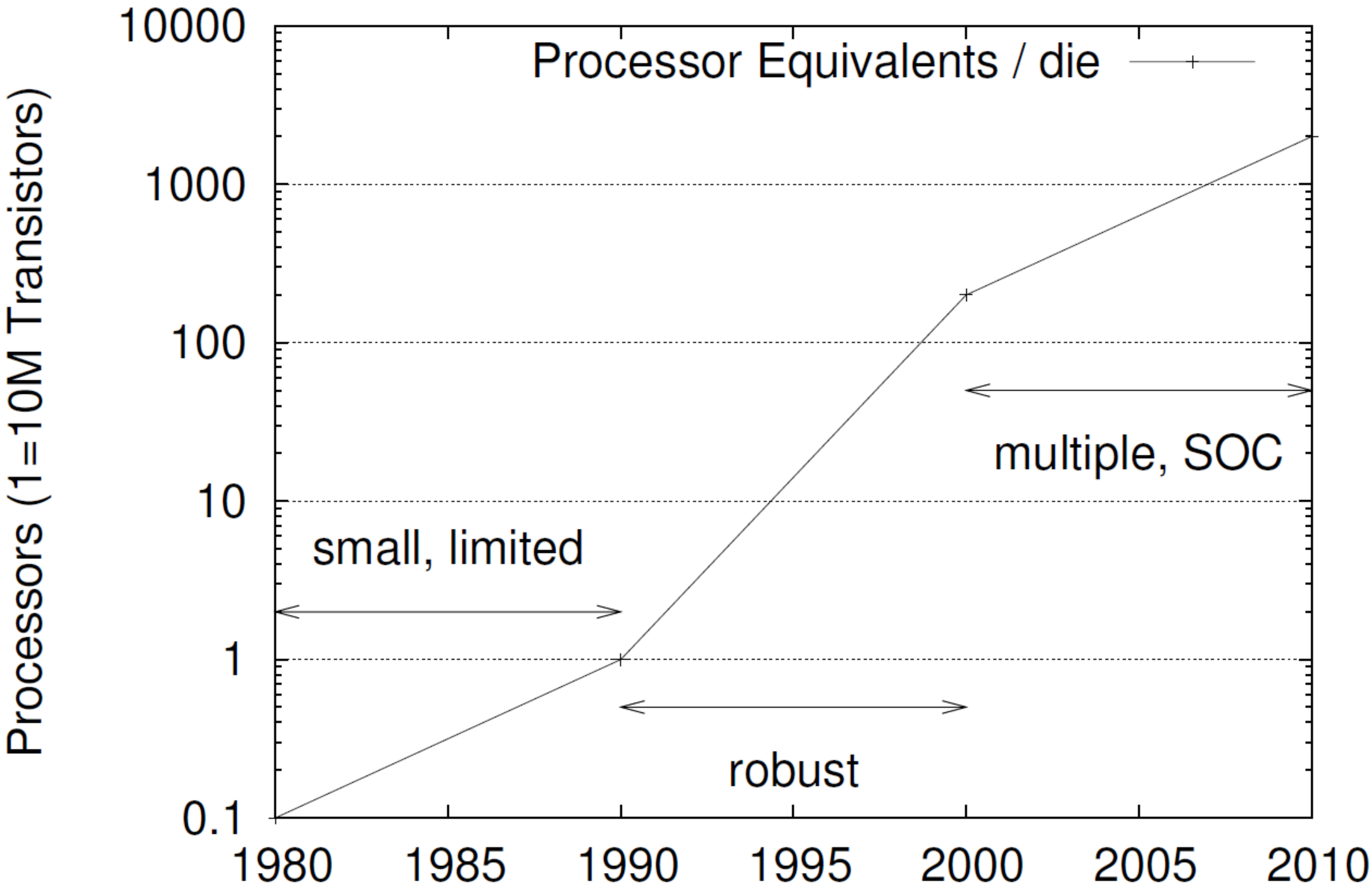
- given embedded ARM processor
  - in an SOC chip
- 1 IALU vs 2 IALU vs 3 IALU vs 4 IALU
  - *instructions per cycle?*
- 16k L1 instruction cache vs 32k L1 i-cache
  - *how much improvement? less power?*
- branch predictor: taken vs not-taken
  - *misprediction rate?*

- aim: explore this large design space

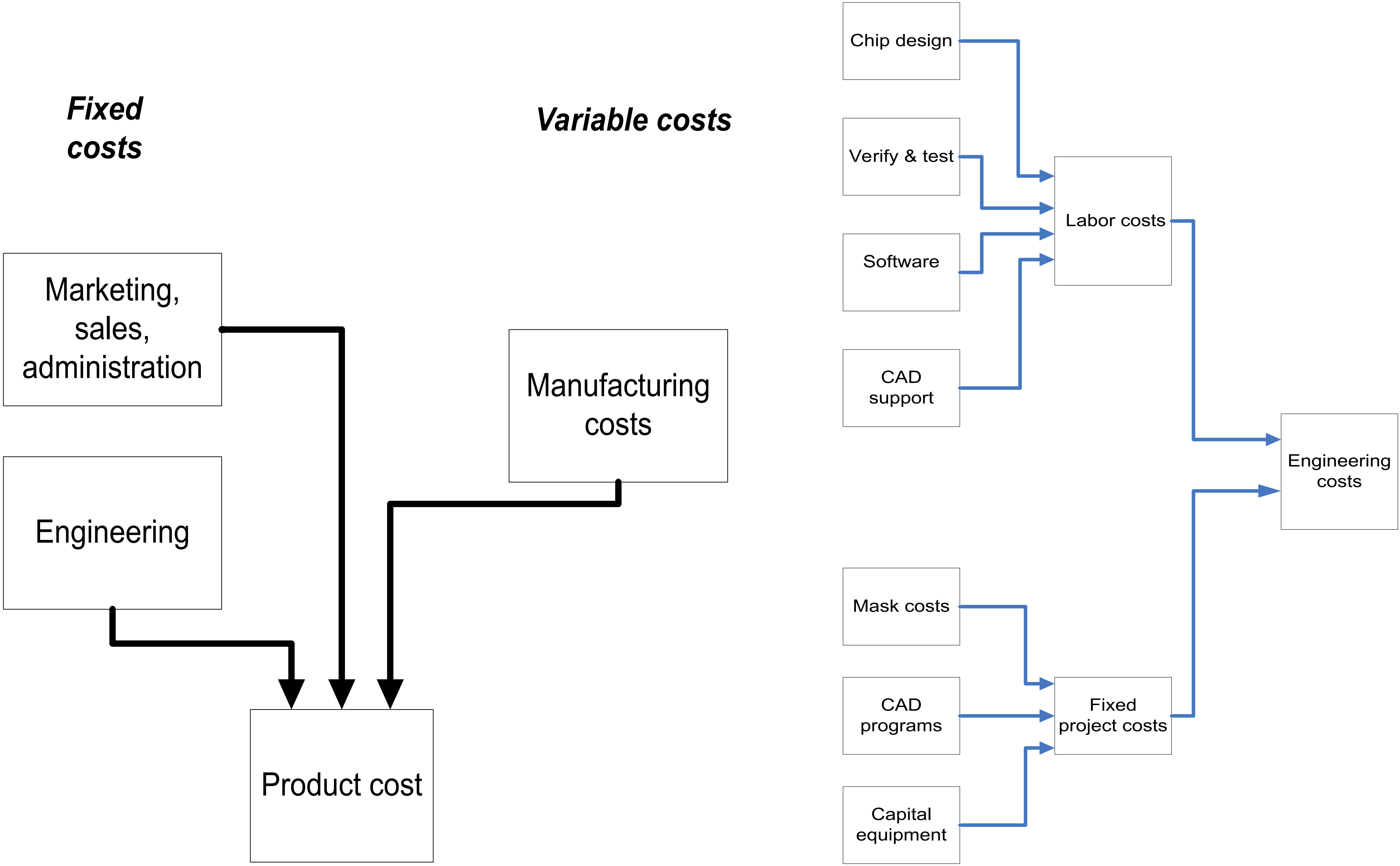
Design cost: product economics

- increasingly product cost determined by
  - design costs, including verification
  - not marginal cost to produce
- manage complexity in die technology by
  - engineering effort
  - engineering cleverness

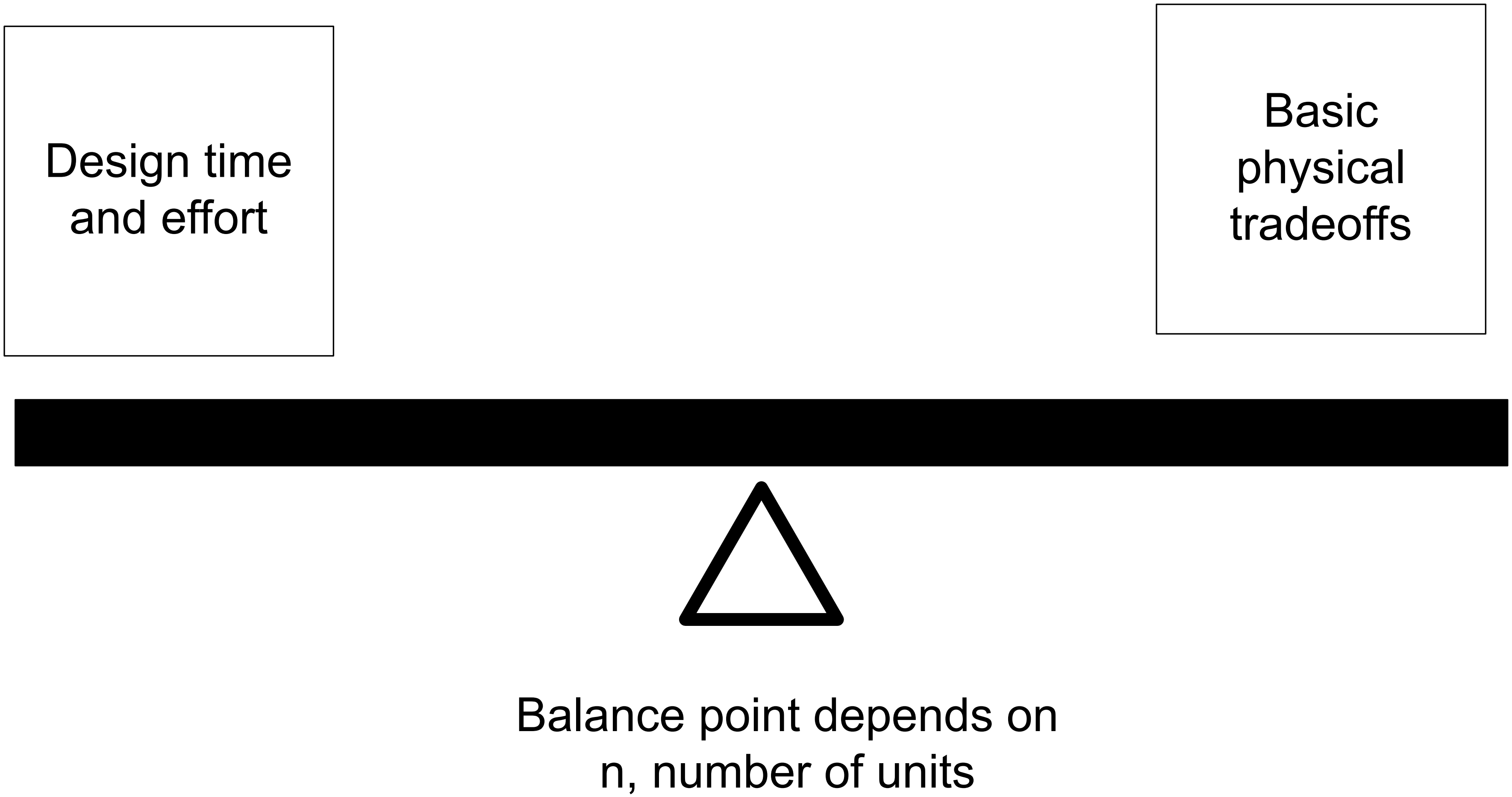
Design complexity



Cost: product program vs engineering



Product volume dictates design effort



Summary

- to design and evaluate an SOC, designers need to understand:
  - its components: processors, memory, interconnect
  - applications that it targets
- SOC economics heavily dependent on:
  - costs: initial design, marginal production
  - volume: applicability, lifetime
- reducing design complexity
  - Intellectual Property (IP)
  - reconfigurable technology

References

- Akram Ben Ahmed, Shohei Miura, A. Ben Abdallah, Run-Time Monitoring Mechanism for Efficient Design of Network-on-Chip Architectures, *to appear in the 6th International Workshop on Engineering Parallel and Multicore Systems (ePaMuS2013'), July 2013.*
- Akram Ben Ahmed, A. Ben Abdallah, **Low-overhead Routing Algorithm for 3D Network-on-Chip**, *IEEE Proc. of the The Third International Conference on Networking and Computing (ICNC'12), pp. 23-32, 2012.*
- Akram Ben Ahmed, A. Ben Abdallah, **LA-XYZ: Low Latency, High Throughput Look-Ahead Routing Algorithm for 3D Network-on-Chip (3D-NoC) Architecture**, *IEEE Proceedings of the 6th International Symposium on Embedded Multicore SoCs (MCSoC-12), pp. 167-174, 2012.*
- Akram Ben Ahmed, A. Ben Abdallah, **ONoC-SPL Customized Network-on-Chip (NoC) Architecture and Prototyping for Data-intensive Computation Applications**, *IEEE Proceedings of The 4th International Conference on Awareness Science and Technology, pp. 257-262, 2012.*
- A. Ben Ahmed, A. Ben Abdallah, **Efficient Look-Ahead Routing Algorithm for 3D Network-on-Chip (3D-NoC)**,IEEE Proceedings of the 6th International Symposium on Embedded Multicore SoCs (MCSoC-12,) pp. 167-174,2012.
- R. Okada, **Architecture and Design of Core Network Interface for Distributed Routing in OASIS NoC**, Technical Report, ASL- Parallel Architecture Group, School of Computer Science and Engineering, The University of Aizu, March 2012.
- A. Ben Ahmed, A. Ben Abdallah, K. Kuroda,**Architecture and Design of Efficient 3D Network-on-Chip (3D NoC) for Custom Multicore SoC**, IEEE Proc. of the 5th International Conference on Broadband, Wireless Computing, Communication and Applications (BWCCA-2010), pp.67-73, Nov. 2010. (**best paper award**) (**slides**)
- K. Mori, A. Esch, A. Ben Abdallah, K. Kuroda, **Advanced Design Issues for OASIS Network-on-Chip Architecture**, IEEE Proc. of the 5th International Conference on Broadband, Wireless Computing, Communication and Applications (BWCCA-2010),pp.74-79, Nov. 2010. **slides**
- T. Uesaka, **OASIS NoC Topology Optimization with Short-Path Link**, Technical Report, Systems Architecture Group,March 2011
- K. Mori, A. Ben Abdallah, OASIS NoC Architecture Design in Verilog HDL, Technical Report,TR-062010-OASIS, Adaptive Systems Laboratory, the University of Aizu, June 2010. **slides**
- Shohei Miura, Abderazek Ben Abdallah, Kenichi Kuroda, PNoC: Design and Preliminary Evaluation of a Parameterizable NoC for MCSoC Generation and Design Space Exploration, The 19th Intelligent System Symposium (FAN 2009), pp.314-317, Sep.2009.
- Kenichi Mori, Abderazek Ben Abdallah, Kenichi Kuroda, **Design and Evaluation of a Complexity Effective Network-on-Chip Architecture on FPGA**, The 19th Intelligent System Symposium (FAN 2009), pp.318-321, Sep. 2009.
- A. Ben Abdallah, T. Yoshinaga and M. Sowa, "**Mathematical Model for Multiobjective Synthesis of NoC Architectures**", IEEE Proc. of the 36th International Conference on Parallel Processing, Sept. 4-8, 2007.
- A. Ben Abdallah, Masahiro Sowa, "**Basic Network-on-Chip Interconnection for Future Gigascale MCSoCs Applications: Communication and Computation Orthogonalization**", JASSST2006, Dec. 4-9th, 2006.

1. Book: **Multicore Systems-on-Chip: Practical Hardware/Software Design, 2nd Edition**, Author: A. Ben Abdallah, Publisher: Springer, (2013) , ISBN-13: 978-9491216916. [Amazon]