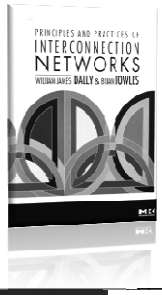
Principles of interconnection networks

Midia Reshadi, Ph.D
CE Department of Science and research Branch
of Islamic Azad University (SRBIAU)
reshadi@ieee.org

Outline

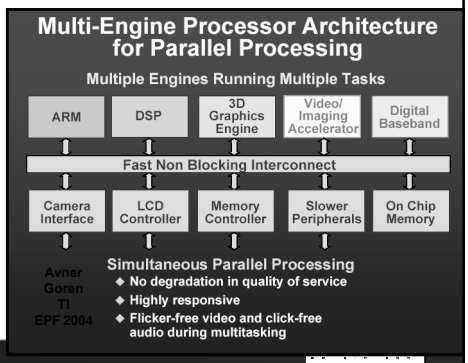
- Principles and Concepts
- How we walk through the concepts
- What we do next?

The base reference of this presentation. The Figures have been snap shot from this book

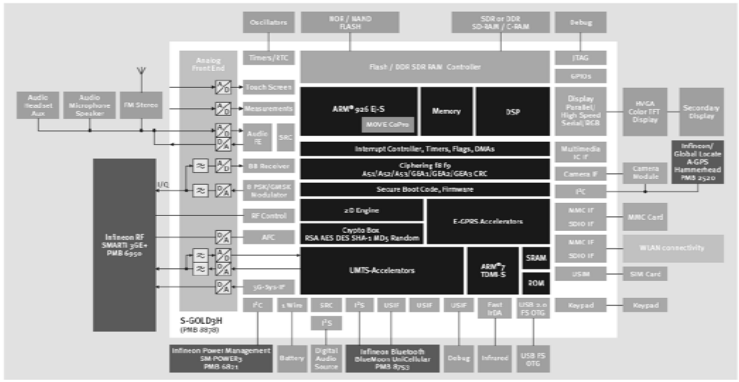


IP Core

- define core as a pre-designed, pre-verified hardware piece that can be used as a building block for large and complex applications on an IC
  - R. Gupta, Y. Zorian, Introducing core-based system design, IEEE Des. Test Comput. 14 (4) (1997) 15–25.
- IP core can be:
  - Processor /Processing Element (PE)
  - Memory
  - DSP
  - IO module
  - …

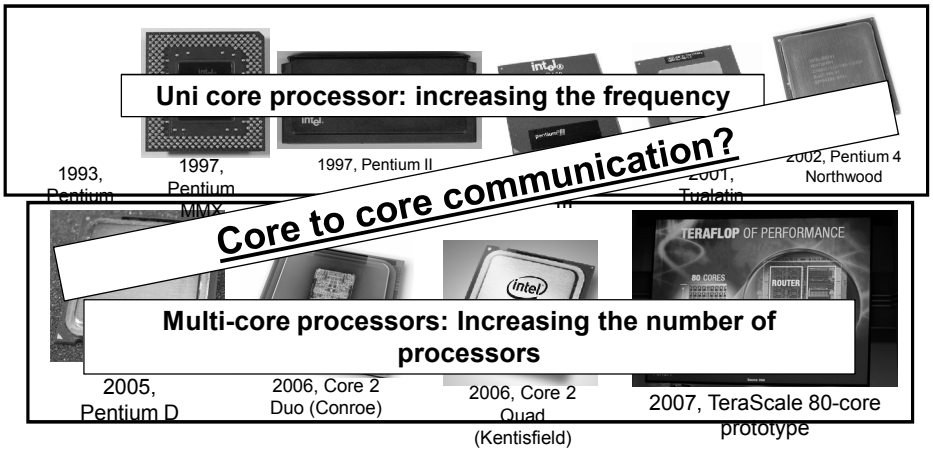


Example: *S-GOLD3H*



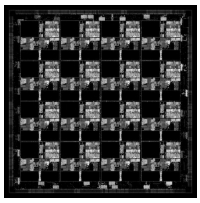
F. Angiolini, Interconnection system for highly integrated computation device, Ph.D Theis.

Trend

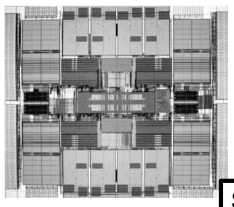


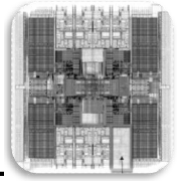
©Midia Reshadi

Multi-core: Chip multiprocessor era

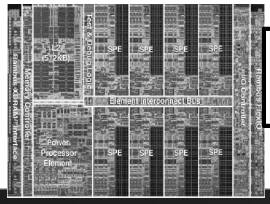


RAW
MIT 16 Ips
2002





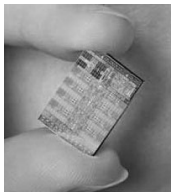
Sun Niagara
8 cores
2005

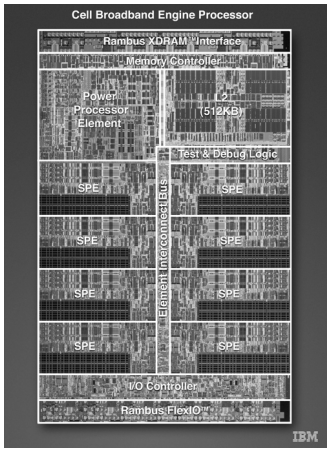


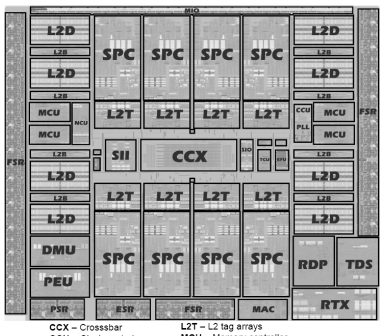
Sony/Toshiba/IBM Cell
9 cores
2006

©Midia Reshadi

IBM Cell

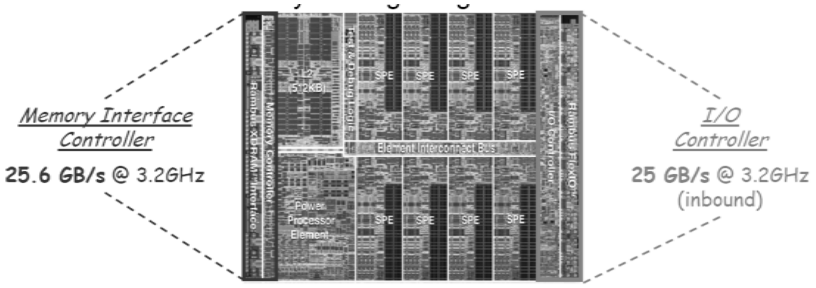




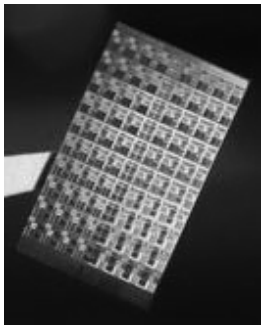


CCX – Crosssbar
CCU – Clock control
DMU/PEU – PCI Express
EFU – Efuse for redundancy
ESR – Ethernet SERDES
FSR – FBD SERDES
L2B – L2 write-back buffers
L2D – L2 data arrays
L2T – L2 tag arrays
MCU – Memory controller
MIO – Miscellaneous I/O
PSR – PCI Express SERDES
RDP/TDS/RTX/MAC – Ethernet
SII/SIO – I/O data path to and from memory
SPC – SPARC cores
TCU – Test and control unit

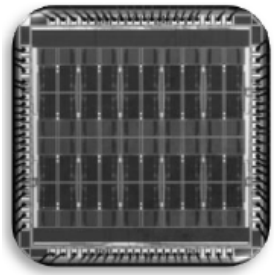
IBM Cell



[Kistler *et al.*, *IEEE Micro* **26** (3) 10–23 (2006)]

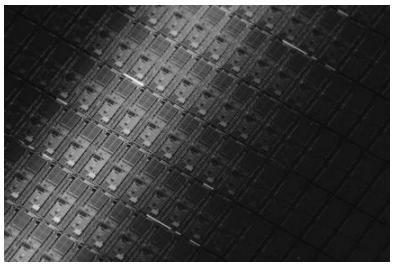


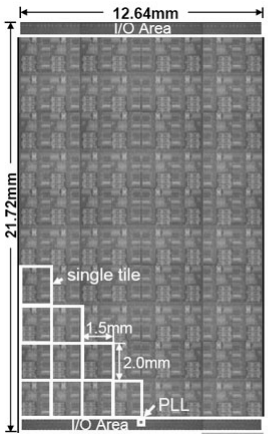
Intel Polaris
80 cores
2007

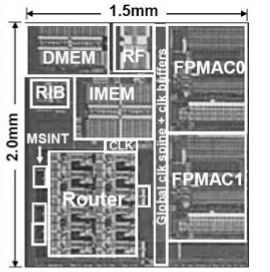


IntellaSysSEAforth40C18
40 cores
2008

Intel 80 Core: Polaris

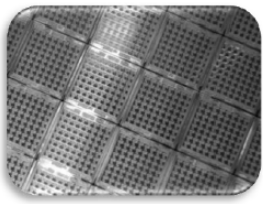




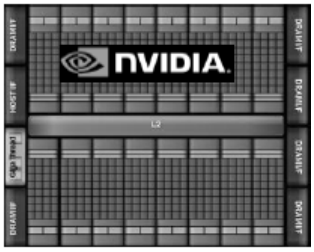


Technology	65nm CMOS Process
Interconnect	1 poly, 8 metal (Cu)
Transistors	100 Million
Die Area	275mm²
Tile area	3mm²
Package	1248 pin LGA, 14 layers, 343 signal pins

Cont,…

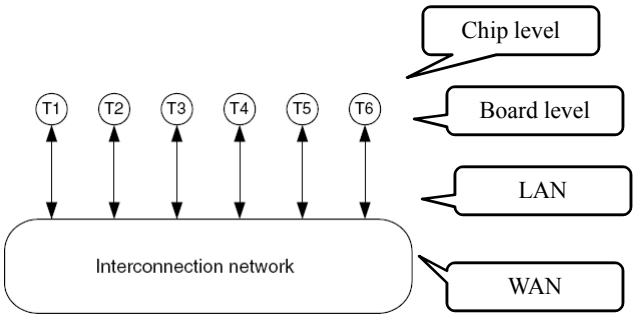


Tilera TILE-Gx100
100 cores
2009



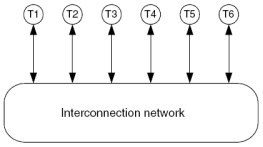
NVIDIA Fermi
512 cores
2010

What is the interconnection network?



Uses of Interconnection Networks

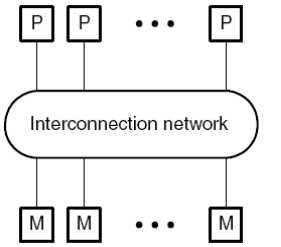
1. The number of terminals
2. The *peak bandwidth of each terminal*
3. The *average bandwidth of each terminal*
4. The required *latency*
5. The *message size or a distribution of message sizes*
6. The *traffic pattern(s) expected*
   - For example, each terminal might send messages to all other terminals with equal probability. This is the *random* traffic pattern.
7. The required *quality of service*
8. The required reliability and availability of the interconnection network



Uses of Interconnection Networks

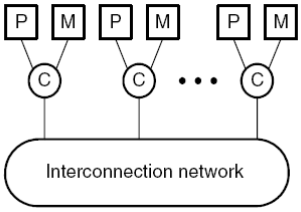
7. The required *quality of service*
   - QoS involves the *fair allocation of resources under some service policy*
   - For example, when multiple messages are contending for the same resource in the network, this contention can be resolved in many ways.
   - FCFS order based
8. The required reliability and availability of the interconnection network
   - *Reliability is a measure of how often the network correctly* performs the task of delivering messages
   - The *availability of a network is the fraction of* time it is available and operating correctly
     - In an Internet router, an availability of 99.999% is typically specified — less than five minutes of total downtime per year.

Processor-Memory Interconnect



Dance-hall

Example:
Cray T3E,



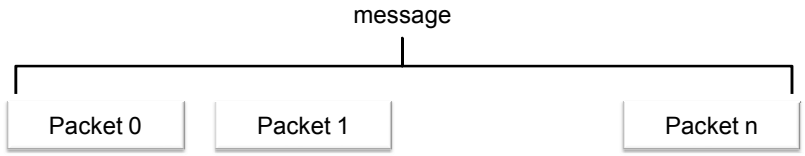
Integrated node

©Midia Reshadi

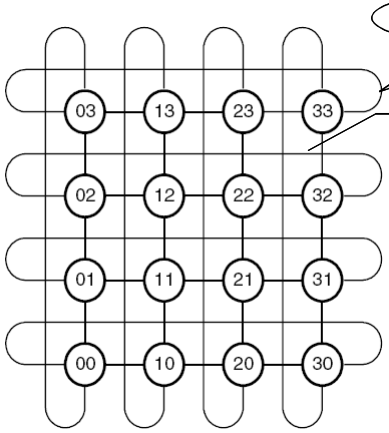
Parameter	*Value*
Processor ports	1–2,048
Memory ports	0–4,096
Peak bandwidth	8 Gbytes/s
Average bandwidth	400 Mbytes/s
Message latency	100 ns
Message size	64 or 576 bits
Traffic patterns	arbitrary
Quality of service	none
Reliability	no message loss
Availability	0.999 to 0.99999

©Midia Reshadi

Note: message and packet



Topology

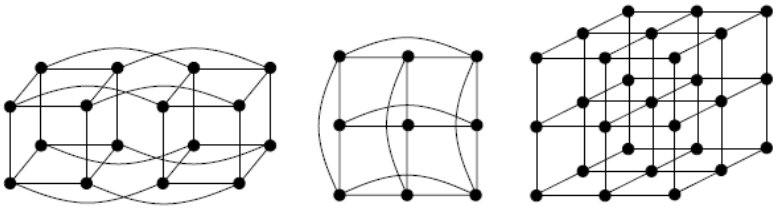


How to model, define?

∀ link:
# of signals:B bits
operating frequency= f hz
Date rate= B.f Bits/sec

Topology can be modeled by a graph G(V,E):
V: vertex
E: Edges

Topology (Cont,…)



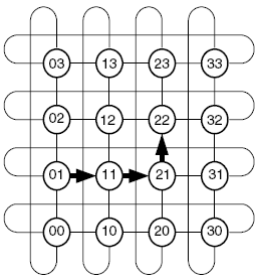
Define as matrix to import to the program

Graph lab tool

Routing

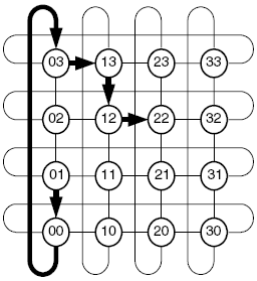
• The routing method employed by a network determines the path taken by a packet from a source terminal node to a destination terminal node.

$$P = \{c_1, c_2, \ldots, c_k\},$$



•Dimension order routing
•routing first in the x-dimension to reach node 21 and then
• in the y-dimension to reach destination node 22.
•This route is a *minimal route*

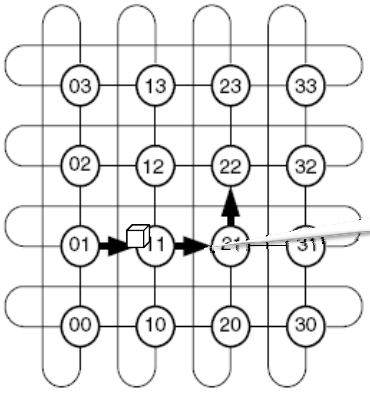
Routing (Cont,…)



This route is non-minimal, taking 5 hops rather than the minimum of 3.

Routing

routes based on the Dimension-ordered routing From 11 to 22



load

Deterministic routing
Adaptive routing

Flow Control

- Flow control manages the allocation of resources to packets as they progress along their route.
- Resources:
  - Channel
  - Buffer
    - Register
    - Memory

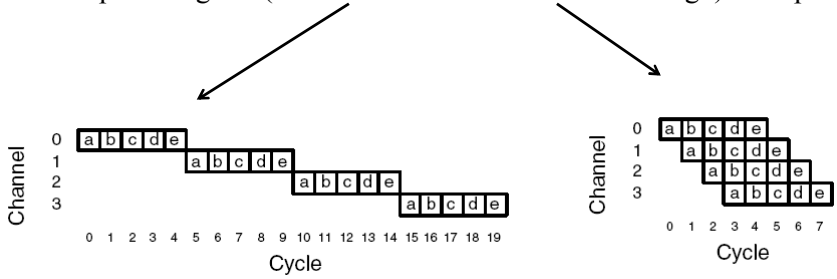
Routing

Topology

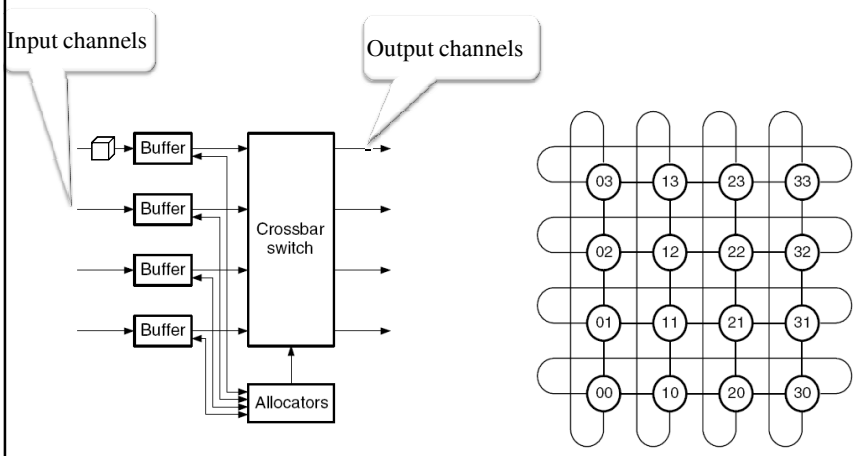
Flow control

Flow control

- Deadlock
- Time space diagram (store and forward/ virtual cut through) examples



Router Architecture

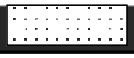


Home work

- Write your idea about following articles:
  - How do you define/describe a topology in your program, I recommend to use Matlab for the test of your idea. This framework lets you to abstract the algorithm and eases the verification.
- Send your solutions to ce.srbiau@gmail.com
- The email subject keyword: PD_P01_89_90_1
- Write your idea in Persian or English (I strongly recommend the English language)

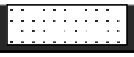
Simulation: Concurrent VLSI Architecture group at Stanford University.

- CVA project
- Downloading:
- svn co https://booksim.svn.sourceforge.net/svnroot/booksim
- Dependency
  - G++ ≥3
- parser
  - LEX
  - YCC



Build & run

- Build:
  - make
- run
  - ./booksim [configfile]



Examples

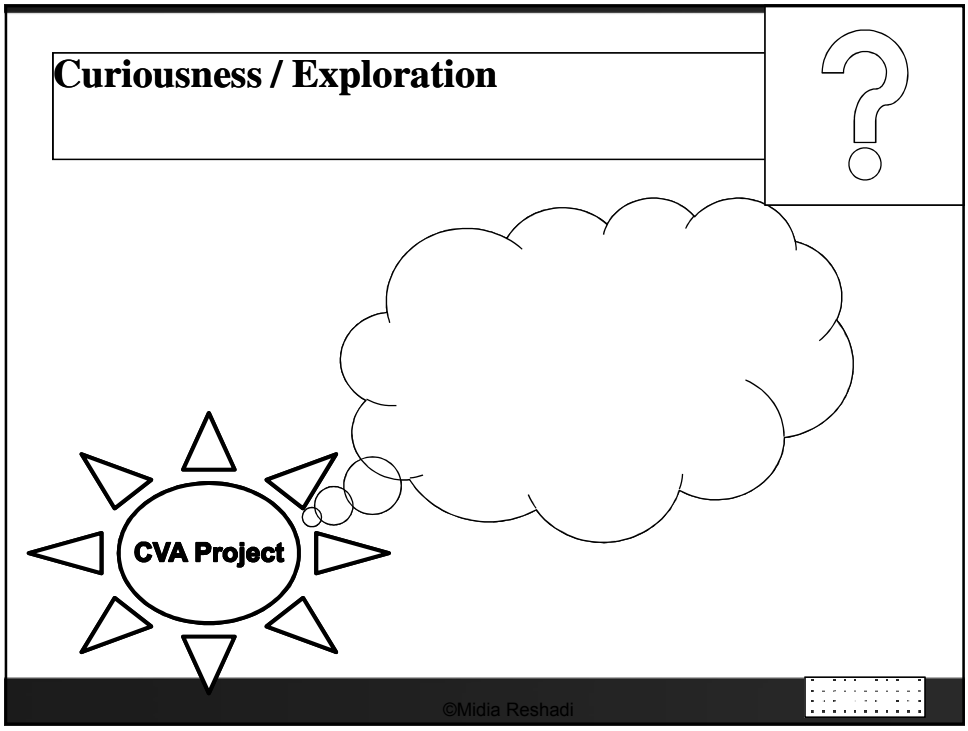
```
// Topology
topology = torus;
k = 4;
n = 2;
// Routing
routing_function = dim_order;
// Flow control
num_vcs = 1;
vc_buf_size = 4
// Traffic
traffic = uniform;
injection_rate = 0.15;
```

```
topology = single;
in_ports = 10;
out_ports = 10;
routing_function = single;
// Flow control
num_vcs = 1;
vc_buf_size = 4
traffic = uniform;
use_read_write = 0;
injection_rate = 0.4;
```


Examples

```
// Topology
topology = mesh;
k = 4;
n = 2;
// Routing
routing_function = dor;
// Flow control
num_vcs = 1;
vc_buf_size = 4
// Traffic
traffic = uniform;
injection_rate = 0.15;
```

CMESH?


Curiousness / Exploration
CVA Project
©Midia Reshadi