NATIONAL TSING HUA UNIVERSITY

Welcome to The Logic Design Lab!

Fall 2021
Parallel Systems

Prof. Chun-Yi Lee

Department of Computer Science
National Tsing Hua University

Announcements

- Lab 6
  - Lab 6 car demonstration on **12/23/2021 (Thu)**
  - **Please read the lab description and specification carefully**
  - **Compile your codes again before merging yours with your teammates**
  - **Please follow the template I/Os and submit your .v files**

- Final project
  - You can begin working on your final projects from now
  - Please avoid copying the works from past years

Final Project

- Use your FPGA to implement creative and interesting works
- Final project report due on **1/14/2022 (Fri)**

Final Project Presentation

- Final project demonstration on **1/13/2022 (Thu)** and **1/14/2022 (Fri)**
  - Around 10 minutes per team

- **No restriction on your presentation style**
  - Be creative!
  - What is special and new in your project
  - Key features

- Order of presentation
  - We will randomly decide the order
  - Let us know if you have constraints

Final Project Report

- Final project report
  - Due on **1/14/2022, 23:59pm (Fri)**
  - Block diagrams and state transition diagrams
  - **Detailed explanation**

- Report contents
  - Introduction
  - Motivation
  - System specification
  - Experimental results
  - Conclusion
  - …And any other sections that you would like to include

Final Project Award

- **Category:** Difficulty and completeness
  - Graded by me and the TAs
    - Difficulty (**35%**)
    - Completeness (**30%**)
    - Source code coding style (**15%**) (Correctness, usage, comments, etc.)
    - Report (**20%**)
  - First place: **NTD $6,000**
  - Second place: **NTD $3,000**
  - Third place: **NTD $1,500**

- **Category:** Best creativity
  - Voted by everyone in the class
  - **NTD $1,000**

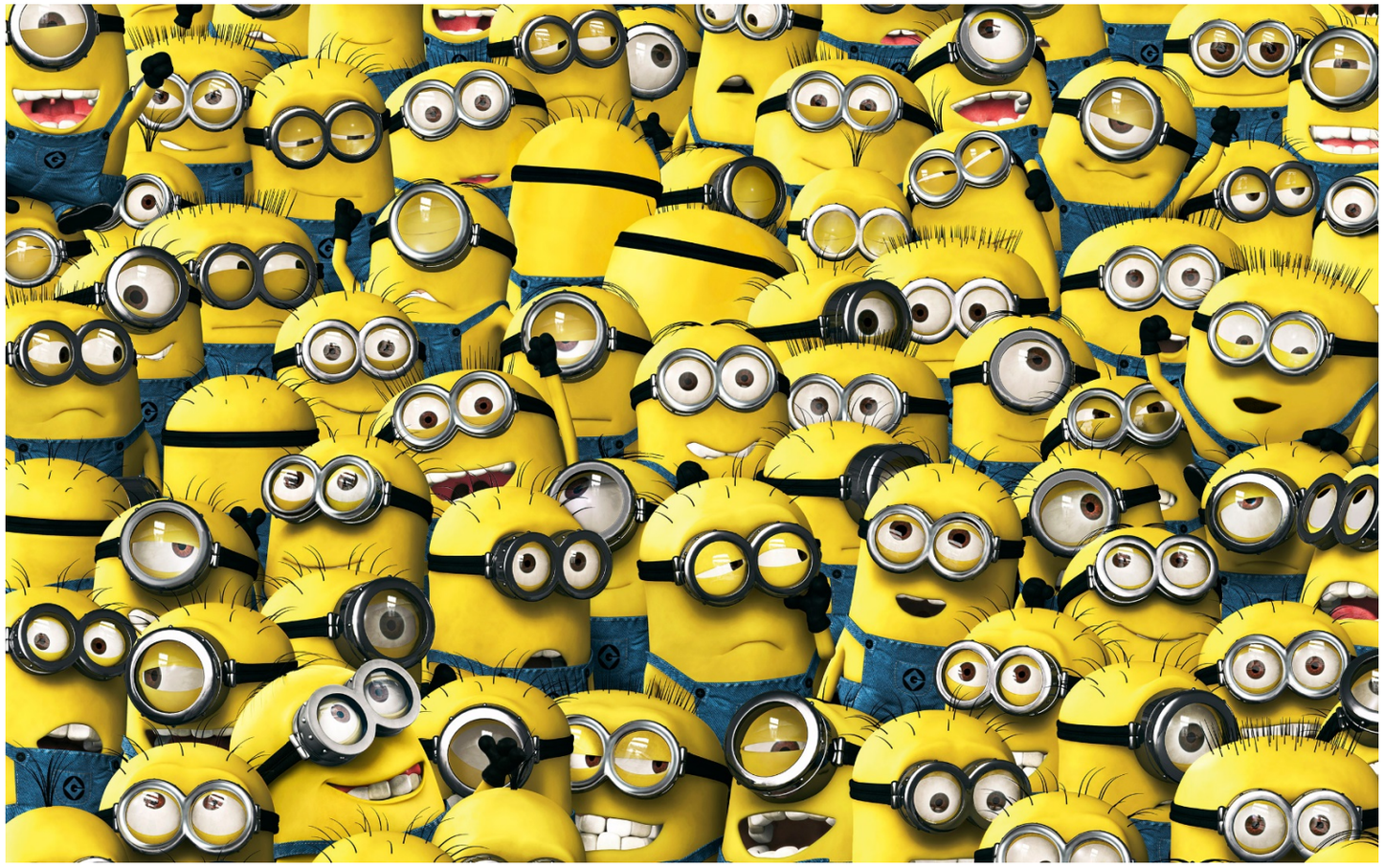
- **Category:** Deep learning models in FPGA
  - FPGA has to be the main device for executing the deep learning models
  - Extra **bonus points** and **cash rewards**. To be announced

"Sometimes it is the people who no one imagines anything of who do the things that no one can imagine."

-Alan Turing

Picture cited from http://theimitationgamemovie.com/ for educational purpose only

Why Parallel Computers?

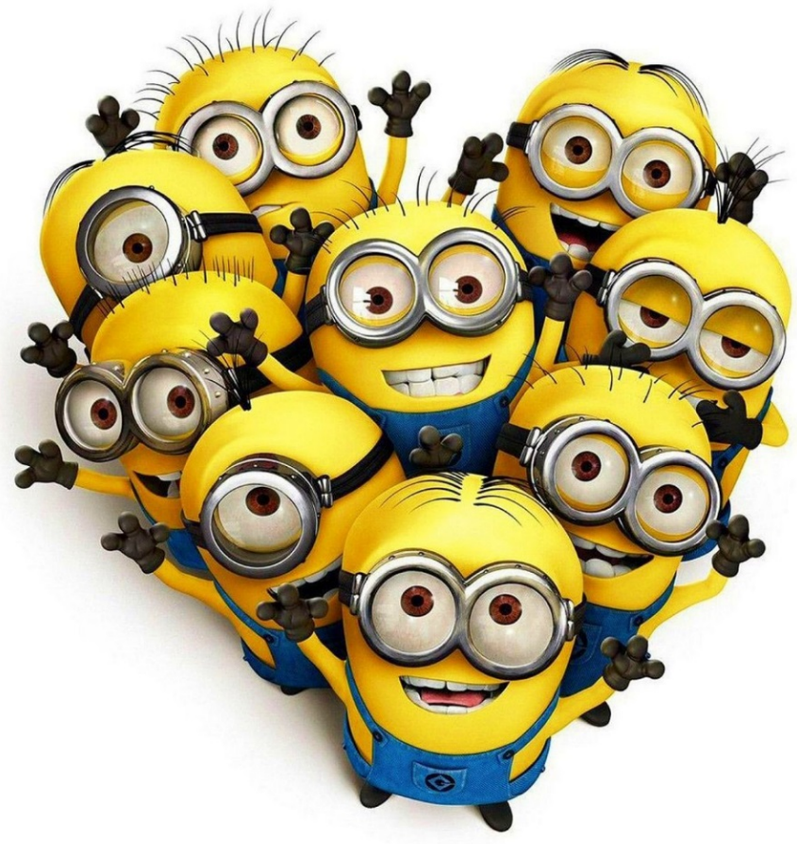


- A parallel computer is a collection of processing elements that cooperates to solve problems quickly
  - We care about performance
  - We care about efficiency
  - We are going to use multiple processors to get it

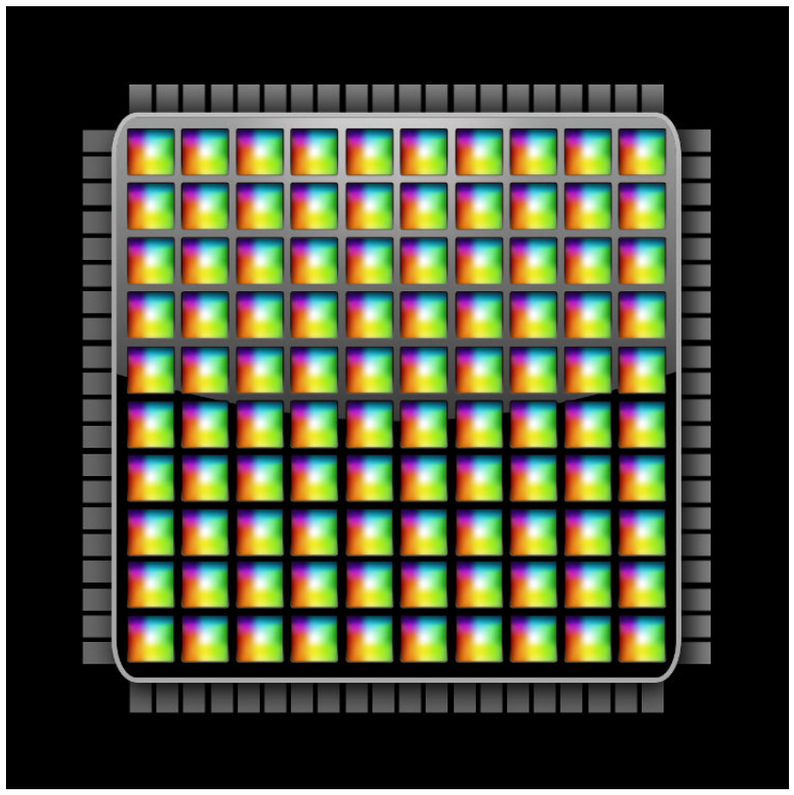
Picture cited from http://www.universalstudios.com/ for educational purpose only

Unity Makes Strength

- Chip Multi-Processors (CMPs)



You need more minions to work



You need more processor to compute

Picture cited from http://www.universalstudios.com/ and http://www.mit.edu for educational purpose only

Speedup

- One major motivation of using parallel processing:
  - Achieving a speedup

- For a given problem
  - Ideal speedup (using n processors) =

$$\frac{\text{execution time (using 1 processor)}}{\text{execution time (using n processor)}}$$

- Why can't we achieve n times speedup?

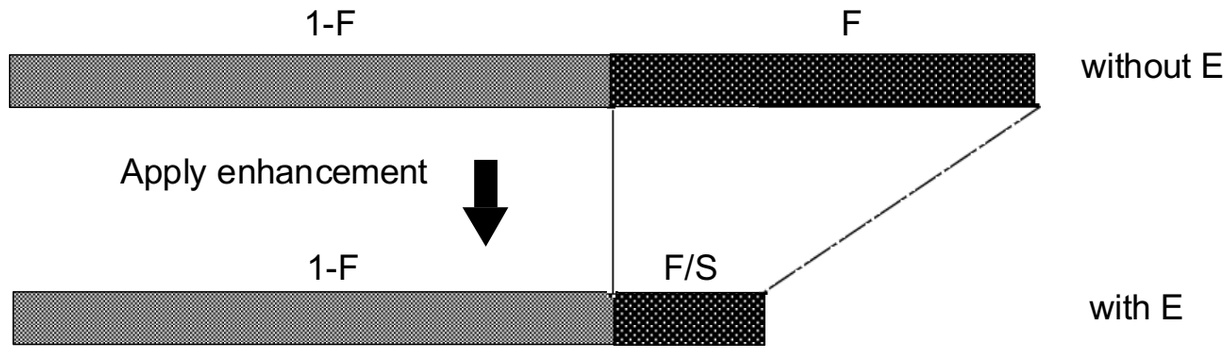
Picture cited from http://www.universalstudios.com/ for educational purpose only

Amdahl's Law

- **Calculate the speedup in latency of the execution of a task**

Speedup = 1 / (1 - F + F/S)

F: portion to be sped up
S: Speedup factor
E: Some enhancement



- **For example, a program with 30% of multiplication, if you can make the multiplier two times faster, then**

Speedup = 1 / (1 - 0.3 + 0.3/2) = 1.18

- **As a result, make the common case faster**

Imbalanced Workload

- Imbalance in work assignment limited speedup
  - Some students (processing elements) ran out work to do (went idle), while others were still working on their assigned task

- Improving the distribution of work improved speedup

Picture cited from http://www.littlecakehouseblog.co.uk/ for educational purpose only

Communication Costs

- The problem has a significant amount of communication compared to computation

- Communication costs can dominate a parallel computation, severely limiting speedup

Picture cited from http://www.zmescience.com for educational purpose only

Parallel Programming

- Parallel thinking
  - Decomposing work into pieces that can safely be performed in parallel
  - Assigning work to processors
  - Managing communication/synchronization among processors so that it doesn't limit speedup

- Abstraction/mechanisms for performing the above tasks
  - Writing codes in popular parallel programming languages

Parallel Hardware Architecture

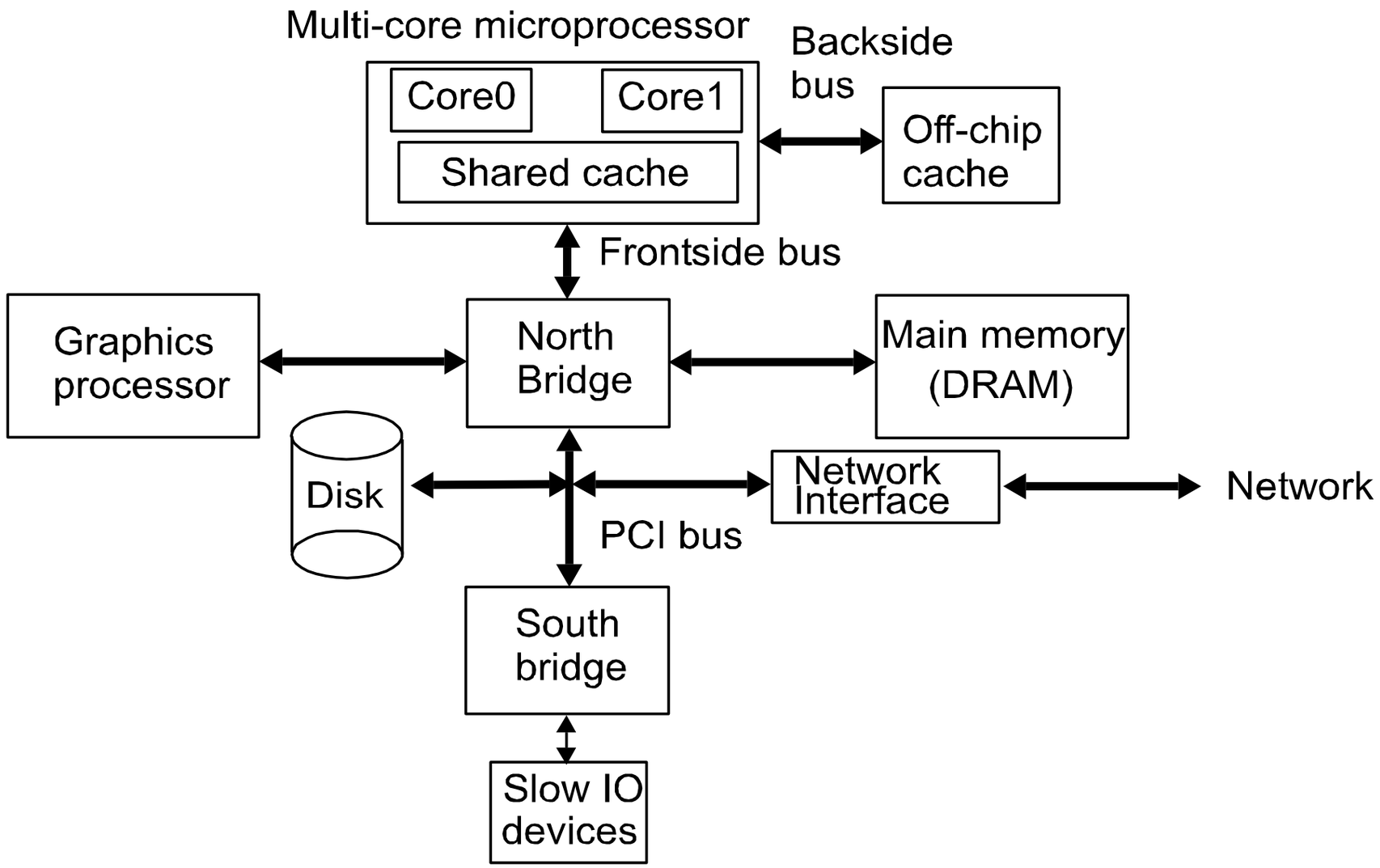
- **Why learning hardware architectures?**
  - To understand the underlining structures
  - To get an idea how a parallel program can be improved

- **Mechanisms to enable parallel programs**
  - Parallel processor architectures
  - On-chip interconnection networks
  - Cache coherence and synchronization
  - Heterogeneous architectures

Thinking About Efficiency

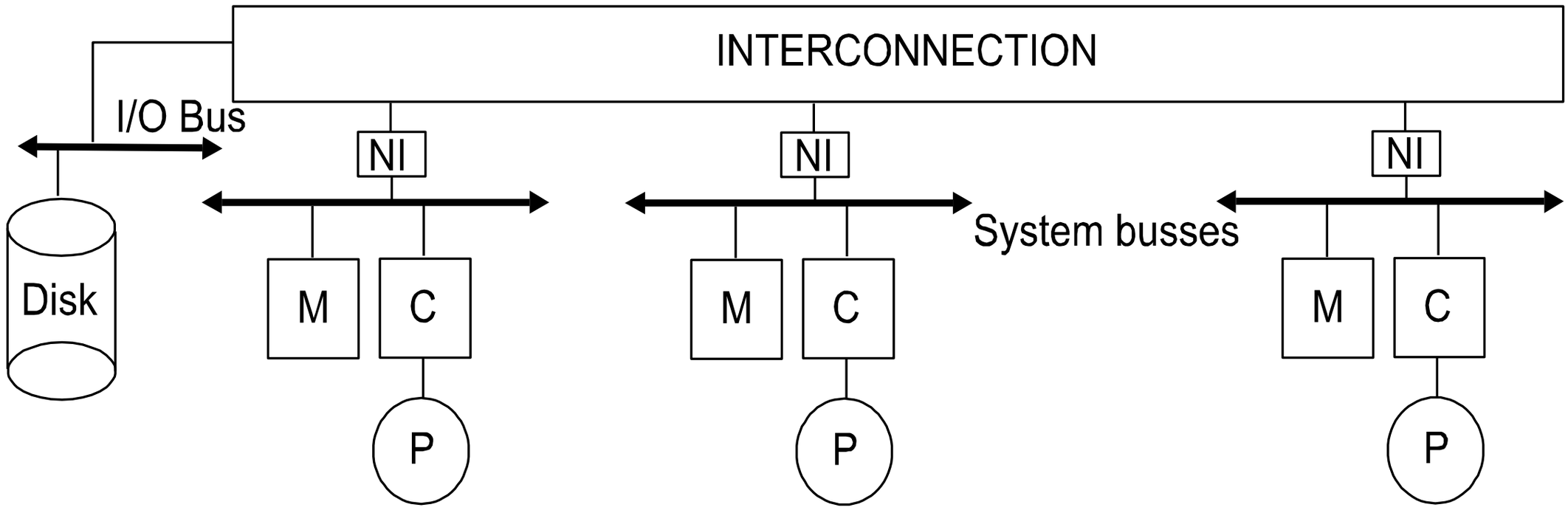
- Speed does not equal to efficiency
- Smartly utilize the available resources
  - Is 2x speedup on computer with 10 processors a good result?
- Choose the right capabilities to put in a system
  - Especially important in embedded systems
  - Hardware/software co-design

Modern PC Architecture



- North bridge
  - Connects to CPU, GPU, memory, networks, etc.
- South bridge
  - Connects to slow I/O devices: PCI, USB, IDE, etc.

Generic High-End Parallel System



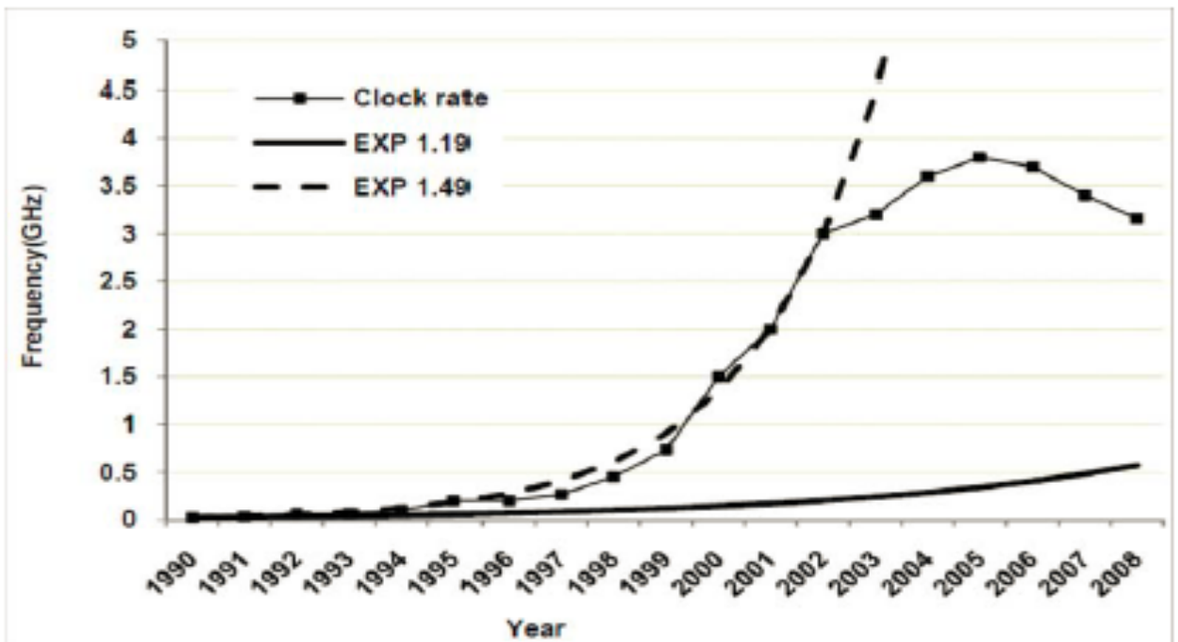
- **Main components**
  - **Processors (P), cache (C), memory (M), network interface (NI), I/O, and network interconnection**

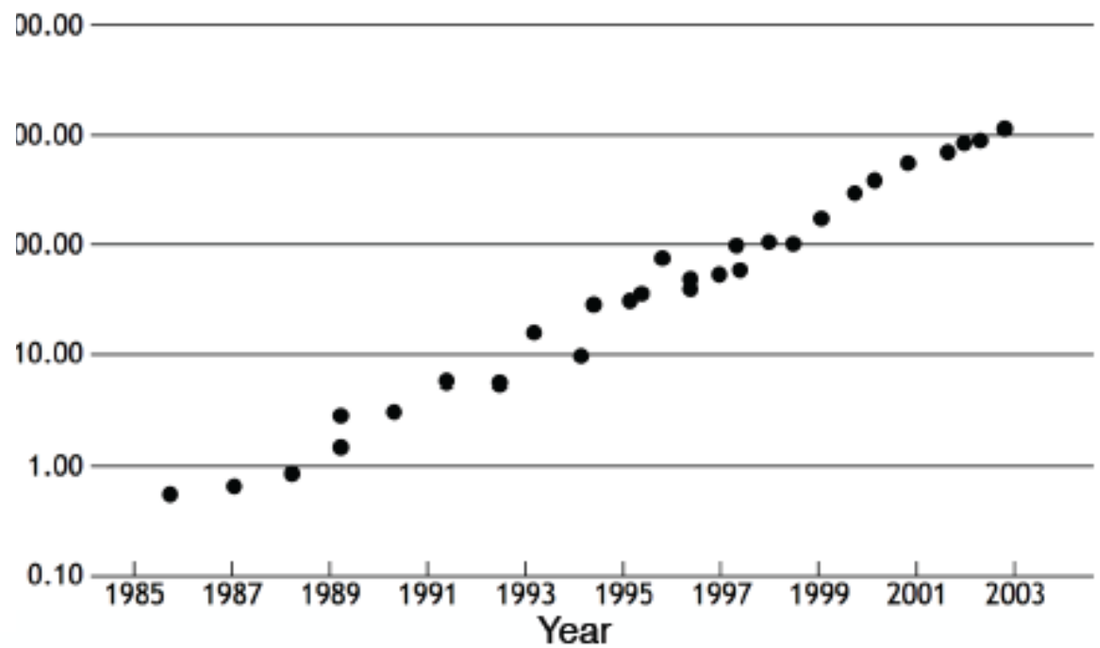
- **Function**
  - **Multiple processing elements (PE) cooperates to complete a task in parallel**
  - **PEs communicates via network interconnections**
  - **Network interconnections incur communication delays**

Why Parallel Processing?

- The answer 15 years ago
  - To achieve performance improvements that exceeded what CPU performance improvements could provide
  - If you just wait for several months, your application would run faster on a new CPU

- Parallelize your codes was often not worth the time
  - Software developers do nothing: CPU performance automatically doubles every 18 months!

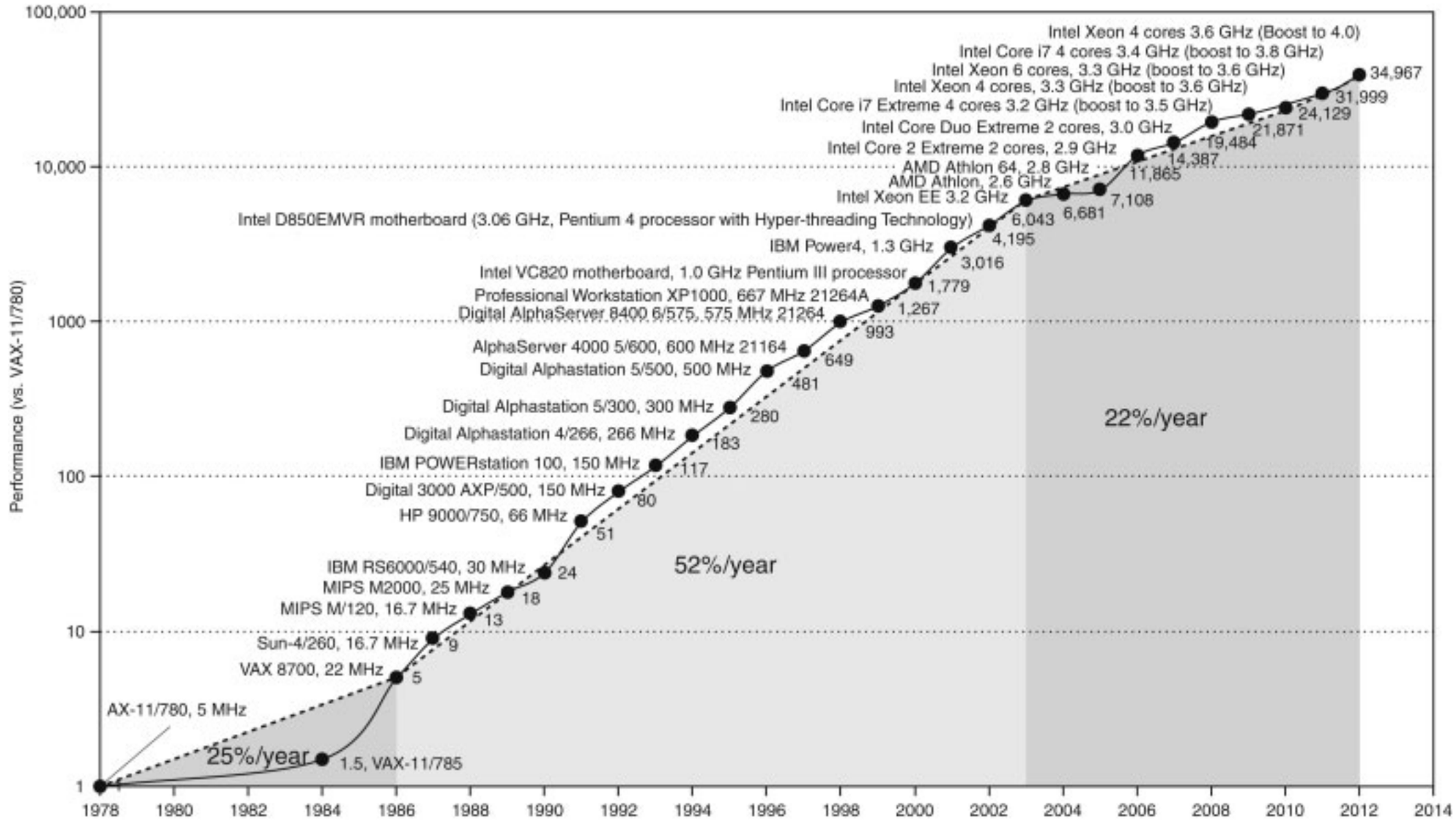




Picture cited from Computer Organization and Design, Fifth Edition by D. Patterson and J. Hennessy for educational purpose only

Performance Improvement

- Two significant reasons for processor performance improvement until 10 years ago
  - Increasing clock frequency
  - Exploring instruction-level parallelism (superscalar execution)

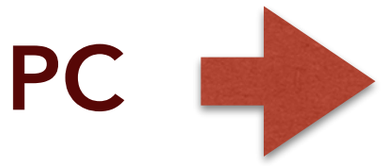


Picture cited from Computer Organization and Design, Fifth Edition by D. Patterson and J. Hennessy for educational purpose only

What Is a Program?

- A sequence of instructions
- The function of a processor is to execute programs
- Execute instructions referenced by the program counter (PC)
- The program counter moves from instruction to instruction, and execute it.

PC



```
lw      $t0, 0($gp)     # fetch i
srl     $t0, $t0, 1     # i/2
addi    $t1, $gp, 28    # &A[0]
sll     $t0, $t0, 2     # turn i/2 into a byte offset (*4)
add     $t1, $t1, $t0   # &A[i/2]
lw      $t1, 0($t1)     # fetch A[i/2]
addi    $t1, $t1, 1     # A[i/2] + 1
lw      $t0, 0($gp)     # fetch i
sll     $t0, $t0, 2     # turn i into a byte offset
addi    $t2, $gp, 28    # &A[0]
add     $t2, $t2, $t0   # &A[i]
lw      $t0, 0($gp)     # fetch i
addi    $t0, $t0, 1     # i+1
sll     $t0, $t0, 2     # turn i+1 into a byte offset
addi    $t1, $gp, 28    # &A[0]
add     $t1, $t1, $t0   # &A[i+1]
addi    $t2, $zero, -1  # -1
sw      $t2, 0($t1)     # A[i+1] = -1
```

Instruction Level Parallelism (ILP)

- **Processors did in fact leverage parallel execution to make programs run faster**
  - Invisible to the programmer

- **Idea**
  - Instructions must be executed in program order.
  - Independent instructions can be executed simultaneously without impacting correctness

- **Superscalar execution**
  - Processor dynamically finds independent instructions in the sequence
  - Execution in parallel (using different execution units)

```
mul    $t1, $t0, $t0
mul    $t1, $t1, $t1
 st    $t1, mem[$t2]
```

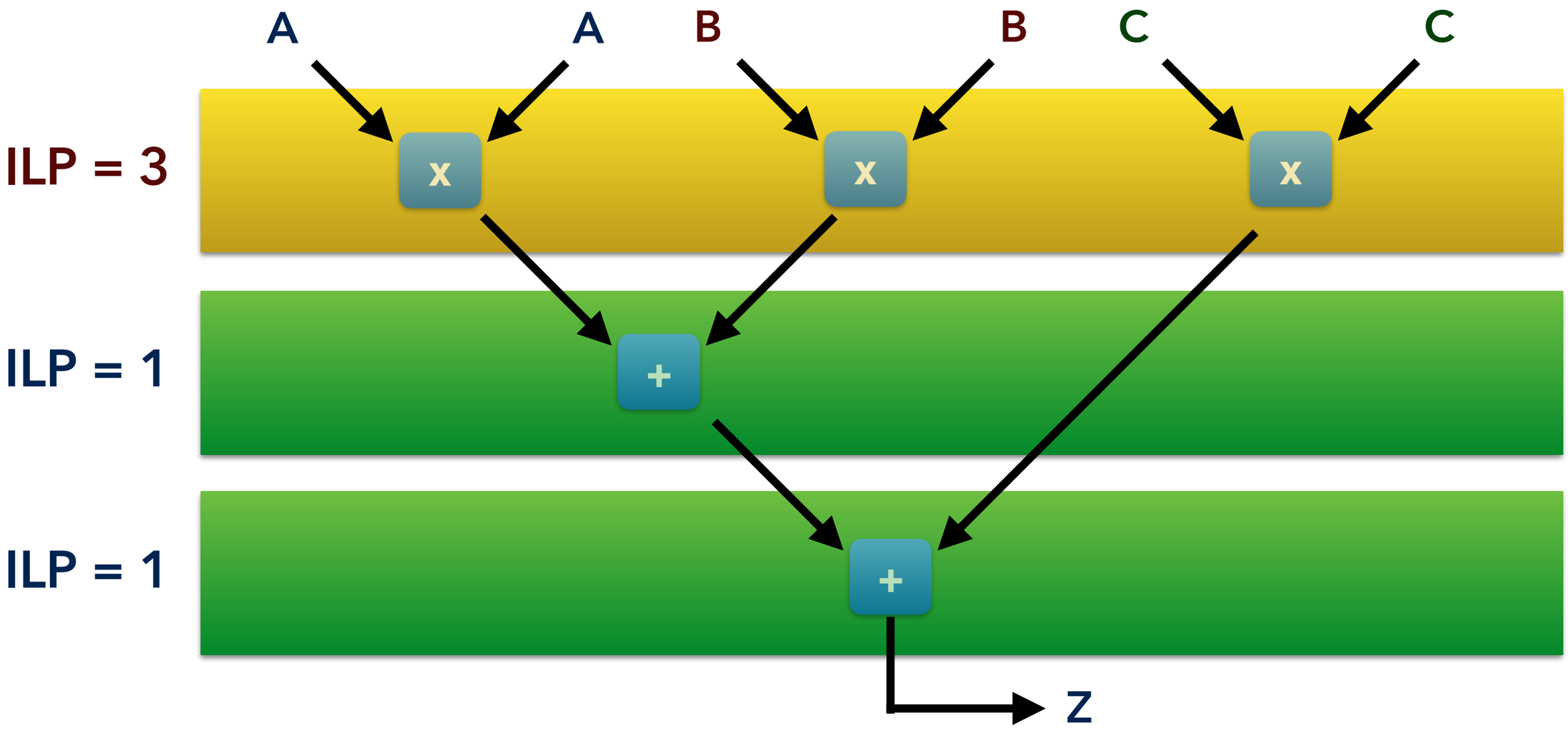
Dependent instructions

```
add    $t1, $t1, $t0
add    $t3, $t3, $t2
...
```

Independent instructions

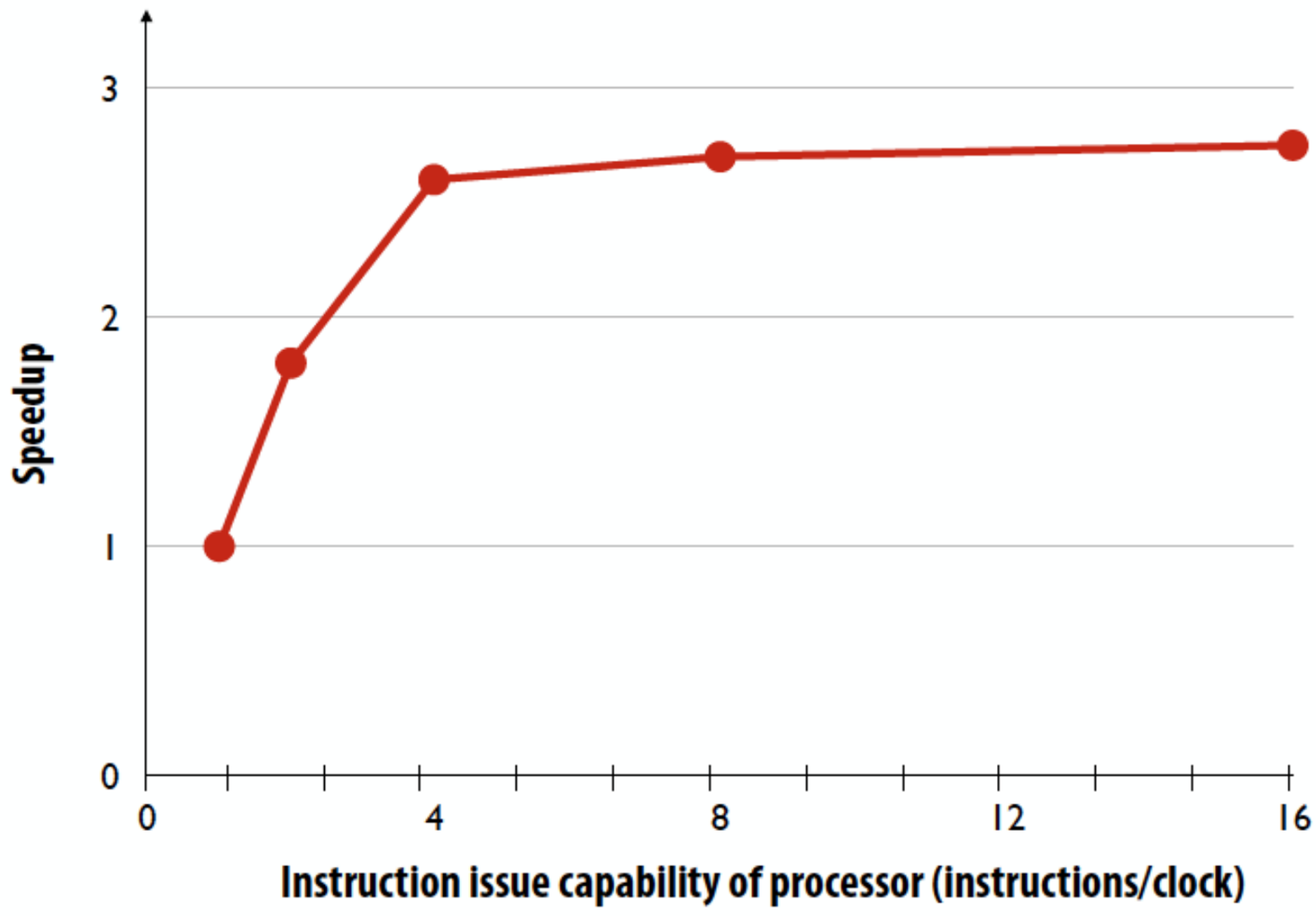
ILP Example

$$Z = A \times A + B \times B + C \times C$$

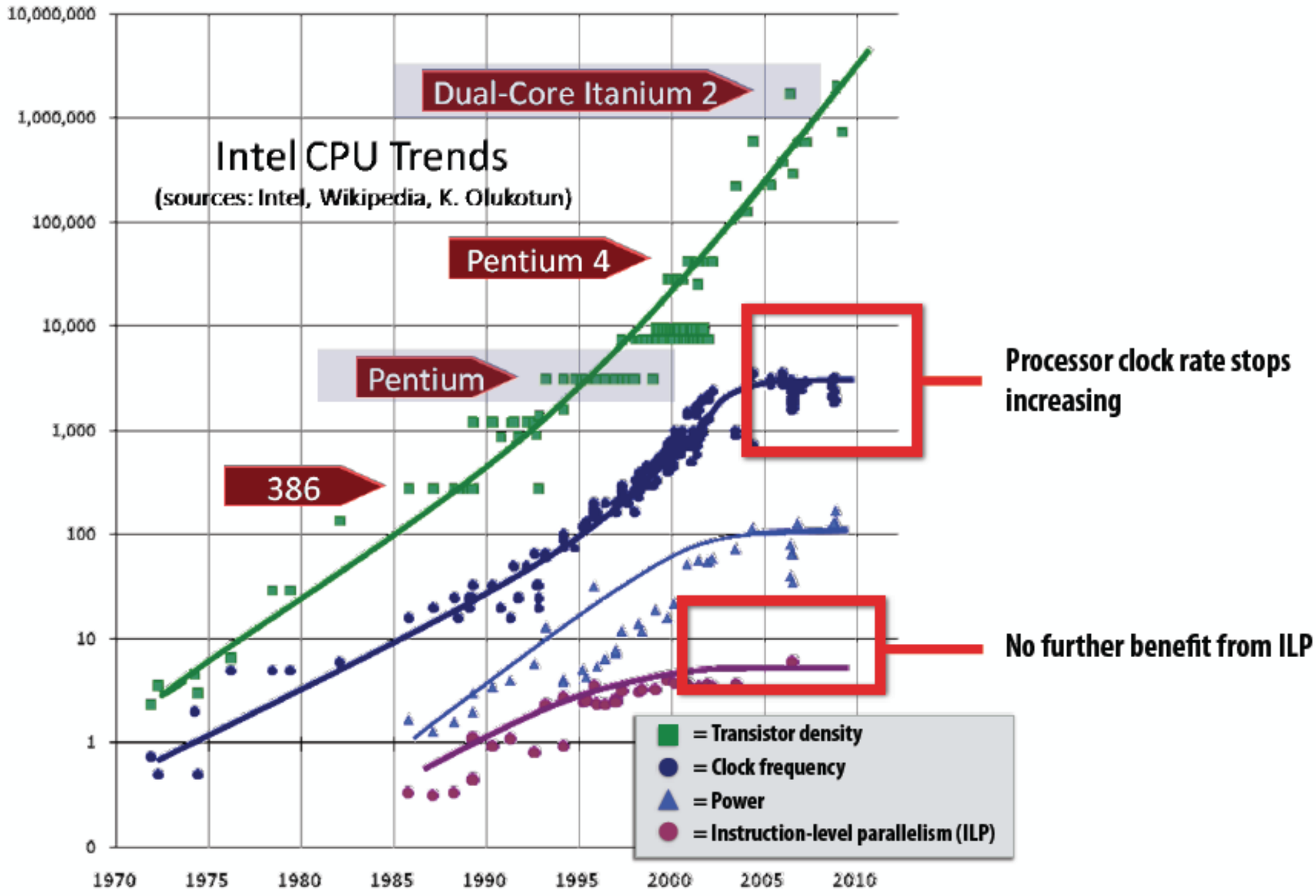


Diminishing Effectiveness of ILP

- Most available ILP exploited by a processor capable of issuing four instructions per clock
- Little performance gains for more instructions per clock



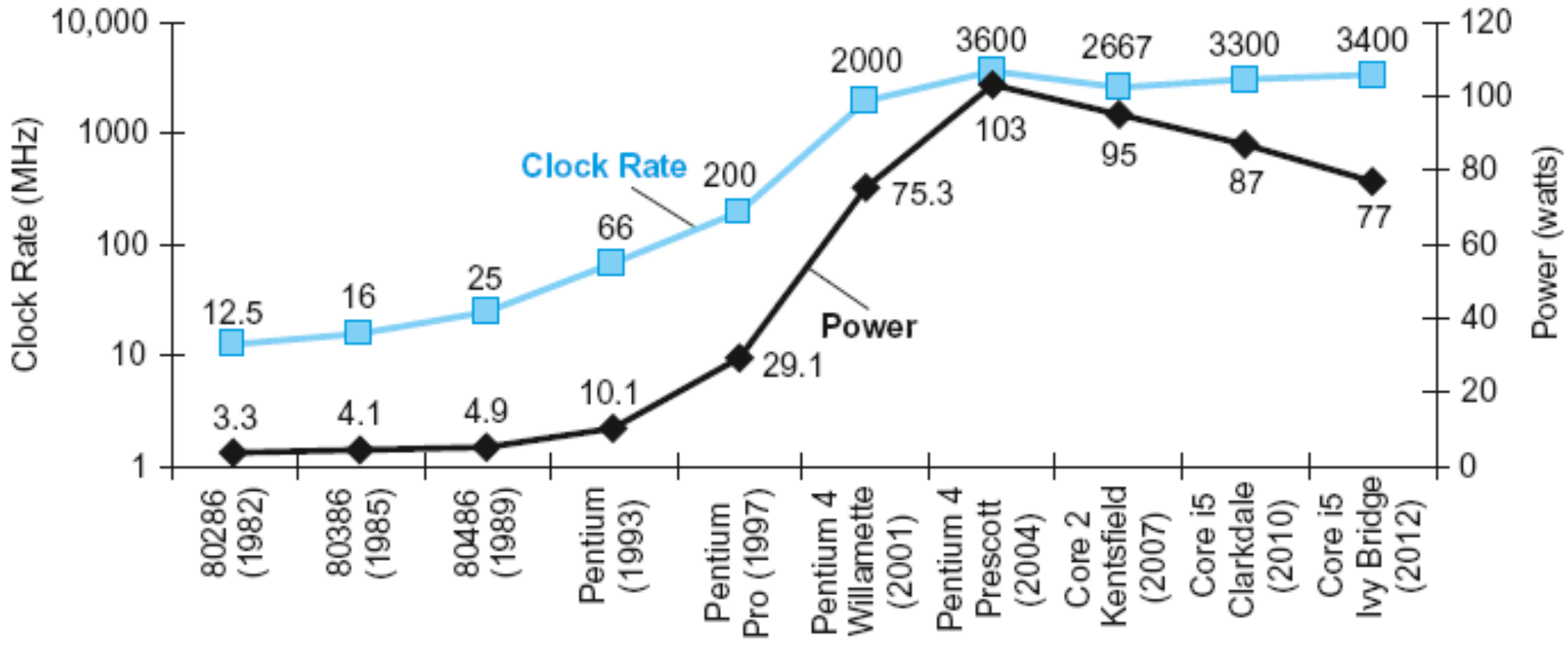
End of Frequency Scaling



Picture credit: “The free lunch is over” by Herb Sutter, Dr. Dobbs 2005 for educational purpose only

The Power Wall

- Harder and harder to reduce voltage further
- Very difficult to increase the frequency further



Power = α x Capacitive load x Voltage^2 x Frequency

30x **5V -> 1V** **1,000X**

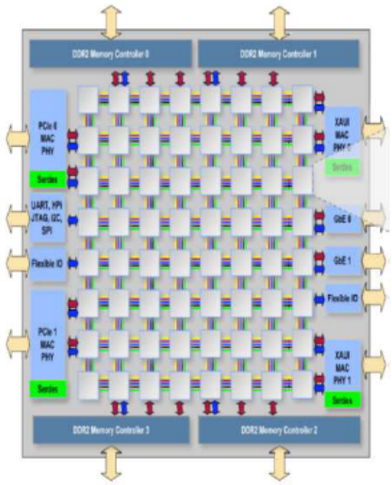
Picture cited from Computer Organization and Design, Fifth Edition by D. Patterson and J. Hennessy for educational purpose only

Why Parallelism?

- The rate of single-instruction stream performance scaling has decreased
  - Frequency scaling limited by power and fabrication technology
  - ILP scaling has reached its limitation
- Architects now building faster processors by adding more processing units to run in parallel
- Software programs must be written to be parallel to gain performance
  - Processor speed does not increase automatically as before!

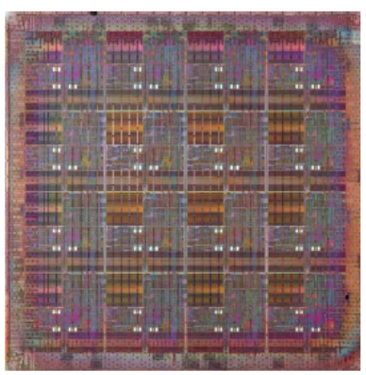
The Era of Chip Multiprocessors

- **Several chip multiprocessors have been developed by academia and industry**

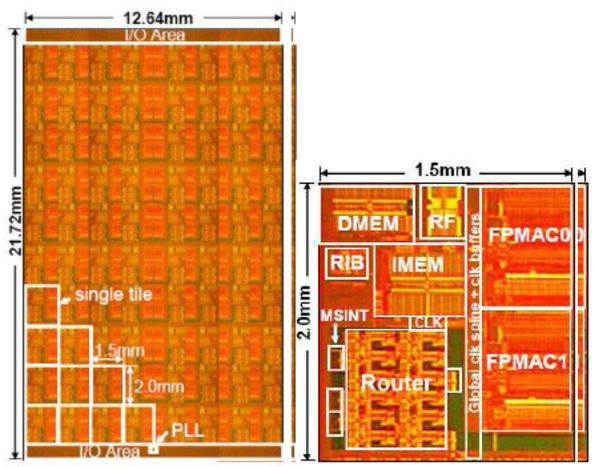


Tilera TILE 64
(90nm, 1GHz)
64-core chip
Five 2D mesh networks

Sun Niagara2
(65nm, 1.4GHz)
8-core chip
Crossbar network



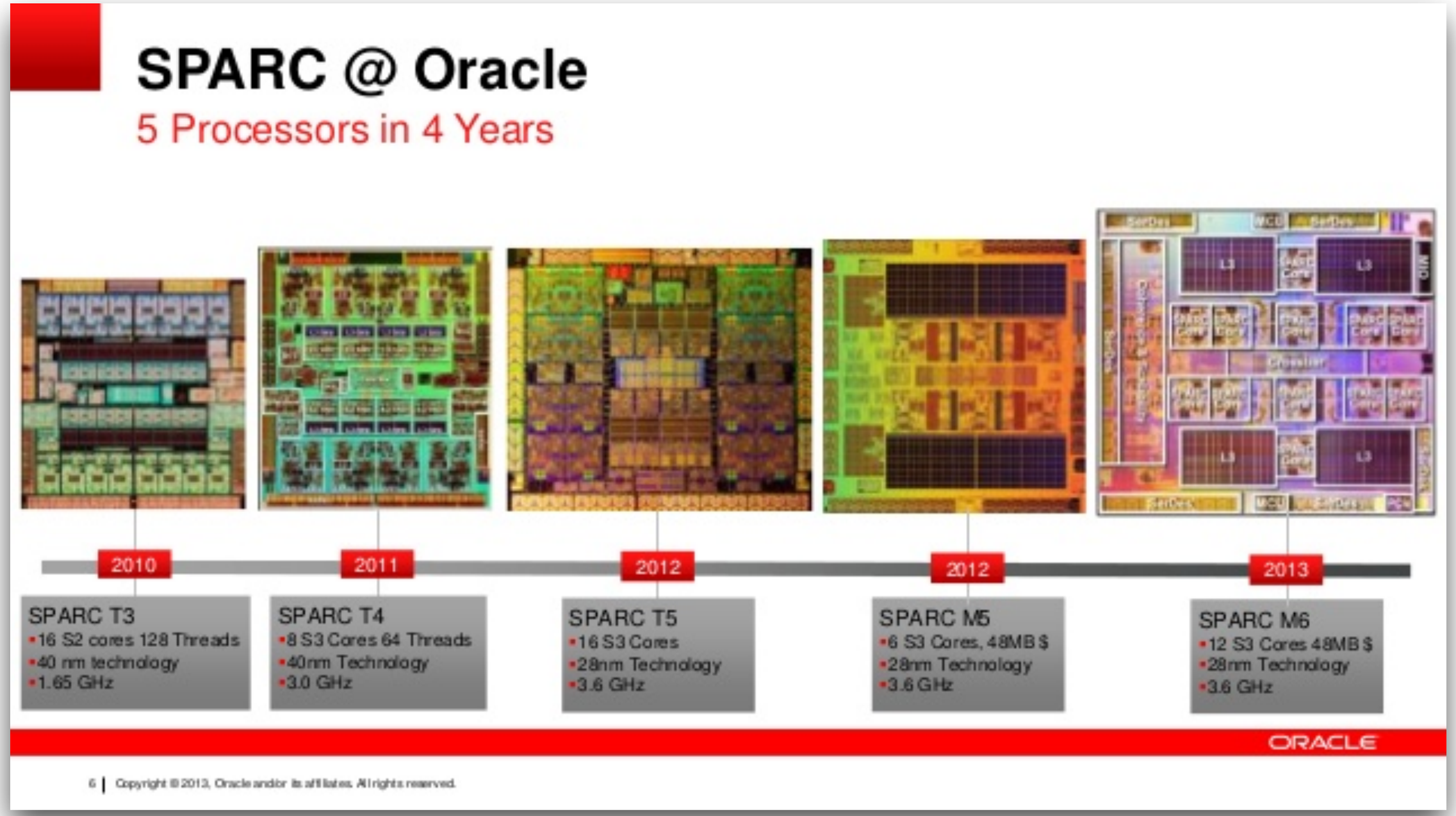
MIT Raw
(0.18um, 300MHz)
16-core chip
Four 4x4 mesh networks



Intel Teraflop
(65nm, 5GHz)
80-core chip
8x10 mesh network

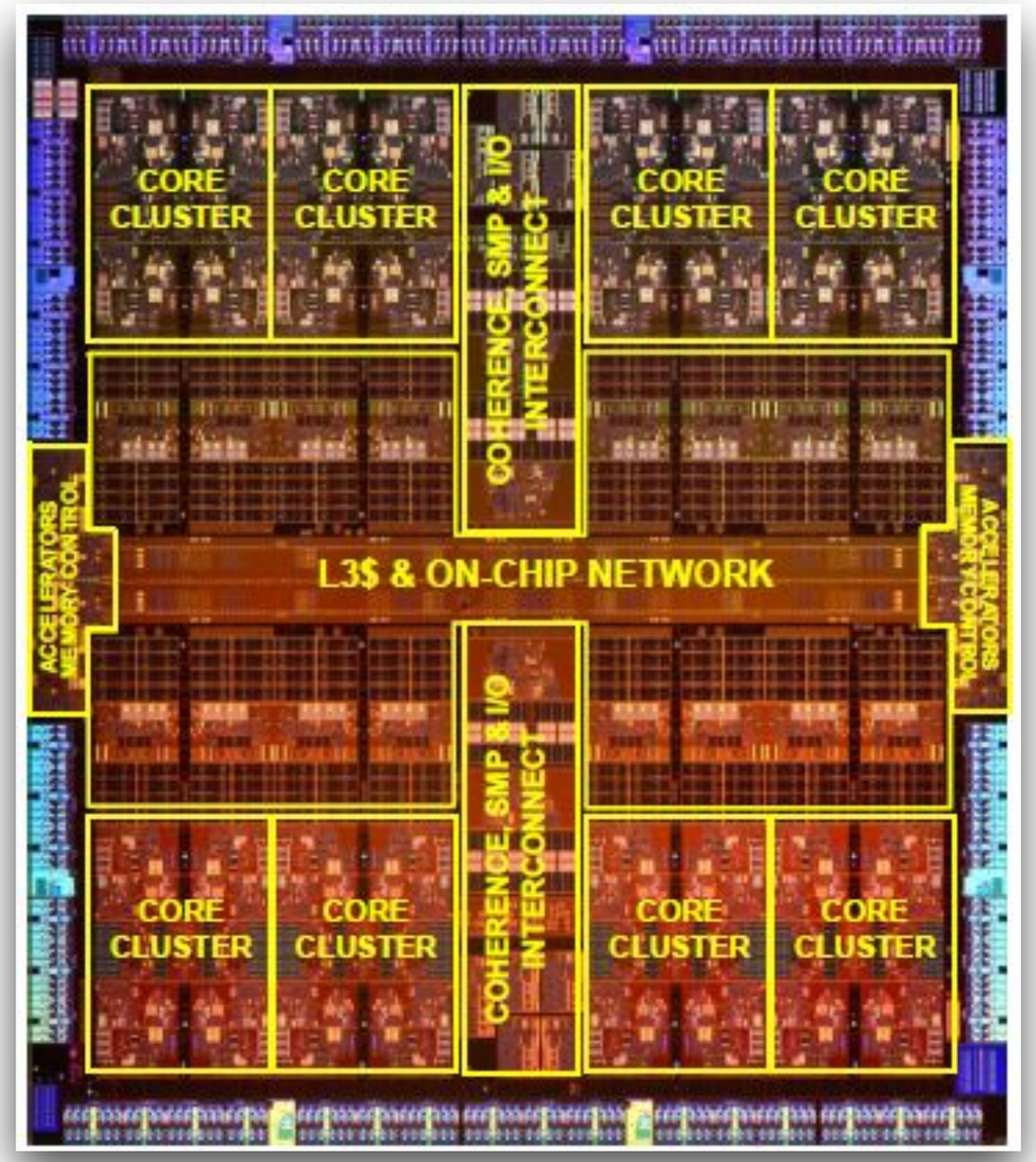
Picture cited from www.oracle.com, www.mit.edu, www.intel.com, and www.tilera.com for educational purpose only

Oracle's Processor Family



Picture cited from www.oracle.com for educational purpose only

Oracle's M7 Processor

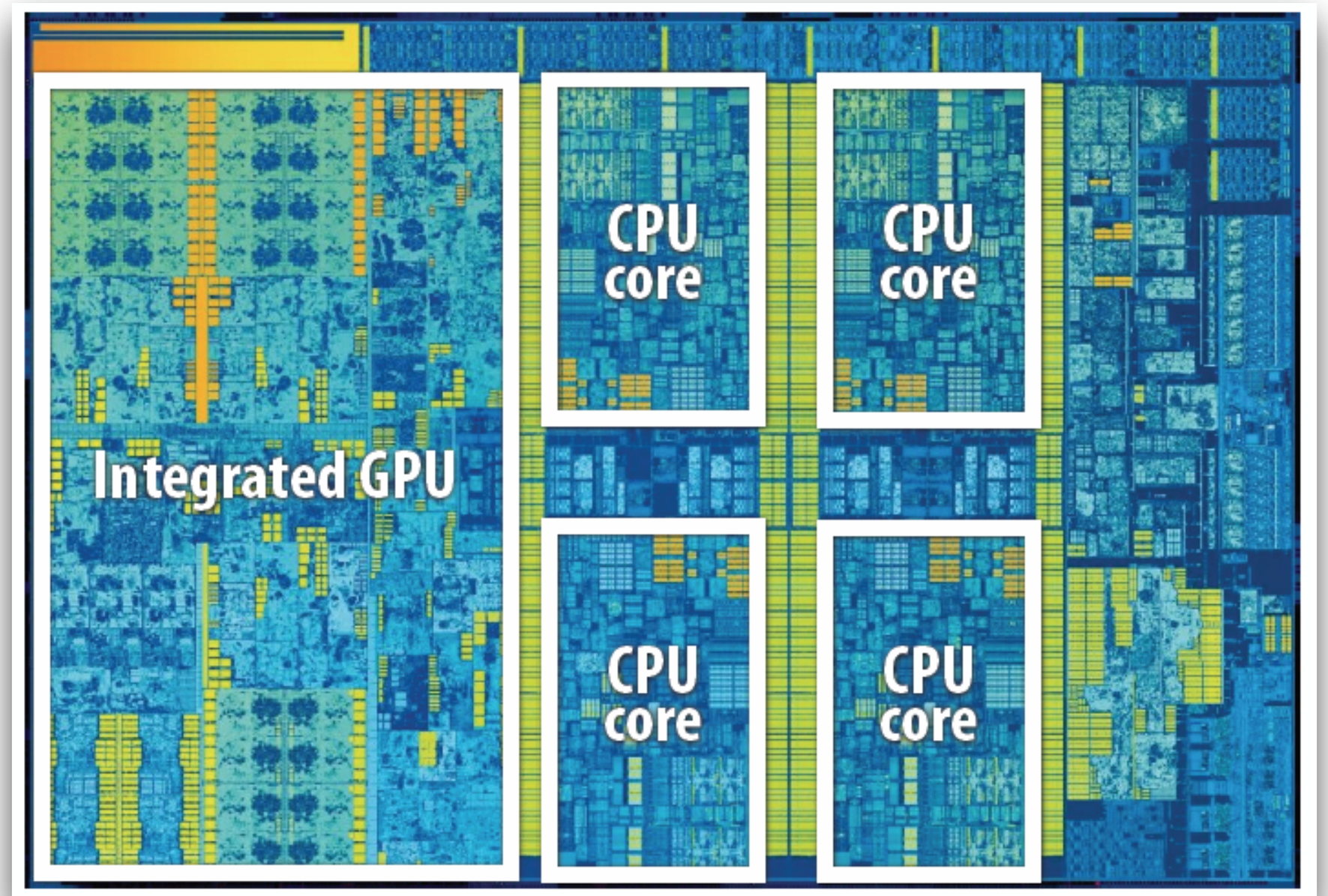


- Server grade processor
- 32 Sparc S4 cores
- 64MB L3 cache
- Application acceleration
- 20nm technology

Picture cited from www.oracle.com for educational purpose only

Intel Skylake (2015)

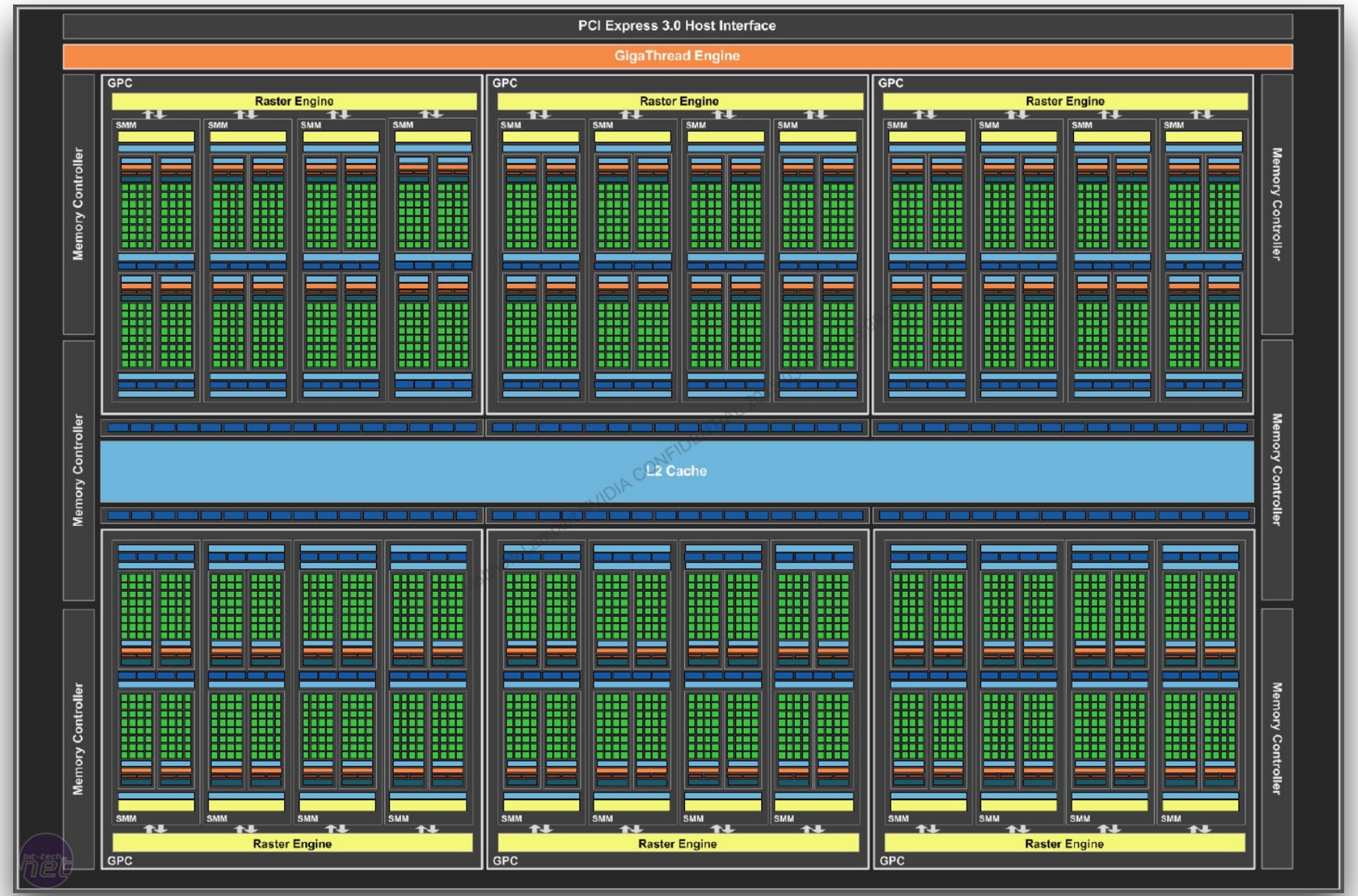
- 6th generation of core i7
- Quad-core CPU plus multi-core GPU integrated on the same chip



Picture cited from www.intel.com for educational purpose only

NVIDIA Titan X (Maxwell)

- 3,072 stream processors
- 12GB GDDR5
- 1,000MHz



Picture cited from www.nvidia.com for educational purpose only

Mobile Parallel Processing

- iPhone 6S A9 processor
- Dual-core CPU + GPU + image processor and more on one chip

APL0898
Samsung
96mm²

APL1022
TSMC
104.5mm²

Picture cited from www.apple.com, www.tsmc.com, and www.samsung.com for educational purpose only

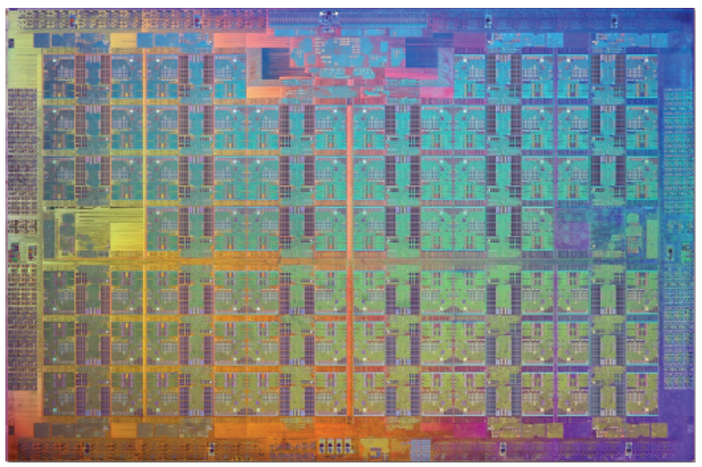
Supercomputing

- Cluster of thousands of CPUs and GPUs
- Oak Ridge National Laboratory: Titan (#2 supercomputer in the world)
  - 18,688 x 16 core AMD CPUs + 18,688 NVIDIA K20X GPUs

Picture cited from https://www.olcf.ornl.gov/titan/ for educational purpose only

Intel Xeon Phi

- 72 simple x86 cores (1.1 GHz, developed from Intel Atom)
- Four-way multi-threading (four threads per core)
- 16-wide vector instructions
- Accelerator for supercomputers



Summary

- Performance of a single processor is improving slowly
- To improve performance of computers, you need
  - Increase the number of processing elements
  - Write parallel programs that utilize parallel computing resources
  - Improve the efficiency of communication and synchronization among the processors in a system

Thank you for your attention!

e © 2013

Picture was taken by Chun-Yi Lee himself, who was an amateur photographer