250P: Computer Systems Architecture

Lecture 1: Introduction

Anton Burtsev
January, 2019

Class details

- Graduate
  - 55 students
- Instructor: Anton Burtsev
- Meeting time: 3:30pm-4:50pm (Mon/Wed)
  - Discussions: 8:00pm-8:50pm (Mon)
- 2 TAs
- Web page
  - https://www.ics.uci.edu/~aburtsev/250P/

More details

- 6-7 small homeworks
- Midterm
- Final
- Grades are curved
  - Homework: 50%, midterm exam: 25%, final exam: 25% of your grade.
  - You can submit late homework 3 days after the deadline for 60% of your grade

This course

- Book: Hennessy and Patterson's
  - Computer Architecture, A Quantitative Approach, 6th Edition
- Topics
  - Measuring performance/cost/power
  - Instruction level parallelism, dynamic and static
  - Memory hierarchy
  - Multiprocessors
  - Storage systems and networks

Course organization

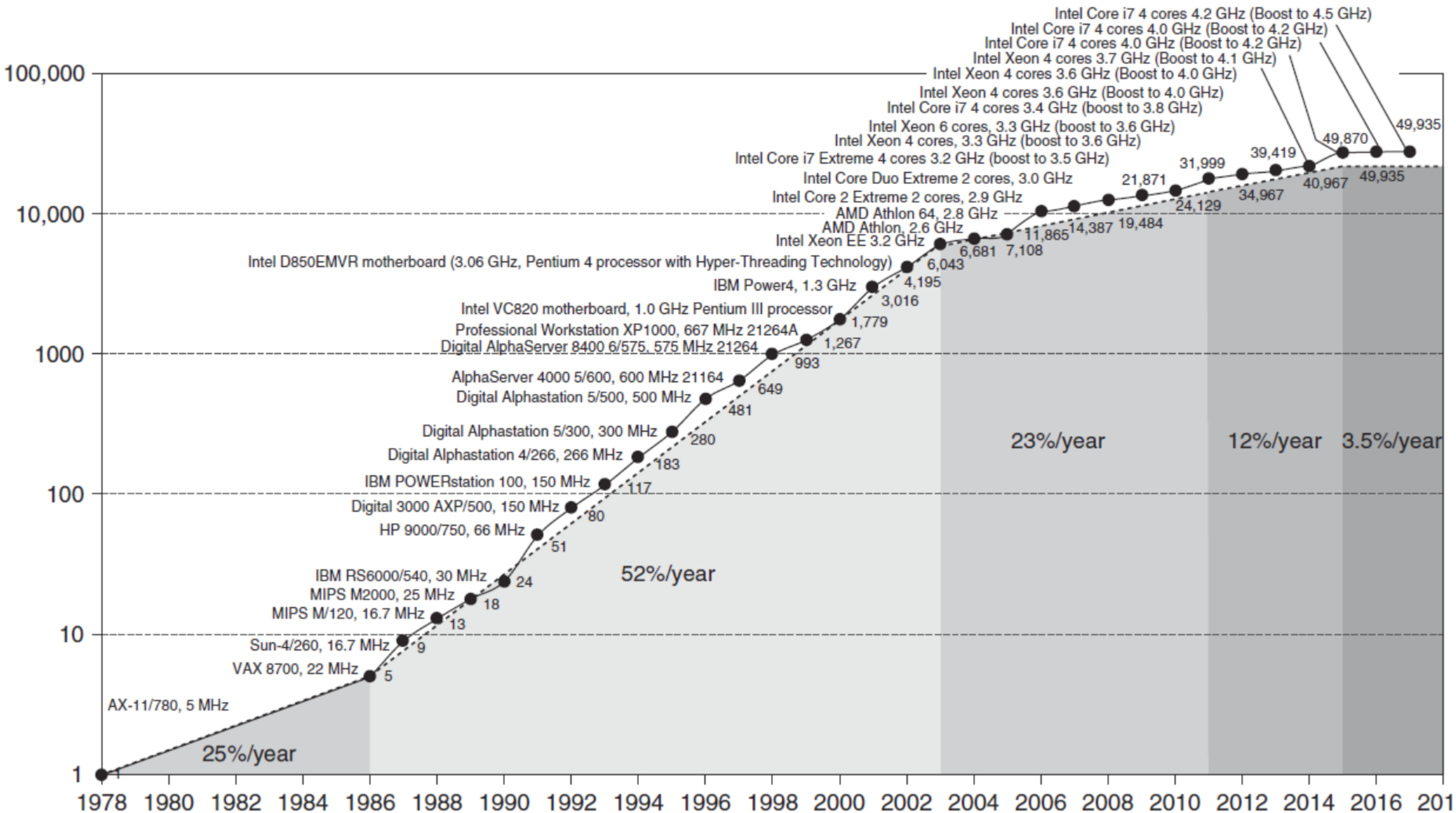
- Lectures
  - High level concepts and abstractions
- Reading
  - Hennessy and Patterson
  - Bits of additional notes
- Homeworks

Computer technology

- Performance improvements:
  - Improvements in semiconductor technology
    - Feature size, clock speed
  - Improvements in computer architectures
    - Enabled by high-level language compilers, general operating systems
    - Lead to RISC architectures

- Together have enabled:
  - Lightweight computers
  - Productivity-based managed/interpreted programming languages

Single processor performance

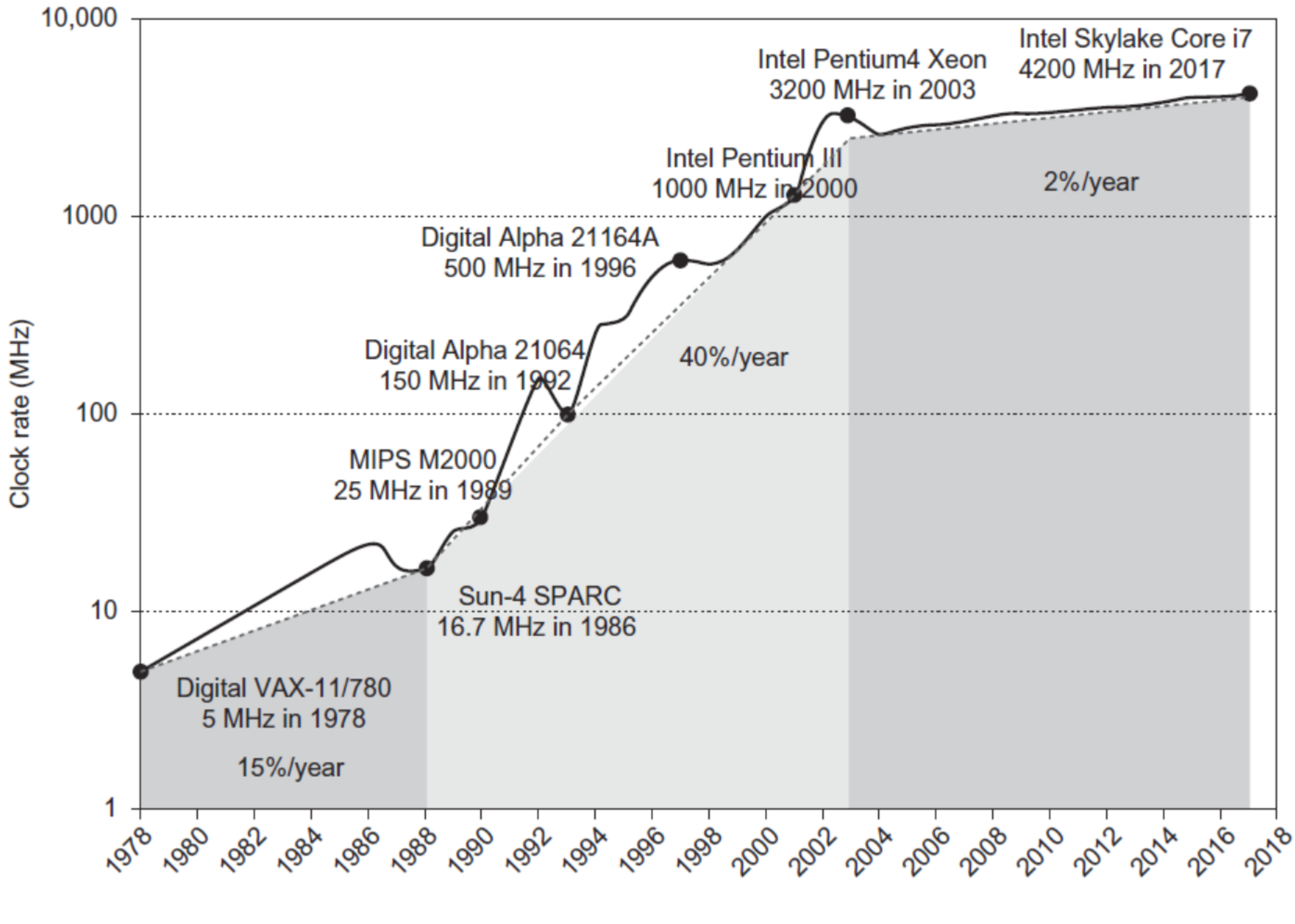


Points to note

- The 52% growth per year is because of faster clock speeds and architectural innovations (led to 25x higher speed)
- Clock speed increases have dropped to 1% per year in recent years
- The 22% growth includes the parallelization from multiple cores

- End of Dennard scaling
- End of Moore's Law: transistors on a chip double every 18-24 months

Clock speed growth



Current trends in architecture

- Cannot continue to leverage Instruction-Level parallelism (ILP)
  - Single processor performance improvement ended in 2003
  - End of Dennard scaling
  - End of Moore's Law

Dennard scaling

- In 1974 Robert Dennard observed that
  - *power density was constant for a given area of silicon even as you increased the number of transistors because of smaller dimensions of each transistor*
- I.e., transistors could go faster but use less power

Moore’s Law

- In 1965 Gordon Moore predicted that
  - *the number of transistors per chip would double every year*
  - *which was amended in 1975 to every two years*
- That prediction lasted for about 50 years
  - But no longer holds

- Example:
  - In 2010 the most recent Intel microprocessor had 1,170,000,000 transistors
  - If Moore’s Law had continued
    - In 2016 we would have 18,720,000,000 transistors
  - Instead, the equivalent Intel microprocessor has just 1,750,000,000 transistors
    - Off by a factor of 10 from what Moore’s Law predicts

What can help performance?

- Note: no increase in clock speed
  - In a clock cycle, can do more work -- since transistors are faster, transistors are more energy-efficient, and there's more of them
- Better architectures:
  - finding more parallelism in one thread, better branch prediction, better cache policies, better memory organizations, more thread-level parallelism, etc.

Current trends in architecture

- Cannot continue to leverage Instruction-Level parallelism (ILP)
  - Single processor performance improvement ended in 2003

  - End of Dennard scaling
  - End of Moore's Law

- New models for performance:
  - Data-level parallelism (DLP)
  - Thread-level parallelism (TLP)
  - Request-level parallelism (RLP)

- These require explicit restructuring of the application

Parallelism

- Classes of parallelism in applications:
  - Data-Level Parallelism (DLP)
  - Task-Level Parallelism (TLP)

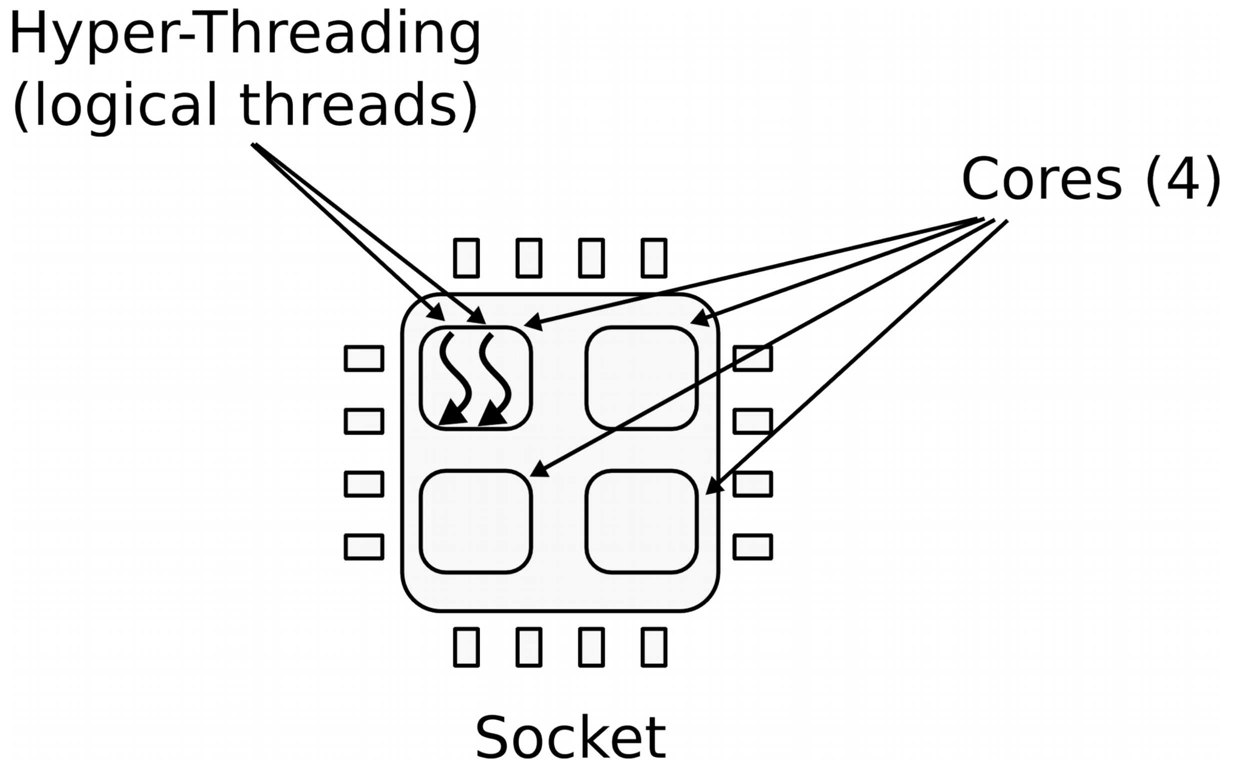
- Classes of architectural parallelism:
  - Instruction-Level Parallelism (ILP)
  - Vector architectures/Graphic Processor Units (GPUs)
  - Thread-Level Parallelism
  - Request-Level Parallelism

Flynn's Taxonomy

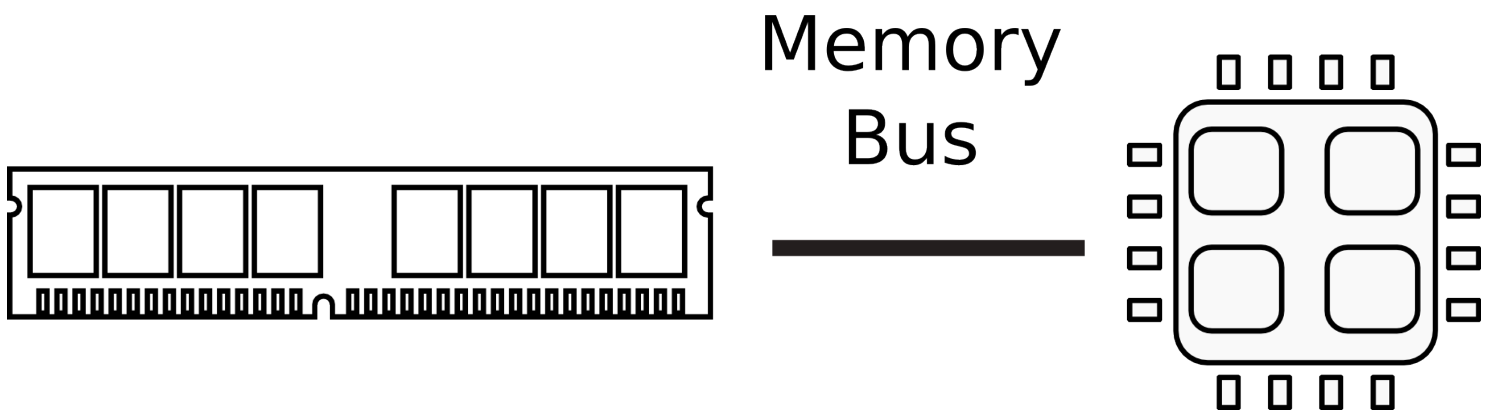
- Single instruction stream, single data stream (SISD)
- Single instruction stream, multiple data streams (SIMD)
  - Vector architectures
  - Multimedia extensions
  - Graphics processor units
- Multiple instruction streams, single data stream (MISD)
  - No commercial implementation
- Multiple instruction streams, multiple data streams (MIMD)
  - Tightly-coupled MIMD
  - Loosely-coupled MIMD

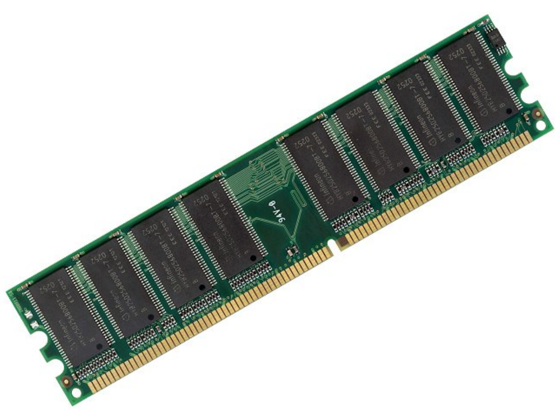
CPU

- 1 CPU socket
  - 4 cores
  - 2 logical (HT) threads each



Memory

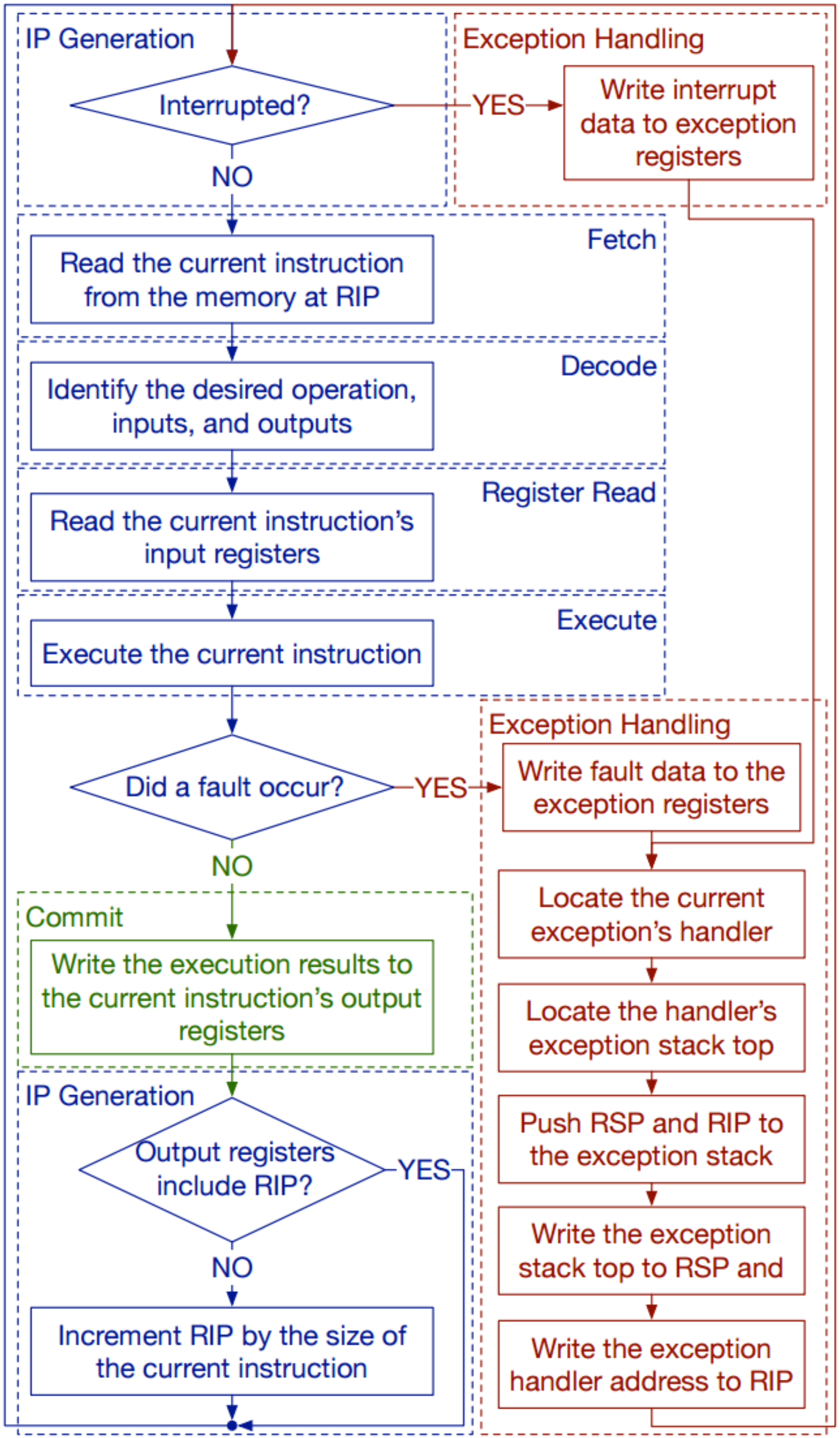




Memory abstraction

WRITE(*addr*, *value*) $\rightarrow \varnothing$ Store *value* in the storage cell identified by *addr*.
READ(*addr*) $\rightarrow$ *value* Return the *value* argument to the most recent WRITE call referencing *addr*.

What does CPU do internally?

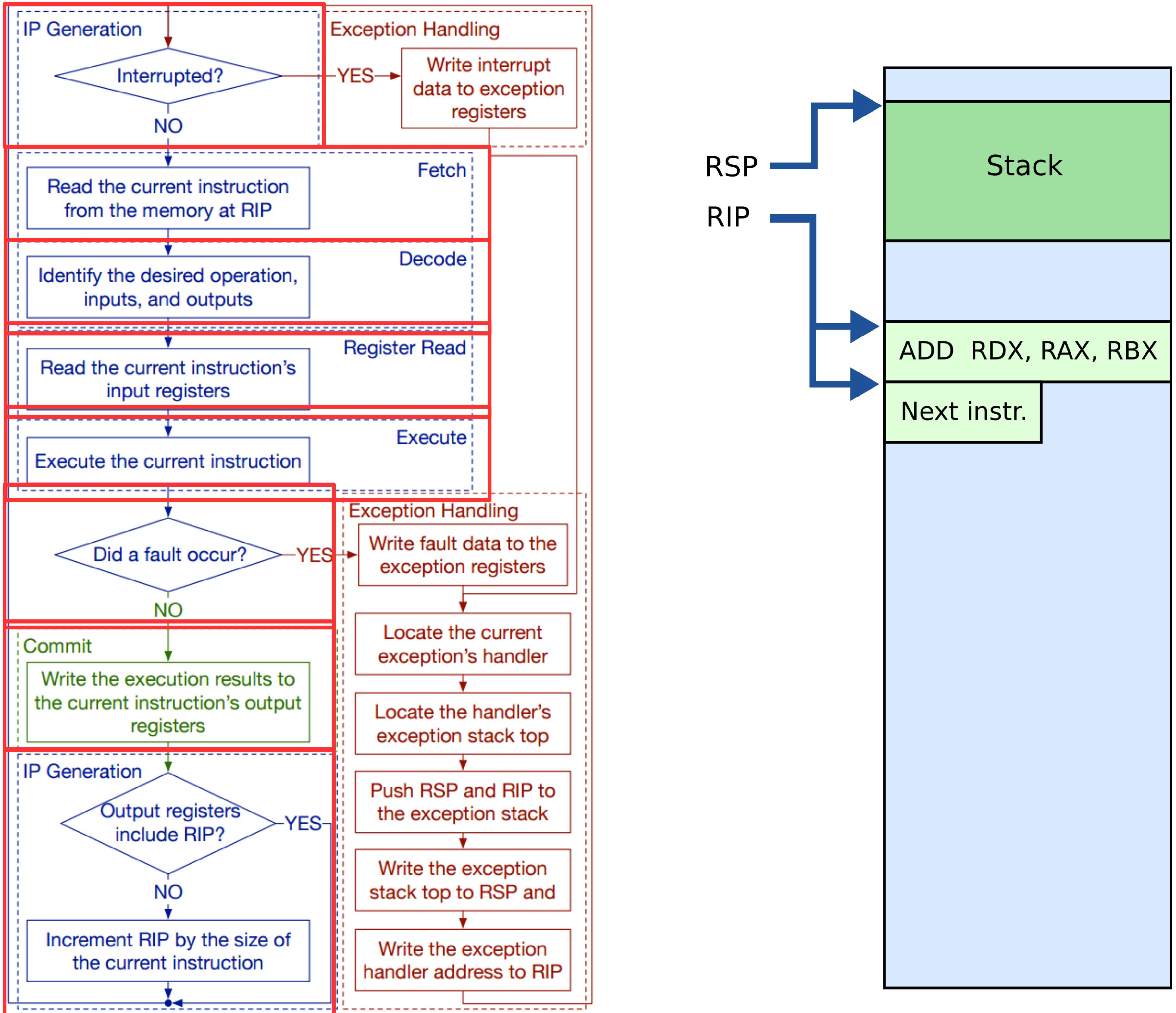


CPU execution loop

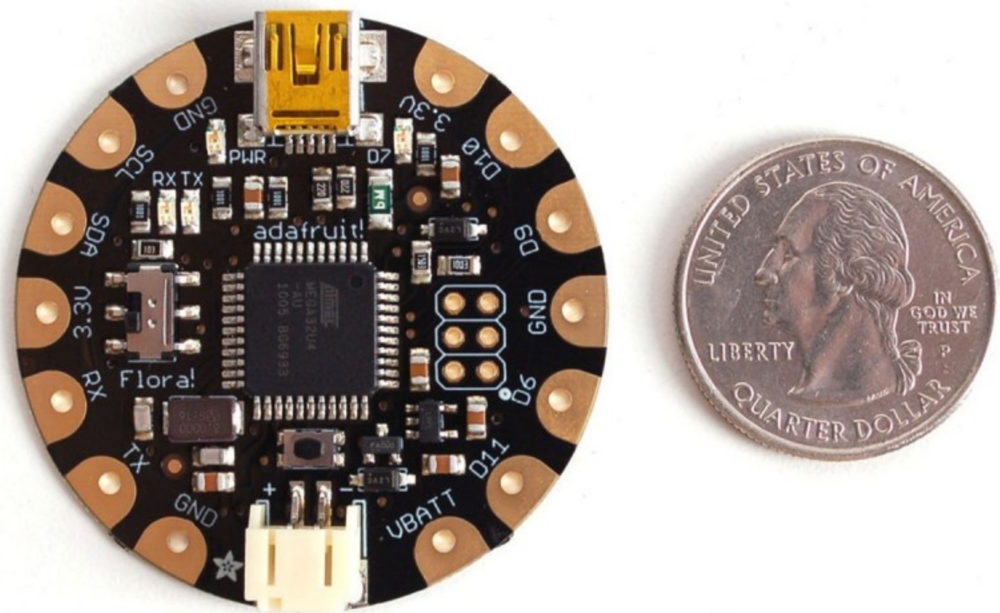
- CPU repeatedly reads instructions from memory
- Executes them
- Example

```
ADD EDX, EAX, EBX

// EDX = EAX + EBX
```

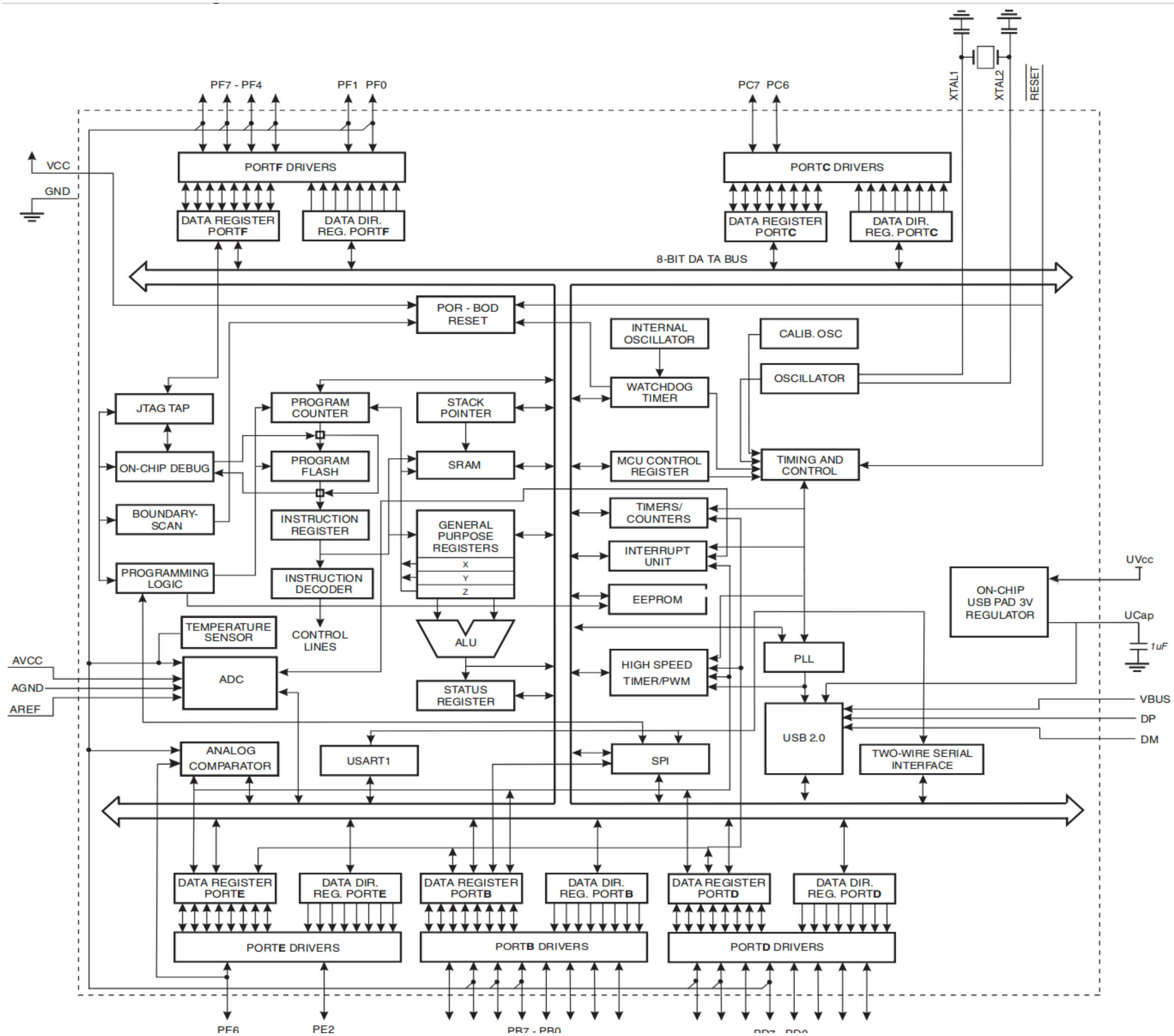

IP Generation
Interrupted?
YES
NO
Exception Handling
Write interrupt data to exception registers
Fetch
Read the current instruction from the memory at RIP
Decode
Identify the desired operation, inputs, and outputs
Register Read
Read the current instruction's input registers
Execute
Execute the current instruction
Did a fault occur?
YES
NO
Exception Handling
Write fault data to the exception registers
Locate the current exception's handler
Locate the handler's exception stack top
Push RSP and RIP to the exception stack
Write the exception stack top to RSP and
Write the exception handler address to RIP
Commit
Write the execution results to the current instruction's output registers
IP Generation
Output registers include RIP?
YES
NO
Increment RIP by the size of the current instruction
RSP
RIP
Stack
ADD RDX, RAX, RBX
Next instr.

Embedded/IoT



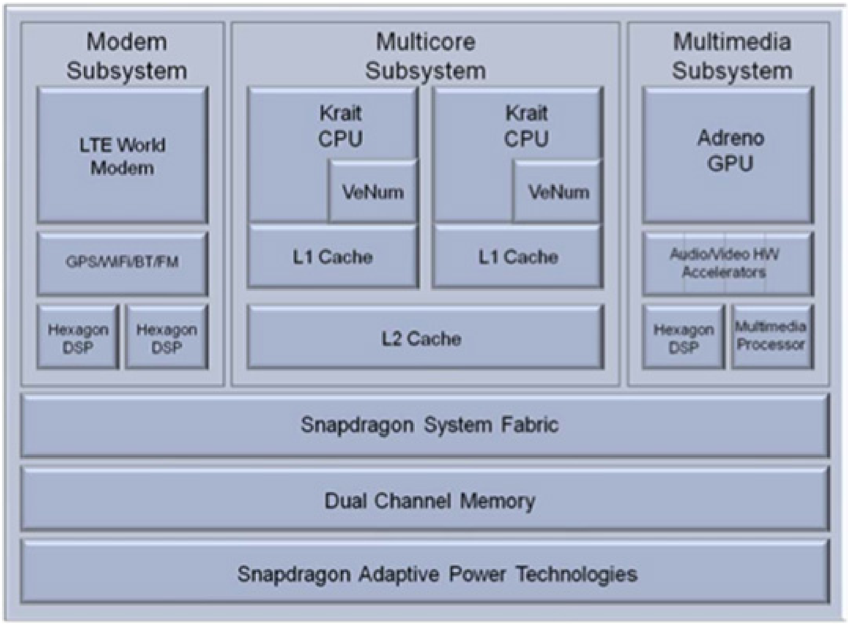
- Arduino Flora
- Atmega32u4 8bit RISC
- 32K flash
- 2.5K RAM
- running at 16MHz

ATmega32U4 8-bit RISC microcontroller



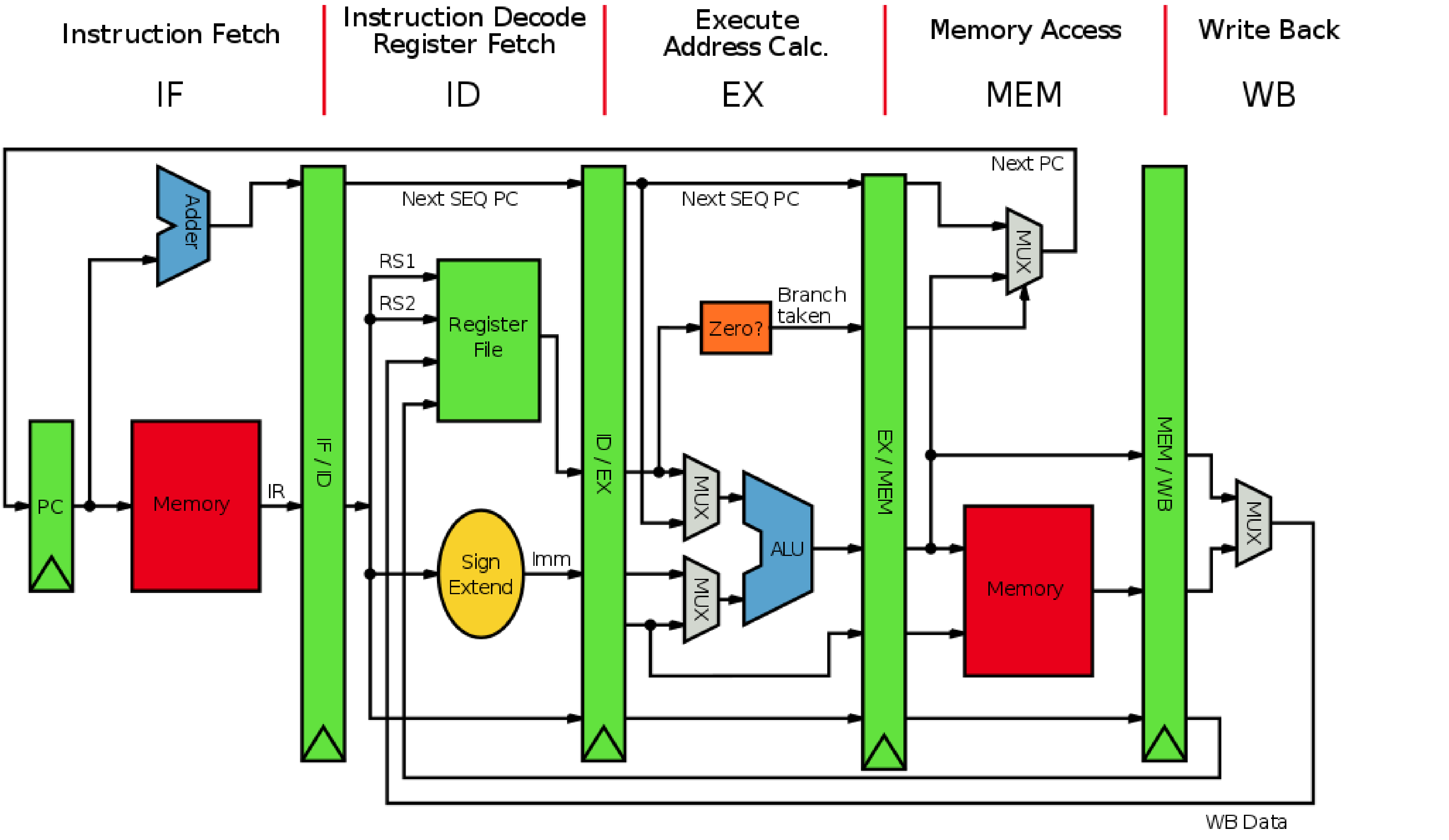
- http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Summary.pdf

Personal Mobile Device

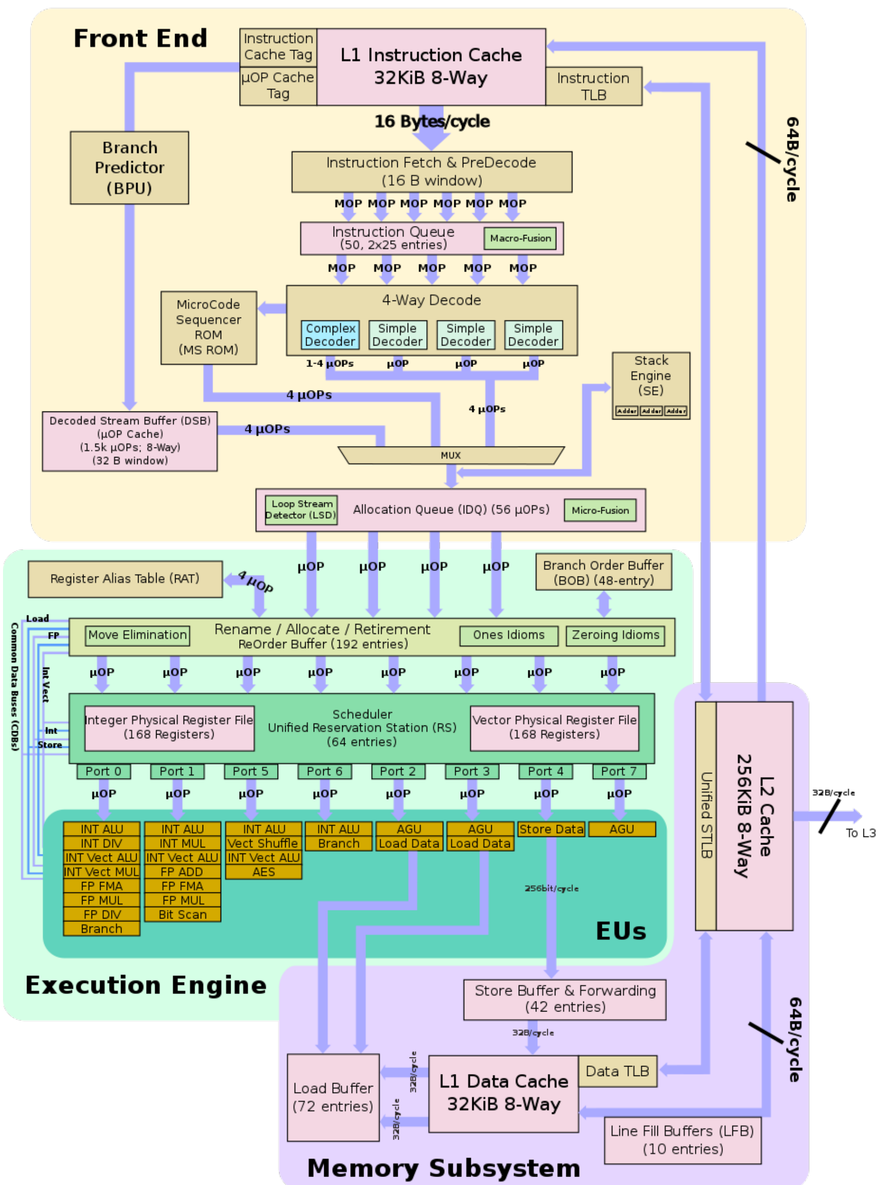


- Qualcomm Snapdragon
  - 1 GHz to 2.7 GHz
  - 2-4 cores
  - 11 stage integer pipeline with 3-way decode and 4-way out-of-order speculative issue superscalar execution

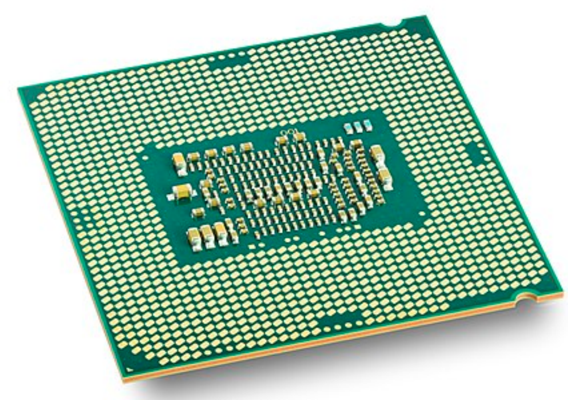
Pipelining


Instruction Fetch
IF
Instruction Decode
Register Fetch
ID
Execute
Address Calc.
EX
Memory Access
MEM
Write Back
WB
Next PC
Next SEQ PC
Next SEQ PC
Adder
RS1
RS2
Register
File
Branch
taken
Zero?
MUX
PC
Memory
IR
IF / ID
ID / EX
MUX
ALU
MUX
EX / MEM
MEM / WB
MUX
Sign
Extend
Imm
Memory
WB Data

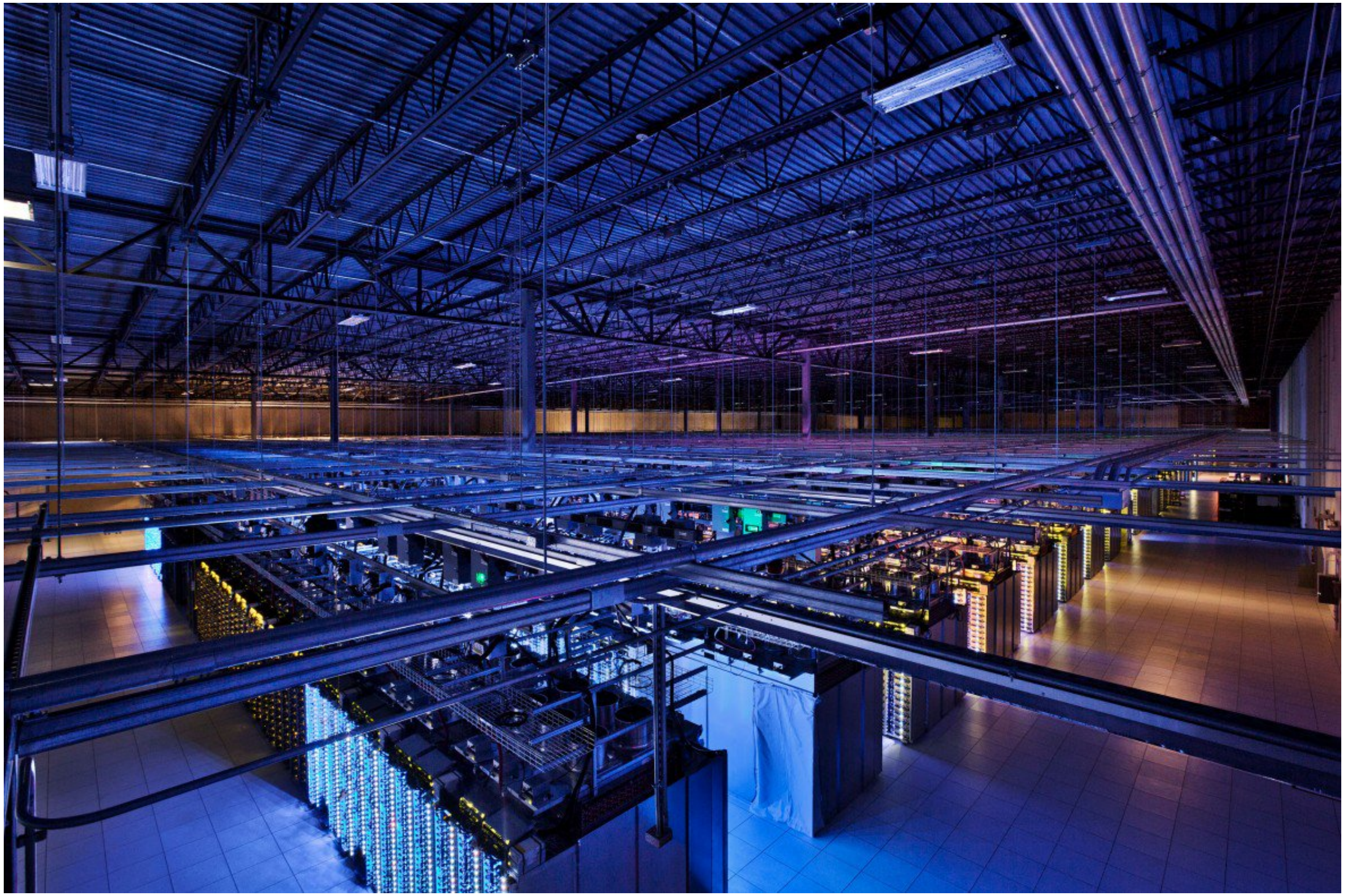
Desktop/Server



- Broadwell
  - 14-19 stage pipeline
  - 2-22 cores

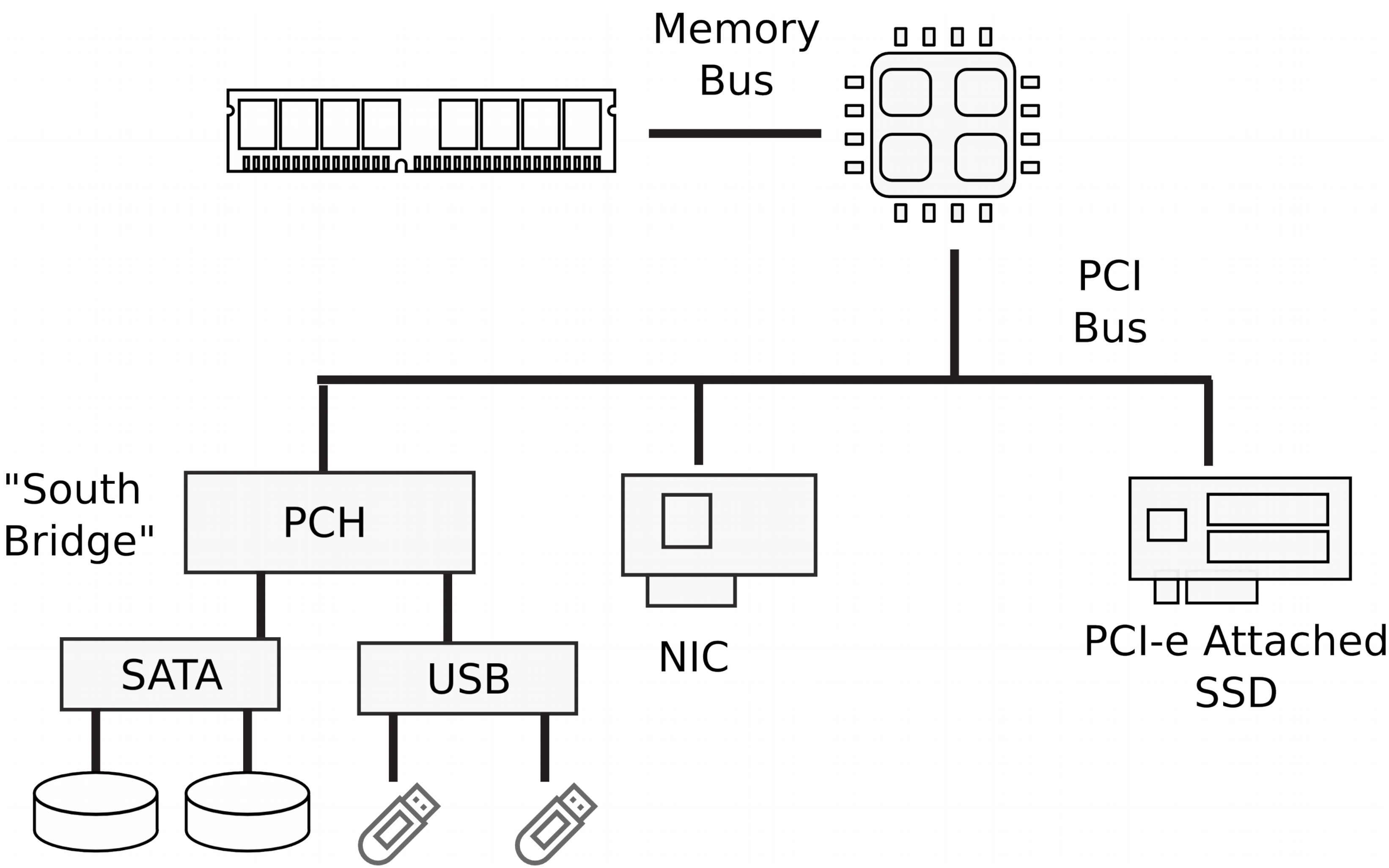


Warehouse-Scale

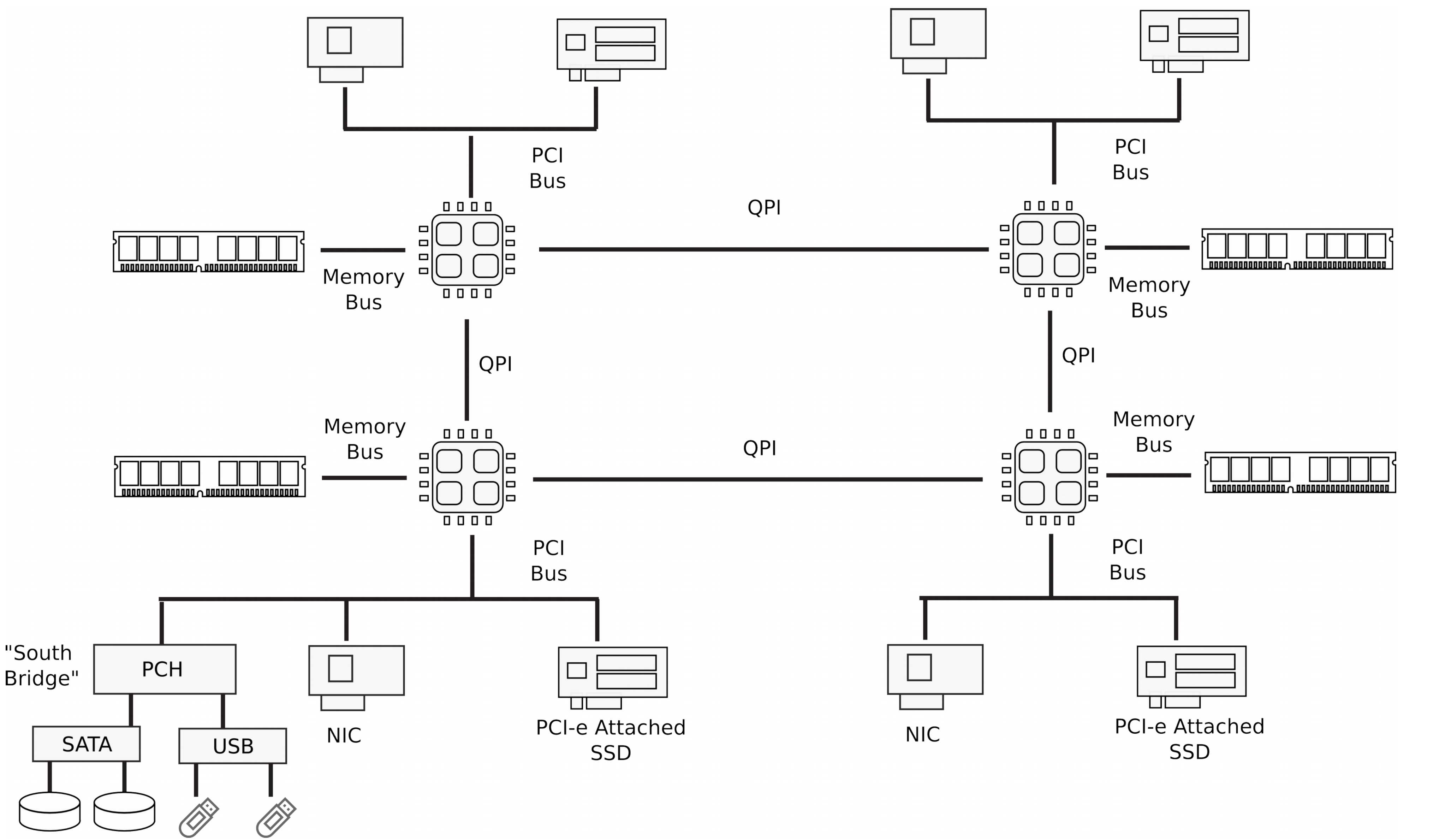


- Google datacenter

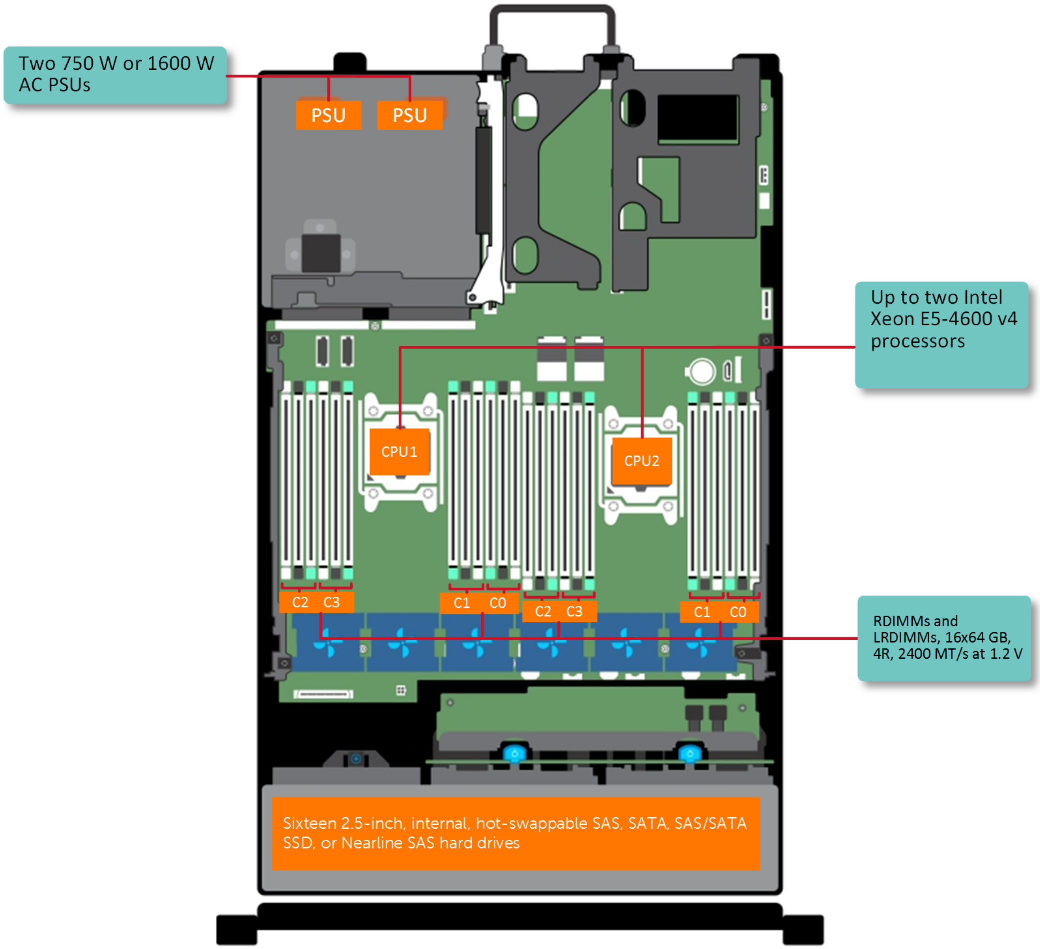
I/O Devices

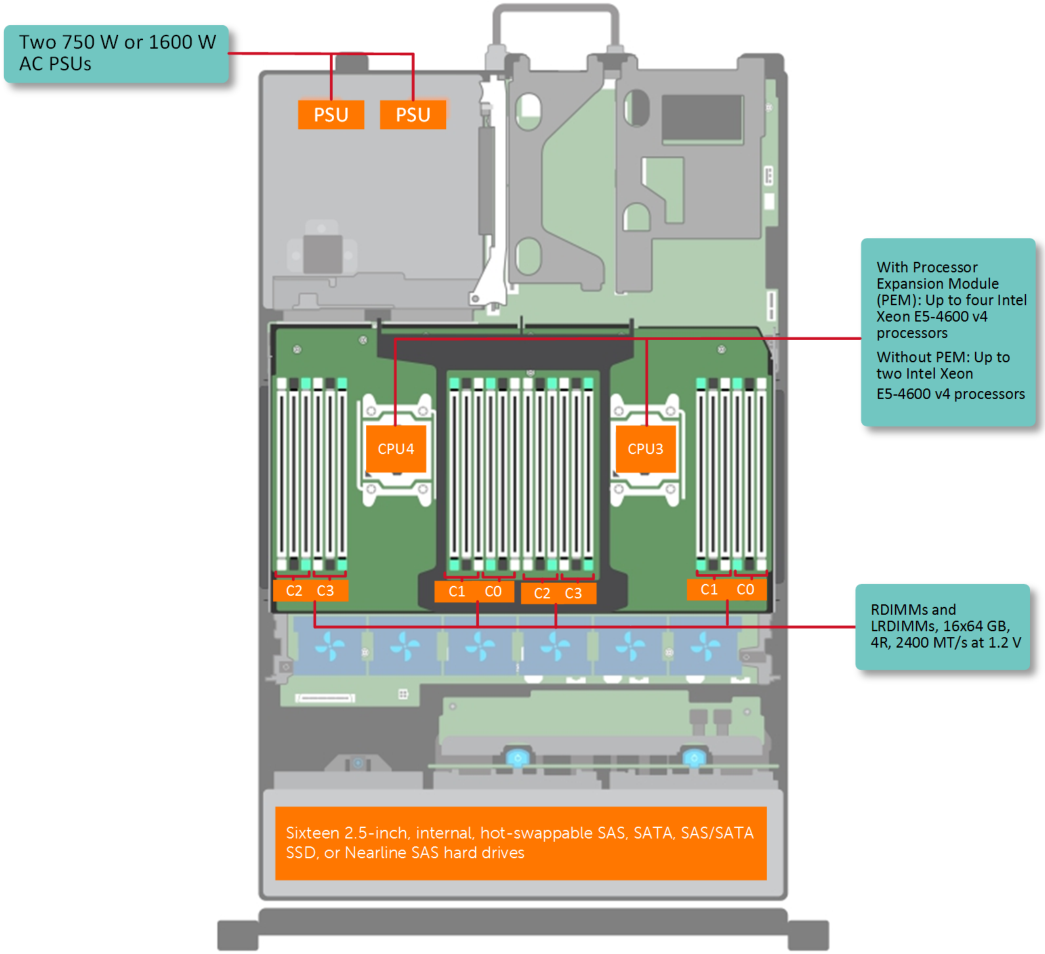

Memory
Bus
PCI
Bus
"South
Bridge"
PCH
NIC
PCI-e Attached
SSD
SATA
USB

Multi-socket machines



Dell R830 4-socket server

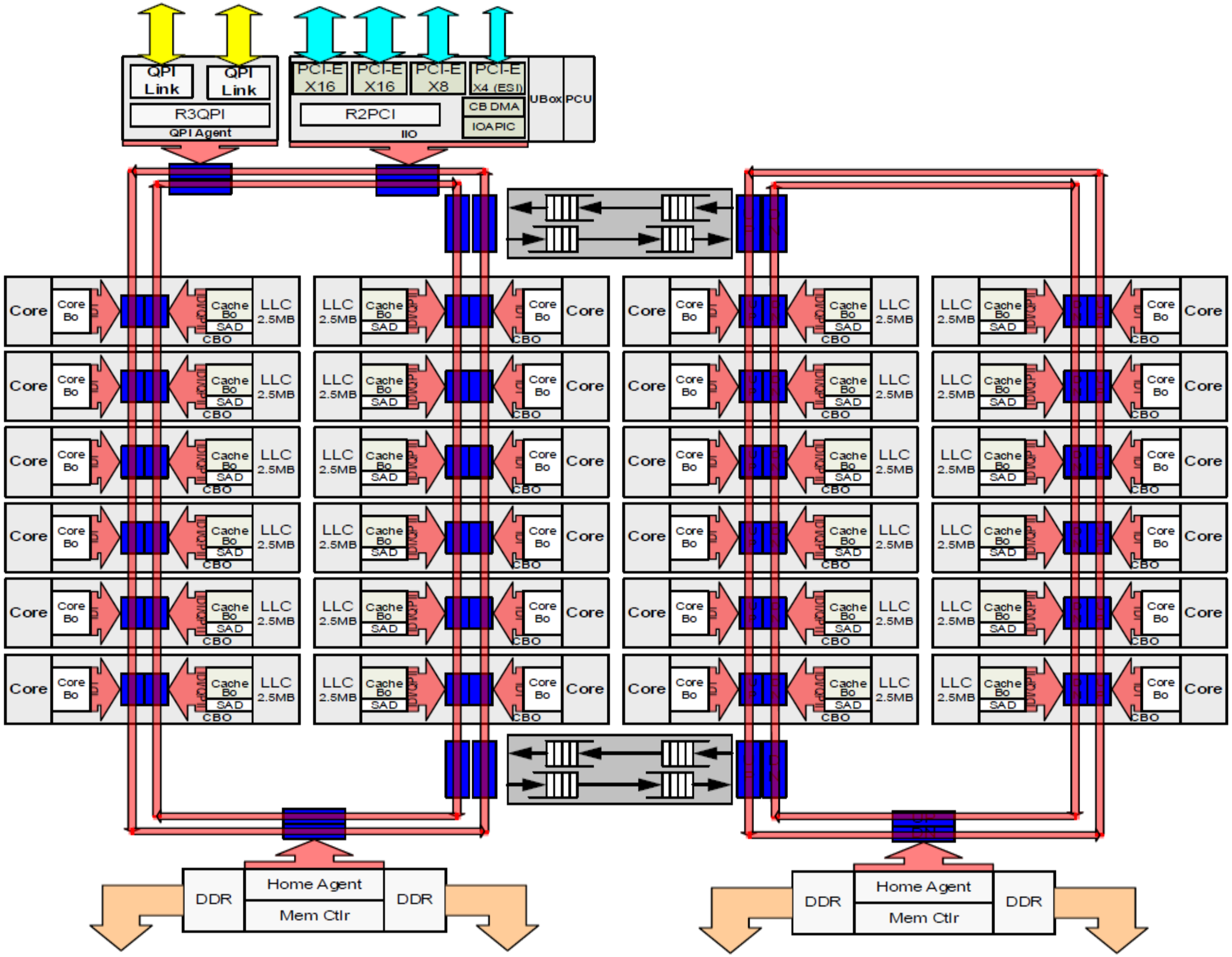




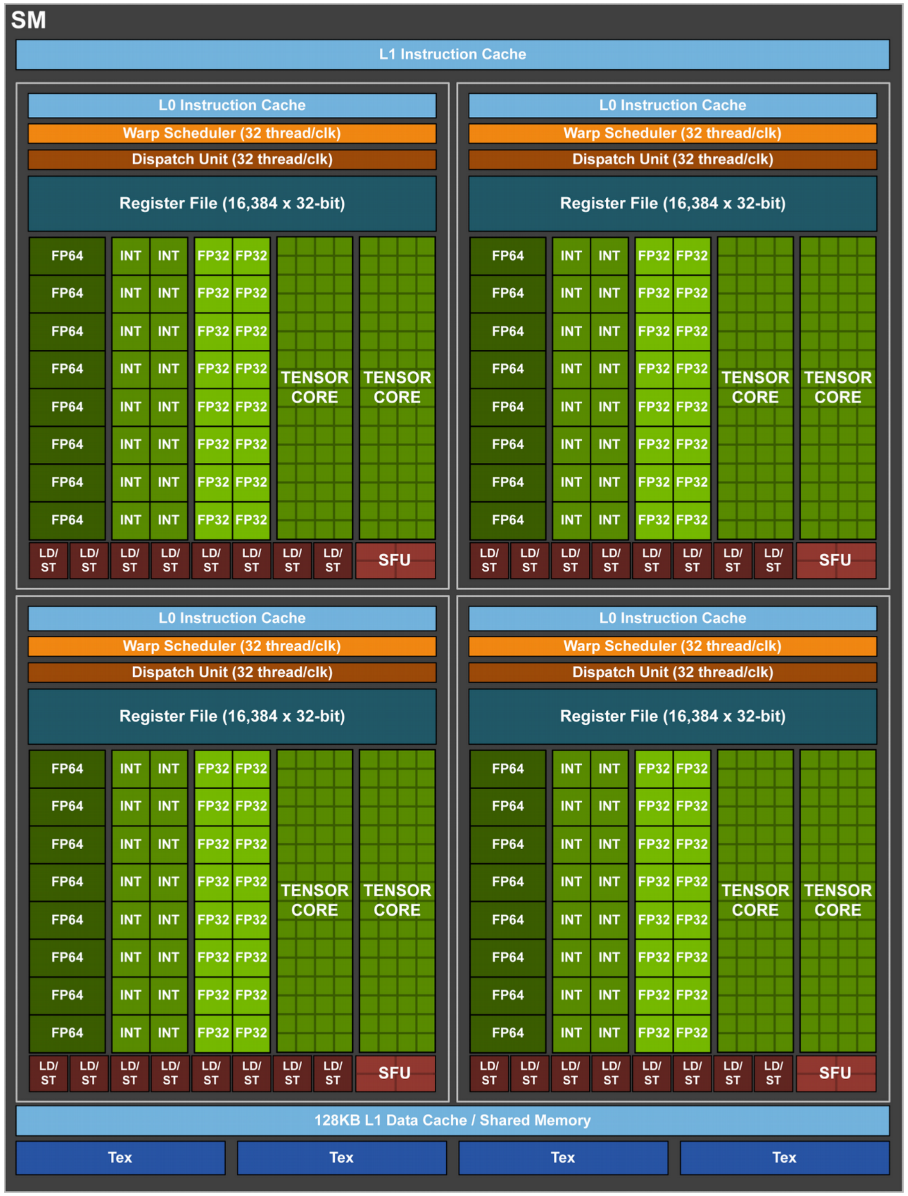
Dell Poweredge R830 System Server with 2 sockets on the main floor and 2 sockets on the expansion

http://www.dell.com/support/manuals/us/en/19/poweredge-r830/r830_om/supported-configurations-for-the-poweredge-r830-system?guid=guid-01303b2b-f884-4435-b4e2-57bec2ce225a&lang=en-us

Intel Broadwell (16+ cores)



SIMD: Nvidia V100



- Volta GV100 Full GPU with 84 SM Units

Thank you!