**Definition: Processor**

**Hardware origins**

18th century machines

**L’Ecrivain**

1770

Programmable, yet not a computer in today’s definition (not Turing complete)
**Definition: Processor**

Digital Computers

**Hardware origins**

- Patents by Konrad Zuse (Germany), 1936.
- First digital computer: Z1 (Germany), 1937: Relays, programmable via punch tape, clock: 1 Hz, 64 words memory à 22-bit, 2 registers, floating point unit, weight: 1 t.
- First freely programmable (Turing complete) relays computer: Z3 (Germany), 1941: 5.3 Hz
- Atanasoff Berry Computer (US) 1942: Vacuum tubes, not Turing complete.
- Colossus Mark 1 (UK) 1944: Vacuum tubes (not Turing complete).
- “First Draft of a Report on the EDVAC” (Electronic Discrete Variable Automatic Computer) by John von Neumann (US), 1945: Influential article about core elements of a computer: Arithmetic unit, control unit (Sequencer), memory (holding data and program), and I/O.
- ENIAC (Electronic Numerical Integrator And Computer) (US) 1946: Programed by plugboard, First Turing complete vacuum tubes based computer, clock: 100 kHz, weight: 27 t on 167 m².

---

**Computer Architectures**

**von Neumann Architecture**

- Control unit
  - Sequentially addresses program and data memory and fetches next instruction. Controls next ALU operations and determines the next instruction (based on ALU status).
  - Arithmetic Logic Unit (ALU)
    - Fetches data from memory. Executes arithmetic/logic operation. Writes data to memory.
- Input/Output
- Memory
  - Program and data is not distinguished. Programs can change themselves.

---

**Harvard Architecture**

- Control unit
  - Concurrently addresses program and data memory and fetches next instruction. Controls next ALU operations and determines the next instruction (based on ALU status).
- Arithmetic Logic Unit (ALU)
- Fetches data from memory. Executes arithmetic/logic operation. Writes data to memory.
- Input/Output
- Program memory
- Data memory

---

**A simple processor (CPU)**

- Decoder/Sequencer
  - Can be a machine in itself which breaks CPU instructions into concurrent micro code.
- Execution Unit / Arithmetic-Logic-Unit (ALU)
  - A collection of transformational logic.
- Memory
- Registers
  - Instruction pointer, stack pointer, general purpose and specialized registers.
- Flags
  - Indicating the states of the latest calculations.
- Code/Data management
  - Fetching, Caching, Storing.
Pipeline

Some CPU actions are naturally sequential (e.g. instructions need to be first loaded, then decoded before they can be executed).

More fine-grained sequences can be introduced by breaking CPU instructions into micro-code.

- Overlapping those sequences in time will lead to the concept of pipelines.
- Same latency, yet higher throughput.
- (Conditional) branches might break the pipelines.
- Branch predictors become essential.

Parallel pipelines

Filling parallel pipelines (by alternating incoming commands between pipelines) may employ multiple ALU's.

- (Conditional) branches might again break the pipelines.
- Interdependencies might limit the degree of concurrency.
- Same latency, yet even higher throughput.
- Compilers need to be aware of the options.

Out of order execution

Breaking the sequence inside each pipeline leads to "out of order" CPU designs.

- Replace pipelines with hardware scheduler.
- Results need to be "re-sequentialized" or possibly discarded.
- "Conditional branch prediction" executes the most likely branch or multiple branches.
- Works better if the presented code sequence has more independent instructions and fewer conditional branches.
- This hardware will require (extensive) code optimization to be fully utilized.
**SIMD ALU units**

Provides the facility to apply the same instruction to multiple data concurrently. Also referred to as "vector units".

- Requires specialized compilers or programming languages with implicit concurrency.

**GPU processing**

Graphics processor as a vector unit. Unifying architecture languages are used (OpenCL, CUDA, GPGPU).

**Hyper-threading**

Emulates multiple virtual CPU cores by means of replication of:
- Register sets
- Sequencer
- Flags
- Interrupt logic

while keeping the "expensive" resources like the ALU central yet accessible by multiple hyper-threads concurrently.

- Requires programming languages with implicit or explicit concurrency.

Examples: Intel Pentium 4, Core i5/i7, Xeon, Atom, Sun UltraSPARC T2 (8 threads per core).

**Multi-core CPUs**

Full replication of multiple CPU cores on the same chip package.

- Often combined with hyper-threading and/or multiple other means (as introduced above) on each core.
- Cleanest and most explicit implementation of concurrency on the CPU level.

- Requires synchronized atomic operations.
- Requires programming languages with implicit or explicit concurrency.

Historically the introduction of multi-core CPUs ended the "GHz race" in the early 2000's.

**Virtual memory**

Translates logical memory addresses into physical memory addresses and provides memory protection features.

- Does not introduce concurrency by itself.
- Is still essential for concurrent programming as hardware memory protection guarantees memory integrity for individual processes/threads.
Alternative Processor Architectures: IBM Cell processor (2001)

- 8 cores for specialized high-bandwidth floating point operations and 128-bit registers
- Theoretical 25.6 GFLOPS at 3.2 GHz


- Low cost 32-bit processor ($8)
- 8 cores with 2 kB local memory
- 40 kB shared memory
- No interrupts!
- 8 semaphores

Multi-CPU systems

- Multi-CPU on the same memory
  - multiple CPUs on the same motherboard and memory bus, e.g. servers, workstations

- Multi-CPU with high-speed interconnects
  - various supercomputer architectures, e.g. Cray XE6:
    - 12-core AMD Opteron, up to 192 per cabinet (2304 cores)
    - 3D torus interconnect (160 GB/sec capacity, 48 ports per node)

- Cluster computer (Multi-CPU over network)
  - multiple computers connected by network interface, e.g. Sun Constellation Cluster at ANU:
    - 1492 nodes, each: 2x Quad core Intel Nehalem, 24 GB RAM
    - QDR InfiniBand network, 2.6 GB/sec
Summary

- History
- Architectures
  - Pipelines
  - Parallel pipelines
  - Out of order execution
  - Vector machines
  - Multi-core CPUs
  - Virtual memory