Computer Systems

Wolfgang Schreiner
Research Institute for Symbolic Computation (RISC-Linz)
Johannes Kepler University

Wolfgang.Schreiner@risc.uni-linz.ac.at
http://www.risc.uni-linz.ac.at/people/schreine

Overview

How can we manage the complexity of computer systems?

- A computer can be regarded as a hierarchy of levels.
  - Each level performs some well-defined function.
  - Each level is implemented on top of the next lower level.
  - The lowest level is the physical level (hardware).
- Each level serves as a layer of abstraction.
  - Its implementation is not interesting to the upper layers.
  - It hides the details of the lower layers from the upper layers.

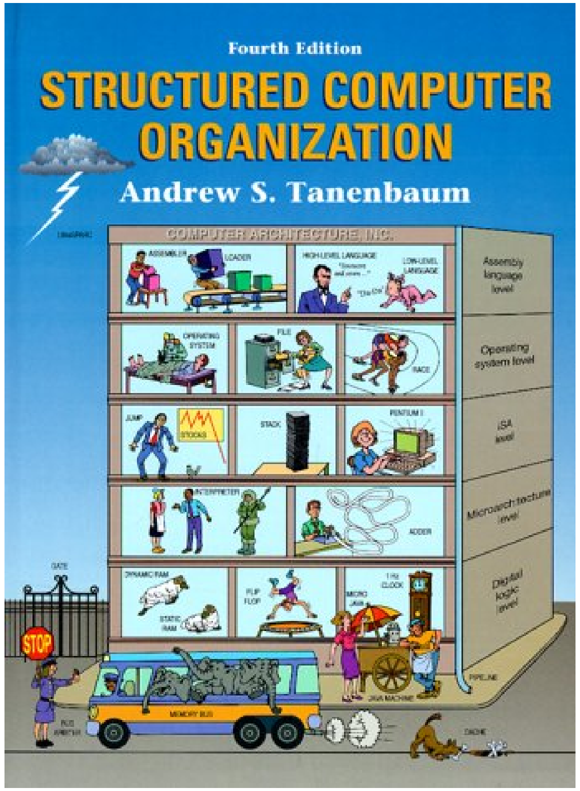
We can understand a computer on different layers of abstraction.

Contents

- Introduction.
  - Overview and historic development.
- Computer systems organization.
  - The components of a computer and their interconnection.
- The hierarchy of levels.
  1. Digital logic: The implementation of computation by digital devices.
  2. Microarchitecture: The structure of a computer processor.
  3. Instruction set architecture: The operations provided by a computer processor.
  4. Operating system: The extension of the instruction set by additional services.
  5. Assembly language: The generation of executable program from textual descriptions.
- Outlook.
  - Compilation: the generation of assembly programs from high-level languages.
  - Computer networks: the interconnection of computer systems to distributed systems.

Literature

Andew S. Tanenbaum: Structured Computer Organization, 4th ed.



Introduction

- Digital computer:
  - Machine that can solve problems by carrying out instructions.
- Program:
  - Sequence of instructions for performing a certain task.
- Machine language:
  - Instructions that can be executed by the hardware of a computer.

    Add two numbers.

    Check a number to see if it is zero.

    Copy a piece of data from one memory location to another.
  - Primitive instructions are as simple as possible.

Machine programs are difficult and tedious for people to use.

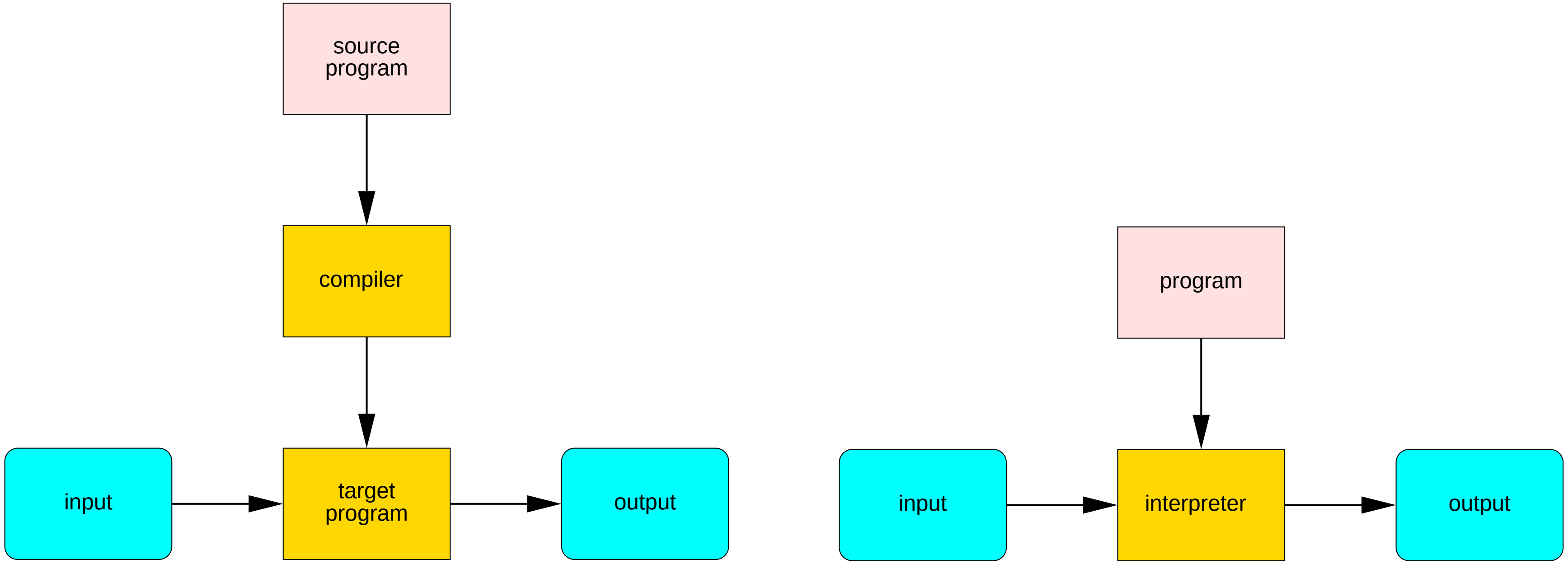
Programming Languages

Design a new set of instructions that is more convenient for people.

- Two different languages:
  - Language L0 executed by computer.
  - Language L1 used by people.
- Translation
  - Replace L1 program (the source) by an equivalent L0 program (the target).
  - The computer executes the generated L0 program.
- Interpretation
  - Write an L0 program that takes L1 programs as inputs and executes them.
  - The computer executes the L0 program (the interpreter).

Fundamental techniques of executing programs in other languages.

Translation and Interpretation



A language translator is also called a compiler.

Virtual Machine

Avoid thinking in terms of translation or interpretation.

- Image virtual machine.
  - Hypothetical computer M1 whose machine language is L1.
  - If such a machine can be constructed, no need for machine executing L0.
- People can write programs for virtual machine.
  - M1 may be built in hardware.
  - L1 programs may be translated to L0 programs.
  - L1 programs may be interpreted by L0 program.

We can neglect implementation of M1 when writing L1 programs.

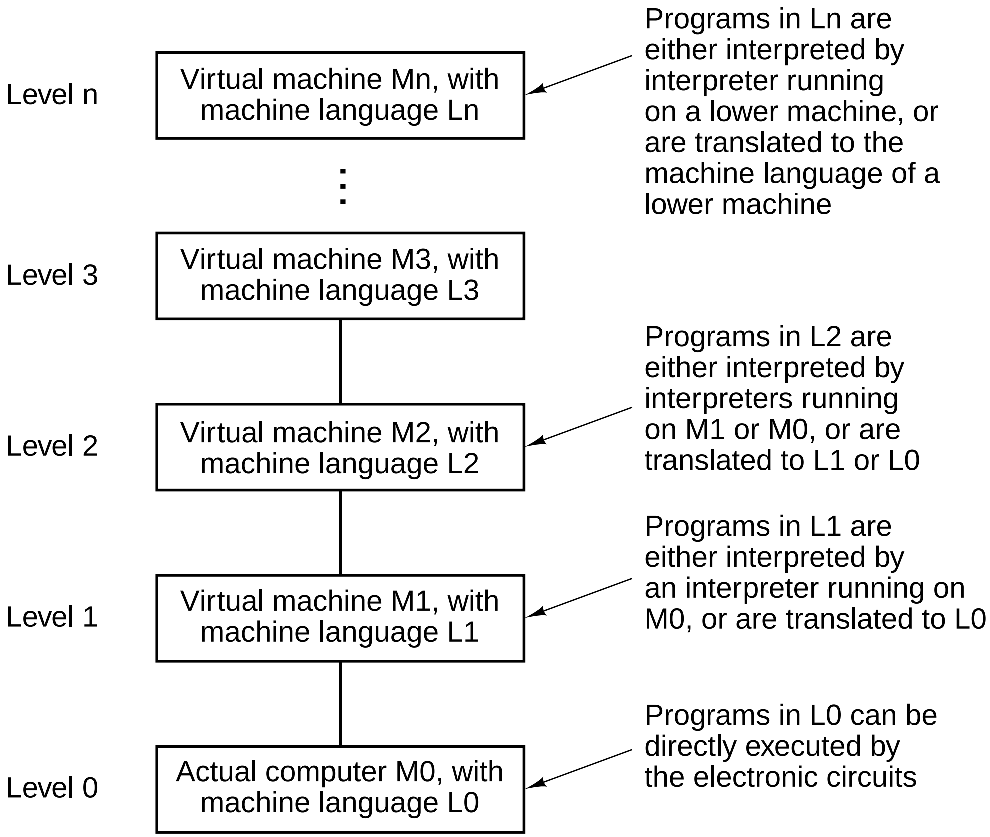
Language Layers

For practical implementation, L0 and L1 should not be too different.

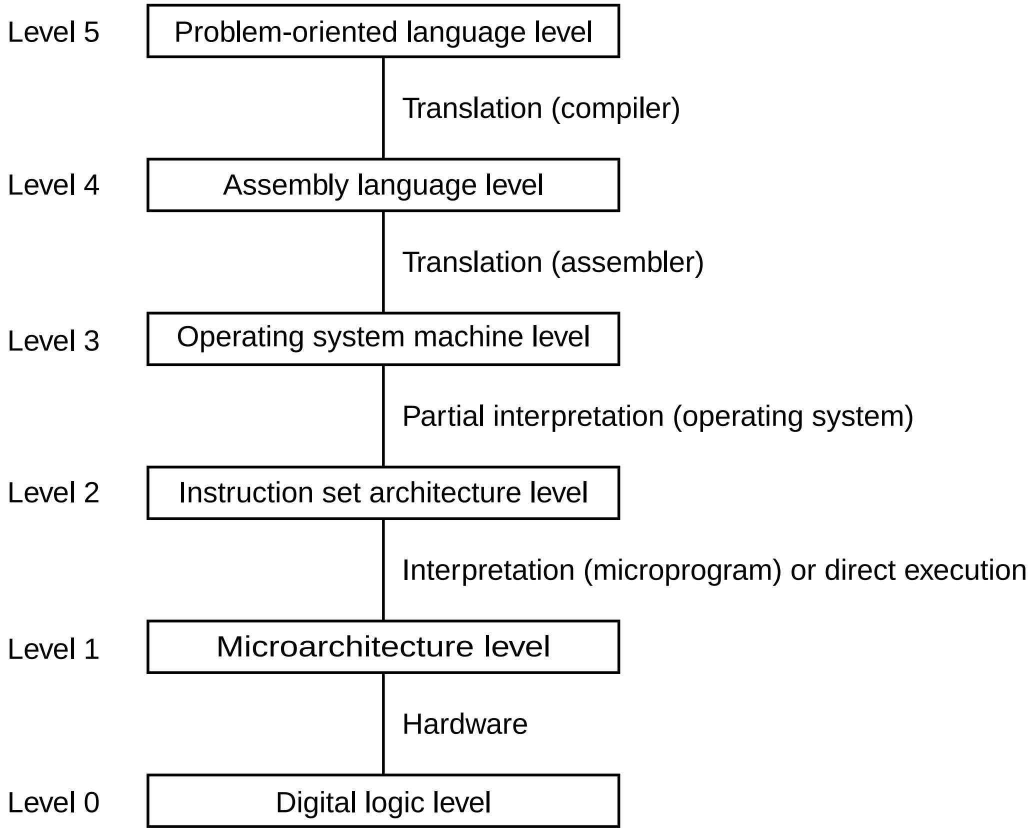
- L1 may be a bit better than L0.
  - L1 is far from ideal for most applications.
- Continue language building process.
  - Invent language L2 that is a bit more people-oriented than L1.
  - Implement virtual machine M2 on top of M1.

A hierarchy of language layers is constructed.

A Multi-Level Machine



Contemporary Multi-Level Machine



Digital Logic Level

Assume: electrical engineering provides transistors.

- Interesting objects are logic gates.
  - Built from transistors (analog components).
  - Modeled accurately as digital devices.
- Operates on digital inputs.
  - Signals representing 0 or 1.
  - Computes simple function on inputs (AND, OR, . . .).
- Gates can be used to implement registers.
  - Groups of 1-bit memories (e.g., 16, 32, or 64).
  - Each 1-bit memory implemented by a handful of gates.

Transition from analog physics to digital computation.

Microarchitecture Level

Assume: digital logic provides gates and registers.

- Arithmetic Logical Unit (ALU)
  - Simple arithmetic operations.
  - Connected to registers by a data path.
- Operation of data path may be controlled by microprogram.
  - Software interpreter inside a processor.
  - Microprogram is interpreter for instructions at level 2.
    * ADD instruction: Fetch instruction, locate operands, bring them into registers, compute sum by the ALU, write result to destination.
  - Nowadays data path is more often controlled directly by hardware.

Architecture of a computer processor.

Instruction Set Architecture Level

Assume: microarchitecture provides basic processor units.

- Set of instructions executed by processor.
  - Carried out by microprogram or in hardware.
  - Published by processor manufacturer in reference manual.

Machine language of a computer processor.

Operating System Machine Level

Assume: ISA provides set of instructions.

- Extend this set of instructions by new services.
  - All ISA instructions are still visible on this layer.
- New services are carried out by operating system.
  - Interpreter running at level 2.
  - Provides additional instructions, different memory organization, ability to run multiple programs concurrently, and other features.

Machine language extended by operating system services.

Assembly Language Level

Assume: OS provides set of instructions and services.

- Assembly language: symbolic form of layer 3 language.
  - Languages on lower levels are numeric; programs are sequences of numbers.
  - Assembly language is textual; programs are sequences of readable commands.
  - Assembler translates assembly program to layer 2 program.

System programmers build here services for application programmers.

Problem-oriented Language Level

Assume: assembly language provides interface to system services.

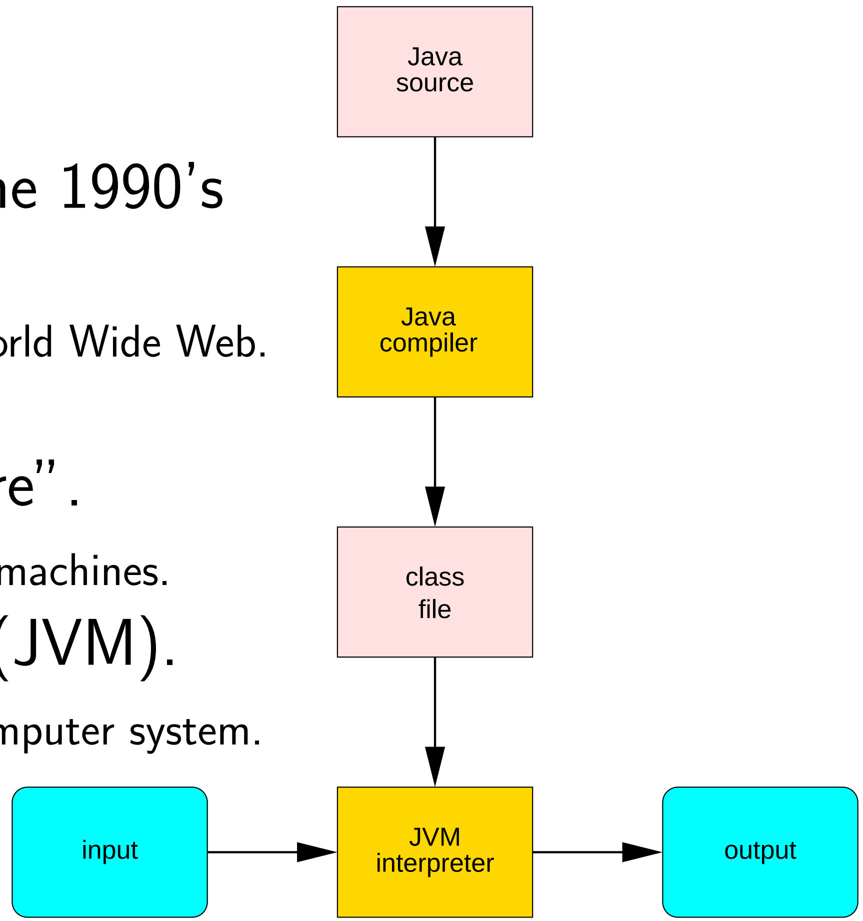
- Problem-oriented languages are compiled to assembly language.
  - Fortran, C, Modula, Ada, C++, Oberon, ...
- Scripting/domain-specific languages are often interpreted.
  - Interpreter written in assembly language or a compiled language.
  - Sh, TCL, Perl, ...
  - Mathematica, Maple, Matlab, ...

Application programmers build here services for users.

The Programming Language Java

`http://www.javasoft.com`

- Developed in the beginning of the 1990's
  - Computer company Sun.
  - Gained popularity with the uprise of the World Wide Web.
  - "Java" is US slang for "coffee".
- Idea: "write once, run everywhere".
  - Target code should run in same way on all machines.
- Solution: Java Virtual Machine (JVM).
  - Abstraction layer between language and computer system.
  - Executes the JVM byte code language.



Computer Architecture

The organization of a computer.

- Architecture:
  - The set of data types, operations, features provided of each machine level to an user of that level (e.g., how much memory is available).
  - Implementation aspects are not part of the architecture (e.g., by what chip technology the memory is implemented).
- Computer Architecture:
  - Those parts of a computer system that are visible to a programmer.

Aspects of a computer that are of interest to a user.

Hardware and Software

- Computer hardware:
  - Tangible objects (ICs, boards, cables, power supplies, memories, ...)
  - Based on electronics, optics, magnetics, ...
- Computer software:
  - Algorithms (instructions to perform a task) in a particular computer representation (programs).
  - Software can be stored on physical media (hard disk, floppy, CD-ROM, ...).
  - Essence of software is their information content, not their physical representation.
- Distinction has considerably blurred.
  - Functionality implemented in hardware can be performed in software and vice versa.
  - Hardware and software are logically equivalent.

"Hardware is just petrified software" (Karen Panetta Lentz).

The Invention of Microprogramming

Originally computers had two levels only.

- 1940s: ISA and digital logic.
  - ISA: all programs were written on this level.
  - Digital logic: programs were executed on this level.
  - Disadvantage: complicated and unreliable circuits.
- 1951: Maurice Wilkes suggested the design of a third level.
  - Built-in interpreter (microprogram) to interpret ISA programs.
  - Only microprogram instructions have to be executed in hardware.
  - Advantage: drastic simplification of hardware.

By 1970 dominant idea in computer architecture.

The Invention of the Operating System

Originally: a computer was operated by a single user at a time.

- Programmer reserved time on the computer.
  - Put in a deck of program cards for the Fortran compiler and a a deck of program cards for the Fortran program (two times).
  - Computer generated deck of program cards for machine program.
  - Programmer put generated deck (and subroutine library deck) into computer for execution.
- 1960s: operator's job was automated by an operating system.
  - Program that was kept in computer at all times.
  - Control cards together with program cards and data cards were inserted.
  - Operating system compiled and executed program.
- Operating systems were extended and refined.
  - New instructions: system calls.
  - Simultenous access from remote terminals: timesharing.

Migration of Functionality to Microcode

Designers realized power of microcode.

- Add instructions by extending the microprogram.
  - Add/modify "hardware" by programming.
- Explosion of machine instruction sets.
  - Instruction sets became bigger and bigger.
  - Perform tasks faster than by existing instructions.
  - Example: integer multiplication/division, floating-point arithmetic, procedure call/return, looping instructions, string instructions, ...

Replace instruction sequences by new machine instructions.

The Elimination of Microprogramming

In the 1960s and 1970s, microprograms grew fat.

- CISC: Complex Instruction Set Computers.
  - Microprograms tended to get slower and slower.
  - Complex instructions were rarely used by compilers.
- Speed up machines by simplification:
  - Eliminate microprograms.
  - Reduce instruction sets.
  - Have remaining instructions be directly executed by hardware.
- RISC: Reduced Instruction Set Computers.

Hardware/software boundary is arbitrary and constantly changing.

Milestones in Computer Architecture

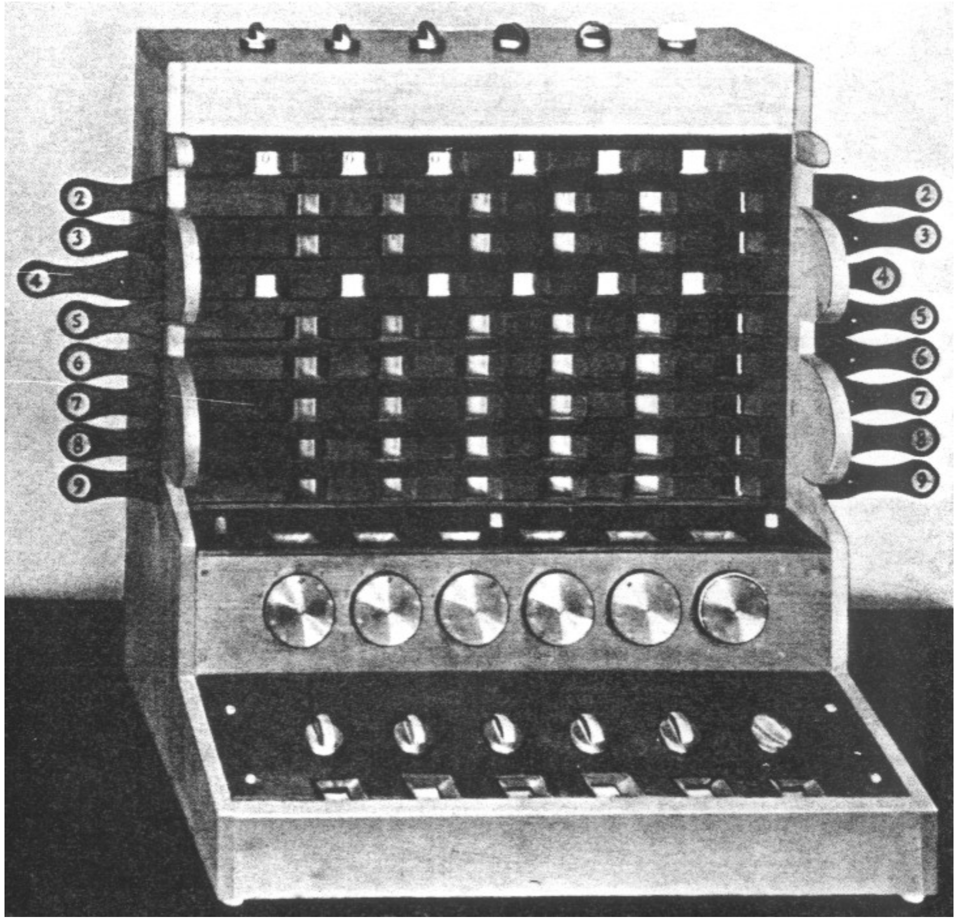
- Generation 0: Mechanical Computers (1642–1945)
- Generation 1: Vacuum Tubes (1945–1955)
- Generation 2: Transistores (1955–1965)
- Generation 3: Integrated Circuits (1965–1980)
- Generation 4: Very Large Scale Integration (1980–2020?)

Structured by fundamental changes in underlying technologies.

Generation 0: Mechanical Computers

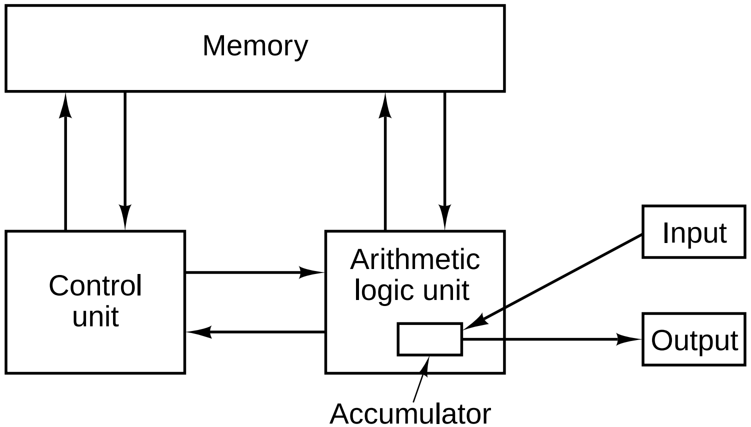
- Early pioneers
  - Wilhelm Schickard (1623): mechanical (addition, subtraction, multiplication, division).
  - Blaise Pascal (1642): mechanical (addition and subtraction).
  - Gottfried Wilhelm von Leibnitz (1670s): mechanical (also multiplicaton and division).
- Charles Babbage: 1820s
  - Difference engine: mechanical, addition and subtraction.
  - Analytical machine: mechanical (never functional), controlled by punched card programs (world's first computer programmer: Ada Augusta Lovelace).
- Konrad Zuse: 1930s-1940s.
  - Z-Series: Electromagnetic relays.
- Howard Aiken: 1940s.
  - Mark I and II: electromagnetic relays.

Schickard's Calculator



Generation 1: Vacuum Tubes

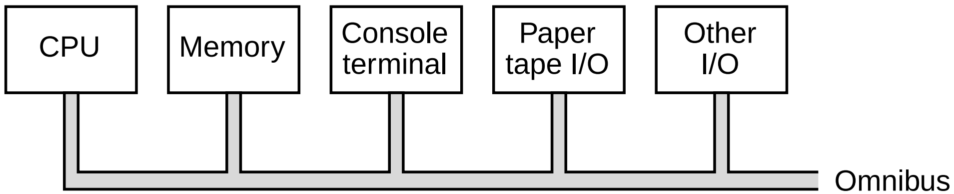
- Colossus: 1943, GB.
  - First electronic digital computer.
  - Cracking the ENIGMA cyphers (Alan Turing participated in design).
- ENIAC: 1946, US.
  - Electronic computer (18,000 vacuum tubes), 20 registers with 10 digit decimal numbers.
  - Computation of tables for heavy artillery.
- von Neumann Machine: John von Neumann, Princeton 1950.
  - Basis of today's architectures.
  - Memory, ALU, control unit, input, output.
- 1958: first computer by IBM.



Generation 2: Transistors

The transistor was invented in 1948 at Bell Labs.

- PDP-1: Digital Equipment Corporation, 1961.
  - First commercial transistorized computer (mini-computer).
  - Visual display with 512 × 512 pixels.
- PDP-8: Digital Equipment Corporation, ca. 1965.
  - A single bus connected components.
  - 50,000 units were sold.
  - Established DEC as a major player.



- The first super-computers emerged.
  - Control Data Corporation (CDC).
  - Seymour Cray: CDC 6600, CDC 7600, Cray-1.

Generation 3: Integrated Circuits

Silicon integrated circuit was invented in 1958.

- Dozens of transistors could be put on a single chip.
  - Computer became smaller, faster, cheaper.
- System/360 series, IBM, 1964.
  - Both scientific and commercial applications.
  - Replaced two separate strands of system designs at IBM.
  - First commercial computer with multiprogramming.
  - First machine that could emulate other computers by microprograms.
  - 32 bit computer whose memory was byte-addressed.
- PDP-11, DEC, end of 1960s.
  - Highly successful, especially at universities.

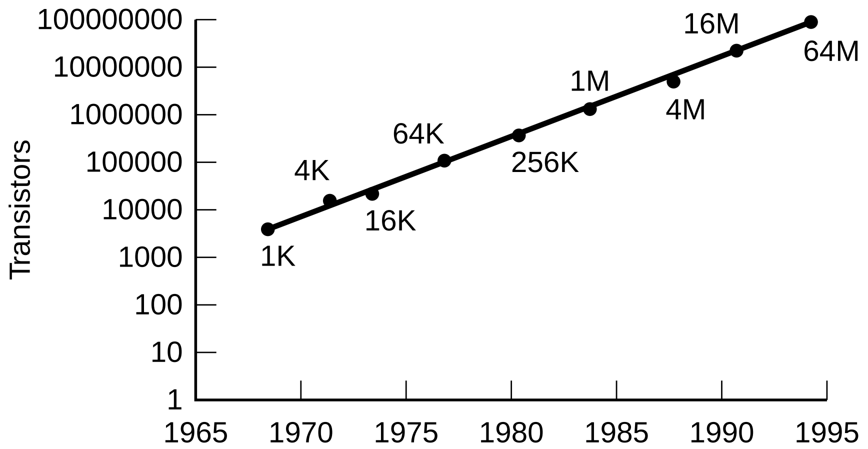
Generation 4: Very Large Scale Integration

In the 1980s, VLSI emerged.

- Tens of thousands transistors could be put on a single chip.
  - Today: millions of transistors.
  - Computers became even faster and cheaper.
- PC: the personal computer.
  - Originally: computer kits without software.
  - Xerox PARC: graphical user interfaces, windows, mouse.
  - Steve Jobs, Steve Wozniak: Apple, Apple II (1970s).
  - IBM PC, 1981: best-selling computer in history.
  - PC clones industry emerged.
- Mid-1980s: new processor designs.
  - RISC architectures, super-scalar CPUs.

Moore's Law

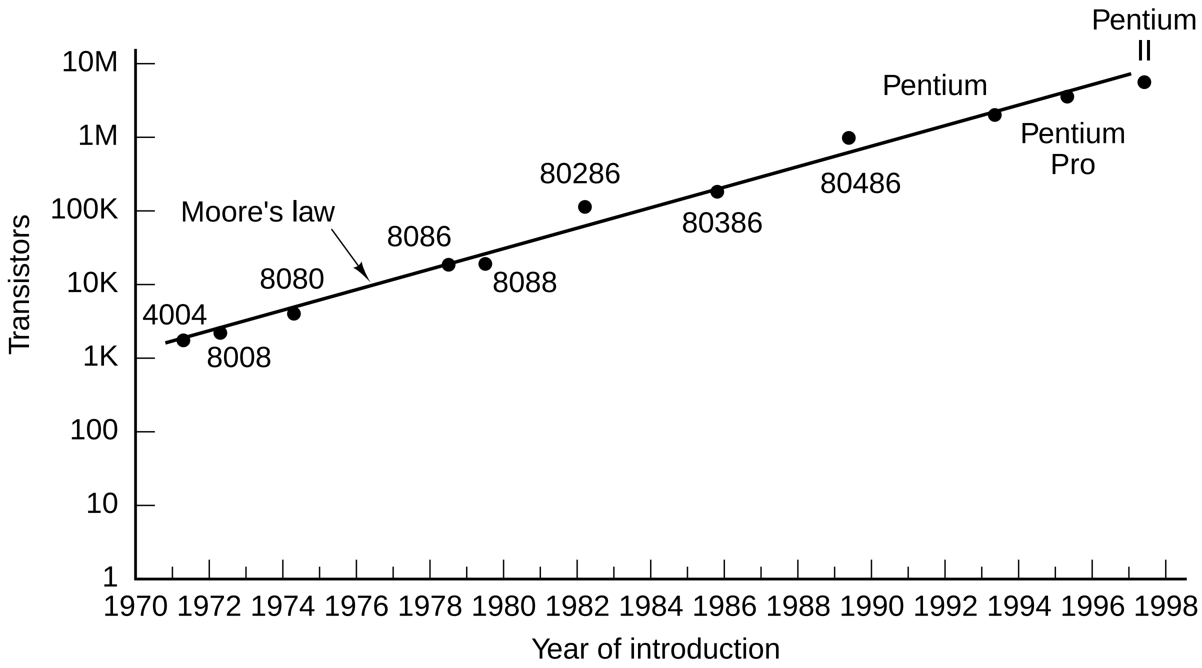
- Observation by Gordon Moore, 1965.
  - Number of transistors per chip doubles every 18 months.
  - Memory sizes and processor speed increases at the same rate.
  - Remarkably correct until today.



Unfortunately, there are physical limits for circuit densities.

Example: The Intel Processor Line

Intel was founded in 1968 by Gordon Moore and Robert Noyce.



Further on: Pentium III, Pentium IV, IA-64.

Generation 5: ?

What will come in 2020?

- 3-dimensional circuit designs.
  - Pack transistors in cubes instead of chips.
- Optical computing.
  - Replace electronics by optics.
- Molecular computing.
  - Use chemical/biological processes for computing.
  - 4 bit computations in test tubes have been performed.
- Quantum computing.
  - Use other physical properties for computing.
  - Special applications only, e.g., quantum cryptography.