ENGR (XMUT) 101
Engineering Technology

A/Prof. Pawel Dmochowski

School of Engineering and Computer Science
Victoria University of Wellington

Victoria
UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

CAPITAL CITY UNIVERSITY

Week 9 Lecture 1a

- Main topics (Weeks 9-15)
  - Introduction to Engineering Technology
  - Number systems
  - Logic Gates
  - Boolean Algebra

ENGR 101 – Engineering Technology

- Engineering
  - Profession in which knowledge of math and natural sciences, gained by study, experience, and practice, is applied with judgement to develop ways to use, economically, the materials and forces of nature for the benefit of humankind.

- Technology
  - Application of scientific knowledge for practical purposes, especially in industry
  - Comprised of the products and processes created by engineers to meet our needs and wants

ENGR 101 – Engineering Technology

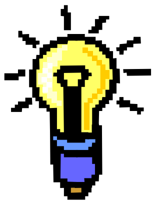
- Science
  - Investigation, understanding, and discovery of nature, its composition, and its behaviour (ie laws of nature)
- Engineering
  - Manipulating the forces of nature to advance humanity
  - Successful engineering design improves quality of life while working within technical, economic, business, societal, and ethical constraints
- Technology
  - Outcome of Engineering

What makes an Engineer?

- Curiosity about how things work

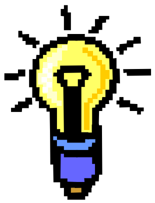
What makes an Engineer?

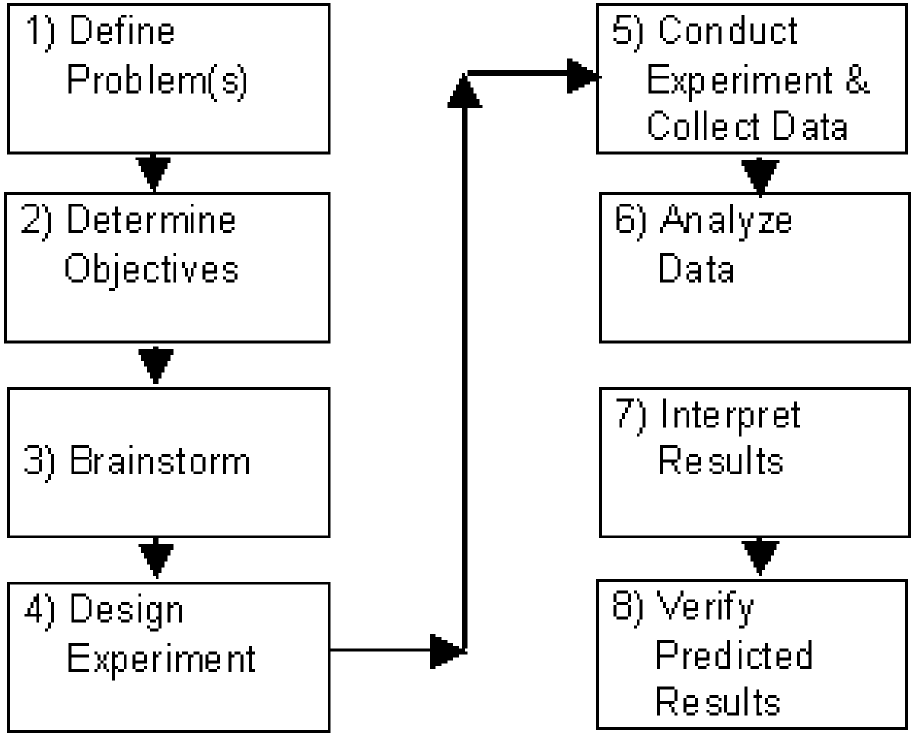
- Curiosity about how things work
- Desire to solve interesting problems



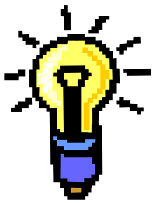
What makes an Engineer?

- Curiosity about how things work
- Desire to solve interesting problems
- Interest in design and experimentation

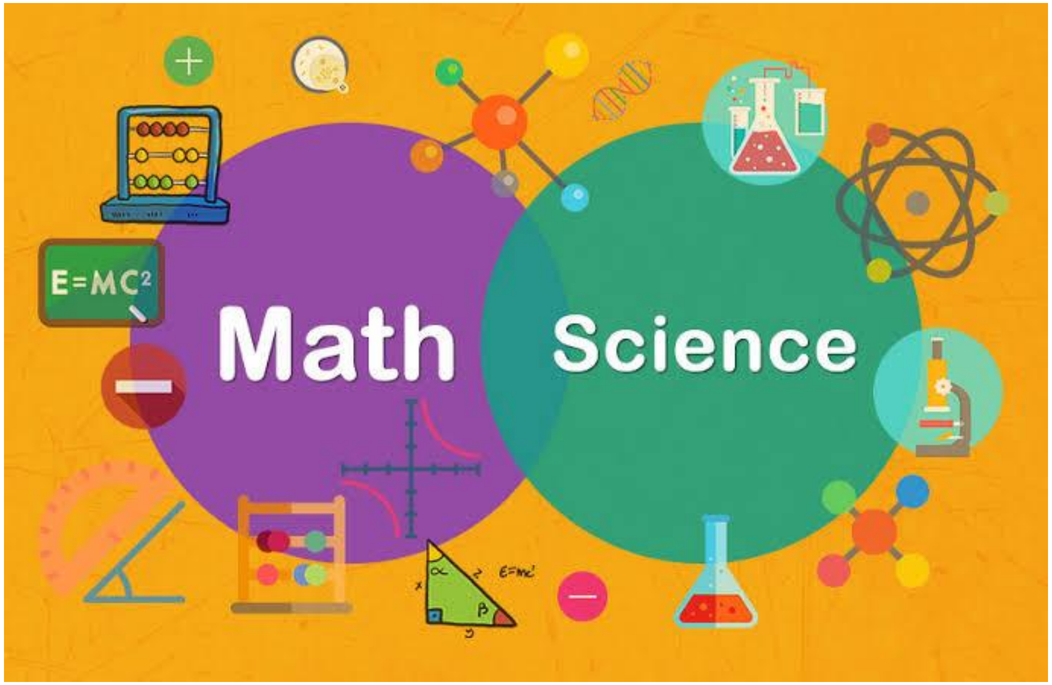




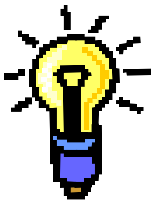
What makes an Engineer?



- Curiosity about how things work
- Desire to solve interesting problems
- Interest in design and experimentation
- Affinity for using math and science



What makes an Engineer?

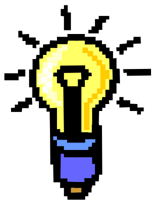


- Curiosity about how things work
- Desire to solve interesting problems
- Interest in design and experimentation
- Some affinity for using math and science
- Good teamwork and communication skills

What makes an Engineer?

- Curiosity about how things work
- Desire to solve interesting problems
- Interest in design and experimentation
- Some affinity for using math and science
- Good teamwork and communication skills
- Creative and adaptable

What makes an Engineer?



- Curiosity about how things work
- Desire to solve interesting problems
- Interest in design and experimentation
- Some affinity for using math and science
- Good teamwork and communication skills
- Creative and adaptable
- Eager to keep learning new things

What makes an Engineer?

- Curiosity about how things work
- Desire to solve interesting problems
- Interest in design and experimentation
- Some affinity for using math and science
- Good teamwork and communication skills
- Creative and adaptable
- Eager to keep learning new things

History of Engineering Technology

- Ancient Era:
  - Great Wall of China, Pyramids of Egypt, etc.
- Middle Era
  - Invention and use of gears
- Renaissance Era
  - Industrial revolution
- Modern Day
  - Computers and networks, etc.

History of Engineering Technology

- Great Wall of China

History of Engineering Technology

- Great Wall of China

- Pyramids of Egypt

History of Engineering Technology

- Great wall of China

- Pyramids of Egypt

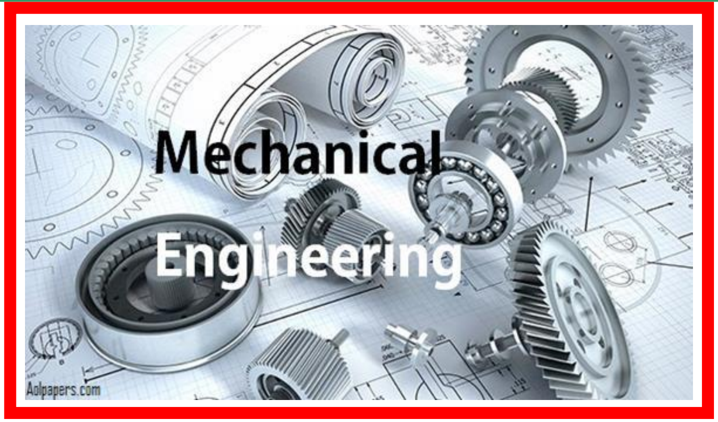
- Mayan step pyramids

Five Traditional Engineering Branches

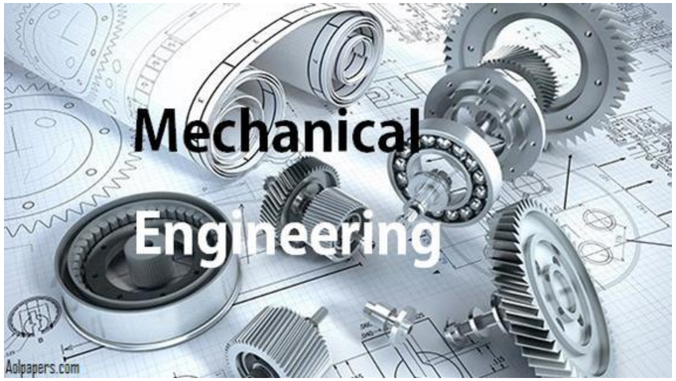
- Civil

Five Traditional Engineering Branches

- Civil

- 

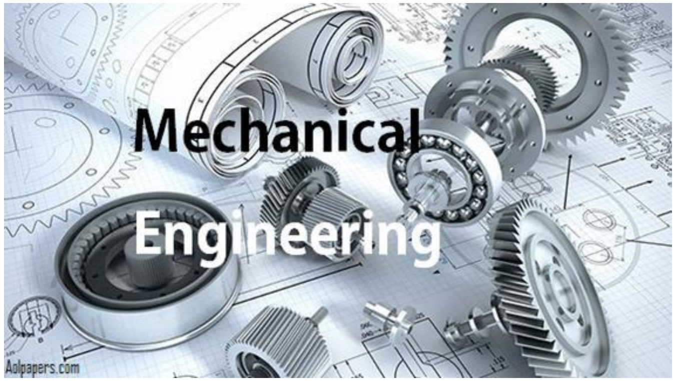
Five Traditional Engineering Branches

- Civil 
- 

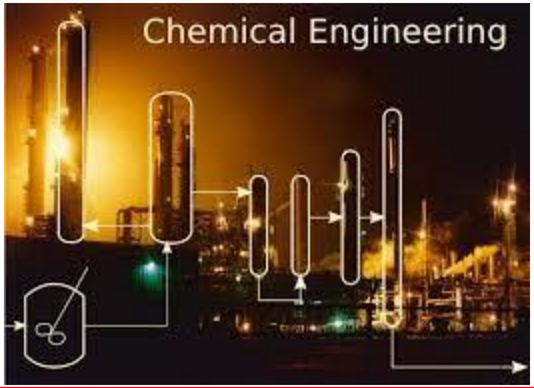
- Mining and Metallurgical 

Five Traditional Engineering Branches

- Civil

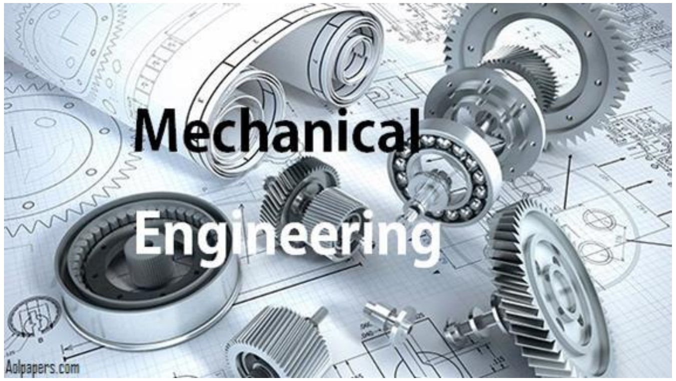
- 

- Mining and Metallurgical

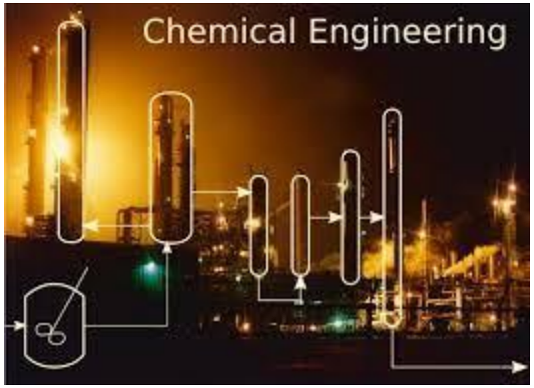
- 

Five Traditional Engineering Branches

- Civil

- 

- Mining and Metallurgical

- 

- Electrical

History of Engineering Technology

- Examples of engineering fields in 2024
  - Computer engineering
  - Electronics and communications engineering
  - Electrical engineering
  - Mechanical engineering
  - Information Technology engineering
  - Civil Engineering
  - Chemical Engineering
  - Aeronautical Engineering
  - Agricultural engineering
  - Mining engineering
  - Biochemical engineering
  - Electrical and Instrumentation Engineering
  - Metallurgical Engineering

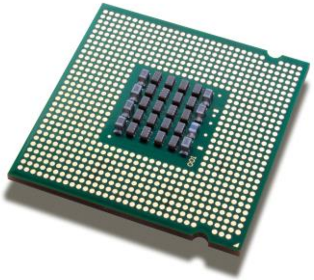
And others …… https://typesofengineeringdegrees.org/

Technology Changes Rapidly

- Hardware
  - Vacuum tubes: Electron emitting devices
  - Transistors: On-off switches controlled by electricity
  - Integrated Circuits(IC/ Chips): Combines thousands of transistors
  - Very Large-Scale Integration(VLSI): Combines millions of transistors
  - Nanotechonology → Nanoelectronics
  - What next?
- Software
  - Machine language: Zeros and ones
  - Assembly language: Mnemonics
  - High-Level Languages: English-like
  - Artificial Intelligence languages: Functions & logic predicates
  - Object-Oriented Programming: Objects & operations on objects

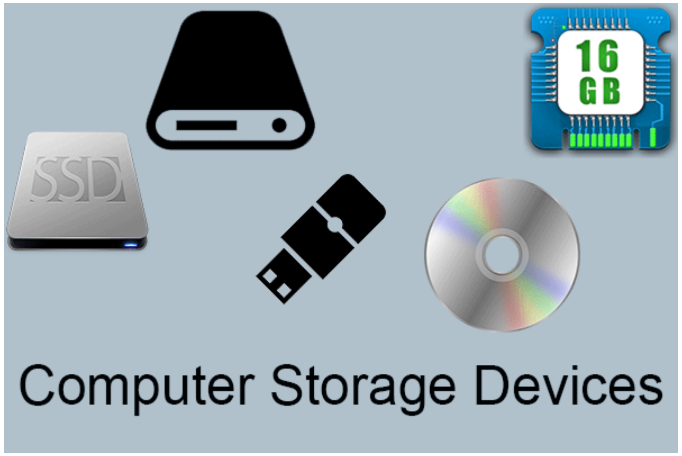
Technology Advances Rapidly

- **Processor**



- **Computer Memory**

- **Disk**



Technology Advances Rapidly

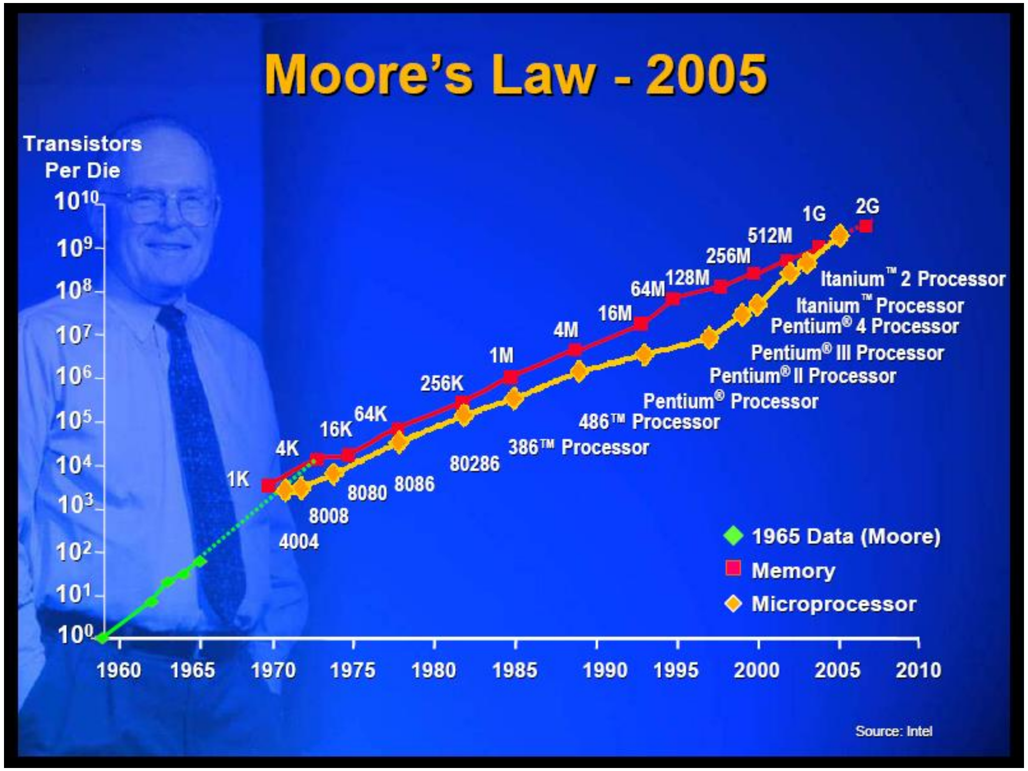
- **Processor**
  - Logic capacity: ⇑ ~ 30% / yr
  - Clock rate: ⇑ ~ 20% / yr

- **Memory**
  - DRAM capacity: ⇑ ~ 60% / yr
  - Memory speed: ⇑ ~ 10% / yr
  - Cost per bit: ⇓ ~ 25% / yr

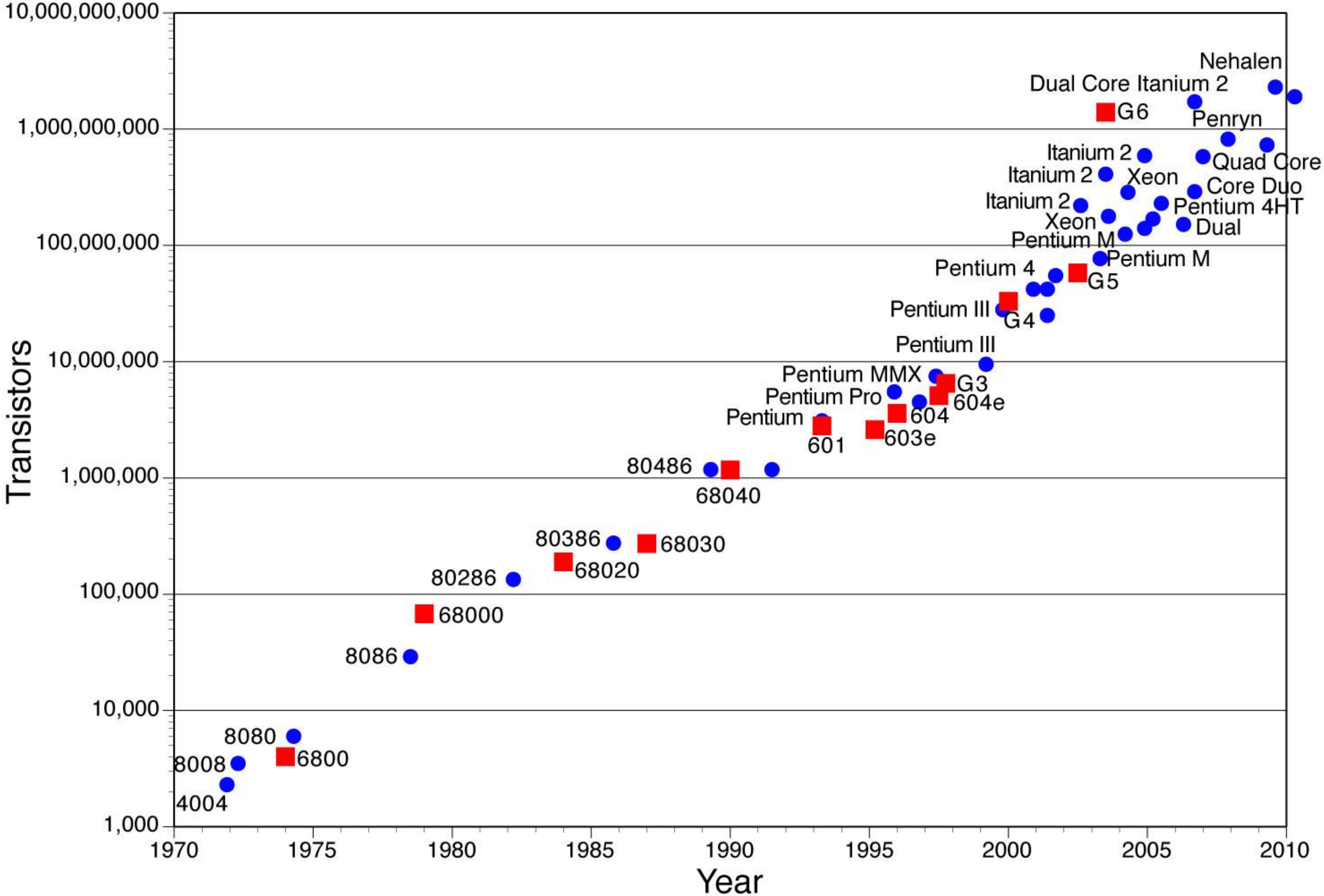
- **Disk**
  - Capacity: ⇑ ~ 60% / yr

Moore’s Law

- The **logic density** of silicon has approximately **doubled every year** since the invention of the silicon chip. This means the amount of information that can be stored on a chip of the same size doubles every year.
- Another formulation is that the **speed** of new computers **doubles every year and a half**



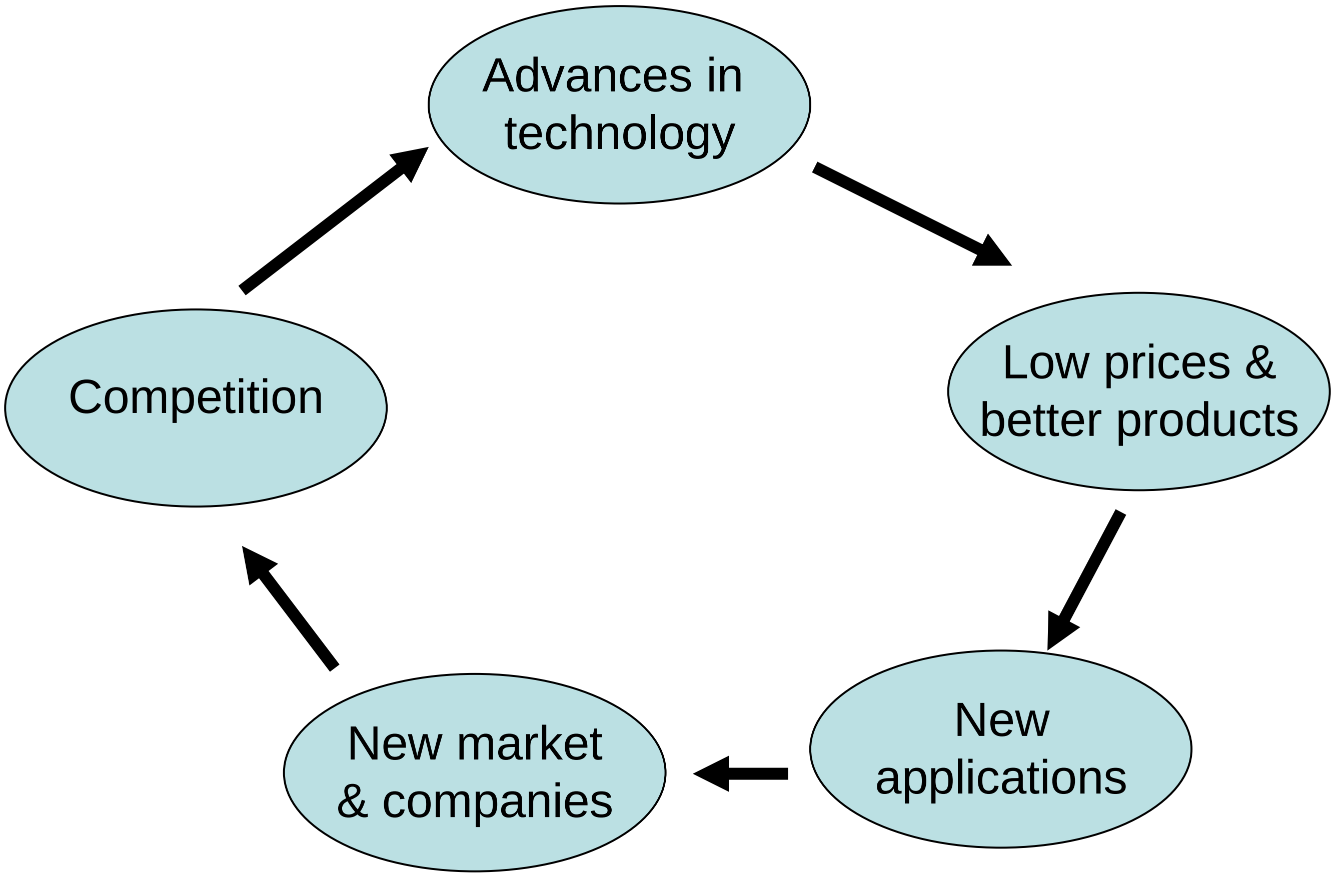
Moore's Law



10,000,000,000
1,000,000,000
100,000,000
10,000,000
1,000,000
100,000
10,000
1,000
Transistors
1970
1975
1980
1985
1990
1995
2000
2005
2010
Year
4004
8008
8080
6800
8086
68000
80286
68020
80386
68030
80486
68040
Pentium
601
603e
604
Pentium Pro
Pentium MMX
604e
G3
Pentium III
Pentium III
G4
Pentium 4
G5
Pentium M
Pentium M
Xeon
Itanium 2
Itanium 2
Itanium 2
Xeon
Dual
Pentium 4HT
Core Duo
Quad Core
Penryn
G6
Dual Core Itanium 2
Nehalen

Virtuous Circle

A result of Moore's law:

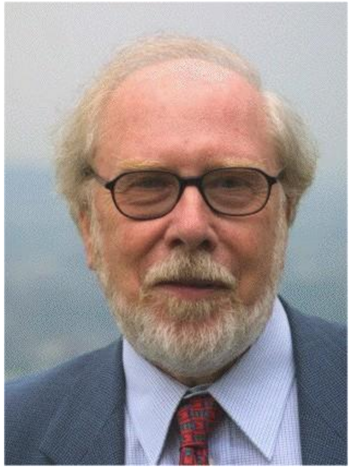


Laws of Software

- Andrew Tannenbaum:

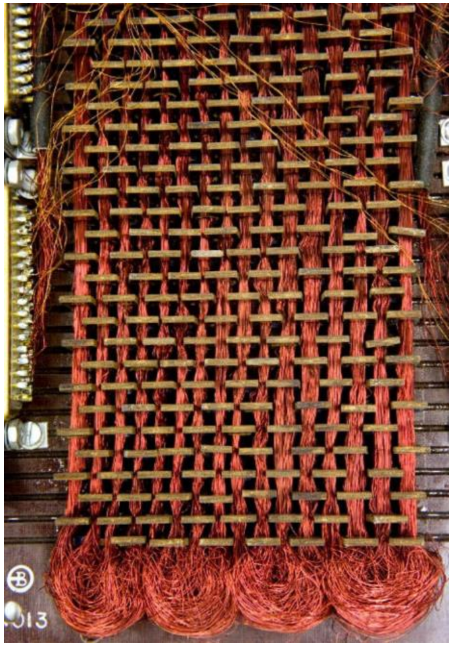
  "Software is a gas. It expands to fill the container holding it"

- Niklaus Wirth:
  - "Software gets slower faster than hardware gets faster"

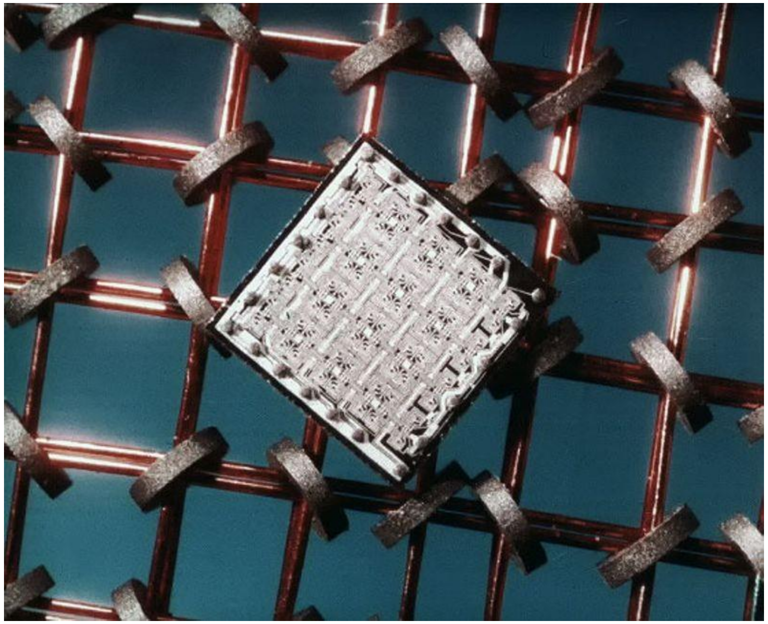


Program Performance

- Performance in the 1970's:
  - Minimize memory space to make programs fast



1969
Apollo guidance computer
read-only rope memory



1970
First IBM computer to use
Semiconductor memory

https://www.computerhistory.org/timeline/memory-storage/

Program Performance

- Performance in the 1970's:
  - Minimize memory space to make programs fast
- Performance now (2022):
  - Performance depend on efficient algorithms, compilers, & computer hardware
    - Memory in hierarchical structure (Cache, RAM, permanent storage)
    - Parallel processors
    - Programmers need to more knowledge of computer organization

Program Performance

Component	Effect on performance	Where is this covered
Algorithm	Determines number of source code statements & I/O operations	COMP 103
Programming language, Compilers, & Architecture	Determine number of machine instructions	COMP 102, NWEN 241
Processor & memory	Determine how fast instructions can execute	NWEN 241
I/O system (HW & OS)	Determines how fast I/O operations may be executed	ENGR 101 NWEN 241

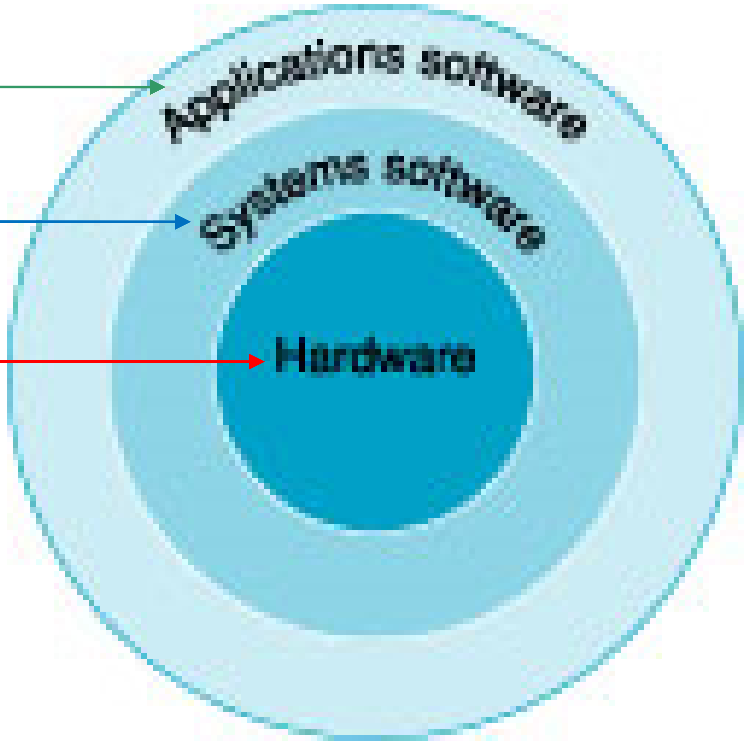
Computer Software (apps)

- Several software layers are organized in hierarchical fashion
  - In complex applications there could be multiple layers of application software

Eg. Wechat or MS Word

Eg. Android or MS Windows 10

Eg. Smartphone or Laptop



Language Evolution

- Machine language
- Assembly language
- High-level languages
- Subroutine libraries
- There is a large gap between what is convenient for computers & what is convenient for humans
- Translation/Interpretation is needed between humans and machines.

Language Evolution

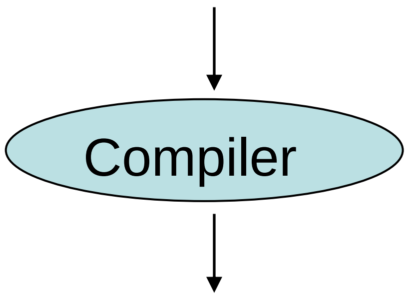
High-level
Language
Program
(in C)

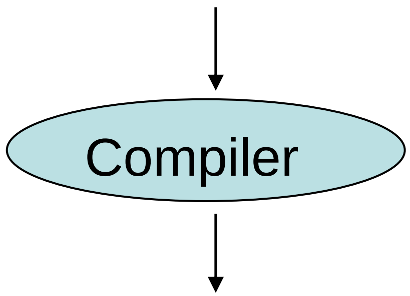
```
swap(int v[ ], int k)
{ int temp;
     temp = v[k];
     v[k] = v[k+1];
     v[k+1] = temp;
}
```

Language Evolution

High-level
language
program
(in C)

```
swap(int v[ ], int k)
{ int temp;
    temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```



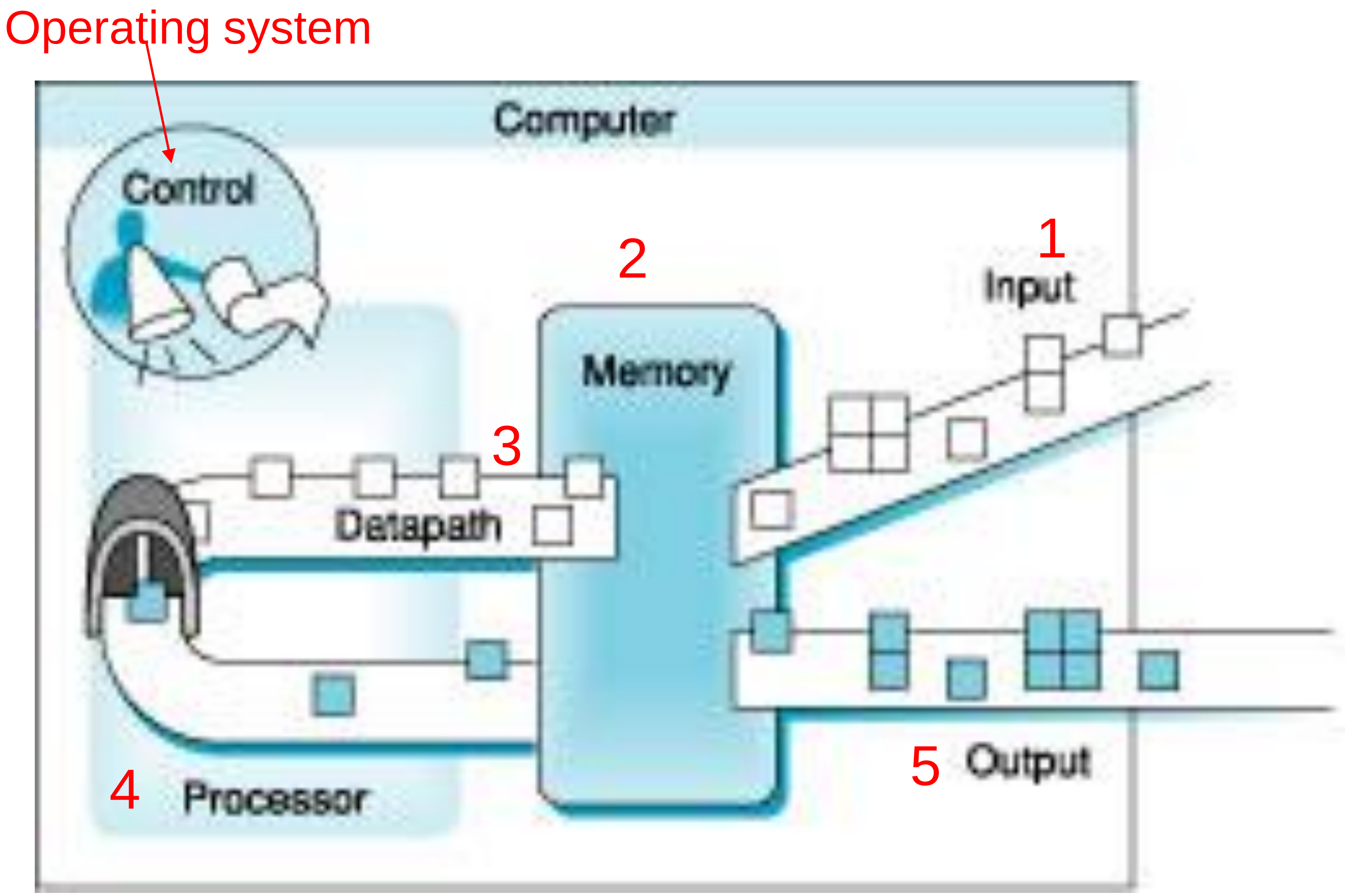


Assembly
language
program
(for MIPS)

```
Swap:
        nul1 $2, $5, 4
        add $2, $4, $2
        lw   $15, 0($2)
        lw   $16, 4($2)
        sw   $16, 0($2)
        sw   $15, 4($2)
        jr   $31
```

Language Evolution

Computer Components



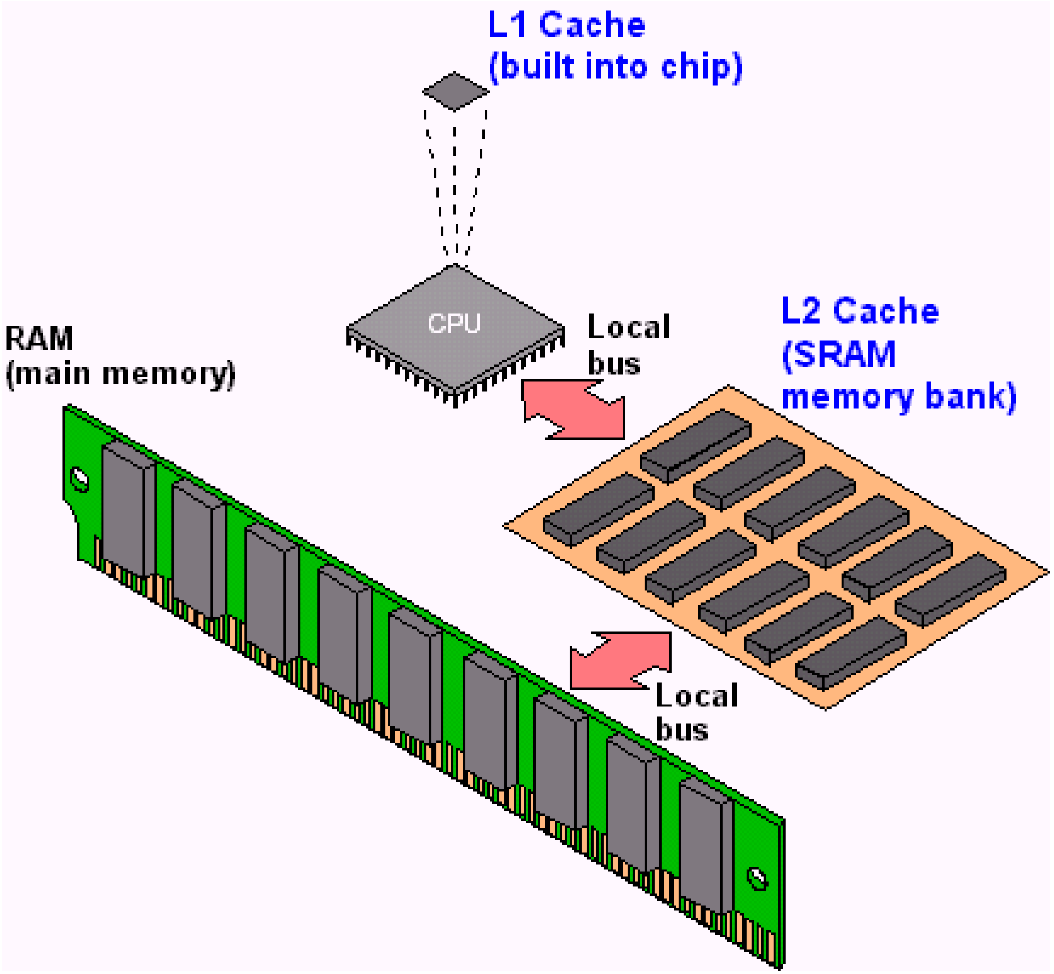
Memory Categories

- Volatile memory
  – Loses information when power is switched-off
  – Random Access Memory (RAM)

- Non-volatile memory
  – Keeps information when power is switched-off
  – Optical & magnetic disks
  – Magnetic tape

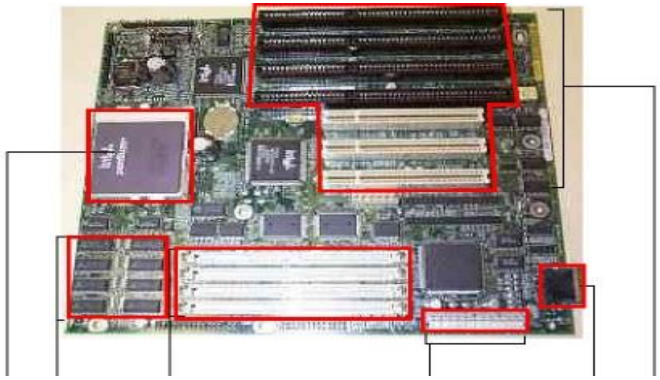
Volatile Memory Types

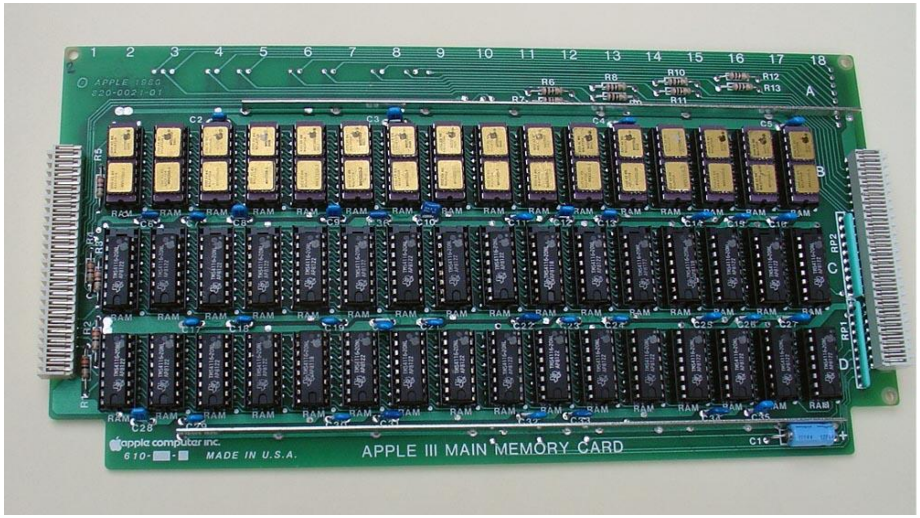
- Cache:
  - Fast but expensive
  - Smaller capacity
  - Placed closer to the processor



Volatile Memory Types

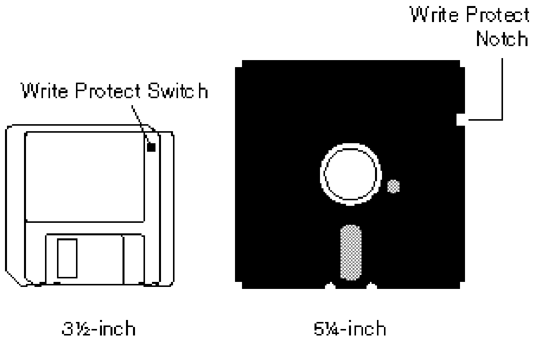
- Cache:
  - Fast but expensive
  - Smaller capacity
  - Placed closer to the processor
- Main memory
  - Less expensive
  - More capacity
  - Slower

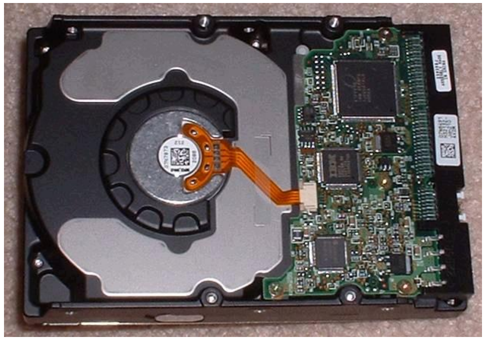




Non-volatile Memory Types

- **Secondary memory**
  - Low cost
  - Very slow
  - Unlimited capacity
- <u>Types</u>
  - Diskettes
  - CD-ROMS
  - Hard disk
  - Flash Drives / SSD
  - Who knows what comes next??



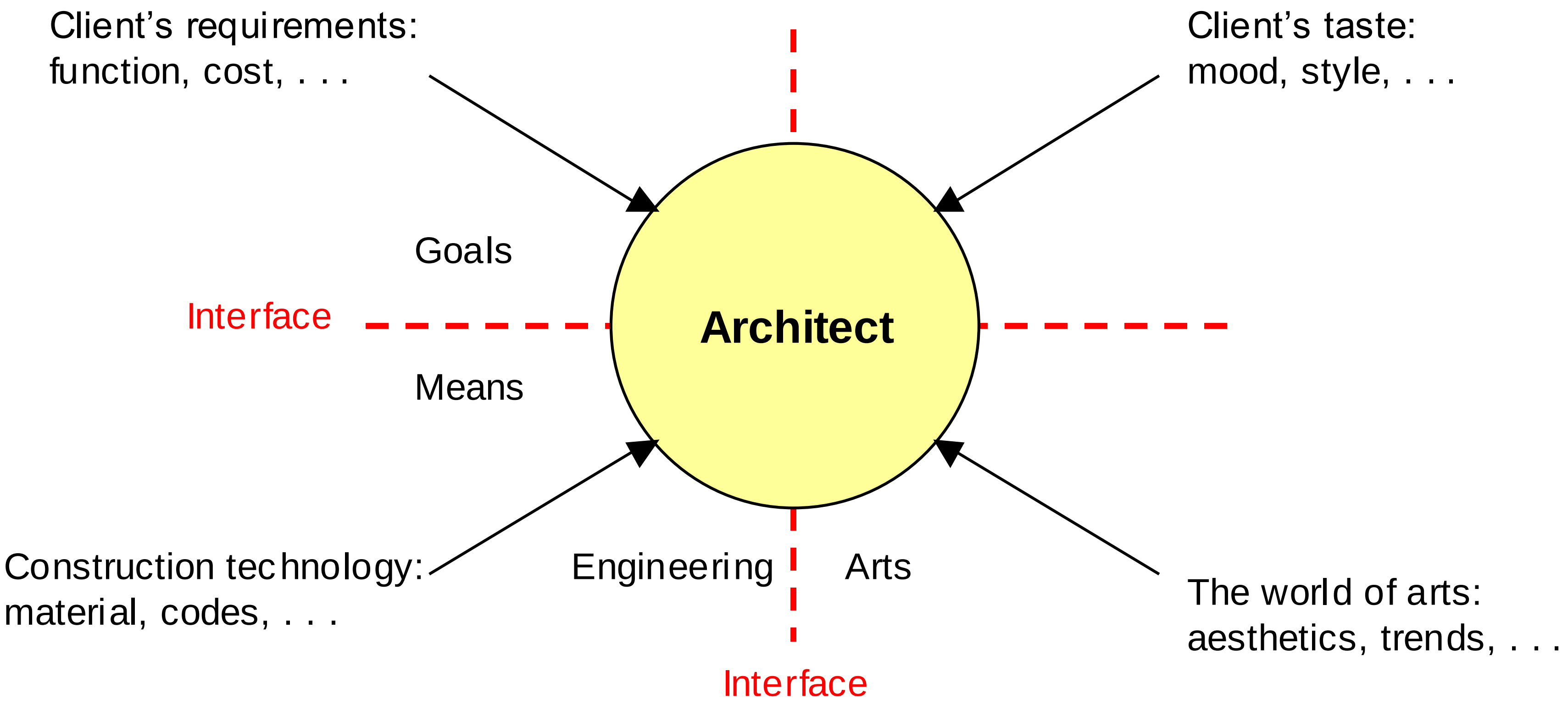


Input-Output (I/O)

- I/O devices have the hardest organization
  - Wide range of speeds
    - Graphics vs. keyboard
  - Wide range of requirements
    - Speed
    - Standard
    - Cost . . .
  - Least amount of research done in this area

What is Computer Architecture?

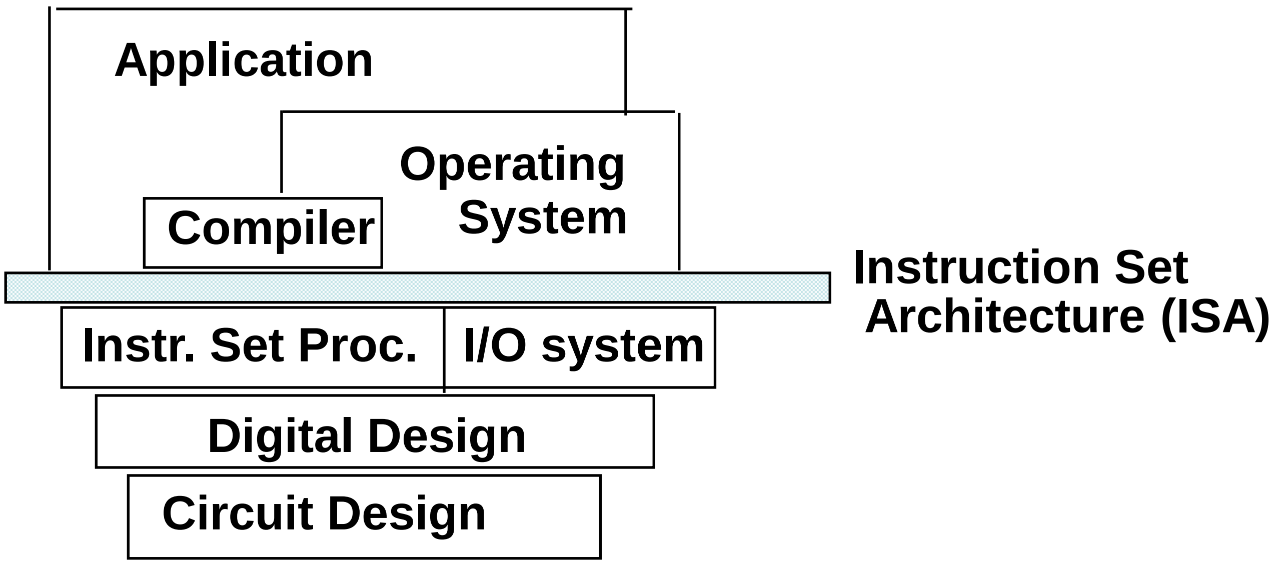
Like a building architect, whose place at the engineering/arts and goals/means interfaces is seen in this diagram, a computer architect reconciles many conflicting or competing demands.



Computer Architecture

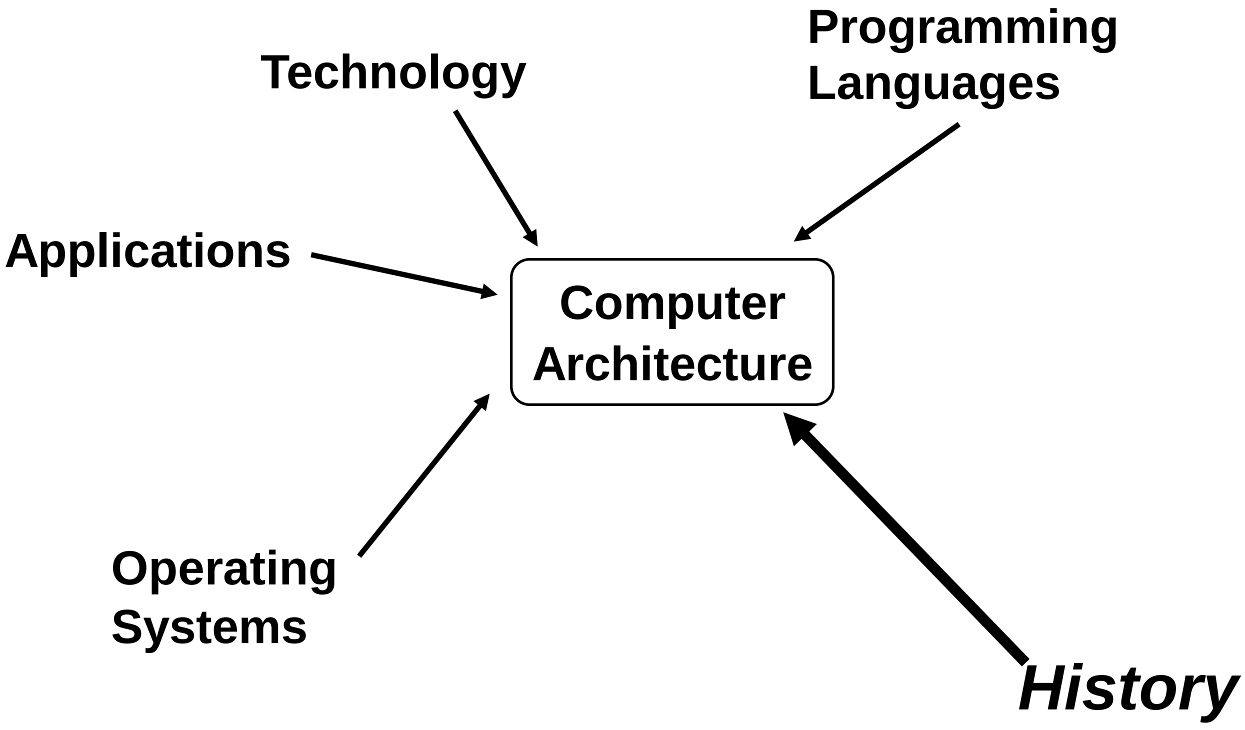
A system concept integrating software, hardware, and firmware to specify the design of computing systems

- **Co-ordination of *levels of abstraction***

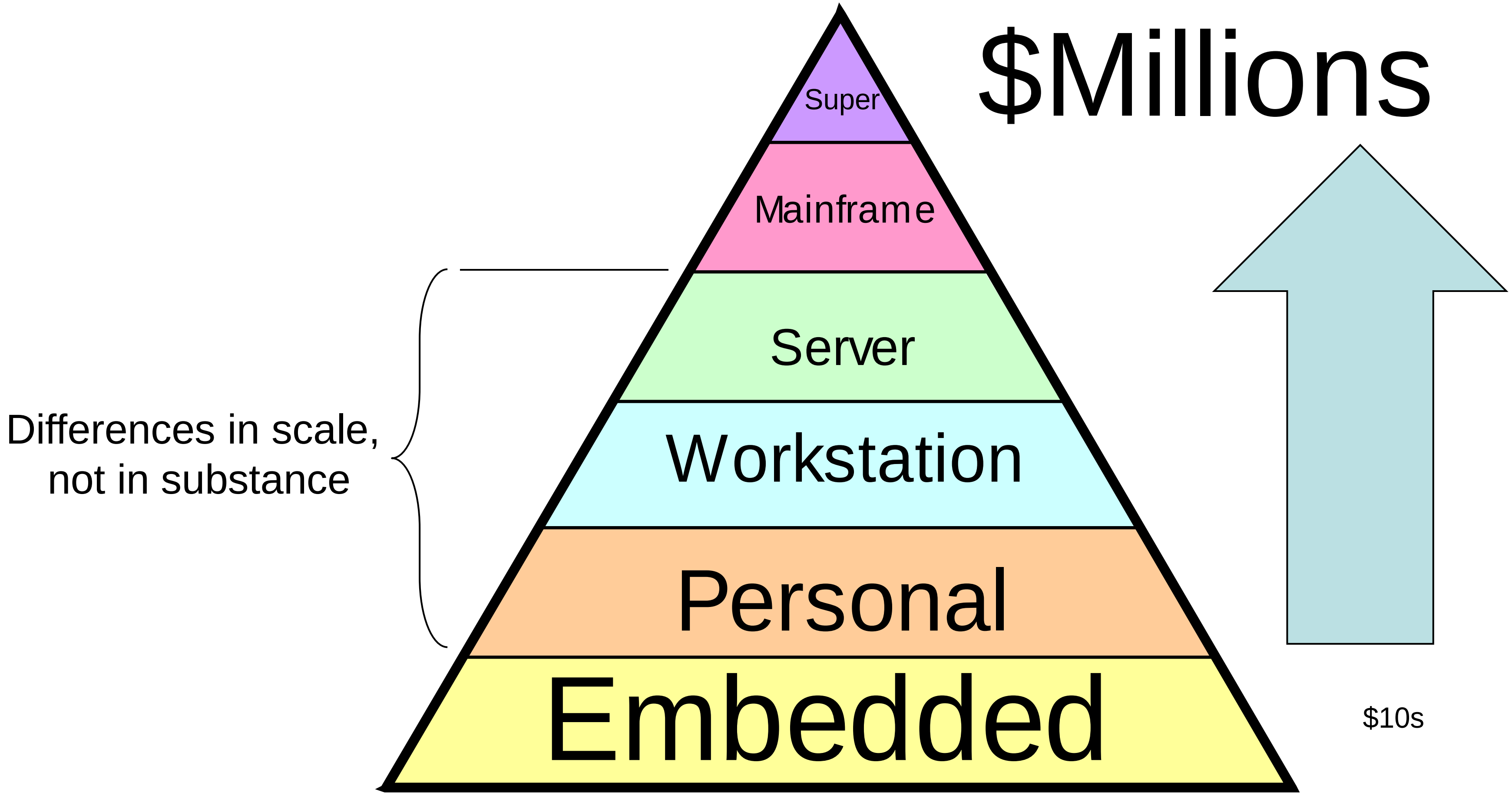


- **Under a set of rapidly changing *Forces***

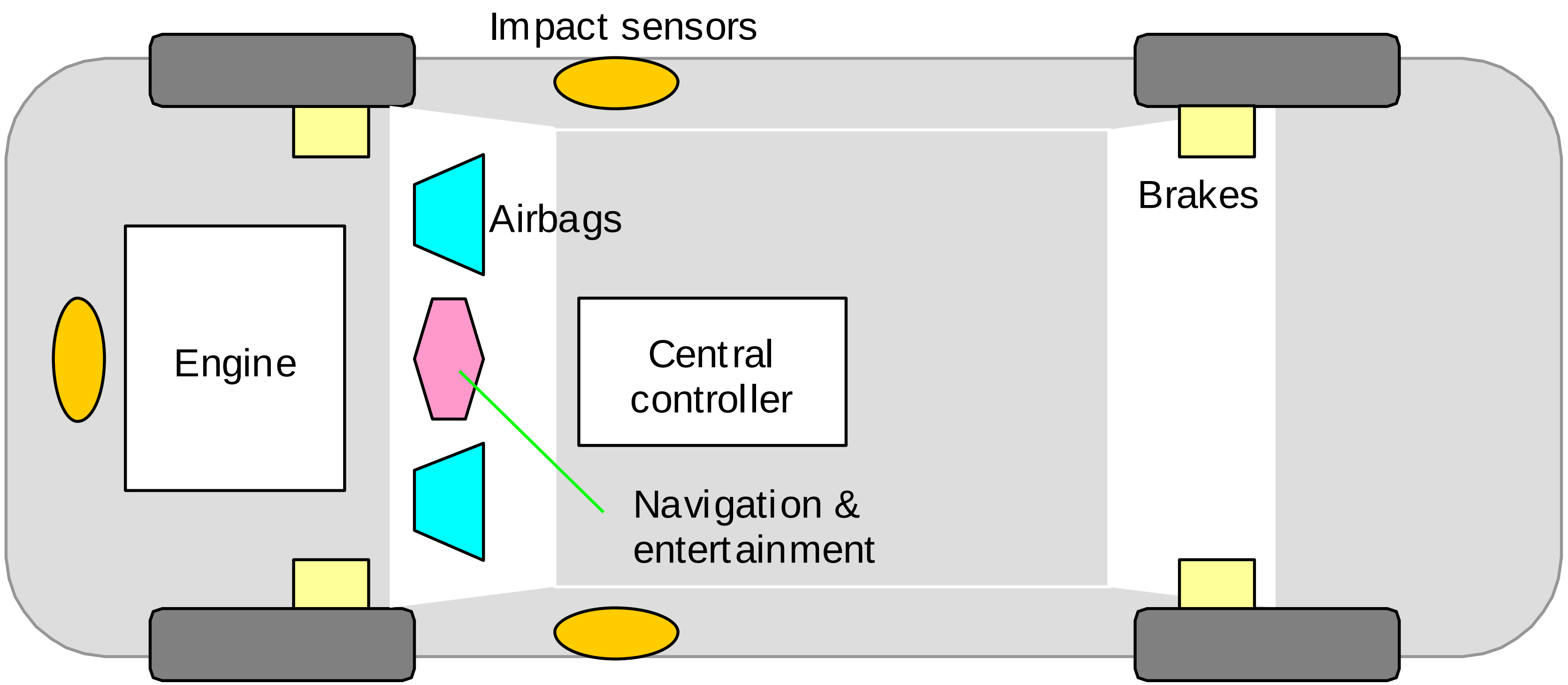
Forces on Computer Architecture



Computer Price/Performance Pyramid

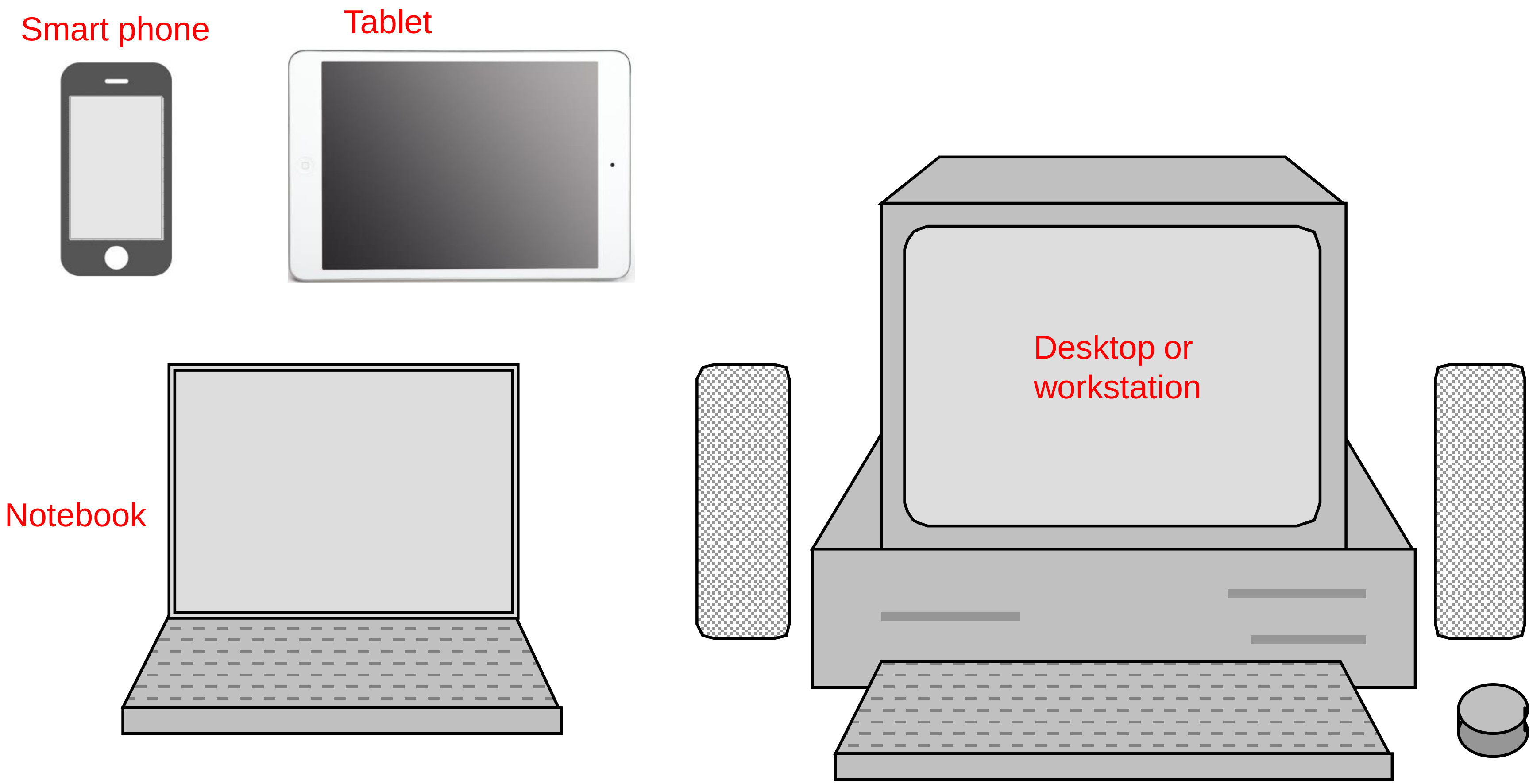


Automotive Embedded Computers



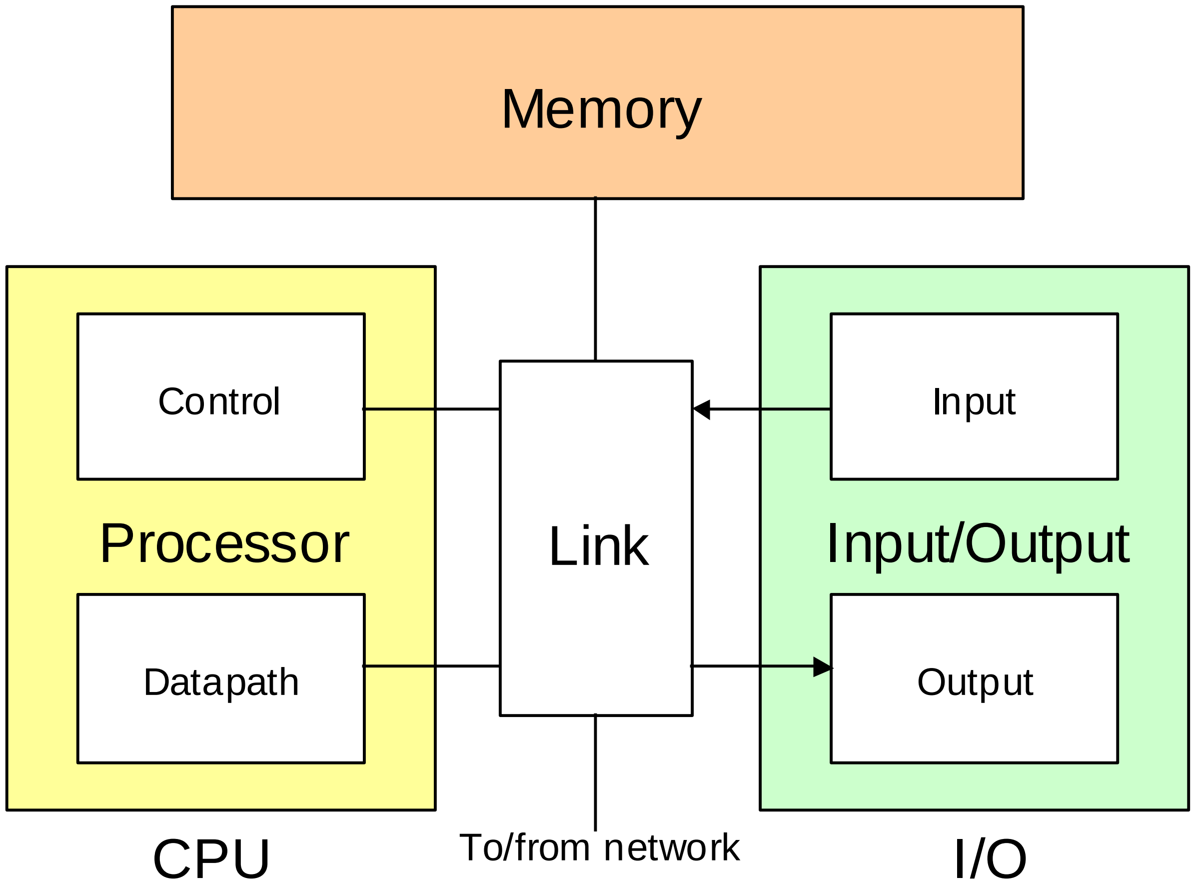
Embedded computers are ubiquitous, yet invisible. They are found in automobiles, home appliances, and many other places.

Personal Computers and Workstations



Notebooks, a common class of portable computers, are much smaller than desktops but offer substantially the same capabilities.

Digital Computer Subsystems

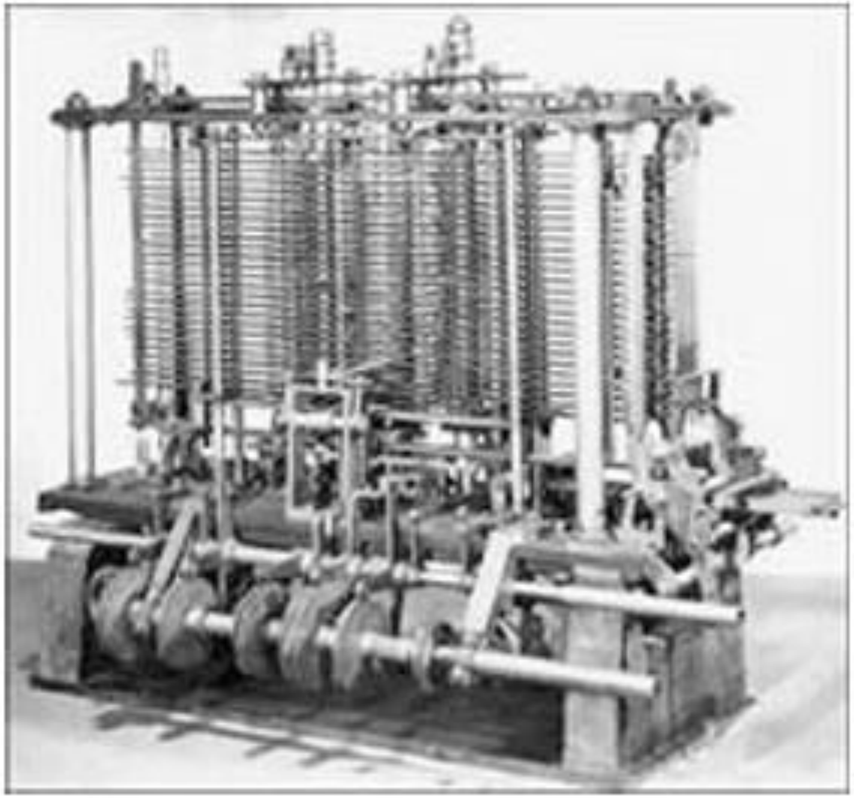


The six main units of a digital computer are the CPU, which comprised of the control unit and the datapath, memory, the I/O devices which are linked together by a simple bus or a more elaborate network.

Generations of Progress

The 5 generations of digital computers.

Generation (Year)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment



Generations of Progress

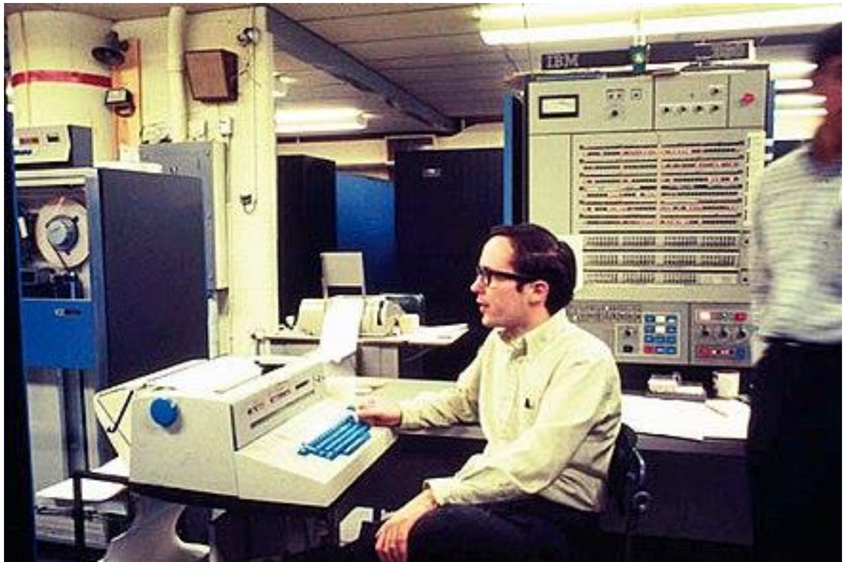
The 5 generations of digital computers, and their ancestors.

Generation (Year)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet

Generations of Progress

The 5 generations of digital computers, and their ancestors.

Generation (Year)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet
2 (1960s)	Transistor	Magnetic core	Drum, printer, text terminal	Room-size mainframe



Generations of Progress

The 5 generations of digital computers, and their ancestors.

Generation (Year)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet
2 (1960s)	Transistor	Magnetic core	Drum, printer, text terminal	Room-size mainframe
3 (1970s)	SSI/MSI	RAM/ROM chip	Disk, keyboard, video monitor	Desk-size mini

SSI – Small Scale Integration { logic gates: AND. OR, NAND, NOR; 1-10 / chip}

MSI – Medium Scale Integration { Flip-flops, adders/counters, MUX & DEMUX

Generations of Progress

The 5 generations of digital computers, and their ancestors.

Generation (Year)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet
2 (1960s)	Transistor	Magnetic core	Drum, printer, text terminal	Room-size mainframe
3 (1970s)	SSI/MSI	RAM/ROM chip	Disk, keyboard, video monitor	Desk-size mini
4 (1980s)	LSI/VLSI	SRAM/DRAM	Network, CD, mouse,sound	Desktop/ laptop micro

LSI – Large Scale Integration { 500 – 20,000 transistors/chip}

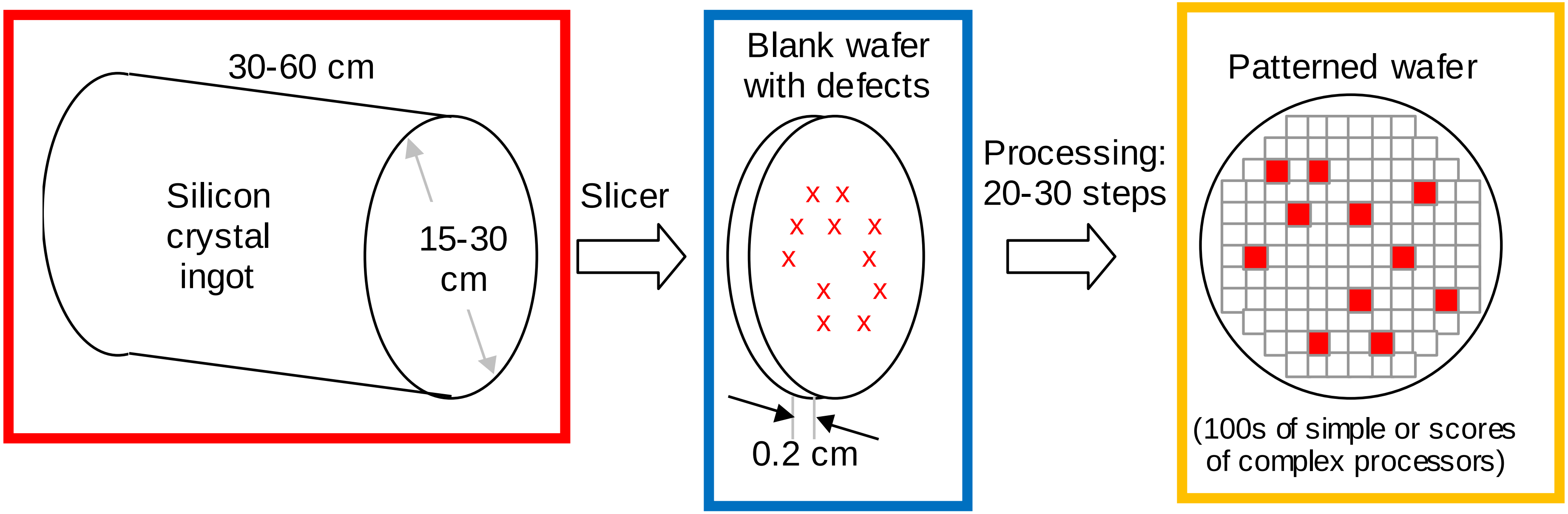
VLSI – Very Large Scale Integration { 20,000 – 1,000,000,000 transistors/chip

Generations of Progress

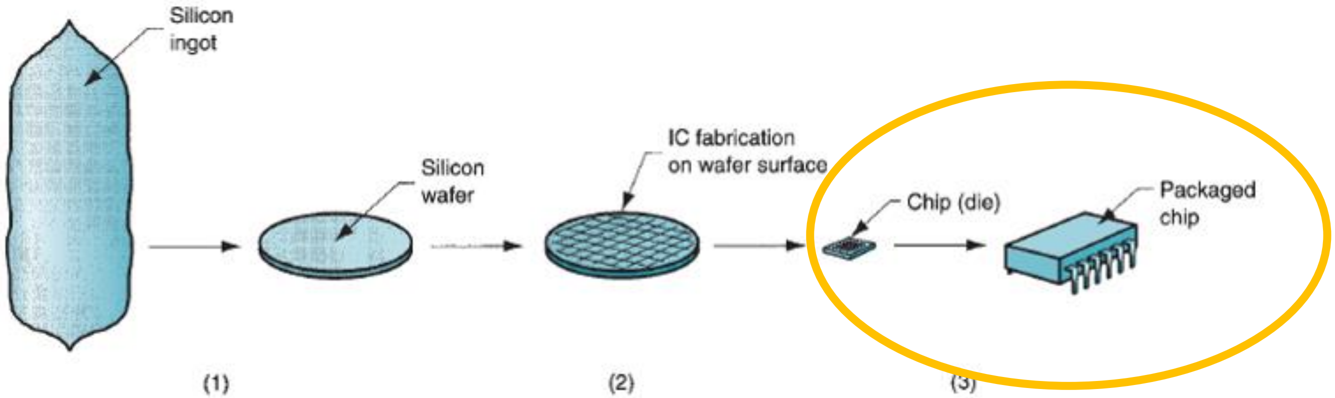
The 5 generations of digital computers, and their ancestors.

Generation (Year)	Processor technology	Memory innovations	I/O devices introduced	Dominant look & feel
0 (1600s)	(Electro-) mechanical	Wheel, card	Lever, dial, punched card	Factory equipment
1 (1950s)	Vacuum tube	Magnetic drum	Paper tape, magnetic tape	Hall-size cabinet
2 (1960s)	Transistor	Magnetic core	Drum, printer, text terminal	Room-size mainframe
3 (1970s)	SSI/MSI	RAM/ROM chip	Disk, keyboard, video monitor	Desk-size mini
4 (1980s)	LSI/VLSI	SRAM/DRAM	Network, CD, mouse,sound	Desktop/ laptop micro
5 (1990s)	ULSI/GSI/ WSI, SOC	SDRAM, flash	Sensor/actuator, point/click	Invisible, embedded

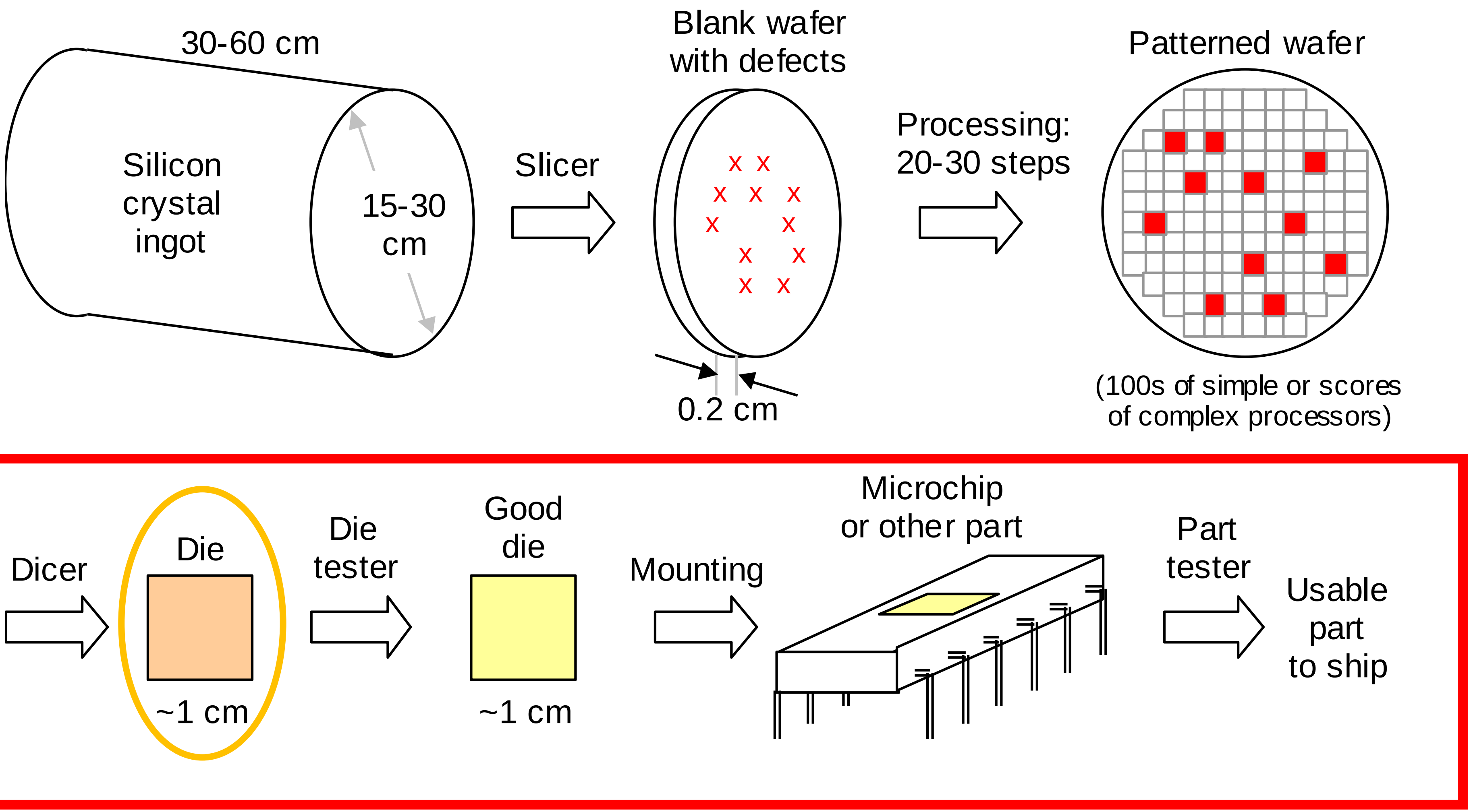
IC Production and Yield



The manufacturing process for an IC part.

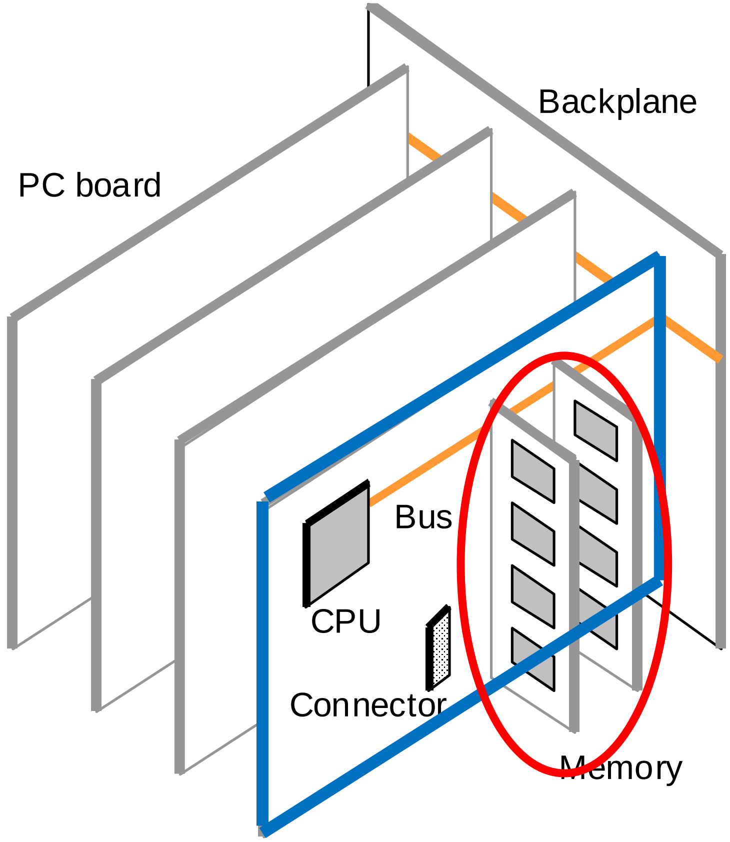


IC Production and Yield



The manufacturing process for an IC part.

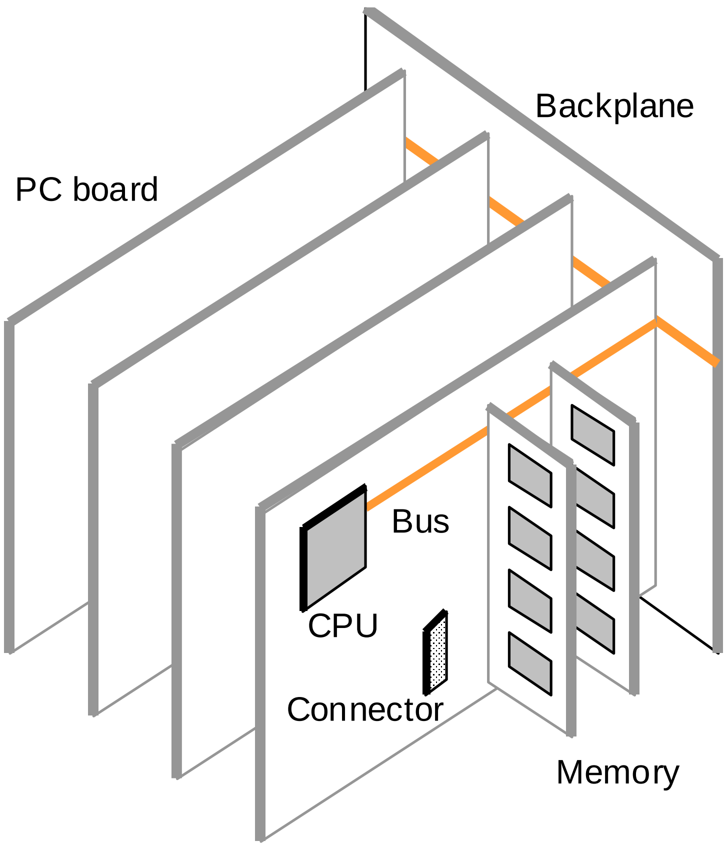
Processor and Memory Technologies



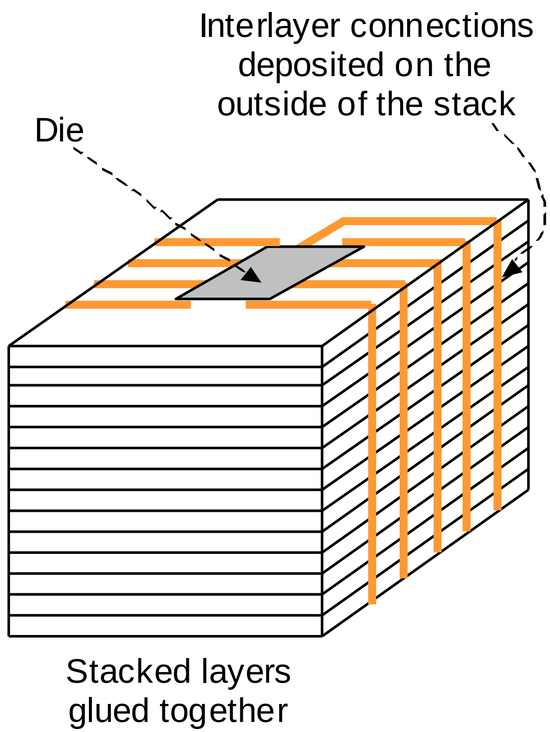
(a) 2D or 2.5D packaging now common

Packaging of processor, memory, and other components.

Processor and Memory Technologies



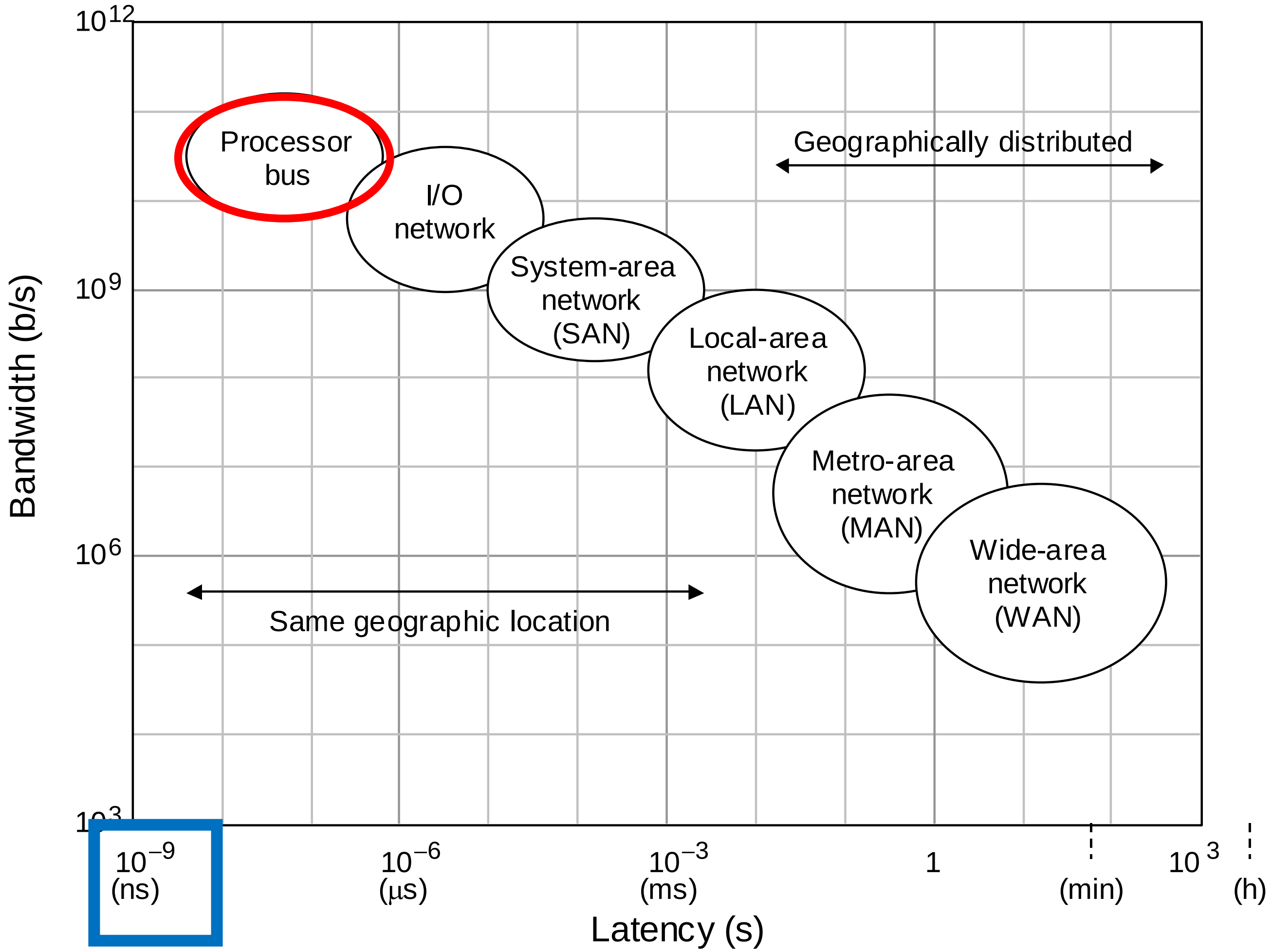
(a) 2D or 2.5D packaging now common



(b) 3D packaging of the future

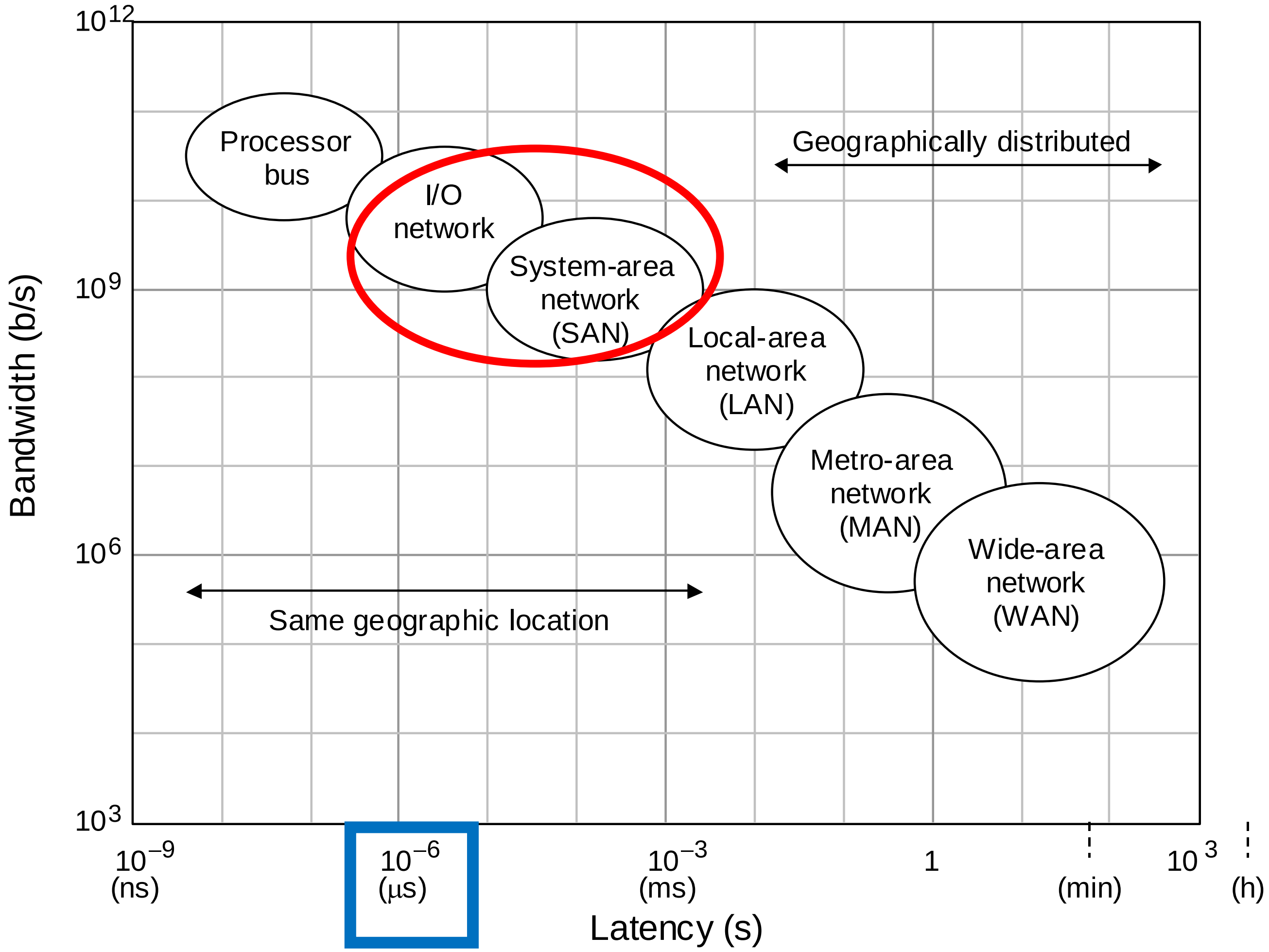
Packaging of processor, memory, and other components.

Communication Technologies



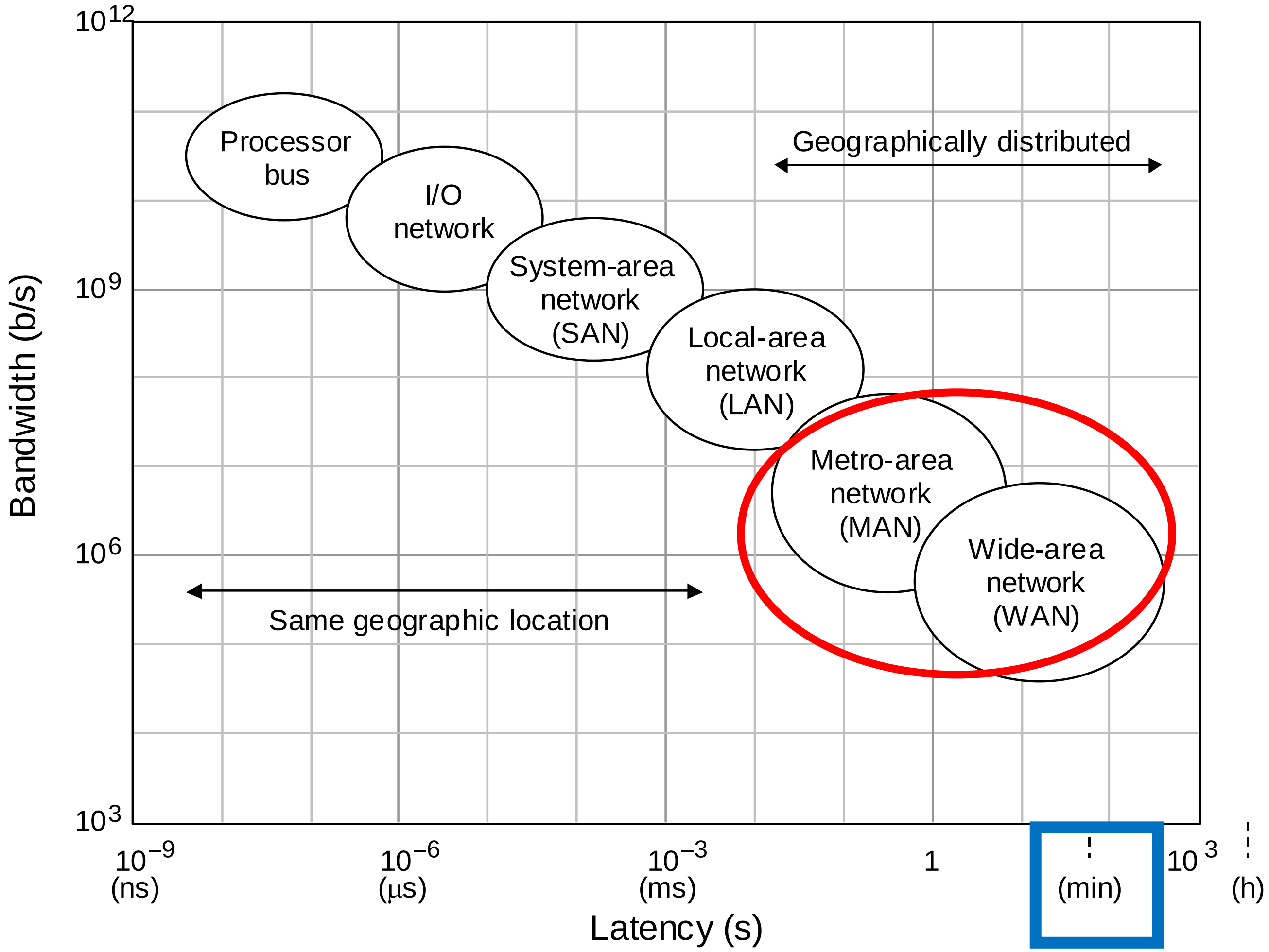
Latency and bandwidth characteristics of different classes of communication links.

Communication Technologies



Latency and bandwidth characteristics of different classes of communication links.

Communication Technologies



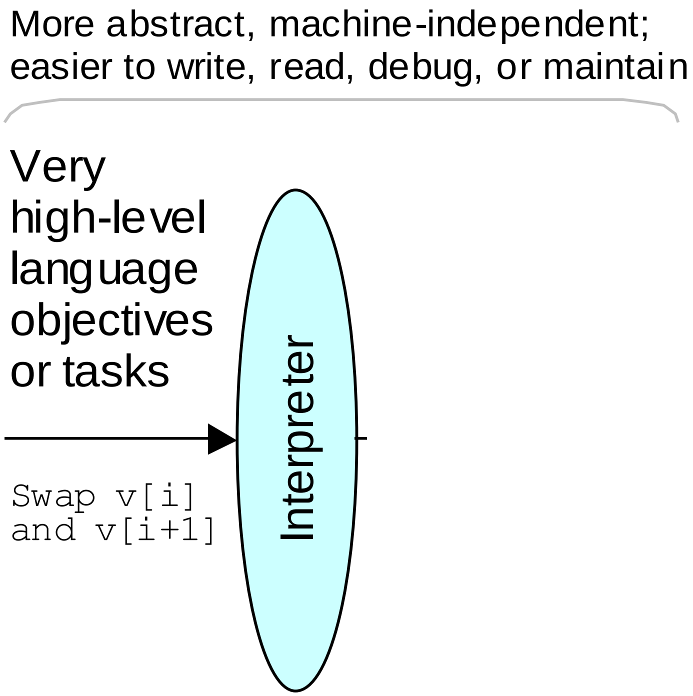
Latency and bandwidth characteristics of different classes of communication links.

High- vs Low-Level Programming

More abstract, machine-independent;
easier to write, read, debug, or maintain

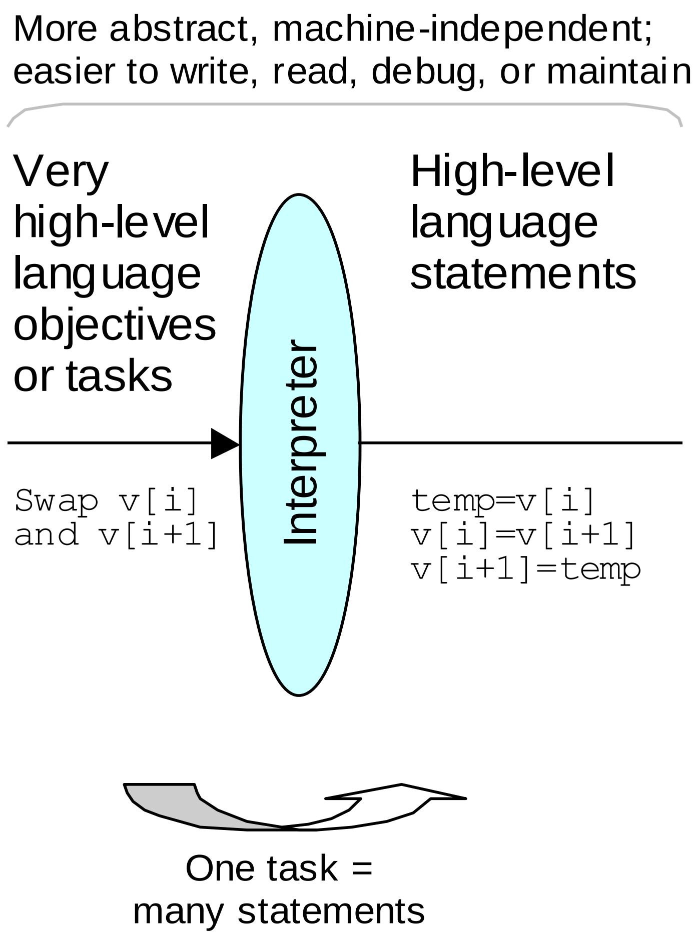
Models and abstractions in programming.

High- vs Low-Level Programming



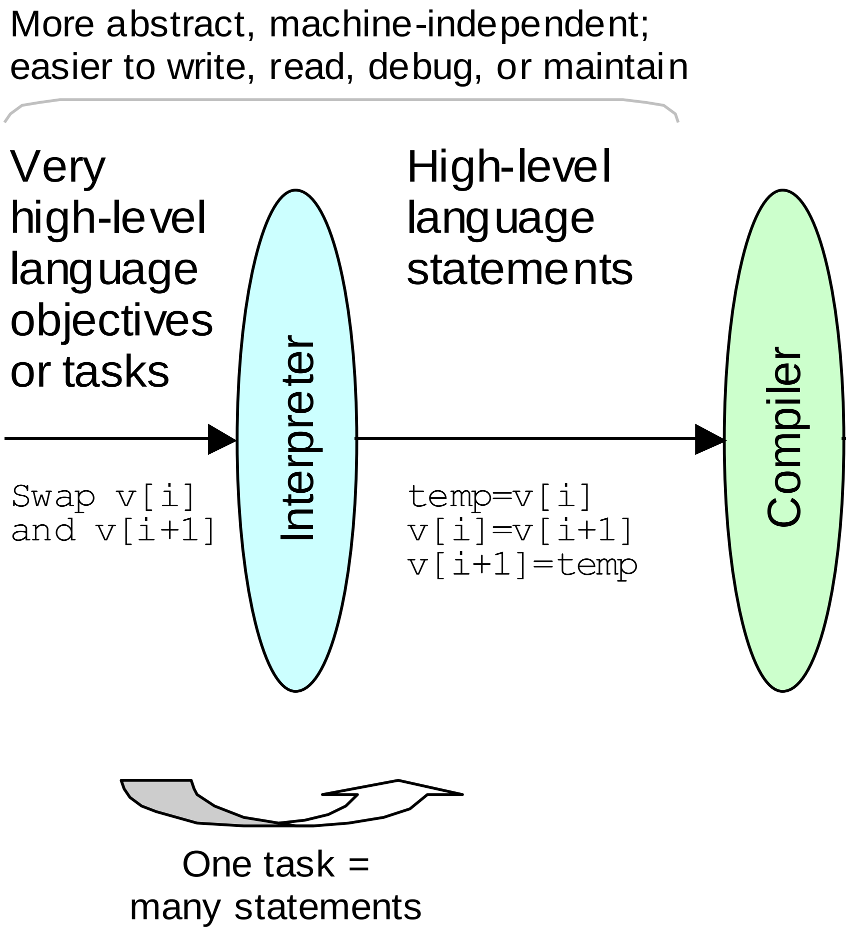
Models and abstractions in programming.

High- vs Low-Level Programming



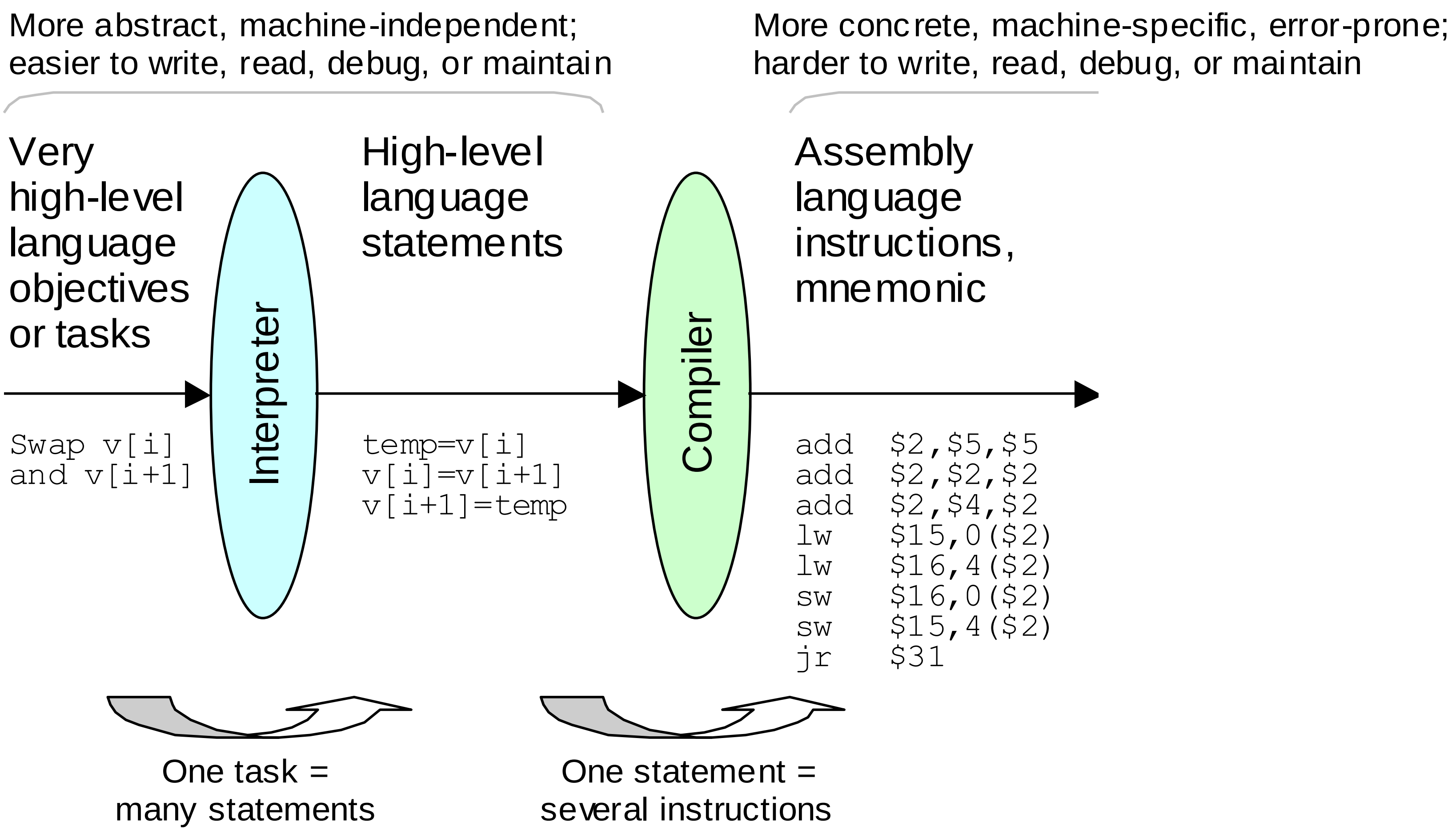
Models and abstractions in programming.

High- vs Low-Level Programming



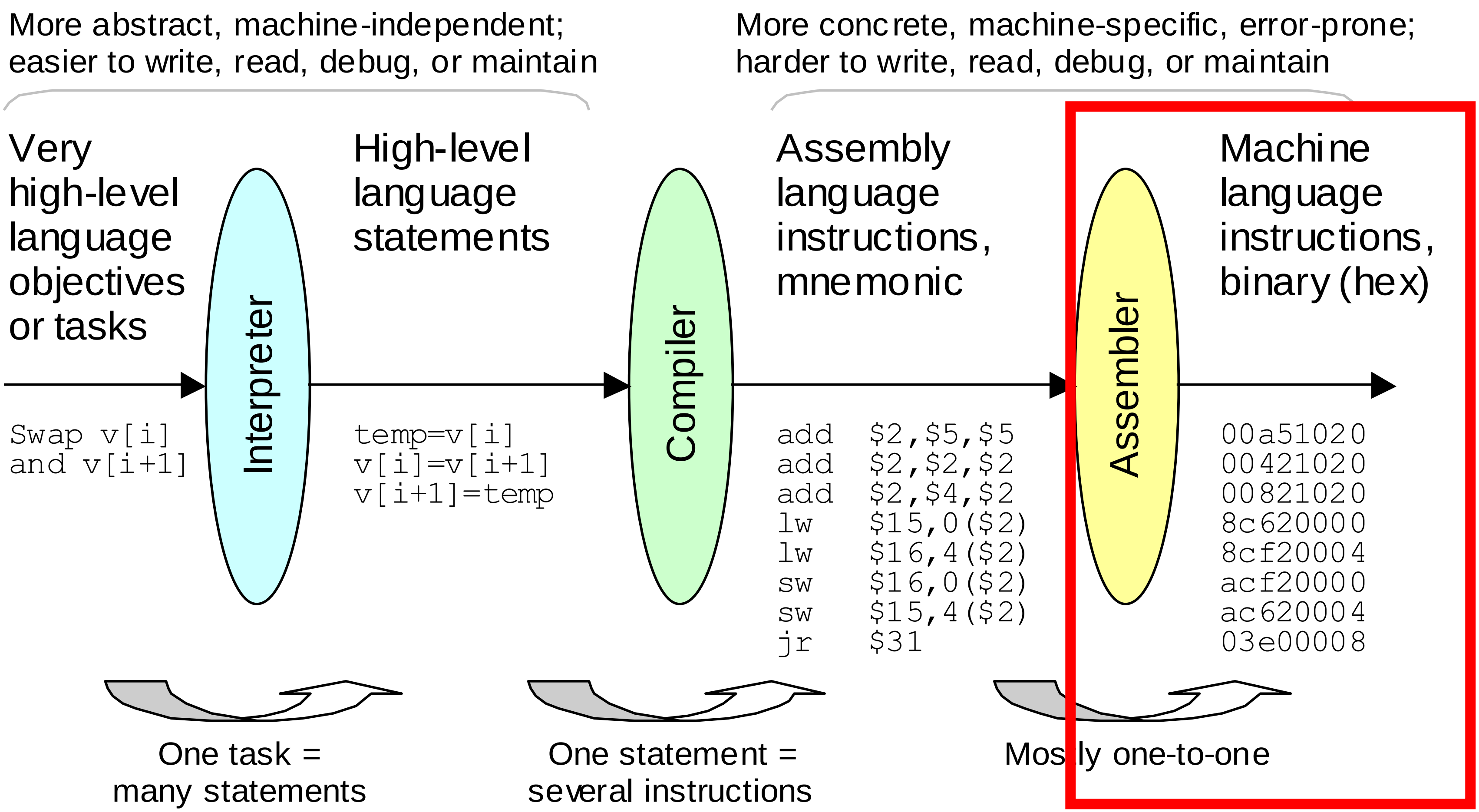
Models and abstractions in programming.

High- vs Low-Level Programming



Models and abstractions in programming.

High- vs Low-Level Programming



Models and abstractions in programming.

Concepts of Performance and Speedup

Performance = 1 / Execution time

is simplified to

Performance = 1 / CPU execution time

Concepts of Performance and Speedup

Performance = 1 / Execution time — is simplified to

Performance = 1 / CPU execution time

(Performance of M_1) / (Performance of M_2) = Speedup of M_1 over M_2
= (Execution time of M_2) / (Execution time M_1)

Terminology: M_1 is *x* times **as fast as** M_2 (e.g., 1.5 times as fast)
M_1 is 100(*x* – 1)% **faster than** M_2 (e.g., 50% faster)

Concepts of Performance and Speedup

Performance = 1 / Execution time

is simplified to

Performance = 1 / CPU execution time

(Performance of M_1) / (Performance of M_2) = Speedup of M_1 over M_2

= (Execution time of M_2) / (Execution time M_1)

Terminology: M_1 is x times **as fast as** M_2 (e.g., 1.5 times as fast)

M_1 is $100(x - 1)\%$ **faster than** M_2 (e.g., 50% faster)

CPU time = Instructions $\times$ (Cycles per instruction) $\times$ (Secs per cycle)

= Instructions $\times$ CPI / (Clock rate)

Instruction count, CPI, and clock rate are not completely independent, so improving one by a given factor may not lead to overall execution time improvement by the same factor.

Elaboration on the CPU Time Formula

CPU time = Instructions × (Cycles per instruction) × (Secs per cycle)
= Instructions × Average CPI / (Clock rate)

Instructions: Number of instructions executed, not number of instructions in a program (dynamic count)

Elaboration on the CPU Time Formula

CPU time = Instructions × (Cycles per instruction) × (Secs per cycle)

= Instructions × Average CPI / (Clock rate)

Instructions: Number of instructions executed, not number of instructions in our program (dynamic count)

Average CPI: Is calculated based on the dynamic instruction mix and knowledge of how many clock cycles are needed to execute various instructions (or instruction classes)

Week 9 Lecture 1b

- Introduction to Engineering Technology
  - Computer architecture
  - Computer prices and performance pyramid
  - Generational progress
  - IC manufacturing process
  - Computer languages, performance and speed