Computer System Organization (CS2.201)

Introduction to Computer Systems

Avinash Sharma

Center for Visual Information Technology (CVIT),

IIIT Hyderabad

Slide content acknowledgment: Dr. Suresh Purini & other public sources

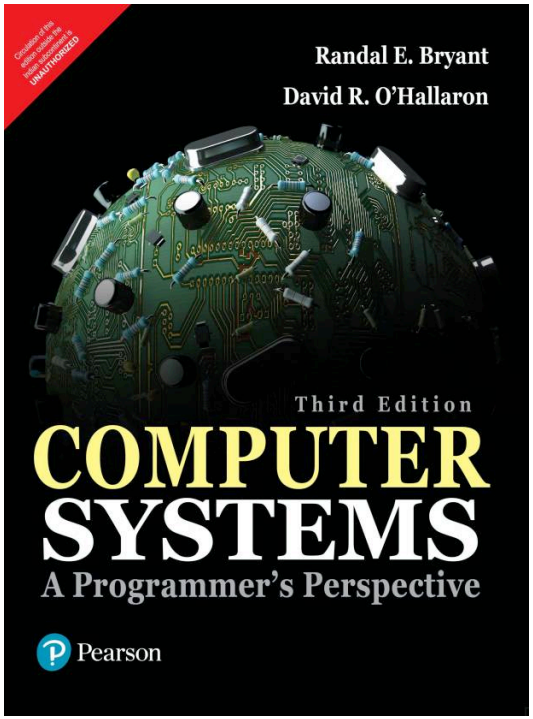
Goal

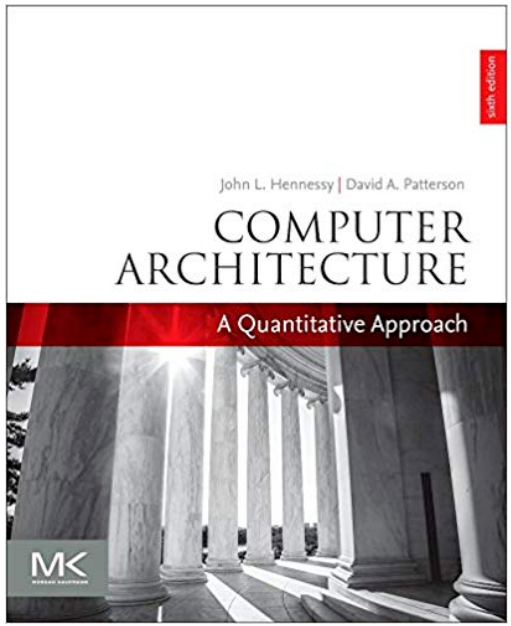
- To study the anatomy of a typical Computer System.
- Examples of Computer Systems
  - Dekstop/Laptop/Notebook
  - Server Machines
  - Embedded Computers
    - Cellphones, Airplanes/Automobiles, Television/Set-top Boxes, etc.

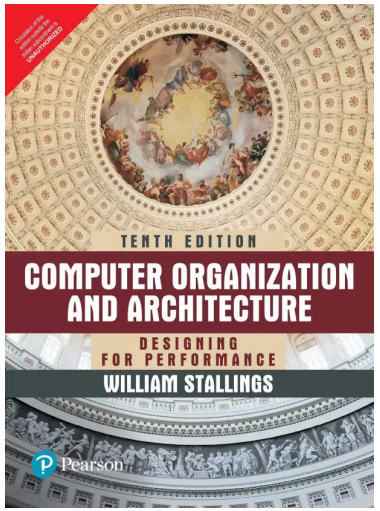
Course Outline

- Computer Arithmetic
- Instruction Set Architecture
- Processor architecture and design
- Memory Hierarchies
- Input/Output
- Virtual Memory

Reference Books







Administrivia

Lecture	Date	Topic	ASSIG. / TUTO.
01	24/05/2021 Mon	Introduction	
02	26/05/2021 Wed	Computer Arithmetic (2.2)	
03	28/05/2021 Fri	Computer Arithmetic (2.2,2.3)	
04	31/05/2021 Mon	Computer Arithmetic (2.3)	
05	02/06/2021 Wed	Intro to ISA x86_64 (Tutorial)	
06	04/06/2021 Fri	Assembly language programming 3.4/ ISA	HW 1
07	07/06/2021 Mon	Assembly language programming 3.4/ ISA	
08	09/06/2021 Wed	Assembly language programming 3.5	
09	11/06/2021 Fri	Assembly language programming 3.5	
10	14/06/2021 Mon	Assembly language programming 3.6	HW 2
11	16/06/2021 Wed	Assembly language programming 3.6	Quiz 1
12	18/06/2021 Fri	Assembly language programming 3.7	
13	21/06/2021 Mon	Assembly language programming 3.7	
14	23/06/2021 Wed	Processor architecture and design (4.1) Y86 ISA	HW 3
15	25/06/2021 Fri	Processor architecture and design (4.1) Y86 ISA	
16	28/06/2021 Mon	Processor architecture and design (4.3) Y86 ISA	
17	30/06/2021 Wed	Processor architecture and design (4.3) Y86 ISA	
18	02/07/2021 Fri	Processor architecture and design (4.5) Y86 ISA	HW 4
19	03/07/2021 Sat	Processor architecture and design (4.5) Y86 ISA	
19	05/07/2021 Mon	Processor architecture and design (4.5) Y86 ISA	
20	07/07/2021 Wed	Processor architecture and design (4.5) Y86 ISA	
21	09/07/2021 Fri	Memory Hierarchy 6.1-6.4	Quiz 2
22	12/07/2021 Mon	Memory Hierarchy 6.1-6.4	
23	14/07/2021 Wed	Memory Hierarchy 6.1-6.4	
24	16/07/2021 Fri	Memory Hierarchy 6.1-6.4	
25	19/07/2021 Mon	OS System calls Intro to process control ~ 8.2 – 8.4	
27	23/07/2021 Fri	OS System calls Intro to process control ~ 8.2 – 8.4	
28	26/07/2021 Mon	OS System calls Intro to process control ~ 8.2 – 8.4	
29	28/07/2021 Fri	OS System calls Intro to process control ~ 8.2 – 8.4	
30	30/07/2021 Mon	Virtual memory high level overview	
END SEMESTER EXAM / TERM PAPER ?			

Administrivia

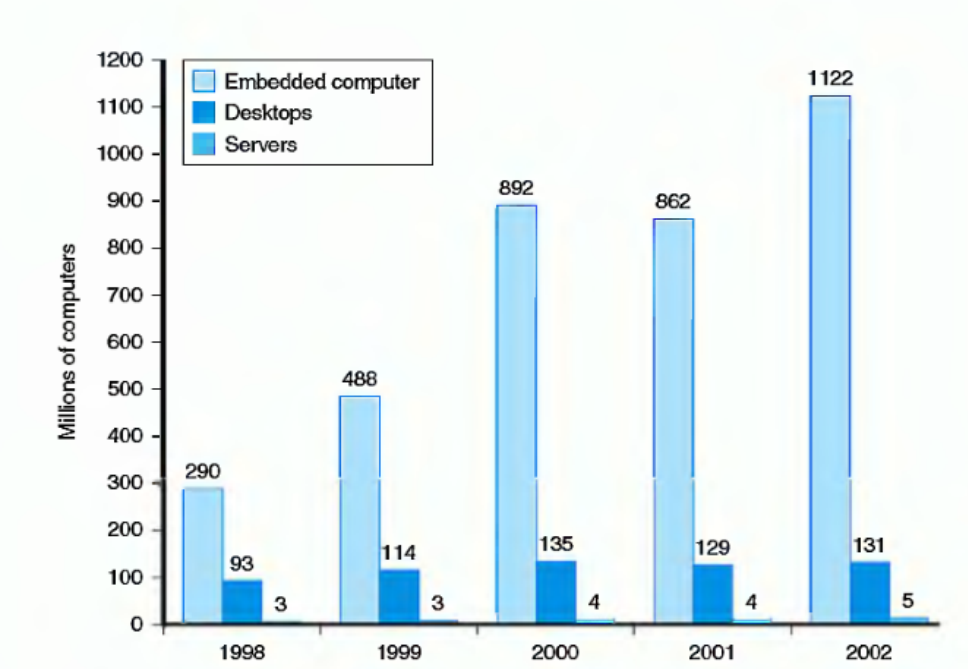
- Grade Distribution (tentative)
  - Continuous evaluations with Exam/Quiz (~60%)
  - Homeworks/Assignment (~40%)

- This is a core course in CS
  - We expect you to work hard to learn it well.
  - Class participation lifts the level of the class
  - We don't want credit-seekers
  - The more your effort, the more everyone benefits!

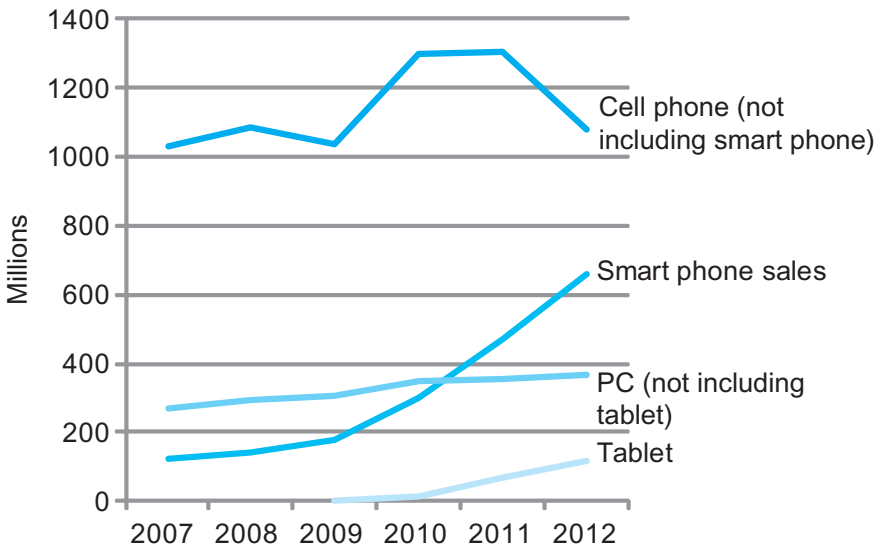
Administrivia

- Co-instructor
  - Ziaul Choudhury ziaul.c@research.iiit.ac.in
- Teaching Assistants
  - Aakash Aanegola aakash.aanegola@students.iiit.ac.in
  - Aakash Jain aakash.jain@students.iiit.ac.in
  - Triansh Sharma <triansh.sharma@students.iiit.ac.in>
  - SAI KALAKONDA sai.shashank@research.iiit.ac.in
  - Samartha S M samartha.s@students.iiit.ac.in
  - Kushagra Agarwal kushagra.agarwal@research.iiit.ac.in
  - Snehith Kumar Chatakonda snehith.kumar@students.iiit.ac.in
  - Aryamaan Jain aryamaan.jain@research.iiit.ac.in

Sales Data



Number of distinct processors sold in Pre-smartphone era

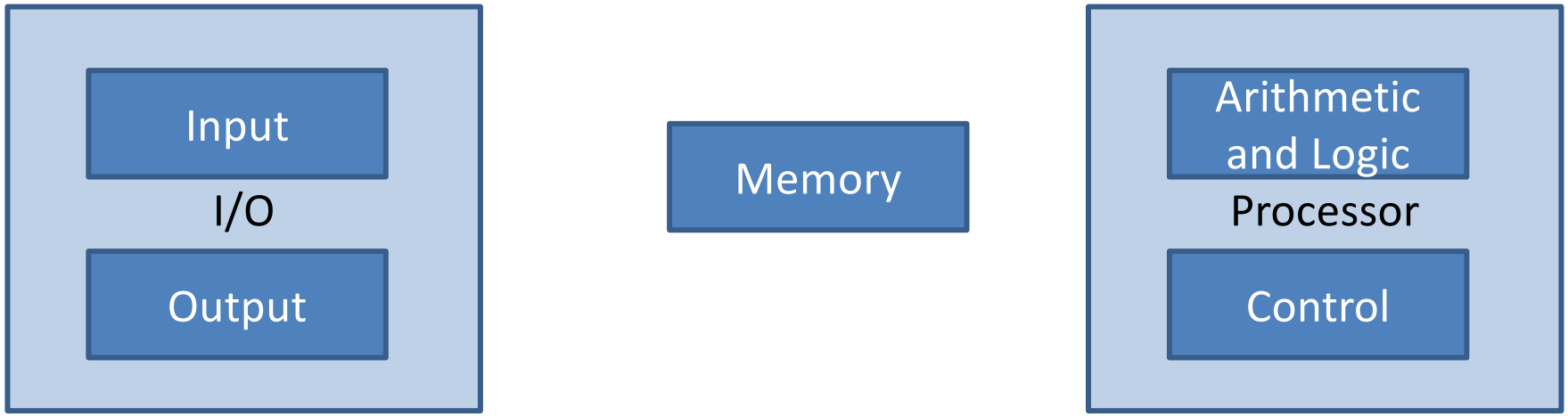


Number of distinct processors sold in PostPC era

Source: Hennessy & Patterson

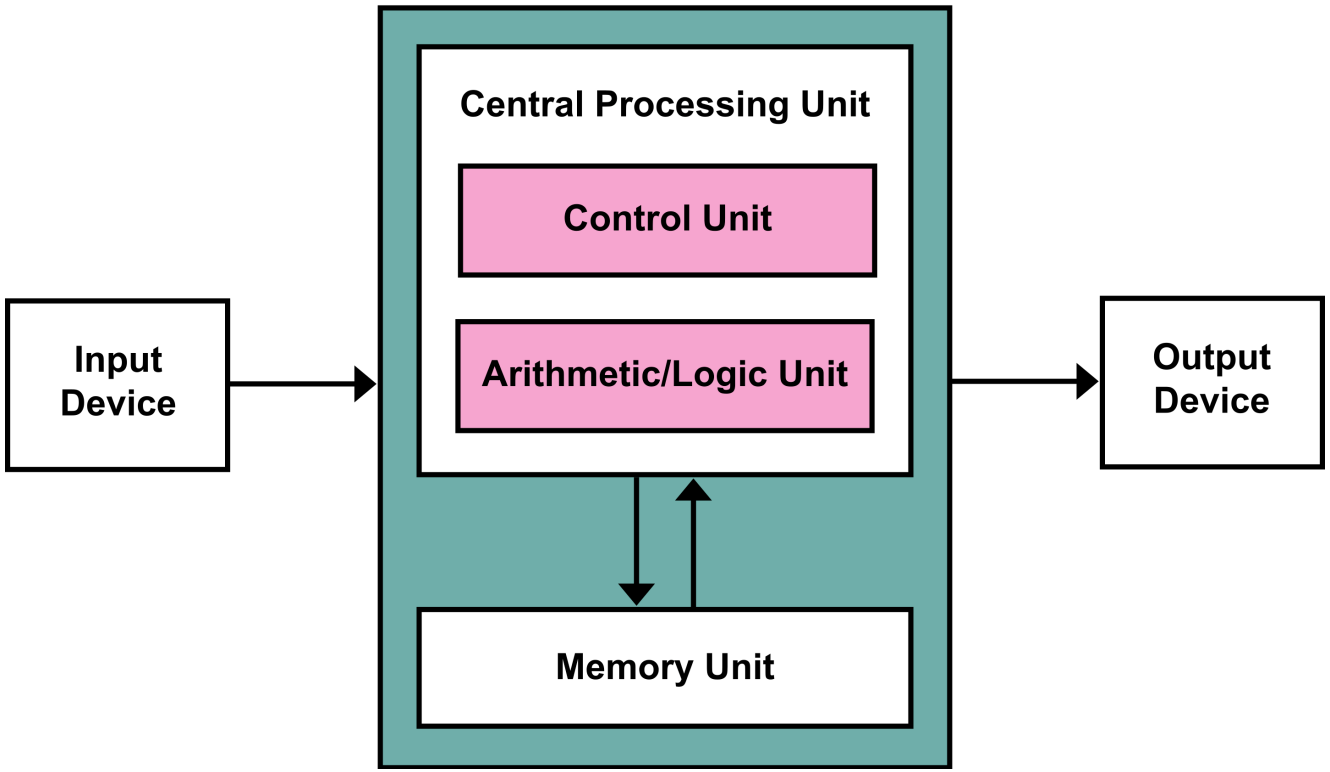
Major Functional Units of a Computer

- Embedded Computers, Desktops and Servers are composed of three main parts functionally.



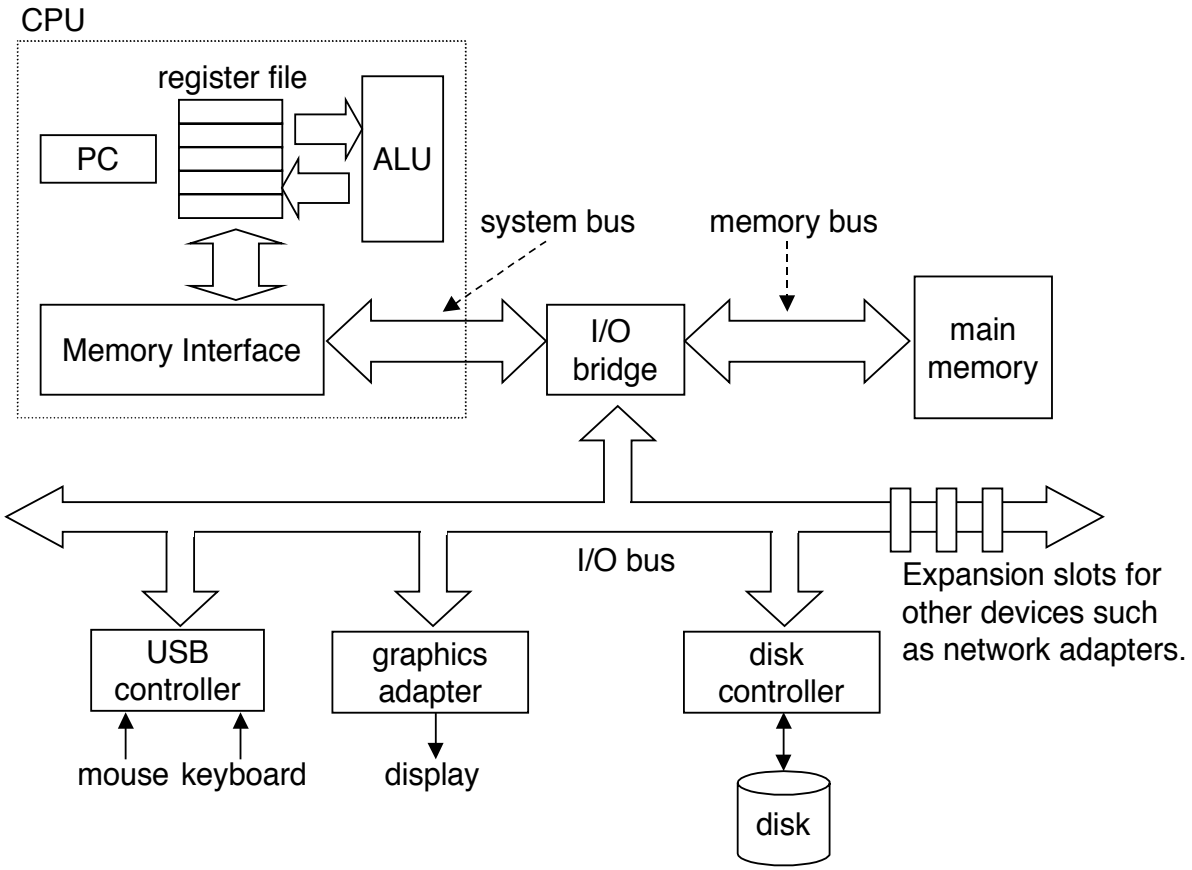
These Three Components can be interconnected in many ways.

Computer Architecture: The von Neumann Model



Source: Wikipedia

Typical Hardware Organization of a System



Source: Bryant & O'Hallaron

Program Execution on Computer System

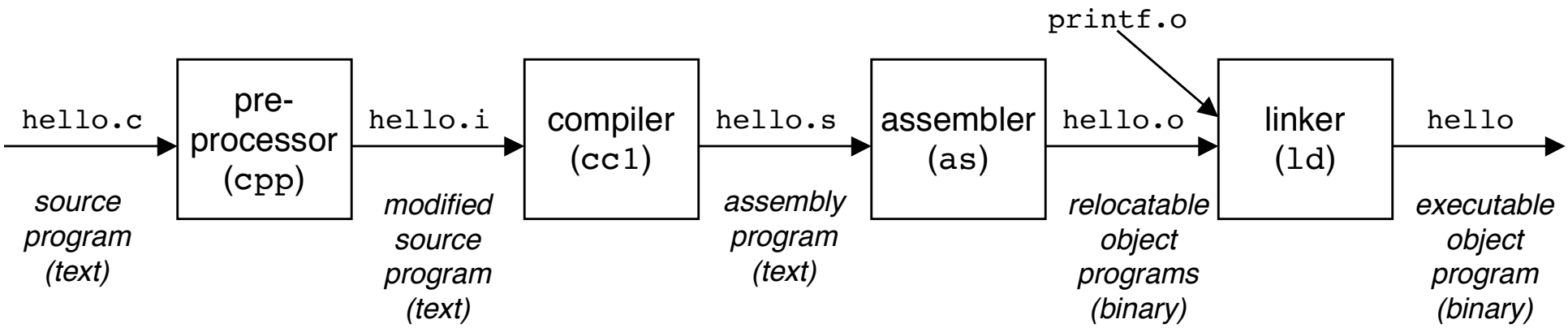
```
1 #include <stdio.h>
2
3 int main()
4 {
5     printf("hello, world\n");
6 }
```

```
 #   i    n     c   l    u    d    e   <sp>   <   s    t    d    i    o     .
 35  105  110   99  108  117  100  101   32   60  115  116  100  105  111   46

h     >    \n   \n  i    n    t   <sp>  m     a   i    n     (    )    \n  {
104   62   10   10  105  110  116   32  109   97  105  110   40   41   10  123

 \n <sp> <sp> <sp> <sp>  p    r    i    n    t    f     (    "   h    e    l
 10   32   32   32   32  112  114  105  110  116  102   40   34  104  101  108

l    o     ,  <sp>  w    o    r    l    d     \   n     "    )    ;    \n  }
108  111   44   32  119  111  114  108  100   92  110   34   41   59   10  125
```

Source: Bryant & O'Hallaron

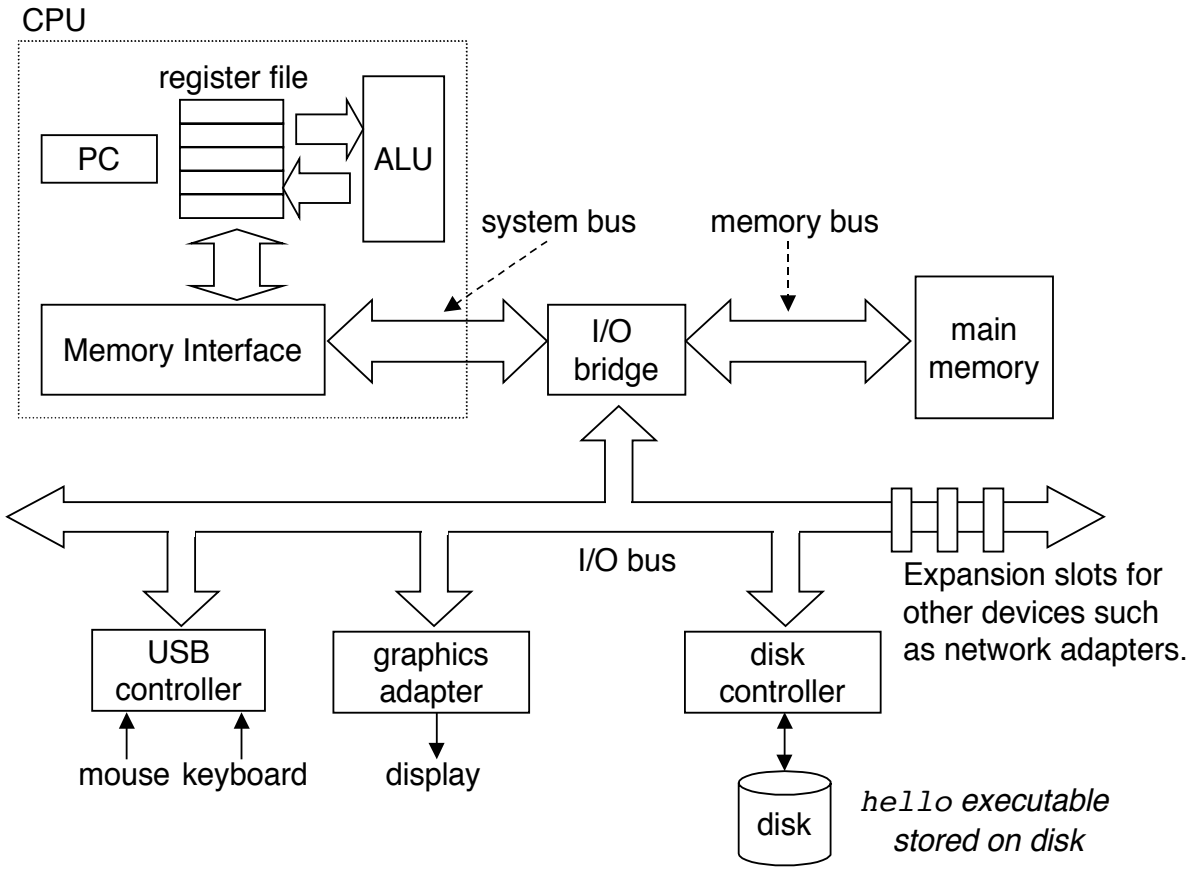
Typical Compilation System

gcc -o hello hello.c



Source: Bryant & O'Hallaron

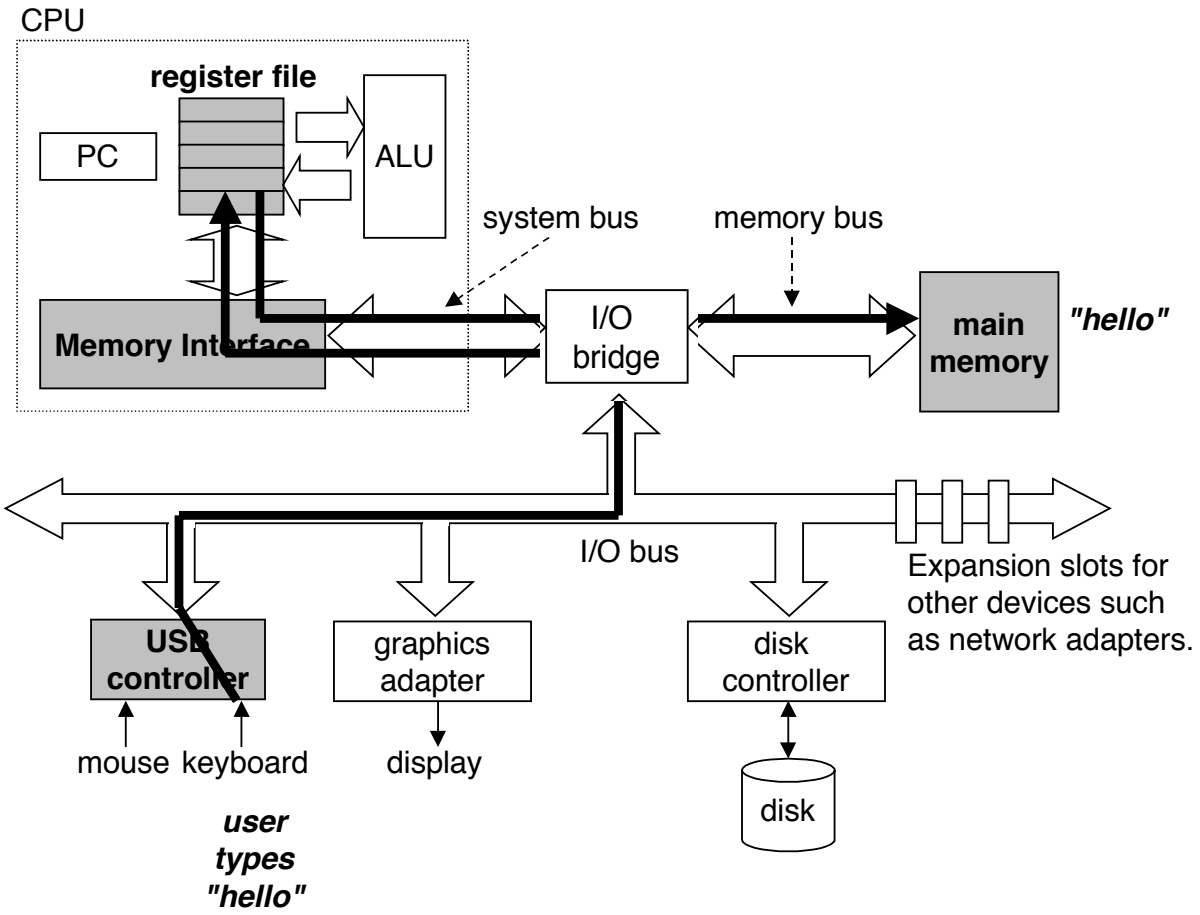
Running the “Hello World” Program




```
unix> ./hello
hello, world
```

Source: Bryant & O’Hallaron

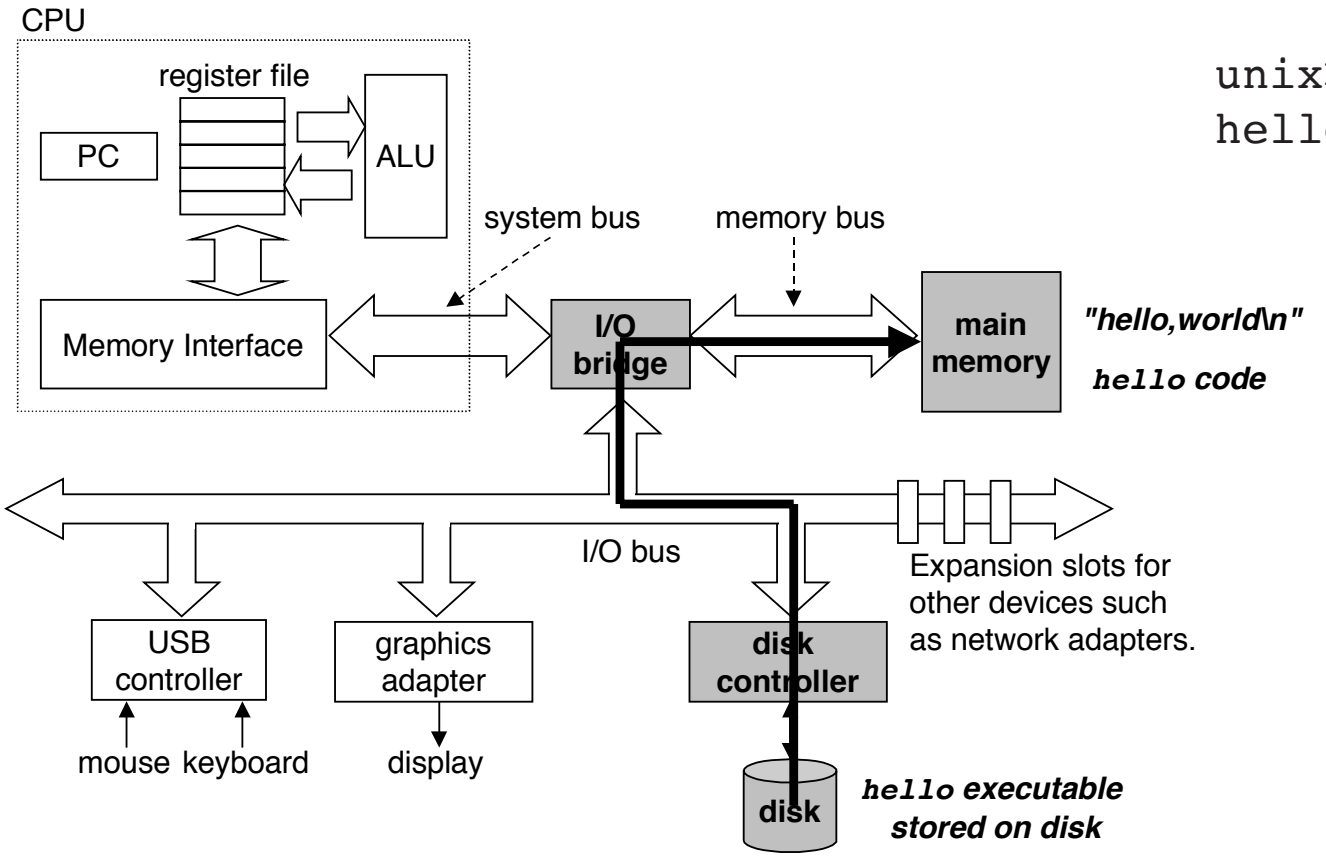
Running the “Hello World” Program




```
unix> ./hello
hello, world
```

Source: Bryant & O’Hallaron

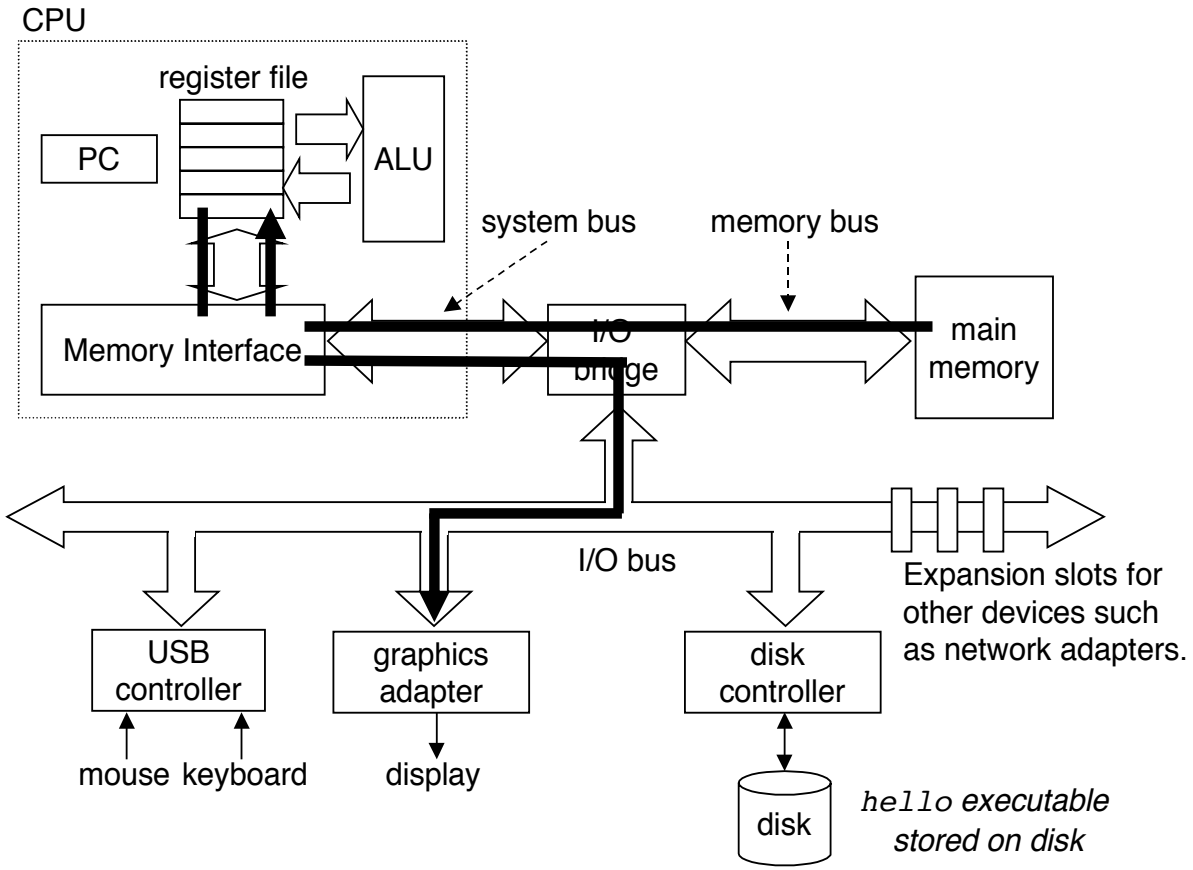
Running the “Hello World” Program




```
unix> ./hello
hello, world
```

Source: Bryant & O’Hallaron

Running the “Hello World” Program




```
unix> ./hello
hello, world
```

Source: Bryant & O’Hallaron

Programming Abstractions

We can program a microprocessor using

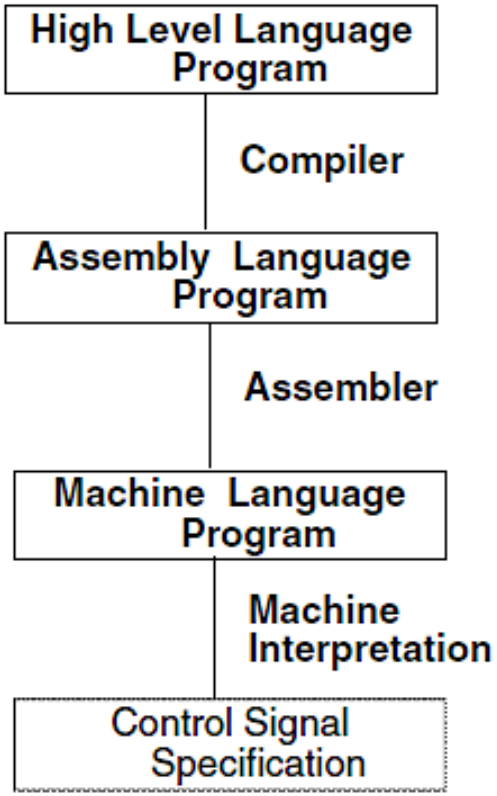
a) Instruction opcodes (also called Machine Code)

b) Assembly language

c) High level programming languages

- The level of abstraction increases from Top to Bottom.
- As the level of abstraction increases, ease of programmability also increases!
- Hmm, but we may lose the fine-grained control over the underlying hardware?

Levels of Abstraction




```
temp := v[k];
v[k] := v[k+1];
v[k+1] := temp;
```

```
lw   $15, 0($2)
lw   $16, 4($2)
sw   $16, 0($2)
sw   $15, 4($2)
```

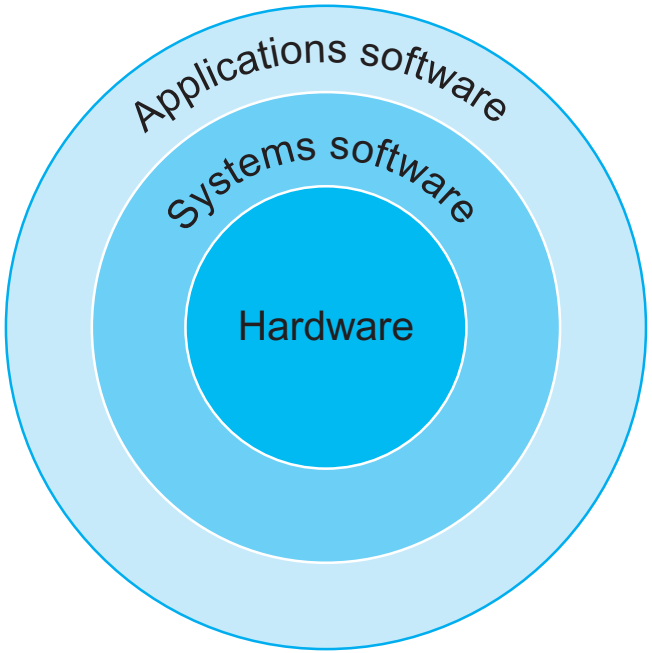
*MIPS Assembly Language

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

*Microprocessor without Interlocked Pipeline Stages

Source: Prof. Cheung (ICL)

Computer System = Hardware + System Software + Application Software



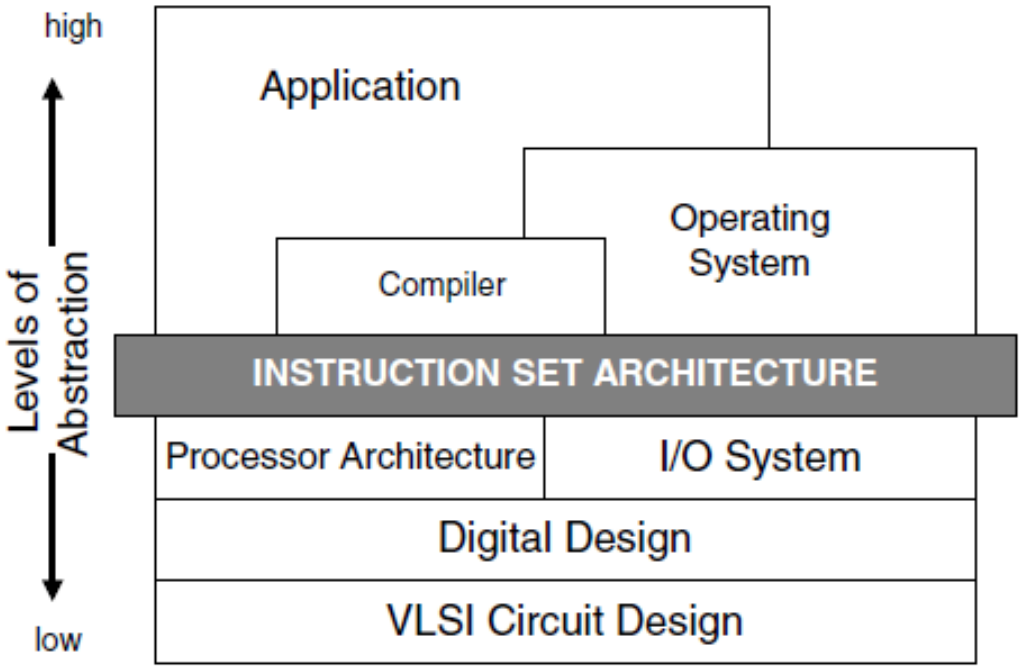
System Software: Operating System, Device Drivers, Loaders, Linkers, Compilers, Assemblers, Editors, ….

Application Software: Web browsers, user-specific applications, …..

Source: Hennessy & Patterson

Instruction Set Architecture (ISA)

- ISA is an abstraction for the Software to interface with the Hardware.
- Advantage: Multiple implementations for the same ISA.
  - Ex: AMD Opteron 64 and Intel Pentium 4 are different Implementations of the ISA.



What does ISA consists of?

- Instruction Set
- Instruction Format
- Data Types and Data Structures (Integer, Floating Point, …)
- Addressing Modes
- Exceptional Conditions
- …….

Technologies for Building Processors and Memories

A transistor is simply an on/off switch controlled by electricity.

Year	Technology used in computers	Relative performance/unit cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit	900
1995	Very large-scale integrated circuit	2,400,000
2013	Ultra large-scale integrated circuit	250,000,000,000

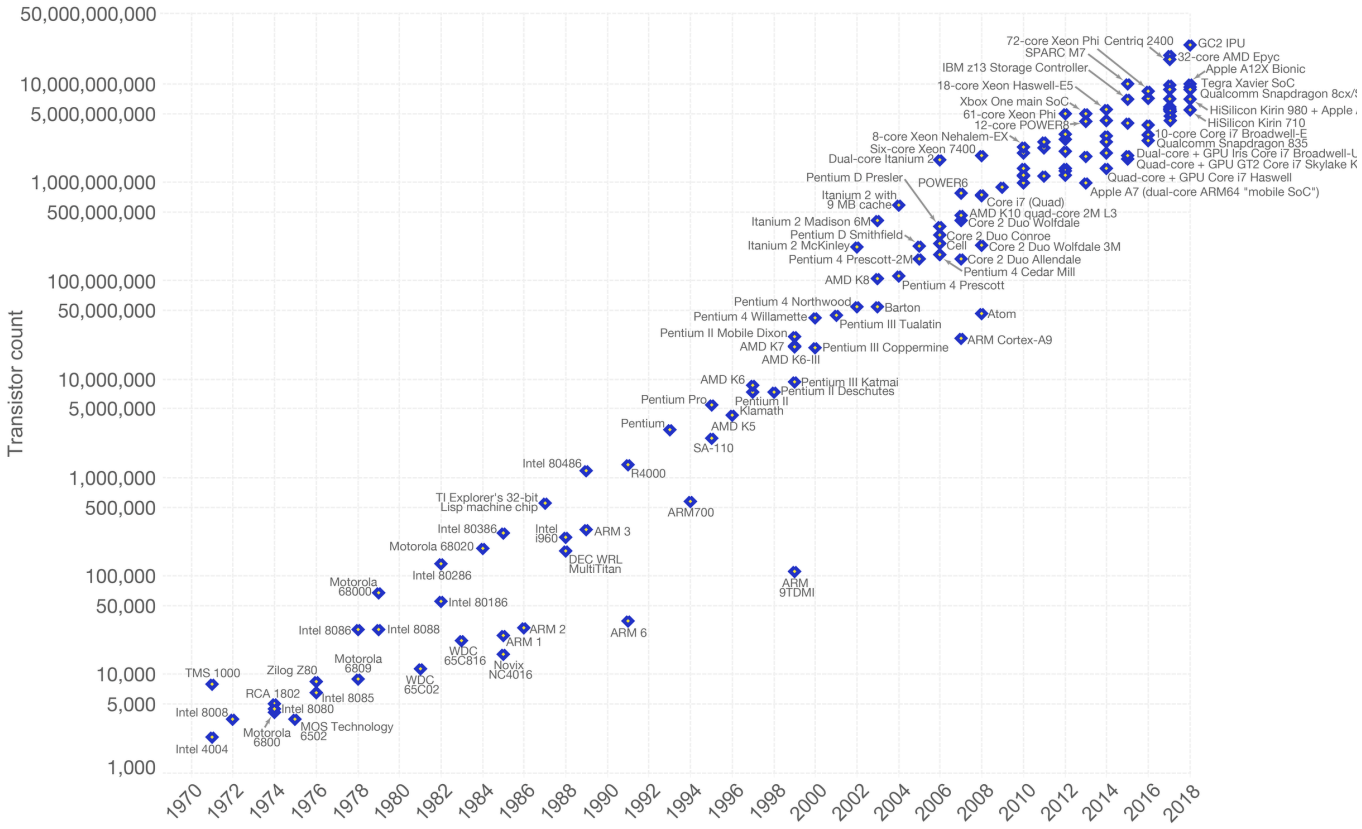
Source: Hennessy & Patterson

Moore’s Law

Moore’s law (1965):

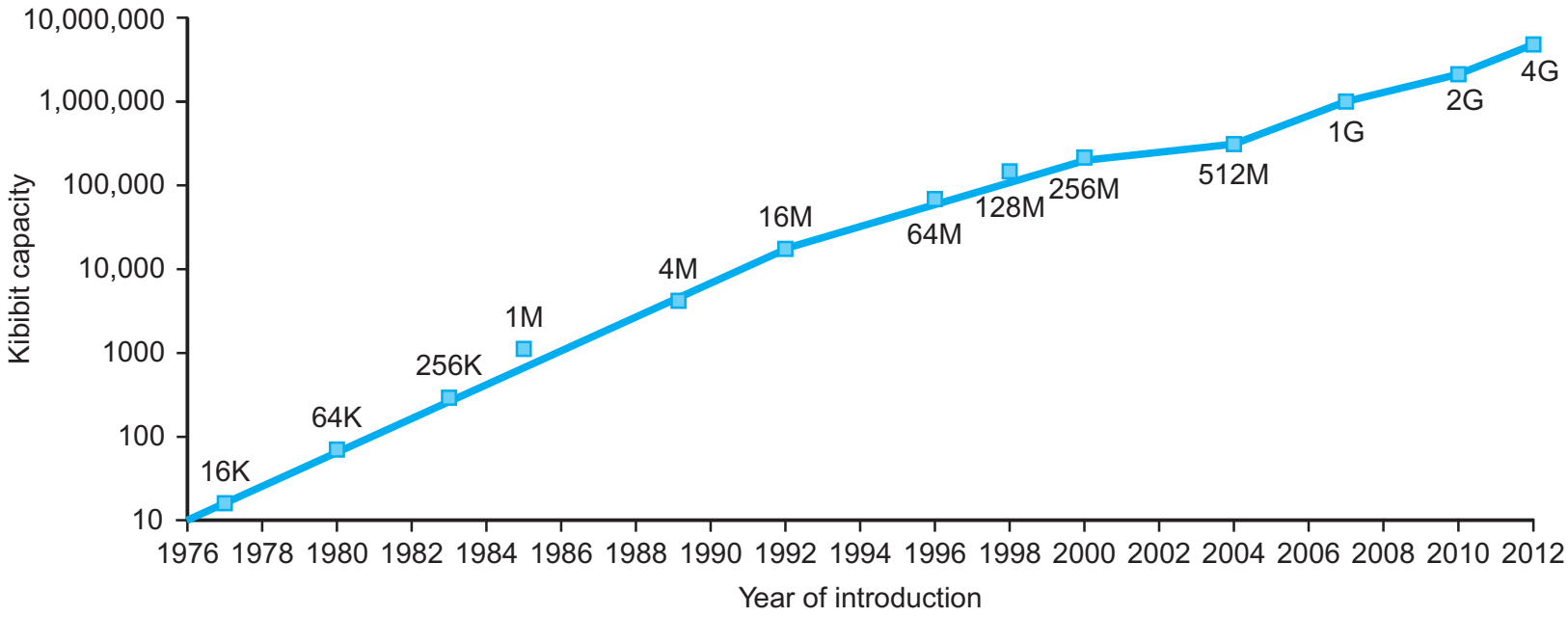
The number of transistors that can be placed inexpensively on an integrated circuit **doubles** every two years.

- Gordon Moore, Intel co-founder



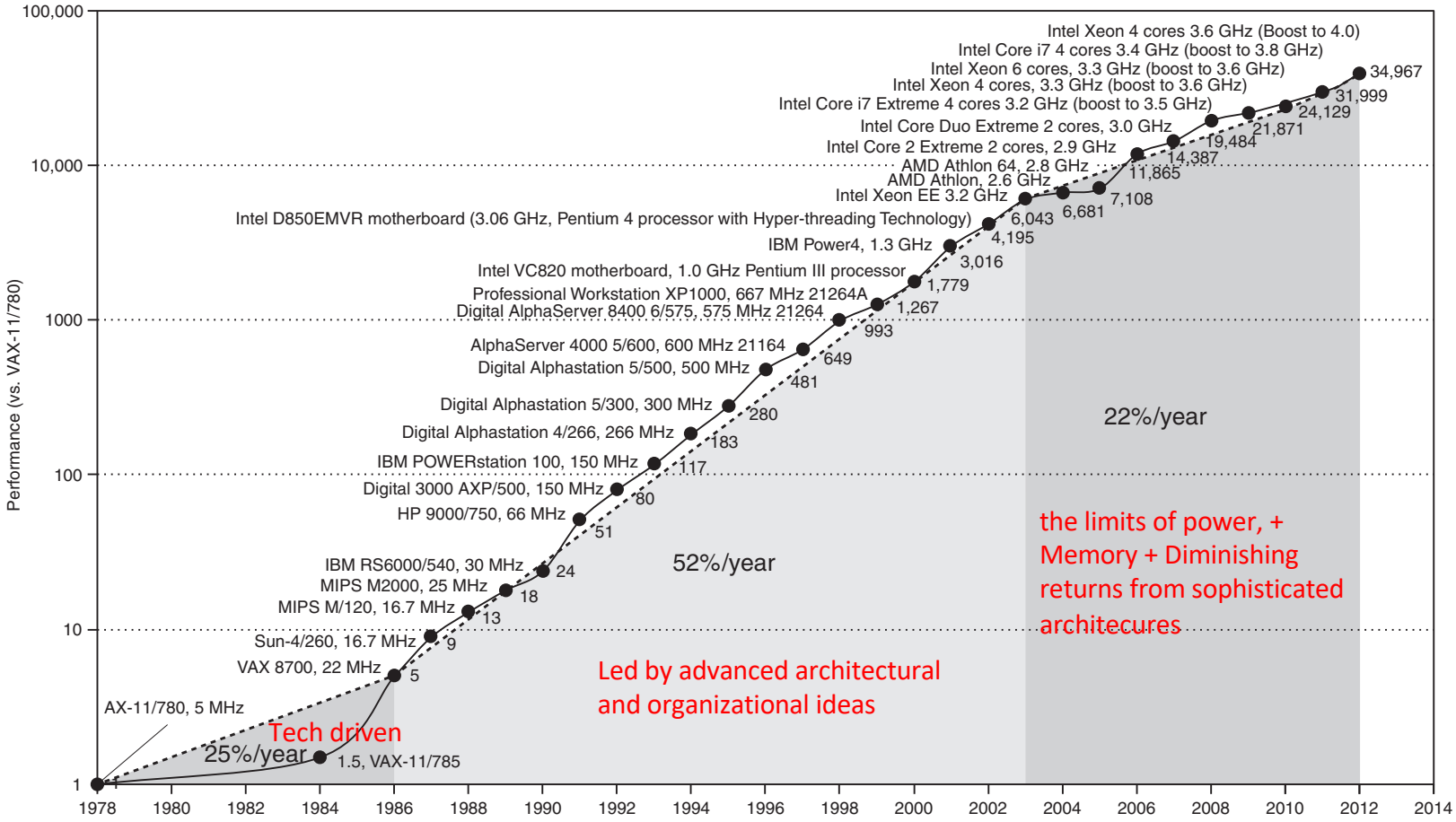
Source: Wikipedia

Moore’s Law & Memory



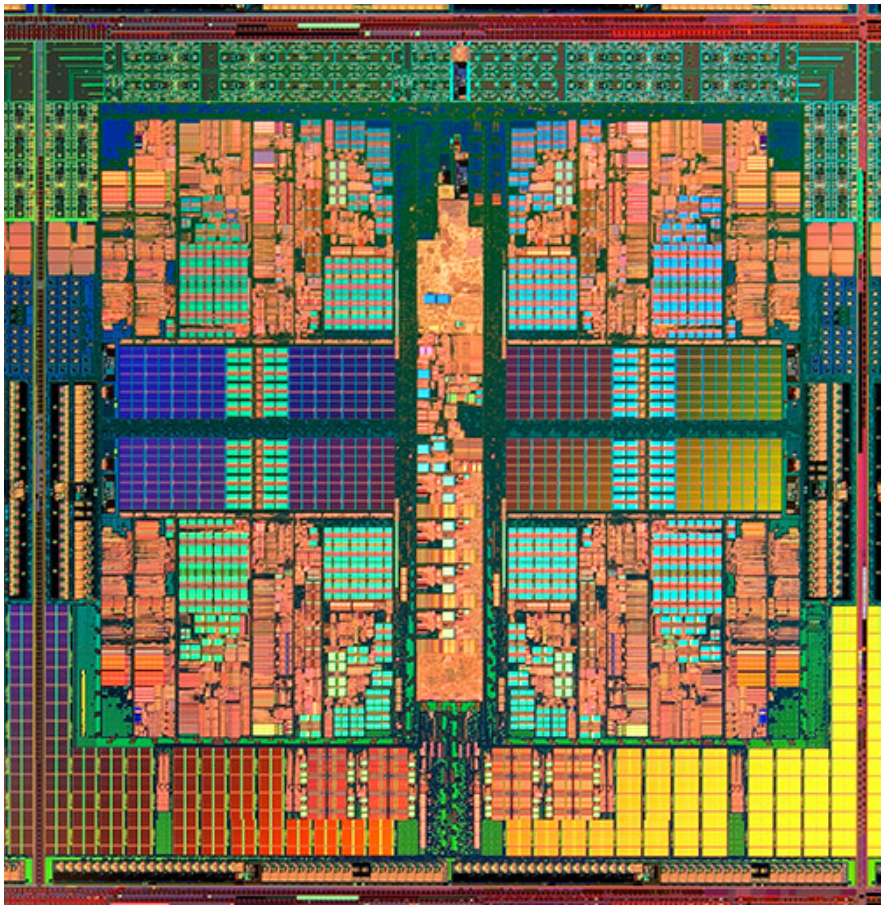
Source: Hennessy & Patterson

Performance Trend (Uniprocessor)



Source: Hennessy & Patterson

Dawn of Multi-core Era



AMD Barcelona
Quadcore

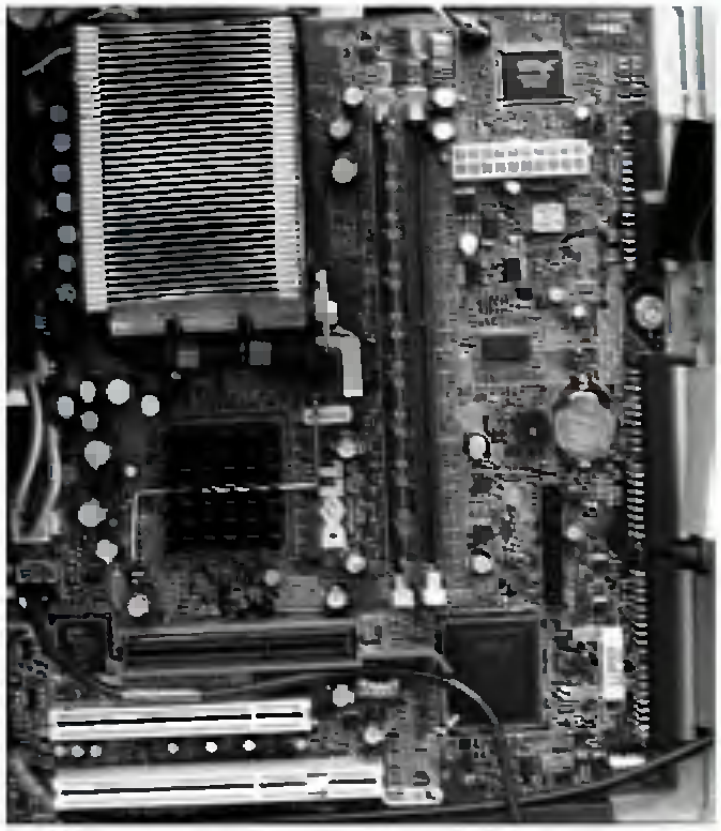
Inside a PC

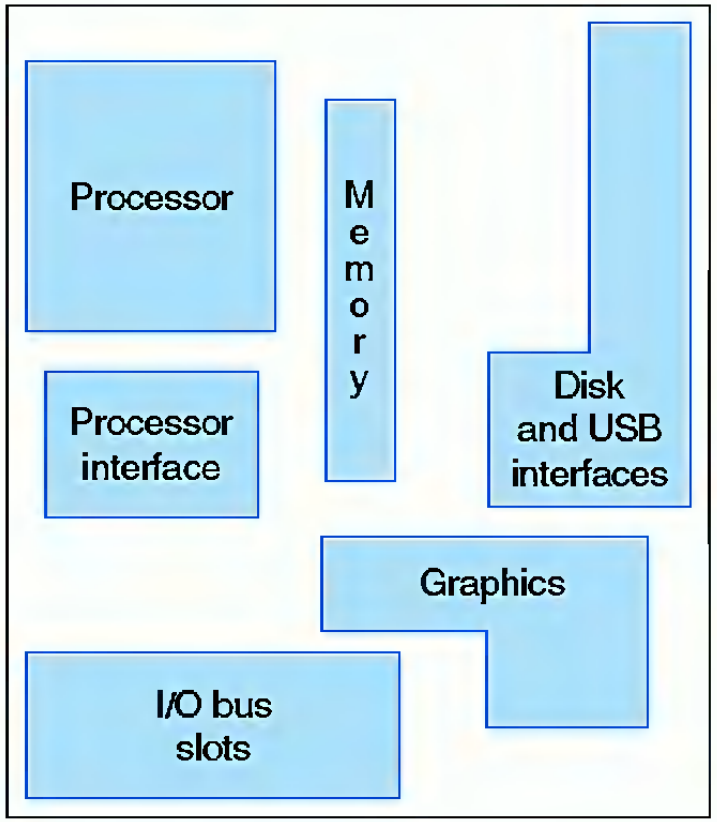
Source: Hennessy & Patterson

Inside a Laptop

Source: HowStuffWorks

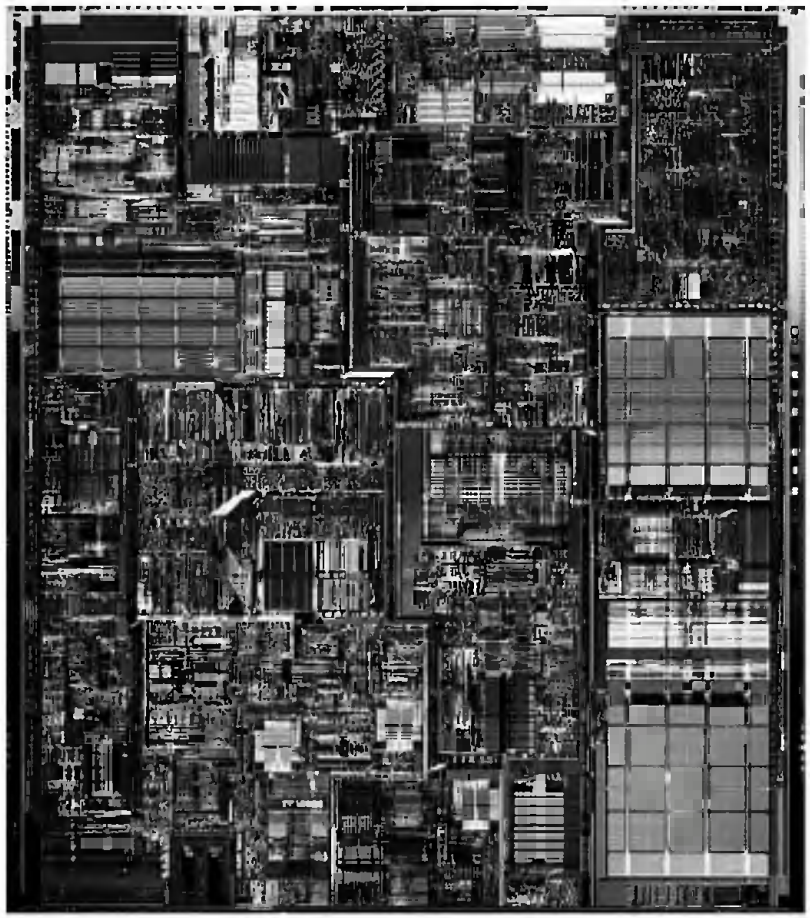
Close-up of a Motherboard

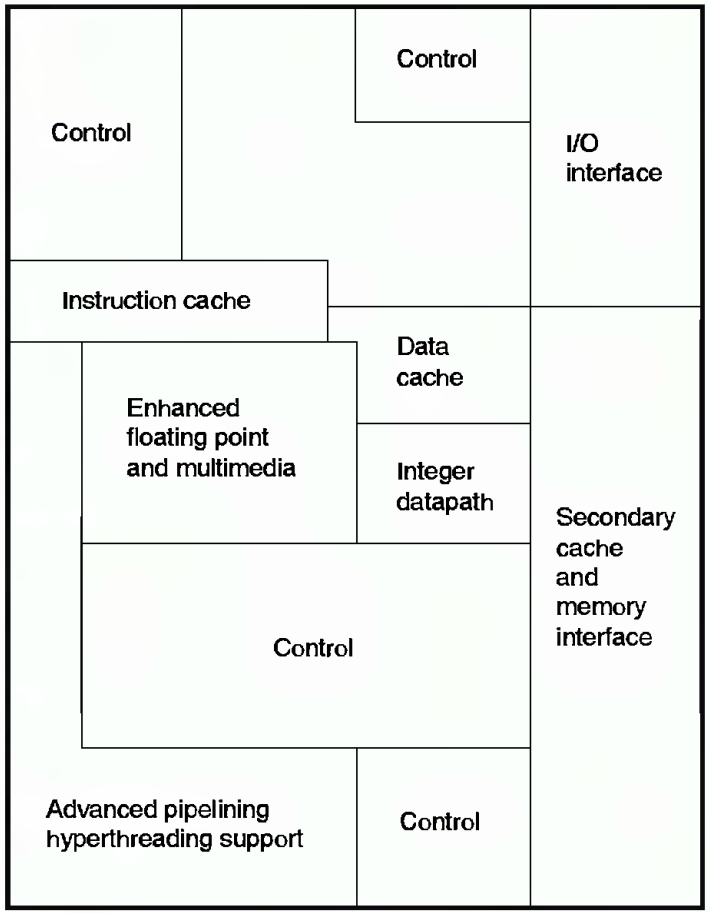




Source: Hennessy & Patterson

Inside a Pentium 4 Processor





Source: Hennessy & Patterson