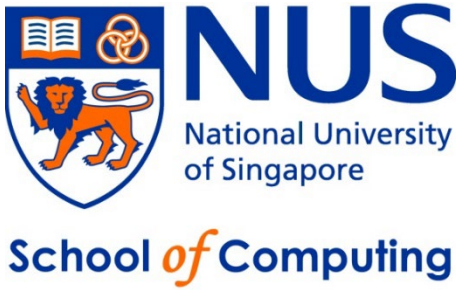
IT5002

Computer Systems and Applications

Introduction

colintan@nus.edu.sg



NUS National University of Singapore
School of Computing

Q & A

- **DO NOT use the Zoom chat for questions. It doesn't appear in the video recordings.**
- **Please ask questions at https://sets.netlify.app/module/61597486a7805d9fb1b4accd**

← OR scan this QR code (may be obscured on some slides)

NUS National University of Singapore
School of Computing

Lecture 1: Introduction

1. **Programming Languages**
2. **Abstraction**
3. **What is a Computer?**
4. **IT5002: It's About Computer Organization and Applications**
5. **What's Next?**

Credit for course notes: A/P Aaron Tan (CS2100)

Programming Languages

Programming language: a formal language that specifies a set of instructions for a computer to implement specific algorithms to solve problems.

Programming Languages

High-level program

Eg: C, Java, Python, ECMAScript

```
int i, a = 0;
for (i=1; i<=10; i++) {
        a = a + i*i;
}
```

```
a = 0
for i in range(1,11):
  a = a + i*i
```

Low-level program

Eg: MIPS (IT5002)

```
      addi $t1, $zero, 10
      add  $t1, $t1, $t1
      addi $t2, $zero, 10
Loop: addi $t2, $t2, 10
      addi $t1, $t1, -1
      beq  $t1, $zero, Loop
```

Machine code

```
00100000000010010000000000001010
00000001001010010100100000100000
. . .
```

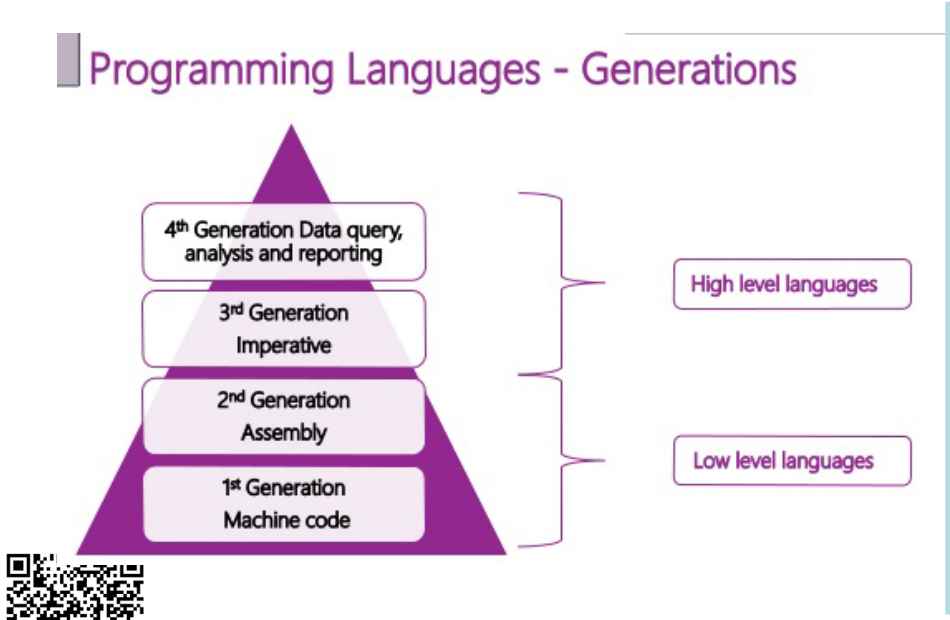
Computers can execute only machine code directly.

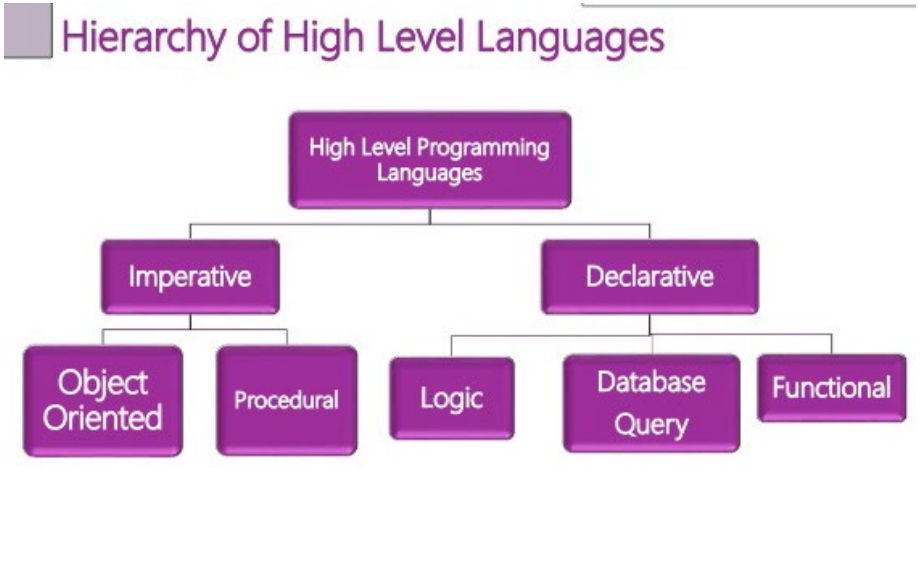
Programming Languages

- 1st Generation Languages
  - Machine language.
    Directly executable by machine.
    Machine dependent.
    Efficient code but difficult to write.
- 2nd Generation Languages
  - Assembly language.
    Need to be translated (assembled) into machine code for execution.
    Efficient code, easier to write than machine code.
- 3rd Generation Languages
  - Closer to English.
    Need to be translated (compiled or interpreted) into machine code for execution.
    Eg: FORTRAN, COBOL, C, BASIC
- 4th Generation Languages
  - Require fewer instructions than 3GL.
    Used with databases (query languages, report generators, forms designers)
    Eg: SQL, PostScript, Mathematica
- 5th Generation Languages
  - Used mainly in A.I. research.
    Declarative languages
    Functional languages (eg: Lisp, Scheme, SML)
    Logic programming (eg: Prolog)

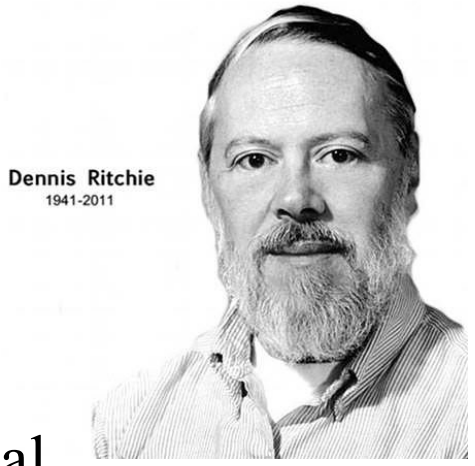
Programming Languages

- "Generational" classification of high level languages (3GL and later) was never fully precise.
- A different classification is based on paradigm.





The C Programming Language



- Created by Dennis Ritchie (1941 – 2011) at Bell Laboratories in the early 1970s.
- C is an imperative procedural language.
- C provides constructs that map efficiently to typical machine instructions.
- C is a high-level language very close to the machine level, hence sometimes it is called "mid-level".
- UNIX is written in C.

HelloWorld.c

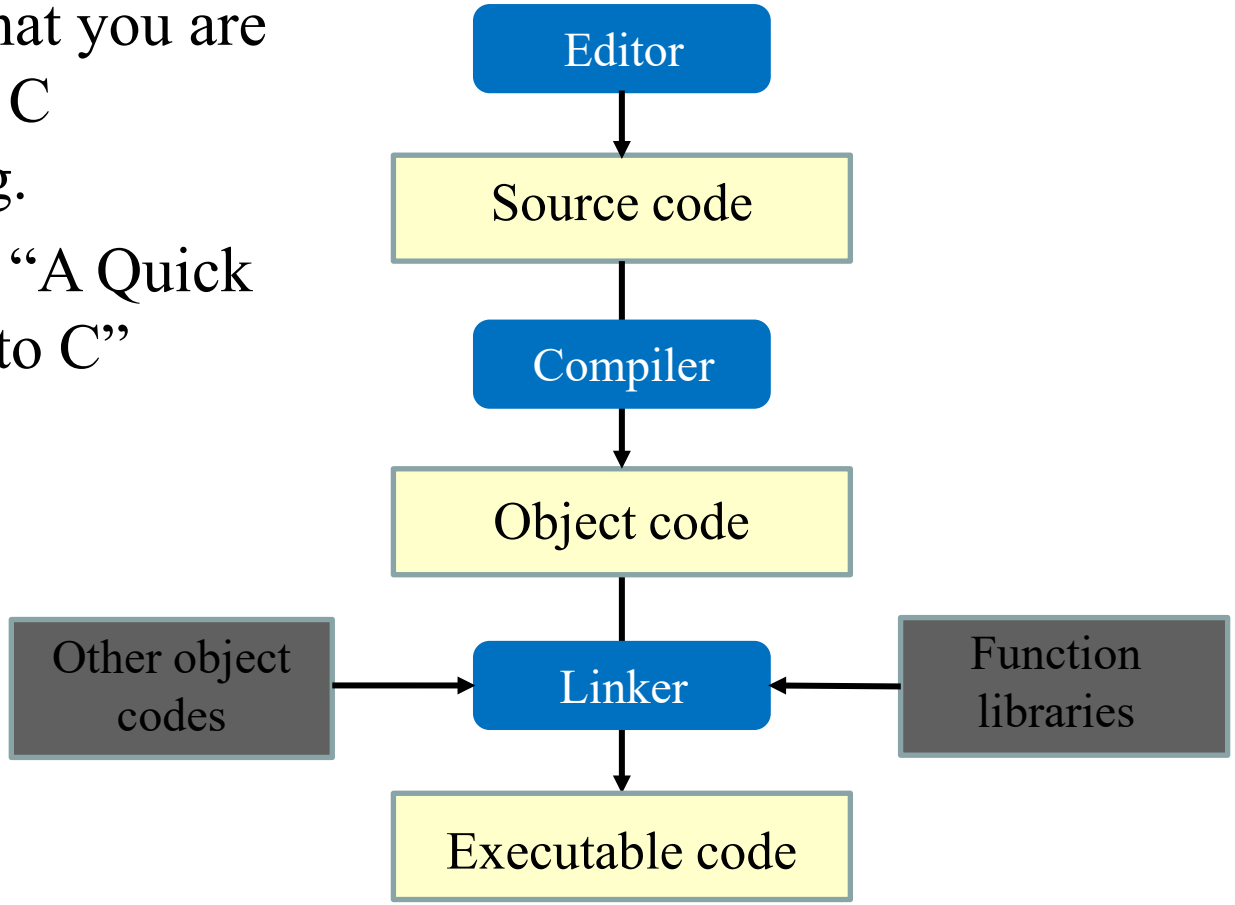
```c
#include <stdio.h>

int main(void) {
   printf("Hello, world\n");
   return 0;
}
```

HelloWorld.py

```python
print("Hello, world")
```

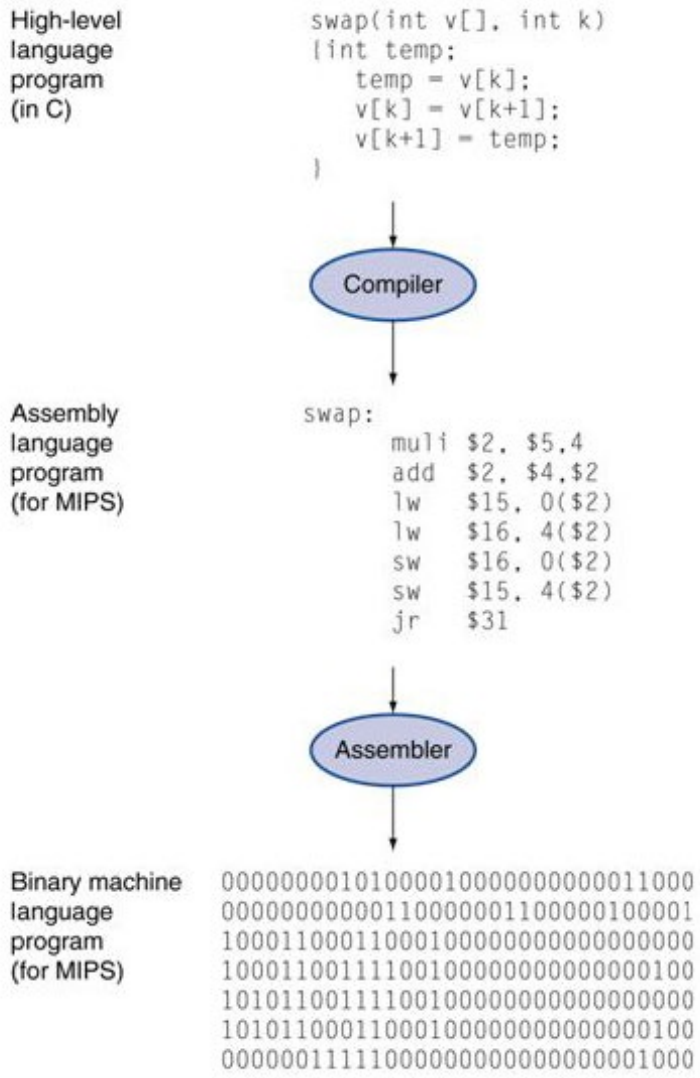
The C Programming Language

- Creating a C program
  - We assume that you are familiar with C Programming.
  - If not see the "A Quick Introduction to C" document.



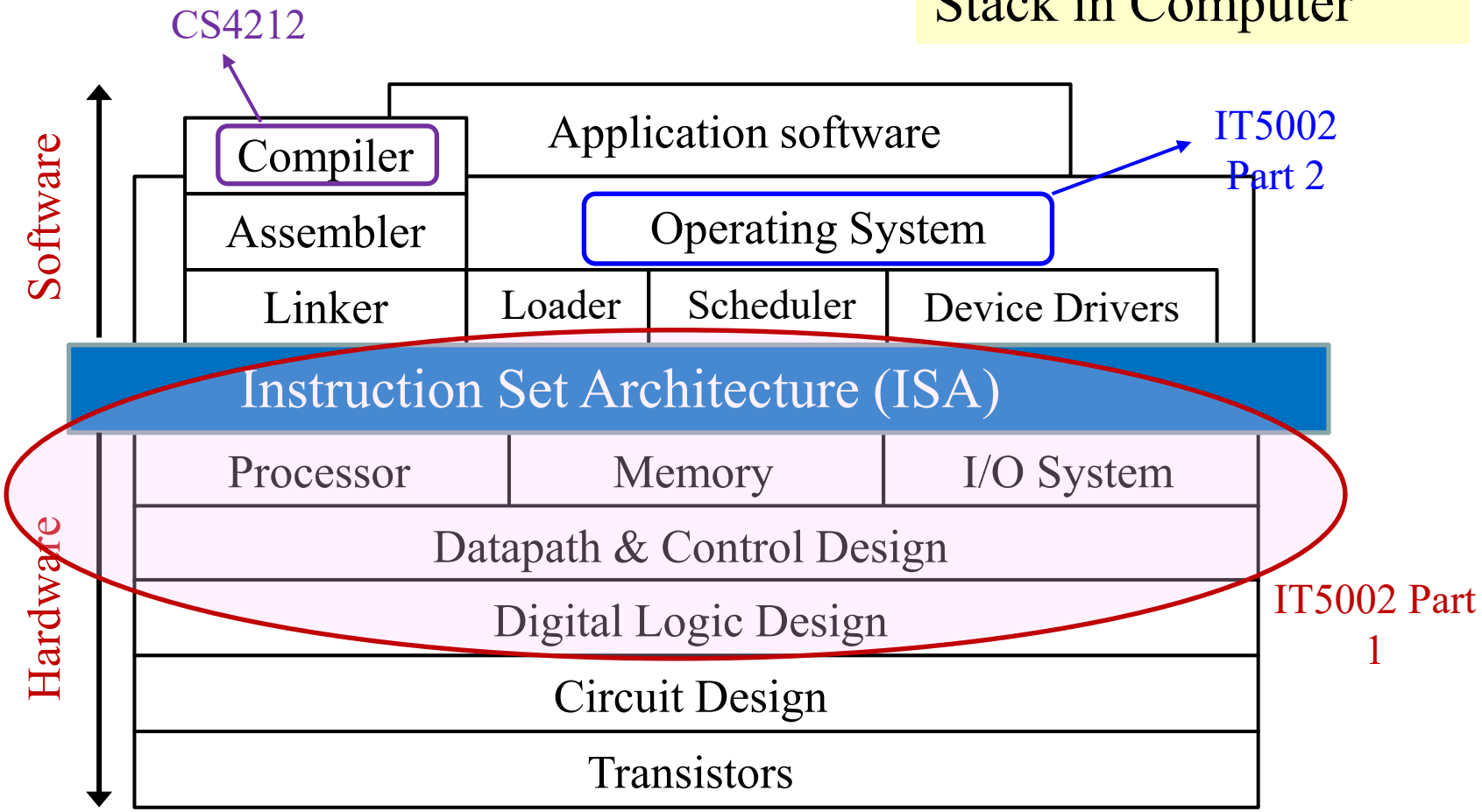
Abstraction

- High-level language
  - Level of abstraction closer to problem domain
  - Provides productivity and portability
- Assembly language
  - Textual and symbolic representation of instructions
- Machine code (object code or binary)
  - Binary bits of instructions and data



Abstraction

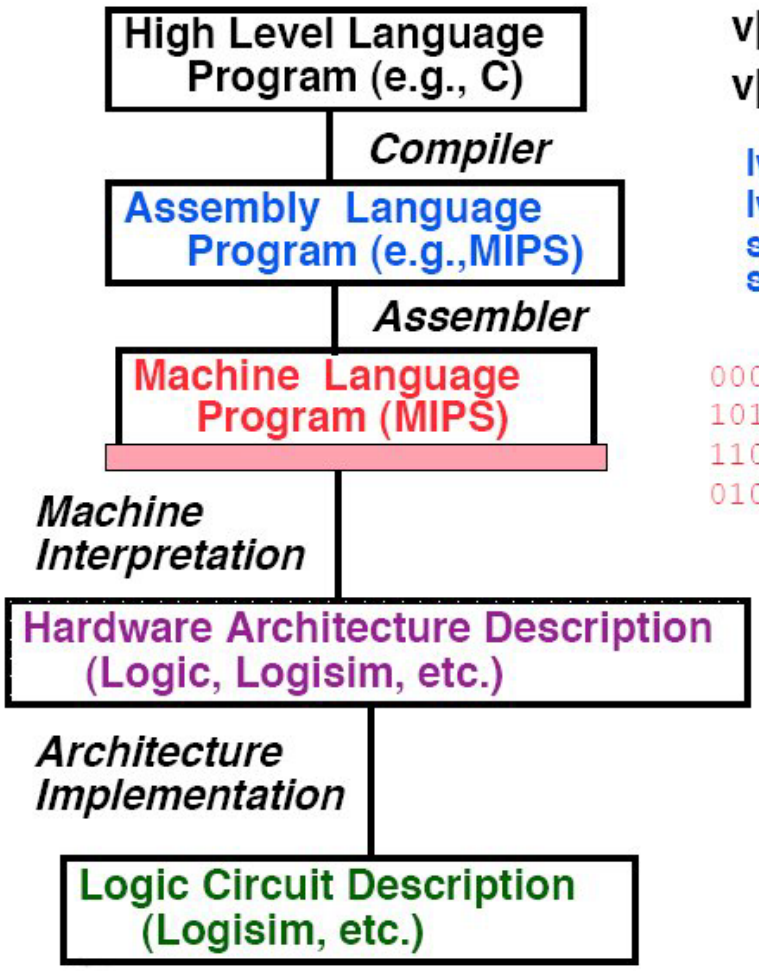
Hardware/Software Stack in Computer



NUS National University of Singapore
School of Computing

Abstraction

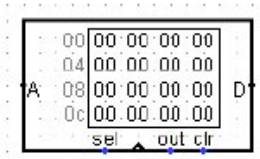
Level of Representation

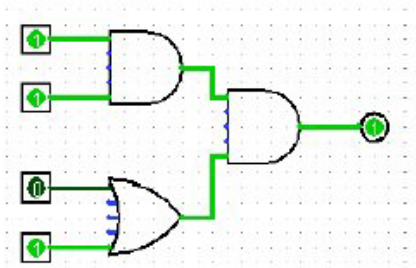



```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
lw   $t0, 0($2)
lw   $t1, 4($2)
sw   $t1, 0($2)
sw   $t0, 4($2)
```

```
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
```





What is a Computer?

MOST RECENT July 8th, 2012 1 COMMENT »

Most Recent Computer Technology

Written by: admin
Tags: breaking

Do you know what is in your computer? Maybe you peeked when the repair technician was installing amazing for you. When you primary open up the CPU and seem inside, a computer is a very intimidating machine. But once you are acquainted with about the dissimilar parts that make up a total computer it gets a lot easier. Today's computer consists of around eight main devices; some of the advanced computers might have a few additional mechanisms. What are these eight main components and what are they used for? We will start with beginner level facts to get you in progress.

First is the Power Supply. The authority provides is used to provide electrical

SHARE 0 Like Digg

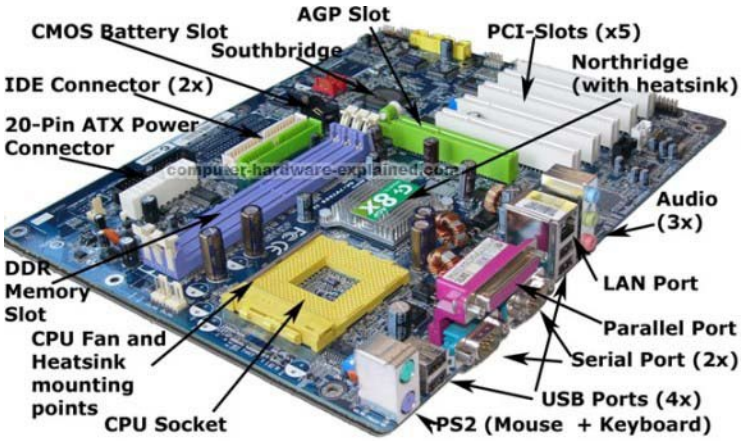
Credit: http://tech3news.com/most-recent-computer-technology/

1. Power supply
2. Motherboard
3. Central Processing Unit (CPU)
4. Random Access Memory (RAM)
5. Hard drive
6. Cooling fan
7. I/O devices

Credit: http://www.overclock3d.net/reviews/cpu_mainboard/the_computer_council_-_clocked_gamer_quad/1

What is a Computer?

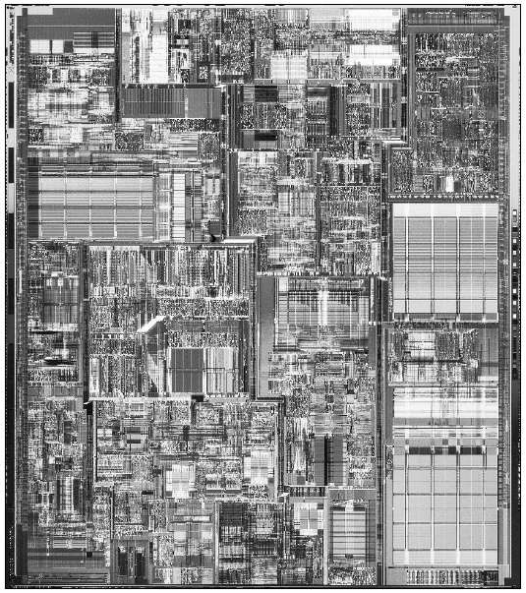
- PC motherboard

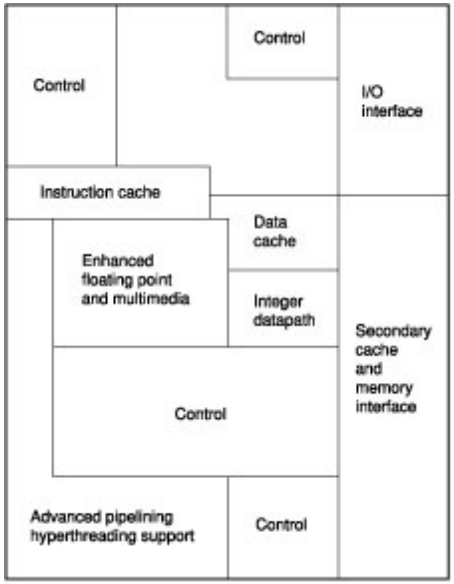


Credit: http://www.computer-hardware-explained.com/what-is-a-motherboard.html

- Pentium processor

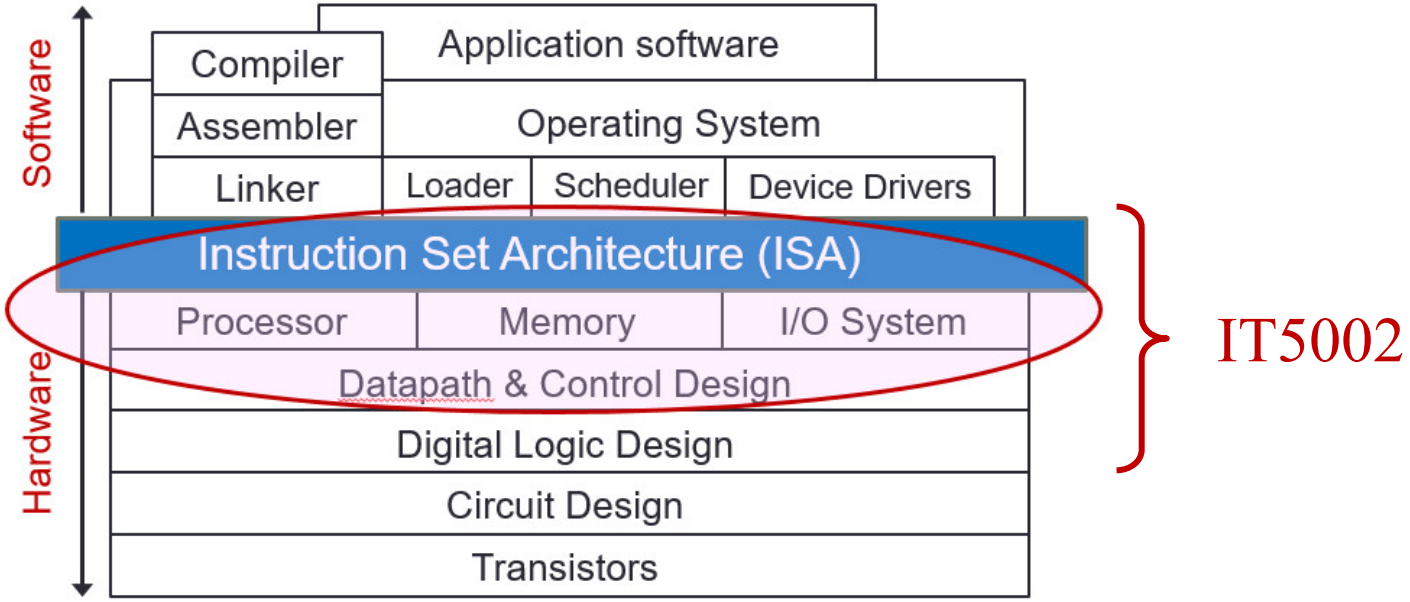
Inside a Pentium chip





IT5002 : It's About Computer Organization

- Computer organisation is the study of internal working, structuring and implementation of a computer system.
- It refers to the level of abstraction above the digital logic level, but below the operating system level.



NUS National University of Singapore
School of Computing

IT5002 : It's About Computer Organization

(From user to builder)

- You want to call yourself a computer scientist/specialist.
- You want to build software people use.
- You need to make purchasing decisions.
- You need to offer "expert" advice.
- Hardware and software affect performance
  - Algorithm determines number of source-level statements
  - Language, compiler, and architecture determine machine instructions (COD chapters 2 and 3)
  - Processor and memory determine how fast instructions are executed (COD chapters 5, 6 and 7)
- Understanding performance (COD chapter 4)