Introduction

Part 2

IT254 Computer Architecture & Microprocessor Interfacing

Why CO important for software programmers?

- The performance of program depends on

Hardware or Software Component	How this component affects performance?
Algorithms	Determines both number of source level statements and number of I/O operations executed
Programming language, compiler	Determine the number of computer instructions for each source level statement
Processor and Memory System	Determine how fast instructions can be executed
I/O System(Hardware and operating system)	Determines how fast I/O operations may be executed

What is computer architecture?

- Computer Architecture is the science and art of selecting and interconnecting hardware components to create computers that meet functional, performance and cost goals.
- Two Analogies could be
  - “Architecture of buildings”
  - “Standard motor car Design”
- So computer architecture mainly tells about
  - overall functionality of the computer
  - basic operation it can perform, how they can be sequenced and so on.

The Bad News

- Digital hardware
  - is complex(millions of components in computer)
  - cannot be understood in one course
  - requires background in electricity and electronics

The Good News

- It is possible to understand architectural components without knowing low-level technical details.
- Programmers only need to know the essentials
  - Characteristics of major components
  - Role in overall system
  - Consequences for programmers

- It can be done by **abstraction**.

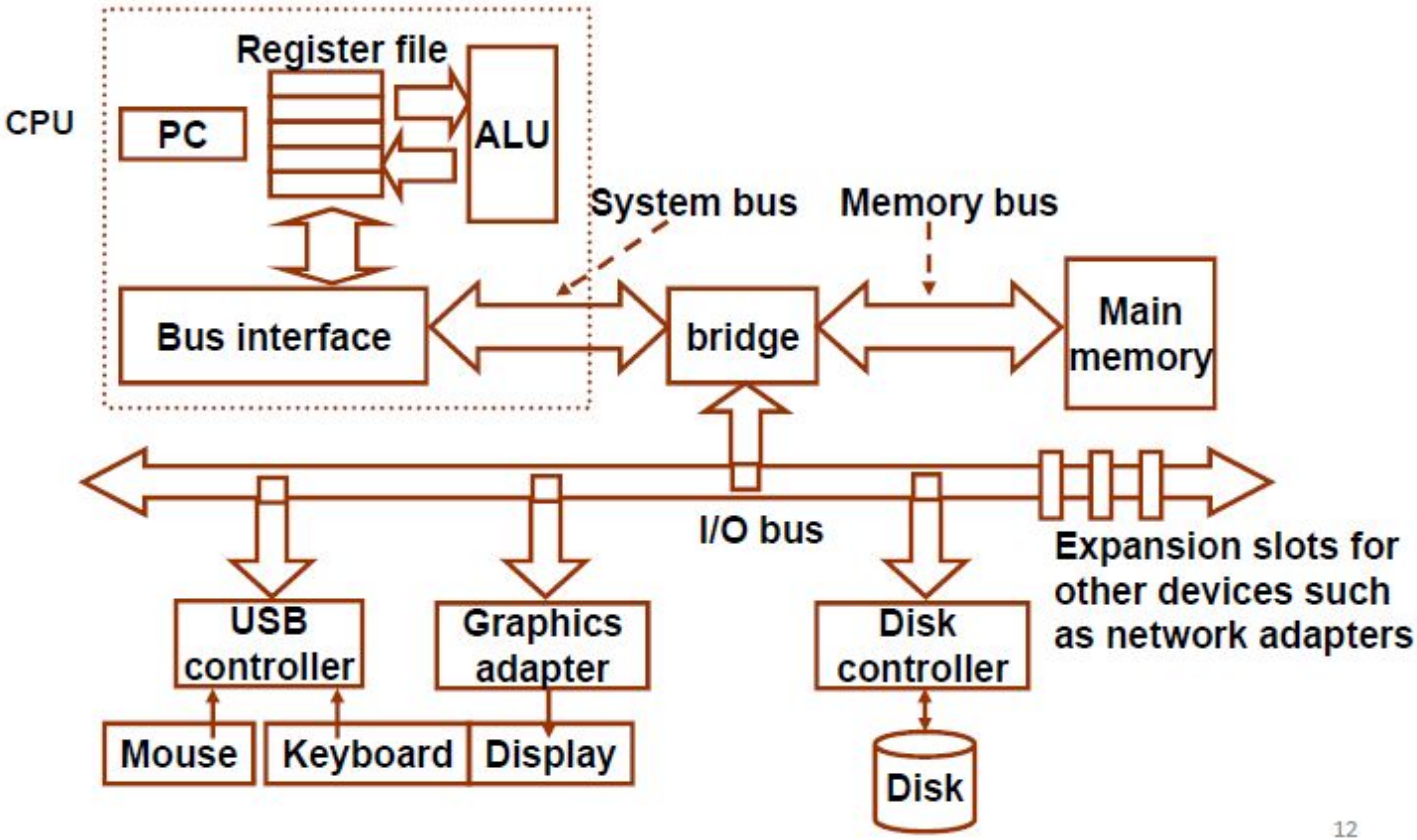
Abstraction

- An abstraction omits unneeded detail and reduces to a set of essential characteristics, helping us to cope with complexity.
- We will look at both software and hardware abstraction.

Software abstraction

- A program is written in high level language.
- It is converted into assembly level program by compiler
- Then assembler converts this assembly level program into machine language understood by computer.
- So user need not worry about knowing assembly or machine language. It is handled by compiler and assembler.

Hardware abstraction



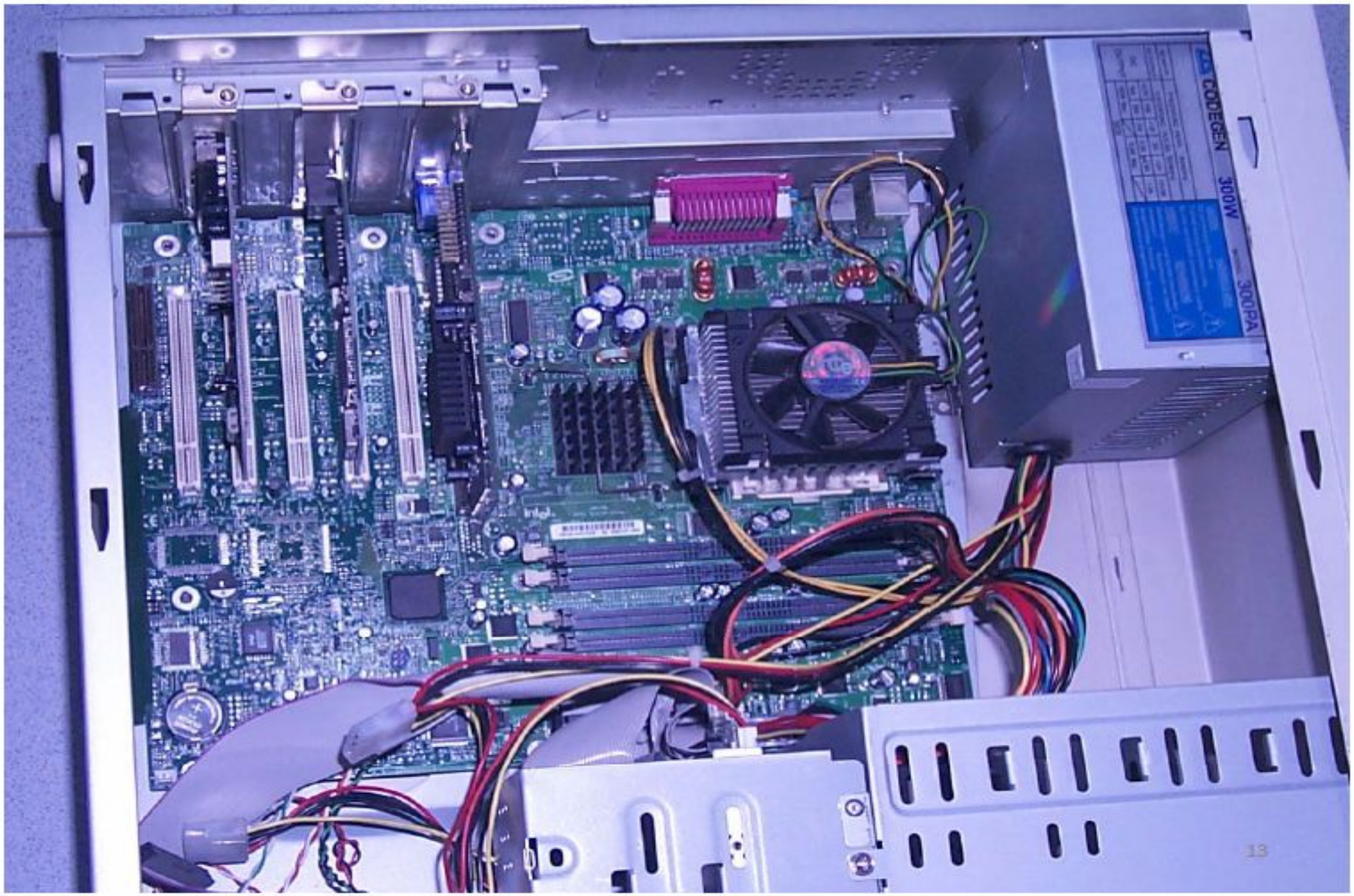
Hardware abstraction

- The figure shows how CPU is connected with input and output device along with their controllers.
- A controller is a hardware that controls the communication between the CPU and the hardware.
- A CPU has
  - PC: Program Counter, Keep record of current instruction.
  - Register file: Store operand currently in use.
  - ALU: Arithmetic Logical unit, operations like arithmetic , relational, logical, comparison etc.
- Bus interface connects CPU to I/O and memory

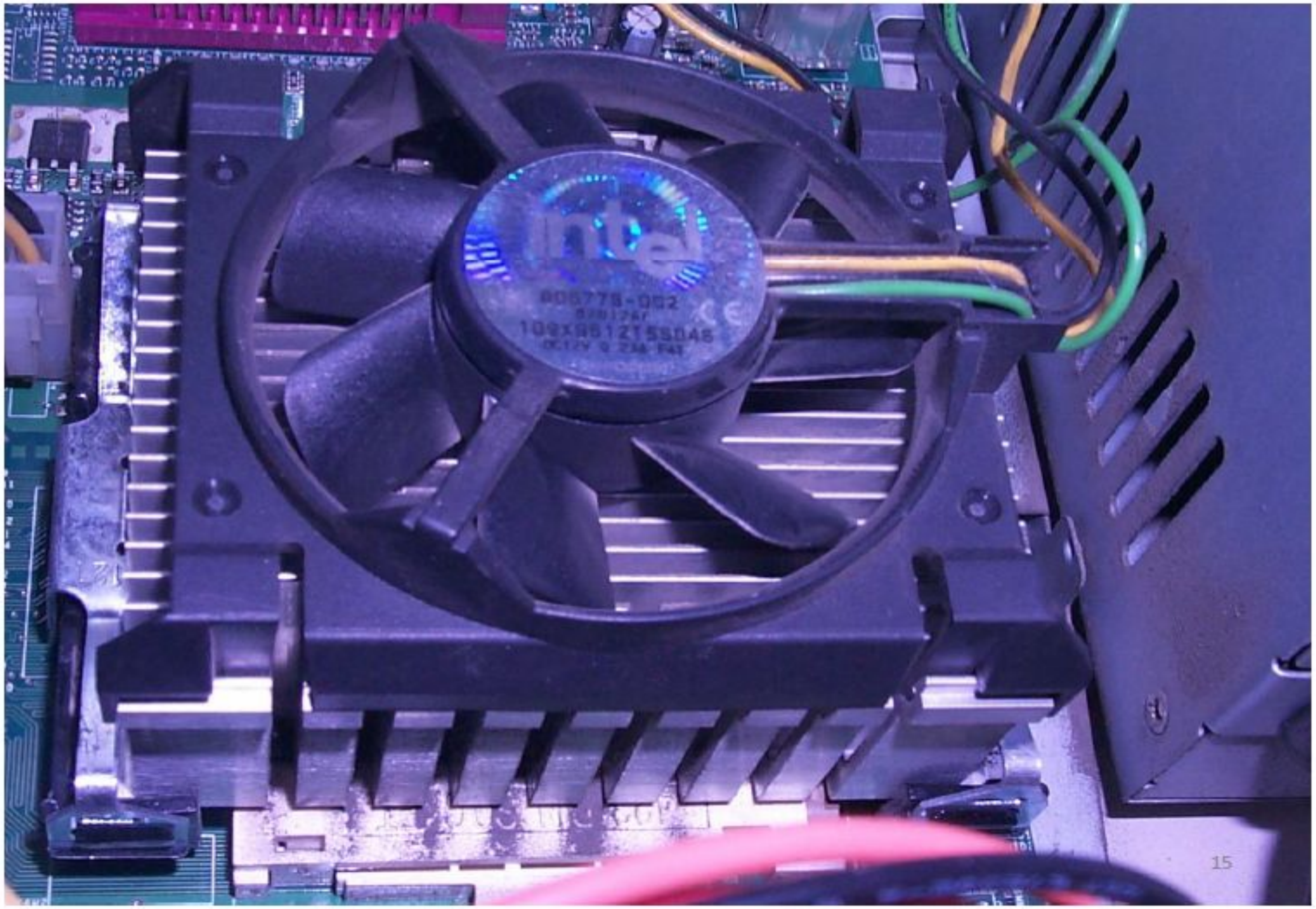
Hardware abstraction

- Bridge connects CPU, main memory and I/O devices via system , memory and I/O bus.
- Bridge communicates with processor and controls interaction with memory , bus etc
- ALU further contains GATES of different types like AND gate, OR gate, NOR gate ,NOT gate and many more.
- GATES further consist of transistors and registers.

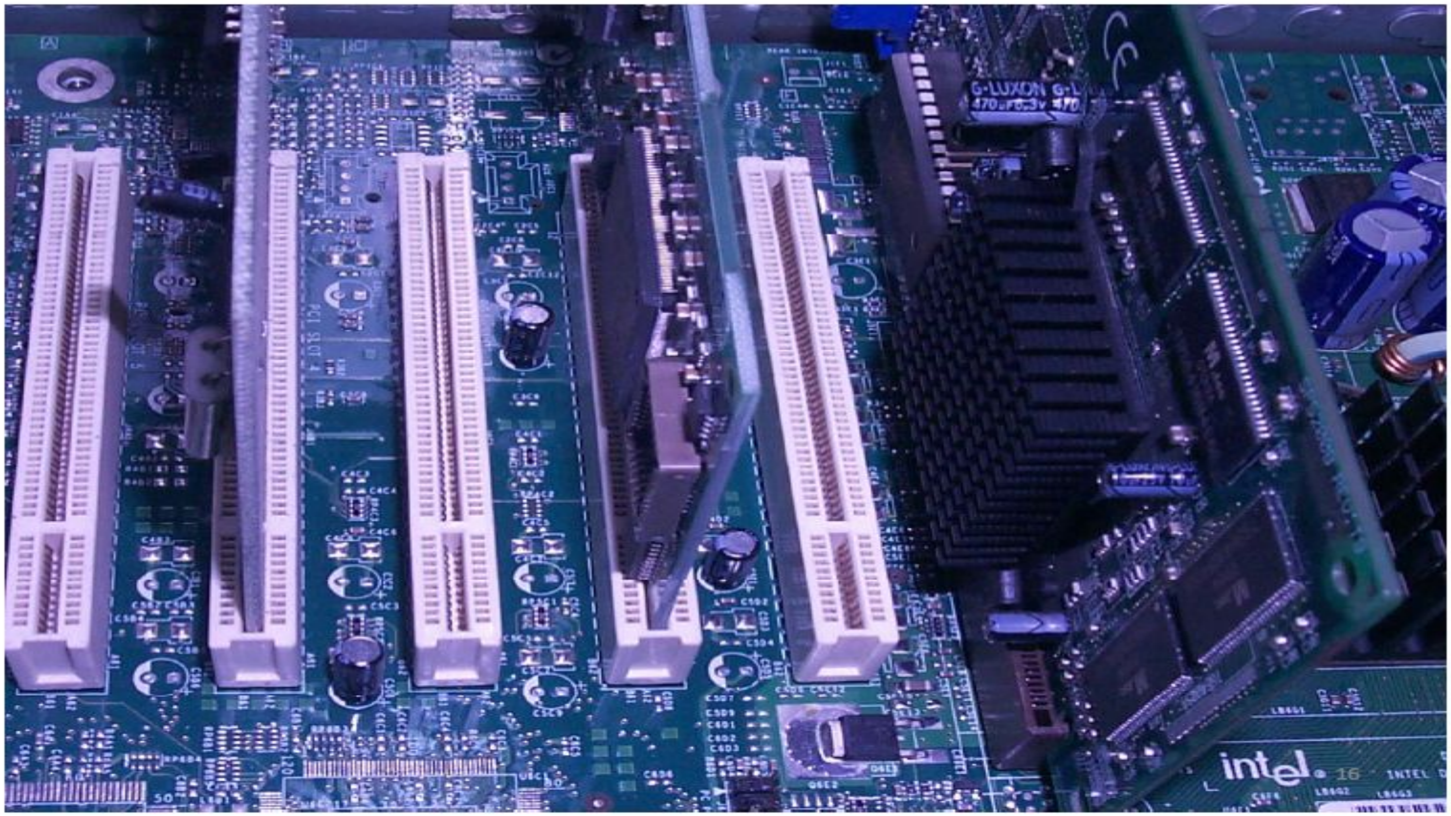
Motherboard & Other components



Looking Closer at processor



Pci (peripheral component interconnect) slots

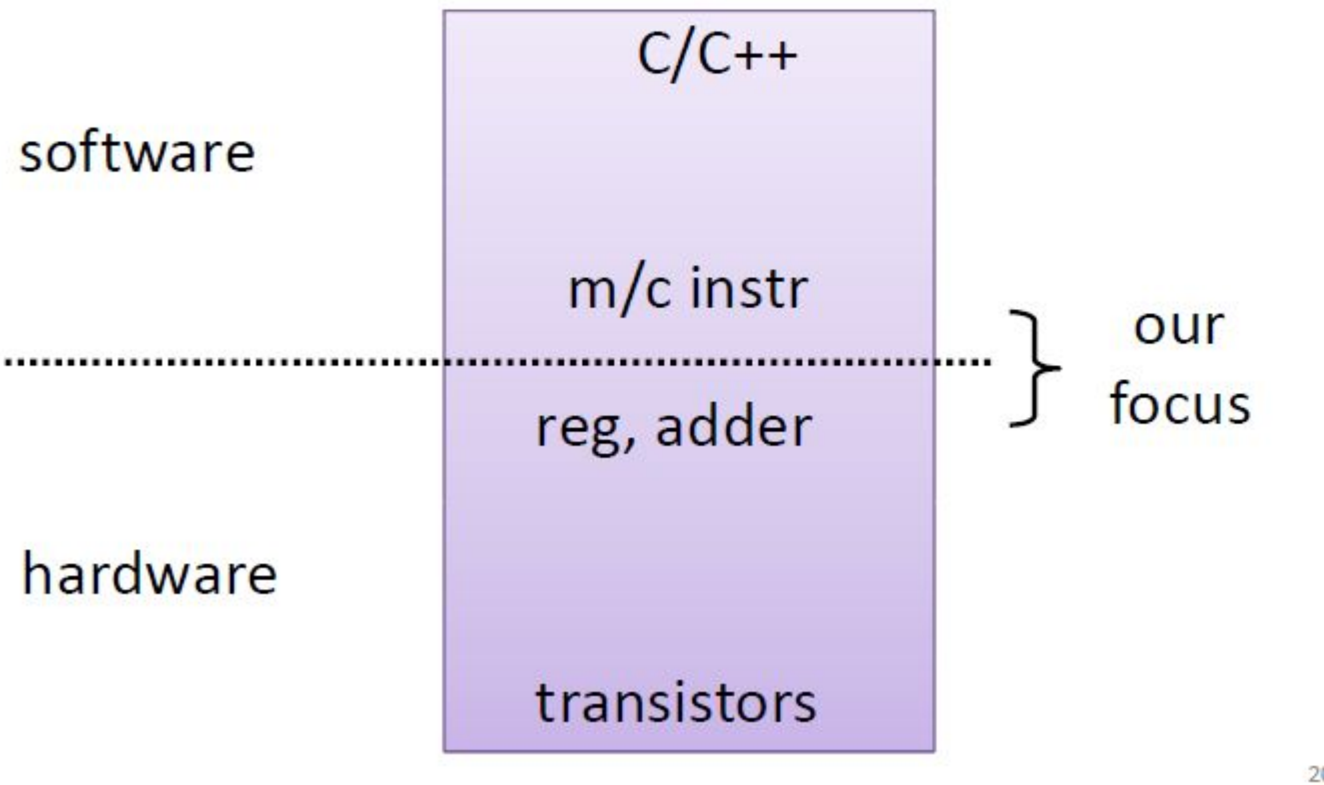


Main memory


ALLIANCE
9702
A00
9704
A00
MAX. RETAIL PRICE
INCL. OF ALL TAXES

Looking closer at hard disk controller

Hardware software interface



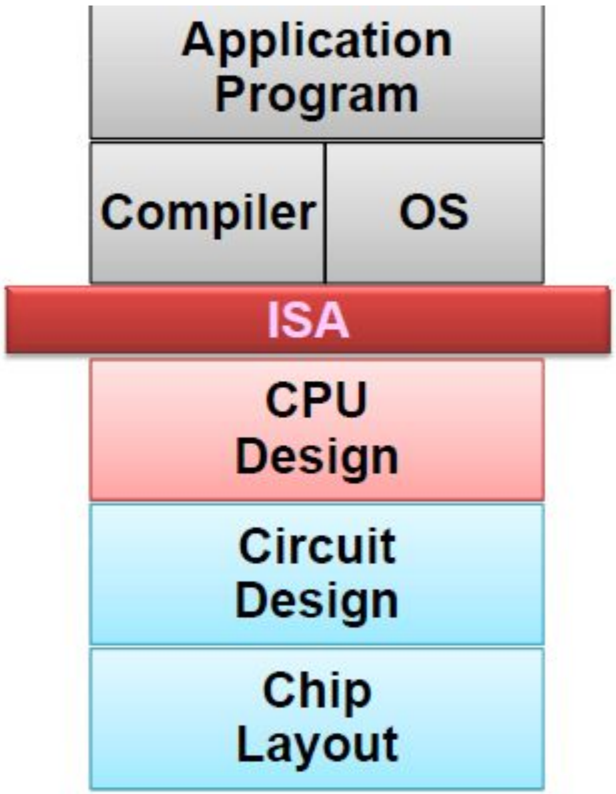
Hardware software interface

- Our focus in computer architecture is on hardware software boundary.
- It is where you have a set of instructions which define the basic capability of a processor and major hardware components which are able to understand those instructions.
- So, if you are a programmer you will see the machine defined by a set of instructions whereas if you are a hardware designer you will see software in terms of those machine instructions which you need to interpret.
- So there are levels of hierarchy here within hardware and level of hierarchy within software

Architecture levels

- Instruction set architecture
  - Lowest level visible to programmer
  - The basic unit of computation is instruction
- Micro architecture
  - Fills gaps between instructions and logic modules.
  - It deals with how information will flow from instruction given by programmer to Logic modules(e.g. ALU, registers etc)
  - Hardware designer is more concerned about micro architecture.

Instruction set architecture



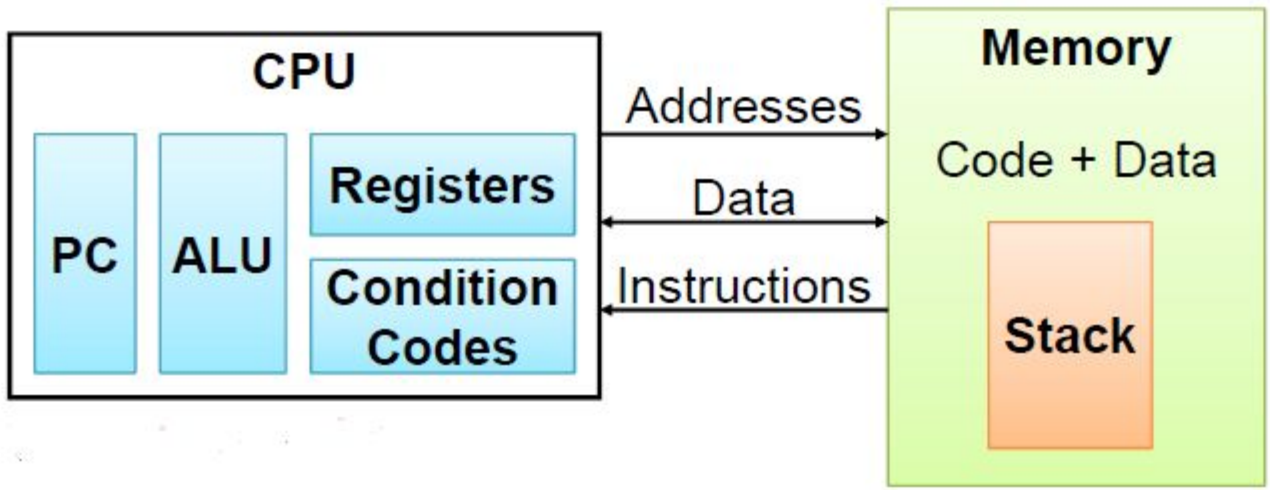
<u>Layers of Abstraction</u>(In terms of software)

- Above – how to program machine- higher level language, OS
- Lower level
    - You have chips, transistors that form gates, gates further form combinatorial circuits, which are kept in ALU.
    - ALU, PC, registers are further arranged in a circuit, which in a broader sense form CPU

Instruction set architecture

- Assembly Language view in terms of hardware
  - Defines how processor state changes from instruction to instruction.
  - The state is defined in terms of
    - contents of its memory,
    - contents of various registers and flip-flops which are there in the processor
  - The programmer at assembly language has to also worry about how instructions are represented;
  - The architect has to worry how these all components will be arranged so that instructions execute efficiently.

The abstract machine



The above figure depicts the basic principle on which computer works

The abstract machine

- Memory contains code or program in machine language, data and stack.
- CPU contains program counter, ALU, registers and condition codes
- Condition codes could be a part of register file or could be separate depending on processor.
- Help in decision making by providing branches and loops in high level languages.

Embedded computers

- An **embedded** system is a **computer** system with a dedicated function within a larger mechanical or electrical system, often with real-time **computing** constraints.
- It requires to design hardware as well as software and so is a tough task.
- Real time operation
- Need highly customized hardware software co-design

Few Examples of Embedded system

- ATM
- Microwave oven
- Digital alarm clock
- Smart card

Why different processors?

- What is the difference between processors used in desk-tops , laptops , mobile phones, washing machines etc?
  - Performance/Speed
  - Power Consumption
  - Cost
  - General purpose/special purpose.

Example of how processors differ

Left Side:
Pentium general purpose processor

Right side:
Micro controller used in embedded device

Five Generations of Computers

- History of computer development divided into 5 generations
- Each generation characterized by a major technological development
- Fundamental changes in terms of
  - Size
  - Cost
  - Power
  - Efficiency
  - Reliability

First Generation – 1940's and 50's: Vacuum Tubes

- Expensive, bulky, unreliable, power guzzlers
- Used punched cards/tapes, magnetic drum memories, machine language

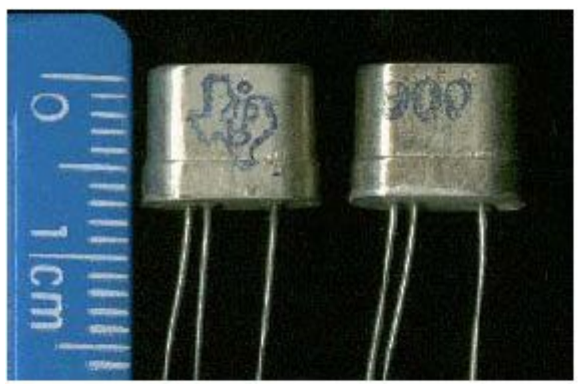
Computer using Vacuum tubes

- **UNIVAC I : (UNIVersal Automatic Computer)**

Speed:	1,905 ops / second
Input/ output:	mag tape, printer
Memory size:	1,000 12-digit words in delay lines
Techno-logy:	vacuum tubes, delay lines, magnetic tape
Floor space:	943 cubic feet
Cost:	$750K + $185K for a high speed printer

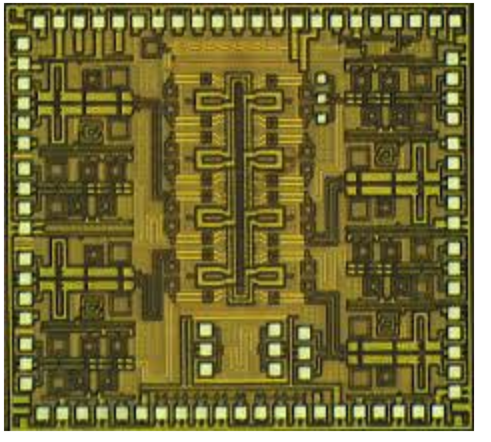
Second Generation – 1950's and 60's: Transistors

- Smaller, faster, cheaper, more energy-efficient and more reliable as compared to vacuum tubes
- Assembly languages, early versions of FORTRAN and COBOL



Third Generation – 1960’s and 70’s: Integrated Circuits

- SSI, MSI, LSI
- Speed and efficiency drastically increased
- Keyboards and monitors
- Operating systems

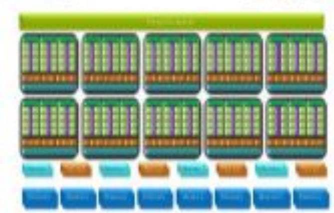


Fourth Generation – 1970’s to Present: Microprocessors

- LSI and VLSI
- Made home computing and embedded computing possible
- Graphics and mouse
- Hand held devices

Fifth Generation - Present and Beyond: Artificial Intelligence

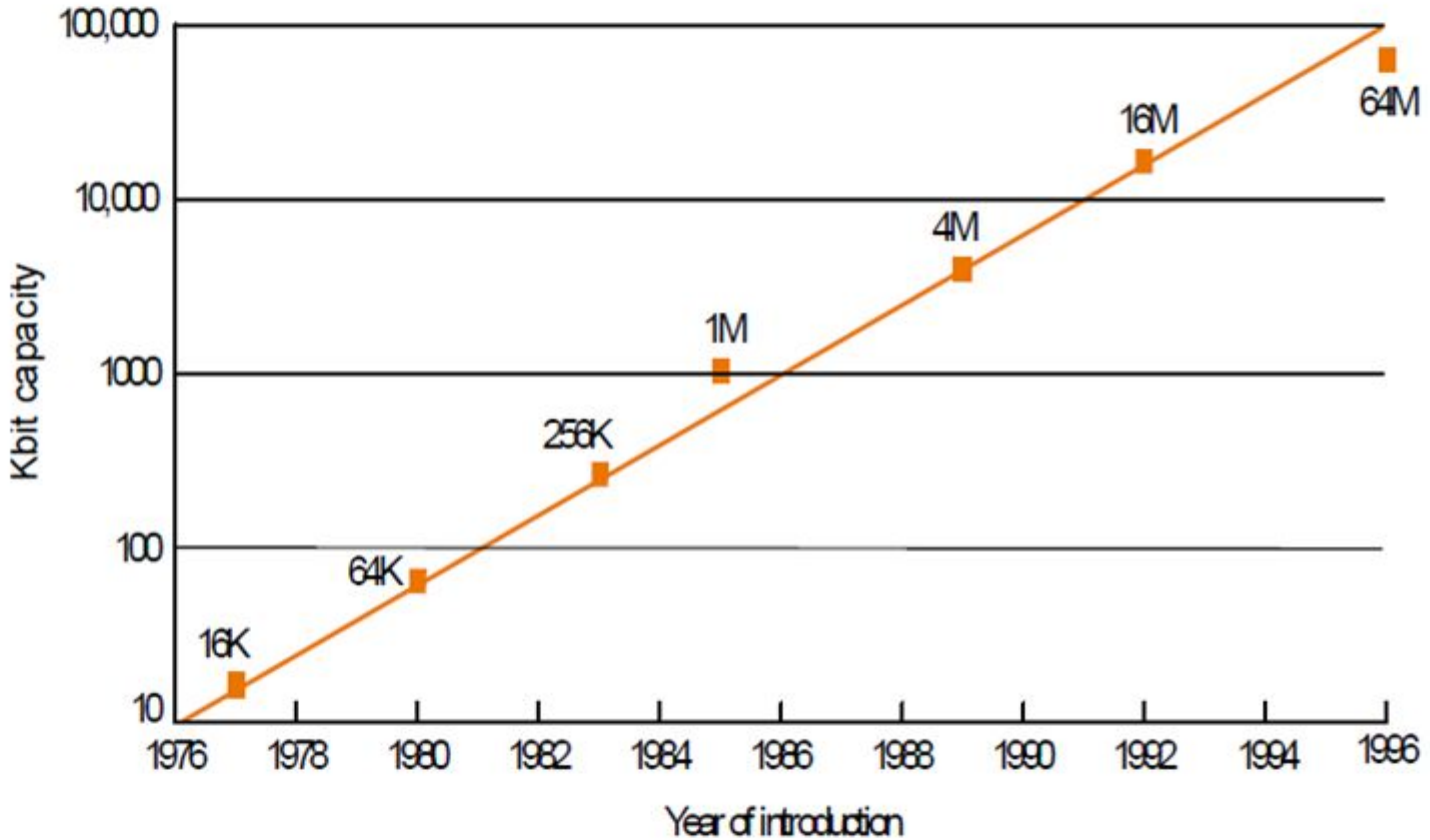
- Voice input/output
- Natural language input/output
- Parallel computing
- Dual Core/Quad Core
- Centrino(intel's wifi adapter),
- Atom
  - Low voltage range CPU from intel
  - Used in embedded applications, robotics, mobile internet device etc
- GPU



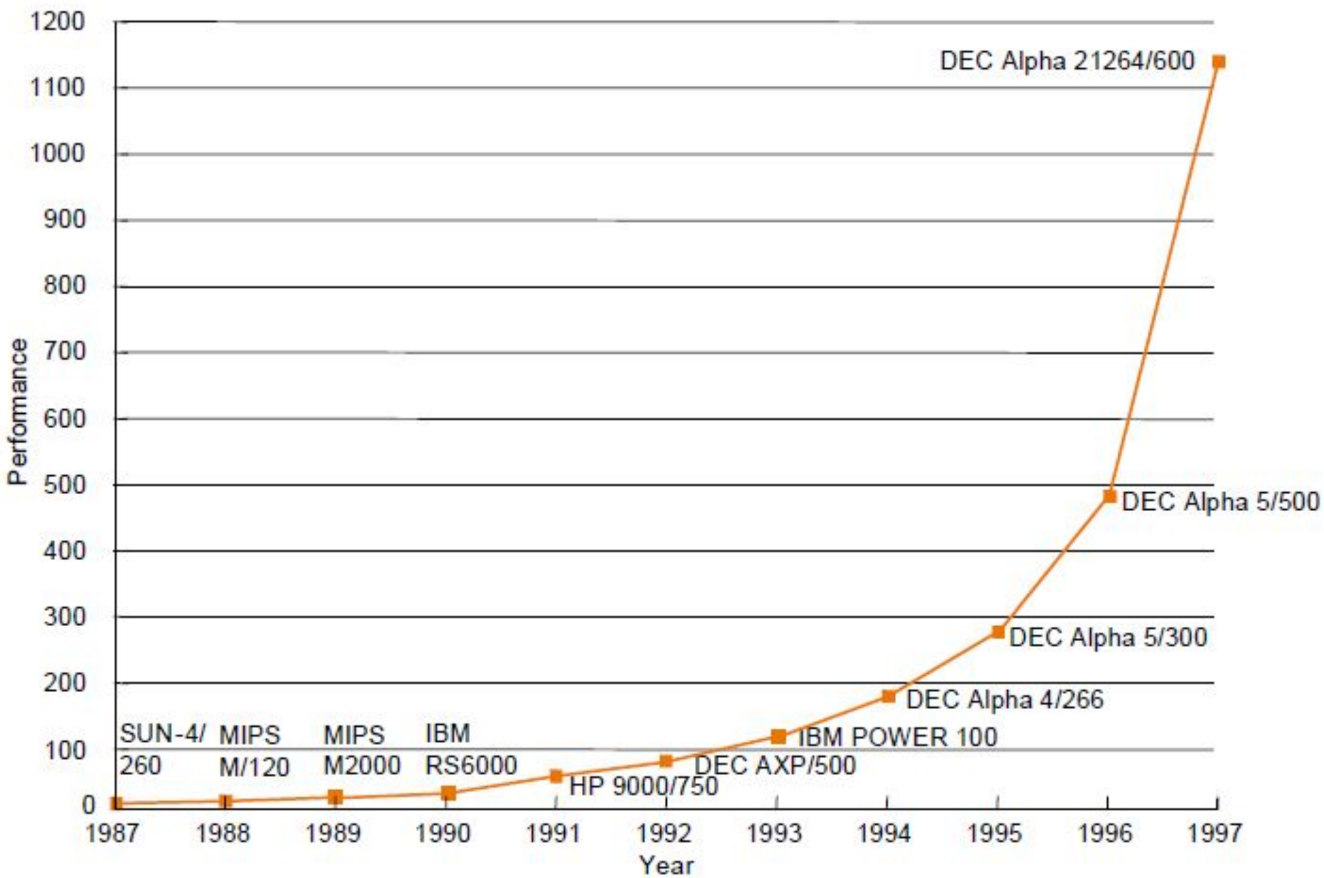
Relative performance per unit cost

Year	Technology	Performance/Cost
1951	Vacuum Tubes	1
1965	Transistors	35
1975	Integrated Circuit	900
1995	VLSI	2,400,000

Growth in DRAM Capacity



Increase in workstation performance



Looking into future

- Grid computing
  - Integrate computer all over world to perform a single task.(e.g. SETI)
- Nano technology
  - Manipulation of matter on atomic level(ranging from size of 1 to 100 nanometres)
- DNA computing(Deoxyribonucliec acid)
  - DNA present in our body are faster than some of the fastest computers available today
  - So research is being made to integrate DNA in computation and make a bio chip.
  - If done successfully we will have more computing power,more memory.
- Quantum computing

Cont..

- Quantum Computing
  - Proposed in 1970
  - Relies on quantum physics properties of atoms or nuclei that allow them to work together as quantum bits or qubits to be computer's processor and memory.
  - Qbits can perform certain calculations exponentially faster then conventional computers.