Computer Systems Architecture

Guoping Qiu

School of Computer Science
The University of Nottingham

http://www.cs.nott.ac.uk/~qiu

The World of Computers

- Computers are everywhere
  - Cell phones
  - Game consoles (PSP, GBA, PS2, Xbox …)
  - Automobiles
  - Home appliances
  - Desktops
- Each uses software

Fundamental Question

Why should someone interested in building software study computer architecture to learn about the organization of the underlying hardware?

Why Study Computer Architecture?

- Makes it possible to write computer programs that are:
  - Faster
  - Smaller
  - Less prone to errors
- Allows programmers to appreciate relative cost of operations and the effect of programming choices
- Helps programmers debug

The bad news

- Digital hardware
  - Is complex
  - 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

Course content

- Basics
  - A taste of digital logic
  - Data representations
- Processors
  - Types of processors
  - Instruction sets and operands
  - Assembly languages and programming

Course content (contd.)

- Memory
  - Storage mechanisms
  - Physical and virtual memories and addressing
  - Caching
- Input/Output
  - Devices and interfaces
  - Buses and bus address spaces
  - Role of device drivers
- Advanced topics

Computer Architecture

- Refers to overall organization of computer system
- Analogous to blueprint
- Specifies
  - Functionality of major components
  - Interconnection among components
- Abstracts away details

Design

- Needed before a computer can be built
- Translates architecture into practice
- Fills in details that architectural specification omits
- Specifies items such as
  - How components are grouped onto boards
  - How power is distributed to boards
- Many designs can satisfy a given architecture

Summary

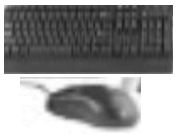
- Understanding architecture helps programmers
- Course covers essentials of computer architecture
  - Digital logic
  - Processors, memory, I /O
  - Advanced topics such as parallelism and pipelining
- We will omit details and focus on concepts

Questions

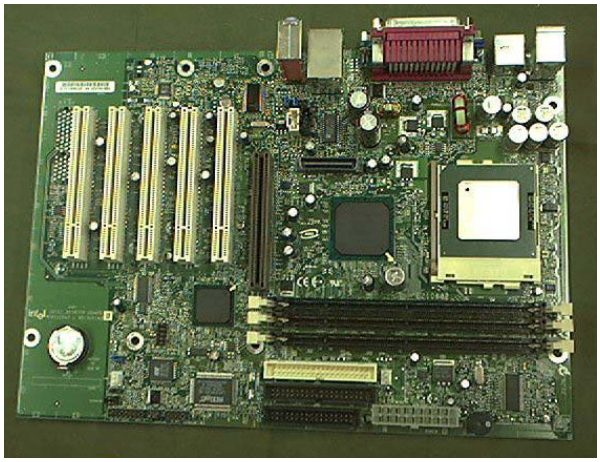
Computer System: User's View

Computer System Components: High Level View

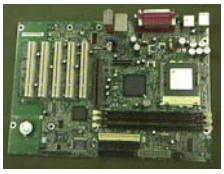
Input (Peripheral) ⟷ **Computer** ⟷ **Output** (Peripheral)

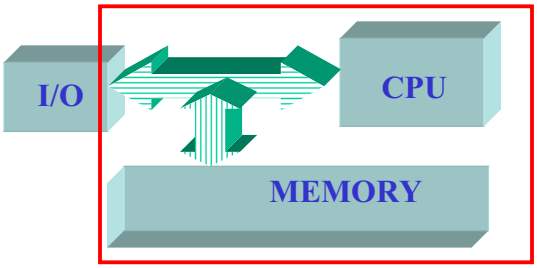


Computer System: Motherboard Level

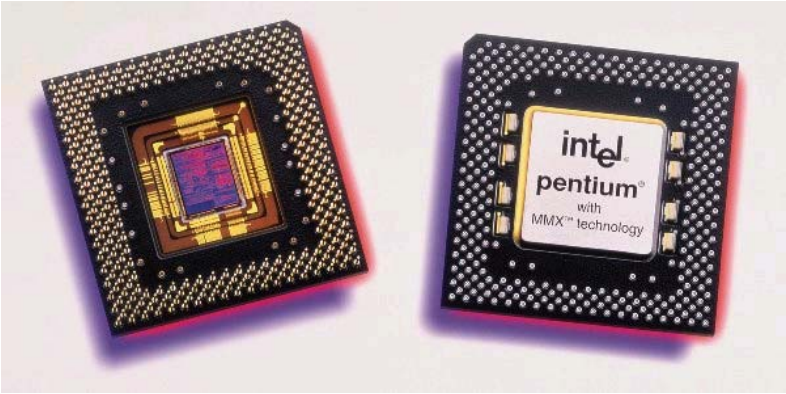


Computer Components: Interconnection

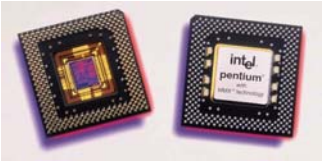




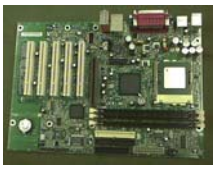
CPU

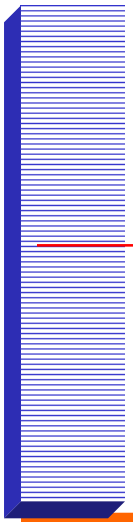


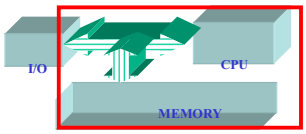
CPU Organization



Memory

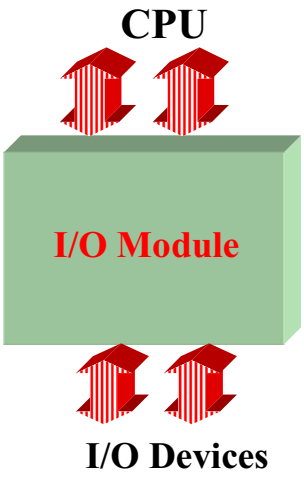




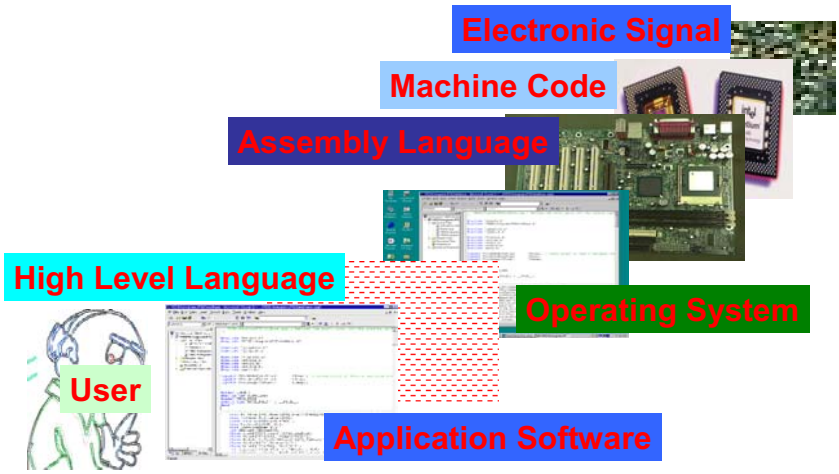


address	content
0000000000	01010101010010101
0000000001	01110101010010101
1111111110	01010101011110101
1111111111	11010111010010101

Input/Output

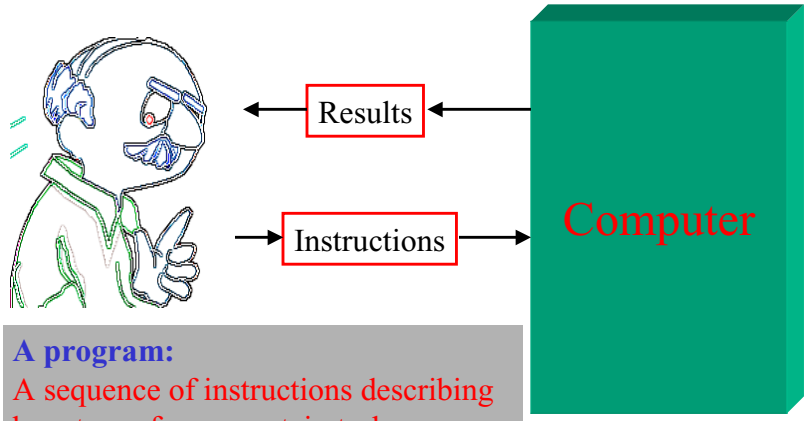


Computer Systems Hierarchy



Computer Systems Hierarchy

A digital computer solves problems by carrying out instructions



A program:
A sequence of instructions describing how to perform a certain task.

Computer Systems Hierarchy

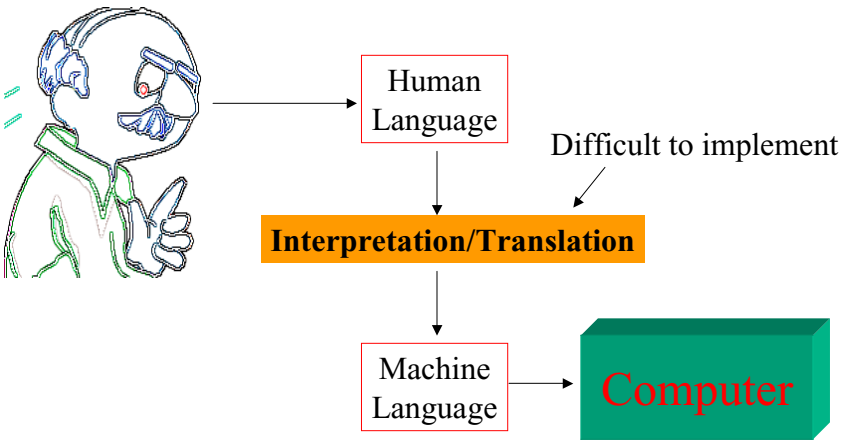
The electronic circuits of a computer can recognize and directly execute a limited set of simple instructions, which are no more complicated than

Add 2 numbers
Check a number to see if it is zero
Copy data from one part of the memory to another part

Machine Language:
A computer's primitive instructions form a language which enables humans to communicate with computers

Machine languages are simple, but difficult to use.

Computer Systems Hierarchy



Computer Systems Hierarchy

Human Language

↓

Machine-like/Human-like language

↓

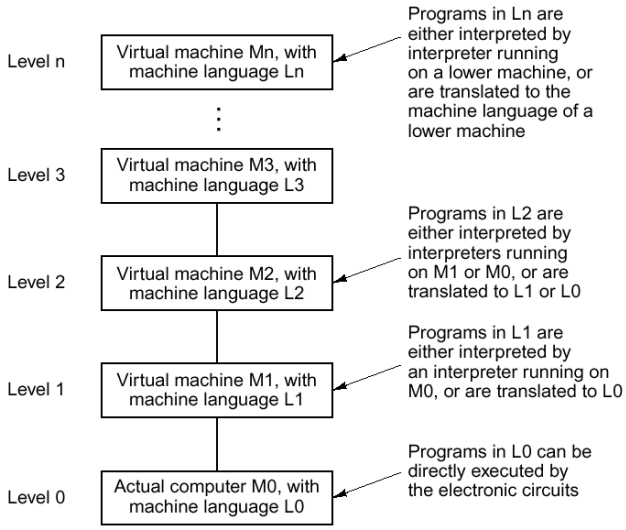
Interpretation/Translation

↓

Machine Language → Computer

Computer Systems Hierarchy

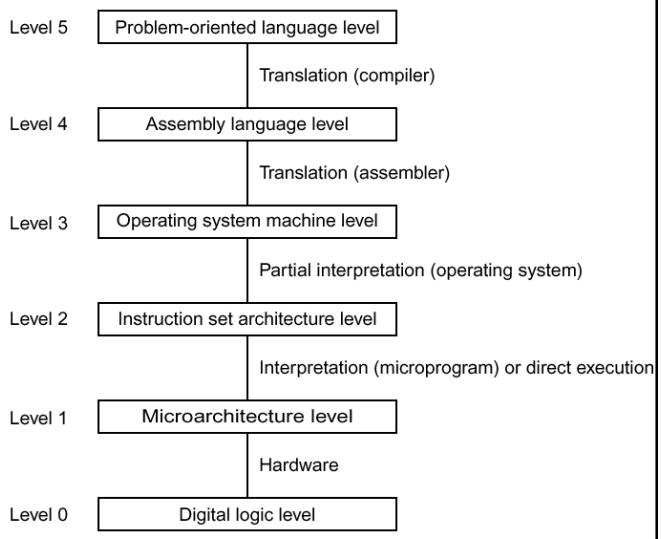
Multilevel virtual machine



Source:
Structured Computer Organization
by Andrew Tanenbaum
Prentice-Hall

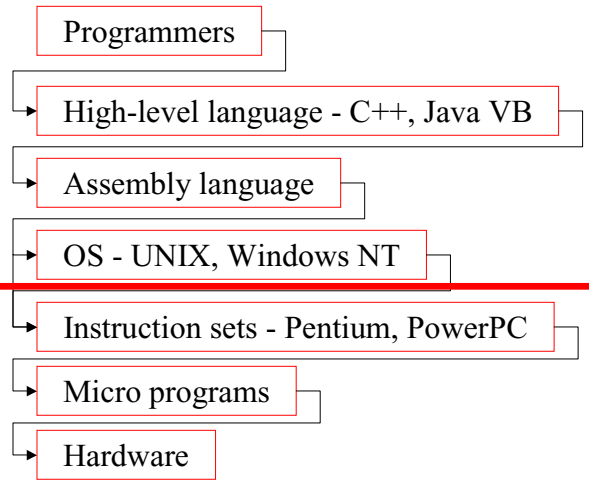
Computer Systems Hierarchy

A 6-level computer



Source:
Structured Computer Organization
by Andrew Tanenbaum
Prentice-Hall

Computer Systems Hierarchy

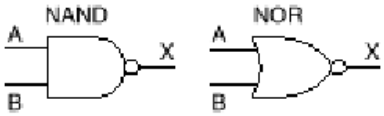


Systems programmers

Computer Systems Hierarchy

Level 0 Digital logic level

Gates AND, OR, NOT, NAND, NOR, XOR etc.



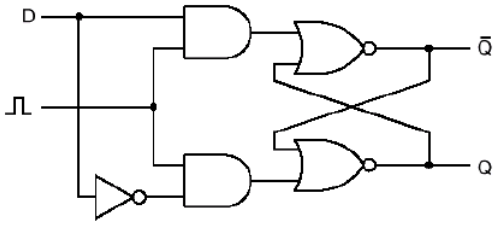
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

Computer Systems Hierarchy

Level 0 Digital logic level

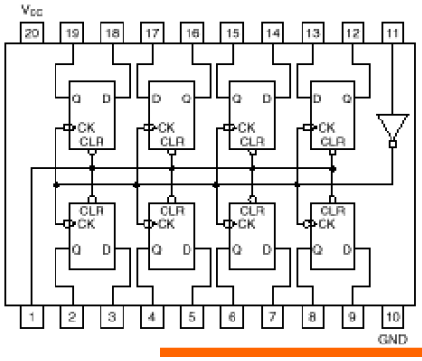
Memory - A small number of gates can form a 1-bit memory



Computer Systems Hierarchy

Level 0 Digital logic level

A group of 1-bit memories combined to form registers.
A register can hold 8 bits, 16 bits, 32 bits or 64 bits etc



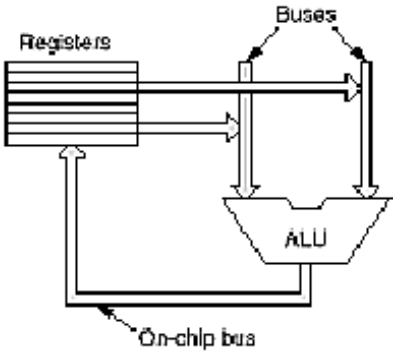
Computer Systems Hierarchy

Level 1 Microstructure level

Registers - 16 bits, 32 bits etc

ALU - Arithmetic Logic Unit

Microprograms



Computer Systems Hierarchy

Level 2 Instruction Set Architecture

intel

CHAPTER 25
INSTRUCTION SET

This chapter presents the instructions in alphabetical order. For each instruction, the forms are given for each operand combination, including object code produced, operands required, execution time, and a description. For each instruction, there is an operational description and a summary of exceptions generated.

25.1. OPERAND-SIZE AND ADDRESS-SIZE ATTRIBUTES

When executing an instruction, the processor can address memory using either 16 or 32-bit addresses. Consequently, each instruction that uses memory addresses has associated with it an address-size attribute of either 16 or 32 bits. The use of 16-bit addresses implies both the use of 16-bit displacements in instructions and the generation of 16-bit address offsets (segment relative addresses) as the result of the effective address calculations. 32-bit addresses imply the use of 32-bit displacements and the generation of 32-bit address offsets. Similarly, an instruction that accesses words (16 bits) or doublewords (32 bits) has an operand-size attribute of either 16 or 32 bits.

The attributes are determined by a combination of defaults, instruction prefixes, and (for programs executing in protected mode) size-specification bits in segment descriptors.

intel

Pentium® Processor Family Developer's Manual

Volume 3: Architecture and Programming Manual

NOTE: The *Pentium® Processor Family Developer's Manual* consists of three books: *Pentium® Processor* Order Number 241428, the *82496/82497/82498 Cache Controller and 82491/82492/82493 Cache SRAM*, Order Number 241429, and the *Architecture and Programming Manual*, Order Number 241430. Please refer to all three volumes when evaluating your design needs.

1995

Computer Systems Hierarchy

Level 2 Instruction Set Architecture

INSTRUCTION SET

intel

AAM—ASCII Adjust AX after Multiply

Opcode	Instruction	Clocks	Description
D4 0A	AAM	18	ASCII adjust AX after multiply

Operation

```
tmpAL ← AL;
AH ← tmpAL / imm8;
AL ← tmpAL MOD imm8;
```

NOTE:

imm8 has the value of the instruction's second byte. The second byte under normally assembly of this instruction will be 0A, however, explicit manipulation of this byte will result in the operation described above and may alter results.

Description

Executes the AAM instruction only after executing a MUL instruction between two unpacked BCD digits that leaves the result in the AX register. Because the result is less than 100, it is contained entirely in the AL register. The AAM instruction unpacks the AL result by dividing AL by the second byte of the opcode, leaving the quotient (most-significant digit) in the AH register and the remainder (least-significant digit) in the AL register.

Flags Affected

The SF, ZF, and PF flags are set according to the result; the OF, AF, and CF flags are unaffected.

Protected Mode Exceptions

None.

Real Address Mode Exceptions

None.

Virtual 8086 Mode Exceptions

None.

25-24

Computer Systems Hierarchy

Level 3 - Operating System

Hybrid level

Most of its instructions are in ISA level - directly carried out by microprograms or hardwired control

Also there are

A set of new instructions
Different memory organization
Run two or more programs concurrently
etc

Computer Systems Hierarchy

Level 4 - Assembly language level

- Symbolic form of the underlying language
- A method to write programs for level 1, 2, and 3
- Easier to use than machine language

Assembler

A program translates the assembly language into level 1, 2, or 3 language and interpreted by the appropriate virtual or actual machine.

Computer Systems Hierarchy

Level 5 - High level language

Application programmers

BASIC, C, C++, C#, etc

Compiler

Translator which translates a high level language to level 4 and 3 languages

A (Very) Brief History of Computers (I)

The first Generation - Vacuum Tubes (1945 -1955)

ENIAC (1943 - 1946)

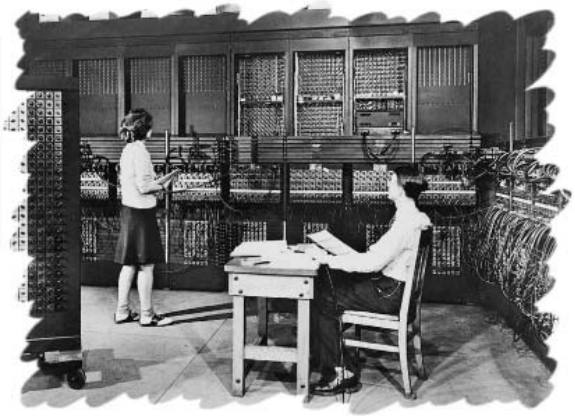
- Intended for calculating range tables of aiming artillery
- Consisted of 18000 tubes, 1500 relays, weight 30 tons, consumed 140 KW
- Decimal machine
- Each digit represented by a ring of 10 vacuum tubes.
- Designed for artillery range table, but used to perform complex calculations to help determine the feasibility of H bomb - general purpose computer
- Programmed with multi-position switches and jumper cables.

John von Neumann (1945 -1952) more later …

- Originally a member of the ENIAC development team.
- First to use binary arithmetic
- Architecture consists of : Memory, ALU, Program control, Input, Output
- Stored-program concept - main memory store both data and instructions

A (Very) Brief History of Computers (II)

Vacuum Tubes



ENIAC

A (Very) Brief History of Computers (II)

The Second Generation - Transistors (1955 -1965)

Transistors

- Transistor was invented in 1948 at Bell Labs by John Barden, Walter Brattain amd William Shockley
- TX-0 (Transistorised eXperimental computer 0), first transistor computer, build at MIT Lincoln Labs
- DEC PDP-1, first affordable microcomputer ($120,000), performance half that of IBM 7090 (the fastest computer in the world at that time, which cost millions)
- PDP-8, cheap ($16,000), the first to use single bus

CDC 6600 (1964)

- an order of magnitide faster than the mighty IBM 7094
- First highly parallelized machine (up to 10 instructions in parallel)
- Separate computational and control units

Burroughs B5000

- First to emphasise software and high level programming languages (Algol 60)

A (Very) Brief History of Computers (III)

The Third Generation - Integrated Circuits (1965 -1980)

IBM System/360
- Family of machines with same assembly language
- Designed for both scientific and commercial computing
- First to allowed microprogramming

.DEC PDP-11
- Was to System/360 what PDP-8 was to 7090
- Very popular with universities, maintained DEC's lead in microcomputer market

The Fourth Generation - VLSI (1980 - ?)

- Lead to PC revolution
- High performance, low cost

Evolution of Intel Microprocessor

Source: http://www.intel.com/intel/museum/25anniv/hof/tspecs.htm

1970s Processors

	4004	8008	8080	8086	8088
Introduced	11/15/71	4/1/72	4/1/74	6/8/78	6/1/79
Clock Speeds	108KHz	200KHz	2MHz	5MHz, 8MHz, 10MHz	5MHz, 8MHz
Bus Width	4 bits	8 bits	8 bits	16 bits	8 bits
Number of Transistors	2,300 (10 microns)	3,500 (10 microns)	6,000 (6 microns)	29,000 (3 microns)	29,000 (3 microns)
Addressable Memory	640 bytes	16 KBytes	64 KBytes	1 MB	1 MB
Virtual Memory	--	--	--	--	--
Brief Description	First microcomputer chip, Arithmetic manipulation	Data/character manipulation	10X the performance of the 8008	10X the performance of the 8080	Identical to 8086 except for its 8-bit external bus

Evolution of Intel Microprocessor

Source: http://www.intel.com/intel/museum/25anniv/hof/tspecs.htm

1980s Processors

	80286	Intel386™ DX Microprocessor	Intel386™ SX Microprocessor	Intel486™ DX CPU Microprocessor
Introduced	2/1/82	10/17/85	6/16/88	4/10/89
Clock Speeds	6MHz, 8MHz, 10MHz, 12.5MHz	16MHz, 20MHz, 25MHz, 33MHz	16MHz, 20MHz, 25MHz, 33MHz	25MHz, 33MHz, 50MHz
Bus Width	16 bits	32 bits	16 bits	32 bits
Number of Transistors	134,000 (1.5 microns)	275,000 (1 micron)	275,000 (1 micron)	1.2 million (1 micron) (.8 micron with 50MHz)
Addressable Memory	16 megabytes	4 gigabytes	16 megabytes	4 gigabytes
Virtual Memory	1 gigabyte	64 terabytes	64 terabytes	64 terabytes
Brief Description	3-6X the performance of the 8086	First X86 chip to handle 32-bit data sets	16-bit address bus enabled low-cost 32-bit processing	Level 1 cache on chip

The University of Nottingham School of Computer Science G51CSA

43

Evolution of Intel Microprocessor

Source: http://www.intel.com/intel/museum/25anniv/hof/tspecs.htm

1990s Processors

	Intel486™ SX Microprocessor	Pentium® Processor	Pentium® Pro Processor	Pentium® II Processor
Introduced	4/22/91	3/22/93	11/01/95	5/07/97
Clock Speeds	16MHz, 20MHz, 25MHz, 33MHz	60MHz,66MHz	150MHz, 166MHz, 180MHz, 200MHz	200MHz, 233MHz, 266MHz, 300MHz
Bus Width	32 bits	64 bits	64 bits	64 bits
Number of Transistors	1.185 million (1 micron)	3.1 million (.8 micron)	5.5 million (0.35 micron)	7.5 million (0.35 micron)
Addressable Memory	4 gigabytes	4 gigabytes	64 gigabytes	64 gigabytes
Virtual Memory	64 terabytes	64 terabytes	64 terabytes	64 terabytes
Brief Description	Identical in design to Intel486™ DX but without math coprocessor	Superscalar architecture brought 5X the performance of the 33-MHz Intel486™ DX processor	Dynamic execution architecture drives high-performing processor	Dual independent bus, dynamic execution, Intel MMX™ technology

The University of Nottingham School of Computer Science G51CSA

44

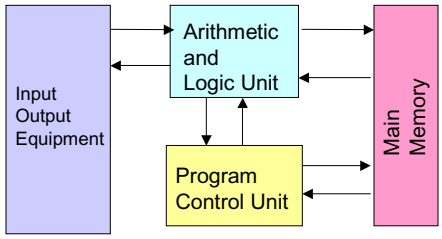
Moore's Law

Moore's Law

Computers double in power roughly every two years, but cost only half as much

The IAS (von Neumann) Machine

Stored Program concept
Main memory storing programs and data
ALU operating on binary data
Control unit interpreting instructions from memory and executing
Input and output equipment operated by control unit

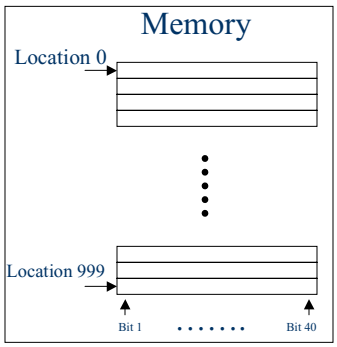


The Structure of IAS Computer

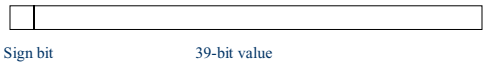
1946 ~ 1952
John von Neumann
Princeton
Institute for Advanced Studies

Almost all of today's computers have the same general structure as the IAS - referred to as von Neumann machines.

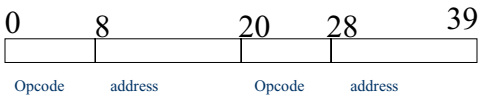
The IAS Machine: Memory



IAS Memory
- 1000 storage locations (words), each 40 bits
- Both data and instruction are stored in the memory



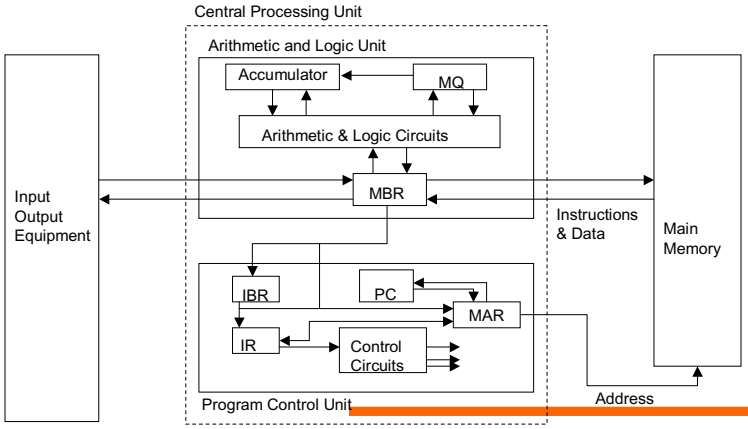
Number word



Instruction word

The IAS Machine: Control Unit

The control unit operates the machine by fetching instructions from memory and executing them **ONE** at a time.



The IAS Machine: Instruction Cycle

The IAS operates by repetitively performing an ***instruction cycle***.

Two sub-cycles:

- During the ***fetch cycle***, the opcode of the NEXT instruction is loaded in to the IR and the address portion is loaded into the MAR

- Once the opcode is in the IR, the ***execute cycle*** is performed. Control circuitry interprets the opcode and executes the instruction by sending out appropriate control signals to cause data to be moved or an operation to be performed by the ALU.