Computer Architecture

Computer History

Vacuum tubes were made by John Ambrose Fleming in 1904 and weren't widely used until 1910
Vacuum tubes were also known as thermionic valves
Initially vacuum tubes were used in radios but their use dramatically increased during WW2
Vacuum tube uses diodes, switch, rectifier and amplifier
It could be used in devices to convert AC to DC, convert voltages or to construct logic gates
vacuum tubes control the flow of electricity and operate on the principles of thermionic emission tubes control flow as in a flip flop circuit
Digital computing relies on the power of millions of binary function
vacuum tubes were faster than the earlier mechanical relays
CRT = Cathode Ray Tube (used and modified in university of Manchester in 1948)
Cathode ray tubes were also utilized as computer memory in early computers and became known as William Kilburn tubes
Computers from 1940 - 1956 were known as first generation computers; massive vaccum tubes were used in first generation computers

COLOSSUS 2 Computers

Began operation in 1943
It contained 1600 vacuum tubes

ENIAC = Electronic Numerical Integrator And Computer

Began operation in 1946
It contained 17400 vacuum tubes
A large scale general purpose electronic data computer, it could perform 5000 simple calculations per second
It weighed 30 tons and spanned 80 feet's

RCA BIZMAC

Began operation in 1956
it contained 25000 vacuum tubes

SAGE (World's Largest Computer System)

Development period 1953 - 1958
Full deployment = 1963
Decommissioned = 1983 - 1984 (overall 22 were constructed)
Contained nearly 60000 vacuum tubes, 17500 diodes and 12000 transistors

Harwell Dekatron Computer (1951 United Kingdom)

Also known as "The Witch"; Wolverhampton Instrument for Teaching Computing from Harwell
Uses 800 dekatron tubes for memory storage
World's oldest working programmable computer

UNIVAC 120

It contained approx. 800 tubes
90000 USD in late 1950.

LEO 1 (first business computer)

a British catering firm created Leo standing for Lion Electronic Office
Released in 1951 it became the first business computer to run world office job
It uses 5936 tubes or valves plus another 400 for improvable equipment
It uses 64 mercury tubes for storage each 5 feet long and weighing over 1000 pounds
Leo had a speaker installed that created sounds as it calculated ant that how digital music was first introduced
a unique model design provided an easy access to its components during failures including tube replacement
Leo computers limited in 1954

UNIVAC 1

The Univac 1 contained 13 different tube types
it contained 5600 tubes , 18000 crystal diodes and 300 relays
it was 25 ft by 50 ft in length
it used about 4500 tubes and 32 chips
----> SWAC = Standard Western Automatic Computer , contained 37 William tubes and 2300 vacuum tubes
THYRATOM= a thyratom is a type of gas filled tubes used as high power electrical switch and controlled rectifier. They can handle much greater currents than similar hard vacuum tubes

DATA REPRESENTATION

Data is represented in 4 ways
---> Decimal number system

It has ten digits ranging from 0-9. Because this system has ten digits; it is also called a base ten number system or denary number system.> Decimal number system

---> **Binary number system**

It uses two digits namely, 1 and 0 to represent numbers. unlike in decimal numbers where the place value goes up in factors of ten, in binary system, the place values increase by the factor of 2.binary numbers are written as X2.consider a binary number such as (1011)2.The right most digit has a place value of 1×20 while the left most has a place value of 1×23.

---> **Octal number system**

Consists of eight digits ranging from 0-7.the place value of octal numbers goes up in factors of eight from right to left.

---> **Hexadecimal number system**

This is a base 16 number system that consists of sixteen digits ranging from 0-9 and letters A-F where A is equivalent to 10,B to 11 up to F which is equivalent to 15 in base ten system. The place value of hexadecimal numbers goes up in factors of sixteen.

A hexadecimal number can be denoted using 16 as a subscript

Hexadecimal notation: A shorthand notation for long bit patterns

Divides a pattern into groups of four bits each

Represents each group by a single symbol

Example: 10100011 becomes A3

The terms bits, bytes, nibble and word are used widely in reference to computer memory and data size.

Bits: can be defined as either a binary, which can be 0, or 1.It is the basic unit of data or information in digital computers.

today's computers information is encoded as patterns of 0s and 1s. These digits are called bits

Byte: a group of bits (8 bits) used to represent a character. A byte is considered as the basic unit of measuring memory size in computer.

A nibble: is half a byte, which is usually a grouping of 4 bits.

Word: two or more bits make a word. The term word length is used as the measure of the number of bits in each word. For example, a word can have a length of 16 bits, 32 bits, 64 bits etc.

A number system is a set of symbols used to represent values derived from a common base

Converting binary numbers to decimal numbers

- To convert a binary number to a decimal number, we proceed as follows:
- Write each digit under its place value.
- Multiply each digit by its corresponding place value.
- Add up the products. The answer will be the decimal number in base ten.

EXAMPLE

Convert 101101 to base 10(or decimal) number

Place value

25 24 23 22 21 20

Binary digits

1 0 1 1 0 1

Multiply each digit by its place value

N10=(1*25) +(0*24)+(1*23)+(1*22)+(0*21)+(1*20)

N10=32+0+8+4+0+1

=4510

Convert Decimal Numbers to Binary Numbers

- To convert decimal to binary numbers, proceed the steps given below:
- Divide the given decimal number by "2" where it gives the result along with the remainder.
- If the given decimal number is even, then the result will be whole and it gives the remainder "0"
- If the given decimal number is odd, then the result is not divided properly and it gives the remainder "1".
- By placing all the remainders in order in such a way, the Least Significant Bit (LSB) at the top and Most Significant Bit (MSB) at the bottom, the required binary number will obtain.

Hexadecimal digit ; Decimal equivalent ;Binary equivalent

00	00	0000
01	01	0001
02	02	0010
03	03	0011
04	04	0100
05	05	0101
06	06	0110
07	07	0111
08	08	1000

09	09	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Converting Hexadecimal to decimal

7DE is a hex number

7DE = (7 * 162) + (13 * 161) + (14 * 160)

7DE = (7 * 256) + (13 * 16) + (14 * 1)

7DE = 1792 + 208 + 14

7DE = 2014 (in decimal number)

- Prefixing an extra sign bit to a binary number
- In decimal numbers, a signed number has a prefix "+" for a positive number e.g. +2710 and "-" for a negative number e.g.-27
- However, in binary, a negative number may be represented by prefixing a digit 1 to the number while a positive number may be represented by prefixing a digit 0. For example, the 7-bit binary equivalent of 127 is 1111111. To indicate that it is positive, we add an extra bit (0) to the left of the number i.e. (0)1111111.
- To indicate that it is negative number we add an extra bit (1) i.e. (1)1111111.
- The problem of using this method is that the zero can be represented in two ways i.e.(0)00000002 and (1)00000002.

Binary addition

The five possible additions in binary are

- 0 + 0 = 0
- 0+ 1 = 1
- 1 + 0 = 1
- 1 + 1 = 10(read as 0, carry 1)
- 1 + 1 + 1 = 11(read as 1, carry 1)

Binary subtraction

The four possible subtractions in binary are:

- 0 – 0 = 0
- 1 – 0 = 1
- 0– 1 = 0
- 1– 1 = 0
- 10 – 1 = 1 (borrow 1 from the next most significant digit to make 0 become 10 hence 10 – 1 = 1)

One's Compliment:

Transforming the 0 bit to 1 and the 1 bit to 0.

Examples:

- 1's complement of "0111" is "1000"
- 1's complement of "1100" is "0011"

Two's Compliment:

2's complement of a binary number is 1 added to the 1's complement of the binary number.

Examples:

- 2's complement of "0111" is "1001"
- 2's complement of "1100" is "0100"

For one's complement, we simply need to flip all bits.

For 2's complement, we first find one's complement. We traverse the one's complement starting from LSB (least significant bit), and look for 0. We flip all 1's (change to 0) until we find a 0. Finally, we flip the found 0. For example, 2's complement of "01000" is "11000" (Note that we first find one's complement of 01000 as 10111). If there are all 1's (in one's complement), we add an extra 1 in the string. For example, 2's complement of "000" is "1000" (1's complement of "000" is "111").

The main difference between 1' s complement and 2' s complement is that 1' s complement has two representations of 0 (zero) – 00000000, which is positive zero (+0) and 11111111, which is negative zero (-0); whereas in 2' s complement, there is only one representation for zero – 00000000 (+0) because if we add 1 to 11111111 (-1), we get 00000000 (+0) which is the same as positive zero. This is the reason why 2' s complement is generally used. Another difference is that while adding numbers using 1' s complement, we first do binary addition, then add in an end-around carry value. But, 2' s complement has only one value for zero, and doesn't require carry values.

Charles Babbage ----> Father of Computer

Objects can be represented in binary and can be made to interact through binary

Ada Lovelace ----> the first programmer to talk about object oriented programming after seeing Charles's model
Object oriented programming allows us to:

1. Inheritance = inherit properties that link to each other
2. Reusability = it can be replicated and after making small changes used for other purposes
3. Abstraction = not all properties or functions of an object are available. Many of the functions of computer are being done inside the processor and only a few of them are able to be interacted with. Lot of properties are not offered for interaction with other objects. Such properties cannot be accessed by the out world

Class and object are different thing. For object to be created class needs to be defined

Operating system uses the functionality of the processor and provides interface so that it can be communicated
KERNAL is a sleek and core part of operating system which helps in booting up a computer
Not all interfaces are public, some are handled within thus they provide abstraction. Despite knowing we have little control over how it happens , little accessibility, without being involved in every minor interaction in object

STORAGE:

ASCII

ASCII stands for American Standard Code for Information Interchange
ASCII : binary equivalence code / representation of special characters and symbols on keyboard.
8 bit binary sequence computes 1 byte of computer memory
ASCII is a standard that assigns letters, numbers, and other characters in the 256 slots available in the 8-bit code.
The question that needs to be answered is that why do we use 8 bits in ASCII and not 7 bits or less?
The answer is that there are many symbols or characters that are needed to be represented in binary sequence
and if we consider 7 bits for representing these symbols and special characters so it will give only 2^7 = 128, 128 combinations which won't be sufficient whereas 2^8 gives us the possibility of 256 different combinations which enough to cater our need for binary representation of symbols letters along with special characters

UNICODE:

Unicode is another way of storing characters with their binary representation or equivalence
The difference between the Unicode and ASCII is that Unicode uses 2 bytes that is 16 bits to represent different characters. The reason why unicode uses 16 bits is that it also includes symbol based languages since it has possibility of greater combinations

8 bits ----------> 1 bytes
1024 bytes ----> 1 KB
1024 KB ------> 1 MB
1024 MB ------> 1 GB
1024 GB ------> 1 TB
1024 TB ------> 1 PB

When we talk about storage we are basically referring to hard disk

Hard Disk Drive:

A hard disk drive (sometimes abbreviated as a hard drive, HD, or HDD) is a non-volatile data storage device. It is usually installed internally in a computer, attached directly to the disk controller of the computer's motherboard.
Hard disk drive are nonvolatile in nature that controls the positioning, reading and writing of the hard disk, which furnishes data storage which means that whatever is stored on the hard disk drive is not temporary and remain there
This is why you can restart a computer, which powers down the HDD, but retain access to all the data when it's back on.

Hard disk drives are commonly used as the main storage device in a computer. HDDs often store operating system, software programs and other files even when power is turned off.
Inside the hard drive are sectors located on tracks, stored on rotating platters. These platters have magnetic heads that move with an actuator arm to read and write data to the drive.
An interface connects hard disk drive to processor
SSD stands for Solid State Drive (have good speeds)

RAM:

RAM stands for Random Access Memory
RAM does not stores data permanently and is thus volatile in nature
As soon as the power is turned off the content stored in Ram is lost
RAM is essentially a device's short-term memory. It temporarily stores (remembers) everything currently running on a device, like all OS-specific services and any web browser
It's very fast, which makes it ideal for things the computer is actively working on, such as applications that are currently running

RAM stores the date that is being currently worked on so that processor can easily and immediately be provided with work so that we can overcome the speed barrior
RAM -----> Primary storage
Processor -----> Secondary storage
Primary storage refers to the main storage of the computer or main memory which is the random access memory or RAM. Secondary storage, on the other hand, refers to the external storage devices used to store data on a long-term basis such as hard disk drives, solid state drives, removable "USB" drives, CDs, and DVDs.

COMPUTER ARCHITECTURE:
The computer architecture consists of 3 parts

1. Instruction set architecture
2. Microprocessor architecture
3. System architecture

A processor is a very fast machine and has a speed around 2.7GHz to 4 GHz
INSTRUCTION SET ARCHITECTURE
Assembly Language:
Assembly language uses a short descriptive word, known as a mnemonic, to represent each of the machine-language instructions
Assembly languages were developed to make programming easier.
Assembly language is referred to as a low-level language, because assembly language is close in nature to machine language and is machine dependent.
An assembly language is a low-level programming language designed for a specific type of processor. It may be produced by compiling source code from a high-level programming language (such as C/C++) but can also be written from scratch. Assembly code can be converted to machine code using an assembler.
There is no language much more simpler than assembly language
It is a more readable interpretation of processors code allowing easier understanding and programming by human

Assembler : An assembler is a program that converts assembly language into machine code. It takes the basic commands and operations from assembly code and converts them into binary code
The assembler basically creates a Mnemonic for remembering every single instruction and then code with it since I is difficult for programmers to remember every single instruction individually . Remember that these mnemonic are created only for easiness of programmers and the computer still takes instructions in binary
Example:
1100(instruction by assembler) -----> Addition(Instruction in assembly language) -----> add(Mnemonic)
1010(instruction by assembler) ------> Delete(Instruction in assembly language) -----> Del(Mnemonic)

An assembler primarily serves as the bridge between symbolically coded instructions written in assembly language and the computer processor, memory and other computational components. An assembler works by assembling and converting the source code of assembly language into object code or an object file that constitutes a stream of zeros and ones of machine code, which are directly executable by the processor.
It basically acts like a form of communication between the human and machines

Machine Language:
A computer's native language, which differs among different types of computers, is its machine language—a set of built-in primitive instructions.
These instructions are in the form of binary code,
Machine language, or machine code, is a low-level language comprised of binary digits (ones and zeros)
High-level languages, such as Swift and C++ must be compiled into machine language before the code is run on a computer.
computers are digital devices, they only recognize binary data and hence an assembler is required to convert assembly language to machine language.

High Level Language:
high-level languages are platform independent, which means that you can write a program in a high-level language and run it in different types of machines. High-level languages are similar to English and easy to learn and use. The instructions in a high-level programming language are called statements.
A program written in a high-level language is called a source program or source code. Because a computer cannot execute a source program; it can only execute codes written in binary, a source program must be translated into machine code for execution. The translation can be done using another programming tool called an interpreter or a compiler.

Compiler:
An interpreter reads one statement from the source code, translates it to the machine code or virtual machine code, then executes it right away
A compiler translates the entire source code into a machine-code file, and the machine-code file is then executed
For a high level language a typing error should not occur since it causes a syntax error and then the function won't take place

Assembler is a kind of mnemonic set giving one to one correspondence with the instruction set -----> Binary set of instruction to operate the machine.
Compiler describes set of mnemonic instructions put into one function and that function is given as a keyword to high level language

compiler assembler
HLL ----------> Assembly Language ----------> Instruction set ---------> Processor -------> executes in machine language
Assembler -----> ISA , Every vendor has their own ISA

PROCESSOR ARCHITECTURE:

ALU stands for arithmetic logic unit
HD ----> RAM ----> Cache -----> Processor (memory train)
SSD is nonvolatile, the contents remain same even after the system is unplugged.
No of cycles allowed per second ----> speed , now one cycle per instructions

Register:
Memory location inside processor
These registers hold the inputs to the arithmetic/logic unit's circuitry and provide storage space for results produced by that unit. To perform an operation on data stored in main memory, the control unit transfers the data from memory into the general-purpose registers, informs the arithmetic/ logic unit which registers hold the data, activates the appropriate circuitry within the arithmetic/logic unit, and tells the arithmetic/logic unit which register should receive the result.
Registers are used to hold the data immediately applicable to the operation at hand; main memory is used to hold data that will be needed in the near future; and mass storage is used to hold data that will likely not be needed in the immediate future.
Processor is very fast --> memory to be made available for it
Content of register is fed to ALU, then result first posted to the register

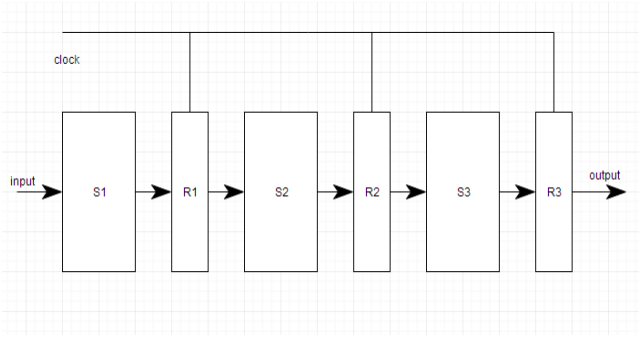
Cache:
Many machines are designed with an additional memory level, called cache memory. Cache memory is a portion (perhaps several hundred KB) of high-speed memory located within the CPU itself. In this special memory area, the machine attempts to keep a copy of that portion of main memory that is of current interest. In this setting, data transfers that normally would be made between registers and main memory are made between registers and cache memory. Any changes made to cache memory are then transferred collectively to main memory at a more opportune time. The result is a CPU that can execute its machine cycle more rapidly because it is not delayed by main memory communication.

ALU:
The arithmetic/logic group consists of the instructions that tell the control unit to request an activity within the arithmetic/logic unit. The arithmetic/logic unit is
capable of performing operations other than the basic arithmetic operations. Some of these additional operations are the Boolean operations AND, OR, and XOR,

Pipelining:
Increasing execution speed is not the only way to improve a computer's performance. The real goal is to improve the machine's throughput, which refers to the total amount of work the machine can accomplish in a given amount of time.
An example of how a computer's throughput can be increased without requiring an increase in execution speed involves pipelining, which is the technique of allowing the steps in the machine cycle to overlap, multiple instructions are overlapped during execution

Pipelining is the process of accumulating instruction from the processor through a pipeline. It allows storing and executing instructions in an orderly process. It is also known as **pipeline processing**. In particular, while one instruction is being executed, the next instruction can be fetched, which means that more than one instruction can be in "the pipe" at any one time, each at a different stage of being processed. Pipeline is divided into stages and these stages are connected with one another to form a pipe like structure. Instructions enter from one end and exit from another end.

Pipelining increases , the total throughput of the machine is increased even though the time required to fetch and execute each individual instruction remains the same.



Instruction Set Architecture

Instruction set architecture defines what the computer look likes to the software. What kind of instructions and registers there are, what are the rules of accessing memory etc. it also defines how these instructions are encoded

Micro Architecture

Micro architecture means how the processor of a computer is designed on a digital level for micro architecture there are only zeroes and ones but micro architecture defines precisely all the zeroes and the ones
Micro architecture always implements some Instruction set architecture.

System Architecture:

The name defines itself, the design will satisfy user requirements such as architecture module, interfaces and data for a system and it is connected to product development. It is the process of taking marketing information and creating product design to be manufacture. Modular systems are made by standardizing hardware and software.

Microarchitecture is known as computer organizations and it is the way when instruction set architecture is a built-in processor. Instruction set architecture is implemented with various microarchitecture and it varies because of changing technology.

Microarchitecture performs in a certain way. It reads the instruction and decodes it, will find parallel data to process the instruction and then will process the instruction and output will be generated.

It is used in microprocessors, microcontrollers. Some architectures overlap multiple instructions while executing but this does not happen in microarchitecture. Execution units like arithmetic logic units, floating-point units, load units, etc. are needed and it performs the operation of the processor. There are microarchitecture decisions within the system such as size, latency, and connectivity of the memories.

To make up the architecture, instruction set architecture is needed because it has a set of instructions that the processor understands. It has two instruction set one is RISC (reduced instruction set computer) and the second is CISC (complex instruction set computer).

Reduced instruction set computer architecture was realized in the 90's by IBM. Instruction has multiple address modes, but programs do not use all of them that is the reason multiple address modes were reduced. This helps the compiler to easily write the instructions, performed is increased.

Complex instruction set architecture is the root of compilers because earlier compilers were not there to write programs, to ease programming instructions are added. The best performance is obtained by using simple instruction from ISA.