

ESE Mains Achiever's Study Plan

Electronics & Communication Engineering

**Computer Organization and
Architecture**



1. Convert the following quantities to IEEE single-precision floating- point format

- A. 128
- B. -32.75
- C. 18.125
- D. 0.0625

Sol. (A) $128 = 2^7$, giving an exponent field of 134, a sign bit of 0, and a fraction field of 0 (because of the assumed 1). Therefore, $128 = 0b0100\ 00\ 11\ 0000\ 0000\ 0000\ 0000\ 0000$ in single-precision floating-point format.

(B) $-32.75 = -10000011_2$ or $-1.0000011_2 \times 2^5$. The single-precision floating- point representation of -32.75 is $0b1\ 10000100\ 000\ 0011\ 0000\ 0000\ 0000\ 0000$.

(c) $18.125 = 10010.001_2$ or $1.0010001_2 \times 2^4$, giving a single-precision floating-point representation of $0b0\ 10000\ 0011\ 001\ 0001\ 0000\ 0000\ 0000\ 0000$.

(d) $0.06525 = 0.001_2$ or 1×2^{-4} , giving a single precision floating-point representation of $0b0\ 0111\ 1011\ 000\ 0000\ 0000\ 0000\ 0000\ 0000$.

2. What values are represented by the following IEEE single-precision floating-point numbers?

- A. $0b1011\ 1101\ 0100\ 000\ 0000\ 0000\ 0000\ 0000$
- B. $0b0101\ 0101\ 0110\ 000\ 0000\ 0000\ 0000\ 0000$
- C. $0b1100\ 0001\ 1111\ 000\ 0000\ 0000\ 0000\ 0000$
- D. $0b0011\ 1010\ 1000\ 000\ 0000\ 0000\ 0000\ 0000$

Sol. (A) For this number, the sign bit = 1, exponent field = 122, exponent = -5 . Fraction field = $100\ 0000\ 0000\ 0000\ 0000\ 0000$, so this binary string represents $-1.1_2 \times 2^{-5} = -0.046875$

(B) Here, we have a sign bit of 0, exponent field of 170, so the exponent is 43, and a fraction field of $110\ 0000\ 0000\ 0000\ 0000\ 0000$, so the value of this number is

$$1.11_2 \times 2^{43} = 1.539 \times 10^{13} \text{ (to 4 significant digits)}$$

$$(C) -1.11_2 \times 2^4 = -30$$

$$(D) 1.0_2 \times 2^{-10} = 0.0009766 \text{ (to 4 significant digits)}$$

3. A hard disk with one platter rotates at 15,000 r/min and has 1024 tracks, each with 2048 sectors. Disk head starts at track 0. (Tracks are numbered from 0 to 1023). The disk then receives a request to access a random sector on a random track. If the seek time of the disk head is 1 ms for every 100 tracks it must cross:

- A. What is the average seek time?
- B. What is the average rotational latency?
- C. What is the transfer time for a sector?
- D. What is the total average time to resolve a request?

Sol. (A) Since the disk head starts at track 0, it will have to travel 0 tracks to handle a request to track 0, 1 track to handle a request to track 1, and so on, up to 1023 tracks to a request to track 1023. On an average, the head will have to travel half of the way to the outermost track, or 511.5 tracks. At 100 tracks/ms, this gives an average seek time of 5.115 ms.

(B) At 15,000 r/min each rotation takes 4 ms. The average rotational latency is half the rotation time, or 2 ms.

(C) Each rotation takes 4 ms. There are 2048 sector per track, so each sector takes $4\text{ms}/2048 = 1.95$ microseconds to pass under the read/write head. Therefore, the transfer time is 1.95 μs .

(D) The average access time is just the sum of the three components, or 7.117 ms (rounding to four significant digits). As stated in the text, this neglects the time to transmit the data to the processor.

4. Explain the different methods in DMA?

Sol. different methods in DMA are

- (i) Burst mode
- (ii) Cycle stealing mode
- (iii) Interleaved DMA

Burst Mode: In this method, until getting the target file transfer, CPU does not get the control on buses i.e. valuable CPU time is wasted and this method is specially used for transferring larger size block of data only

Cycle stealing method: In this method, CPU steals the buses (for one clock cycle time) when buses are not used by CPU.

- CPU time is not wasted but this method is used to transfer smallest size of data.

Interleaved DMA: In this method, both DMA controlled are CPU share the buses (control time)

- CPU uses the Buses on even clock cycles and DMA controller uses the Buses on odd clock cycles
- This method is used to transfer medium size of the Data.

5. Explain Different Interrupts?

Sol. Interrupt: It is a sudden brake in normal program execution; interrupt occurs in 2 ways

- (i) Hardware
- (ii) Software

Hardware interrupt: It occurs through the I/O device signal.

It is used to connect IO devices

Software interrupt: It occurs while executing a program.

It is used to change the program execution sequence.

Vector interrupt: Device supplies its vector Address, after accepting vector interrupt, program counter is loaded with the address which is supplied by the device

Non-vectorized interrupt: In this, no address is reserved for the interrupting source.

Maskable interrupt: This type of interrupt request can be permitted or Rejected

Non Maskable interrupt: This type of interrupt request is compulsory permitted immediately

- All software interrupts are maskable and vectored interrupts
- Non Maskable Hardware interrupt is used to know the power failure status
- Non- vectored Hardware Interrupt is used for connecting more number of I/O devices because one vector interrupt permits only one IO device

6. What are different types of languages used in computer?

Sol. Types of languages are:

Machine language

Computer can understand only instructions, which are implemented in hardware circuitry level. For example, simple addition, subtraction, division, comparison, moving data, control jumping etc. All these instructions are encoded in binary (1 & 0), and they can be executed by the computer directly.

This machine language consider as first generation language, those days it is used for testing the machine. Actually, it is difficult to write programs, understand, and find errors in machine code of 1's & 0's; it leads to a lot of typing mistakes, because this language contain only 1 and 0 symbols. This language is also called binary or low level, language.

Assembly language

This is somewhat better than binary language. Here every machine language instruction is given a symbolic name, so that programs can be written easily using these names instead of binary instruction in ones and zeros. Assembly languages is also called low-level language, since every instructions supported by hardware.

However, main drawback of assembly is, it is quite cumbersome to write programs in a low-level language, since a large number of instructions must be written even for a simple task such as computing and printing the roots of a quadratic equation. Moreover, machine language differ computer to computer.

High-level language

This is English like language, with support of vast number of mathematical operators. In addition, it can adjustable to different kinds of machines, so programmer can easily code the programs without bothering about hardware. C is a high level language with support of assembly language so it is often called middle level language.

7. What is function? How many types of functions we have? What is importance of main () function?

Sol. Function is a sub-program, which contains one or more instructions designed to perform specific task. Function can be classified into library and user-defined functions. The library functions are pre-redefined (prewritten) functions, which are designed and written by the C manufacturers to provide solutions for routine tasks in the programming.

For example: `printf ()`, `scanf ()` etc are input/output functions to interact with monitor and keyboard; Mathematical functions like `pow ()`, `sqrt ()` etc are useful in calculating the power & square root values etc. The vendors of C supply these predefined functions in compiled form (i.e., in Object code format) with C software. The collection of such compiled functions is called a Function Library.

We can write our own functions called user-defined function as per our requirements just like the library functions `pow ()`, `sqrt ()` etc.

Let us see the following instructions, which are nothing but calling a function

```
a = pow (3, 5); /* calling a power function to calculate 3 power 5 */
```

```
b = sqrt (28); /* calling a square root function to calculate root of 28*/
```

```
clrscr (); /* clear the next mode screen, just like "cls" command is DOS*/
```

When the instruction `a = pow (3, 5)` gets executed, the control jumps into power function (sub-program), after calculating the power value. The control comes back with result and assigns to the 'a'. Later executes further instructions.

8. What is the purpose of library functions?

Sol. The library functions are stored in a separate file (s) (*.lib) in compiled format and they are automatically linked to the program at the time of compilation and these are hidden to the programmer.

File inclusion statements are used to include such library function, which are declared in that files. These files are called header files, for example, `maths .h` file contains declarations of mathematical functions such as `pow ()`, `sqrt ()` etc. (But actual code of functions are stored in a separate files, they are *.lip) Whenever a particular function is used in the program, we should include associated library header file as `#include < mat/h>`.

9. Program to accept an integer number (+ve or -ve) from keyboard and display it in negative. If user entered positive, then it is converted into negative by multiplying with 1. If user entered negative, then print without converting it.

Sol. #include < stdio.h >

```
Void main ()
{
int n;
printf ("Enter Number with any sign:");
scanf ("%d", &n);
if (n > 0)/* if 'n' is +ve then change it into negative */
n = n* -1;/* observe that NO braces {}' are used*/
printf ("\n Result = % d", n);
}
```

In the above program the instruction $n = n * -1$ is to be executed only when the given number is +ve. If the given number is already -ve then we are not changing its value. i.e. $n = n * -1$ will not be executed.

Input: - 2

Output: -2

Input: 4

Output: 4

Note: Here 'if' body contains only one instruction, so, braces {} are optional to the body

10. What are different scheduling algorithms used in CPU?

Sol. A process Scheduler schedules different processes to be assigned to the CPU based on particular scheduling algorithms. There are six popular process scheduling algorithms.

- First-Come, First-Served (FCFS) Scheduling
- Shortest-Job-Next (SJN) Scheduling
- Priority Scheduling
- Shortest Remaining Time
- Round Robin (RR) Scheduling
- Multiple-Level Queues Scheduling

First Come, First Served (FCFS)

- Jobs are executed on first come, first served basis.
- It is a non-pre-emptive scheduling algorithm.
- Easy to understand and implement.
- Its implementation is based on FIFO queue.
- Poor in performance, as average wait time is high.

Shortest Job Next (SJN)

- This is also known as shortest job first, or SJF.
- This is a non-pre-emptive scheduling algorithm.
- Best approach to minimize waiting time.
- Easy to implement in Batch systems where required CPU time is known in advance.

Priority Based Scheduling

- Priority scheduling is a non-pre-emptive algorithm and one of the most common scheduling algorithms in batch systems.
- Each process is assigned a priority. Process with highest priority is to be executed first and so on.
- Processes with same priority are executed on first come first served basis.
- Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Shortest Remaining Time

- Shortest remaining time (SRT) is the primitive version of the SJN algorithm.
- The processor is allocated to the job closest to completion, but it can be pre-empted by a newer ready job with shorter time to completion.
- Impossible to implement in interactive systems where required CPU time is not known.
- It is often used in batch environments where short jobs need to be given preference.

Round Robin Scheduling

- Round Robin is a pre-emptive process scheduling algorithm.
- Each process is provided a fix time to execute; it is called a quantum.
- Once a process is executed for a given time period, it is pre-empted and other process executes for a given time period.
- Context switching is used to save states of prompted processes.

Multiple-Level Queues Scheduling

Multiple-level queues are not independent scheduling algorithm. They make use of other existing algorithms to group and schedule jobs with common characteristics.

- Multiple queues are maintained for processes with common characteristics.
- Each queue can have its own scheduling algorithms.
- Priorities are assigned to each queue.

For example, CPU-bound jobs can be scheduled in one queue and all I/O-bound jobs in another queue. The Process Scheduler then alternately selects jobs from each queue and assigns them to the CPU based on the algorithm assigned to the queue.

11. Explain ACID property of transactions.

Sol. ACID properties:

Consistency:

Execution of a transaction in isolation (that is, with no other transaction executing concurrently) preserves the consistency of the database. This is typically the responsibility of the application programmer who codes the transactions.

Atomicity:

Either all operations of the transaction are reflected properly in the database, or none. Clearly lack of atomicity will lead to inconsistency in the database.

Isolation:

When multiple transactions execute concurrently, it should be the case that, for every pair of transactions T_i and T_j , it appears to T_i that either T_j finished execution before T_i started, or T_j started execution after T_i finished. Thus each transaction is unaware of other transactions executing concurrently with it.

Durability:

After a transaction completes successfully, the changes it has made to the database persist, even if there system failures.



OUR TOP GRADIANS IN GATE 2020



Ghanendra Singh

AIR-6

ECE



Himanshu Kumar

AIR-9

EE



Nikhil Kumar

AIR-9

EE



Raja Majhi

AIR-30

ECE



**Samanwaya Deep
Chakraborty**

AIR-41

ECE



Abhishek

AIR-45

ECE

Classroom

Vision 2021-Course for ESE & GATE (Batch-3)

Electronics & Communication Engineering



800+hrs
Live Class



7000+
Practise
Question



Test Series

Vision 2021

A Course for **ESE & GATE** Electronics Aspirants
Batch-3

Why take this course?

- › **650+ Hours** of Live Classes for ESE & GATE Technical Syllabus
- › **150+ Hours** of Live Classes for ESE Prelims Paper 1 Syllabus
- › **750+ Quizzes** & Conventional Assignments for Practice
- › Subject & Full-Length **Mock Tests** for GATE & ESE



MN Ramesh | Rakesh Talreja | Chandan Jha | Vijay Bansal