Q.1(a) Explain Von Neumann model of architecture in details. **05**

Ans.: Von Neumann Architecture :

Programming computers with huge numbers of switches and cables was slow, tedious, and inflexible. Von Neumann came to realize that the program could be represented in digital form in the computer's memory, along with the data. Von Neumann also saw that the clumsy serial decimal arithmetic used by the ENIAC, with each digit represented by 10 vacuum tubes (1 on and 9 off) could be replaced by using parallel binary arithmetic.

The basic design, which he first described, is now known as a **Von Neumann machine**. It was used in the EDSAC, the first stored program computer, and is still the basis for nearly all digital computers.

A sketch of the architecture is given in figure.

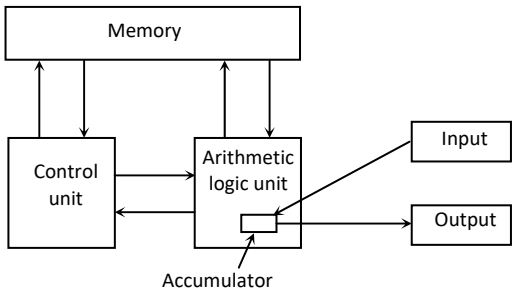


Fig.: The original von Neumann machine

The von Neumann machine had five basic parts : the memory, the arithmetic logic unit, the control unit, and the input and output equipment. The memory consisted of 4096 words, a word holding 40 bits, each a 0 or a 1. Each word held either two 20-bit instructions or a 40-bit signed integer. The instructions had 8 bits devoted to telling the instruction type, and 12 bits for specifying one of the 4096 memory words.

Q.1(b) Explain differences between RISC and CISC. **05**

Ans.: RISC versus CISC characteristics :

Architectural characteristics	CISC	RISC
Instruction set size and instruction formats	Large set of instructions with variable formats (16-64 bits per instruction)	Small set of instructions with fixed (32 bit) format and most register based instructions.
Addressing modes	12 – 24	Limited to 3 − 5
GPRs and cache design	8−24 GPRs, mostly with a unified cache for instructions and data, recent designs also use split caches	Large numbers (32−192) of GPRs with mostly split data cache and instruction cache.

Clock rate and CPI	33−50MHz in 1992 with a CPI between 2 and 15	50 − 150MHz in 1993 with one cycle for almost all instructions and on average CPI < 1.5
CPU control	Most microcoded using control memory (ROM), but modern CISC also uses hardwired control.	Most hardwired without control memory.

Q.1(c) Explain nano programming. 05

Ans.: Computer programming in Nano is one of the newest developments. It was believed that a Nano mechanical computer could run a million times faster than a microprocessor based computer. This is because that one out of the million components of a computer is made of mechanical space.

The Nano computer language is believed to work well with the present day computers systems. The primary use of this programming language is on graphics. With the Nano-X graphics system you could create much fancier graphical programs. To make it work, you have to specifically create the program with the Window, Unix or Macintosh interface in mind.

The Nano computer language primarily came from the nano-technology. Nano technology refers from the fields of applied science that control matter on its molecular and atomic scale.

This program is easy to learn and apply. Texts can be typed immediately into the interface. It is also simple to insert text into the program with the use of some editing configuration. There is also nano editor software that you can use with the main program base so that saving, cutting, pasting and searching becomes fairly straight forward.

Q.1(d) Explain difference between hardwired and softwired. 05

Ans.:

Hardwired control	softwired/Micro-programmed control
(1) Speed	
comparatively fast	comparatively slow
(2) Control functions	
Implemented in hardware	Implemented in software
(3) Flexibility	
more flexible, to accommodate new system specifications or new instructions	Not flexible to accommodate new system specification or new instructions for that redesign is required.
(4) Ability to handle large/complex instruction sets somewhat difficult.	Ability to handle large complex instruction sets is easier.
(5) Ability to support operating systems and diagnostic features are very difficult	Ability to support operating system and diagnostic features are easier.

(6) Design process is somewhat complicated	Design process is orderly and systematic.
(7) Applications	
Used in RISC microprocessor	Used in mainframes and some microprocessor
(8) Instruction size usually under 100 instructions	instruction size usually over 100 instruction.
(9) Chip area efficiency	
uses less area	uses more area

Q.2(a) With a neat flowchart, explain the procedure for division of floating-point numbers carried out in the computer. **10**

Ans.: Floating-point multiplication and division are much simpler processes than addition and subtraction, as the following discussion indicates.

We first consider multiplication. If either operand is 0, 0 is reported as the result. The next step is to add the exponents. If the exponents are stored in biased form, the exponent sum would have doubled the bias. Thus, the bias value must be subtracted from the sum. The result could be either an exponent overflow or underflow, which would be xd reported, ending the algorithm.

Finally, let us consider the flowchart for division depicted in figure. Again, the first step is testing for 0. If the divisor is 0, an error report is issued, of the result is set to infinity, depending on the implementation. A dividend of 0 results in 0. Next, the divisor exponent is subtracted from the dividend exponent. This removes the bias, which must be added back in. Tests are then made for exponent underflow or overflow.

The next step is to divide the significands. This is followed with the usual normalization and rounding.

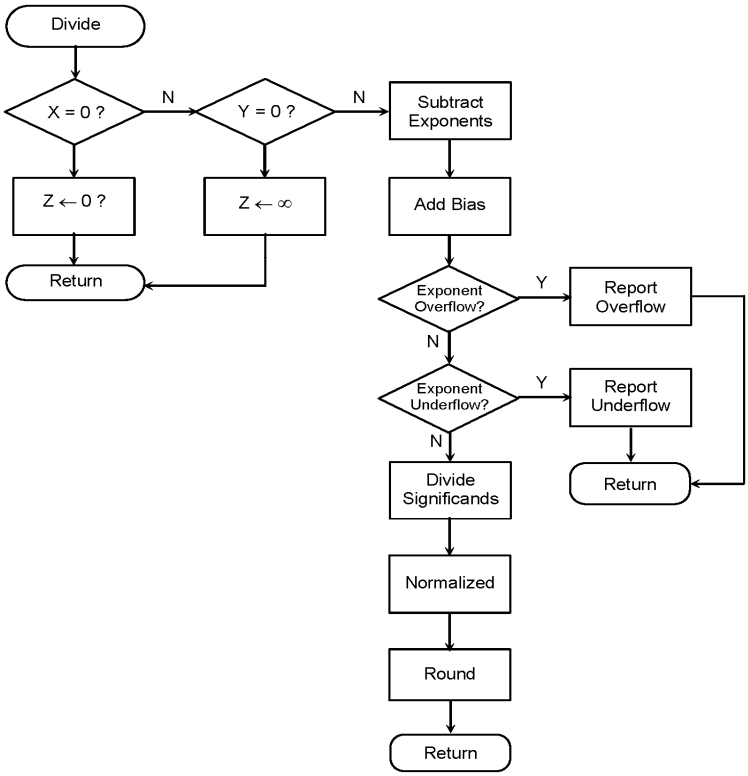


Fig. : Floating point division (z ← x/y).

Division :

Division is somewhat more complex than multiplication but is based on the same general principle. As before, the basis for the algorithm is the paper–and–pencil approach and the operation involves repetitive shifting and addition or subtraction. Figure (a) shows an example of the long division of unsigned binary integers. It is instructive to describe the process in detail. First, the bits of the dividend are examined from left to right, until the set of bits examined represents a number greater than or equal to the divisor; this is referred to as the divisor being able to divide the number. Until this event occurs, 0s are placed in the quotient from left to right. When the event occurs, a 1 is placed in the quotient and the divisor is subtracted from the partial dividend. The result is referred to as a partial remainder. From this point on, the division follows a cyclic pattern. At each cycle, additional bits from the dividend are appended to the partial remainder until the result is greater than or equal to the divisor. As before, the divisor is subtracted from this number to produce a new partial remainder. The process continues until all the bits of the dividend are exhausted.

The algorithm can be summarized as follows :

1. Load the divisor into the M register and the dividend into the A, Q registers. The dividend must be expressed as a 2n–bit two's complement number. Thus, for example, the 4–bit 0111 becomes 00000111, and 1001 becomes 11111001.

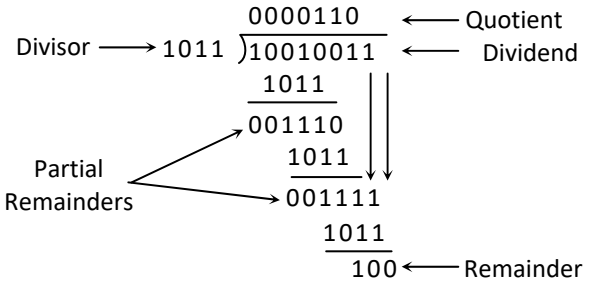


Fig. (a) : Division of unsigned binary integers

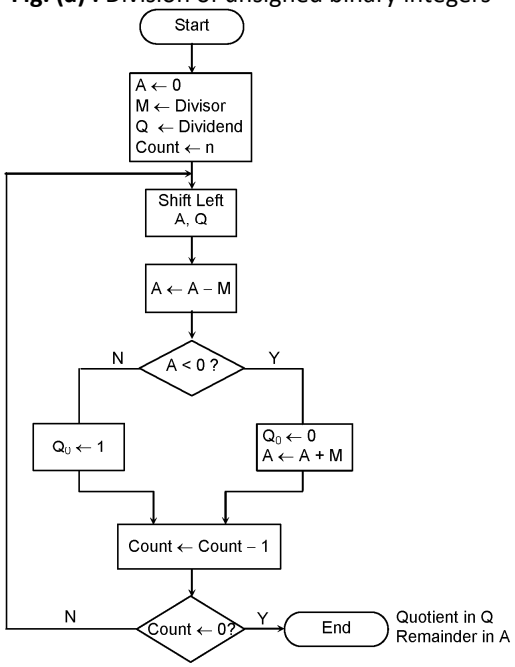


Fig. (b) : Flowchart for unsigned binary division

A	**Q**	**M = 0011**
0000	0111	Initial Value
0000	1110	Shift
1101		Subtract
0000	1110	Restore
0001	1100	Shift
1110		Subtract
0001	1100	Restore
0011	1000	Shift
0000		Subtract
0000	1001	Set $Q_0 = 1$
0001	0010	Shift
1110		Subtract
0001	0010	Restore

(a) (7) ÷ (3)

A	**Q**	**M = 1101**
0000	0111	Initial Value
0000	1110	Shift
1101		Add
0000	1110	Restore
0001	1100	Shift
1110		Add
0001	1100	Restore
0011	1000	Shift
0000		Add
0000	1001	Set $Q_0 = 1$
0001	0010	Shift
1110		Add
0001	0010	Restore

(b) (–7) ÷ (–3)

A	Q	M = 0011
1111	1001	Initial Value
1111	0010	Shift
0010		Add
1111	0010	Restore
1110	0100	Shift
0001		Add
1110	0100	Restore
1100	1000	Shift
1111		Add
1111	1001	Set $Q_0 = 1$
1111	0010	Shift
0010		Add
1111	0010	Restore

(c) (–7) ÷ (3)

A	Q	M = 1101
1111	1001	Initial Value
1111	0010	Shift
0010		Subtract
1111	0010	Restore
1110	0100	Shift
0001		Subtract
1110	0100	Restore
1100	1000	Shift
1111		Subtract
1111	1001	Set $Q_0 = 1$
1111	0010	Shift
0010		Subtract
1111	0010	Restore

(d) (–7) ÷ (–3)

Fig. (c) : Examples of two's complement division.

2. Shift A, Q left 1 bit position.
3. If M and A have the same signs, perform A ← A – M; otherwise, A ← A + M.
4. The above operation is successful if the sign of A is the same before and after the operation.
   (a) If the operation is successful or (A = 0 and Q = 0), then, set $Q_0 \leftarrow 1$.
   (b) If the operation is unsuccessful and ($A \neq 0$ or $Q \neq 0$), then set $Q_0 \leftarrow 0$ and restore the previous value of A.
5. Repeat steps 2 through 4 as many times as there are bit positions in Q.
6. The remainder is in A. If the signs of the divisor and dividend were the same, then the quotient is in Q; otherwise, the correct quotient is the two's complement of Q.

The reader will note from Fig. that, (7) ÷ (3) and (–7) ÷ (–3) produce different remainders. This is because the remainder is defined as,

D = Q * V + R.

where, D = Dividend, Q = Quotient, V = Divisor, R = Remainder.
The results of Fig. are consists with this formula.

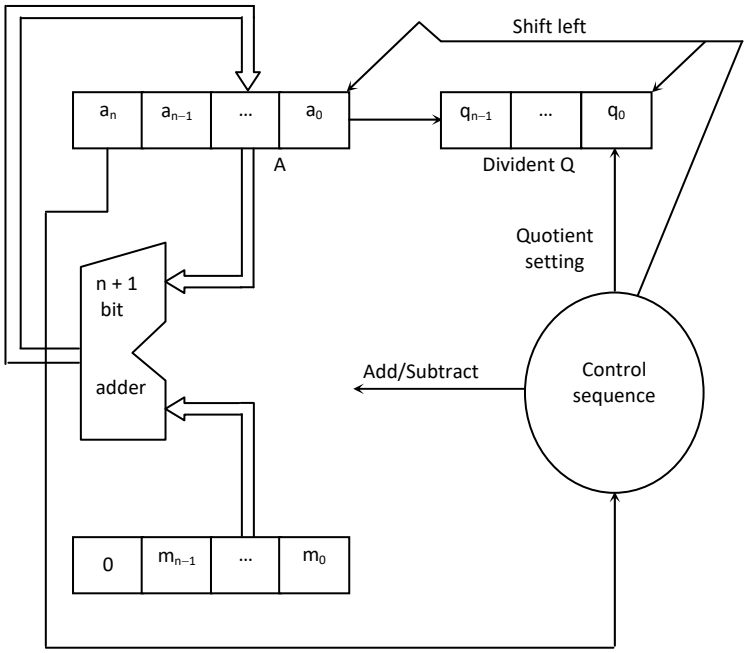
Q.2(b) Give algorithm for restoring division to handle both positive and negative integer in two complement code. **10**

Ans.: Figure below shows a logic circuit arrangement that implements this restoring–division technique.

Algorithm :

1) An n–bit positive divisor is loaded into register m and n–bit positive divident is loaded into register Q as the start of the operation.
2) Register A is set to 0.
3) After the division is complete, the n–bit quotient is in register Q and the remainder is in register A.
4) The required subtractions are facilitated by using 2's complement arithmetic.

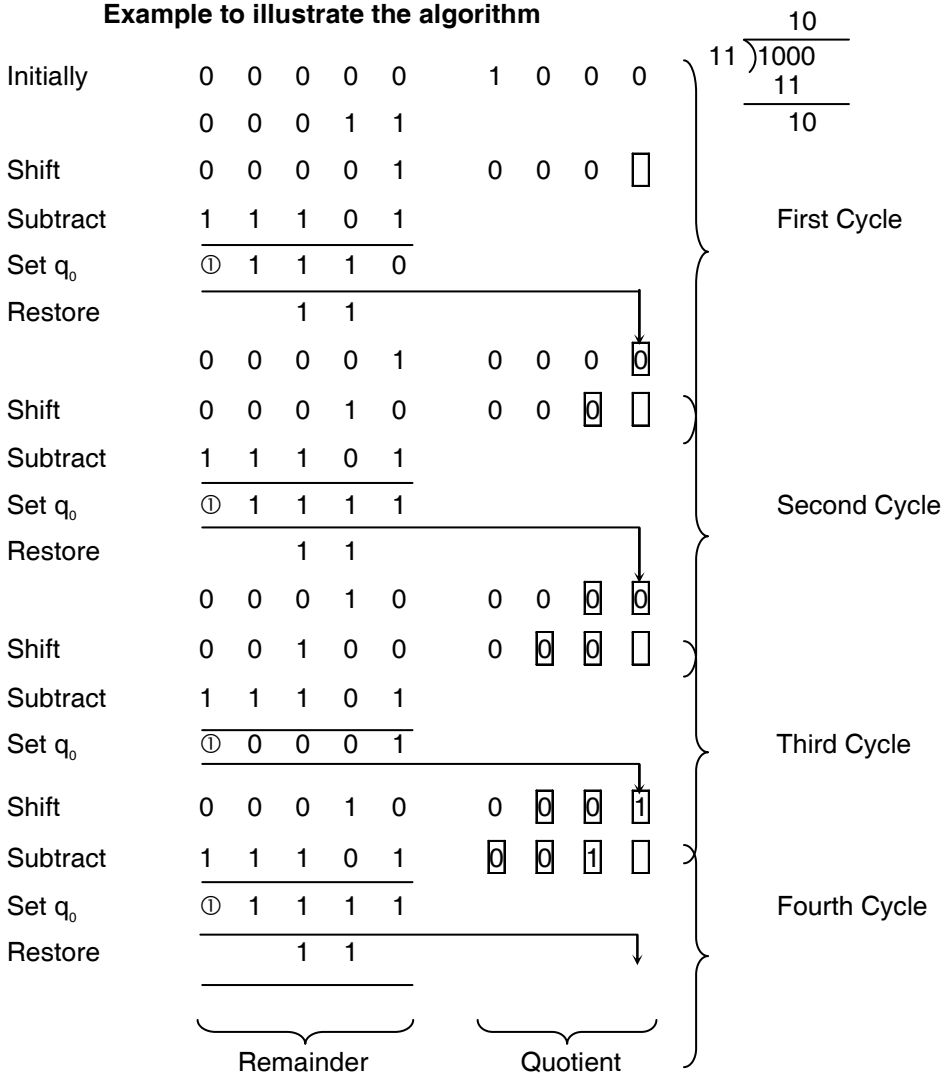
5) The extra bit position at the left end of both A and M accommodates the sign bit during subtraction.



The algorithm that performs the division. Do the following n times.

- Shift A and Q left one binary position.
- Subtract M form A and place the answer back in A.
- If the sign of A is 1, seg q_0 to 0 and odd 4 back to A (that is restore A)
- Otherwise, set q_0 to 1.

Example to illustrate the algorithm



Q.3(a) Write a short note on Memory characteristics. **10**

Ans.: **Memory Characteristics :** The properties to be considered when evaluating any memory technology are :

i) Cost : The price should include the cost of information storage cells as well as the cost of the peripheral equipment or access circuitry essential to the operation of memory.

cost = price of complete memory system / total bits of storage capacity.

ii) Access time : It is the time required to read or write a fixed amount of information. e.g. one word from the memory. Access time depends upon the physical characteristics of the storage medium and also on the types of access mechanism used. It is usually calculated from the time a read request is received by the

memory unit to the time a read request is made available to the memory output terminals. The access time measured in words per second is another widely used performance measure for storage devices. Thus, low cost and high access rates are desirable memory characteristics.

iii) Access modes : It is the order or sequence in which information can be accessed. Memory can be accessed randomly or sequentially. In random access memories each storage location can be accessed independently of the other locations whereas in serial access memory storage locations can be accessed only in a certain predetermined sequence.

iv) Alterability : Memories whose contents cannot be altered on line are called Read Only Memories (ROMs) Memories in which reading or writing can be done online are called read write memories. All memories used for temporary purpose are read write memories.

v) Permanent of storage : The physical processes involved in storage are sometimes inherently unstable, so that stored information may be lost over a period of time unless appropriate action is taken. There are three important memory characteristics that can destroy information : destructive readout, dynamic storage and volatility. In destructive readout, the memory contents are called as the memory is read.

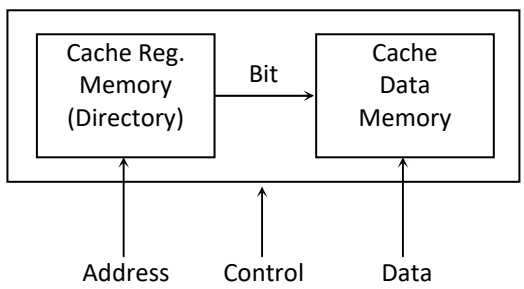
vi) Cycle time and data transfer rate : The minimum time that must elapse between the initiation of two different access by memory can be greater than access time, this loosely defined term is called cycle time of the memory. It is generally convenient to assume that cycle time is the time needed to complete any read or write operation in memory.

vii) Physical characteristics : Many different physical properties of matter are used for information storage. The most important properties used for this purpose, all classified as electronic, magnetic, mechanical and optical. A factor determining the physical size of a memory unit is the storage density measured in bits per unit area. In general, memories with no moving parts have much higher reliability than memories such as magnetic disks which involves considerable mechanical motion.

Q.3(b) Explain the elements of cache design. **10**

Ans.: Cache Organisation :

- Memory words are stored in the "cache data memory" and are grouped into small pages. Thus, contents of cache's data memory are copy of a set of main memory work. Each cache black is marked with its block address, referred to as a 'tag', so cache knows to what pair of memory space the block belongs.



- The tag addresses contain, that are currently assigned to cache which can be non-continuous is stored in a special memory the 'Cache tag memory' or directory.
  Example : If B_j is block containing D_j data in M_1. Then, B_j is in cache tag memory and D_j is in cache data memory.
- To improve the performance of the computer, the cache memory is used. Hence, the access time of cache should greater than main memory. Therefore, of main memory is implemented with DRAM technology having on access time t_{A1} = 50 ns, then cache might be implemented with an SRAM technology having an access time t_{A2} = 10 ns.

Q.4(a) A virtual memory system has a 16 K word logical address space, 8K word physical address space with page size of 2K word. The page address trace of a program has been found to be **10**

7 5 3 2 1 0 4 1 6 7 4 2 0 1 3 5.

List the four pages resident in the memory after each page reference for the following replacement policies :(a) FIFO (b) LRU

Ans.:

7	**5**	**3**	**2**	**1**	**0**	**4**	**1**	**6**	**7**	**4**	**2**	**0**	**1**	**3**	**5**

FIFO

7	7	7	7	1	1	1	1*	1	1	1	2	2	2	2	5
	5	5	5	5	0	0	0	0	7	7	7	7	1	1	1
		3	3	3	3	4	4	4	4	4	4*	4	4	3	3
			2	2	2	2	2	6	6	6	6	0	0	0	0

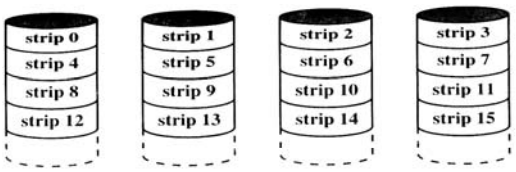
LRU

7	7	7	7	1	1	1	1*	1	1	1	2	2	2	5
	5	5	5	5	0	0	0	0	7	7	7	1	1	1
		3	3	3	3	4	4	4	4	4*	4	4	3	3
			2	2	2	2	2	6	6	6	0	0	0	0

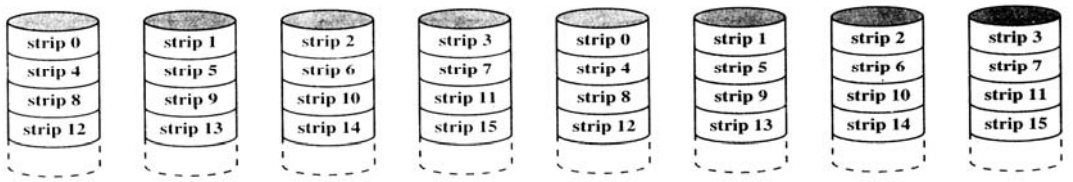
Q.4(b) Explain RAID and its level. **10**

Ans.: Raid :

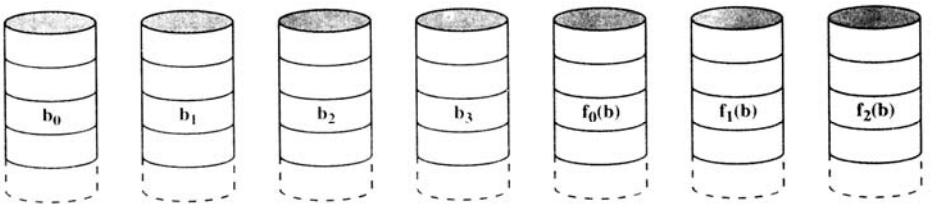
RAID levels



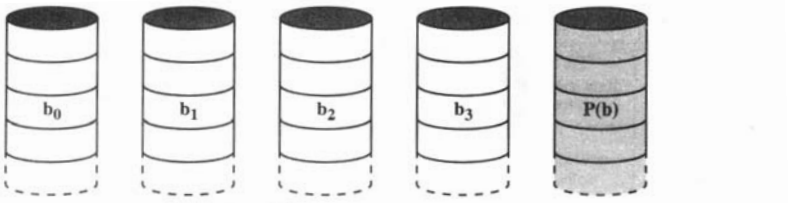
(a) RAID 0 (Nonredundant)



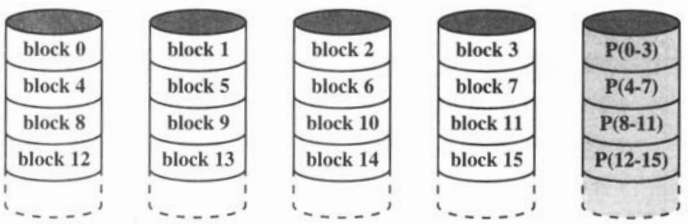
(b) RAID 1 (Mirrored)



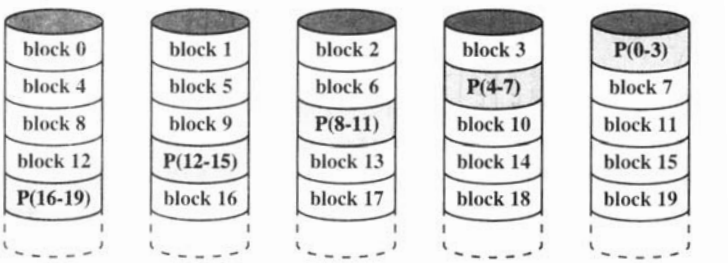
(c) RAID 2 (Redundancy through Hamming code)



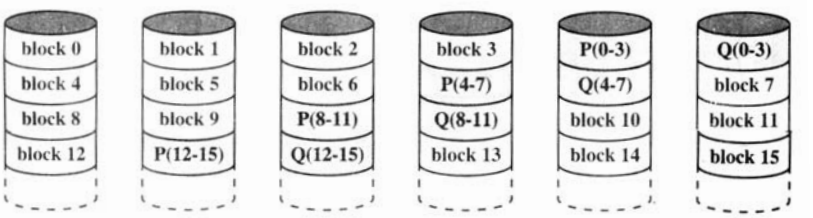
(d) RAID 3 (Bit-interleaved parity)



(e) RAID 4 (Block-level parity)



(f) RAID 5 (Block-level distributed parity)



(g) RAID 6 (Dual redundancy)

These are schemes to provide redundancy at lower cost by using disk stripping combined with parity bits. Different RAID organizations, or RAID levels, have differing cost, performance and reliability characteristics.

RAID level 0 :

- Stripping at the level of blocks ; non − redundant.
- Used in high − performance applications where data loss is not critical.

RAID level 1 :

- Mirrored disks, offers best write performance
- Popular for applications such as storing log files in a database system.

RAID level 2 :

- Memory style Error correcting codes (ECC) with bit stripping.

RAID level 3 :

- Bit interleaved parity; a single parity bit can be used for error corrections not just detection.
- When writing data, parity bit must also be computed and written.
- Faster data transfer than with a single disk, but fewer I/Os per second since every disk has to participate in every I/O.
- Subsumes level 2 (provides all its benefits, at lower cost).

RAID level 4 :

- Block interleaved parity; uses block level striping, and keeps a parity block on a separate disk for corresponding blocks from N other disks.
- Provides higher I/O rates for independent block read than level 3 (block read goes to a single disk, so blocks stored on different disks can be read in parallel.
- Provides high transfer rates for reads of multiple blocks.
- However, parity block becomes a bottleneck for independent block writes since every block write also writes to parity disk.

RAID level 5 :

- Block interleaved distributed parity; partitions data and parity among all N+1 disks, rather than storing data in N disks and parity in 1 disk.
- For example, with 5 disks, parity block for nth set of blocks is stored on disk (n mod 5) + 1, with data blocks stored on the other 4 disks.
- Higher I/O rates than level 4 (Block writes occur in parallel if the blocks and their parity blocks are on different disks.)
- Subsumes level 4

RAID level 6 :

- P + Q redundancy scheme; similar to level 5, but stores extra redundant information to guard against multiple disk failures.
- Better reliability than level 5 at a higher cost; not used widely.

Table : RAID Levels

Category	Level	Description	Disks required	Data availability	Large I/O data transfer capacity	Small I/O request rate
Striping	0	Nonredundant	N	Lower than single disk.	Very high.	Very high for both read and write.

Mirroring	1	Mirrored	2N, 3N, etc.	Higher than RAID 2, 3, 4 or 5; lower than RAID 6.	Higher than single disk for read similar to single disk for write.	Upto twice that of a single disk for read similar to single disk for write.
Parallel access	2	Redundant via Hamming code	N + m	Much higher than single disk; higher than RAID 3, 4 or 5.	Higher of all listed alternatives.	Approximately twice that of a single disk.
	3	Bit-interleaved parity	N + 1	Much higher than single disk; comparable to RAID 2, 4 or 5.	Highest of all listed alternatives.	Approximately twice that of a single disk.
Independent access	4	Block-interleaved distributed parity	N + 3	Much higher than single disk; comparable to RAID 2, 3 or 5.	Similar to RAID 0 for read; significantly lower than single disk for write.	Similar to RAID 0 for read; generally lower than single disk for write.
	5	Block-interleaved distributed parity	N + 1	Much higher than single disk; comparable to RADID 2, 3 or 4.	Similar to RAID 0 for read; lower than single disk for write.	Similar to RAID 0 for read; generally lower than single disk for write.
	6	Block-interleaved dual distributed parity	N + 2	Highest of all listed alternatives.	Similar to RAID 0 for read; lower than RAID 5 for write.	Similar to RAID 0 for read; significantly lower than RAID 5 for write.

Q.5(a) Explain IEEE floating point represent? 10

Ans.:

- Floating point representation is redundant in the sense that the same number can be represented in more than one way.
  For example : 1.0×10^{18}, 0.1×10^{19}, 1000000×10^{12} and 0.000001×10^{24} are possible representation of a quintillion.
- It is generally desirable to have a unique or normal form for each representable number in a floating–point system. Consider the common case where the mantissa is a sign–magnitude fraction and a base of 'r' is used.
- The mantissa is said to be normalized if the digit to the right of the radix point is not zero, that is no leading zeroes appears in the magnitude part of the number. For example, 0.1×10^{19} is the unique normal form of a quintillion using base 10, a decimal mantissa and a decimal exponential binary fraction in 2's complement code is normalized when the sign bit differs from one bit to its right. Normalization restricts the magnitude |M| of a fractional binary mantissa to the range $1/2 \leq |M| < 1$.
- Normal forms can be defined similarly for other floating–point codes. An un-normalized number is normalized by shifting the mantissa to the right or left and appropriately incrementing or decrementing the exponent to compensate for the mantissa shift.
- The representation of zero posses some special problem. The mantissa must of course, be zero but the exponent can have any value, since $0 \times B^E = 0$ for all the values of E. Round off error result in a mantissa that is nearly, but not exactly zero. For the entire floating point number to be close to zero, its exponent must be a very large negative number – K. This requirement suggests that the exponent used for representing zero should be the negative number with the largest magnitude that can be contained in the exponent field of the number format. If K–bits are allowed for

the exponent including its sign, then 2^K exponent bit pattern are available to represent signed integers which can range either from -2^{K-1} to $2^{K-1} - 1$ or from $-2^{K-1} + 1$ to 2^{K-1}, so that, K is 2^{K-1} or $2^{K-1} - 1$.

- The floating point exponent field E contains an integers that is the desired exponent value plus K.
- The quantity K is called the bias and an exponent encoded in this way, is called a biased exponent.

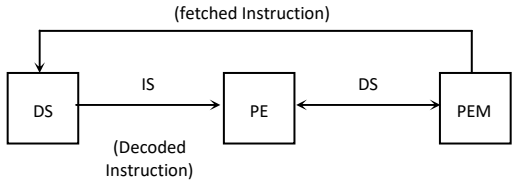
Q.5(b) Explain Flynn classification in detail. **10**

Ans.: Flynn's classification of Computer Organisation :

The classification suggested by Flynn, is based on multiplicity of instruction stream (IS) and data stream (DS).

1. Single Instruction Single Data (SISD)

This is normal single processor system (UNI processor) which contains single decoded instruction stream (IS) which operates on a single data stream (DS).

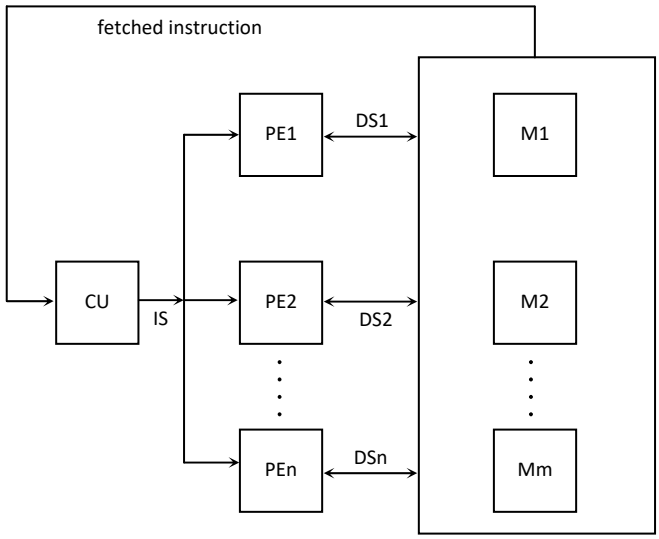


- Here control unit (CU) fetches instruction stream from processing element memory (PEM), which is decoded to generate single decoded instruction stream (IS). This is executed by processing element (PE) using single data stream (DS).
- Once the result data is generated by PE, it is stored back to the PEM.

Example : of SISD architecture are INTEL, 8085, 8086, 80386.

2. Single Instruction Multiple Data (SIMD)

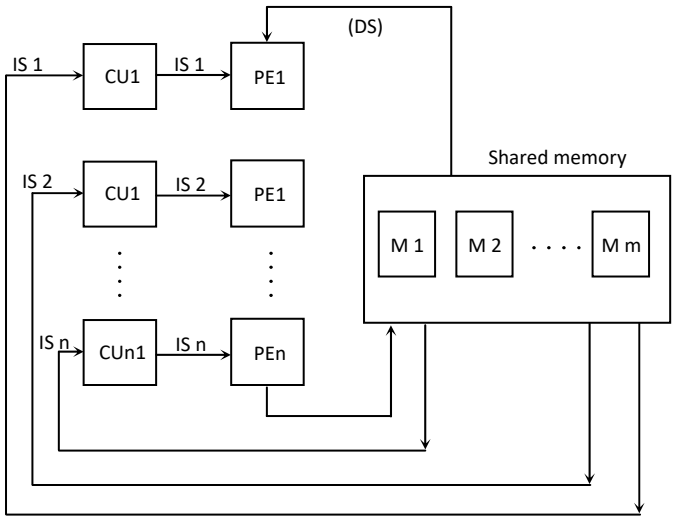
- This is also referred as Array Processor.



Here the single instruction stream is fetched from shared memory modules or taken from external front end system. It is decoded by a single control unit (CU), to generate decoded instruction steak (IS). This instruction is broadcasted to multiple processing elements (PES), which will operate on different data sets. Hence the name given (SIMD). The result generated by processing elements (PEs) are stored back into memory modules.

3. Multiple Instruction Single Data (MISD)

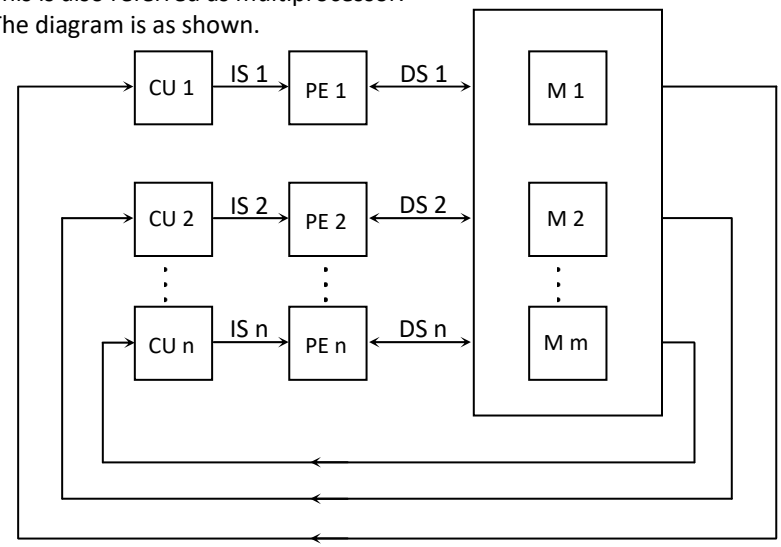
Here multiple instruction streams are fetched form shared memory modules by multiple control units which in turn generates multiple decoded instructions stream (IS). These are operated on single data streak (DS) taken from shared memory modules. MISD architecture is not realized but just a prototype model is designed.



4. Multiple Instruction Multiple Data (MIMD)

This is also referred as multiprocessor.

The diagram is as shown.



Here, multiple instruction streams are fetch by control units. These instruction stream are decoded to get multiple decoded instruction stream (IS), which operates on multiple data stream (DS) taken from shared memory modules. Here, each processing element executes one instruction stream and operate exactly on one data stream.

Q.6(a) Write short note on DMA. **10**

Ans.: Direct Memory Access

The hardware required to design Direct Memory Access is as shown below :

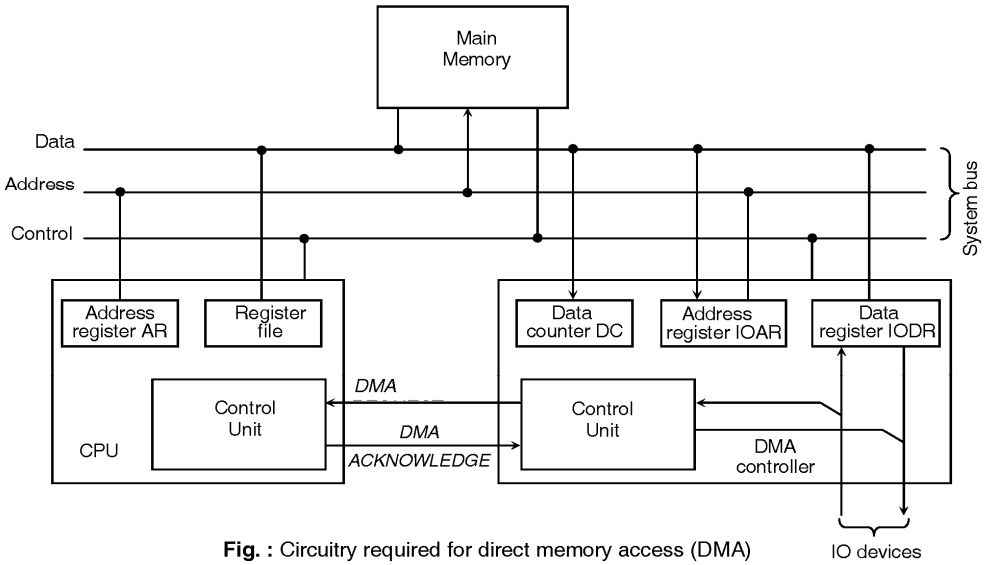


Fig. : Circuitry required for direct memory access (DMA)

- In this structure of DMA, the both DPU & DMA controller have access to main memory via a shared system bus having data, address and control lines.
- The I/O devices are connected to system bus for transferring of data via a special interface circuit shown as **"DMA controller"**.
- It contains the data register (IODR) then address register (IOAR) and data counter register (DC) which enables the DMA controller to transfer data to or form the different regions of main memory. The IOAR register of DMA contains the base address of the memory region where transfer operation is to be done. This register is automatically decremented or incremented after each word is transferred.
- The data counters (DC) contain the number of words remain to be transferred. This counter value is automatically decremented after each transfer of word and tested for zero. When his DC reaches to zero, the DMA controller stop the DMA transfer.
- It normally provide with the interrupt capability. It can send interrupt to CPU to indicate end of data transfer.
- DMA transfer can be done by two ways :

Steps involved in DMA systems are :

1. The CPU executes two I/O instructions, which load the DMA register IOAR & DR with their initial values. The IOAR is loaded with base address of the region of memory used for transfer. The DC will contain number of words to be transferred to or from memory.

2. When a DMA controller is ready to transmit or revive the data, it will activate the DMA request interrupt signal to CPU. The CPU will wait for the next DMA breakpoint, then it will activate the DMA ACKNOWLEDGEMENT signal.
3. The DMA controller now transfer data, to or from the main memory. After a word is transferred it update DC & IOAR registers.
4. If DC is not yet zero, & I/O device not ready to send or receive the data, then DMA controller release the system bus to CPU by deactivating the REQUEST line CPU respond to DMA controller by deactivating the DMA acknowledgement line.
5. If DC reaches to zero, then DMA controller should stop the transfer and send interrupt request signal to CPU; CPU responds by halting the I/O device or by initiating a DMA transfer.

Need of DMA :

A modest increases in hardware enables an I/O device to transfer block of information to or form 'M'(memory) without CPU intervention. For enabling this task I/O device should have to generate memory address and transfer data to or from the bus CPU initiating each block transfer.

Hence I/O device should require an interface between I/O data & main memory that can carry out transfer without program execution of CPU. Such I/O device interface circuit is called DMA_controller_& level of I/O channel is called Direct Memory Access (DMA) without CPU intervention.

Working of DMA & its benefits :

DMA comes into action due to following drawbacks of interrupt driven I/O & simple programmed I/O.

Drawbacks :

1. Data Transfer must transverse a path through a CPU.
2. The I/O transfer rate is limited by the speed with which the CPU can test & service a device.
3. The CPU is tied up in managing an I/O transfer, a number of instructions must be executed for each I/O transfer.
   When large volume of data are to be moved, a more efficient technique is required i.e. nothing but DMA.

DMA function :

DMA involve an additional module on the system bus called "DMA module", which is capable of taking over control of the system bus from the CPU.

Q.6(b) Compare between interrupt based data transfer and DMA based data transfer 10

Ans.:

	Interrupt based data Transfer	DMA based data Transfer
1.	In Interrupt based data Transfer the I/O devices are directly interrupted to the microprocessor by (INT) command.	In DMA based data transfer, the I/O devices send interrupt through DMA controller.
2.	The processor is involved in the data transfer to or from I/O device.	Processor is not getting involved in the process of data transfer.

3.	The processor is not loosing the control over the system bus at any time.	The processor looses the control over system bus and DMA controller will take charge over bus until the data transfer gets completed. It return the system buses to μP after completion of data transfer process.
4.	In main program if any interrupt occur it will execute corresponding ISR & after that it will return to next line of the same program.	There are various methods of DMA data transfer – Byte / cycle stealing mode – Burst / Demand mode – Continuous / Block mode.
5.	No extra hardware required in this method	Extra hardware required called as DMA controller (like 8237)

There are few similarities between Interrupt driven data transfer and DMA based data transfer.

1. Both Interrupt based and DMA based data transfers are used for transferring the data to or from I/O devices.
2. In both, I/O devices send a request to get served by the processor.
3. In both, memory read as well as I/O write operations can take place.
4. In both, Interrupt based & I/O based data transfer can take place through system buses only.
5. In both, microprocessor time is not wasted at all in the data transfer.

❑ ❑ ❑ ❑ ❑