Model Questions

palniraj21@gmail.com

Date. _____
Page No. _____

Long Answer Questions

1. What is meant by instruction set completeness? Is instruction set of basic computer complete? Discuss instruction cycle of basic computer with suitable flowchart.

Ans An instruction set is said to be complete if it contains sufficient instructions to perform operations in following categories:

A. Functional Instructions : Arithmetic, logic, and shift instructions
Examples: ADD, CMA, INC, CIR, CIL, AND, CLA

B. Transfer Instructions: Data transfers bet^n the main memory & the processor register Examples: LDA, STA

C. Control Instructions: Program Sequencing and control
Examples: BUN, BSA, ISZ

D. Input/output Instructions: Input and Output
Examples: INP, OUT

Instruction set of Basic Computer is complete because:

A. ADD, CMA (complement), INC can be used to perform addition and subtraction and CIR (Circular right shift), CIL (Circular left shift) instruction can be used to achieve any kind of shift operations. Addition subtraction and shifting can be used together to achieve multiplication and division. AND, CMA and CLA (clear Accumulator) can be used to achieve any logical operations.

B. LDA instruction moves data from memory to register and STA instruction moves data from register to memory.

C The branch instructions BUN, BSA and ISZ together with skip instruction provide the mechanism of program control and sequencing.

Date.
Page No.

D. INP instruction is used to read data from input device and OUT instruction is used to send data from processor to output device.

Instruction Cycle

In Basic Computer, a machine instruction is executed in the following cycle

1. Fetch an instruction from memory
2. Decode the instruction
3. Read the effective address from memory if the instruction has an indirect address
4. Execute the instruction

Upon the completion of step 4, control goes back to step 1. to fetch, decode and execute the next instruction. This process is continued indefinitely until HALT instruction is encountered.

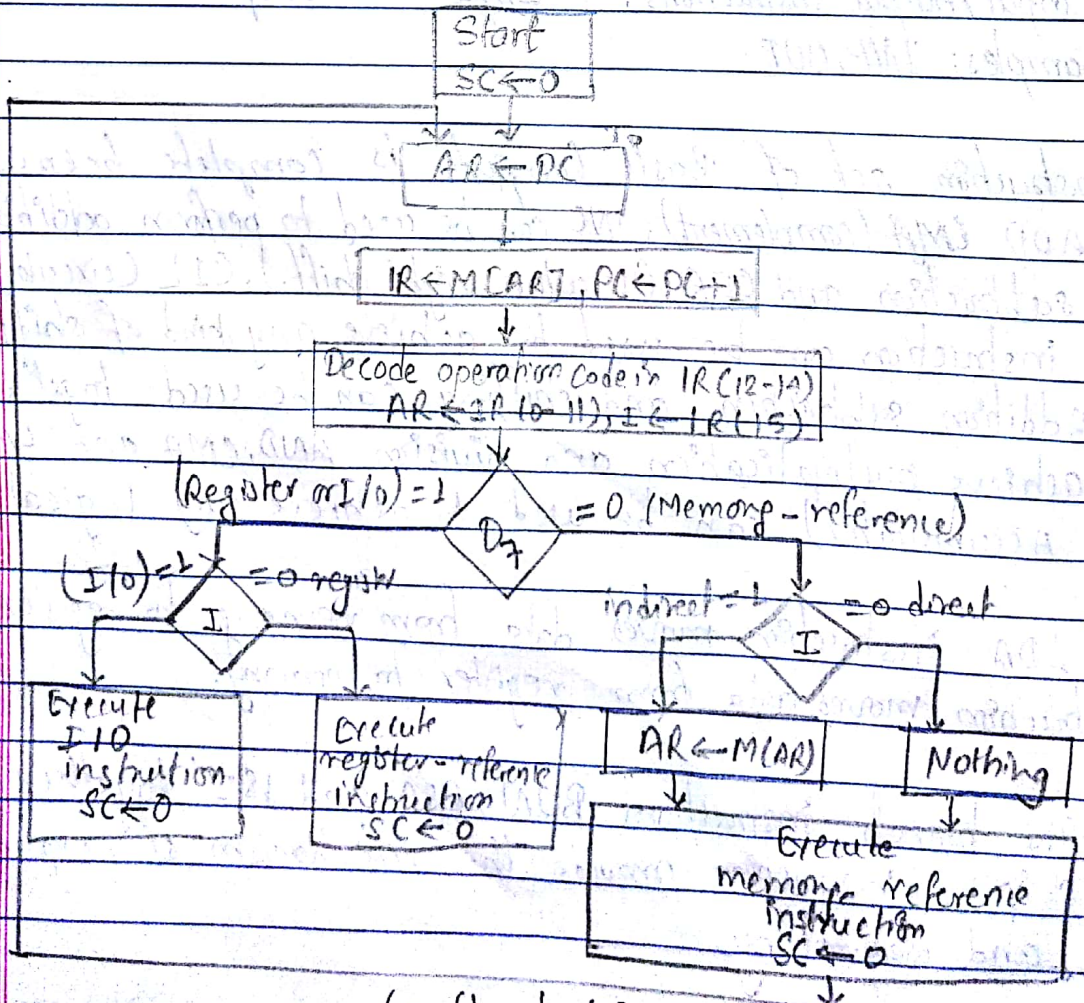


fig. flowchart for Instruction cycle

Date.
Page No.

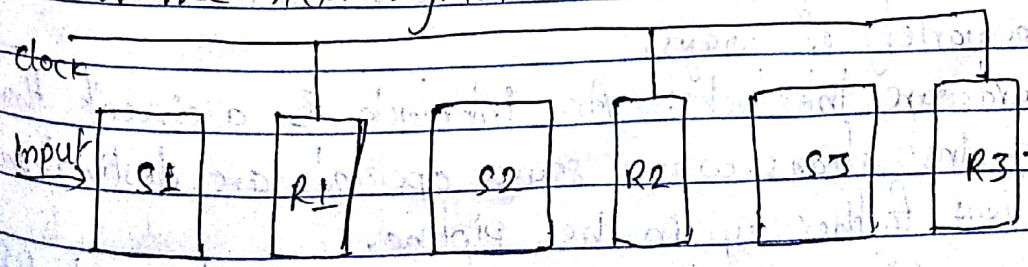
The timing signal that is active after decoding is T3. During time T3, the control unit determines the type of instruction that was just read from memory. Following presents an initial configuration for the instruction cycle and shows how the control determines the instruction type after decoding.

2. What is the concept behind pipeling? Diccuss different types of pipeline conflicts and their possible solutions briefly.

Ans

Pipeling 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.

Pipelining is a technique where multiple instructions are overlapped during execution. Pipeline is divided into stages and these stages are connected with one another to form a pipe like structure. Instruction enter from one end and exit from another end. Pipelining increases the overall instruction throughput.
In pipelining system, each segment consists of an input register followed by a combinational circuit. The register is used to hold data and combinational circuit performs operations on it. The output of combinational circuit is applied to the input register of the next segment.



Pipeline system is like the moder day assembly line setup infactories. for example in a car manufacturing industry, huge assembly lines

Date.
Page No.

are setup and at each point, there are robotic arms to perform a certain task, and then the car moves on ahead to the next arm.

Pipeline Conflicts

In general, there are three major difficulties that cause the instruction pipeline to deviate from its normal operations.

1. Resource Conflicts: caused by access to memory by two segments at the same time.
2. Data dependency: conflicts arise when an instruction depends on the results of a previous instruction, but this result is not yet available.
3. Branch difficulties: arise from branch and other instructions that change the value of PC.

Solutions

1. Resource Conflict Solution: It can be resolved by using separate instruction and data memory.
2. Data Dependency solution: A collision occured when an instruction cannot proceed because the previous instructions did not complete certain operations. A data dependency occurs when an instruction needs data and are not yet available. pipelined computers deal with such conflict beth data dependency in a variety of ways:
   a. Hardware Interlocks: An interlock is a circuit that detects instructions whose source operands are destinations of instructions farther up in the pipline.
   b. Operand forwarding: It uses a special hardware to detect a conflict and then avoid it by routing the data through special paths beth pipeline segments.

Date.
Page No.

c. Delayed Load: The compiler for such computers is designed to detect a data conflict & reorder the instruction necessary to delay the loading of the conflicting data by inserting no-operation instructn

3. Handling of branch instructions: One of the major problem in operating an instruction pipeline is the occurance of branch instructn. Pipelined computers employ various hardware techniques to minimize the performance degradation caused by instruction branching.

a. Prefetch Target Instructions: Both the branch target instruction & the instruction following the branch are pre-fetched and are saved until the branch instruction is executed. If branching occurs then branch traget instruction is continuous.

b. Branch target buffer (BTB): A branch target buffer is an associative memory included in fetch segment of the pipeline. Each entry in the BTB consists of the address of previously executed branch instructions and the target instruction of that branch. It also stores the next few instructions after the branch target instruction. This way, the branch instructions that have occured previously are readily available in the pipeline with out interruption.

c Loop buffer: This is a small very high speed register file maintained by the instruction fetch segment of the pipeline. When a program loop is detected, it is stored in the loop buffer including all the branches. This program loop can be executed directly without having access to memory.

d. Branch prediction: A pipeline with branch prediction uses some additional logic to guess the outcome of a conditional branch instruction before it is executed. The pipeline then begins prefetching the instruction stream from the predicted path.

e. Delayed Branch: In this procedure, Compiler detects the branch instructions, and re-arranges the machine language code by inserting useful instructions that keep the pipeline operating without interruption. It is used in RISC processors.
eg: No operation instruction

3. Multiply (-13) x (+40) using Booth multiplication algorithm.

Given $\leftarrow BR$ $\leftarrow QR$

~~13 = 000~~ 40 = 0101000, 13 = 0001101

1's comp. 13 = 1110010

2's complement of 13 i.e (-13) = 1110011

∴ Let QR = 0101000, BR = 1110011, $\overline{BR}$ = 0001100, $\overline{BR}+1$ = 0001101

AC	QR	Qn+1	SC	Action
0000000	0101000	0	7	
0000000	0010100	0	6	ASHR
0000000	0001010	0	5	ASHR
0000000	0000101	0	4	ASHR
+ $\overline{BR}+1$ 0001101				due to 10 AC ← AC + $\overline{BR}$ + 1
0001101			3	
0000110	1000010	1		ASHR
1110011				
1111001				AC ← AC + BR
1111100	1100001	0	2	ASHR
0001101				
0010001				AC ← AC + $\overline{BR}$ + 1
0000100	1110000	1	1	ASHR
+ 1110011				
1110111				AC ← AC + BR
1111011	1111000	0	0	ASHR

Here, we get, 1111011 1111000 & the most significant bit is called sign bit. It indicates the sign ie 1 indicates -ve sign. Now we have to find 2's complement of result except the sign bit. 520 = 0000010 0000 1000

1's comp 520 = 1111101 1111 0111

2's comp (-520) = 1111101 1111 1000 + 1

Hence (-13) x (+40) = -520

Ans

Date.
Page No. 7

Short Answer Questions

Q. What is overflow? Explain overflow detection process with signed and unsigned number addition with suitable example.

Ans When two numbers of n digits are added and the sum occupies n+1 digits, we say that an overflow has occured. A result that contains n+1 bits can't be accomodated in a register with a standard length of n-bits.

For this reason many computers detect the occurence of an overflow setting corresponding flip-flop.

An overflow may occurs if two numbers added are both +ve or -ve both. For eg: two signed binary no. +70 & +80 are stored in two 8-bit registers

Carries: 0 1		Carries: 1 0	
+70	0 1000110	-70	1 0111010
+80	0 1010000	-80	1 0110000
+150	1 0010110	-150	0 1101010

Since the sum of numbers 150 exceeds the capacity of the register (since 8-bit register can store values ranging from +127 to -128), hence the overflow.

Overflow detection:

An overflow condition can be detected by observing two carries: carry into the sign bit position and carry out of the sign bit position.

Example: Above 8-bit register, if we take the carry out of the sign bit position as a sign bit of the result, 9-bit answer so obtained will be correct. Since answer can not be accommodated within 8-bits, we say that an overflow occured.

If these two carries are equal ==> no overflow.

If these two carries are not same ==> overflow condition is produced.

If two carries are applied to an exclusive-OR gate, an overflow will be detected when output of the gate is equal to 1.

5. Write down different arithmetic Microoperations & design a 4-bit binary adder-subtractor.

Ans. The Basic Arithmetic microoperation are Addition, subtraction, increment, decrement.

The addition arithmetic microoperations are Add with carry, subtract with borrow, Transfer/Load.

Summary of typical arithmetic microoperation

symbole	Description
1. $R_3 \leftarrow R1 + R2$	contents of R1+R2 transfered to R3
2. $R3 \leftarrow R1 - R2$	contents of R1-R2 transfered to R3
3. $R2 \leftarrow \overline{R2}$	complement of contents of R2 (1's complement)
4. $R \leftarrow \overline{R2} + 1$	2's complement of R2
5. $R3 \leftarrow R1 + \overline{R2} + 1$	Content of R1 + 2's complement of R2 transfer to R3
6. $R1 \leftarrow R1 + 1$	Increment the contents of R1 by one
7. $R1 \leftarrow R1 - 1$	Decrement the contents of R2 by one

Binary Adder-Subtractor: The addition & subtraction operation can be combined into one common circuit by including an exclusive OR gate with each full adder.

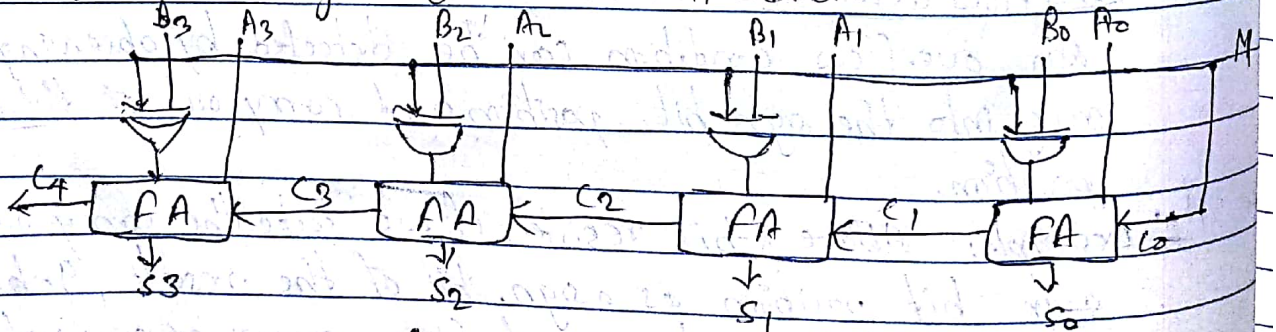


Fig: 4-bit Adder-subtractor

The Mode input M controls the operation:

- When M=0, the circuit is an adder & when M=1 the circuit is subtractor
- When M=0: $B \oplus M = B \oplus 0 = B$ i.e full adder receives the value of B, input carry is 0 & circuit performs A+B.
- When M=1: $B \oplus M = B \oplus 1 = \overline{B}$ & $C_0 = 1$ i.e B inputs are complemented & 1 is added through the input carry. The circuit performs A+2's complement of B.

Date.
Page No. 9

6. What is the purpose & advantages of common bus system? Explain common bust system of basic computer.

Ans The registers in the Basic Computer are connected using a bus. This gives a savings in circuitry over complete connections beth registers. Three control lines, S2, S1 and S0 control which register the bus selects as its input.

S_2	S_1	S_0	Register
0	0	0	X (nothing)
0	0	1	AR
0	1	0	PC
0	1	1	DR
1	0	0	AC
1	0	1	IR
1	1	0	TR
1	1	1	Memory

Either one of the registers will have its load signal activated, or the memory will have its read signal activated which will determine where the data from the bus gets loaded. The 12-bit registers, AR & PC, have 0's loaded. ~~The 12-bit register AR~~ onto the bus in the high order 4 bit positions. When the 8-bit register OUTR is loaded from the bus, the data comes from the low order 8 bits on the bus.

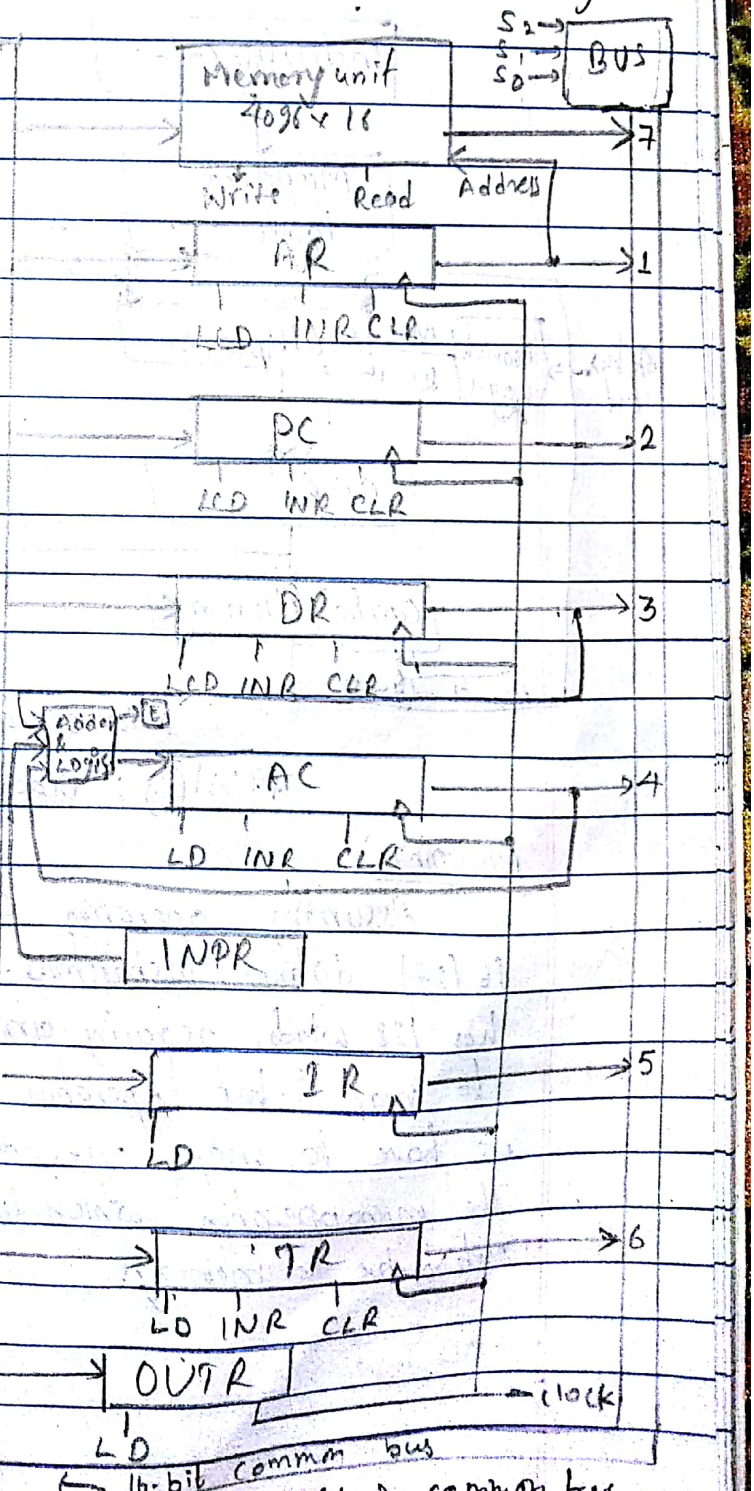


fig: Basic computer register connected in a common bus.

7. What is meant by address sequencing? Draw diagram of address sequencer & explain in detail about mapping of instruction.

Ans Each computer instruction has its own microprogram routine in control memory to generate the microoperation that execute the instruction. Process of finding address of next micro-instruction to be executed is called address sequencing.

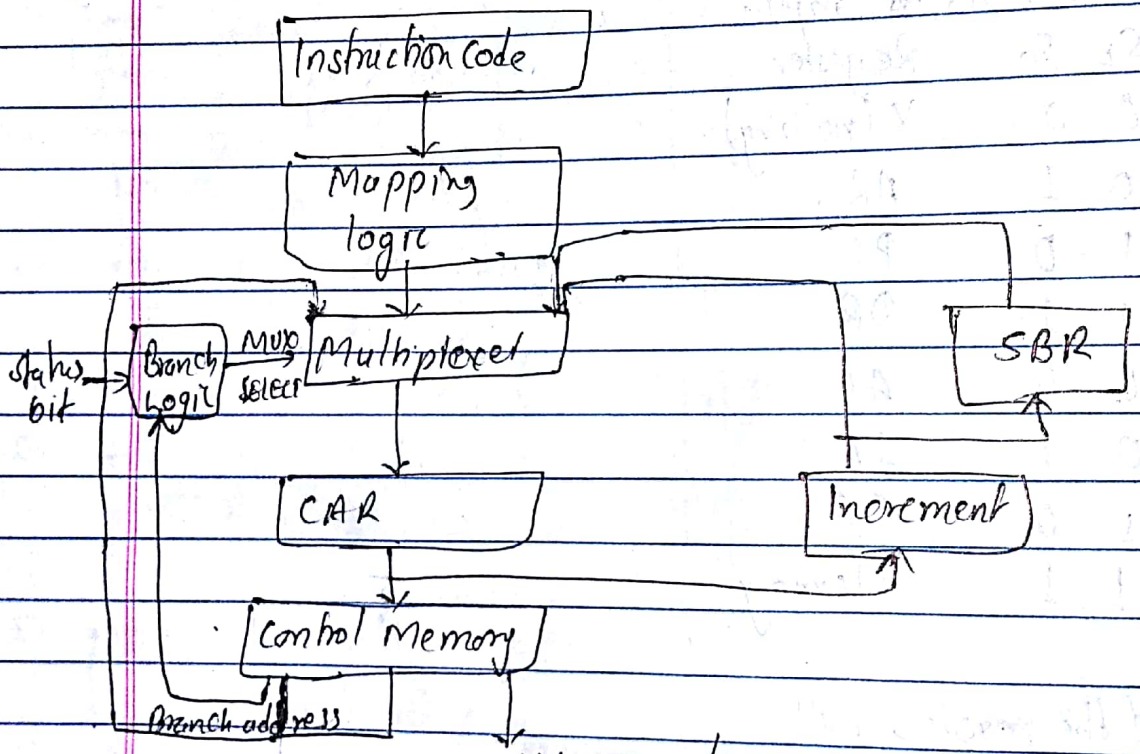


Fig: Block diagram of address sequencer.

Mapping:

Assuming operation code of 4 bits which can specify 16 (24) distinct instructions. Assume further & control memory has 128 words, requiring an address of 7 bits. Now we have to map 4-bit operation code into 7 bit control address. Thus, we have to map op-code of an instruction to be addressed the microoperation which is the starting microinstruction of its subroutine in memory.

Date.
Page No. 11

8. Consider that we are provided with the following memory content. What will be the value loaded in AC if addressing mode is direct, indirect, immediate, PC-relative & index relative.

PC = 200

R1 = 400

XR = 100

AC

Address	Memory
200	Load to AC | Mode
201	Address = 500
202	Next instruction
399	450
400	700
500	800
600	900
702	325
900	900

Ans We have 2-word instruction "load to AC" occupying address 200 and 201. First word specifies an operation code & mode & second part specifies an address part (500 here)

Mode field specify any one of a number of modes. For each possible mode we calculate effective address (EA) and operand that must be loaded into AC.

Direct addressing mode: EA = address field 500 and AC contains 800 at that time.

Immediate mode: Address part is taken as the operand itself. So AC = 500 (obviously, EA = 201 in this case)

Indirect mode: EA is stored at memory address 500. So EA = 800. And operand in AC is 300.

Relative mode:

PC relative: EA = PC + 500 = 702 & operand is 325. (since after fetch phase PC is incremented)

Indexed addressing: EA = XR + 500 = 600 and operand is 900.

Register mode: Operand is in R1, AC = 400

Register indirect mode: EA = 400, So AC = 700
Autoincrement mode; Same as register indirect except R1 is incremented to 401 after execution of the instruction.
Autodecrement mode: decrements R1 to 399, So AC is now 450.

Summary:

Address mode	Effective Address	Contents of AC
Direct address	500	800
Immediate operand	201	500
indirect address	800	300
Relative address	702	325
Indexed address	600	900
Register	-	400
Register indirect	400	700
Auto increment	400	700
Autodecrement	399	450

9. Define priority interrupt. Explain Daisy-chaining method of handling interrupt priority.

Ans Priority interrupt determines which interrupt is to be served first when two or more requests are made simultaneously. Also determine which interrupts are permitted to interrupt the computer while another is being serviced.
higher priority interrupts can make requests while servicing a lower priority interrupt.

Daisy-chaining Priority (serial)

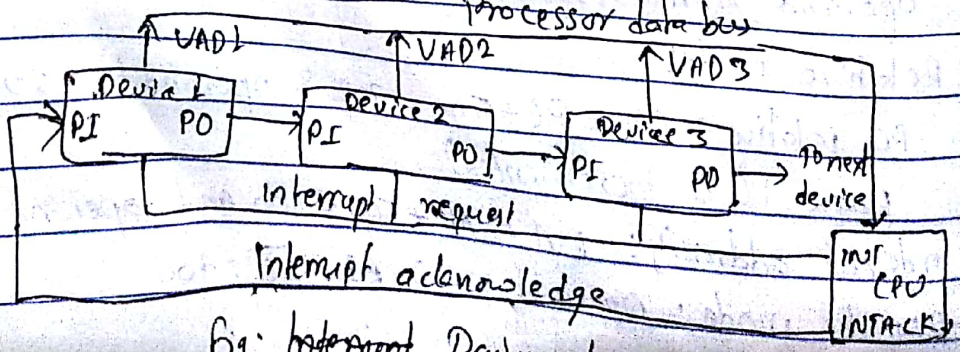
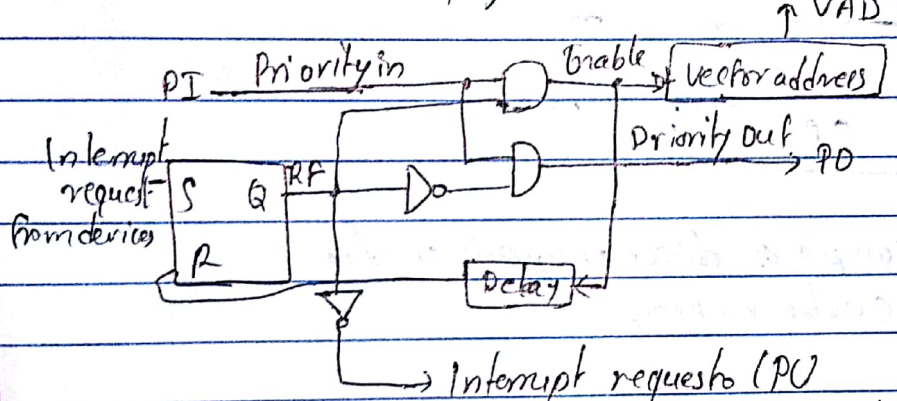


fig: Interrupt Daisy chain priority interrupt

Date.
Page No. 15

- Interrupt Request from any device
- CPU responds by INTACK
- Any device receives signal (INTACK) at PI puts the VAD on the bus.
- Among interrupt requesting devices the only devices which is i) physically closest to CPU gets INTACK and it blocks INTACK to propagate to the next device.



PI	PF	PO	Enable
0	0	0	0
0	1	0	0
1	0	1	0
1	1	1	1

Fig: one stage of Daisy chain priority arrangements

10. What is cache mapping? Explain direct mapping with suitable eg. examples

Ans If a particular address is found in the cache, the block of data is sent to the CPU, and the CPU goes about its operation until it requires something else from memory.
When the CPU finds what is needs in the cache, a hit has occured. When the address requested by the CPU is not in the cache, a miss has occured & the required address along with its block of data is brought into the cache according to how it is mapped.

Direct mapping
Main memory locations can only be copied into one location in the cache. This is accomplished by dividing main memory into pages that correspond in size with the cache.

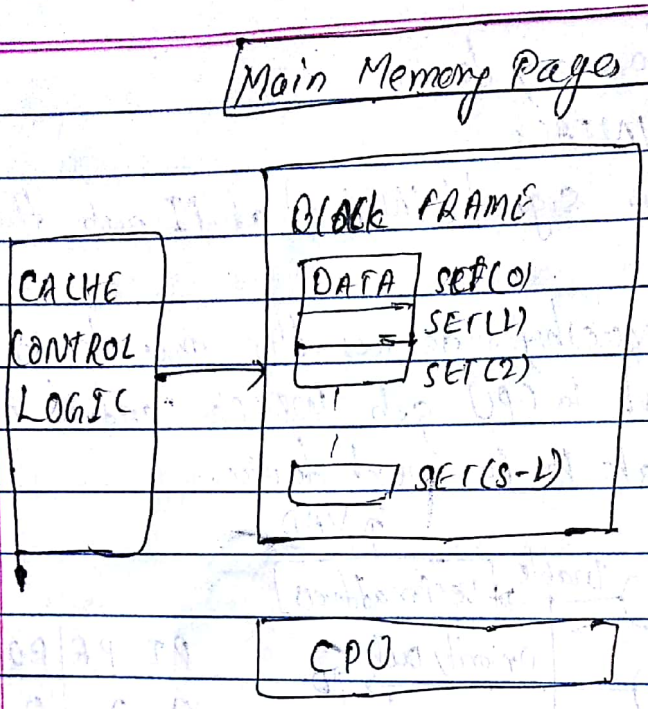


Fig: Example of direct mapping used in cache memory

11. What → space-time diagram? Discuss pipeline speedup eqn with suitable example.

Ans
- A task is the total operation performed going through all segment of pipeline.
→ The behaviour of a pipeline can be illustrated with a space time diagram.
→ This shows the segment utilization as a function of time.

Clock cycle → / segment ↓	1	2	3	4	5	6	7	8	9
1	T_1	T_2	T_3	T_4	T_5	T_6			
2		T_1	T_2	T_3	T_4	T_5	T_6		
3			T_1	T_2	T_3	T_4	T_5	T_6	
4				T_1	T_2	T_3	T_4	T_5	T_6

Fig: Space diagram for 4-segment pipeline

~~Speed~~

Date.
Page No. 15

Speed up

- k segment pipeline with a clock cycle time tp is used to execute n tasks.
- The first task Ti requires a time equal to ktp to complete its operation since there are k segments in the pipe.
- The remaining (n-1) task finish at the rate of one task per clock cycle will be completed after time (n-1)tp equal to (n-1)tp.
- As the number of task increases, n becomes larger than k-1 and k+n-1 approaches the value of n. Under this condition,

$$\text{Speed up} = \frac{tp}{tn}$$

12. Write short notes on:

a) Excess-3 code: The excess 3 code is a decimal code used in computers. This is un-weighed code.

Excess -3 code: BCD binary equivalent + 3 (0011)

Decimal	BCD	Excess-3
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

b. Input-Output Processor (IOP)

- IOP is a process with DMA capability that communicate with I/O devices. In this configuration, the computer system can be divided into a memory unit & a no. of processors comprised of CPU & one or more IOPs)

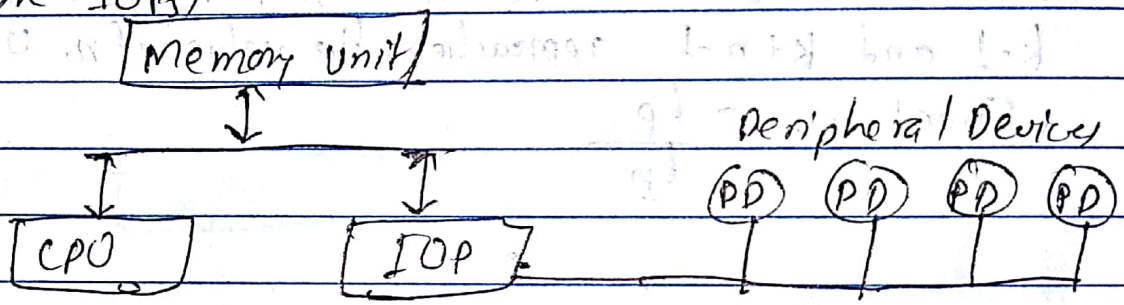


Fig: Block diagram of IOP.

*** Best of Luck ***