Computer Architecture and Microprocessors(PCIT-108)

Part-A

1.Basic Computer Organization and Design: Instruction Codes, Computer Registers, Computer Instructions, Timing and Control, Instruction Cycle, Memory Reference Instructions, I/O and Interrupt, Design of Basic Computer, Design of Accumulator Logic **[5 Hours]**

2.Programming the Basic Computer: Machine Language, Assembly Language, Assembler, Program Loops, Programming Arithmetic and Logic Operations, Subroutines, I/O Programming **[4 Hours]**

3.Central Processing Unit: General Register Organization, Stack Organization, Instruction Formats, Addressing Modes, Data Transfer and Manipulation, Program Control, Reduced Instruction Set Computer, Complex Instruction Set Computer **[5 Hours]**

4.Pipeline and Vector Processing: Parallel Processing, Pipelining, Arithmetic Pipeline, Instruction Pipeline, RISC Pipeline, Vector Processing, Array Processors **[5 Hours]**

Part-B

5.Memory Organization: Memory Hierarchy, Main Memory, Auxiliary Memo Associative Memory, Cache Memory, Virtual Memory, Memory Management Hardware **[5 Hours]**

→ 8085

Multiprocessors: Characteristics of Multiprocessors, Interconnection Structures, Interprocessor Arbitration, Interprocessor Communication and Synchronization, Cache Coherence **[4 Hours]**

6.Microprocessor and Microcontroller: 8051 Architecture, Pin Diagram of 8051, External Memory Interfacing, Stacks, Interfacing 8051 to LCD, Parallel and Serial ADC, DAC, Stepper Motor Interfacing and DC Motor Interfacing **[8 Hours]**

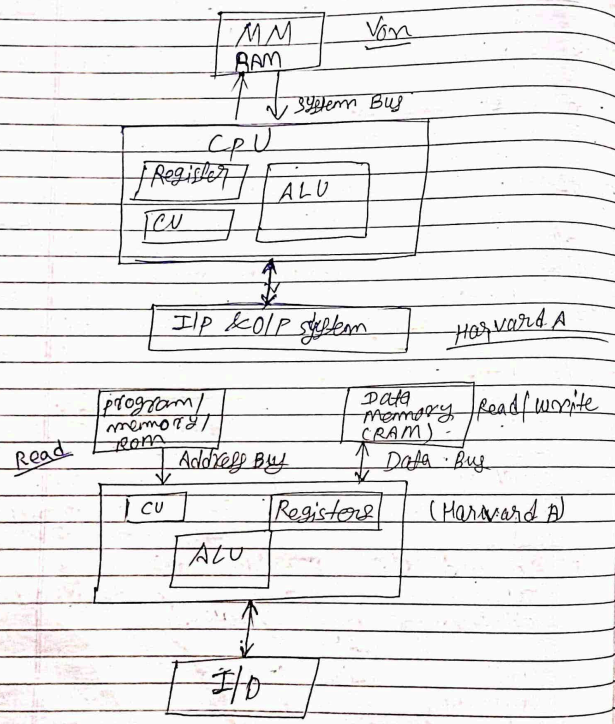
Applications and Trends of Microprocessor Technology: Memory and MPU Design, Development and Troubleshooting Tools, High-End-High-Performance Processors, Embedded Systems **[4 Hours]**

Q diff b/w Computer Architecture and Computer organization.

(i) parameters	Computer Architecture	Computer organization
Work	what the computer do	How the computer will do.
Design issues	High level design issue	Deal with low Level design issue
Attribute	Designing is attribute of computer Architecture.	Utilization of the resource in proper and planned manner.
what involves	CA involves instruction sets, instruction cods and modes.	CA organization it involves circuit design, ALU, CPU etc
Example	Ip, Ip → S, C	D, D → S, C
view	It provide External view of the system	It provides internal view of the system.

* CA and CO provides us with systematic approach to deriving solution to any problem.
* Diff Hardware / software
* Diff b/w system program / Application program
* what is CA and CO :-

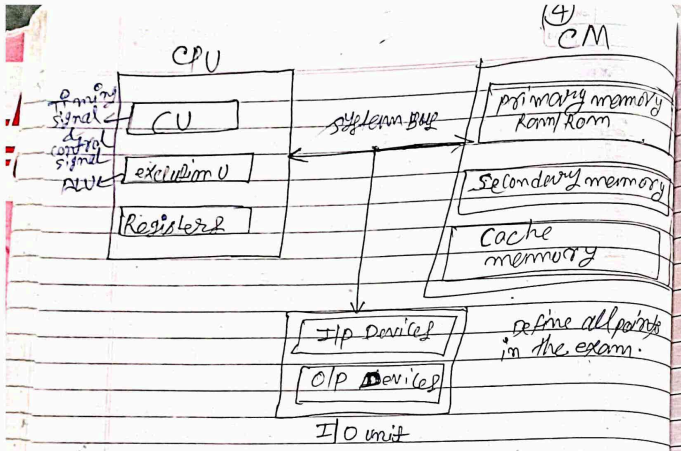
(2m) Diff b/w VON Neumann Architecture and Harvard Architecture



M.

Parameters	Von Neumann Arc	Harvard Arc
Memory	Data & program stores in the same memory	Data & program stores in the seprate memory
memory type	RAM for data & program	RAM → Data memory ROM → Program memory
Buses	Common Bus	Seprate data Bus and address Bus.
Program execution	program execution Serially & take more cycle.	program execution parally & take less cycle.
Machine	take more machine cycle	Machine take less machine cycle.
Control signal	less control signal	more control
Space	less space	more space
cost	less cost	more cost
Ex →	Digital signal processors	Modern microcontroller
Types of control signals	Memory read Memory write	Data → Memory Read & W Program → memory Read

(4)



define all parts in the exam.

Q.1

* why we are using register if we have main memory.

ans⇒

main memory: main memory refers to RAM (random access memory) is a type of volatile memory in a computer system that provides fast and temporary storage for actively running programs and data.

Register: fastest memory available till yet, it stores temporary data, it is smallest size memory. 8bit/16 byte. It is sequence of bits or sequence of flipflop. It stores intermediate or temporary results.

→ Data is present in the main memory and operation is performed on cpu that is ALU (speed ka exe speed is very fast

(5)

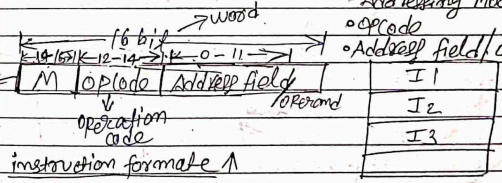
main memory comparable to ALU is very slow.

↓
Speed mismatch
↓
Burden → to overcome, insert register

- Control unit
  - ① → Timing signal
  - ② → Control signal
  - it will let you know
- (i) Timing signal: which instruction will execute first.
- (ii) Control signal: It will control, handle multiple registers.

Computer Instructions

Q.2 * Instruction: Instructions comprises of groups called as fields, field include
- Addressing Mode
- Opcode
- Address field/Operand



instruction formate ↑

WORD: "A 16 bit instruction is called as WORD"

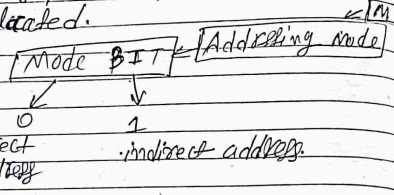
* Opcode: It specified to the what operation to performed.

Ex⇒ [2+3]
↳ Opcode (0000-1101)

opcode ≠ 111 = 2^3.

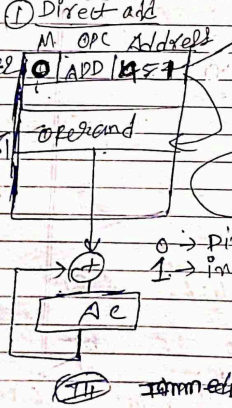
Address field : It containg the location of operand that is registors / memory.

* Mode Bit : M specifie how the operand will be located.

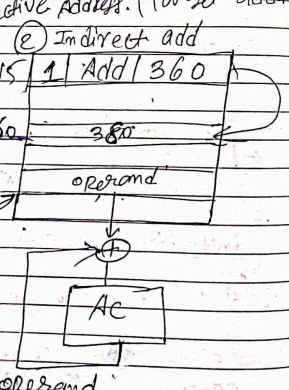


* Instruction Code : An instruction code is a group of bits that instruct the computer to perform a specific operation. There are three addressing modes used for address portion of instruction code.

(i) Direct add

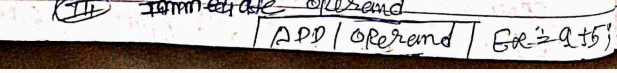


(e) Indirect add



0 → Direct Ad
1 → indirect

(iii) ~~Immediate~~ operand



* what is effective address

Ans * what is inside in effective effective address.

In effective address there is target address of the operand.

* How to write a instruction

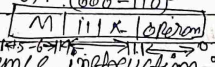
Ans ⇒ AC ← AC + M[AR] → memory address of address register

accumulator

* Instruction → Memory Reference instruction
→ Register reference Instruction
→ Input/output instruction

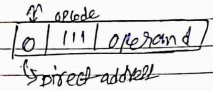
A basic computer has three types of instruction code formate.

(i) Memory reference instruction : In Memory reference instru 12 bits of memory is used to specify the address and 1 bit is used to specify the addressing mode (I).

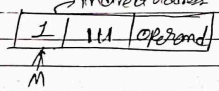


(ii) Register reference instruction : The register reference instruction is represented by

the opcode 111 with 0 in the left ~~with the 0 on the left~~ bit of the instruction

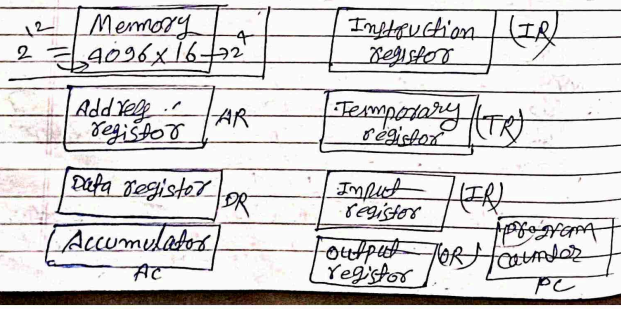


(iii) I/p img 8 : The I/p img is represented by the opcode to be 111 and the 1 in the left most bit of the instruction.



* Computer registor : Register is memory element which is used to store the temporary data, it is made of flip-flops.

Types of Computer registor



Reg name	Reg. symbol	No of Bits	function
Address registor	AR	12 Bits (0-11)	It stores the address of operand
data registor	DR	16 Bits (0-15)	It provides data, whole instruction
Accumulator	AC	16 Bits (0-15)	It holds the intermediate data
program counter	PC	12 Bits (0-11)	It stores address of the instruction
Instruction registor	IR	16 Bits (0-15)	It Stores the complete instruction formate
Temporary registor	TR	16 bits	It holds the temporary data
Input registor	IR	minimum 8 bit more as per requirement	It takes information from the user
output registor	OR	minimum(8) more as per requirement	It prints the output on the monitor, printer

Q.12 Marks: Write a program to add two bit no, where sum is 8 bits.

Memory Address	Machine Mnemonic Code	opcode/data address	Mnemonic	Comments
9000	3E	Opcode	MVI A, 02H	Moves the data to be inputted to AC.
9001	02	Data	—	The data entered here will be inputted to the
9002	06	Opcode	MVI B, 02H	This moves the inputted data to register B
9003	02	Data	—	Data entered here will be stored in register B
9004	80	Opcode	ADD B	A=A+B adds Contents of ACC and B then stores it in Acc.
9005	32	Opcode	STA 9502	stores the contents of Acc to specific memory location.
9006	02	Address L	—	Address of lower Bits
9007	95	Address H	—	Address Higher Bits
9008	76	Opcode	HLT	Stops the program

Explain:

Step-1 : Press RESET

Step-2 : Press Exe mem

Step 3 : insert 9000 address to move the data to Ac. and press next.

Step4 : put the data (07) which you want to perform operating at 9001 address this data will be go to the AC. and press next. (07)

Step5 : at 9002 this moves the second data to register B.

Step 6 : at 9003 enter second data that will be stored in register B.

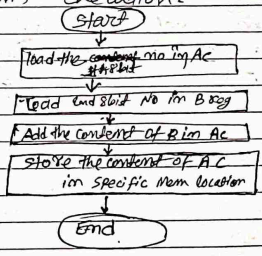
Step 7 : at 9003 chose the Mnemonic code what do you want to perform with that data for exam addition enter Mnemonic code 80 for subtraction enter 90. After that this will add content of Ac and B and again stores in Ac and press next.

Step 8 : At 9005 it stored the input where you want to store result at specific memory location then press next.

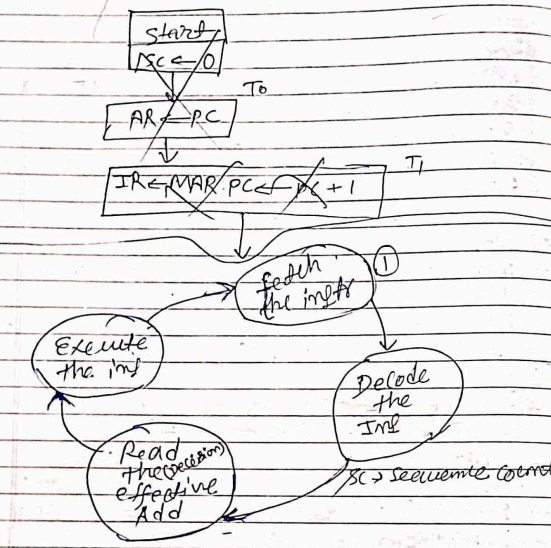
Step 9 : at 9006 Enter your lower bits of address that will be stored in L register and press Next

Step 10 : at 9007 Enter higher bits of address that will be stored in H register and press next.

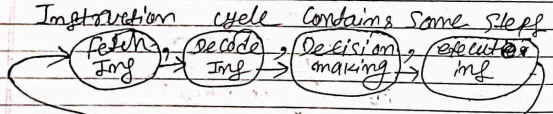
Step 11 : 9008 (HLT) Now press next to stop the program execution.



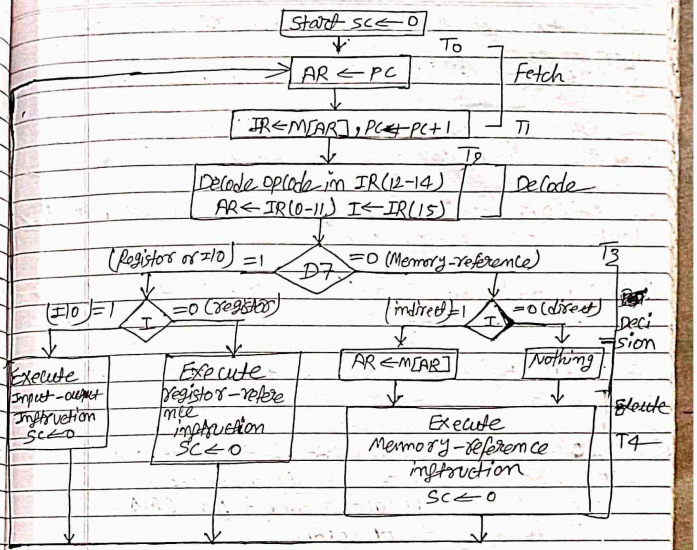
Introduction cycle



Instruction cycle contains some steps



Instruction cycle



step 1: Start SC ← 0
step 2: T0 : AR ← PC } fetch from memory
step 3: T1 : IR ← M[AR], PC ← PC+1 }
step 4: T2 : decode opcode IR(12-14) } decode
AR ← IR(0-11), I ← IR(15)
step 5: T3 : decision
D7 → indirect
D7 → direct T3 cycle
· Memory reference
· register reference instruction
· I/O reference instruction
step 6: T4 : execute

pyq -> Draw ~~the~~ And Explain the Architecture of the 8085

fetch cycle

- start sequence counter to zero before fetching.
- and timing pulse start from T0
- during T0 the instruction which are present in program counter place that instruction into address register.
- After that Program counter will be incremented by 1.

Decode cycle

- After fetching the instruction, decode the instruction during pulse T2.
- what operation has to be performed that is stored in IR(12-14) and address of operand at IR(0-11) and whether the address is direct or indirect that is in IR(15).
- decode the opcode of instruction.

Decision cycle & Execute cycle

at T3

- Before Execute we have to take decision ~~what~~ wheather the data is 0 or 1 from 000-110 it will 0 and if it will 111 so Data will be D7 memg 1

(i) Let's Consider D7 = 0

- D7 = 0 mean it is memory reference
- After that Up will check wheathe I is 0 or 1
- if I = 0 memg direct address so do nothing
- execute the memory reference instruction
- make make sequence counter to SC → 0

- If the I=1 means indirect address then place the effective address into address register.
- After that execute memory reference instruction make sequence counter to zero.

Again let's Consider when D7 = 1

- If D7 = 1 up will check I whether the I is 0 or 1
- If the I = 0 means It is register reference
- After that execute register-reference instruction and make sequence counter to 0 →
- If I = 1 means Input/output instruction
- After that Execute the input-output instruction and make the sequence counter to zero.

(2M) Demonstrate the execution of the following instructions
(i) LDA addr (ii) ADC r (iii) CMA (iv) PUSH rp.

(i) LDA addr (Load Accumulator): This instruction loads the accumulator with the content of the memory location specified by the address 'addr'.
Ex→ if 'LDA 2000' → it loads the accumulator with the content of memory location 2000.

(2) ADC r (Add with carry): This instruction adds the contents of register 'r' along with the content of the accumulator.
Ex→ 'ADC B' it adds the content of register B to the accumulator.

(3) CMA (Complement Accumulator): This instruction flips the contents of the accumulator.
Ex→ if Ac has 10101010 and After CMA→ it becomes → 01010101.

(4) PUSH rp (Push register pair onto the stack): This instruction pushes the content of the register pair 'rp' onto the stack.
Ex→ if 'PUSH BC' it pushes the content of registers B and C onto the stack.

Q.9 → Draw And Explain the Architecture of the 8085 microprocessor.

Microprocessor is of 8-bit introduced in 1977. Its actual name is 8085A. It contains 6200 transistors approx. Its dimensions are 16mm x 22mm. It is having 40 pins dual inline-package.

The architecture of 8085 microprocessor consists of several components including Accumulator, registers, program counter, stack pointer, instruction register, decoder, flags register, data bus, address bus and control bus, ALU.

* Multiplexer: Microprocessor have 6 general purpose registers B, C, D, E, H, L each of 8 bits that can be used as 16 bit pair register BC, DE and HL only combine with each other. these registors are used to store memory addresses and data along with that there is also W and Z temporary register but it cannot be used by programmers each of 8 bit.

* PC →
In multiplexer SP and PC, INC/DEC are used. The program counter is a 16-bit register that contains address of the next instruction after fetching one instruction. PC is incremented

*INC/DEC: inc/dec is used to increment/decrement the memory location by 1.

* SP → Stack pointer is a 16-bit register that is used to manage the stack and store data temporarily. It is also used to keep track of the top of the stack.

* Data bus: The data bus is an 8-bit bus that is used to transfer data between the microprocessor and memory

*Address buffer & address-data buffer: The content stored in the stack pointer and program counter is loaded into the address buffer and address data buffer to communicate b/w CPU.

* Address bus: Address bus is a 16 bit bus that is used to address memory (16 bit) and other devices. also it is used to select the memory location (out) that the microprocessor wants to access.
(A15-A8) AB

* Control bus: Control bus is a set of signals that controls the operations of up including read, write, reset, interrupt operations.
(AD7-AD0) DB

* Instruction register: Instruction register is an 8 bit register that contains the current instruction that is being executed.

* Instruction Decoder: Instruction decoder is used to decode the instruction coming from instruction register.

* Timing and control unit: It provides timing and control to the microprocessor to perform operations. Timing and control signals are: READY, RD, $\overline{WR}$, ALE S0, S1, IO/$\overline{M}$, RESET IN, RESET OUT, HLDA, HOLD.

* Accumulator: It is an 8-bit register used to perform arithmetic, logical I/O & load/store operations. It is connected to internal data bus & ALU.

* Temp register: It is an 8-bit register, which holds (from) → the temporary data of arithmetic and logical operations.

* Flag register: Flag register is an 8-bit register that contains status of the result. These flags include ① Sign flag (+ve, -ve) ② zero flag (zero) ③ parity flag (even no. of 1 bits) ④ carry flag (carry)

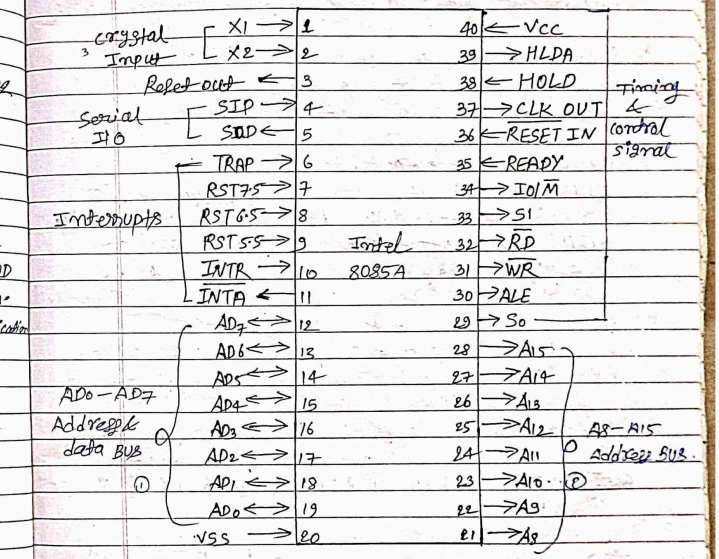
* Arithmetic and logic unit (ALU): As the name suggest, it performs arithmetic and logic operations like addition, subtraction, AND, OR etc. ALU is of 8 bit.

* Interrupt control: As the name suggests it controls the interrupts during a process. When a microprocessor is executing a main program and whenever an interrupt occurs, the microprocessor shifts the control from the main program to process the incoming request. After the request is completed, the control goes back to the main program. There are 5 interrupt signals ① INTR ② RST 7.5 ③ RST 6.5 ④ RST 5.5, ⑤ TRAP

priorities of interrupts → TRAP > RST 7.5 > RST 6.5 > RST 5.5 > INR

* Serial Input/output control: It controls the serial data communication by using SID and SOD. The SID pin is used for serial I communication. The SOD pin is used for serial output communication.

Q PIN Diagram of 8085 microprocessor:
Microprocessor introduced in 1977. It is of 8-bit up, it's actual name is 8085A, it contains 6200 transistors approx. It's dimensions are 164mm x 222 mm. It is having 40 pins & Dual inline package. Ibacdp → abcdIP



1. Data BUS (AD0–AD7): This is 8-bit bidirectional bus which transmits and receives data between the microprocessor and memory or I/O device.

2. Address Bus (A0 – A15): These 16-pins carry 2 memory addresses and I/O port address during different operations. (8×2)

3. Crystal input (1 & 2): These two pins are connected to an external crystal or resonator that maintain internal frequency of microprocessor.

4. Reset out (3): This signal indicates that the microprocessor unit is being reset. The signal can be used to reset other devices.

5. Serial Input/output (4-5):
SID: Serial input data is a data line for serial input. It takes data from Ac using RIM.
SOD: Serial output data is a data line for serial output. It take 1 bit from Ac to Serial port 8085.

6. power supply & clock frequency:
- VCC (40): +5V power supply to active the IC of Up.
- VSS (20): Ground reference or ground connection for Up. (Answer: discuss various types of interrupts in 8085)

7. Interrupts: 8085 has five interrupt signals that it is used to interrupt a program execution.
(i) TRAP (ii) RST 7.5 (iii) RST 6.5 (iv) RST 5.5 (v) INTR

priority of interrupts:
TRAP > RST 7.5 > RST 6.5 > RST 5.5 > INTR
(Non vectored; Maskable interrupt; Non maskable interrupt; Vectored Int)

(1) TRAP: It is non maskable interrupt
- It has highest priority.
- TRAP is used for power failure and emergency shutoff. Ex ⇒ Power failure, system reset

- RST 7.5: • It is maskable interrupt,
• It has the second highest priority, it is positive edge triggered only.
Ex ⇒ real time clock updates
- RST 6.5: It is maskable interrupt, It has third highest priority, it is level triggered only. Ex ⇒ I/O operations or data transfer
- RST 5.5: It is a maskable interrupt. It has the fourth highest priority. It is also level triggered, this interrupt is very similar to RST 6.5. Ex ⇒ background processes
- INTR: It is a maskable interrupt, It has the lowest priority, It is a general purpose interrupt request signal.
Ex ⇒ keyboard interrupt.
- INTA: It is an acknowledgement signal sent by the microprocessor after INTR is received. It is an active low signal.

Timing & control signal:
- HLDA: HLDA stands for Hold Acknowledge. The Up use this pin to acknowledge the HOLD signal.
- HOLD: It indicates that another device is requesting the use of address and data bus. The microprocessor stops using the address and data bus and resumes after finishing the current cycle.
- clock out: This signal can be used as the system clock for other devices.

- Reset IN: It is active low signal used to reset the microprocessor.
- READY: It indicates that the ~~device~~ µP is ready to send or receive data. if READY is low then the CPU has to wait for READY to go high. (synchronize the external device)
- IO/$\overline{M}$: This ~~signal~~ pin is used to differentiate between IO and memory operations when it is high then it indicates IO and when it is low then it indicates memory operation.
- S1, S0: It indicates the current operation being performed by the microprocessor.
- $\overline{RD}$: This is the active-low signal that indicates that the microprocessor is ready to read data from memory or I/O device.
- $\overline{WR}$: This is the active-low signal that indicates that the microprocessor wants to write data to memory or I/O device.
- ALE: It is an Address latch Enable signal, when it is high it indicates address and when it down it indicates data.

Maskable interrupt: Interrupts which can be ignored by µP for sometimes which can be disable or enable.
Non-maskable → TRAP.

Q Difference b/w Assembly language and machine language.

Assembly language	Machine language
(i) Assembly language is a low-level language that uses mnemonics and symbols to represent machine instructions, it's a human readable machine code.	Machine language is the lowest level programming language that it uses binary digits either 0 or 1 to represent instructions. It is machine readable.
(ii) ONly understand by ~~computer~~ human	only understand by computer
(iii) It is represented by mnemonics such as MOV, Add, Sub, LDA;	It is represented by 0 or 1s.
(iv) Easy to understand.	difficult to understand.
(v) Execution is slow as ML	Execution is fast as AL.
(vi) It is machine dependent and it is not portable.	It is hardware dependent
(vii) No need of translator.	Need of translator to convert mnemonics into machine language.
(viii) Modification and error debugging can be done.	Modification and errors ~~bedu~~ debugging cannot be done

Trick = DUETM

PYQ → diff b/w Microprocessor and microcontroller.

Microprocessor	Microcontroller
Definition → Microprocessor is a processor that execute instruction one by one and control I/O devices.	A microcontroller is a small computer on a single integrated circuit that is designed to control specific task.
(2) Speed → Has a high clock speed	Has a lower clock speed.
(3) Cost → Too costly.	Less costly.
(4) Size → large in size	small in size
(5) Applic → computers, laptops	Microwave, washing machine
(6) It is 32 bit or 64 bit architecture.	It is of 8, 16 or 32 bit architecture.
(7) It requires More memory	It requires less memory.
(8) Memory, I/O ports, timers etc. connected to the CPU externally	CPU and all other elements are integrated into a single chip.
(9) Use → widely use in computer system	widely use in embedded system.

12 Marks PYQ

* Addressing Modes: The different ways of specifing the location of an operand in an instruction are called Addressing Mode.

Types of Addressing Mode

(1) No address field is required
   (1) Implied Mode / Implicit
   (2) Immediate Mode
(2) Address specifies the memory location.
   (3) direct Mode
   (4) Indirect Mode
(3) Address specifies the processor register
   (5) register direct Mode
   (6) register indirect Mode
   (7) Auto increment and auto decrement Mode.
   (8) Address field plus content of CPU register [displacement Addressing]
   (9) Relative addressing Mode
   (10) Index Addressing Mode
   (11) Base register Addressing Mode

(1) Implied Mode: Operands are specified implicitely in the definition of instruction.

Ex → CMA,	CLA
Complement the content of Accumulator	clear the content of Accumulator

(2) Immediate Mode : Operand is specified in the instruction itself.

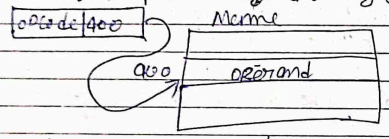
Example →

ADD R1, #7

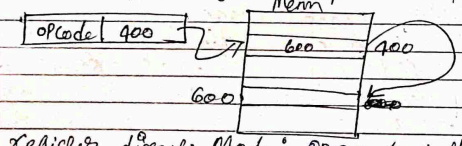
R1 → R1 + 7

M	opcode	#7

(3) Direct Mode : In this actual/effective address of operand is directly given in the instruction.



(4) Indirect Mode : In this effective address of operand is not directly given in instruction, there is next a memory address where operand is present

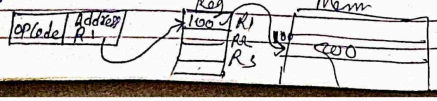


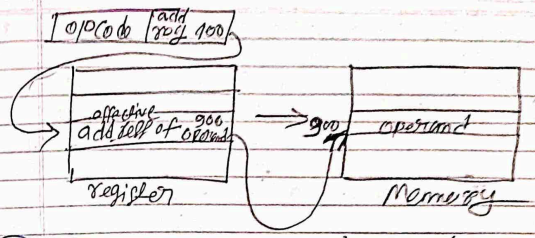
(5) Register direct Mode : Operand in the register. address field of the instruction refers to the CPU register that contain the operand.

Ex → MOV R1, R2

(6) Register indirect Mode : Register contains address of operand rather than operand itself.

LD(R1)





(7) Auto increment and auto decrement Mode : It is similar to register indirect Mode, effective address of the operand is a content of the register specified in the instruction.

In auto increment Mode register is increemented after its value is used to access the memory.

Ex → EA = (R1) +

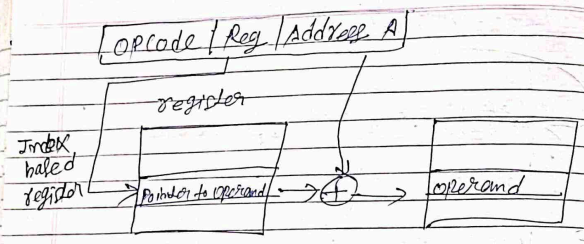
EA = (R1)

In auto decrement Mode the register is decremented before its value is used to access the memory.

(8) ~~Displacement Addressing~~ Relative addressing Mode : The operand's address is specified relative to the program counter

Ex → Relative Add = PC + address part

(9) Indexed addressing Mode : Index is added to a base address to calculate the operand's memory → Ex → Indexed add = XR + address part



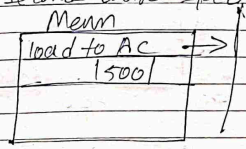
DI's = Add + → offset value.

Note → relative addressing Mode:
→ reg → program counter.
→ index → index reg
→ base → base reg.

Effective Address = pc + offset

(10) Base register addressing mode: A base register holds a base address and an offset value is added to this base address to determine the memory address of the operand.

(9) the two word ing there in address 200 at address 201 there is a instruction named as load to Accumulating, the first word of specified load opcode and second word specified address. pc has value



the first word of instruction specified the operation code and mode and second word specified the address.

PC is the value of 200 for the fetching the instruction.

the content of processor reg → 400
and the content of index reg is 100

find the effe for the each possible Mode calculate the effective add and the operand loaded in the memory.

* Numericle on Addressing Mode

PC = 200

R1 = 400

XR = 100

AC

Memory

	Load to AC	Mode
200	Address = 500	
201	Next instruction	
399	450	
400	700	
500	800	
600	900	
702	325	
800	300	

Q For each possible mode we calculate the effective address and the operand that must be loaded into AC.

(1) immediate mode → operand value in address field
EA = 201 , AC = 500

(2) Direct address mode → Address field containing effective address
EA = 500 AC = 800

(3) Indirect address Mode : Address field specifies the address where the effective address of operand is stored in memory
EA = 800 AC = 300

(4) Register direct Mode : Register Containing Operand
EA = — AC = 700

(5) Register indirect mode : Register Containing effective address of Operand.
EA = 400 AC = 700

(6) Auto-increment Mode : Like register indirect mode except register value is incremented after it is used.
EA = 400 AC = 700
just like register indirect except

(7) Auto decrement Mode : value is decremented before it is used.
EA = 399 AC = 450

(8) Relative Address : PC is added to address part
Pc + address part
202 + 500 = 702
EA = 702 AC = 325

(9) Indexed address mode : Index register is added to address part
XR + address part
100 + 500 = 600
EA = 600 AC = 900

(10) Base reg = + address field

		EA	AC
*	Immediate addres	201	500
(2)	Direct address	500	800
(3)	Indirect address	800	300
(4)	Register direct Mode	—	400
(5)	Register indirect mode	400	700
(6)	Auto decrement Mode	400	700
(7)	Auto decrement mode	399	450
(8)	Relative address	702	325
(9)	Indexed address Mode	600	900

(PYQ) what are addressing Modes. An instruction is stored at location 400 with its address field at location 401. The address field has the value 500. A processor register R contains the number 100. Evaluate the effective address if the addressing mode of the ins is (a) direct (b) immediate (c) relative (d) register indirect (e) index with R as the index register.

R = 100 | Idx = 100
PC = 400
EA = ?

400	Mode / opcode
401	Address = 500
500	operand

(a) → EA = 500
(b) → EA = 401 AC = 500
(c) PC + Address = 402 + 500 = 902
(d) 100 EA =
(e) index + address
100 + 500 = 600

Q An instruction is stored at location 300 with its address field at location 301. The address field has the value 400. A processor register R1 contains the number 200. Evaluate the effective address if the addressing mode of the instruction.
(a) Direct
(b) immediate
(c) Relative
(d) Register direct
(e) index with R1 as the index register

PC = 300
R1 = 200

300	Mode / opcode
301	Address = 400
400	operand

EA =
(a) 400 (b) EA = 301
(c) 302 + 400 = 702
(d) 200
(e) 200 + 400 = 600

* Flag register: flag reg tells about status of current result, flag is change by ALU after every operation.

S	Z		Ac		P		CY

(1) Carry flag: It tells about whether there is carry or not
A = 10H, B = 20H ADD B → AC ← AC + B
CY → 0 CY = 1
(2) Parity flag: It is calculated by no of 1's.
Ex → 10001 0000 → No of 1's → 2 → P = 1
10000000 → No of 1's → 1 → P = 0
(3) Auxulicary flag: Carry from lower nibel to higher nibel is there
AC = 0
0010 0000
1101 0000
0001 0000
0010 0000
AC → 1
(4) zero flag: It tells wheather the flag is zero or not: 80 H, 80 H
10000000
10000000
Z = 1
00011
01011
01110 → Z = 0

(5) Sign flag :- It tells the sign of result. by MSB one. Can know the no is +ve or -ve, it directly copy the MSB of the result.

MSB → 1 -ve
→ 0 +ve

Ex →
10000000
10000000
100000000
Result
0 → +ve
if 1 → -ve

* Why register is embedded in cpu chip.

ans → register is embedded in CPU chip for quick access to store, and retrieve data during the execution of instructions.

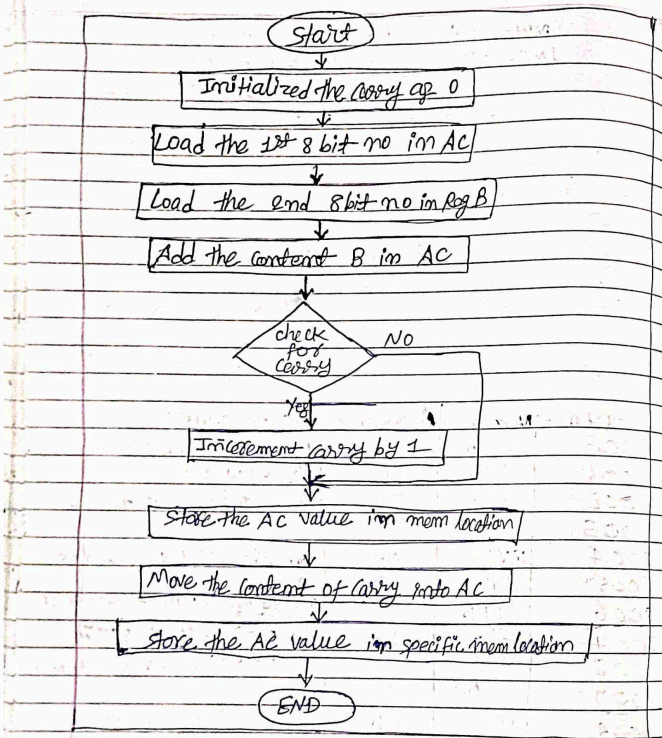
Q write a assembly language program to add two 8 bit no where sum = 16 bit.

INP1 → 84 1000,0100
INP2 → 75 0111,0101
C=00 1111 1001 op=f9
f 9

INP1 → 90
INP2 → 90 1001,0000
1001,0000
C=1 0010 0000
↓ 8bit → 20 op = 20 ↓ 8bit
16 bit

Memory Add.	Mnemonic	Machine Code	comments
8000	MVI A	3E	Loads the number to the Accumulator
8001	90H	90	
8002	MVI B	06	loads the number to the register B
8003	90H	90	
8004	MVI C	0E	Loads the carry generated to register.
8005	00H	00	
8006	ADD B	80	Add the number in register B to register A.
8007	INC	D2	Jump to next step if
8008	JMP	0B	if no carry is generated
8009	AHEAD	80	
800A	INR C	0C	Increment the carry
800B	AHEAD	32	stores the result
800C	STA	03	at 8503H
800D		85	
800E	MOV A,C	79	Move carry to AC
800F		32	Stores the carry
8010	STA	04	generated in 8504H
8011		85	
8012	HLT	76	STOP

Flowchat



Explain → Imp1 = 90 , Imp2 = 90
move 90 in → AC (MVI A)
move 90 in → B Reg (MVI B)
initialize carry by 0 (MVI C, 00)
Add B in Ac : (ADD B)
If No carry Jump for Store (INC).
INC Carry by 1 · (INR C)

Q why we take hexadecimal in CAM.

and move carry into AC → (MOV A, C)
Store the content of AC in memory (STA)
MLT the program execution (HLT)

output = 90
carry = 01

MST-2

4Marks

* Instruction set : An instruction is binary pattern, designed inside Up to perform a specific function.
→ The entire group of instructions that o Up support is called instruction set. each ing is represented by an 8 bit binary value. 8085 has 246 instructions

* Instruction : The 8 bits of binary value is called as Instru byte.

* classification of Ins byte :

(i) Data Transfer Ins | Arithmetic Ins | Logical Ins | Branching Ins | I/O or M/C control Ins

Answer key (Oct) ① Data Transfer Instruction

Discuss various memory reference instructions

Mnemonic	operand	Description	Example
① Move	Rd, RS	Copy from	Move B, C
	~~[illegible]~~		
	M, RS		Move B, M
	Rd, M (H,L) (20,30)		
② MVI	Rd, Data	Move immediate the 8 bit data	MVI B, 07
	M, Data		MVI M, 57 (H,L) (20,30)
③ LDA	16 bit address	Load Accumulator	LDA 2034H
④ LDAX	B/D register pair	Load Accumulator indirect	LDAX B
⑤ LXI	Register pair, 16 bit data	load the register pair immediately	LXI H, 2034 H
⑥ LHLD	16 bit address	load HL pair direct	LHLD 2040 H
⑦ STA	16 bit address	store accumulator direct	STA 2500 H
⑧ STAX	register pair	store Ac indirect	STAX B
⑨ SHLD	16 bit address	Store HL pair direct.	SHLD 2550 H
⑩ SCHG	None	Exchang the HL pair with DE	
⑪ SPHL	None	copy the HL pair to the stack pointer	
⑫ PCHL	None	load the Program counter with HL content	

④ Data transfer instruction: These instructions move the data between registers or b/w memory and register. if one of the operand is an memory location its location is specified by contents of HL register.

⑥ LHLD: this instruction copy the contents of memory location pointed by 16 bit address into register L.
It copies the contents of next memory location into register H.

4 marks

1. (13) PUSH	register pair	push the register pair into the stack	PUSH B
(14) POP	register pair	pop the stack to the register pair	POP H

* PUSH : the contents of reg pair are copied into stack, stack pointer is decremented and the contents of higher order register (B D H A) are copied into stack. Stack pointer is again decremented and contents of low order register (C, E, L, flags) are copied into stack.

* pop: the contents of the top of the stack are copied into register pairs, the contents of the location pointed out by stack pointer are copied to low order register (C, E, L, flags). stack pointer is incremented and contents of location are copied to the high order register pair.

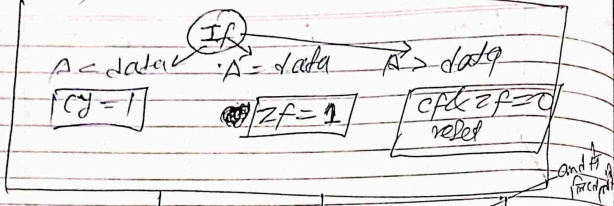
(2) Arithmetic Instruction:

Instruction	Operand	Description	Example
① ADD	R M	Add with carry	ADD B
② ADC	R and M	Add register or memory to Ac with carry	ADC B ADC M A ← A + B + cy
③ ADI	8 bit data	Add immediate to Accumulator	ADI 45H A ← A + 45H
④ DAD	Reg pair	Add the specified register pair to the HL pair	DAD B H-L ← H-L + B-C
⑤ SUI	8 bit data	subtract immed immediate fr Accumulator.	SUI 45H A ← A − 45H
⑥ SBB	R M	subtract with borrow	SBB B A ← A − B − CY
⑦ INR	R M	increment the reg or memory by 1.	INR B INR M
⑧ INX	R	the contents of reg pair are Incremented by one.	INX..
⑨ DEC		decrement the reg and memory by 1	
DCX	decrement	decremant the reg pair by 1	DEX H

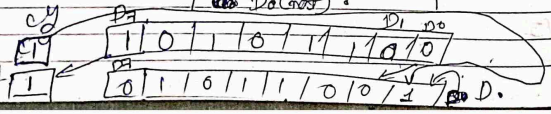
(3) logical instructions:

			Example
CMP	R M	Compare the contents of reg or memory with the contents of Ac	CMP A, B
CPI (allows 8085)	8 bit data	Compare immediate with the accumulator	CPI 8 bit data.
ANA	R M	Logically and the contents of reg or mem with contents of Ac	
ANI	8 bit data	logically and the 8 bit data with the Accumulator	

Img ⇒ 8.1



ORA	R M	
ORI	8 bit data	
XRA	R M	Logically XOR the content of reg or mem with the content of Ac and stored in Acc
RAL	Null	Rotate accumulator left to carry each bit of the Ac is rotated left by 1 position through carry flag, bit d7 is placed in the carry flag and the carry flag is placed in the least significant position (D0 Count).



RAR	None	Rotate the content of Accu right through carry

each binary bit of Ac is rotated right by 1 position through carry flag.

bit D0 is placed in the cy flag and the cy flag is placed in the most significant position with D7.

RLC	None	Rotate Ac left

each binary bit of Ac is rotated left by one position, bit d7 is placed in the position of D0

RRC	None	Rotate Ac right

bit D0 is placed in the position of D7

CMA	None	Complement the contents of Ac
CMC	None	Complement the content of carry flag
STC	None	set carry

④ Branching Instruction

unconditional branching
Jmp
① JMP
② Call
③ Return

conditional branching
Jmp
JIC
JNC
JZ
JNZ

JMP to the address given if the carry flag is set as one.

JIC : JMP to the A.

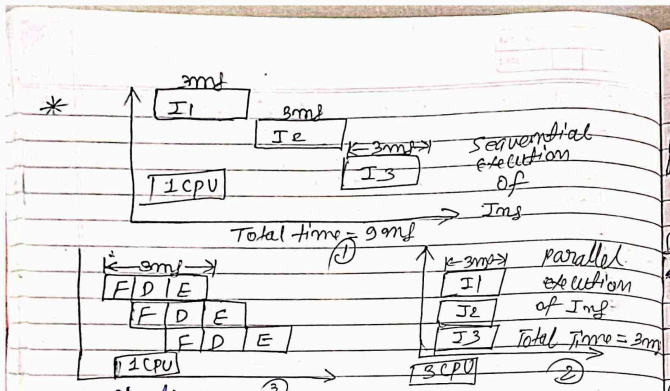
JNC : JP

JZ : JMP to address if carry flag is zero

⑤ Input output and machine control Instruction

EI
DE
SIM
RIM

NOP → reduce the delay factor
HLT →
IN 80
OUT 80



RISC & CISC

Q. diff b/w RISC & CISC

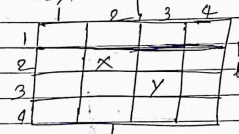
* RISC : RISC stands for Reduced Set Computer

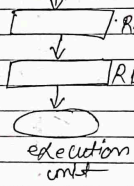
Q. diff b/w RISC & CISC

Q ↓ parameter	RISC	CISC
① Stands for	Reduced Instruction Set Computer	Complex Instruction Set Computer.
② Instruction size & format	Small set of instruction with fixed format	large set of instruction with variable formate
③ Inst. fetch time	Instruction fetch time same for all ins	It vary with respect to instruction formate size
④ Instruction set	Small & simple	large & complex
⑤ Data transfer	Register to register	Memory to memory.
⑥ Addressing Modes	Less no of Modes of most of the Ins are based on in register	More no of Modes of complex instructions are available with different varity.
⑦ Compiler design.	Less complex design	More complex design
⑧ Execution speed.	Fast execution speed	Less execution speed.
⑨ Pipelining	More efficient	Less efficient.
⑩ Program size	Program size long because weak code density.	program size small. Because better code density.
⑪ Cost	less cost	More cost

parameter	RISC	CISC
(2) Ins cycle count	single Ins cycle	several Ins cycle
(3) Control unit	It is Hard wired control unit base	It is micro program control unit base
(4) Example	SPARC FUGAKU 2020 (supercomputer)	MOTOROLA 6800 Intel 8060

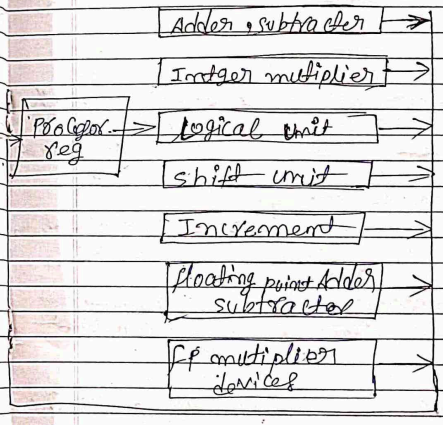
```
LOAD A, 2:2          MULT 2:2, 3:3
LOADB, 3:3           MULT x, y
PROD A, B            x = x * y
STORE 2:2, A
STA
```





* Parallel Processing : If we execute instruction in sequential manner then it takes more time to execute the ins and performance degrades So to enhance the performance of instruction we use parallel processing.

* major advantage of parallel processing.
  (i) Performance increases
  (ii) Time decrease
  (iii) high throughput.



* classify MT flynn's classification

PP

SISD SIMD MISD MIMD

S → single
M → multiple
I → Ins stream
D → data stream

where
Is = memory
DS = processor

- Flynn's classification:
is the most popular taxonomy of the computer architecture proposed by micle J Flynn in 1966 based on no of Ins and data executed by processing unit.
- Instruction stream: Instruction stream is defined as it is sequence of ins.

Data stream : It is defined as sequence of data including input or temporary results.

Q diff among sequential, parallel and pipeline

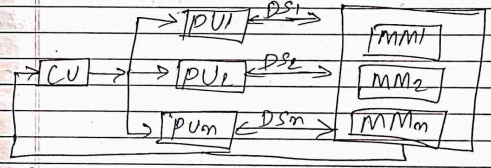
① SISD : It is a uniprocessor machine which is capable of executing single instruction operating on the single data stream. Instructions are executed sequentially but may be overlapped in their execution stages.

Ex → 

Ex
single CPU workstation
minicomputers

② SIMD: It is capable of executing same instruction on same CPU but operating on different data stream. ~~It has single control unit which issue one ins at one time.~~ Single Instruction: All the processing units execute the same inst at the given clock cycle.

Multiple data stream: Each processing units can operate on different data stream.



Ex → Array processor
Vector pipeline
SIMD computers has single control unit which issue one ins at a time but it has multiple ALU and/or processing unit to carry out on multiple data set simultaneously

(4) MISD : Multiple ins operates on single data stream, it is capable of executing different instruction on different PUs but all of them are operating same data set, machine built using MISD model are practically not useful a few machine are built but none of them are commercially available.

Ex→ Systolic Array.

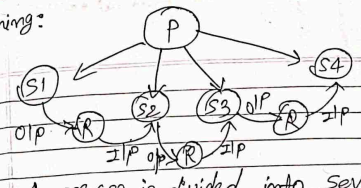
MIMD: This system is capable of executing multiple instructions on the multiple data set,

Mutiple stream : Every processor may be executing a different ins stream.

Multiple data : Every processor will be working with different data set.

Ex→ IBM 370

Pipelining:



A process is divided into several sub operations where each sub operation is associated with segment O/P of each segment will be stored in R Register. is passed as an input to next segment.

Here all the segments work independently so that we can execute all segments concurrently.

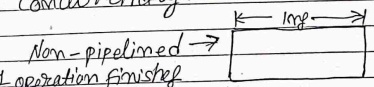


fig. Breakup the computational task into segments

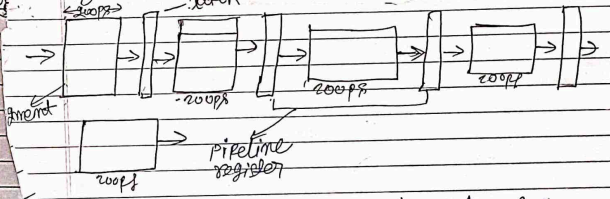


fig → Separate each segment with pipeline Reg.

Key idea : overlap the exe. of multiple ins so that multiple ins perform the tasks simultaneously.

Why it is called pipeline : Here we are transfering the information from one segment to other segment.

Pipeline register   in 200 PS →

(Q8) How pipelining improves performance of a microprocessor.

* Why the concept of pipeline arises

CPU follows the sequential order of execution so instruction execution slows down that is It is having ~~my~~ uniprocessor system. It has a serial execution, to increase the performance or speed we use the pipelining (a way of speeding up the exe of the instruction).

Key idea : overlap the exe of multiple ins so that multiple ins perform the task simultaneously.

* Why it is called pipeline : Here we are transfering the information from one segment to other segment.

Pipelining: It is a technique of decomposing a sequential process into suboperations with each suboperation being executed in a special dedicated segment that operates concurrently with all the segments in the pipeline in an overlapping manner. Each segment performs the partial processing dictated by the way task is partitioned. The result obtained from each segment is transfer to the next segment.

Assume no of cycles

$A_i * B_i + C_i$

for i=1 to 3 → 3rd suboperation

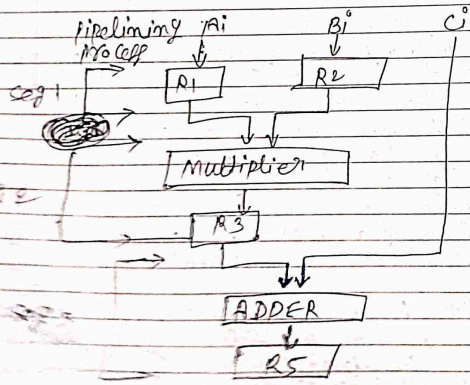
1st Sub operations

*

Segment 1 $R_1 \leftarrow A_i, R_2 \leftarrow B_i$

Segment 2 $R_3 \leftarrow R_1 * R_2, R_4 \leftarrow C_i$

Segment 3 $R_5 \leftarrow R_3 + R_4$



i=1 to 7

Instruction Pipeline

clock cycle	R1	R2	R3	R4	R5
i=1	A1	B1	—	—	—
i=2	A2	B2	A1*B1	C1	—
i=3	A3	B3	A2*B2	C2	A1*B1+C1 →
i=4	A4	B4	A3*B3	C3	A2*B2+C2 →
i=5	A5	B5	A4*B4	C4	A3*B3+C3 →
i=6	A6	B6	A5*B5	C5	A4*B4+C4 →
i=7	A7	B7	A6*B6	C6	A5*B5+C5 →
i=8	—	—	A7*B7	C7	A6*B6+C6 →
i=9	—	—	—	—	A7*B7+C7 →

* Instruction pipeline: An Instruction pipeline is a technique used in designing a computer to increase their instruction throughput (No of Inst executed per second). In Inst pipeline a stream of execution can be executed in an overlaping manner. Inst pipelining is used achieve inst level parallelism within single processor. each inst in a computer is proceed with following sequence of step.

(i) Fetch the inst from the memory.
(ii) Decode the instruction.
(iii) fetch the operand from memory.
(iv) execute the instruction.
(v) store the result at the suitable place.

In inst pipeline these steps of many inst can be carried out in overloaded manner to reduced the overall execution time.

IP

4 Stages	5 Stages	6 Stages
Inst pipeline	Inst pipeline	Inst pipeline
↓	↓	IF
IF	IF	ID
ID	ID	CO (Cal operand data)
EX	FO (Fetch operand)	FO
WB (write back)	EX	EX
	WB	WB

Explain all points:

1 clock = 1 μs

Q Consider a system where a clock is triggering at the speed of 1 MHz, ~~of the~~ ~~system which clock is~~ in pipeline processor there are 4 stages and each stage takes only 1 clock cycle if a program has 10 instructions then it will take how much time to calculate

(i) Time without pipeline
(ii) Time with pipeline
(iii) why CPI = 1
(iv) Speed up and maximum speed up
(v) efficiency
(vi) derivation of ~~cross~~ to calculate maximum speed up.

Ans → Given → Speed = 1 MHz (10^6 times clock frequency (triggering in 1s))

$S = 10^6$ Hz ∴ Stage = 4

$F = 1 \times 10^6$ Hz $f = \frac{1}{T}$ $T = 10^{-6}$

T = 1 μs

Time ~~with~~ without pipeline.

(i) IF I_1 I_2
ID I_1 I_2
EX I_1 I_2
WB I_1 I_2
4 4

I_3 I_4 I_5
I_3 I_4 I_5
I_3 I_4 I_5
I_3 I_4 I_5
4 4 4 4

I_6 I_7 I_8
I_6 I_7 I_8
I_6 I_7 I_8
I_6 I_7 I_8
4 4 4

I_9 I_{10}
I_9 I_{10}
I_9 I_{10}
4 4

1 IWP = 48cc (Time without pipline)

10 TWP = 4 × 10 = 40 clock cycle → 40 μs

TWP = (sum of clocks for each phase of an ins) * (no of Ins.) * (time of 1 clock cycle)

(2) Time with pipelining:

IF	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
ID	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
EX	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
WB	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}

4cc 1cc 1cc 1cc 1cc

(Time with pipeline) = (no of phases) + ((no of Ins) − 1) * (time of 1 clock cycle)

TP = [m + (n − 1)] * 1 μs

TP = [4 + (10 − 1)] × 1 μs

TP = 13 μs

(c) why CPI = 1 : Here 1st inst took 4 cc then in each second 1 inst gets executed because of the overlaping condition so here CPI (clock per Inst) for 1st inst is 4 cc and for the rest of the inst is 1 cc so far CPI is equal to 1.

(iv) Speed up and maximum speedup

$$\text{Speed up} = \frac{\text{Time without pipeline}}{\text{Time with pipeline}} = \frac{40\,\mu s}{13\,\mu s} = 3.01\,\mu s$$

maximum speed up = no of stages
= 4 cc (4 times)

(v) effeciency : $\frac{\text{Speed up}}{\text{max speedup}} \times 100$

$$\text{eff} = \frac{3.01\,\mu s}{4} \times 100 = \frac{301}{4} = 76.92\%$$

when $n = 100 = 97 \rightarrow$ TWP = 400 ns
and $n = 10000 = 100$ (APP) $\rightarrow$ TP = 103 ns

$$\text{Speedup} = \frac{400}{10} = 3.88$$

max speedup = no of stage = 4

$$\text{eff} = \frac{3.88}{4} \times 100 = \frac{388}{4} = 97\%$$

like that
n = 19000

(vi)

Speedup → m = stages, $m << n$
n = Inst count

$$\text{Speedup} = \frac{TWP}{TP} \Rightarrow \frac{m \times n}{m + (n-1)} = \frac{m \times n}{n} = m$$

$$= \frac{10,00,000}{10,00,003}$$

$$= \text{speedup} = m \cong n$$

* WAP to evaluate Arithmetic expression $X = (A+B) * (C+D)$ using 3 byte, 2 byte, 1 byte and 0 byte instruction.

(i) using 3 byte instruction

ADD	R1, A, B	// R1 ← M[A] + M[B]
ADD	R2, C, D	// R2 ← M[C] + M[D]
MUL	X, R1, R2	// R3 ← M[X] = R1 * R2

K = (A+B) * (C+D)
R1 R2

Advantage → • Short length.
• code is written in reduce length of line

Disadvantage → inst length is increased.
Complex to handle the inst

using 2 byte instruction

```
MOV R1, A   // R1 ← M[A]
ADD R1, B   // R1 ← R1 + M[B]
MOV R2, C   // R2 ← M[C]
ADD R2, D   // R2 ← R2 + M[D]
MUL R1, R2  // R1 ← R1 * R2
MOV X, R1   // M[X] ← R1
```

(iii) using 1 byte instruction

in load ins

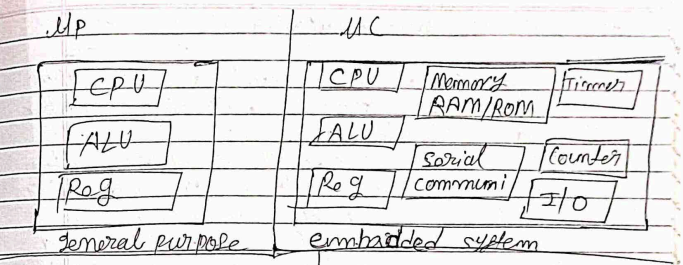
```
LOAD A   // Ac ← M[A]   Ac is the destin
ADD B    // Ac ← Ac + M[B]   and operand is
STORE T  // M[T] ← Ac   the source in
LOAD C   // Ac ← M[C]   store ins AC
ADD D    // Ac ← Ac + M[D]   operand is the
MUL T    // Ac ← Ac * M[T]   destination
STORE X  // M[X] ← Ac
```

(iv) using 0 byte instruction

$X = (A+B) * (C+D)$

```
PUSH A   // TOS ← A
PUSH B   // TOS ← B
ADD      // TOS ← (A+B)
PUSH C   // TOS ← C
PUSH D   // TOS ← D
ADD      // TOS ← (C+D)
MUL      // TOS ← (C+D) * (A+B)
POP X    // M[X] ← TOS
```

Diagram



PAGE NO.
DATE

(Eg) Consider a pipeline having 4 phases with duration 60, 50, 90 and 80 ns. Given latch delay is 10 ns. Calculate

(i) Pipeline cycle time
(ii) Non-pipeline execution time
(iii) Speed up ratio
(iv) Pipeline time for 1000 tasks
(v) Sequential time for 1000 tasks
(vi) Throughput

Given: four stage pipeline
Delay of Stages = 60, 50, 90 and 80 ns
Latch delay = 10 ns

(i) Pipeline cycle Time:
cycle time = Maximum delay due to any stage + Delay due to its register
= Max (60 + 50 + 90 + 80) + 10 ns
⇒ 90 + 10 ns = 100 ns

(ii) Non-Pipeline Execution Time:
Non-Pipeline execution time for one inst^{n} =
Sum of delay of stages
⇒ 60 ns + 50 ns + 90 ns + 80 ns = 280 ns

(iii) Speed up Ratio:
Speed up = time without pipeline / Time with pipeline
Speedup = 280 ns / 100 ns = 2.8

(iv) Pipeline Time for 1000 tasks:

Pipeline time for 1000 tasks = Time taken for 1st task + Time taken for remaining 999 tasks

⇒ [redacted]..

⇒ 1 × 4 cycle time + 999 × Clock cycle

⇒ 4 × 100ns + 999 × cycle time

⇒ 4 × 100ns + 999 × 100ns

⇒ 400ns + 99900 ns

⇒ 100300ns

(v) Sequential Time for 1000 tasks:

non-pipeline time for 1000 tasks

⇒ no of tasks/instruction * time taken for one task

⇒ 1000 × 280ns ⇒ 280000ns

(vi) Throughput:

Throughput for pipelined execution =
Number of instructions executed per unit time

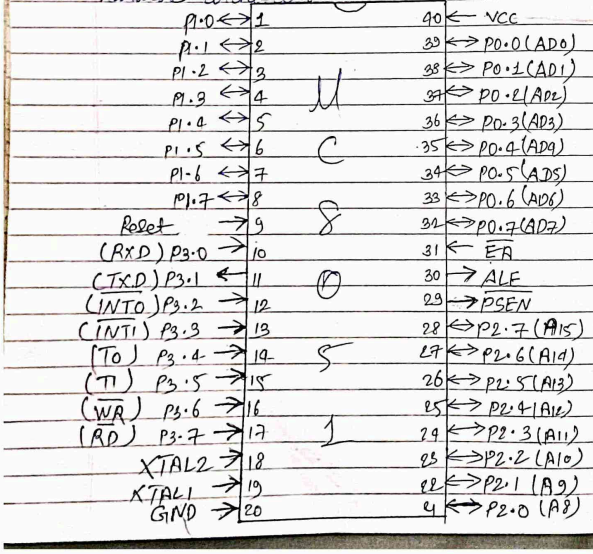
⇒ 1000 tasks / 100300ns

⇒ no of instruction / pipeline time for 1000 tasks

⇒ 1000 tasks / 100300 ns ⇒ **1000 tasks / 100300 ns**

(Q5) Draw the pin diagram of 8051 microcontroller and explain the functionality of each pin.

ans → 8051 microcontroller used in embedded systems like remote control, washing machine, timer, counter etc. 8051 is a complete computer system built on single chip. It has CPU, RAM, ROM serial and parallel ports, interrupts etc. on the single chip that is a complete computer for small application. 8051 operates on 12MHz of clock frequency. It has 8 bits of ALU. It is comprised of 8 bits of dataline. It follows harward architecture.



- pin 1 to 8 (Port 1): Pin 1 to pin 8 are assigned to port 1 for simple I/O operations. They can be configured as input or output pins depending on the logic control, if logic zero (0) is applied to it act as an output pin and if logic one (1) it acts as an input pin. These ping are bidirectional.
- pin 9 (Reset): Reset pin, it is an active high signal it is used to reset the microcontroller to its initial values when power is on.
- pin 10 to 17 (port 3): pin 10 to pin 17 are port 3 ping which are also referred to as P3.0 to P3.7 [redacted].

  RXD: 10th pin is RXD (Serial Data Receive) which is for serial input. Through this input signal micontroller receives data for serial communication.
- TXD: 11th pin is TXD (Serial data transmit) which is serial output pin. Through this [redacted] UC transmits data for serial communication.
- INT0, INT1: 12th and 13th ping are INT0, INT1 for external hardware interrupt 0 and 1 both are active low signal and INT0 priority is high.
- T0 and T1: 14th and 15th pin are for Timer 0 and Timer 1 external input.
- (WR): 16th pin is for external memory write this is active low signal.
- RD: 17th pin is for external memory read this is also active low signal.

* Pin 18 to pin 19 (XTAL2 and XTAL1): These ping are connected to an external oscillator which is generally a crystal oscillator. They are used to provide an external clock frequency of 1 MHz to 30 MHz

* pin 20 (GND): This pin is connected to the ground It has 0V power supply.

* pin 21 to 28 (port 2): pin 21 to 28 are port 2. These ping are used for interfacing external devices to get system clock.

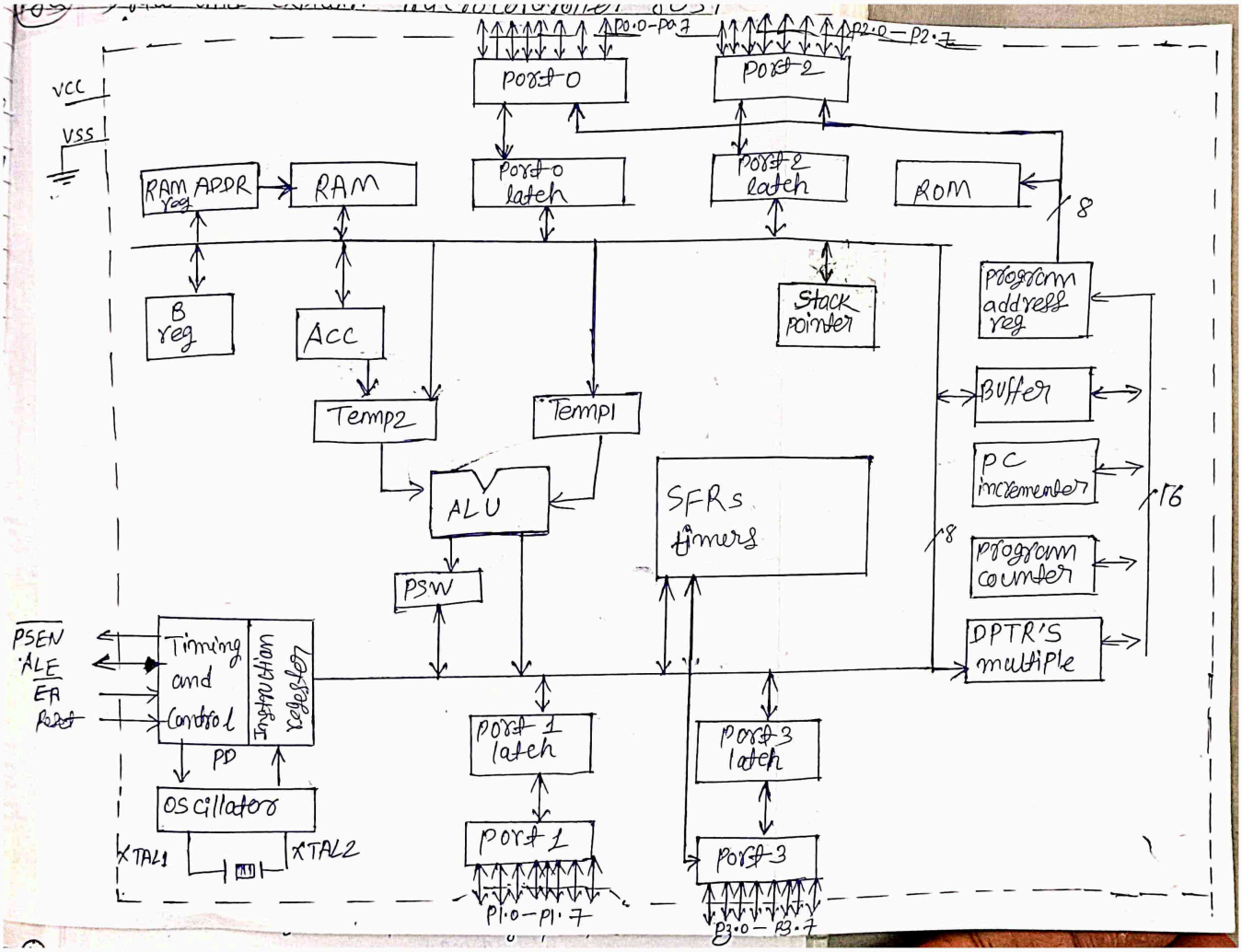
* pin 29 ($\overline{PSEN}$): program store Enable. It is an active low-signal pin. this is used to read external memory.

* Pin 30 (ALE): Address latch Enable. It is an active high signal pin. this pin is used to differentiate between address bus and data bus.

* pin 31 ($\overline{EA}$): External Access input it is active low signal pin. It is used to enable/disable external memory interfacing.

* Pin 32 to pin 39 (port 0): Pin 32 to Pin 39 are port 0 ping. They are bidirectional input/output ping port 0 is also designated as AD0 - AD7 because 8051 multiplexes address and data through port 0 to save ping.

* Pin 40 (Vcc): This pin provides power supply voltage +5 Volt to the circuit.


P0.0-P0.7
P2.0-P2.7
Port 0
Port 2
VCC
VSS
RAM ADDR Reg
RAM
Port 0 latch
Port 2 latch
ROM
8
B reg
Acc
Stack pointer
Program address reg
Buffer
Temp2
Temp1
PC incrementer
16
ALU
SFRs timers
8
Program counter
PSW
DPTR'S multiple
PSEN
ALE
EA
RST
Timing and Control
Instruction register
PD
Port 1 latch
Port 3 latch
Oscillator
XTAL1
XTAL2
Port 1
Port 3
P1.0-P1.7
P3.0-P3.7

(Q5) Draw and explain microcontroller - 8051 Architecture.

8051 microcontroller used in embadded system like remote control, washing machine, timer counter etc. 8051 is a complete computer system built on single chip. It has CPU, RAM, ROM serial and parallel ports, interrupts etc. on the single chip that is a complete computer for small application. 8051 operates on 12MHz of clock frequency. It has 8 bits of ALU. It is comprises of 8 bits of dataline. It follows harvard architecture. Mc supports a range of instn, arithmatic logical data transfer, branching and lopping instn.

The architecture of 8051 mc is divided into 5 sub divisions:

(i) CPU (Ac, PC, SP, GPR, DPTR)
(ii) I/O ports
(iii) Data memory (RAM)
(iv) program memory (ROM)
(v) Timing and control unit

(1) CPU: It is a 8 bit processor that can execute the instruction at the clock frequency of 12MHz. It consists of several register that are used to store the data & to control the operations of Mc. registers

(i) Accumulator: Acc is used to store the result of arithmetic & logical operations performed by CPU.

(ii) Program counter: It keeps track of address of the next instruction to be executed.

(iii) Stack pointer: It is a register that points to the top of the stack.

(iv) General purpose reg: General purpose registers are used to store the data during program execution.

(v) DPTR: It is 16 bit data pointer. It consists of 2 separate reg DPH & DPL (Data pointer high) and (Data pointer low). DPTR usually used for storing the data & immediate value.

(2) Data memory (RAM): RAM is a volatile memory used to store temporary data, it can either be internal RAM or external RAM

RAM
- internal (128 bytes)
- external (64 KB)

The general purpose, ~~register~~ memory is divided into 2 section

General purpose register	Specific purpose register
GPR: store the data during prog. execution.	Control the operation of uc & interface external devices.

(3) Program memory (ROM): ROM is ~~a volatile~~ a non-volatile memory, all programs and instructions are stored in ROM its size varies from 2 KB to 64 KB (but we usually use 4 KB).

(4) I/o ports: It is used to interface with the outside world it provides physical connection. I/o port provides gateway for passing the data in the outside, world. O/p allows ~~to~~ to control ~~to~~ the peripheral devices like monitor, printer.

in I/o port each port can be consider as either I/p or o/p.
port 0 → IO + AD0-AD7 (lower addr (datalines for mem)
port 2 → IO + A8 - A15 (Higher addr (datalines for mem)
port 3 → IO + Special fun^n (Interrupt, serial comm^n, timer
port 1 → IO.

(5) Timer and control unit: The main fun^n of timer is to make a delay. there are 2 timers T0 and T1 each of 16 bit both can generate two delay concurrently to produce the suitable delay based on the requirement of processor. It transmits the signal to the processor whenever the particular delay gets generated.

The 8051 micro controller also has several other I/o ping including Reset pin, XTAL1 & XTAL2 pins, and the ALE pin. These ping are used for various purposes, such as system reset, clock input and address bus control.

(Ex-8) Write a program to add 2 16 bit numbers

Mnemonics	Memory Address	Machine code	Description / Comments
LHLD	8000	2A	Load HL pair Direct
	8001	50	Address to store first
	8002	80	16 bit operand.
XCHG	8003	EB	Exchange H-L pair with DE
LHLD	8004	2A	Load HL pair Direct
	8005	52	Memory Address to store
	8006	80	end 16 bit No.
MVI,C	8007	0E	for carry generate
	8008	00	initialize register c with 0
DAD,D	8009	19	Double Addition of HL & DE reg pair
INC	800A	D2	if no carry, jump Ahead
	800B	0E	LSB
	800C	80	MSB
INRC	800D	0C	Increment Carry by 1
SHLD	800E	22	Store H-L pair Direct
	800F	54	Memory address to store the
	8010	80	result at memory.
MOVA,C	8011	79	Move the carry to Accumulator
STA	8012	32	Store the content of ACC.
	8013	56	Memory Address to store
	8014	80	Carry of Acc.
HLT	8015	76	End of the program

Input 1, at location → 8050 → [redacted]45 → 8051 → A6
→ A 6 4 5
1010, 0110 0100 0101

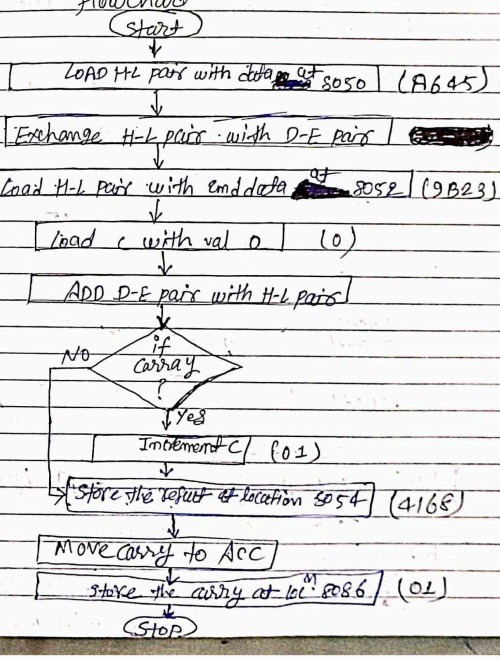
Input 2, at location → 8052 → [redacted]23 → 8053 → 9B
→ 9 B 2 3
1001 1011 0010 0011

Carry at location: 8086 → 01

Addition → 1010 0110 0100 0101
1001 1011 0010 0011
(carry → 1) 1000
4 1 6 3

Sum at location → 8054 → [redacted]68 → 8055 → 41.
carry, at location → 8056 → 01

Flowchart

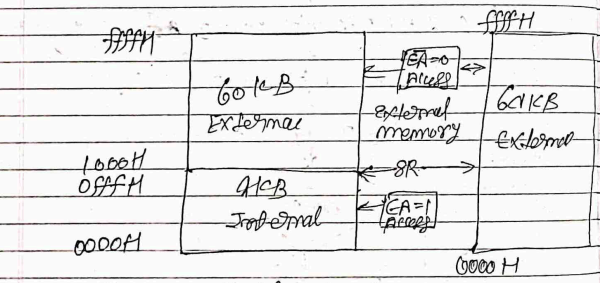


Explain -

→ Inp 1 = A695 Input 2 = 9B23
→ load A695 to LHLD reg. pair
→ XCHG exchange HL pair data into DE
→ Again load HL pair with 9B23 data
→ Move 0 to reg C
→ double addition of HL and D-E pair value
→ if carry increment C by 1
→ if No carry store result at memory
→ Move carry to ACC
→ store the Ac at mem location.
→ HLT the program.

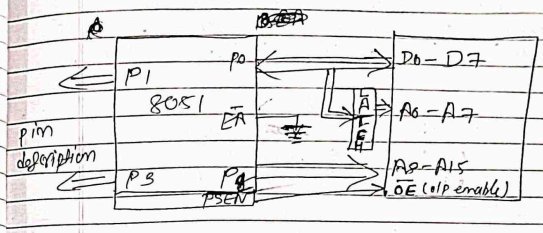
* External Memory Interface

(i) External program memory :

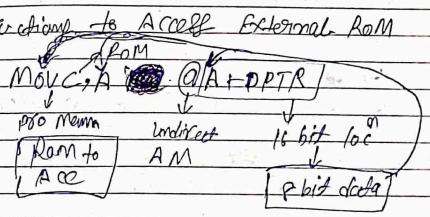


EA → Exec Enable Access
if EA=1 (1=Vcc) → EA=1 → EA=0
Access Internal memory → 0000H - 0FFFH

EA → GND if EA=0 (GND) EA=0 → EA=1
Access external Mem



* Instructions to Access External ROM



Numericals on ROM & RAM

* How many 256 MB RAM chips are required to build 4 GB of RAM.

Note ⇒

1 byte	$= 2^3$ bits
1 KB	$= 2^{10}$ bytes
1 MB	$= 2^{20}$ bytes
1 GB	$= 2^{30}$ bytes

No of chips required = total given memory size / size of available chip

Size of decoder = input × output

No of chips = $\frac{4\text{ GB}}{256\text{ MB}} = \frac{2^2 \times 2^{30}}{2^8 \times 2^{20}} = \frac{2^{35}}{2^{31}} = 2^4 = 16$

No of decoder = $4 \times 2^4 = 4 \times 16$

* How many 128 KB × 8 RAM chips are required to build 1 MB of RAM

No of chips = $\frac{1\text{ MB}}{128\text{ KB}} = \frac{1 \times 2^{20} \times 2^3}{2^7 \times 2^{10}} = \frac{2^{20}}{2^{17}} = 2^3 = 8$ chips

No of decoder = $2^3 \times 3 = 3 \times 2^3 = 3 \times 8$

Q. How many 64 KB × 8 RAM chips are required to build 1 MB of ram also calculate size of the decoder.

Ans

No of chips = $\frac{1\text{ MB}}{64\text{ KB} \times 8} = \frac{1 \times 2^{20} \times 2^3}{2^6 \times 2^{16}} = \frac{2^{20}}{2^{16}} = 2^4 = 16$

No of decoder = $4 \times 2^4 = 4 \times 16$

Q. How many 256 MB × 8 RAM chips are required to build 32 GB of RAM also Calculate the size of decoder.

Sol:

No of chips = $\frac{32\text{ GB}}{256\text{ MB} \times 8} = \frac{2^5 \times 2^{30}}{2^8 \times 2^{20} \times 2^3} = \frac{2^{35}}{2^{31}} = 2^4 = 16$

No of decoder = $4 \times 2^4 = 4 \times 16$

Alternate:

No of chips = $\frac{32\text{ GB}}{256\text{ MB} \times 8} = \frac{32 \times 2^{30}}{2^8 \times 2^{20} \times 2^3} = \frac{32 \times 2^{30}}{2^{31}} = \frac{2^5 \times 2^{30}}{2^{31}} = 2^4 = 16$

No of decoder = $4 \times 2^4 = 4 \times 16$

2 Marks

* Explain the role of control unit in memory.

ans→ The control unit fetches instruction from the CPU's memory, decoding and executing of these instructions. It serves as the brain of the memory

(i) Fetching inst^{n}: The control unit retrieves instructions from memory one at a Time.

(ii) Decoding inst^{n}: The control unit then decodes the instructions, figuring out what operation needs to be performed like addition, subtraction etc.

(iii) Sending control signals: Based on the decoded instruction, the control unit sends electronic signals to ALU or memory unit. These signals tell those components what actions to take.

(Pyq) difference between software and hardware interrupts.

Hardware interrupts	Software interrupts
(i) An interrupt that is generated from an external device	A type of interrupt that is generated by an instruct^{n} in the program.
(ii) Asynchronized events	Synchronized events
(iii) Do not increment the program counter	Increment the program counter.
(iv) Do not get a higher priority	Get a higher priority
(v) Generated by external device	Generated by executing instruction.

(Pyq) Mention what are the basic components of a microprocessor.

ans→ Some common components of a microprocessor

(i) Control unit (ii) I/O units

(iii) Arithmetic logic unit ALU

(iv) Registers (v) cache Memory

(ii) I/O units: These are the external peripherals devices that is used by microprocessor to read and write operations like keyboard, mouse, printer.

(iii) ALU: ALU performs all the mathematical and logical operations such as addition, subtraction comparison.

(iv) Registers: These are temporary storage locations inside the microprocessors that is much faster than secondary memory. It hold data and instructions that are currently used by ALU and CU.

(v) Cache memory: This is a small but super-fast type of memory that placed b/w CPU and main memory. The cache stores frequently used data and instructions so the processor can access them quickly without going to the main memory all the time.

(Pyq) Determine the number of clock cycles that it takes to process 200 tasks in a 6 segment pipeline

ans→ given: tasks = 200 (No of clock cycles = ?)

No of Segment = 6

No of clock cycles = $K + (n-1)$

where K is the first inst^{n} that will take k segment clock cycle.

where (n-1) times that will take only 1 clock cycle

No of clock cycle = k + (n-1)
= 6 + (200 - 1)
= 6 + 199
= 205 clock cycle

(Pg8) Difference between Direct and indirect Addressing Modes:

Parameter	Direct Addressing Mode	Indirect Addressing Mode
Address Field	Address field contains the effict effective address.	Address field containg reference of effective addrg
memory reference	Address field requires only one memory reference	Requires two memory reference
Processing speed	It is faster than indirect addressing mode	It is slower than direct addressing mode
calculation	No further calculations is required to perform the operation	Requires further calculations to find the effective address.
Address space	It occupies a smaller amount of space then the indirect mode	It occupies a large amount of space than the direct mode
Overhead	No additional overhead diagram → page 6.	Additional overhad to search for operand. diagram page 6

(Pg9) Elaborate LDA Ac and Store Ac instruction with syntax.

(i) LDA (Load Accumulator) Ac Instruction:
Syntax: LDA Ac
Description: This instruction loads the value stored in memory into the accumulator
Example: if mem loct 1000 contains the value 42 then LDA 1000 would load 42 into the accumulator reg.

(ii) Store Ac instruction:
Syntax: STA Ac
Description: This instruction stores the value currently in the Accumulator into specified memory location.
Example: If the accumulator reg containg the value 42 then STA 2000 would store 42 in to mem loct in 2000.

(Pg9) Difference between register and memory

Parameter	register	memory
Definition	Hold operands or instruction being processed	Holds instruction and data for programs
size	smaller It holds the small amount of data around 32 bits to 64 bits	It stores the large amount of data around some GB to TB.
Speed	It is faster than memory	Slower compared to registers.
Example	AC (Accumulator	RAM (Random accell memory
Location	Located inside the CPU	Located outside the CPU.

(P4) what do you mean by cache coherence

ans ⇒ Cache coherence refers to the consistency of data stored in multiple caches across a computer system. cache coherence ensures that all caches have a consistent view of memory when one processor writes data to a memory that is also cached by another processor. cache coherence mechanisms ensure that the other processor's cache is updated or invalidated to reflect the new data. The prevents inconsistency or conflicts that could arise if diff processors have different version of data.

(P4) what do you understand by programmed I/O?

Programmed I/O (input/output) is a method of data transfer between a computer and external devices where the CPU manages each data transfer operation directly.

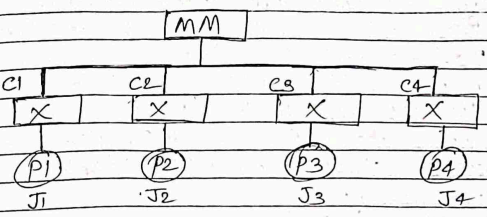
(P5) what is interrupt?

An interrupt is a signal sent by a hardware device to the processor indicating that it requires attention. when an interrupt occurs the processor temporarily stops execution of its current task and jump to execute a specific interrupt handler routine, which handles the interrupt request

(P5) How auxiliary memory is different from associative memory?

ans ⇒ Auxiliary memory typically refers to secondary storage devices like hard drives

* Cache coherence



If multiple processes are trying to access the same data, if one process updates the data in its cache but not in main memory and other processes still have unupdated data in its cache. This leads inconsistency and error in the program output. Cache coherence solves this problem by ensuring all processes have same and updated version of the data it leads to consistency and accuracy of the program output.

(P6) what do you mean by cycle stealing mode.

ans ⇒ Cycle stealing mode is a method of accessing computer memory (RAM) or bus without interfering with the CPU. It is similar to direct memory access (DMA) for allowing I/O controllers to read or write RAM without CPU involvement.

or SSDs used to store data. when the computer is turned off.

Associative memory is a type of computer memory that enables rapid searching and retrieval of information based on content rather than address. It's often used in specialized applications where quick access to large amounts of data is essential like database systems.

(pyq) Define Embedded systems
An Embedded system is a specialized computing system that performs dedicated functions or tasks within a large system. They are typically characterized by their real-time operation, low power consumption and small size. Embedded systems are often found in everyday devices such as household appliances, automotive systems, medical devices. They usually consist of a micro-controller or microprocessor and various peripherals devices.
Virtual memory do by os Note

(pyq) what do you mean by cache coherence
ans⇒

4 Marks

(Pyq) Difference between Hardwired control unit and microprogrammed control unit

Parameter	Hardwired control unit	microprogrammed control unit
Control Signal Generation	It ~~is generated by~~ generates control signal using logic circuits	It generates control signal using micro instructions.
Speed	It is faster than microprogrammed control unit	It is slower than Hardwired control unit.
Modification	Difficult to modify	Easy to modify
Cost	More expensive	less expensive
Complexity	It can't handle complex instructions	It can handle complex instructions
used in	RISC	CISC

(Pyq) Explain status bit register conditions in detail

Status register, also known as the flag register that holds various status bit that reflect the result of arithmetic and logical operations performed by the processor. Each bit in the status register indicates a specific condition or flag.

(1) Sign (S) Bit (bit 7): It reflects the sign of the result after operation. if the most significant bit of the result is set the sign bit is set to 1, if MSB is clear the sign bit is set to 0. Example: After operation: if the result 10101010 the sign bit will be 1 because the MSB is 1

(2) zero (Z) Bit (bit 6): It indicates whether the result of an operation is zero or not. if the result is zero, the zero bit is set to 1, if the result is non-zero, the zero bit is set to 0. Example: After an operation: if the result is 00000000 the zero bit will be 1

(3) Auxilliary carry (Ac) Bit (4): It reflects the carry-out from 3 to bit 4 during arithmetic operation. Example: During an addition operation: if a carry-out occurs from bit 3 to bit 4, the Auxiliary carry bit is set to 1.

(4) parity (P) Bit (bit 2): It indicates whether the number of set bits in the result is even or odd if result contains an even number of set bits the parity bit is set to 1. If the result contains an odd number of set bits, the parity bit is set to 0.

Example: After an operation, if the result has even parity (eg 10101010) the parity bit will be 1.

(Pyq) Write a short note on High-End-High performance processors.

ans ⇒ High-end: high-performance processors are advanced central processing units (CPUs) designed to deliver exceptional computing power, speed, and efficiency. These processors are typically used in high-performance computing like Supercomputers, workstations, gaming, content creation.

Here are some key features.

S, HCS, MA, AIS, LCS, HMB

(i) Multicore Architecture (ii) High clock speeds (iii) Large cache sizes (iv) Advanced instruction set Architecture (v) High memory bandwidth (vi) Advanced Manufacturing processors (vii) Specialized features (viii) Scalability

(Imp)

- Compare different types of instruction formats.

(i) zero address instructions
(ii) one address instructions
(iii) Two address instructions
(iv) Three address instructions

(i) Zero Address instructions: These instructions do not specify any operands or addresses. Instead, they operate on data stored in registers or memory locations.

Example: X = (A+B) * (C+D)

push	A	TOP = A
push	B	TOP = B
ADD		TOP = A+B
push	C	TOP = C
push	D	TOP = D
ADD		TOP = C+D
Mul		TOP = (C+D) * (A+B)
pop	X	M[X] = TOP

(ii) One Address instructions: These instructions specify one operand or address which typically refers to a memory location or register. The instruction operates on the contents of that operand and the result may be stored in the same or a different location.

Example: X = (A+B) * (C+D)

LOAD	A	AC = M[A]
ADD	B	AC = AC + M[B]
STORE	T	M[T] = AC
LOAD	C	AC = M[C]
ADD	D	AC = AC + M[D]
MUL	T	AC = AC * M[T]
STORE	X	M[X] = AC

(iii) Two Address instructions: These instructions specify two operands or addresses which may be memory locations or registers. The instruction operates on the contents of both operands and the result may be stored in the same or different locations.

Ex⇒ X = (A+B) * (C+D)

MOV	R1, A	R1 = M[A]
ADD	R1, B	R1 = R1 + M[B]
MOV	R2, C	R2 = M[C]
ADD	R2, D	R2 = R2 + M[D]
MUL	R1, R2	R1 = R1 * R2
MOV	X, R1	M[X] = R1

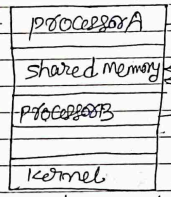
(iv) Three Address instructions: These instructions specify three operands or addresses which may be memory locations or registers. The instruction operates on the contents of all three operands and the result may be stored in the same or a different location.

Example : $X = (A+B) * (C+D)$

ADD	R1, A, B	R1 = M[A] + M[B]
ADD	R2, C, D	R2 = M[C] + M[D]
MUL	X, R1, R2	M[X] = R1 * R2

(Q10) Illustrate inter processor communication in a shared multiprocessor environment in computer architecture.

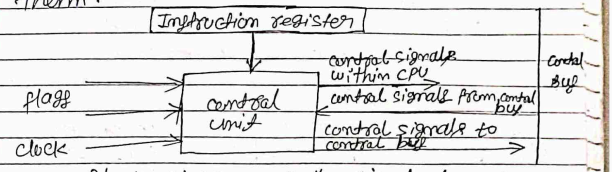
ans⇒ Inter processor communication refers to the exchange of data between two or more processors or cores in a computer system. It is important because it enables different processors to work together and share resource which can lead to better performance and more efficient use of system resources.



here two processor A and B shares a shared memory

(Q9) What is the need of control unit in computer? Draw the control unit of a basic computer. Discuss how fetch and decode phases are carried out.

ans⇒ The control unit in a computer is essential for coordinating and managing the various operations of the CPU. Its primary function include fetching instructions from memory, decoding those instructions and executing them.



Block diagram of the control unit

(i) Fetch phase : In this phase, the control unit retrieves the next instructions from memory using the program counter. The PC holds the memory address of the next instruction to be fetched. Once fetched, the instruction is stored in the instruction register.

(ii) Decode phase : After fetching an instruction, the control unit decodes it to determine what operation needs to be performed. This involves interpreting the instruction and identifying the operation code (opcode) that specifies the action to be carried out.

(iii) Execute phase : Once the instruction is decoded, the control unit directs the appropriate components of the CPU to execute the instruction. This might involve performing arithmetic or logical operations, accessing data from memory or transferring data between different components.

(Q30) Elaborate the various types of flag registers in 8085.

Ans ⇒ In 8085 microprocessor the flag register contains status flags that indicate various conditions after an arithmetic or logical operation. There are five flags in the flag register.

D7	D6	D5	D4	D3	D2	D1	D0
S	Z	–	AC	–	P	–	CY
sign flag	zero flag	auxiliary carry flag			parity flag		carry flag

(i) Sign flag (S): It indicates sign of the result if the MSB is 0 it indicates the result is positive and if MSB is 1 it indicates the result is negative

Input1 = 70 Input2 = 80

01110000 10000000

add input1 + input2 ⇒

```
  01110000
+ 10000000
  11110000
```

→ sign flag is = 1 (Negative)

(ii) zero(z) flag: It indicates whether the result is zero or not if result is 0 then zero flag is set to 1 otherwise it is reset (0)

Z = 0

(iii) Auxiliary carry (AC) flag: It indicates a carry from bit 3 to bit 4 during arithmetic operation

```
  0 1 1 1 | 0 0 0 0
  1 0 0 0 | 0 0 0 0
  1 1 1 1 | 0 0 0 0
```

→ carry = 0

A = 0

(iv) parity (P) flag: It indicates whether the no of 1s in the result is even or odd if the result has even no of 1s the parity flag is set to 1 otherwise it is set to 0.

Ex ⇒ result = 11110000

No of 1's = 4 (even) P = 1

(v) carry (CY) flag: It indicates a carry or borrow during arithmetic operation, if there is a carry of the MSB then flag is set to 1 otherwise set to 0.

Ex ⇒ result = 11110000

No carry CY = 0

(Q31) How parallel processing work? Discuss the various types of parallel processors?

Ans ⇒ Parallel processing works by dividing a task into smaller sub-tasks and executing them simultaneously across multiple processors. This allows for more efficient utilization of computational resources and faster completion of tasks. There are several types of parallel processors ① SIMD ② MISD ③ SISD ④ MIMD

Do from assignment.

(Q32) Explain Auxiliary memory and its devices

Ans ⇒ Auxiliary memory also known as secondary storage or external memory, refers to the devices that store data for long term even when the power is turned off, auxiliary memory retains data persistently.

Common auxiliary memory devices include

① Hard Disk Drives (HDD): This use rotating magnetic disks to store data. It offers high storage capacity at low cost.

② SSD (Solid State Drives): SSDs use flash memory to store data, providing faster access times and better durability compared to HDDs.

③ Magnetic Tape: It is used for large scale backups and archival storage, magnetic tape offers high capacity and low cost but slower access.

④ Optical Drives: Devices like CD, DVD and Blu-ray drives use optical discs to store data use for data backup and distribution.

(Q) Functions of timing unit in basic computer.

Clock management: It synchronize the operations of the CPU and other components with the system clock, ensuring that instructions are executed at the proper rate and that data transfer occur within specified time intervals.

Timing generation: It generates the timing signals required for coordinating the execution of instruction and synchronizing the operations of various components such as the CPU, memory and I/O devices

(Q) Non pipelined system takes 130ns to process an inst. A program of 1000 instructions is executed in non-pipelined system. Then same program is processed with processor with 5 segment pipelined with clock cycle of 30 ns/stage. Determine speed up ratio of pipline.

ans: Non pipeline system
130 ns for 1 inst^n
so for 1000 inst^n = 130 × 1000 = 130000 ns

since,

$$\text{Speed up ratio} = \frac{\text{Time without pipeline}}{\text{Time with pipeline}}$$

Time without pipeline = 130000 ns

so now,

$$\text{Time with pipeline} = \left(\left(\text{no of phases}\right) + \left(\text{no of inst} - 1\right)\right) * \left(\text{time of 1 clock cycle}\right)$$

∵ given that
phase = 5
1 clock cycle = 30 ns / stage

∴ Time with pipeline = (5 + (1000 − 1)) * 30ns
= (5 + 999) * 30ns
⇒ 1004 * 30ns
= 30120 ns

so, now

$$\text{Speed up ratio} = \frac{130000\ ns}{30120\ ns} = 4.316\ ns$$

(Pg9) Write a program to exchange the data at 5000H and 6000H locations in 8085 µp.

Memory Address	Mnemonics	Machine Code	Comments
8000	LXI H, 5000H	21, 00, 50	Load memory location 5000H into HL pair reg.
8001	MOV A, M	7E	Move the data at mem location 5000H into Accumulator
8002	MOV B, A	47	Move the data of Ac into reg B
8003	LXI H, 6000H	21, 00, 60 (MSB LSB)	Load memory loct 6000H into HL pair
8004	MOV A, M	7E	Move the data at mem loct 6000H into Accumulator
8005	MOV C, A	4F	Move the data of Ac into reg C
8006	LXI H, 5000H	21, 00, 50 (MSB LSB)	Load memory loct 5000H into HL pair
8007	MOV A, C	79	Move the data from reg C to AC
8008	MOV M, A	77	Move the data of Ac to mem location 5000H
8009	LXI H, 6000H	21 00, 60	Load mem loct 6000H into HL pair
800A	MOV A, B	78	Move the data from reg B to AC
800B	MOV M, A	77	Move the data of Ac to mem loc
800C	HLT	76	Stop the program execution

H → 5000H → 10
Je → 6000H → 20

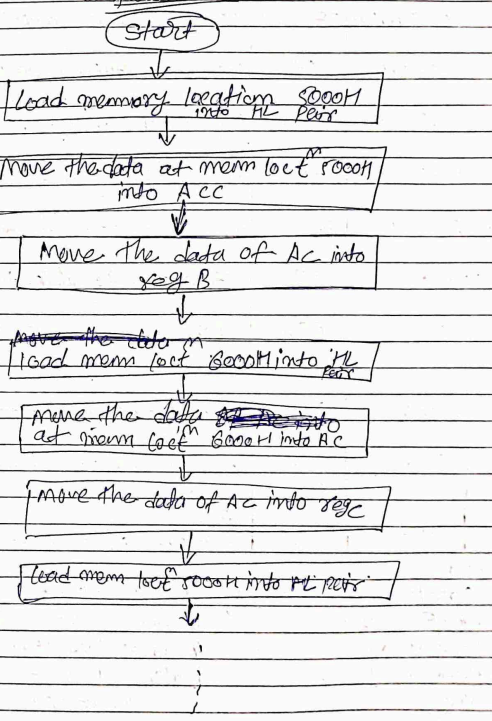
Before exchanging

Mem loc	5000 H
value	10
Mem loc	6000 H
value	20

After exchanging

Mem loc	5000 H
value	20
Mem loc	6000 H
value	10

Flowchart



12 MARKS

(Q7) write an assembly language program to swap two 8-bit numbers stored in an 8085 µp.

Memory Address	Mnemonics Code	Machine code	Comments
8000	MVI A, 5000H	3E, 25H	Load 5000H mem addr into Ac
8001	MOV B, A	47	Move Ac value to reg B
8002	MVI A, 6000H	3E, 35H	Load 6000H mem loc^{n} into Ac
8003	MOV C, A	4F	Move accumulator to c
8004	MOV A, B	78	Move content of B to Ac
8005	STA	32	Store the Ac value to mem loc^{n} 6000H
8006	—	00H	LSB
8007	—	60H	MSB
8008	MOV A, C	79	Move content of reg C to Ac
8009	STA	32	Store Ac value to mem loc^{n} 5000H
800A		00H	LSB
800B		80H	MSB
800C	HLT	76	Stop the program execution

before swapping

Mem loc	5000H
value	25
Mem loc	6000H
value	35

Memory

→ Afterswapping

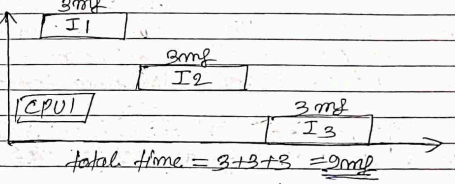
Mem loc	5000H
value	35
Mem loc	6000H
value	25

Memory.

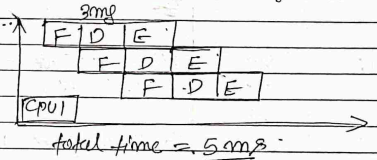
(Q8) How pipelining improves performance of a microprocessor.

ans⇒ Pipelining improves performance of microprocessor by executing multiple instructions simultaneously. It involves dividing a process into different stages and allowing them to execute simultaneously in overlaping manner.

Example: Sequential execution of instruction



pipeline execution of instruction



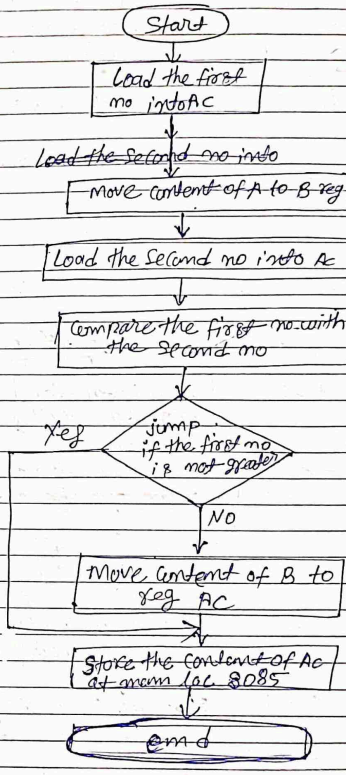
Since, sequential exe. of instruction takes 9 ms to execute 3 instn where as pipelin execution takes only 5ms to exe all 3 inst with only one CPU, hence pipelining improves performance of microprocessor.

(pg 47) write an assembly language program to find the maximum of two 8 bit numbers using the 8085 microprocessor.

Memory Add	Mnemonics	Machine code	Comments
8000	MVI A, 24H	3E 24H	Load the first no into Ac
8001	MOV B, A	47	Move the first no to reg B
8002	MVI A, 26H	3E 26H	Load the second no into Ac
8003	CMP B	B8	Compare the first no with the second no.
8004	JNC	DA	Jump to 8009 if the first no is not greater
8005	8009	8009	
8006	MOV A, B	78	Move the first no to the Ac
[illegible]			
[illegible]			
8007	STA	32	Store the content of Ac to Mem address.
8008	85	LSB	
8009	80	MSB	
800A	HLT	76	End of program execution

Imp1 → 24
Imp2 → 26
→ CMP B ⇒ 24 < 26
destination ⇒ 8085 ⇒ 26

flowchart



Pyq) How many references to memory are needed for each type of instruction to bring an operand into a processor register?

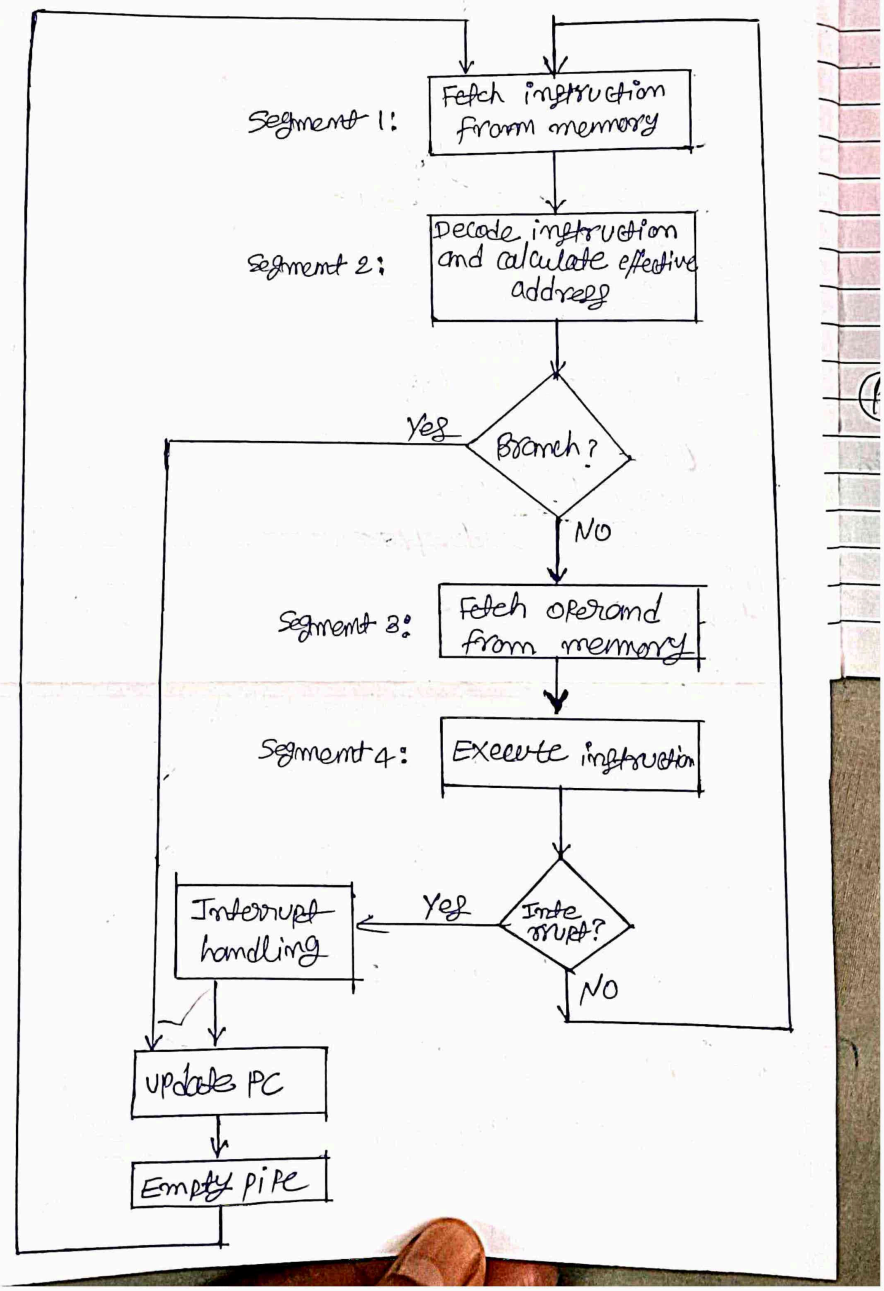
ans ⇒ For a direct address instruction you need two reference to memory.

(i) Read the instruction (ii) Read the operand itself

And for an indirect address instruction, need three references to memory.

(i) To Read instruction
(ii) To read effective address
(iii) To read operand

Pyq) Demonstrate the flowchart of Instruction pipeline



In general every instruction must be processed by the computer in following order.

(i) fetching the instruction from memory
(ii) Decoding the obtained instruction
(iii) calculating the effective address
(iv) fetching the operands from the given memory
(v) Execution of the instruction
(vi) storing the result in a proper place

The instruction cycle is divided into four parts segment 1: The implementation of the instruction feteh segment can be done using the fifo.

Segment 2: In the second segment, the memory instruction is decoded and the effective address is then determined in a separate arithmetic circuit

Segment 3: in third segment some operands would be fetched from memory.

Segment 4: This instruction would finally be executed in the very last segment of a pipeline organization.

(Q.8) write the meaning and explanation of following instructions (a) SIM (b) CMP (c) XRI (d) JC (e) STA

(i) SIM: SIM stands for Set interrupt MASK SIM is an instruction that enables or disables interrupts. The SIM instruction allows the programmer to control whether interrupts

are allowed to interrupt the current program being executed by the processor

(b) CMP: CMP stands for compare, CMP is an instruction used to compare two operands. It is used in conditional branching to determine if two values are equal, greater or less than each other.

(c) XRI: XRI stands for Exclusive OR immediate. XRI is a instruction that performs a bitwise exclusive OR (XOR) operation between a specified register or memory location and an immediate value. This instruction is commonly used in digital logic and programming to manipulate binary data.

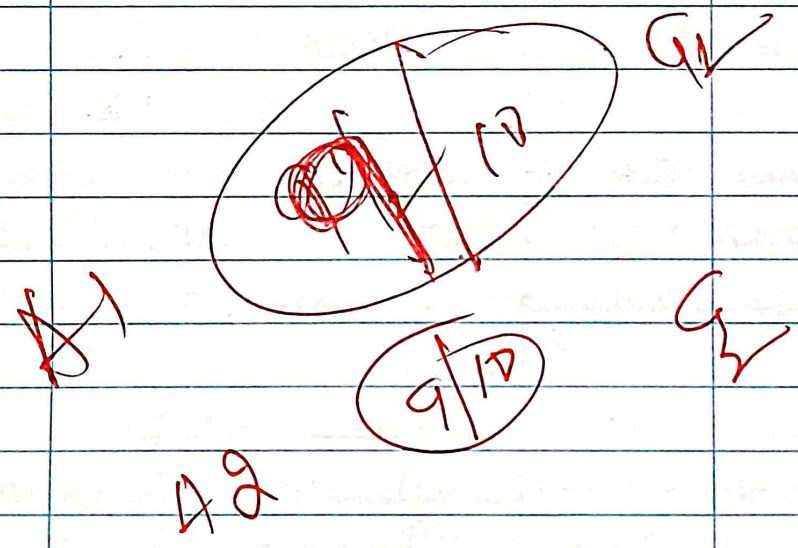
(d) JC: Jc stands for Jump if Carry, JC is a conditional branch instruction that cause a jump to a specified memory location if the carry flag is 1. The carry flag is typically set as a result of arithmetic operations like addition or subtraction.

(e) STA: STA stands for store Accumulator. STA is an instruction, used to transfer the contents of the accumulator register to a specified memory location. This instruction is commonly used to store the result of calculations :

INDEX

Name	RAUSHAN KUMAR
School	GNE College
Subject	CAM
ID/Roll	2221139/220375 1
Class	IT (B2)
Section	B2

S.No.	Date	Title	Page	Teacher's Sign / Remark
01				
02				
03				
04				
05				
06				
07				
08				
09				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				



Name : Raushan Kumar
CRN : 2221139
URN : 2203751

FREEMIND

Date

Page

Q1. Difference between microprocessor and microcontroller.

Parameter	① Microprocessor	① Microcontroller
Definition	Microprocessor is a processor that execute instruction one by one and control I/O device.	A microcontroller is a small computer on a single integrated circuit that is designed to control specific task.
Speed	② It has a high clock speed.	② It has a lower clock speed.
Cost	③ It is too costly.	③ It is less costly.
Size	④ Large in size	④ Small in size.
Application	⑤ Computers, laptops	⑤ Micro wave, washing machine
Requires	⑥ It requires more money	⑥ It requires less memory
Use	⑦ Widely use in computer system	⑦ Widely use in embedded system.
Connectivity	⑧ Memory, I/O ports, timers etc. connected to the CPU externally.	⑧ CPU and all other elements are integrated into a single chip.
Architecture	It is 32 bit or 64 bit architecture.	It is of 8, 16 or 32 bit architecture.

FREEMIND

Date

Page

Q2 What is the purpose of Assembly language?

- Direct Hardware interaction.
- Learning Computer Architecture.
- plateform-specific Development
- performance optimization.
- Memory efficiency
- Reverse Engineering
- Human readable machine code/language.
- Modification and error debugging can be ~~easy~~ easly done.

Q3 Difference b/w 1 byte, 2 byte and 3 byte instruction with the help of example.

parameter	1 byte	2 byte	3 byte
Memory size	It requires 1 byte memory.	It requires 2 byte memory.	It requires 3 byte memory.
Information.	It encodes opcode.	It encodes opcode and operand	It encodes opcode and 16-bit operand.
Example	MOV A, 42	ADD A, B	JMP ~~start~~ go ahead
use cases	Simple operations within the CPU	Balance b/w instruction size and capability	Necessary for architectures with large memory space.
Execution speed	Execution is fast.	Moderate	slow

FREEMIND

Date

Page

(Q4) Difference b/w low level, middle level and high level language.

Parameter	low level lan	Middle level lan	High level lan
Abstraction	very low close to machine code	Moderate dealy with some hardware details	High closer to human language
Readability	Difficult to read and write	More readable than low but less than high.	Eafy to read an write
portability	low	Moderate	High
Development speed	slower	Moderate	faster.
Performance	potentially better due to direct hardware control	Moderate	potentially lower due to compilation overhead.
Example	Machine code Assembly language	C, C++	python, Java Java script

FREEMIND

Date

Page

Q5 A computer register T of 8-bits is having hexa-decimal CB as its initial value. what will be the value of status bits CY, S, Z and Ac after adding the immediate operand hexadecimal E9 to T.

ans ⇒

```
  CB
+ E9
-----
 1B4
```

(1) cy (carry) :- There is a carry generated in the addition (CB + E9), so CY = 1

(2) S (sign) : The most significant bit of the result (1B4) is 1, indicating a negative number - so, S.

(3) Z (zero) : The result (1B4) is not zero, so Z = 0

(4) Ac (Auxiliary carry) : Auxiliary carry is used in BCD arithemetic it's not relevent in this case, so Ac = 0.

CY = 1

S = 1

Z = 0

Ac = 0

FREEMIND

Date

Page

(Q6.) The two word instruction of address 200 and 201 is a load to AC with an address field equal to 500, pc = 200, R1 = 400, XR = 100.

ans →

pc = 200

R1 = 400

XR = 100

AC

Mode	Effective address	Accumulator
IA	201	500
DAM	500	800
IAM	800	300
RM	—	400
RIM	400	700
AI	400	700
RAM	702	325
AD	399	450
IAM	600	900

address	memory
200	load to Ac/Mode
201	Address = 500
	⋮
399	450
400	700
	⋮
500	800
	⋮
600	900
	⋮
702	325
	⋮
800	300

FREEMIND

Date

Page

Assignment - 2

Q1 differentiate between virtual memory and cache memory.

Parameters	Virtual Memory	cache memory
1. Purpose	increases main memory capacity.	increases CPU accessing speed.
2. Memory unit	virtual memory is not a memory unit, its a technique.	cache memory is a memory unit.
3. Size	Larger than cache memory.	smaller than virtual memory.
4. Management	Managed by the OS	Managed by hardware (CPU)
5. Usage	It executes programs	It stores used data.
6. Address Mapping	Mapping frameworks are required.	No Mapping frameworks are required.
7. Speed	Slower than cache memory	Faster accessing speed than virtual memory.
8. data	less frequently used data / program	More frequently used data / instructions.

FREEMIND

Date

Page 1

Q2. With the help of applications, state how the use of microprocessor makes daily life ~~eatis~~ easier / micro-controller also.

ans ⇒ Here is how microprocessor and microcontroller makes daily life easier through various application.

Microprocessor - Application

1. Smartphones: Smartphones these tiny computers in our pockets rely on microprocessors for everything from making calls, sending texts to running apps and games.

2. Computers and Laptops: Computers and laptops are also rely on microprocessor from browsing the web to running software, microprocessor handles all the complex calculations ~~needed for~~.

3. Appliances: Modern appliances like refrigerators and washing machine often have microprocessors that control functions, optimize settings, ~~and~~

FREEMIND

Date

Page

4. Gaming : High- graphic and immersive gameplay require powerful microprocessor to render the visuals and handle complex physics simulations.

Microcontroller Application

1. Embedded systems : Microcontrollers are the brains of embedded systems. like traffic light controllers, robots, micro wave, medical devices.

2. Consumer Electronics : from thermostats and microwave ovens to toys and remote controls, microcontroller manage specific functions and user instructions.

3. Vehicles : Modern cars are packed with microcontrollers that control everything from engine management to airbag deployment and entertainment systems.

Q3 Elaborate the concept of memory hierarchy with the help of neat and clean diagram.

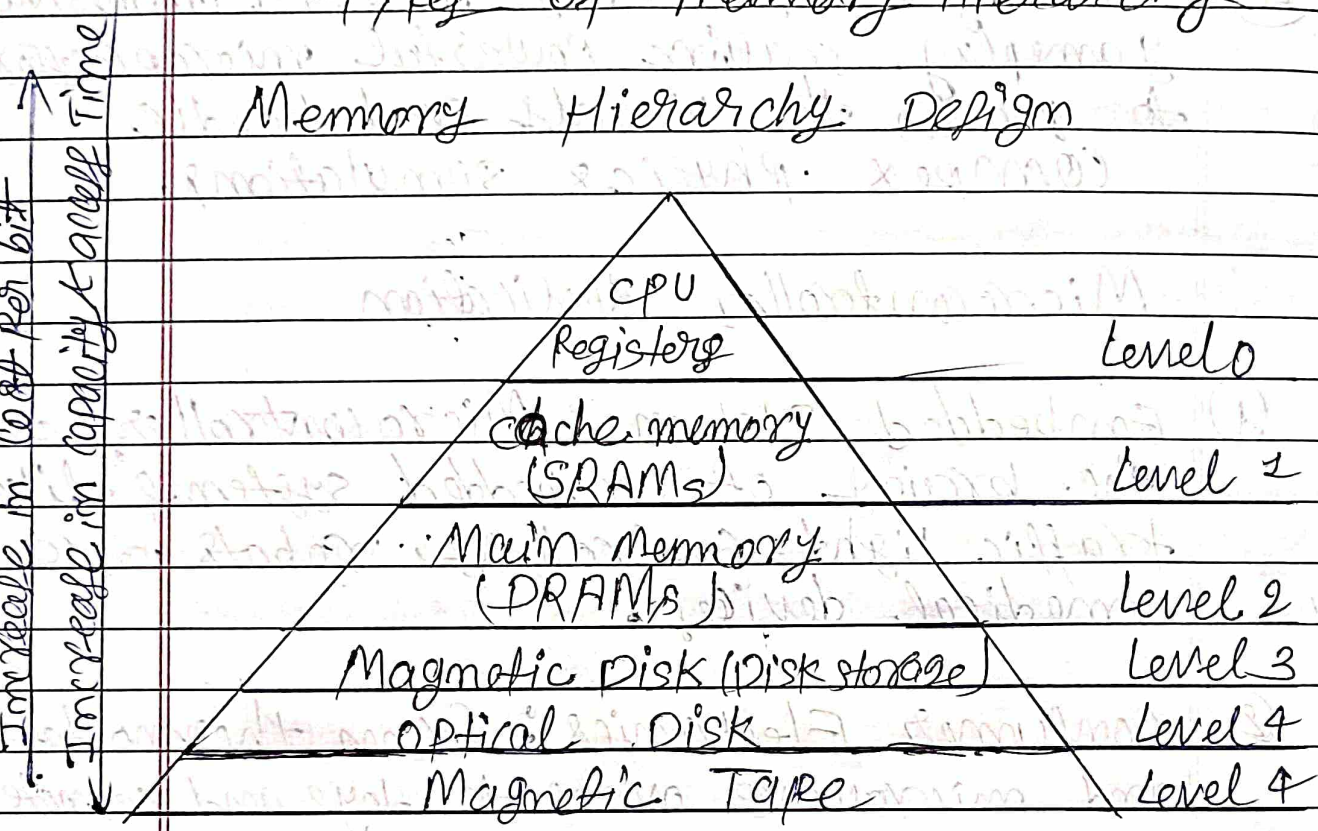
FREEMIND

Date

Page

Types of Memory Hierarchy

Memory Hierarchy Design



① Registers: These are the fastest and smallest storage locations in a computer's memory hierarchy. Registers are located in the CPU itself and are used to store data that the CPU needs to access quickly. range from 16 to 64 bits.

② Cache Memory: cache memory is a small, fast memory unit located close to the CPU. It stores frequently used data and instructions.

FREEMIND

Date

Page

It is designed to minimize the time, ~~it takes~~ to access data by providing the CPU with quick access.

(3) Main Memory: Main memory also known as RAM (Random Access Memory) It is primary memory of computer system. It has a larger storage capacity than cache memory, but it is slower, Main memory is used to store data and instructions.

Types of Main memory

- Static RAM: Static RAM stores the binary information in flip flops and information remains valid until power is supplied. It has a faster access time and it used in cache memory.
- Dynamic memory: It stores the binary information of a charge on the capacitor. It requires refreshing circuitry to maintain the charge on the capacitor.

(4) Secondary storage: Secondary storage such as hard disk drives (HDD) and solid state drives, is a non-volatile memory unit that has a larger storage

FREEMIND

Date

Page

capacity than main memory. It is used to store data and instructions that are not currently in use by the CPU. It has the slowest access time and expensive type of memory in the memory hierachy.

(5.) Magnetic Disk: Magnetic Disks are simply circular plates that are fabricated with either a metal or a plastic or a magnetized material. The magnetic disks work at a high speed inside the computer and these are frequently used.

(6) Magnetic Tape: Magnetic Tape is simply a magnetic recording device that is covered with a plastic film. It is generally used for the backup of data. It is slower therefor it requires some amount of time for accessing the strip.

(9a) Discuss the memory hierarchy in Computer system with respect to speed, size, & cost.

FREEMIND

Date

Page

unit

	speed	size	Cost
Registers	fastest	very small	Very expensive
cache memory	fast	larger than Registers	Expensive but affordable, than registers.
Main memory	slower than cache	larger than cache	less expensive than cache but still significant cheaper per unit of storage compared to main memory.
Magnetic Disk	slower than main memory	larger than main memory	
Magnetic Tape	slower than magnetic disk	Extremely large	cheapest per unit storage.

(Pyq) what is the ~~one~~ Need and ~~significance~~ significanle of Memory hierarchy

ans → The memory hierarchy is need for efficient data access in computer systems. It organize memory into different levels based on speed, size and cost, allowing faster access to frequently used data and reduce latency. The hierarchy is essential for optimizing performance and minimizing cost in computer system.

FREEMIND

Date

Page

(Q5) Differentiate between multiprocessor and multicomputer.

Parameter	Multiprocessors	Multicomputer
Architecture	shared memory architecture	Distributed memory architecture.
~~Communication Memory Access~~	Inter process communication via shared memory	Message passing b/w distinct processors.
Memory Access	shared memory accessed by all processors.	Each processor have its own local memory.
Scalability	Limited scalability due to shared memory access	scalable as processors can be added easily.
Programming	shared memory is required in programming	Message passing is required in programming.
Latency	lower latency	Higher latency
Example	~~Symmetric multiprocessor system~~	Grid computing.

FREEMIND

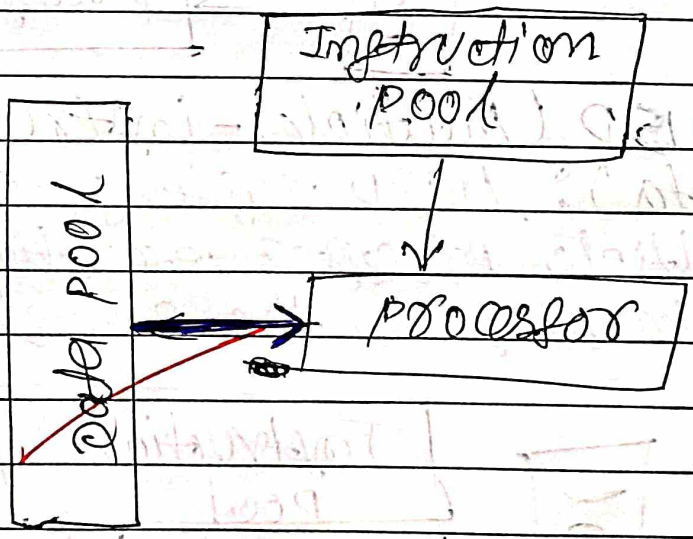
Date

Page

Q6 classify the organization of computer using flynn's criteria.

ans⇒ flynn's taxonomy classifies computer architecture based on the ~~num~~ number of concurrent instruction streams and data streams. It categorizes them into four main types:

① SISD (single Instruction, Single Data):
SISD systems consist of a single processor capable of executing one instruction at a time on a single data stream.



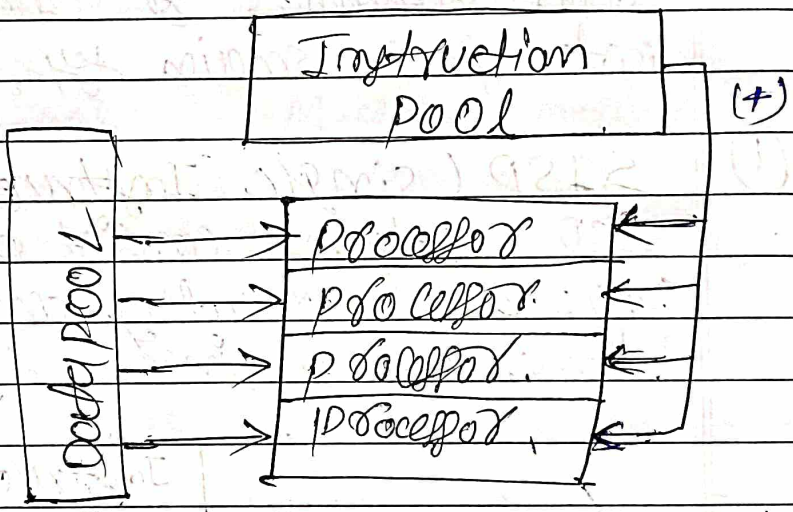
Ex⇒ desktop

FREEMIND

Date

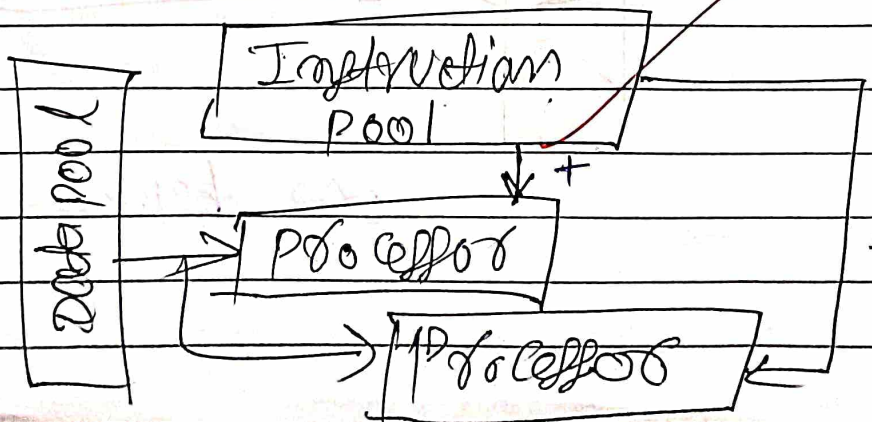
Page

(2.) Single-instruction, multiple-data (SIMD):
SIMD systems comprise multiple processors that execute the same instruction on different data streams simultaneously suited for tasks involving vector and matrix operations.



Ex⇒ GPUs

(3) MISD (Multiple-instruction, single-data): MISD systems comprise multiple processors executing different instructions on the same data stream



Ex⇒ RAID

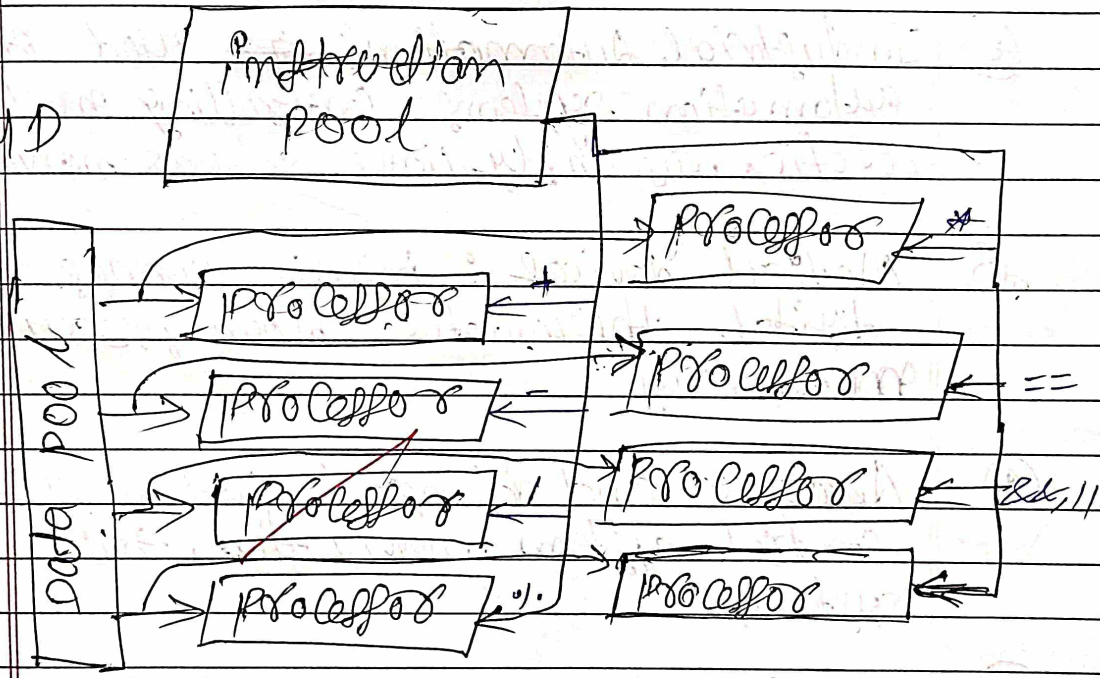
FREEMIND

Date

Page

④ MIMDC Multiple - instruction, multiple-data): MIMD comprises of multiple processors executing multiple instructions on multiple data stream concurrently. Each processor has separate instruction and data streams allowing for asynchronous operation.

MIMD



Ex⇒ cluster

FREEMIND

Date

Page

* Real world Applications of Microcontroller (PYQ)

ans ⇒ (1) Automotive systems: used in automobiles for engine management, controlling fuel injection system, managing airbags, anti-lock braking systems, climate control.

(2) Industrial Automation: used in automation systems, controlling machinery, robotics, assembly lines, sensors monitoring

(3) Medical devices: insulin pumps, digital thermometers, blood pressure monitoring (PYQ)

(4) Aerospace and defense: used in flight control system, navigation, guidance systems.

(5) IOT (internet of Things): widely used in IOT devices.

ans ⇒

ans ⇒

(2)

(3)

FREEMIND

Date

Page

Processing

(P4) Explain how the parallel ^ improves the performance of multiprocessing environment.

ans⇒ Parallel processing in multiprocessing environment improves performance by allowing multiple tasks to be executed simultaneously across multiple processors. This overall increases throughput and reduces the time it takes to complete tasks compared to sequential processing.

(P5) Specify the applications of microprocessor in household, consumer electronics and in medical science.

ans⇒ ① Household applications: Microprocessors are used in appliances like washing machine, refrigerators, microwave ovens and AC.

② Consumer electronics: Devices like smartphone, tablets, TVs and gaming uses microprocessors.

③ Medical Seciences: Devices like insulin pumps, digital thermometers, blood pressure monitors.

FREEMIND

Date

Page

(Q5) what is the need significance of memory hierarchy

ans⇒ Memory hierarchy is arranging different types of storage devices in a computer based on their size, cost and access time/speed. The main purpose of memory hierarchy is to achieve efficient operations by organizing the memory to reduce access time.