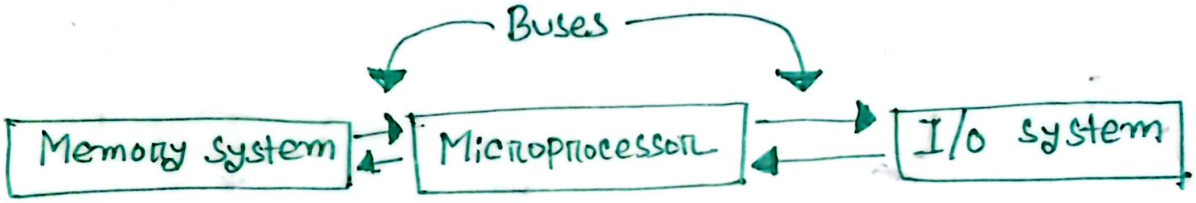
Sub.: (20-21) Sifat Tanmoy

SAT SUN MON TUE WED THU FRI

DATE : / /

1. (a) Who developed the Analytical Engine? Draw the block diagram of a microprocessor-based Computer system and show the 8086 architecture. (3)

→ Charles Babbage.



Dynamic RAM (DRAM)
Static RAM (SRAM)
Cache
Read-only (ROM)
Flash memory
EEPROM
SDRAM
RAMBUS
DDR DRAM

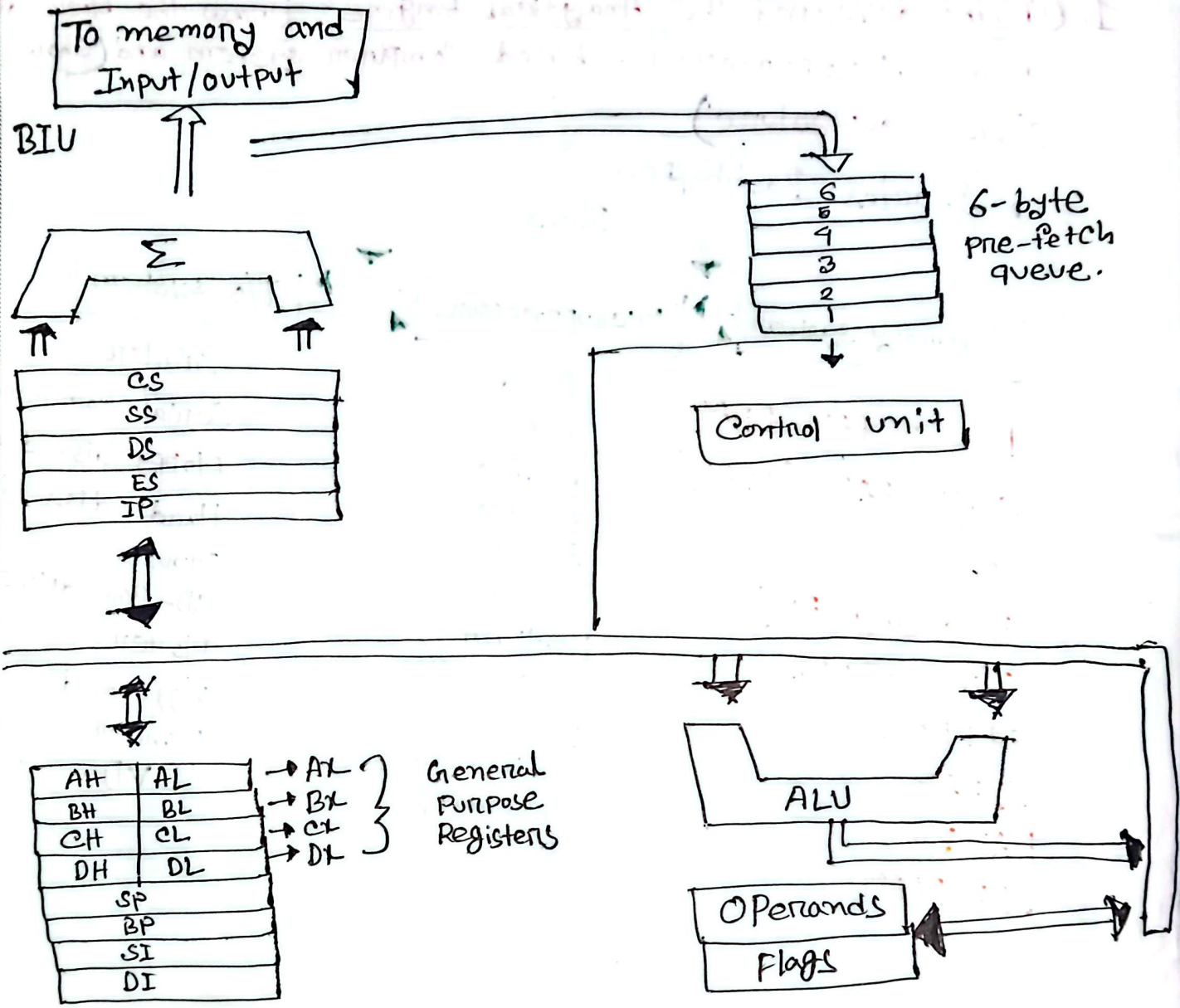
8086
8088
80186
80286
80386
80486
Pentium
Pentium Pro
Pentium II
Pentium III
Pentium 4
Core 2

Printer
Serial communications
Floppy disk drive
Hard disk drive
Mouse
CD-ROM drive
Plotter
Keyboard
Monitor
DVD.

Sub.: _____

SAT ☐ SUN ☐ MON ☐ TUE ☐ WED ☐ THU ☐ FRI ☐

DATE : / /



Architecture of 8086 Microprocessor

United

Sub.: _____ SAT SUN MON TUE WED THU FRI

DATE : / /

(b) Describe the function and purpose of each program-visible register in the 8086 microprocessors. (9)

Detail the flag register and the purpose of each flag bit.

Program-visible registers of 8086.

1. General purpose Registers → (AX, BX, CX, DX).
2. Segment Registers → (CS, DS, SS, ES)
3. Pointer and index Register → (SP, BP, SI, DI)
4. Flag Register. → (CF, ZF, SF, OF, etc)
5. Instruction Pointer. → IP [Points to the next instruction to be executed in the code segment].

The flag register is a special register that indicates the status of the CPU after arithmetic or logic operations.

Main Flags and their purposes:

S (Sign) : 1 if result is negative.

Z (Zero) : 1 if result is zero.

AC (Auxiliary carry) : 1 if carry from bit 3 to bit 4.

P (Parity) : 1 if result has even number of 1s.

CY (carry) : 1 if carry/borrow occurs.

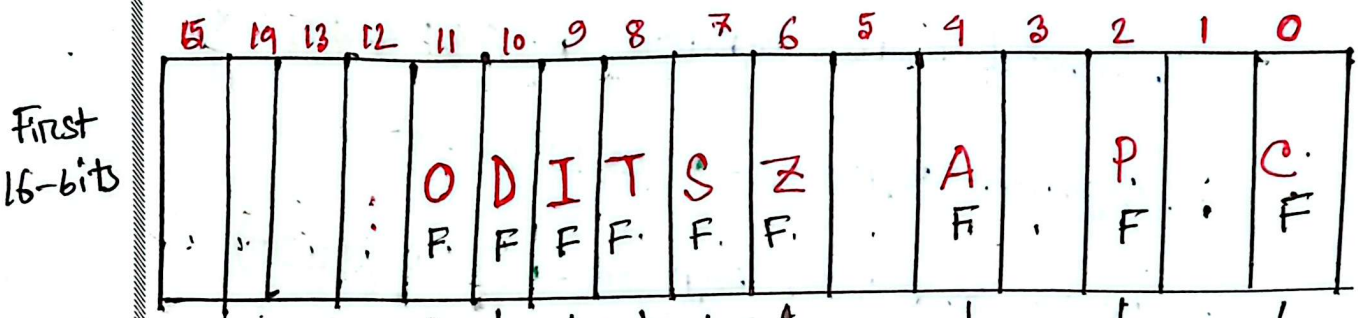
United

Sub.: _____

SAT SUN MON TUE WED THU FRI

DATE : / /

CPAZ



Overflow flag

Direction flag

Interrupt flag

Trap flag

Sign Flag (SF)

Zero flag

Auxiliary carry Flag (AF)

Parity Flag (PF)

Carry Flag (CF)

United

Sub.: _____ SAT SUN MON TUE WED THU FRI

DATE : / /

(c) Describe how memory is accessed using real mode memory-addressing techniques. (3)

In real mode, memory is accessed using a segment: offset addressing technique.

There are four main segment registers:

- CS → Code segment
- DS → Data segment
- SS → Stack segment
- ES → Extra segment.

The segment register gives the starting address of a 64 kB block, and the offset specifies the location within that block

The physical address is calculated as:

$$\text{Physical address} = (\text{segment} \times 16) + \text{offset}$$

Example: if segment = 1000h and offset = 0020h,

then physical address 10020

This allows the CPU to access up to 1 MB of memory using 20-bit addresses.

United

Sub.: _____ SAT SUN MON TUE WED THU FRI

DATE : / /

(d) What is the flat mode memory system? Detail the operation of memory-paging mechanism. —(4)

In a flat mode memory system, the entire memory is treated as continuous block of addresses, rather than being divided into segments.

All programs use a single, linear address space, where logical addresses = physical addresses.

This means there's no segmentation - every memory location can be accessed directly using a 32-bit or 64-bit address.

Memory-Paging →

1. CPU generates a virtual address.

2. TLB check: if entry found, get frame number.

3. Page table maps page → frame (valid bit = 1)
→ Combine frame + offset → access RAM.

United

Sub.: _____ SAT SUN MON TUE WED THU FRI

DATE : / /

(4) If valid bit = 0 ⇒ Page fault

(5) OS picks a free frame.

(6) Faulting instruction restarts and memory access succeeds.

Sub.: C-3 SAT SUN MON TUE WED THU FRI

DATE : / /

(2) (a) What assembly language directive indicates the start of the code segment? Explain the operation of each data-addressing mode.

Code segment

The directive marks the beginning of the code section in an assembly program, where executable instruction are written:

Example:

```
Code segment
    MOV AX, BX
Code ENDS.
```

C-3

Data Addressing Modes: (MOV AL, 25H)

1. Immediate - Operand is given in instruction
2. Register - Operand is in a Register - (MOV AX, BX)
3. Direct - Memory address is specified directly (MOV AL, [2000H])
4. Register Indirect - Register holds memory address " ", [BX] - SI=05H
5. Indexed - Effective address = Index Reg + displacement " ", [BX+10H]
6. Based - " = Base Reg + " , " ", [BX+10H]
7. Based-Indexed " = Base + Index + ",

MOV AL, [BX, SI, 05H]

United

Sub.: _____ SAT SUN MON TUE WED THU FRI

DATE : / /

⑥ Illustrate the operation of the MOV AX, [BX] instruction when BX = 1000H and DS = 0100H. Note that this instruction is shown after the contents of memory are transferred to AX.

MOV AX, [BX] copies the 16-bit data from memory into AX.

Given, DS = 0100H
BX = 1000H.

Physical address = (DS X 10H) + BX.
= (0100H X 10H) + 1000H.
= 2000H.

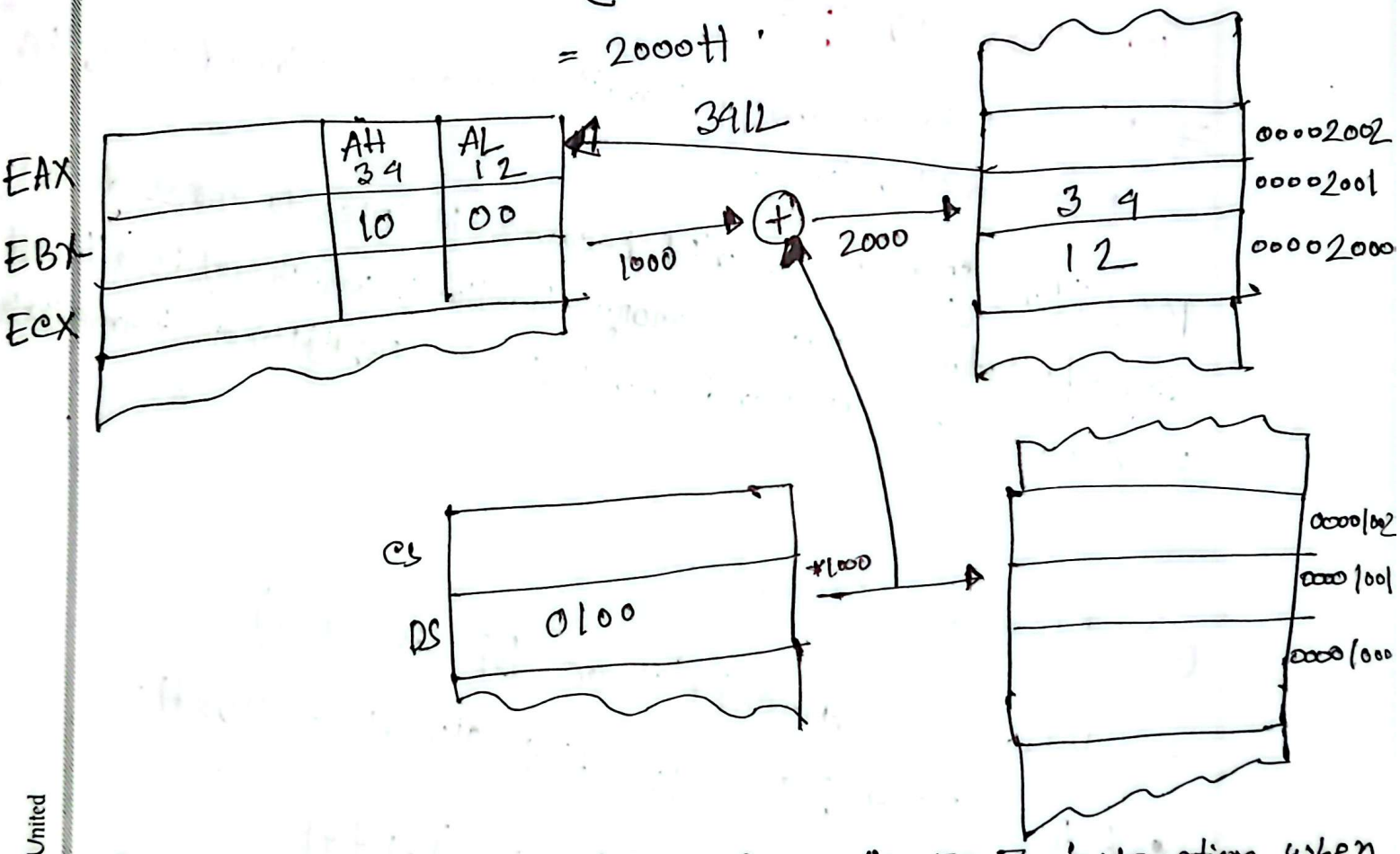


Figure: The operation of the MOV AX, [BX] instruction when BX = 1000H and DS = 0100H.

United

Sub.: 3 marks (20-21)

SAT SUN MON TUE WED THU FRI

DATE : / /

(C3-29)

@ What is wrong with a MOV [BX], [DI] instruction?

Suppose that DS = 1000H, SS = 2000H,

BP = 1000H, DI = 0100H.

Determine the memory address accessed by each of the following instructions, assuming real mode operation?

(i) MOV AL, [BP+DI] (iii) MOV EDX, [BP]

(ii) MOV CX, [DI].

⇒ Memory to memory transfers are not allowed with the MOV instruction.

MOV [BX], [DI] is invalid because it tries to move data directly between two memory locations, which not is allowed in 8086. One operand must be a register or immediate.

@ MOV AL, [BP+DI]

⇒ Uses SS segment (Since BP is used)

- EA = BP + DI = 1000H + 0100H = 1100H.
- Physical address = (SS X 10H) + EA
  = (2000H X 10H) + 1100H
  = 21100H.

United

Sub.: _____ SAT SUN MON TUE WED THU FRI

DATE : / /

(b) MOV CX, [DI]

- uses Ds segment
- EA = DI = 0100H.
- Physical address = (Ds x 10H) + EA.
  = (1000H x 10H) + 0100H.
  ⇒ 10,100H.

(c) MOV EDX, [BP]

- uses ss segment.
- EA = BP = 1000H.
- Physical address = (ss x 10H) + EA.
  = (2000H x 10H) + 1000H.
  = 21000H.

United

Sub.: (3 marks) 9 1 3 3

SAT SUN MON TUE WED THU FRI

DATE : / /

(d) Describe the sequence of events that place data onto the stack or remove data from the stack. Illustrate the PUSH and POP instructions;

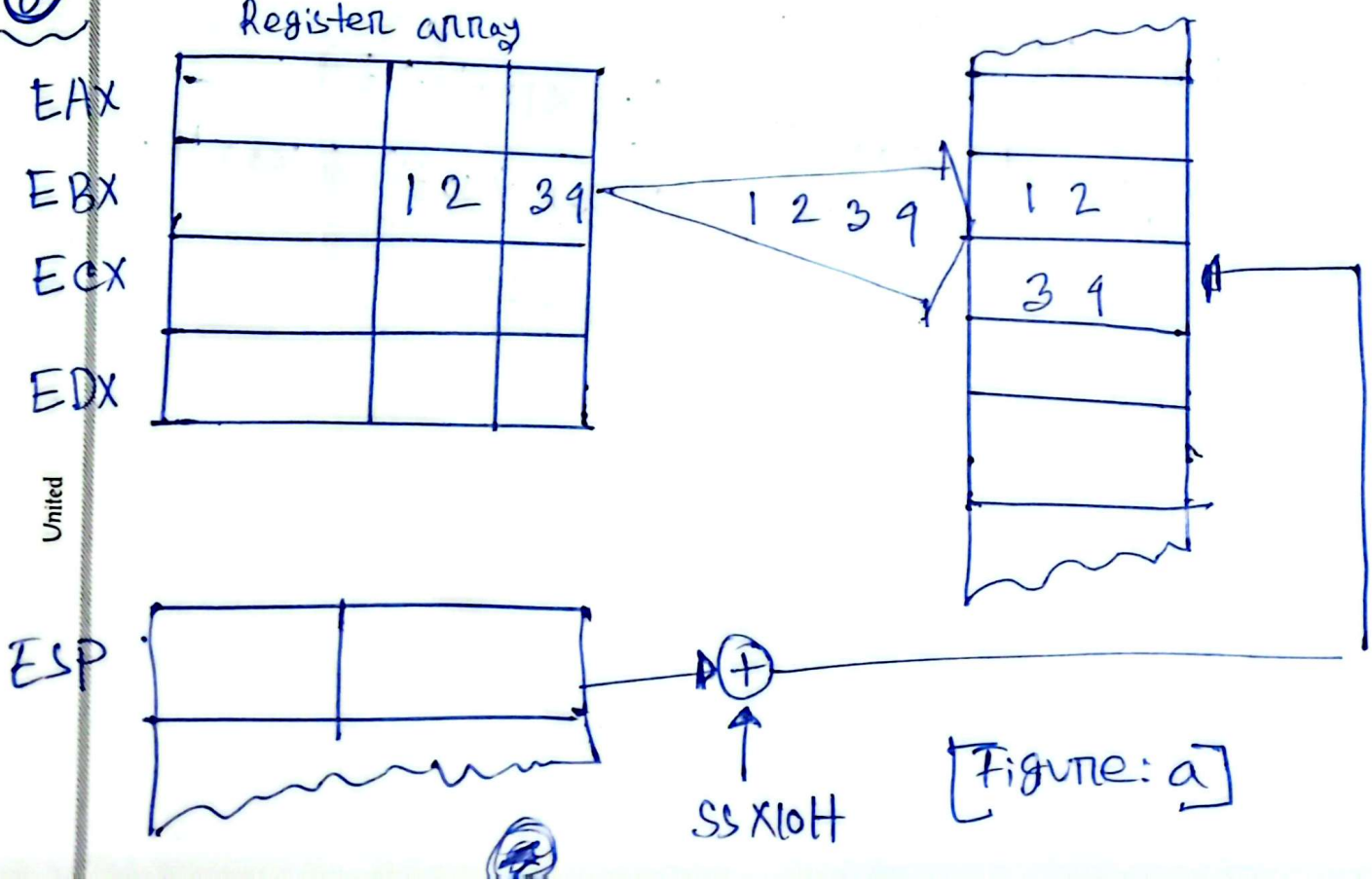
(a) PUSH BX places the contents of BX onto the stack.

(b) POP CX removes data from the stack and places them into CX. Both instruction are shown after execution.

PUSH: - Decreases SP by 2 and stores data at the new top of the stack (in ss segment)

- POP: Reads data from the top of the stack into a register and increases SP by 2.

(a)



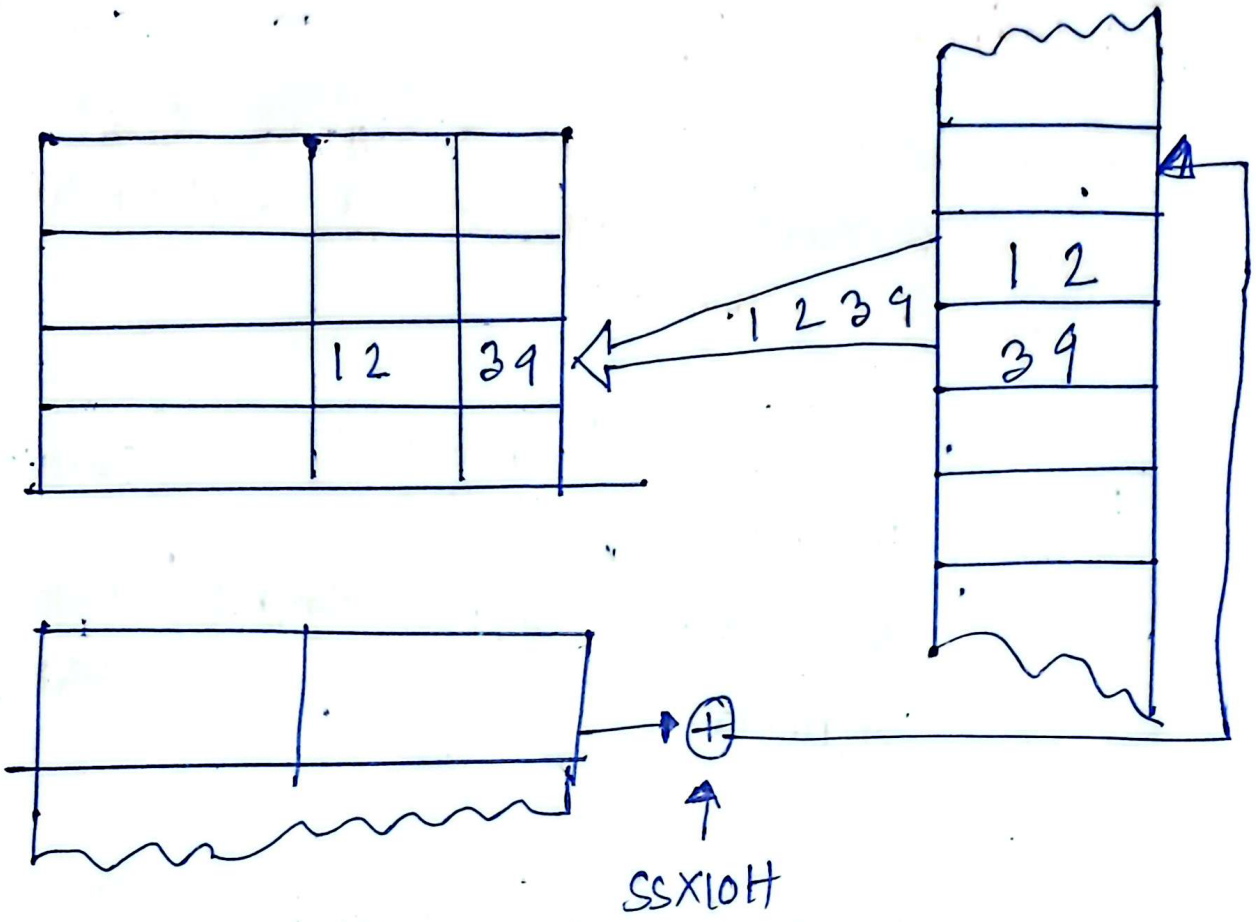
[Figure: a]

Sub.: _____

SAT ☐ SUN ☐ MON ☐ TUE ☐ WED ☐ THU ☐ FRI ☐

DATE : / /

(b)



[Figure: 1b]

United

Sub.: (3 marks)

SAT SUN MON TUE WED THU FRI

DATE : / /

(3) (a) What is the difference between intersegment and intrasegment jump?

If a near jump uses a signed 16-bit displacement, how can it jump to any memory location within the current code segment?

Type of Jump	Segment involved	Registers affected	Description
Intrasegment Jump	Same code segment	Only IP changes	Control stays within the
Intersegment Jump	Different code segment	Both CS and IP changes.	Control transfers to another code segment.

United

Sub.: _____ SAT SUN MON TUE WED THU FRI

DATE : / /

- The near Jump reads a 16-bit signed displacement (range −32768 to +32768)
- CPU sets IP: (next IP) + displacement.
- Only IP changes, CS is unchanged, so control transfers within the current 64 KB code segment.
- Any location in the segment is reachable directly if within ±32 KB, or via multiple near jumps for farther spots.

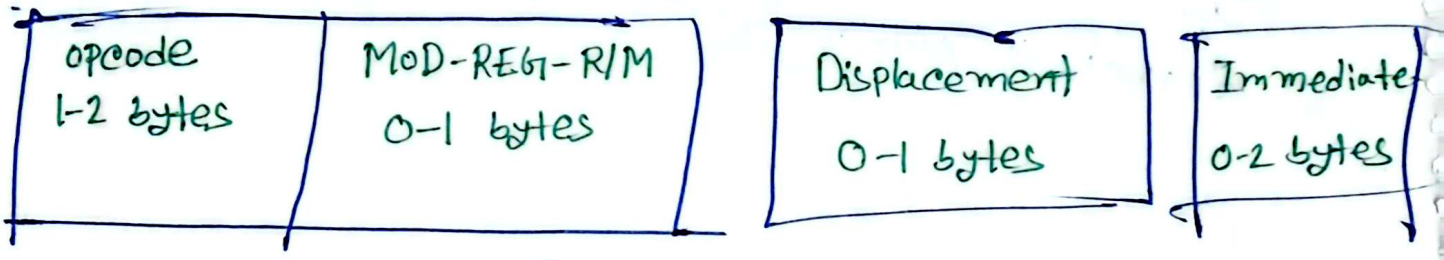
Sub.: (4 marks) (c-4)

SAT SUN MON TUE WED THU FRI

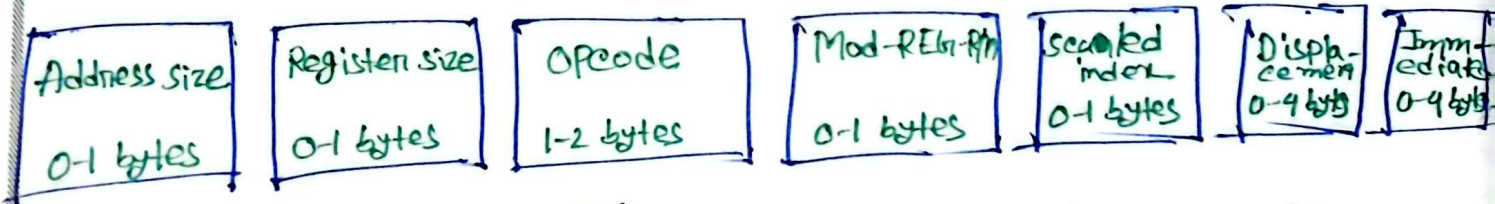
DATE : / /

(3) (b) Define opcode. Draw the formats of the 8086 - core instructions.

⇒ An opcode (operation code) is the part of an instruction that tells the CPU which operation to perform, such as add, move or jump.



(a) 16-bit instruction mode



(b) 32 bit instruction mode.

United

Sub.: 3 marks

SAT SUN MON TUE WED THU FRI

DATE: / /

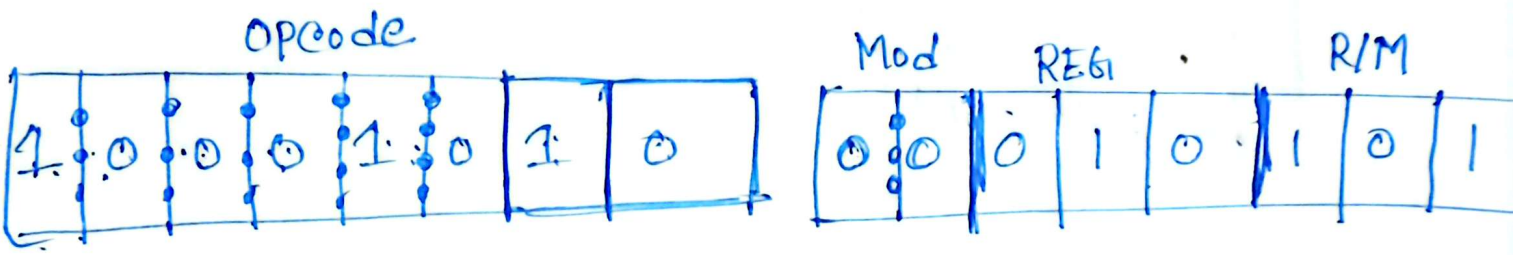
Write down the purpose of direction flag.

Convert MOV DL, [DI] instruction to its machine language form.

⇒ If the direction flag is cleared it selects auto-increment for the string instructions and if the direction flag is set is selects auto increment.

Effect: DF = 0 ; indexes increment → Process low → high address

DF = 1 ; " decrement → Process high → low "



Opcode = MOV

D = transfer to register.

W = byte.

MOD = No displacement.

REG = DL.

R/M = DS:[DI]

Sub.: 4 marks.

SAT SUN MON TUE WED THU FRI

DATE : / /

(d) What is wrong with a MOV CS, AX instruction? Show the shift and rotate instructions in details.

⇒ The contents of CS will change causing an unpredictable jump.

Mov CS, AX is invalid because the CS register cannot be changed directly by a MOV instruction; it can only be modified by far jump, far call, or interrupt.

Shift and rotate instructions:

1. Shift instructions:

* Move bits left or right; zeros are filled in.

- Types
  - SHL/SAL: shift left; MSB → CF, LSB = 0
  - SHR: shift right; LSB → CF, MSB = 0.
  - SAR: Arithmetic right; sign bit unchanged

Sub.: _____ SAT SUN MON TUE WED THU FRI

DATE : / /

2. Rotate instructions:

- Bits are ~~Rotated Rotated~~ rotated in a circle.
- Types:

- ROL / ROR : Rotate left/right; end bit re-enters from opposite end.
- RCL / RCR : Rotate through carry flag.

Sub.: 3 marks

SAT SUN MON TUE WED THU FRI

DATE : / /

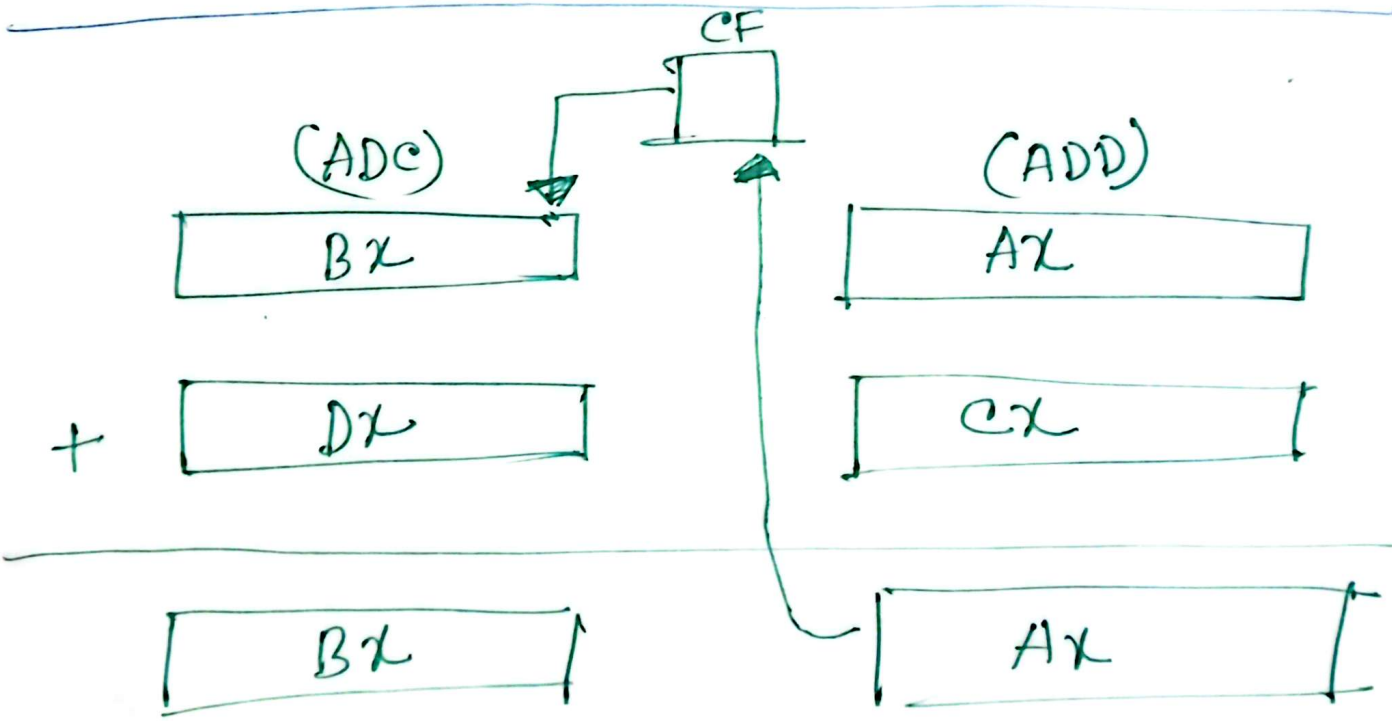
④ @ What is the purpose of ADC (addition-with-carry) instruction?

Illustrate and describe how the carry flag (C) links the two 16-bit-additions into one 32-bit addition?

⇒ The ADC (Addition-with-carry) instruction adds two operands along with the carry flag (CY) from a previous operation.

- It is mainly used for multi-byte addition.

Example: A ← A+R+CY.



[Figure]

United

Sub.: 4 marks.

SAT SUN MON TUE WED THU FRI

DATE : / /

(b) What is wrong with the INC [BX] instruction?

(1) What is the difference between the NOT and the NEG instruction?

(2) What happens if AH = 02H and DL = 43H. When the INT 21H instruction is executed?

---

(1) INC [BX] is invalid because the assembler cannot determine the operand size (byte or word). You must specify it explicitly (e.g, INC BYTE PTR [BX] or INC WORD PTR [BX].

---

(2) It displays the characters stored in DL on the screen.

if DL = 43H, which is ASCII for 'C'

when AH = 02 H and INT 21H is executed, Dos function "Display output" is called.

United

Sub.: 3 marks

SAT SUN MON TUE WED THU FRI

DATE : / /

(a) Use both conditional and unconditional Jump instructions to control the flow of a program.

Illustrate a near jump that adds the displacement (00211) to the contents of IP.

Conditional Jump instructions

Assembly Language	Tested condition	Operations.
JA	Z=0 and C=0	Jump if above
JB	C=1	Jump if below.
JC	C=1	Jump if carry.
JL	S!=0	Jump if less than
JNC	C=0	Jump if no carry.

Unconditional Jump instructions

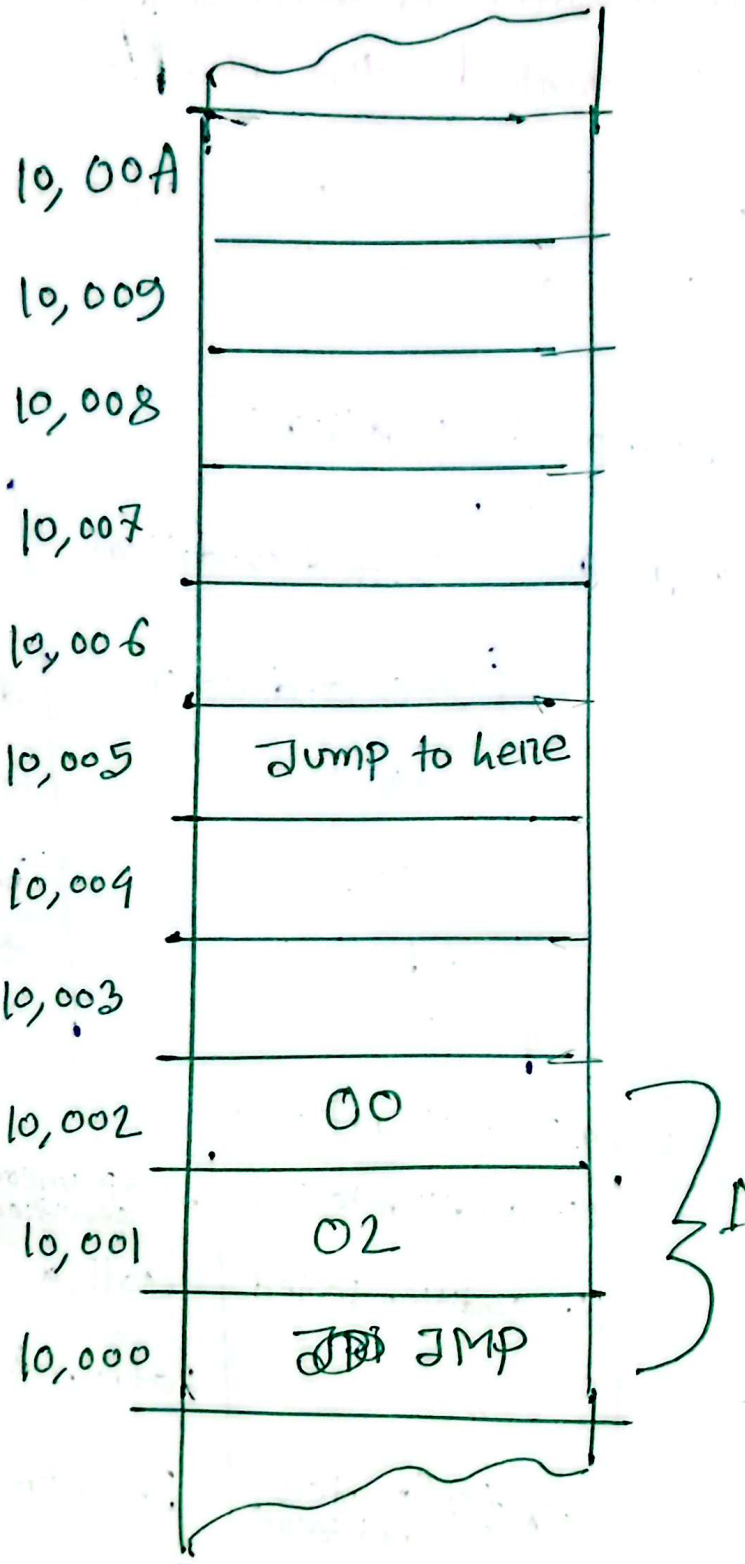
JMP	No condition tested	Jump unconditionally to the specified label.
CALL	No condition tested	Call a subroutine
RET	"	Return from subroutine.
LOOP	CX ≠ 0 (Counter not zero)	Decrement CX.
JMP Short	No condition tested	Short Jump within -128 to 127 bytes of current location

United

Sub.: _____

SAT ☐ SUN ☐ MON ☐ TUE ☐ WED ☐ THU ☐ FRI ☐

DATE : / /



CS = 1000H
IP = 0002H.
New IP = 0005H

[Figure: Near Jump]

Sub.: 4 marks.

SAT SUN MON TUE WED THU FRI

DATE : / /

(d) What is interrupt service procedure (ISP)? Use the call and return instructions to include procedures in the program structure

⇒ An interrupt is either a hardware-generated call or a software generated call. At times, an internal interrupt is called an exception.

Either type interrupts the program by calling an interrupt service procedure (ISP) or interrupt handler.

- Call is used to transfer control to a procedure, saving the return address on the stack.

- RET is used to return control from the procedure to the calling program by popping the return address from the stack.