Solution

First Sessional, Session: 2019-2020

Subject: Microprocessor, Code: KCS-403

SECTION-A

1(a) A microprocessor is a multipurpose, programmable logic device that reads binary instructions from a storage device called memory accepts binary data as input and processes data according to those instructions and provides result as output. The power supply of 8085 is +5V and clock frequency in 3MHz.

1(b) RLC: - Each binary bit of the accumulator is rotated left by one position. Bit D7 is placed in the position of D0 as well as in the Carry flag. CY is modified according to bit D7. Any other bit is not affected.

RAL: - Each binary bit of the accumulator is rotated left by one position through the Carry flag. Bit D7 is placed in the Carry flag, and the Carry flag is placed in the least significant position D0. CY is modified according to bit D7.

1(c) CALL is a 3-Byte instruction, with 1 Byte for the opcode, and 2 Bytes for the address of the subroutine. CALL mnemonics stands for “call a subroutine”. After executing the instructions written in the subroutine we shall want to return control to the next instruction written after the CALL instruction then we shall use mnemonic RET. Here RET stands for Return from the subroutine. RET is a 1-Byte instruction.

1(d) an interrupt is an input signal to the processor indicating an event that needs immediate attention. An interrupt signal alerts the processor and serves as a request for the processor to interrupt the currently executing code, so that the event can be processed in a timely manner. Therefore, if the request is accepted, the processor responds by suspending its current activities. Interrupt signal may be issued in response to hardware or software events and in this regard, they are classified as:

Hardware interrupt is caused by any peripheral device like mouse, keyboard, printer etc.
Hardware interrupt: TRAP, RST 7.5, RST 6.5, RST 5.5, INTR

A software interrupt occurs when an application software terminates or when it requests the operating system for some service. A software interrupt is generated by software and is considered one of the ways to communicate with the kernel or to invoke system calls, especially during error or exception handling.

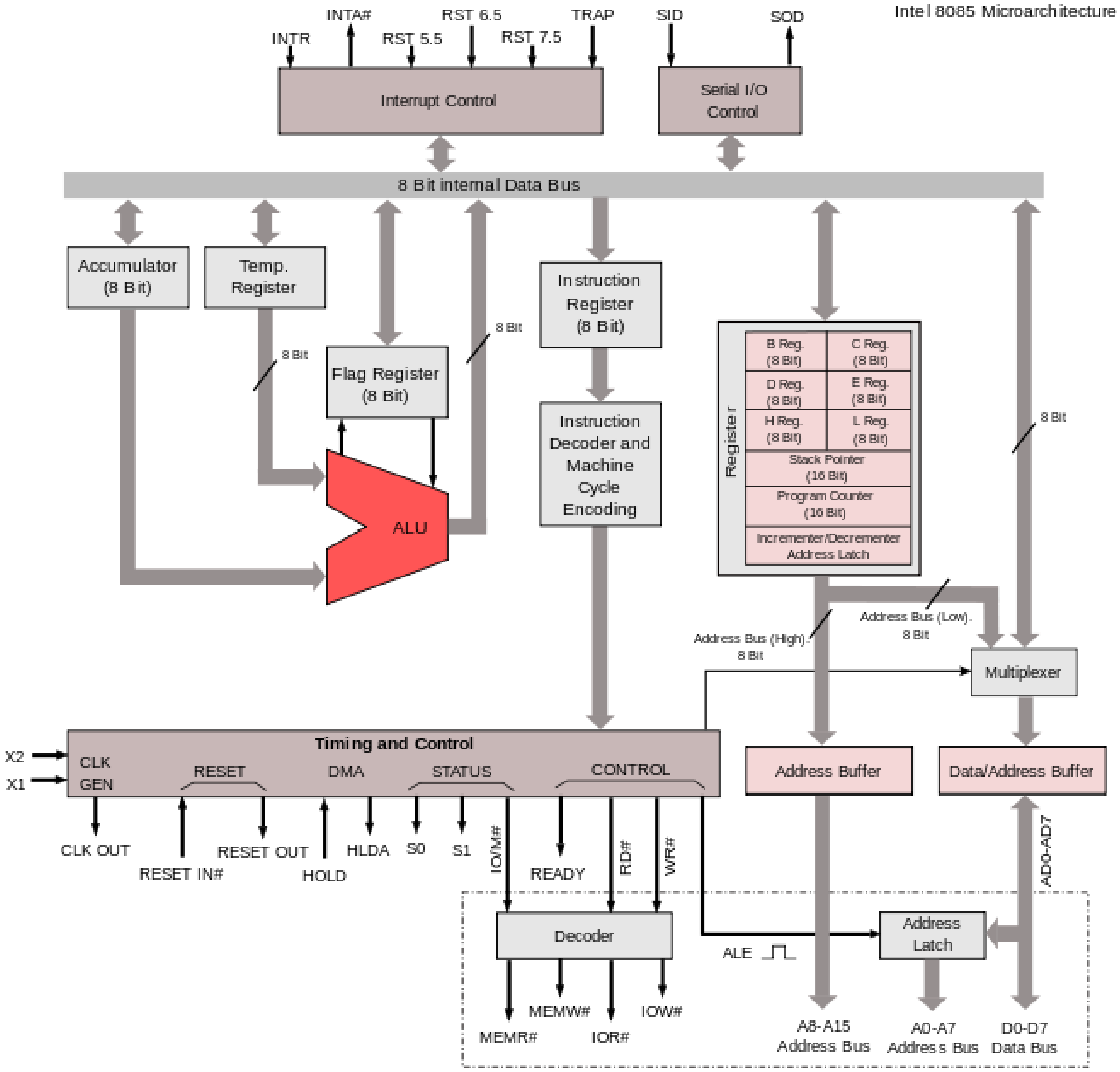
Software interrupt: RST n, where, n vary from 0 to 7.

1(e) **RAM** is Random Access Memory. **RAM** is a type of volatile memory. Data in RAM is not permanently written. When you power off your computer the data stored in RAM is deleted.

ROM is Read Only Memory. **ROM** is a type of non- volatile memory. Data in ROM is permanently written and is not erased when you power off your computer.

SECTION-B

2(a)



Operations of 8085 Microprocessor

The main operation of ALU is arithmetic as well as logical which includes addition, increment, subtraction, decrement, logical operations like AND, OR, Ex-OR, complement, evaluation, left shift or right shift. Both the temporary registers as well as accumulators are utilized for holding the information throughout in the operations then the outcome will be stored within the accumulator. The different flags are arranged or rearrange based on the outcome of the operation.

Flag Registers

The flag registers of microprocessor 8085 are classified into five types namely sign, zero, auxiliary carry, parity and carry. The positions of bit set aside for these types of flags. After the operation of an ALU, when the result of the most significant bit (D7) is one, then the sign flag will be arranged. When the operation of the ALU outcome is zero then the zero flags will be set. When the outcome is not zero then the zero flags will be reset. In an arithmetic process, whenever a carry is produced with the lesser nibble, then an auxiliary type carry flag will be set. After an ALU operation, when the outcome has an even number then the parity flag will be set, or else it is reset. When an arithmetic process outcome in a carry, then carry flag will be set or else it will be reset. Between the five types of flags, the AC type flag is employed on the inside intended for BCD arithmetic as well as remaining four flags are used with the developer to make sure the conditions of the outcome of a process.

Control and Timing Unit

The control and timing unit coordinates with all the actions of the microprocessor by the clock and gives the control signals which are required for communication among the microprocessor as well as peripherals.

Decoder and Instruction Register

As an order is obtained from memory after that it is located in the instruction register, and encoded & decoded into different device cycles.

Register Array

The general purpose programmable registers are classified into several types apart from the accumulator such as B, C, D, E, H, & L. These are utilized as 8-bit registers otherwise coupled to stock up the l6 bit of data. The permitted couples are BC, DE & HL, and the short term W & Z registers are used in the processor & it cannot be utilized with the developer.

Special Purpose Registers

These registers are classified into four types namely program counter, stack pointer, increment or decrement register, address buffer or data buffer.

Program Counter

This is the first type of special purpose register and considers that the instruction is being performed by the microprocessor. When the ALU completed performing the instruction, then the microprocessor searches for other instruction to be performed. Thus, there will be a requirement of holding the next instruction address to be performed in order to conserve time. Microprocessor increases the program when an instruction is being performed, therefore that the program counter position to the next instruction memory address is going to be performed.

Stack Pointer in 8085

The SP or stack pointer is a 16-bit register and functions similar to a stack, which is constantly increased or decreased with two throughout push and pop processes.

Increment or Decrement Register

The 8-bit register contents or else a memory position can be increased or decreased with one. The 16-bit register is useful for incrementing or decrementing program counters as well as stack pointer register content with one. This operation can be performed on any memory position or any kind of register.

Address-Buffer & Address-Data-Buffer

Address buffer stores the copied information from the memory for the execution. The memory & I/O chips are associated with these buses; then the CPU can replace the preferred data by I/O chips and the memory.

Address Bus and Data Bus

The data bus is useful in carrying the related information that is to be stock up. It is bi-directional, but address bus indicates the position as to where it must be stored & it is uni-directional, useful for transmitting the information as well as address input/output devices.

Timing & Control Unit

The timing & control unit can be used to supply the signal to the 8085 microprocessor for achieving the particular processes. The timing and control units are used to control the internal as well as external circuits. These are classified into four types namely control units like RD’ ALE, READY, WR’, status units like S0, S1, and IO/M’, DM like HLDA, and HOLD unit, RESET units like RST-IN and RST-OUT.

8085 Microprocessor Instruction Set

The instruction set of 8085 microprocessor is nothing but instruction codes used to achieve an exact task, and instruction sets are categorized into various types namely control, logical, branching, arithmetic, and data transfer instructions.

Addressing Modes of 8085

The addressing modes of 8085 microprocessor can be defined as the commands offered by these modes which are utilized for denoting the information in different forms without altering the content. These classified into five groups namely immediate, register, direct, indirect and implied addressing mode.

2(b) Generations of microprocessor:

First generation –

From 1971 to 1972 the era of the first generation came which brought microprocessors like INTEL 4004 Rockwell international PPS-4 INTEL 8008 etc.

Second generation –

The second generation marked the development of 8 bit microprocessors from 1973 to 1978. Processors like INTEL 8085 Motorola 6800 and 6801 etc came into existence.

Third generation –

The third generation brought forward the 16 bit processors like INTEL 8086/80186/80286 Motorola 68000 68010 etc. From 1979 to 1980 this generation used the HMOS technology.

Fourth generation –

The fourth generation came into existence from 1981 to 1995. The 32 bit processors using HMOS fabrication came into existence. INTEL 80386 and Mororola 68020 are some of the popular processors of this generation.

Fifth generation –

From 1995 till now we are in the fifth generation. 64 bit processors like PENTIUM, celeron, dual, quad and octa core processors came into existence.

Types of microprocessors:

Complex instruction set microprocessor –

The processors are designed to minimize the number of instructions per program and ignore the number of cycles per instructions. The compiler is used to translate a high level language to assembly level language because the length of code is relatively short and an extra RAM is used to store the instructions. These processors can do tasks like downloading, uploading and recalling data from memory. Apart from these tasks these microprocessor can perform complex mathematical calculation in a single command.

Example: IBM 370/168, VAX 11/780

Reduced instruction set microprocessor –

These processor are made according to function. They are designed to reduce the execution time by using the simplified instruction set. They can carry out small things in specific commands. These processors complete commands at faster rate. They require only one clock cycle to implement a result at uniform execution time. There are number of registers and less number of transistors. To access the memory location LOAD and STORE instructions are used.

Example: Power PC 601, 604, 615, 620

Super scalar microprocessor –

These processors can perform many tasks at a time. They can be used for ALUs and multiplier like array. They have multiple operation unit and perform tasts my executing multiple commands.

Application specific integrated circuit –

These processors are application specific like for personal digital assistant computers. They are designed according to proper specification.

Digital signal multiprocessor –

These processors are used to convert signals like analog to digital or digital to analog. The chips of these processors are used in many devices such as RADAR SONAR home theatres etc.

2(c) timing diagram of MVI A, 46H

Explanation of the command – It stores the immediate 8 bit data to a register or memory location.

MVI A, 46 H

Opcode: MVI

Operand: A is the destination register and 46 is the source data which needs to be transferred to the register. ’46’ data is stored in the A register.

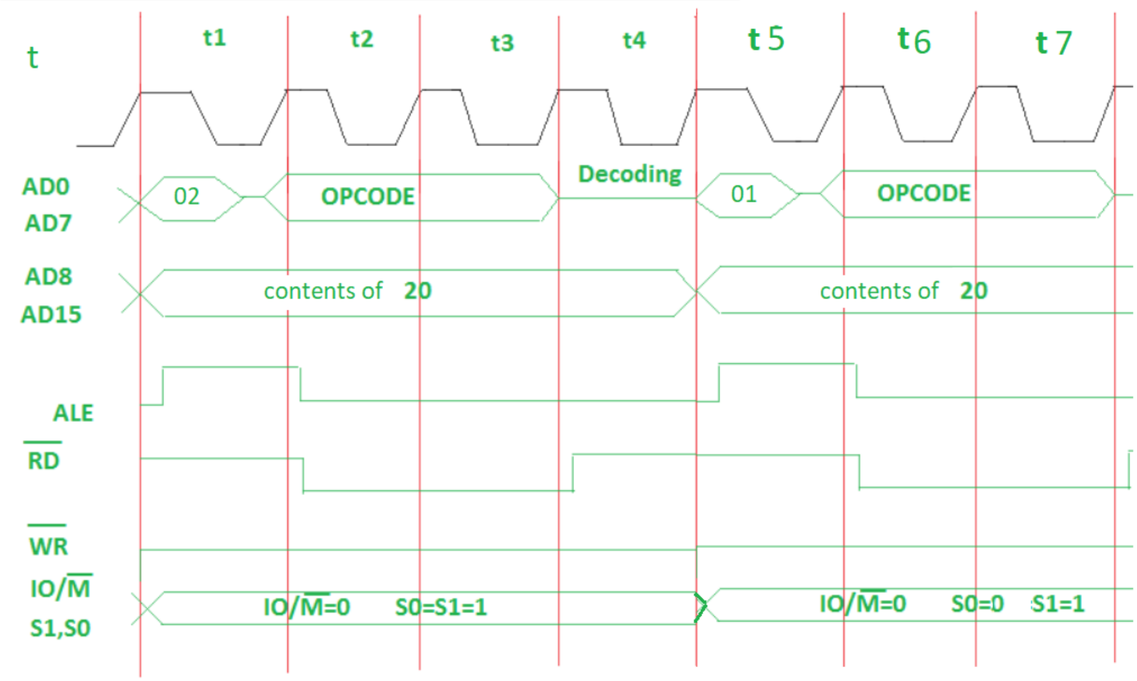
Algorithm –

Here, opcode is ‘MVI A’ and data is 46. Assume the memory address of the opcode and the data. For example:

MVI A, 46

2000: Opcode

2001: 46



In Opcode fetch (t1-t4 T states) –

00 – lower bit of address where opcode is stored, i.e., 00
20 – higher bit of address where opcode is stored, i.e., 20.
ALE – Provides signal for multiplexed address and data bus. Only in t1 it used as address bus to fetch lower bit of address otherwise it will be used as data bus.
RD (low active) – Signal is 1 in t1, t2 & t4, no data is read by microprocessor. Signal is 0 in t3, data is read by microprocessor.
WR (low active) – Signal is 1 throughout, no data is written by microprocessor.
IO/M (low active), S0 and S1 – Signal is 1 in throughout, operation is performing on input/output.

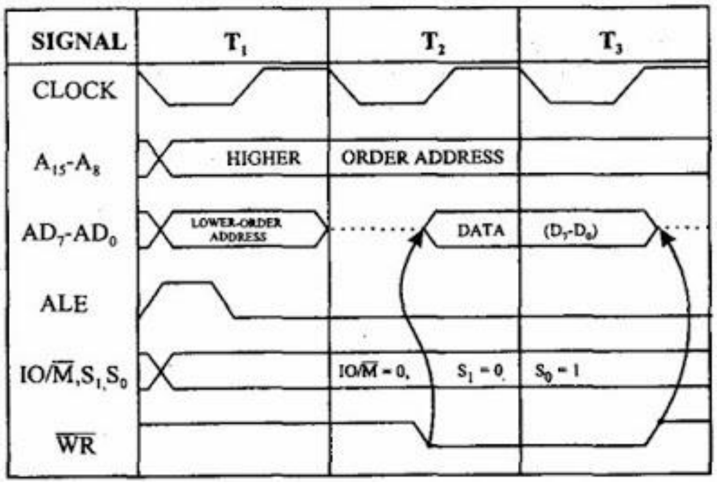
In Memory read (t5-t7 T states) –

00 – lower bit of address where opcode is stored, i.e, 01
20 – higher bit of address where opcode is stored, i.e, 20.
ALE – Provides signal for multiplexed address and data bus. Only in t5 it used as address bus to fetch lower bit of address otherwise it will be used as data bus.
RD (low active) – Signal is 1 in t1, t2 & t4, no data is read by microprocessor. Signal is 0 in t3, data is read by microprocessor.
WR (low active) – Signal is 1 throughout, no data is written by microprocessor.
IO/M (low active) and S1 – Signal is 1 in throughout, operation is performing on input/output.
S0 – Signal is 0 throughout, operation is performing on memory.

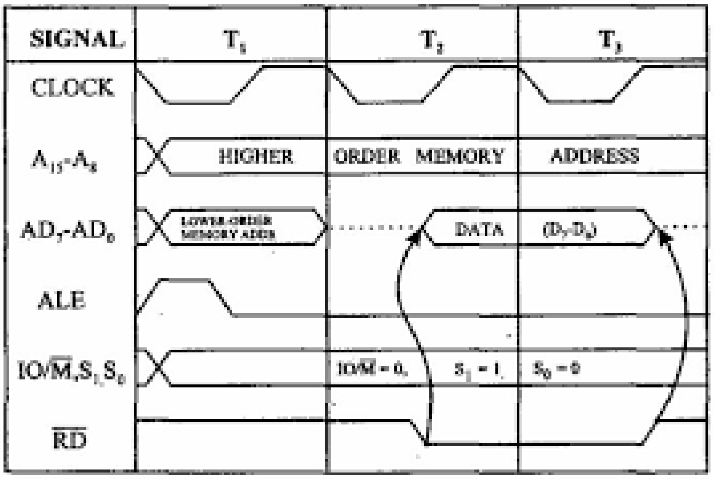
2(d)

MEMORY ADDRESS	**MNEMONICS**	**COMMENTS**
2503	LXI D 2001	;D <= 20, E <= 01
2503	LXI H 3001	;H <= 20, L <= 01
2506	MVI C 04	;C <= 04
2508	MOV B, M	;B <= M[H-L]
2509	LDAX D	;A <= M[D-E]
250A	MOV M, A	;M[H-L] <= A
250B	MOV A, B	;A <= B
250C	STAX D	;M[D-E] <= A
250D	INX H	;[H-L] <= [H-L] + 1
250E	INX D	;[D-E] <= [D-E] + 1
250F	DCR C	;C <= C – 1
2510	JNZ 2508	;JUMP TO 2508 IF C NOT EQUAL TO 0
2513	HLT	;STOP THE PROGRAM

Q3 (a) **Memory Read**



Memory Write



3(b) Types of addressing modes –
In 8085 microprocessor there are 5 types of addressing modes:

Immediate Addressing Mode –
In immediate addressing mode the source operand is always data. If the data is 8-bit, then the instruction will be of 2 bytes, if the data is of 16-bit then the instruction will be of 3 bytes.
Examples:
MVI B 45 (move the data 45H immediately to register B)
LXI H 3050 (load the H-L pair with the operand 3050H immediately)

Register Addressing Mode –
In register addressing mode, the data to be operated is available inside the register(s) and register(s) is (are) operands. Therefore the operation is performed within various registers of the microprocessor.
Examples:
MOV A, B (move the contents of register B to register A)
ADD B (add contents of registers A and B and store the result in register A)
INR A (increment the contents of register A by one)

Direct Addressing Mode –
In direct addressing mode, the data to be operated is available inside a memory location and that memory location is directly specified as an operand. The operand is directly available in the instruction itself.
Examples:
LDA 2050 (load the contents of memory location into accumulator A)
LHLD address (load contents of 16-bit memory location into H-L register pair)

Register Indirect Addressing Mode –
IN register indirect addressing mode, the data to be operated is available inside a memory location and that memory location is indirectly specified b a register pair.
Examples:
MOV A, M (move the contents of the memory location pointed by the H-L pair to the accumulator)
LDAX B (move contains of B-C register to the accumulator)

Implied/Implicit Addressing Mode –
In implied/implicit addressing mode the operand is hidden and the data to be operated is available in the instruction itself.
Examples:
CMA (finds and stores the 1’s complement of the contains of accumulator A in A)
RRC (rotate accumulator A right by one bit)
RLC (rotate accumulator A left by one bit)

Q4(a) Interrupts in 8085 microprocessor

When microprocessor receives any interrupt signal from peripheral(s) which are requesting its services, it stops its current execution and program control is transferred to a sub-routine by generating CALL signal and after executing sub-routine by generating RET signal again program control is transferred to main program from where it had stopped.

When microprocessor receives interrupt signals, it sends an acknowledgement (INTA) to the peripheral which is requesting for its service.

Interrupts can be classified into various categories based on different parameters:

Hardware and Software Interrupts –

When microprocessors receive interrupt signals through pins (hardware) of microprocessor, they are known as Hardware Interrupts. There are 5 Hardware Interrupts in 8085 microprocessor. They are – INTR, RST 7.5, RST 6.5, RST 5.5, TRAP

Software Interrupts are those which are inserted in between the program which means these are mnemonics of microprocessor. There are 8 software interrupts in 8085 microprocessor. They are – RST 0, RST 1, RST 2, RST 3, RST 4, RST 5, RST 6, RST 7.

Vectored and Non-Vectored Interrupts –

Vectored Interrupts are those which have fixed vector address (starting address of sub-routine) and after executing these, program control is transferred to that address.

Vector Addresses are calculated by the formula 8 * TYPE

INTERRUPT	VECTOR ADDRESS
TRAP (RST 4.5)	24 H
RST 5.5	2C H
RST 6.5	34 H
RST 7.5	3C H

For Software interrupts vector addresses are given by:

INTERRUPT	VECTOR ADDRESS
RST 0	00 H
RST 1	08 H
RST 2	10 H
RST 3	18 H
RST 4	20 H
RST 5	28 H
RST 6	30 H
RST 7	38 H

Non-Vectored Interrupts are those in which vector address is not predefined. The interrupting device gives the address of sub-routine for these interrupts. INTR is the only non-vectored interrupt in 8085 microprocessor.

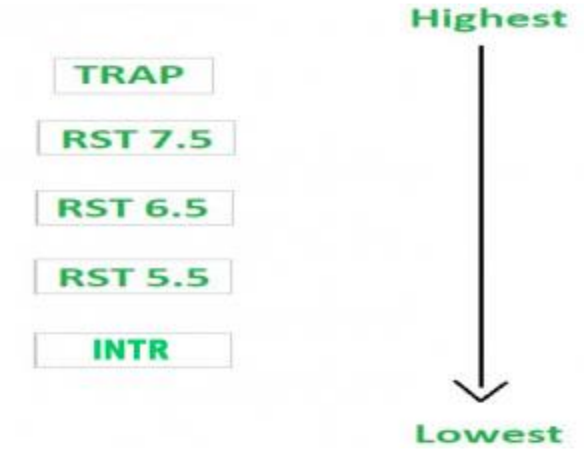
Maskable and Non-Maskable Interrupts –

Maskable Interrupts are those which can be disabled or ignored by the microprocessor. These interrupts are either edge-triggered or level-triggered, so they can be disabled. INTR, RST 7.5, RST 6.5, RST 5.5 are maskable interrupts in 8085 microprocessor.

Non-Maskable Interrupts are those which cannot be disabled or ignored by microprocessor. TRAP is a non-maskable interrupt. It consists of both level as well as edge triggering and is used in critical power failure conditions.

Priority of Interrupts –

When microprocessor receives multiple interrupt requests simultaneously, it will execute the interrupt service request (ISR) according to the priority of the interrupts.



Instruction for Interrupts –

Enable Interrupt (EI) – The interrupt enable flip-flop is set and all interrupts are enabled following the execution of next instruction followed by EI. No flags are affected. After a system reset, the interrupt enable flip-flop is reset, thus disabling the interrupts. This instruction is necessary to enable the interrupts again (except TRAP).
Disable Interrupt (DI) – This instruction is used to reset the value of enable flip-flop hence disabling all the interrupts. No flags are affected by this instruction.
Set Interrupt Mask (SIM) – It is used to implement the hardware interrupts (RST 7.5, RST 6.5, RST 5.5) by setting various bits to form masks or generate output data via the Serial Output Data (SOD) line. First the required value is loaded in accumulator then SIM will take the bit pattern from it.

4(b)
Classification of Instruction Set
- Data Transfer Instruction
- Arithmetic Instructions
- Logical Instructions
- Branching Instructions
- Control Instructions

Data Transfer Instruction

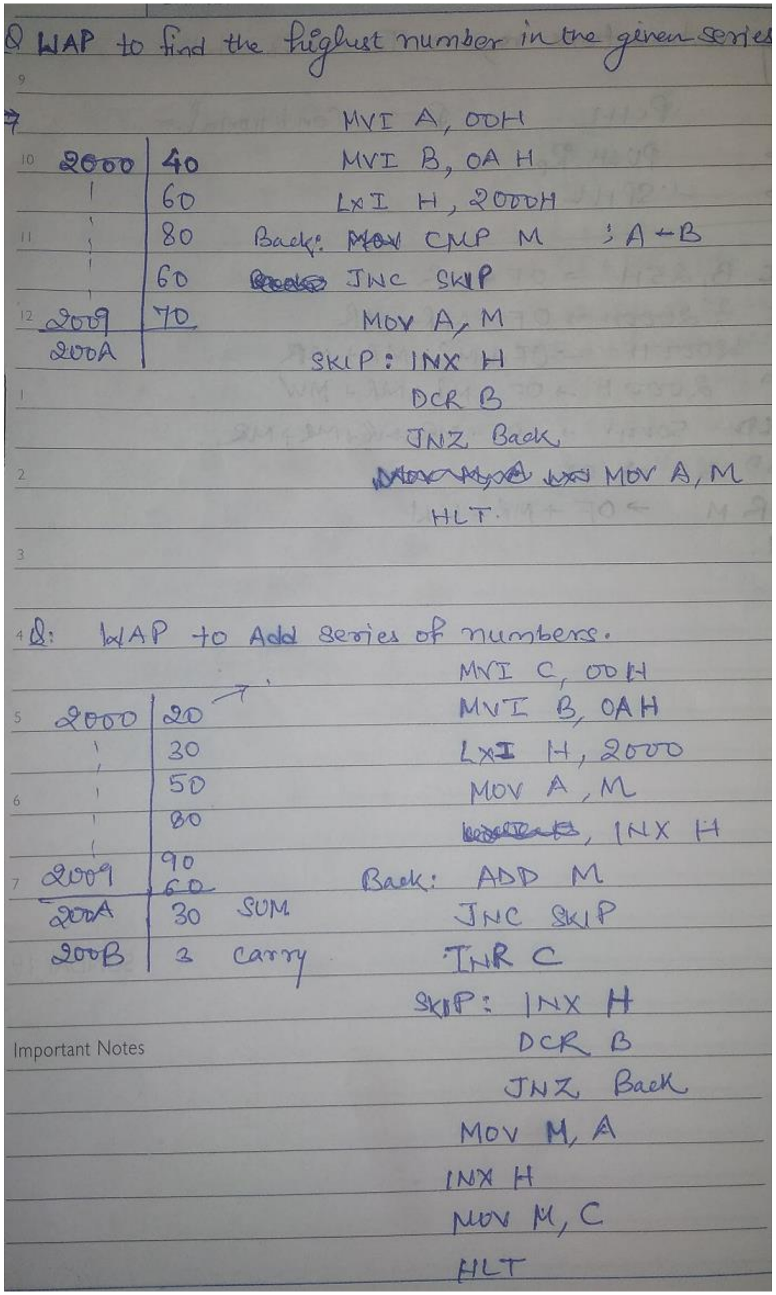
Opcode	Operand	Meaning	Explanation
MOV	Rd, Sc M, Sc Dt, M	Copy from the source (Sc) to the destination (Dt)	This instruction copies the contents of the source register into the destination register without any alteration. **Example** − MOV H, L

Opcode	Operand	Meaning	Explanation
MVI	Rd, data M, data	Move immediate 8-bit	The 8-bit data is stored in the destination register or memory. **Example** − MVI B, 55H
LDA	16-bit address	Load the Accumulator	The contents of a memory location, specified by a 16-bit address in the operand, are copied to the Accumulator. **Example** − LDA 2034H
LDAX	B/D Reg. pair	Load the Accumulator indirect	The contents of the designated register pair point to a memory location. This instruction copies the contents of that memory location into the Accumulator. **Example** − LDAX B
LXI	Reg. pair, 16-bit data	Load the register pair immediate	The instruction loads 16-bit data in the register pair designated in the register or the memory. **Example** − LXI B, 3225H
LHLD	16-bit address	Load H and L registers direct	The instruction copies the contents of the memory location pointed out by the address into register L and copies the contents of the next memory location into register H. **Example** − LHLD 3225H
STA	16-bit address	16-bit address	The contents of the Accumulator are copied into the memory location specified by the operand. This is a 3-Byte instruction, the second Byte specifies the low-order address and the third Byte specifies the high-order address. **Example** − STA AB00H
STAX	16-bit address	Store the Accumulator indirect	The contents of the Accumulator are copied into the memory location specified by the contents of the operand. **Example** − STAX 4050H
SHLD	16-bit address	Store H and L registers direct	The contents of register L are stored in the memory location specified by the 16-bit address in the operand and the contents of H register are stored into the next memory location by incrementing the

Opcode	Operand	Meaning	Explanation
			operand. This is a 3-Byte instruction, the second Byte specifies the low-order address and the third Byte specifies the high-order address. **Example** − SHLD 3225H
XCHG	None	Exchange H and L with D and E	The contents of register H are exchanged with the contents of register D, and the contents of register L are exchanged with the contents of register E. **Example** − XCHG
SPHL	None	Copy H and L registers to the stack pointer	The instruction loads the contents of the H and L registers into the stack pointer register. The contents of the H register provide the high-order address and the contents of the L register provide the low-order address. **Example** − SPHL
XTHL	None	Exchange H and L with top of stack	The contents of the L register are exchanged with the stack location pointed out by the contents of the stack pointer register. The contents of the H register are exchanged with the next stack location (SP+1). **Example** − XTHL
PUSH	Reg. pair	Push the register pair onto the stack	The contents of the register pair designated in the operand are copied onto the stack in the following sequence. The stack pointer register is decremented and the contents of the high order register (B, D, H, A) are copied into that location. The stack pointer register is decremented again and the contents of the low-order register (C, E, L, flags) are copied to that location. **Example** − PUSH B
POP	Reg. pair	Pop off stack to the register pair	The contents of the memory location pointed out by the stack pointer register are copied to the low-order register (C, E, L, status flags) of the operand. The stack pointer is incremented by 1 and the contents of that memory location are copied to the high-order register (B, D, H, A) of the operand.

Opcode	Operand	Meaning	Explanation
			The stack pointer register is again incremented by 1. **Example** − POP B
OUT	8-bit port address	Output the data from the Accumulator to a port with 8 bit address	The contents of the Accumulator are copied into the I/O port specified by the operand. **Example** − OUT 01H
IN	8-bit port address	Input data to Accumulator from a port with 8-bit address	The contents of the input port designated in the operand are read and loaded into the Accumulator. **Example** − IN 04H

5(a)



Q WAP to find the highest number in the given series

```
                  MVI A, 00H
2000 | 40         MVI B, 0A H
     | 60         LXI H, 2000H
     | 80   Back: ~~MOV~~ CMP M    ; A←B
     | 60         JNC SKIP
2009 | 70         MOV A, M
200A |      SKIP: INX H
                  DCR B
                  JNZ Back
                  MOV A, M
                  HLT.
```

Q: WAP to Add series of numbers.

```
                        MVI C, 00H
2000 | 20               MVI B, 0AH
     | 30               LXI H, 2000
     | 50               MOV A, M
     | 80               INX H
2009 | 90         Back: ADD M
     | 60
200A | 30  SUM          JNC SKIP
200B | 3   carry        INR C
                  SKIP: INX H
                        DCR B
                        JNZ Back
                        MOV M, A
                        INX H
                        MOV M, C
                        HLT
```

Important Notes

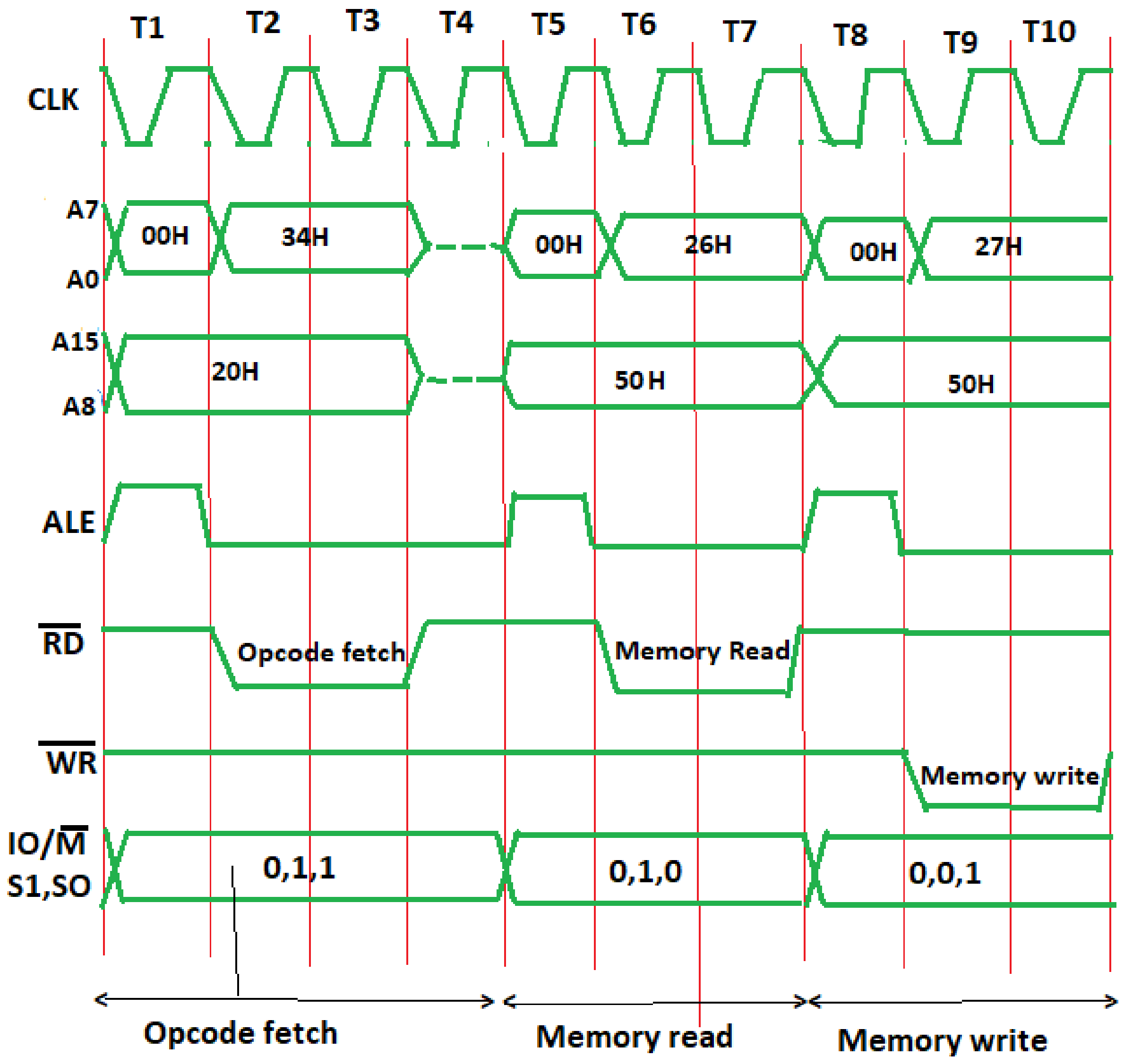
5(b)
The content present in the designated register/memory location (M) is incremented by 1 and the result is stored in the same place. If the operand is a memory location, it is specified by the contents of HL pair.

INR M
Opcode: INR
Operand: M
M is the memory location (say 5000H) and suppose the data present at M (or 5000H) is 26H, which is need to be incremented by 1. Hex code- 34H

Algorithm –

The instruction INR M is of 1 byte; therefore the complete instruction will be stored in a single memory address.
For example:

2000: INR M
The opcode fetch will be same as for other instructions in first 4 T states.
Only the Memory read and Memory Write need to be added in the successive T states.
For the opcode fetch the IO/M (low active) = 0, S1 = 1 and S0 = 1.
For the memory read the IO/M (low active) = 0, S1 = 1 and S0 = 0. Also, only 3 T states will be required.
For the memory write the IO/M (low active) = 0, S1 = 0 and S0 = 1 and 3 T states will be required.


T1
T2
T3
T4
T5
T6
T7
T8
T9
T10
CLK
A7
A0
00H
34H
00H
26H
00H
27H
A15
A8
20H
50H
50H
ALE
$\overline{RD}$
Opcode fetch
Memory Read
$\overline{WR}$
Memory write
IO/$\overline{M}$
S1,SO
0,1,1
0,1,0
0,0,1
Opcode fetch
Memory read
Memory write