



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

Subject Code: 17431

Subject Name: Microprocessor & Programming

**Important Instructions to examiners:**

- 1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
- 2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
- 3) The language errors such as grammatical, spelling errors should not be given more importance (Not applicable for subject English and Communication Skills).
- 4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgment on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.

1. Attempt any **FIVE** of the following:

Marks 20

- a) Describe how 20 bit physical address is generated in 8086 microprocessor. Give one example.

(Description – 2 Marks , Example – 2 Marks)

**Ans: Formation of a physical address:-** Segment registers carry 16 bit data, which is also known as base address. BIU attaches 0 as LSB of the base address. So now this address becomes 20-bit address. Any base/pointer or index register carry 16 bit offset. Offset address is added into 20-bit base address which finally forms 20 bit physical address of memory location.

**Example:-** Assume DS= 2632H, SI=4567H

DS : 26320H .....0 added by BIU(or Hardwired 0)  
+ SI: 4567H

-----  
2A887H

( OR Any Same Type of Example can be considered)



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION  
Model Answer

Subject Code: 17431

Subject Name: Microprocessor & Programming

---

b) Describe various addressing mode used in 8086 instructions with example.

(Any 4 addressing modes : ½ Mark explanation, ½ Mark one example of each)

Ans: Different addressing modes of 8086 :

**1. Immediate :** In this addressing mode, immediate data is a part of instruction, and appears in the form of successive byte or word.

E.g.: *MOV AX, 0050H*

**2. Direct :** In the direct addressing mode, a 16 bit address (offset) is directly specified in the instruction as a part of it.

E.g.: *MOV AX, [1 0 0 0 H]*

**3. Register :** In register addressing mode, the data is stored in a register and it is referred using the particular register. All the registers except IP may be used in this mode.

E.g.: 1) *MOV AX, BX* 2) *ROR AL, CL* 3) *AND AL, BL*

**4. Register Indirect:** In this addressing mode, the address of the memory location which contains data or operand is determined in an indirect way using offset registers. The offset address of data is in either *BX* or *SI* or *DI* register. The default segment register is either *DS* or *ES*.

e.g. *MOV AX, [BX]*

**5. Indexed :** In this addressing mode offset of the operand is stored in one of the index register. *DS* and *ES* are the default segments for index registers *SI* and *DI* respectively

e.g. *MOV AX, [SI]*

**6. Register Relative :** In this addressing mode, the data is available at an effective address formed by adding an 8-bit or 16-bit displacement with the content of any one of the registers *BX*, *BP*, *SI* and *DI* in the default either *DS* or *ES* segment.

e.g. *MOV AX, 50H[BX]*

**7. Based Indexed:** In this addressing mode, the effective address of the data is formed by adding the content of a base register (any one of *BX* or *BP*) to the content of an index register (any one of *SI* or *DI*). The default segment register may be *ES* or *DS*.

e.g. *MOV AX, [BX][SI]*



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

**8. Relative Based Indexed :** The effective address is formed by adding an 8-bit or 16-bit displacement with the sum of contents of any one of the base register (*BX* or *BP*) and any one of the index registers in a default segment.

e.g. *MOV AX, 50H[BX][SI]*

**9. Implied addressing mode:**

No address is required because the address or the operand is implied in the instruction itself.

E.g *NOP,STC,CLI,CLD,STD*

**c) List all 16 bit registers in 8086 and write their functions.**

(List : 2 Marks , Correct function of any 4: ½ Mark each)

**Ans:** EU contains 8 general purpose registers : AX, BX, CX, DX, SP, BP, SI, DI, DS, CS, SS, ES, IP and flag register.

**AX, BX, CX, DX :** used as eight 8-bit registers i.e AL, AH, BL, BH, CL, CH, DL, DH or 16-bit register. AL functions as 8-bit accumulator and AX functions as 16-bit accumulator.

**CX :** used as counter register. **BX :** used as pointer register. **DX :** used for I/O addressing.

**SP , BP :** used as pointer register , SP holds 16-bit offset within stack segment and BP contains offset within the data segment

**SI, DI :** The register SI is used to store the offset of source data in data segment. The register DI is used to store the offset of destination in data or extra segment.

**DS, CS, SS and ES :** These are used for Data, Code, Stack and Extra Data ( Strings) respectively.

**IP:** is used as an instruction pointer which holds the address of the next instruction to be executed by the microprocessor.

**Flag register** is used to hold the status of arithmetic and logic operations along with control flags.

**d) State the function of the following pins of 8086**

(i) **NMI**

(ii)  $\overline{TEST}$

(iii)  $\overline{DEN}$

(iv)  $\overline{MN/MX}$

(Correct one function : 1 Mark each)



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

**(i) NMI**

An edge triggered signal on this pin causes 8086 to interrupt the program it is executing and execute Interrupt service Procedure corresponding to Type-2 interrupt.

NMI is Non-maskable by software.

**(ii)  $\overline{TEST}$**

- Used to test the status of math co-processor 8087.
- If  $\overline{TEST}$  signal goes low, execution will continue, else processor remains in an idle state.

**(iii)  $\overline{DEN}$**

- This is active low signal, to indicate availability of valid data over AD0-AD15.
- Used to enable transceivers(bi-directional buffers) 8286 or 74LS245 to separate data from multiplexed address/data signal .

**(iv)  $MN/\overline{MX}$**

- This signal indicates operating mode of 8086, minimum or maximum.
- When this pin connected to
  - 1) Vcc, the processor operates in minimum mode,
  - 2) Ground , processor operates in maximum mode.

**e) Describe memory segmentation in 8086 microprocessor and list it's four advantages.**

**Ans : (Description: 2 Marks, Any 4 Advantages : ½ Mark each)**

**Memory Segmentation:** The memory in 8086 based system is organized as segmented memory. 8086 can access 1Mbyte memory which is divided into number of logical segments. Each segment is 64KB in size and addressed by one of the segment register. The 4 segment register in BIU hold the 16-bit starting address of 4 segments. CS holds program instruction code. Stack segment stores interrupt & subroutine address. Data segment stores data for program. Extra segment is used for string data.

**Advantages of segmentation**

- 1) With the use of segmentation the instruction and data is never overlapped.
- 2) The major advantage of segmentation is Dynamic relocatability of program which means that a program can easily be transferred from one code memory segment to another code memory segment without changing the effective address.



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

- 3) Segmentation can be used in multi-user time shared system.
- 4) Segmentation allows two processes to share data.
- 5) Segmentation allows you to extend the addressability of a processor i.e., address up to 1MB although the actual addresses to be handled are of 16 bit size.
- 6) Programs and data can be stored separately from each other in segmentation.

**f) Identify addressing modes in following instructions:**

- (i) **MOV AX, 2050H**
- (ii) **STC**
- (iii) **MOV AL, DS:[SI]**
- (iv) **INC BX**

**Ans: (Correct Addressing Mode 1 Mark each)**

- (i) MOV AX,2050H -> Immediate Addressing mode
- (ii) STC -> Implied addressing mode
- (iii) MOV AL,DS:[SI] -> Indexed addressing mode
- (iv) INC BX -> Register addressing mode

**g) Write difference between the following instructions JNC 2000 H and JMP 2000 H**

**Ans: (Any 2 Correct Difference: 2 Mark each)**

<b>JNC 2000H</b>	<b>JMP 2000H</b>
Conditional jump instruction	Unconditional jump instruction
JNC Causes short jump , after comparison if the condition is true, based on the current contents of the status flag as carry flag	JMP Causes IP to be modified so that the next instruction is fetched from the location specified in code segment
Allows Short Jump to 2000H if Carry flag is set to 0.	Allows to jump to location 2000H without any flag checking.



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

2. Attempt any **TWO** of the following: Marks 16
- a) Write algorithm and assembly language programme for 8086 to find largest number among block of data. Assume block size =15.

**Ans: (Algorithm : 4 Marks , Correct Program : 4 Marks)**  
**Any other program logic or 16 bit data may be considered**

- 1) Initialize data segment, byte counter and memory pointer to read members from array.
- 2) Read no. from array.
- 3) Increment memory pointer to read next number.
- 4) Decrement byte counter
- 5) Compare two numbers If number > next number then perform step no. 7
- 6) Replace number with next number which is largest
- 7) Increment memory pointer to read next number from array.
- 8) Decrement counter by 1
- 9) If byte counter is not equal to 0 then perform step No. 5
- 10) Store largest number.

**DATA SEGMENT**

```
ARRAY DB 15H,45H,08H,78H,56H,23H,02H,48H,10H,41H,55H,24H,35H,32H,29H
LARGEST DB 00H
```

**DATA ENDS**

**CODE SEGMENT**

```
START:ASSUME CS:CODE,DS:DATA
```

```
MOV DX,DATA
MOV DS,DX
MOV CX,0DH
MOV SI,OFFSET ARRAY
MOV AL,[SI]
UP: INC SI
CMP AL,[SI]
JNC NEXT
MOV AL,[SI]
NEXT: DEC CX
JNZ UP
MOV LARGEST,AL
MOV AX,4C00H
INT 21H
```



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

**CODE ENDS**  
**END START**

**b) Define MACRO, Give an example indicating how a MACRO can be used in 8086 ALP**

**Ans: (Definition or Syntax : 2 Marks , Macro Example :3 Marks , Usage in the program:3M)**

**Macro**

- Small sequence of the codes of the same pattern are repeated frequently at different places which perform the same operation on the different data of same data type, such repeated code can be written separately called as Macro.
- When assembler encounters a Macro name later in the source code, the block of code associated with the Macro name is substituted or expanded at the point of call, known as macro expansion.
- Macro called as open subroutine.

**(OR)**

**Macro definition or (Macro directive):**

Syntax:

**Macro\_name MACRO[arg1,arg2,.....argN)**

.....

**Endm**

**Example:**

**MyMacro MACRO p1, p2, p3 ; macro definition with arguments**

**MOV AX, p1**

**MOV BX, p2**

**MOV CX, p3**

**ENDM ;indicates end of macro.**

**data segment**

**data ends**

**code segment**

**start: assume cs:code,ds:data**

**mov ax,data**

**mov ds,ax**

**MyMacro 1, 2, 3 ; macro call**

**MyMacro 4, 5, DX**

**mov ah,4ch**

**int 21h**



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION  
Model Answer

Subject Code: 17431

Subject Name: Microprocessor & Programming

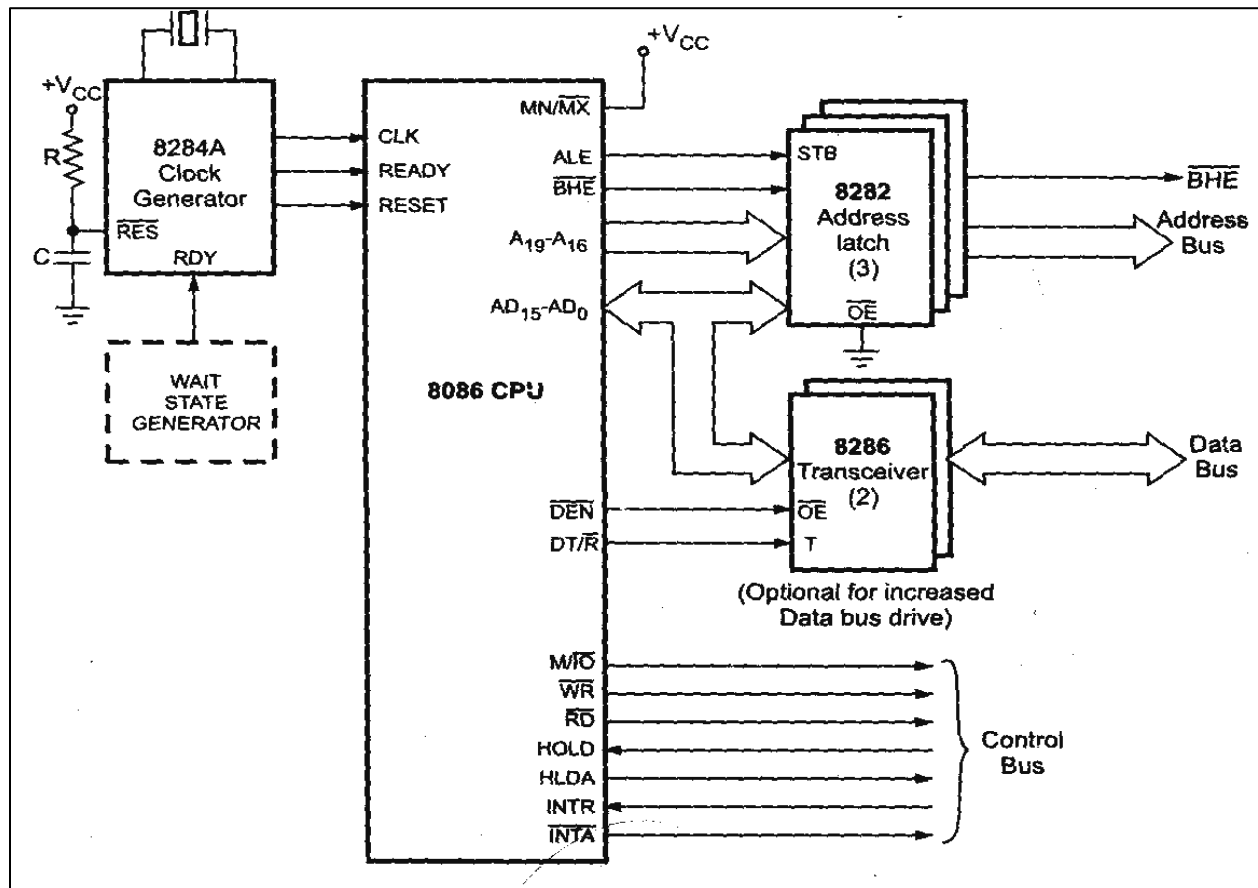
code ends  
end start

( OR Any Same Type of Example can be considered)

- c) Explain the minimum mode configuration of 8086 microprocessor. Write comparison between 8086 in minimum mode and maximum mode.

(Explanation: 4 Marks [Diagram not necessary], Comparison - Any 4 points: 1 Mark each)

Ans:



- When MN/MX pin is in logic 1, the 8086 microprocessor operates in minimum mode system.





**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

- In this mode, the microprocessor chip itself gives out all the control signals.
- This is a single processor mode.
- The remaining components in the system are latches, transceivers, clock generator, memory or I/O devices.
- This system has three address latches(8282) and two octal data buffers(8286) for the complete 20-bit address and 16 bit data Separation.
- The latches are used for separating the valid address from the multiplexed address/data signals and the controlled by the ALE signal generated by 8086.
- Transceivers are the bi-directional buffers. They are required to separate the valid data from the time multiplexed address/data signal. This is controlled by two signals, DEN & DT/ $\bar{R}$ .
- DT/ $\bar{R}$  indicates that the direction of data, ie. from or to the microprocessor.
- $\overline{DEN}$  signal indicates the valid data is available on the data bus.
- This system contains memory for the monitor and users program storage. It also contains I/O devices to communicate with the processor.
- The clock generator in the system is used to generate the clock and to synchronize some external signals with the system clock.

Sr. No.	Minimum mode	Maximum mode
1.	MN/ $\overline{MX}$ pin is connected to V <sub>CC</sub> . i.e. MN/ $\overline{MX}$ = 1.	MN/ $\overline{MX}$ pin is connected to ground. i.e. MN/ $\overline{MX}$ = 0.
2.	Control system M/ $\overline{IO}$ , $\overline{RD}$ , $\overline{WR}$ is available on 8086 directly.	Control system M/ $\overline{IO}$ , $\overline{RD}$ , $\overline{WR}$ is not available directly in 8086.
3.	Single processor in the minimum mode system.	Multiprocessor configuration in maximum mode system.
4.	In this mode, no separate bus controller is required.	Separate bus controller (8288) is required in maximum mode.
5.	Control signals such as $\overline{IOR}$ , $\overline{IOW}$ , $\overline{MEMW}$ , $\overline{MEMR}$ can be generated using control signals M/ $\overline{IO}$ ,	Control signals such as $\overline{MRDC}$ , $\overline{MWTC}$ , $\overline{AMWC}$ , $\overline{IORC}$ , $\overline{IOWC}$ and $\overline{AIOWC}$ are generated by bus controller 8288.



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

	$\overline{RD}$ , $\overline{WR}$ which are available on 8086 directly.	
6.	ALE, $\overline{DEN}$ , $DT/\overline{R}$ and $\overline{INTA}$ signals are directly available.	ALE, $\overline{DEN}$ , $DT/\overline{R}$ and $\overline{INTA}$ signals are not directly available and are generated by bus controller 8288.
7.	HOLD and HLDA signals are available to interface another master in system such as DMA controller.	$\overline{RQ}/\overline{GT0}$ and $\overline{RQ}/\overline{GT1}$ signals are available to interface another master in system such as DMA controller and coprocessor 8087.
8.	Status of the instruction queue is not available.	Status of the instruction queue is available on pins $QS_0$ and $QS_1$ .

**3. Attempt any FOUR of the following:**

**Marks 16**

**a) Describe reentrant procedure with the help of schematic diagram.**

**(Diagram 2 marks, explanation - 2 Marks)**

**Any other example diagram can also be considered.**

In some situation it may happen that Procedure 1 is called from main program  
Procedure 2 is called from procedure 1 and procedure 1 is again called from procedure 2.  
In this situation program execution flow re-enters in the procedure 1. These types of  
procedures are called re-entrant procedures.

A procedure is said to be re-entrant, if it can be interrupted, used and re-entered without losing or writing over anything.

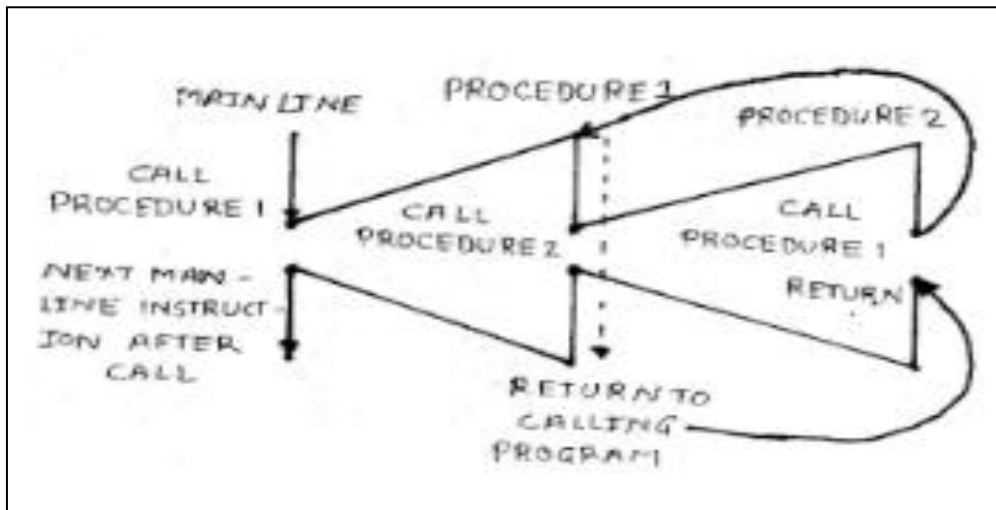


**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

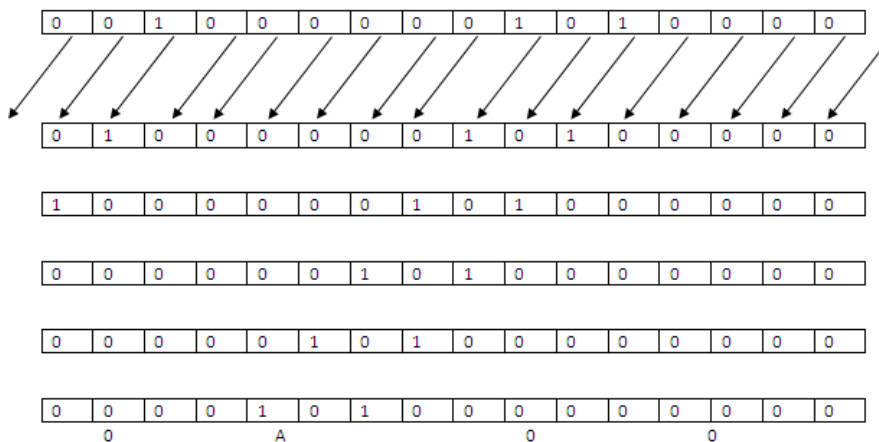


b) What will be content of register BX after execution of instructions?

```
MOV BX, 2050H
MOV CL, 05H
SHL BX, CL
```

(Correct answer : 4 Marks )

For steps 1 M may be considered





**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

Content of BX After execution = **0A00H**

**c) State functions of following assembly language programming tool.**

**(i) Assembler**

**(ii) Linker**

**(Any two functions of each : 2 Marks)**

**i) Assembler**

1. Assembler is a program that translates assembly language program to the correct binary code.
2. It also generates the file called as object file with extension .obj.
3. It also displays syntax errors in the program, if any.
4. It can be also be used to produce list(.lst) and .crf files

**ii) Linker**

- a) It is a programming tool used to convert Object code into executable program.
- b) It combines ,if requested ,more than one separated assembled modules into one executable module such as two or more assembly programs or an assembly language with C program.
- c) It generates .EXE module

**d) Differentiate between following instructions:**

**(i) ROL, RCL**

**(ii) ADD, ADC**

**(iii) MOV, LXI**

**(iv) JMP, JNC**

**( Any one Difference :1Mark)**

**( Note: Third instruction is not a valid 8086 instruction; hence 1Mark may be given for attempting)**



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

**Ans:**

ROL	RCL
Rotate left byte or word	Rotate through carry left byte or word
Syntax: ROL Destination, Count	Syntax: RCL Destination, Count
Can be used to Swap the nibbles	Cannot be used to swap the nibbles.

ADD	ADC
Add byte or word	Add byte or word with carry flag
Syntax : ADD Destination, Source	Syntax : ADC Destination, Source
Destination $\leftarrow$ Source +Destination	Destination $\leftarrow$ Source +Destination + Carry

MOV	LXI
Transfer byte or word from from source to destination	Used in 8085.

<b>JMP</b>	<b>JNC</b>
Unconditional jump instruction	Conditional jump instruction
JMP Causes IP to be modified so that the next instruction is fetched from the location specified in code segment	JNC Causes short jump , after comparison if the condition is true, based on the current contents of the status flag as carry flag
Allows jumping to location without any flag checking.	Allows Short Jump to if Carry flag is set to 0.



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

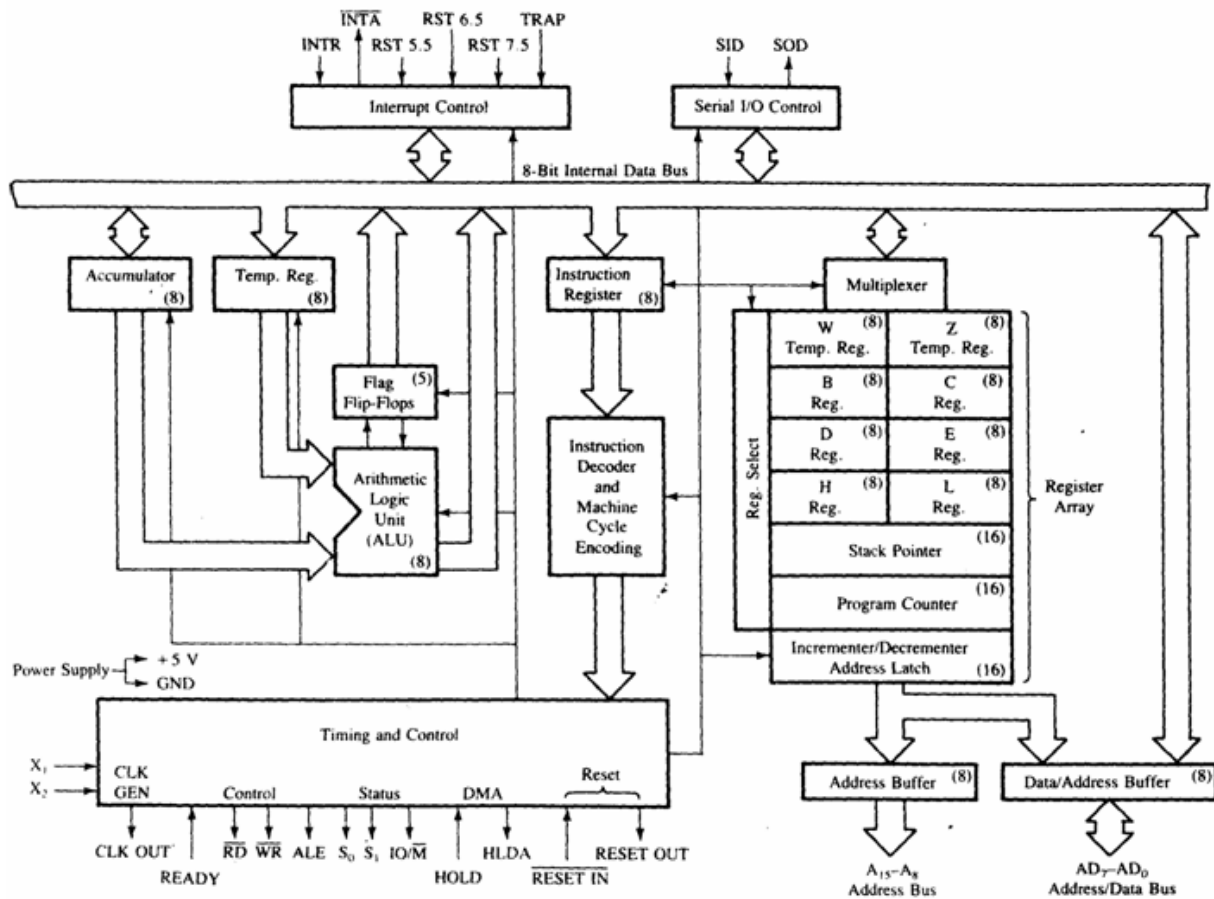
**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

**e) Draw a neat labeled functional block diagram of 8085. State the function of ALU.**

**(Diagram : 3Marks; Any one function 1Mark)**



Functions:

1. ALU performs the arithmetic operations and logical operation such as Add, Subtract, AND, OR, XOR on 8 bit data.
2. It stores the answer in the accumulator.



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

f) Write an ALP to add BCD numbers in an array of 10 numbers. Assume suitable array with BCD numbers. Store the result at the end of array.

(Correct Program : 4M) Any other logic may be considered.

```
DATA SEGMENT
ARRAY DB 10H,02H,03H,04H,03H, 05H, 06H, 10H,02H,03H
DATA ENDS
CODE SEGMENT
START: ASSUME CS: CODE, DS:DATA
MOV AX, DATA
MOV DS, AX
MOV CL, 0AH
MOV AL,00H
MOV SI,OFFSET ARRAY
UP: ADD AL,[SI]
DAA
INC SI
LOOP UP
MOV [SI+1], AL
MOV AX, 4C00H
INT 21H
CODE ENDS
END START
```

4. Attempt any **FOUR** of the following:

**Marks 16**

a) Write an ALP for 8086 to sort the array in ascending order. Draw flowchart. [Assume array of size 10]

( Correct program 2 Marks, flowchart 2 Marks)



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

**DATA SEGMENT**

ARRAY DB 06H,09H,22H,02H,07H,10H,11H,12H,13H,14H

DATA ENDS

**CODE SEGMENT**

ASSUME CS:CODE,DS:DATA

START: MOV AX,DATA

MOV DS,AX

MOV BL,0AH

STEP1: MOV SI, OFFSET ARRAY

MOV CL,09H

STEP2: MOV AL,[SI]

CMP AL,[SI+1]

JC DOWN

XCHG AL, [SI+1]

XCHG AL,[SI]

DOWN: ADD SI, 01

LOOP STEP2

DEC BL

JNZ STEP1

MOV AH,4CH

INT 21H

CODE ENDS

END START



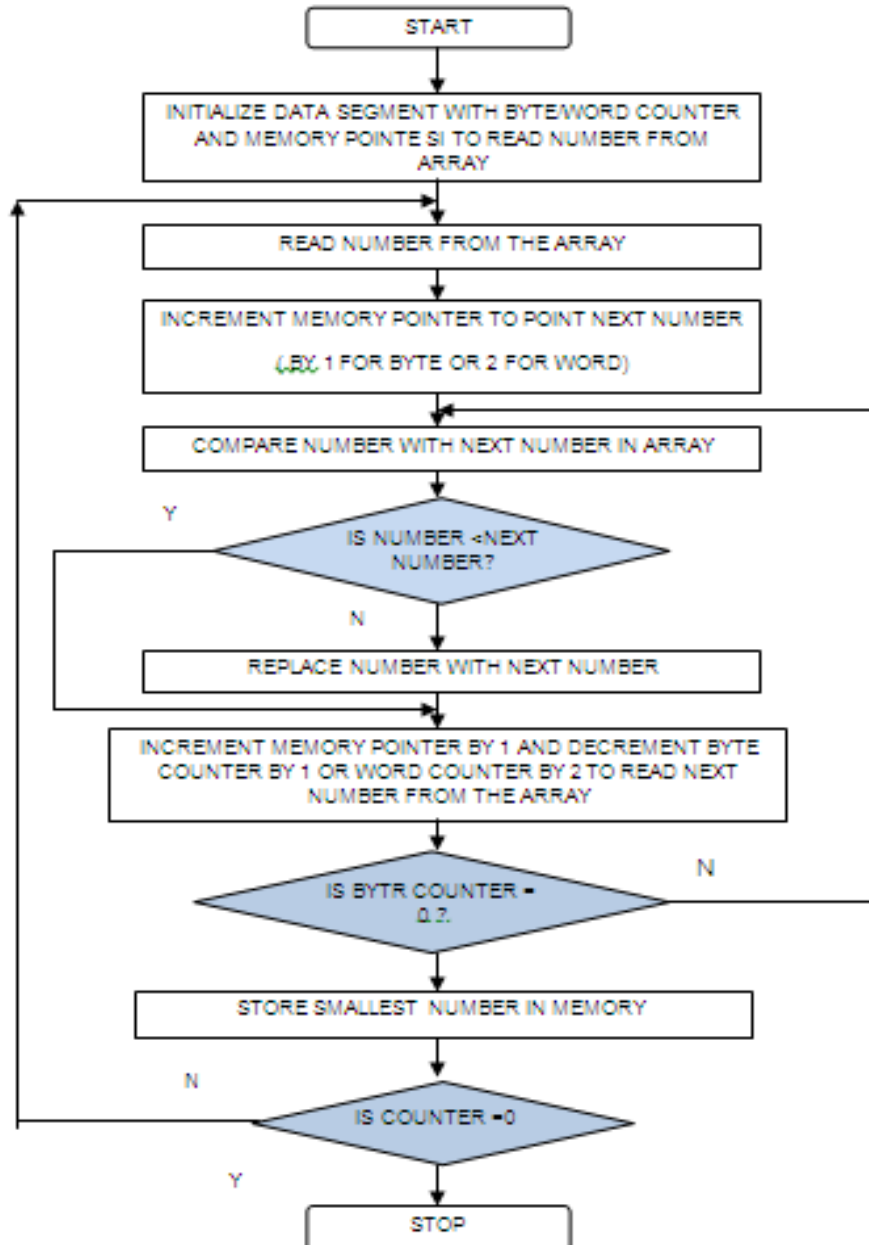


MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

Subject Code: 17431

Subject Name: Microprocessor & Programming





**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

**b) Explain NEAR CALL and FAR CALL procedure.**

**( Any four points 1 Mark each)**

<b>NEAR call</b>	<b>FAR call</b>
A near call is a call to procedure which is in same code segment	A far call is a call to procedure which is in different segment
The contents of CS is not stored	The contents of CS is also stored along with offset.
In near call contents of SP is decremented by 2 and contents of offset address IP is stored	In Far call contents of SP are decremented by 2 and value of CS is loaded. Then SP is again decremented by 2 and IP is loaded.
Example : CALL Delay	Example :CALL FAR PTR Delay

**c) Write the difference between PROCEDURE and MACRO.**

**(Any 4 points – 1Mark each)**

<b>MACRO</b>	<b>PROCEDURE</b>
Macro is a small sequence of code of the same pattern, repeated frequently at different places, which perform the same operation on different data of the same data type.	Procedure is a series of instructions is to be executed several times in a program, and called whenever required.



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

The MACRO code is inserted into the program, wherever MACRO is called, by the assembler.	Program control is transferred to the procedure, when CALL instruction is executed at run time.
Memory required is more, as the code is inserted at each MACRO call	Memory required is less, as the program control is transferred to procedure.
Stack is not required at the MACRO call.	Stack is required at Procedure CALL.
No overhead time required.	Extra overhead time is required for linkage between the calling program and called procedure.
Parameter passed as the part of statement which calls macro.	Parameters passed in registers, memory locations or stack.
RET is not used	RET is required at the end of the procedure
Macro is called using: <Macro_name> [argument list]	Procedure is called using: CALL <Procedure_name>
Directives used: MACRO, ENDM, LOCAL	Directives used: PROC, ENDP, FAR, NEAR
Example: Procedure Name PROC ----- Procedure Statements ----- Procedure Name ENDP.	Example: Macro_name MACRO ----- ----- instructions ----- ENDM



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

Subject Code: 17431

Subject Name: Microprocessor & Programming

d) Write the difference between 8085 and 8086 with respect to

- (i) Register size
- (ii) Address bus size
- (iii) Pipelining
- (iv) Segmented memory

(for each comparison 1 Mark)

Ans:

	8085	8086
<b>Registers size</b>	8 Bit registers	16 Bit registers
<b>Address bus size</b>	16	20
<b>Pipelining</b>	No	Yes (6 byte Instruction Queue)
<b>Segmented memory</b>	No	Yes (code segment, data segment, stack segment, extra segment )

e) Describe various string instructions in brief.

(Any 4 instructions (Operation or syntax) -1 Mark each)

Ans:

1] MOVS/ MOVSB/ MOVSW - Move String byte or word.

Syntax

MOVS destination, source

MOVSB

MOVSW



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

**Operation: ES:[DI]<----- DS:[SI]**

It copies a byte or word a location in data segment to a location in extra segment. The offset of source is pointed by SI and offset of destination is pointed by DI. CX register contain counter and direction flag (DF) will be set or reset to auto increment or auto decrement pointers after one move.

**Example**

LEA SI, Source

LEA DI, destination

CLD

MOV CX, 04H

REP MOVSB

**3] CMPS /CMPSB/CMPSW: Compare string byte or Words.**

**Syntax**

**CMPS destination, source**

CMPSB

CMPSW

**Operation: Flags affected < ----- DS:[SI]- ES:[DI]**

It compares a byte or word in one string with a byte or word in another string. SI holds the offset of source and DI holds offset of destination strings. CX contains counter and DF=0 or 1 to auto increment or auto decrement pointer after comparing one byte/word.

**Example**

LEA SI, Source

LEA DI, destination

CLD

MOV CX, 100

REPE CMPSB



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

**4] SCAS/SCASB/SCASW:** Scan a string byte or word.

**Syntax**

**SCAS/SCASB/SCASW**

**Operation: Flags affected < ----- AL/AX-ES: [DI]**

It compares a byte or word in AL/AX with a byte /word pointed by ES: DI. The string to be scanned must be in the extra segment and pointed by DI. CX contains counter and DF may be 0 or 1.

When the match is found in the string execution stops and ZF=1 otherwise ZF=0 .

**Example**

LEA DI, destination

MOV AL, 0DH

MOV CX, 80H

CLD

REPNE SCASB

**5] LODS/LODSB/LODSW:** Load String byte into AL or Load String word into AX.

**Syntax: LODS/LODSB/LODSW**

**Operation: AL/AX < ----- DS: [SI]**

It copies a byte or word from string pointed by SI in data segment into AL or AX. CX may contain the counter and DF may be either 0 or 1

**Example**

LEA SI, destination

CLD

LODSB



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

**6] STOS/STOSB/STOSW (Store Byte or Word in AL/AX)**

**Syntax STOS/STOSB/STOSW**

**Operation: ES:[DI] < ----- AL/AX**

It copies a byte or word from AL or AX to a memory location pointed by DI in extra segment CX may contain the counter and DF may either set or reset.

**f) Explain pipelining in 8086 microprocessor. How is queuing useful in speeding up the Operation of 8086 microprocessor.**

**( Pipelining 3 Marks, usefulness 1 Mark)**

**Ans:**

- In 8086, pipelining is the technique of overlapping instruction fetch and execution mechanism.
- To speed up program execution, the BIU fetches as many as six instruction bytes ahead of time from memory. The size of instruction prefetch queue in 8086 is 6 bytes.
- While executing one instruction other instruction can be fetched. Thus it avoids the waiting time for execution unit to receive other instruction.
- BIU stores the fetched instructions in a 6 level deep FIFO . The BIU can be fetching instructions bytes while the EU is decoding an instruction or executing an instruction which does not require use of the buses.
- When the EU is ready for its next instruction, it simply reads the instruction from the queue in the BIU.
- This is much faster than sending out an address to the system memory and waiting for memory to send back the next instruction byte or bytes.
- This improves overall speed of the processor.



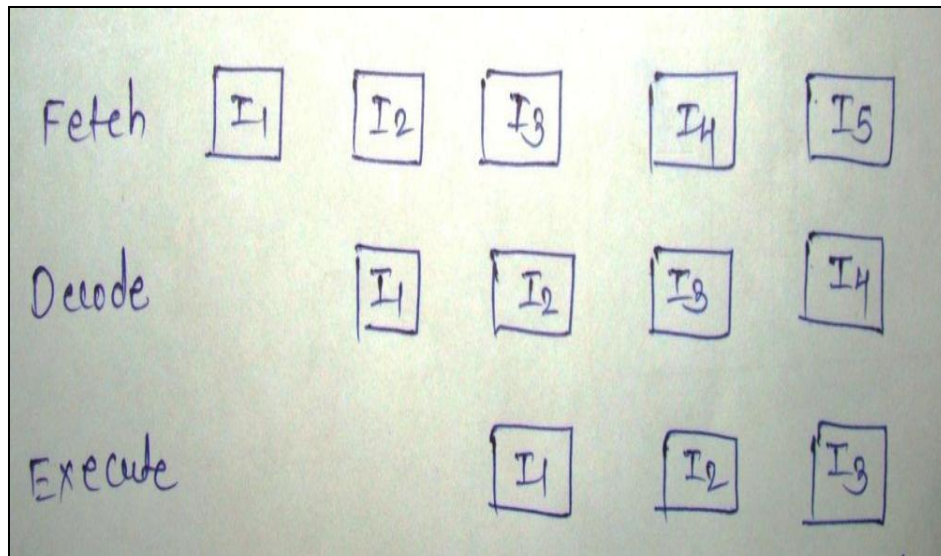
MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

Subject Code: 17431

Subject Name: Microprocessor & Programming

---



5. Attempt any TWO of the following:

Marks 16

- a) Draw architecture of 8086 and label it. Write the function of BIU and EU.  
( Diagram :4 Marks; Any TWO functions of each unit : 2Marks)

Ans:





MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION  
Model Answer

Subject Code: 17431

Subject Name: Microprocessor & Programming

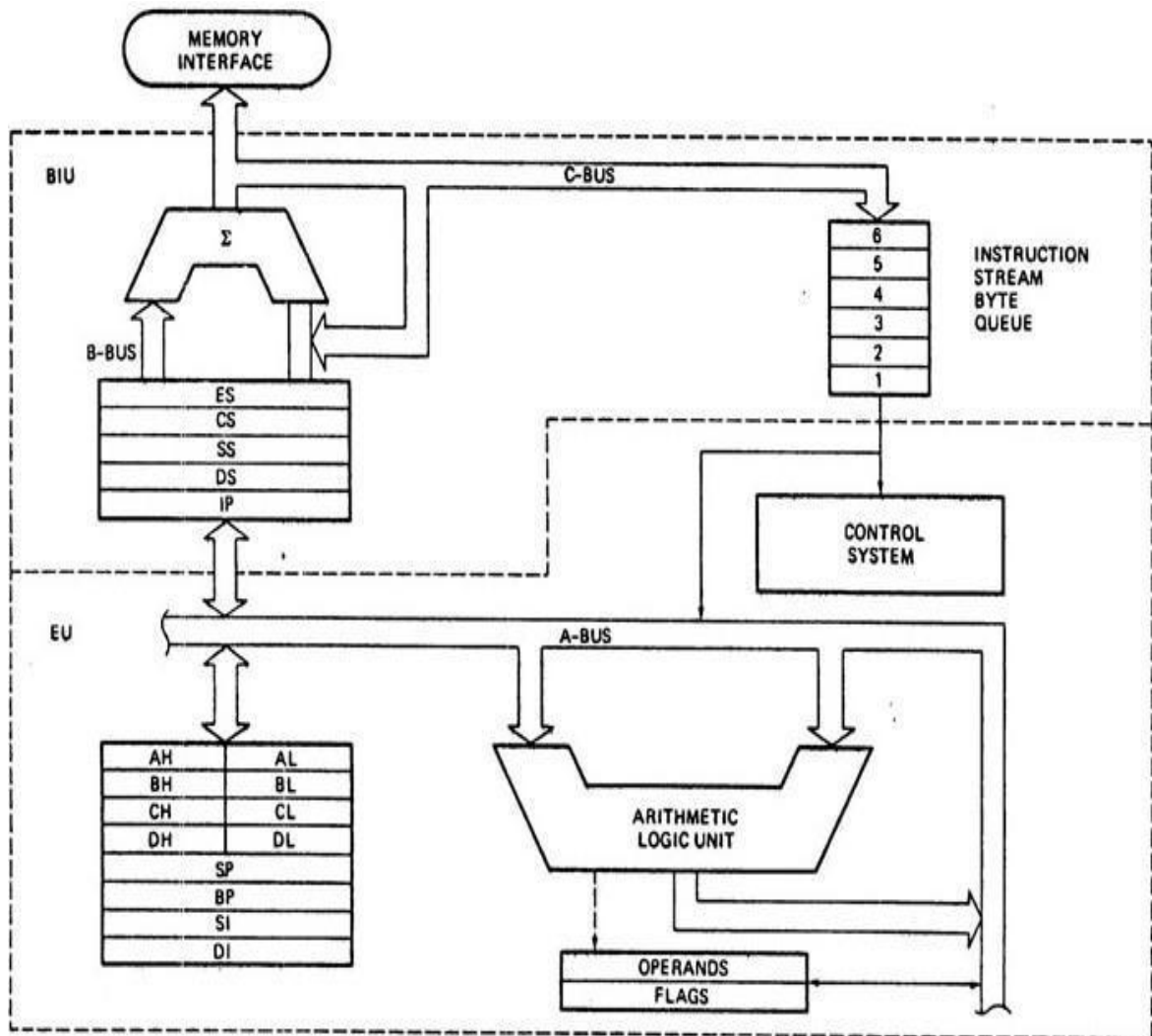


FIG. architecture of 8086



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

**FUNCTIONS OF EXECUTION UNIT:**

- 1.To tell BIU to fetch the instructions or data from memory
- 2.To decode the instructions.
- 3.To generate different internal and external controls signal.
- 4.To execute the instructions.
- 5.To perform Arithmetic and Logic Operations

**FUNCTIONS OF BUS INTERFACE UNIT:**

- 1.Communication with External devices and peripheral including memory via bus.
- 2.Fetch the instruction or data from memory.
- 3.Read data from the port.
- 4.Write the data to memory and port.
- 5.Calculation of physical address for accessing the data to and from memory

**b) Write ALP to concatenate two strings with algorithm**

**String 1 : “Maharashtra board”**

**String 2 : “ of technical Education”**

**(Correct Program:6 Marks; Algorithm :2Marks ) ( Any other logic may be considered)**

**Ans:**

**Algorithm:**

1. Initialize data segment.
2. Initialize memory pointers for source and destination string.
3. Move memory pointer of source string to the end of string.
4. Move memory pointer of destination string to the end of string.
5. Copy characters from destination string to source string.
6. Stop.

**ALP:**

```
.MODEL SMALL
.DATA
    STR_S DB 'Maharashtra board $'
    STR_D DB 'of technical Education $'

.CODE
```



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

```
MOV AX, @DATA
MOV DS, AX

MOV SI, OFFSET STR_S

NEXT:
MOV AL, [SI]
CMP AL, '$'
JE EXIT
INC SI
JMP NEXT

EXIT:
MOV DI, OFFSET STR_D
UP: MOV AL, [DI]
CMP AL, '$'
JE EXIT1
MOV [SI], AL
INC SI
INC DI
JMP UP
EXIT1:
MOV AL, '$'
MOV [SI], AL

MOV AH, 4CH
INT 21H
ENDS
END
```

(OR)

```
DATA SEGMENT
ST1 DB " Maharashtra board$"
LEN1 DB 0
ST2 DB " of technical Education$"
LEN2 DB 0
R DB ?
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA, ES:DATA
START: MOV AX, DATA
MOV DS, AX
MOV ES, AX
MOV SI, OFFSET ST1 ; Length of the first string in LEN1
```



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

```
MOV AL,'$'  
NEXT1: CMP AL,[SI]  
JE EXIT1  
INC LEN1  
INC SI  
JMP NEXT1  
EXIT1: MOV SI, OFFSET ST2           ; Length of the second string in LEN2  
NEXT2: CMP AL,[SI]  
JE EXIT2  
INC LEN2  
INC SI  
JMP NEXT2  
EXIT2: MOV SI, OFFSET ST1           ; copy first string to R  
MOV DI, OFFSET R  
MOV CL, LEN1  
REP MOVSB  
MOV SI, OFFSET ST2           ; Concat second string to R  
MOV CL, LEN2  
REP MOVSB  
MOV AH,4CH  
INT 21H  
CODE ENDS  
  
END START
```

**C) Describe following assembler directives.**

- (i) **DB**
- (ii) **ASSUME**
- (iii) **SEGMENT**
- (iv) **EQU**

*( For each Directive – 2Mark)*

**ANS:**

**i. DB: - Define byte**

This directive is used to define a byte type variable.  
It can be used to single or multiple byte variable.



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

The range of byte store is from 0 to 255 for unsigned number and -128 to + 127 for signed numbers.

**GENERAL FORM:**

NAME\_OF\_VARIABLE DB INITIALIZE\_VALUE(S)

E.g. Num1 DB 36H

**ii) ASSUME :**

Assume directive is used to tell Assembler the name of the logical segment it should use for the specified segment.

When program is loaded, the processor segment register should point to the respective logical segments.

**GENERAL FORM:**

ASSUME Seg\_reg: Seg-name,.....,Seg\_reg:Seg\_name

Where,

ASSUME is a assembler directive.

Seg\_reg is any of the segments register i.e CS, DS, ES, SS.

Seg-name is the name of a user defined segment and must be any valid symbol except reserved keywords.

**Example:-Assume CS: MSBTE\_CODE, DS: MSBTE\_DATA**

**iii) SEGMENT:**

The directive SEGMENT is used to indicate the beginning of the logical segment.

The directive SEGMENT follows the name of the segment.

The directive SEGMENT and ENDS must be enclosed the segment data, code, extra or stack of program.

**GENERAL FORM:**

Segment-name SEGMENT [word/public]

**Examples:**

**My\_data SEGMENT**

.....

Program data definition here

**My\_data ENDS**



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

iv) **EQU: Equate to**

The EQU directive is used to declare the symbols to which some constant value is assigned.

**GENERAL FORM:**

Symbol-name EQU expression

**Examples:**

Num EQU 100

Increment EQU INC

6. Attempt any **FOUR** of the following:

**Marks 16**

- a) Write an ALP to count the number of '1' in a 16 bit number.  
Assume the number to be stored in BX register. Store the result in C<sub>X</sub> register.  
(Correct Program -4 Marks, Any other logic may be considered )

**Ans:**

```
.MODEL SMALL
.DATA
    NUM DW 0008H
    ONES DB 00H

.CODE
MOV AX, @DATA ; initialize data segment
MOV DS, AX

MOV CX, 10H ; initialize rotation counter by 16
MOV BX, NUM ;load number in BX

UP: ROR BX, 1 ; rotate number by 1 bit right
    JNC DN ; if bit not equal to 1 then go to dn
    INC ONES ; else increment ones by one

DN: LOOP UP ;decrement rotation counter by 1 anf if not zero then go to up

MOV CX, ONES ;move result in cx register.

MOV AH, 4CH
INT 21H
```



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

ENDS  
END ; end of program.

**b) Explain maskable and non maskable interrupt used in 8086.**

**(Maskable any two points : 2Marks; non maskable any two points : 2Marks)**

**Ans:**

**Non-Maskable interrupt:**

- 8086 has a non-maskable interrupt input pin (NMI) that has highest priority among the external interrupts.
- The NMI is not maskable internally by software.
- **TRAP** (single step-type1) is an internal interrupt having highest priority amongst all the interrupts except **Divide by Zero** (Type 0) exception.
- The NMI is activated on a positive transition (low to high voltage).
- The NMI pin should remain high for at least two clock cycles and need not synchronized with the clock for being sensed.

**Maskable interrupt:**

- 8086 also provides a INTR pin, that has lower priority as compared to NMI.
- The INTR signal is level triggered and can be masked by resetting the interrupt flag.
- It is internally synchronized with the high transition of the CLK.
- For the INTR signal, to be responded to in the next instruction cycle; it must go high in the last clock cycle of the current instruction or before that.



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

(Autonomous)

(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**

**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

- c) **Explain stack operation. Why PUSH and POP instructions are used before and after CALL sub-routine?**

**( PUSH and POP explanation each -1½ Marks, Reason - 1Mark)**

**Ans: PUSH: push to stack**

This instruction pushes the contents of the specified register/memory location on to the stack. The stack pointer is decremented by 2 after each execution of the instruction, to point the top element of the stack.

The higher byte is pushed first and then the lower byte. Thus out of the two decremented stack addresses the higher byte occupies the higher address and the lower byte occupies the lower address.

**Example: PUSH AX**

PUSH DS

PUSH [5000H] ; content of location 5000H and 5001H in DS are pushed onto the stack.

**POP: pop from stack**

This instruction when executed, loads the specified register/memory location with the contents of the memory location of which the address is formed using current stack segment and stack pointer as usual.

The stack pointer is incremented by 2 to point the current stack top element.

**Example: POP AX**

POP DS

POP [5000H] ;

**Reason:**

**PUSH and POP instruction should be used before and after subroutine call to avoid data loss of all the register of CPU.** Before subroutine call, all the registers' content of the main program may be pushed onto the stack one by one using **PUSH instruction** into stack so that these registers can be used by subroutine. After subroutine execution, **POP instruction** should be used to get the old elements from the stack into the registers.

**Note: Number of PUSH and POP instruction should be equal.**





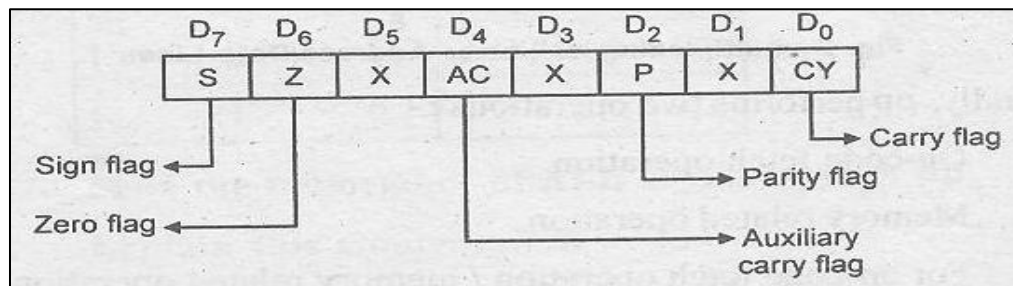
MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

Subject Code: 17431

Subject Name: Microprocessor & Programming

- d) Draw the flag register format of 8085 microprocessor and explain all the flags.  
(Diagram– 2Marks, Explanation– 2Marks)



**Fig: Format of flag register of 8085  $\mu$ p**

**i) Carry flag (CY):**

When  $\mu$ p performs addition/subtraction of 8 bit if the carry/borrow is generated from the MSB, then the carry flag is set (CY=1), otherwise it resets the carry flag (CY=0).

**ii) Auxiliary carry flag (AC)/ Half carry/ Nibble carry:**

When  $\mu$ p performs addition of 8 bit number and if the carry is generated from D3bit, then auxiliary carry flag is set, otherwise it is reset.

**iii) Parity flag (P):**

When  $\mu$ p performs addition or logical operations on 8 bit number and if number of 1's bit in 8 bit result is even number, then it is called as Even parity and parity flag is set (P=1) otherwise it is called as Odd parity and parity flag is reset (P=0).

**iv) Zero Flag(Z):**

When  $\mu$ p performs arithmetic and logical operation of two 8 bit numbers, if the result obtained is zero, then flag is set (Z=1), otherwise it is reset (Z=0).

**v) Sign flag (S):**

When  $\mu$ p performs arithmetic and logical operations on signed numbers and if the MSB of the result is 1, then sign flag is set. i.e. for negative number sign flag is set (S=1), otherwise it is reset (S=0).



**MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION**  
**(Autonomous)**  
**(ISO/IEC - 27001 - 2005 Certified)**

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

---

- e) Write algorithm to transfer block of data from source address to destination address and vice versa [overlapping block transfer].

**(Algorithm-4 Marks)**

**(Any other logic may be considered)**

**Ans:**

**The concept of swapping blocks by including a third temporary block is used in the following algorithm.**

**Algorithm:**

1. Initialize data segment
2. Initialize word counter.
3. Initialize memory pointers for destination and temporary array.
4. Read numbers from destination array.
5. Copy it to temporary array.
6. Increment memory pointer for destination and temporary array for next number.
7. Decrement word counter by one.
8. If word counter not equal to zero then go to step 4.
9. Initialize memory pointers for source and destination array.
10. Read numbers from source array.
11. Copy it to destination array.
12. Increment memory pointer for source and destination array for next number.
13. Decrement word counter by one.
14. If word counter not equal to zero then go to step 10.
15. Initialize memory pointers for temporary and source array.
16. Read numbers from temporary array.
17. Copy it to source array.
18. Increment memory pointer for temporary and source array for next number.
19. Decrement word counter by one.
20. If word counter not equal to zero then go to step 16.
21. Stop.



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION  
(Autonomous)  
(ISO/IEC - 27001 - 2005 Certified)

**SUMMER-15 EXAMINATION**  
**Model Answer**

**Subject Code: 17431**

**Subject Name: Microprocessor & Programming**

- f) Compare between Jump and call instruction in 8086.  
( Any 4 points :1 Mark each)

Any other points may be considered

Ans:

JUMP	CALL
It is used to transfer control of execution to the specified address using 16-bit displacement or CS:IP	It is used to transfer program control to the subprogram or subroutine.
There are two basic types of jump: a. Conditional jump b. Unconditional jump	There are two basic types of CALLs a. Near (inter segment) b. Far (intra segment)
If target of <b>JMP</b> is in the same code segment, it requires only IP to be changed to transfer control to the target location and known as NEAR JUMP and for different code segment CS:IP value is required and known as FAR JUMP.	When 8086 executes <b>CALL</b> instruction old IP value is pushed on to stack for NEAR CALL and old CS: IP value is pushed on to stack for FAR CALL.
Examples:  JMP down ; unconditional  JNC down ; conditional	Examples: 1.CALL delay  2. CALL show 3. CALL FAR PTR show