Microprocessors

Microprocessor Basics

1. What are the differences between microprocessors and microcontrollers?
   ⇒
2. For each of the following devices, would you use a microcontroller or a microprocessor?
   a. Keyboard - Microcontroller
   b. Mouse - Microcontroller
   c. Headphone - Microcontroller
   d. Computer - Microprocessor
   e. TV Remote - Microcontroller
   f. Smart TV - Microprocessor
   g. Regular fridge - MIcrocontroller
   h. Mobile phone - Microprocessor
3. Draw the basic block diagram of a microprocessor based system and briefly describe each component.
   ⇒
4. Briefly describe which component of a microprocessor based system is used for accessing data from a harddisk?
   ⇒ Input-Output Interface
5. Briefly describe the 3 steps involved in executing instructions inside a microprocessor
   ⇒ Fetch → Decode → Execute (Look up the description in slides)
6. What are the differences between RAM and ROM?
   ⇒
7. For each of the following, mention which memory device is used to store the information:
   a. Storing movies/pictures on a computer → Hard Disk
   b. Viewing movies/pictures on a computer → RAM
   c. Computer BIOS → ROM
8. What are the 3 types of bus inside a microprocessor? Briefly describe the purpose of each bus.
   ⇒ Address, Data, Control (Look up the description in slides)
9. ATMega328p is an 8 bit microcontroller. It has 32KB on-board flash memory (ROM)
   a. What is the data bus size of the microcontroller?
      ⇒ 8 bit microcontroller → 8 bit data bus (Word size = 8 bit)
   b. What is the minimum address bus required for accessing the on-board flash memory?
      ⇒ 32KB flash memory = 2^{15} bytes → minimum 15 bit address bus required
10. Calculate the maximum supported RAM size for the following address bus lengths:
    a. 16 bit bus → 64 KBytes memory (10 bit supports 1KByte)
    b. 20 bit bus → 1 MBytes memory (20 bit supports 1MByte)
    c. 24 bit bus → 16 MBytes memory
    d. 32 bit bus → 4 GBytes memory (30 bit supports 1GByte)
    e. 64 bit bus → 2^{64} Bytes memory
11. What is an assembler? Explain in short.

Microprocessor Architecture

1. What are the two functional units of the 8086 microprocessor? Explain why we need the two units.
   ⇒
2. What does the Bus Interface Unit do?
   ⇒
3. What does the Execution Unit do?
   ⇒
4. Draw the 16-bit 8086 processor architecture
   ⇒
5. Which functional unit does the following register belong to?
   a. Segment Register ⇒ BIU
   b. Instruction Pointer ⇒ BIU
   c. Index Register ⇒ EU
   d. General Purpose Register ⇒ EU
   e. Instruction Queue ⇒ BIU
   f. Flags ⇒ EU
6. What is the advantage of an instruction queue in a processor?
   ⇒
7. With example, show how instruction queuing can speed up time required for processes in the 8086 microprocessor.
   ⇒
8. What is the purpose of segment registers in the 8086 microprocessor?
   ⇒
9. What is the role of flag register in the 8086 microprocessor
   ⇒

Memory Partitioning and Segmentation

1. What is the bit length of the internal registers of the 8086 processor?
   ⇒ 16 bit
2. Why cannot the 8086 processor access the full memory without memory partitioning?
   ⇒ Address bus is 20 bit but registers are 16 bit
3. What do you understand by the term “Memory Segmentation”?
   ⇒
4. What are the advantages of overlapping segmentation compared to non-overlapping segmentation?
   ⇒
5. In the 8086 processor, what is the minimum and maximum segment size with the overlapping segmentation scheme?
   ⇒ Minimum segment size = 10h Byte = 16 Byte
   Maximum segment size = 10000h Byte = 64 KByte
6. What is the minimum and maximum number of segments supported by the 8086 microprocessor?
   ⇒ Minimum number of segments supported = 10h = 16
   Maximum number of segments supported = 10000h = 65536
7. Assume the segments are non-overlapping. Find which segment the following physical addresses belong to in the 8086 processor. Then for each of those segments, find the lowest and highest addresses.
   ⇒ A maximum of 16 non-overlapping segments are possible in the 8086 processor. Segment starting addresses would be 0, 10000h, 20000h, 30000h …… F0000h
   a. *12345h* ⇒ Segment no = 1, lowest = 10000h, highest = 1FFFFh
   b. *10h* ⇒ Segment no = 0, lowest = 00000h, highest = 0FFFFh
   c. *10010h* ⇒ Segment no = 1, lowest = 10000h, highest = 1FFFFh
   d. *FFFFEh* ⇒ Segment no = F, lowest = F0000h, highest = FFFFFh
8. Explain why *8AB3Fh* cannot be the starting physical address of an 8086 memory segment.
   ⇒ Starting addresses should be divisible by 16 (10h) but 8AB3Fh is not. So it cannot be the starting address of a memory segment.
9. Find the largest and smallest possible segment address for each of the following physical addresses. Also, mention the logical addresses for each case. (Maximum segment size can be 64KBytes)
   ⇒ To find the smallest possible segment address, we assume the largest possible offset and
   To find the largest possible segment address, we assume the smallest possible offset.
   a. *12345h*
      ⇒ 12345h rounded **down to nearest 10s**= 12340h,
      largest segment address = 1234h
      12345h - FFFFh = 02346h, rounded **up to nearest 10s**= 02350h,
      Smallest segment address = 0235h
   b. *10h*
      ⇒ 00010h rounded **down to nearest 10s**= 00010h,
      largest segment address = 0001h
      00010h - FFFFh = F0011h, rounded **up to nearest 10s**= F0020h,
      Smallest segment address = F002h
   c. *10010h*
      ⇒ 10010h rounded **down to nearest 10s**= 10010h,
      largest segment address = 1001h

10010h - FFFFh = 00011h, rounded **up to nearest 10s**= 00020h,
Smallest segment address = 0002h

d. FFFFEh

⇒ FFFFEh rounded **down to nearest 10s**= FFFF0h,
largest segment address = FFFFh
FFFFEh - FFFFh = EFFFFh, rounded **up to nearest 10s**= F0000h,
Smallest segment address = F000h

10. A memory location has a physical address of 9A7B1 H. Determine the offset address if the segment number is 40FF H.

⇒ 9A7B1h - 40FF0h = 597C1h Offset cannot be larger than 4 hex digits (invalid address)

11. You are given the following instruction >>> **ADD AX, [10h].** You are provided the following data: **DS = AB12h; SS = 2567h; CS= 29C1h.** Find the effective address location for the given instruction.

⇒ The source operand has the logical address = DS:10h = AB12:0010h

12. Find the smallest and the second largest segment address for each of the following physical addresses. Also, mention the logical addresses (**segment: offset pair**) for each case:
    a. FFFEFh
    ⇒ FFFEFh rounded **down to nearest 10s**= FFFE0h,
    largest segment address = FFFEh
    Second largest segment = FFFDh
    FFFEFh - FFFFh = EFFF0h, rounded **up to nearest 10s**= EFFFh,
    Smallest segment address = EFFFh
    b. 2h
    ⇒ 00002h rounded **down to nearest 10s** = 00000h
    Largest segment address = 0000h
    Second largest segment = FFFFh
    00002h - FFFFh = F0003h, rounded **up to nearest 10s**= F0010h,
    Smallest segment address = F001h

13. Explain to which addressing mode does the instruction “HLT AX” belong
    ⇒ Invalid instruction. HLT does not take any operands

14. Explain to which addressing mode does the instruction “RET AX” belong
    ⇒ Invalid instruction. RET does not take any operands

15.

Address	10600h	10601h	20600h	20601h	30600h	30601h
Data	12h	34h	56h	78h	10h	20h

Given DS = 2000h, SS = 1000h, CS = 3000h, BP = 0400h, DI = 0200h. Now **deduce** what data will be stored in BX if the instruction MOV BX, [BP + SI] is executed.
⇒ PA = 1000 x 10h + (0400 + 0200)h = 10600h [SS will be used for BP].

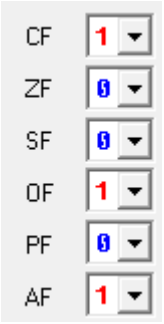
So BX will store 3412h

16. Instead of 64KB, assume the size of each segment for an 8086 is 256 bytes. In that case, deduce the 2nd last address of a segment whose starting address is 10000h.
⇒ 2^8 = 256. So the maximum 8 bit offset is FF.
2nd last address = 10000 + 00FE = 100FEh

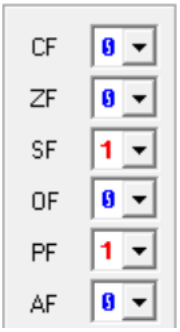
17. Instead of 64KB, assume the size of each segment for an 8086 is 4KB. In that case, deduce the 2nd last address of a segment whose starting address is 10000h.
⇒ 4KB = 1KB x 4 = 2^10 bytes x 2^2 = 2^12 bytes
So the maximum 12 bit offset is FFF.
2nd last address = 10000 + 0FFE = 10FFEh

Flag Register

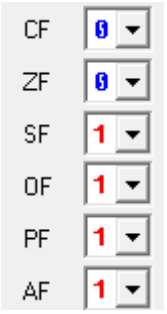
1. Write down the purpose of each of the following flag bit and give on example of its use:
2. Suppose two hexadecimal numbers (i) **96C9** and (ii) **99XY** are to be added by an Intel 8086 microprocessor. Here X and Y represent two unknown hexadecimal digits.
    a. After the 8086 **adds** the numbers (i) and (ii), deduce the minimum values of X and Y needed to get the value of AF = 1
    ⇒ 96C9 = 1001 0110 1100 1001
    99XY = 1001 1001 X Y
    AF = 1 is the carry over from the last four bits, so we can safely set X = 0
    To get a carry of 1, values of Y can be greater than 0111,
    Thus, minimum values of X and Y are 0000, 0111 respectively
    b. Using the deduced values of X and Y obtained from (a), find the values of PF, SF, OF and CF for the given **addition** operation.
    ⇒ The numbers to be added are 96C9, 9907. Flags will be:



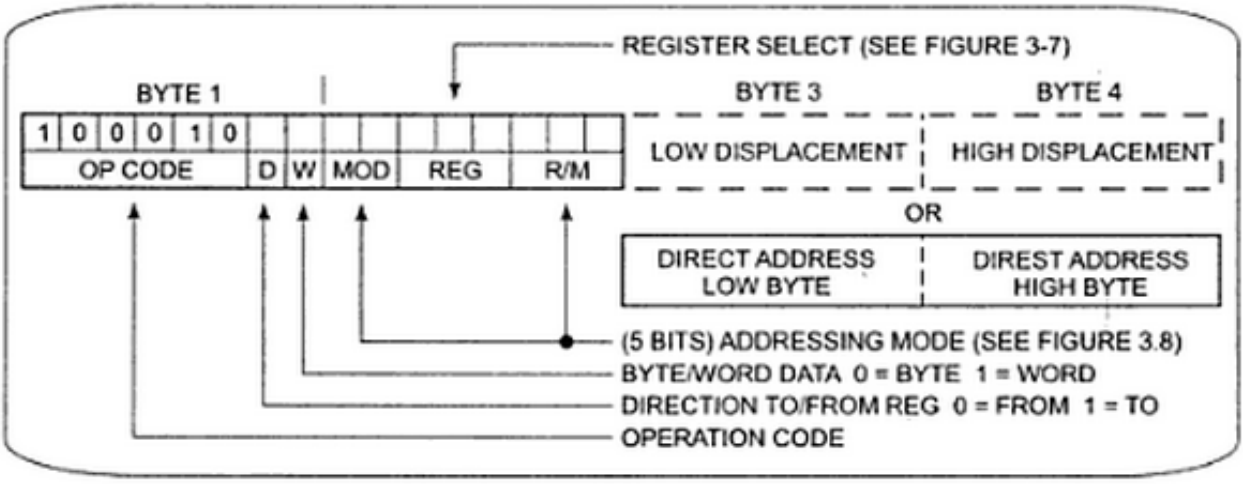
3. Calculate the OF, PF, AF, SF, CF after the execution of the given instructions.
    >>> MOV AX, 0AB56h >>> MOV BX, 3965h >>> ADD AX, BX



4. Calculate the OF, PF, AF, SF, CF after the execution of the given instructions.
    >>> MOV AX, 7EBFh >>> MOV BX, 4A52h >>> ADD AX, BX



Addressing Modes and Machine Code



MOD / RM	00	01	10	11	
				W = 0	W = 1
000	[BX] + [SI]	[BX] + [SI] + d8	[BX] + [SI] + d16	AL	AX
001	[BX] + [DI]	[BX] + [DI] + d8	[BX] + [DI] + d16	CL	CX
010	[BP] + [SI]	[BP] + [SI] + d8	[BP] + [SI] + d16	DL	DX
011	[BP] + [DI]	[BP] + [DI] + d8	[BP] + [DI] + d16	BL	BX
100	[SI]	[SI] + d8	[SI] + d16	AH	SP
101	[DI]	[DI] + d8	[DI] + d16	CH	BP
110	d16 (direct address)	[BP] + d8	[BP] + d16	DH	SI
111	[BX]	[BX] + d8	[BX] + d16	BH	DI

1. Suppose you are given the following machine code for an instruction: **88160080h.** Deduce the original assembly language instruction denoted by the above mentioned machine code.
   ⇒
   First 4 hex digits 8816 = 1000 1000 0001 0110

OPCODE						D	W	MOD		REG			R/M		
1	0	0	0	1	0	0	0	0	0	0	1	0	1	1	0

   D = 0, from register
   W = 0, byte size data
   MOD = 00, R/M = 110 ⇒ d16 (direct address of 16 bits)
   W = 0, REG = 010 ⇒ DL

   Last 4 hex digits 0080 ⇒ lower byte = 00, higher byte = 80

   Thus, the instruction will be **MOV [8000h], DL**

2. Consider the instruction >>> ***MOV C5A4h, CX***
   Deduce the corresponding machine code
   ⇒ Incorrect instruction, cannot move register data to immediate data.
   Correct instruction, **MOV [C5A4h], CX**
3. Consider the instruction >>> ***MOV [AX], [BX+SI+23FFh]***
   What will be the length of the corresponding machine code in bytes?
   ⇒ 2 bytes required for encoding instruction, 2 bytes required for the displacement value
   Total 4 bytes required.
4. Convert **89806910h** from machine language to its corresponding assembly language. Show all of your work.
   ⇒
   First 4 hex digits 8980 = 1000 1001 1000 0000

OPCODE						D	W	MOD		REG			R/M		
1	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0

   D = 0, from register
   W = 1, word size data
   MOD = 10, R/M = 000 ⇒ [BX] + [SI] + d16
   W = 1, REG = 000 ⇒ AX

   Last 4 hex digits 6910 ⇒ lower byte = 69, higher byte = 10

   Thus, the instruction will be **MOV [BX+SI+1069h], AX**

5. Suppose the instruction **MOV DI, [BP+42h]**, appears in a program. What is its machine language for the given instruction?

   ⇒ 8B 7E 42

OPCODE						D	W	MOD		REG			R/M		
1	0	0	0	1	0	1	1	0	1	1	1	1	1	1	0

6. Given DS = **1000h**, CS = **3000h**, SS = **8A40h**, SI = **2000h**, CX = **43AEh**, DI = **030Fh**. We also execute the **JMP [SI + 1000h]** instruction. Now, answer the questions based on the given table and data:
   a. **Explain** to what addressing mode the instruction **IN 05h, DX** belongs.
   b. **Deduce** the physical address of the instruction that 8086 will jump to.
   c. "**If we are using overlapping segmentation, the physical address "00015h" can be a part of 3 different segments**". **Explain** with logic any arguments you have for or against this statement.

Physical Address	8 bit hex data
14001h	78h
14000h	56h
13001h	34h

13000h	12h
03000h	ABh

⇒ **a.** IN 05h, DX is invalid instruction. Source has to be AX or AL only

b. Note: JMP BX will set IP = BX, JMP [BX] etc will set IP value to whatever is stored at the location DS:BX
[SI+1000h] points to the physical address denoted by **DS:(SI+1000) =** 1000h:3000h = **13000h**

The **word size data** stored at location 13000h (lower byte), 13001 (higher byte) = 3412h
This value will be the new value of IP

Thus, JMP [SI+1000h] will jump to the logical address = CS:IP = **3000h:3412h**
Physical address of the jump location = **33412h**

Code to verify this behavior in emu8086:

```
org 100h
MOV AX, 1000h
MOV DS, AX
MOV SI, 2000h
MOV AX, 7856h
MOV [SI+1000h], AX
JMP [SI+1000h]
ret
```

7. Suppose you are given the following values; DS = 1234h, CS = 2345h, SS = 3456h, BX = 0010h, DI = 0021h. Assume we are using overlapping segmentation here. Additionally, you are also given two instructions:
   Instruction 01 >>> **MOV AX, [BP]**
   Instruction 02 >>> **MOV [BX+DI+1019H], AX**

   Now based on these values answer the following questions:
   a. Explain with proper reasoning to which addressing mode does Instruction 01 belong.
      ⇒ MOV AX, [BP] is register indirect
   b. Calculate what physical address is the destination of Instruction 02 referring to? Your final answer should be in hexadecimal.
      ⇒ Logical address of the destination **= DS: BX+DI+1019**
      = 1234h : (0010+0021+1019)
      = 12340h + 104Ah = 1338Ah
   c. Using the physical address obtained from "b", identify the largest segment the obtained physical address can be a part of and deduce the corresponding logical address.

8. The OPCode for MOV = 100010. Deduce the machine code (in hexadecimal) for the following instruction:

>>> MOV [BX+DI+46DCh], SI

⇒ 89 B1 DC 46

OPCODE						D	W	MOD		REG			R/M		
1	0	0	0	1	0	0	1	1	0	1	1	0	0	0	1

9. For each of the following statements, state the addressing modes.

(i) MOV BX, 1000H ⇒ Immediate

(ii) MOV DI, 1008H ⇒ Immediate

(iii) MOV AX, B [BX+SI+2]; B is an array ⇒ Base relative plus index

(iv) MOV AX, B [BX] ⇒ Register relative

10. Now consider the following instruction " MOV J, [BX + K + L] " whose equivalent Machine code is XY A8 86 02 h. Here XY is an unknown 8-bit hex value. Additionally, J represents a specific general purpose register and K is a specific index register. The given instruction basically moves data from the address 474D2 h and copies it to the "J" register.

CS	DS	BX	SI	DI	IP
286A h	458E h	0A2E h	N/A	N/A	N/A

a. Deduce what are the J and K registers. Give reasons behind your answer.

⇒ K register can be either SI or DI (BX+SI or BX+DI)

J is currently unknown

The displacement value has two bytes, (d16 displacement): higher byte = 02h, lower byte = 86h

From this information, we can narrow it down to the following:

MOD / RM	00	01	10	11	
				W = 0	W = 1
000	[BX] + [SI]	[BX] + [SI] + d8	[BX] + [SI] + d16	AL	AX
001	[BX] + [DI]	[BX] + [DI] + d8	[BX] + [DI] + d16	CL	CX

From the code XY A8 86 02, we get the following:

OPCODE						D	W	MOD		REG			R/M		
1	0	0	0	1	0			1	0	1	0	1	0	0	0
X				Y				A					8		

R/M value is 000, K register will be SI (first row)

REG value is 101, so it can be either CH or BP. CH is the general purpose register here, BP is a pointer register. So J will be CH.

b. Deduce the values of L and XY

⇒ From (a) we have found that the MOV operation moves byte size data to the CH register

So we can fill up the table accordingly:

OPCODE						D	W	MOD		REG			R/M		
1	0	0	0	1	0	1	0	1	0	1	0	1	0	0	0
X				Y				A				8			

Thus, XY would be 1000 1010 = 8A

As for L, The displacement value has two bytes, (d16 displacement): higher byte = 02h, lower byte = 86h

Thus, L = 0286h

Nobody asked but the instruction is MOV CH, [BX+SI+0286h] and machine code 8A A8 86 02

c. Deduce the value that resides inside the K register.

⇒ [BX + K + L] / [BX+SI+0286h] has a physical address 474D2h from the question.

Given, DS = 458Eh

BX = 0A2Eh

We deduce that the logical address DS:BX+SI+0286h has physical address value 474D2h

Thus,

474D2h = DSx10 + BX + SI + 0286h = 458E0h + 0A2Eh + SI + 0286h

⇒ SI = 0F3Eh

11. a. Convert the following machine codes to their corresponding assembly language instructions:

- 8BD0
- 8BDA

OPCODE						D	W	MOD		REG			R/M		
1	0	0	0	1	0	1	1	1	1	0	1	0	0	0	0
8				B				D				0			

D = 1, to register W = 1, word size data

MOD = 11, R/M = 000 ⇒ AX W = 1, REG = 010 ⇒ DX (to register DX)

Thus, the instruction will be **MOV DX, AX**

OPCODE						D	W	MOD		REG			R/M		
1	0	0	0	1	0	1	1	1	1	0	1	1	0	1	0
8				B				D				A			

D = 1, to register W = 1, word size data

MOD = 11, R/M = 010 ⇒ DX W = 1, REG = 011 ⇒ BX (to register BX)

Thus, the instruction will be **MOV BX, DX**

b. If, before the execution of the above lines of code, the values of the general-purpose registers were AX = 7, BX = 6, CX = 5 and DX = 3. Calculate the values of these mentioned registers after the codes you found in part a are executed.

	AX	BX	CX	DX
	7	6	5	3
MOV DX, AX	7	6	5	7
MOV BX, DX	7	7	5	7

12. ⇒ a. First instruction has no displacement/direct address, requires 2 bytes
Second instruction has 16 bit displacement, and requires 4 bytes.
Total bytes required for the two instructions = 2 + 4 = 6 bytes

b. First instruction: MOV AX, [BX]

OPCODE		100010	Byte 1	= 10001011
Direction (to AX)		1		= 8Bh
Word		1		
MOD	no disp.	00	Byte 2	= 00000111
REG	AX	000		= 07h
R/M	[BX]	111		

Second Instruction: MOV 1234h[SI], AX

OPCODE		100010	Byte 1	= 10001001
Direction (from AX)		0		= 89h
Word		1		
MOD	16 bit disp	10	Byte 2	= 10000100
REG	AX	000		= 84h
R/M	[SI]+d16	100		
Displacement, lower			Byte 3	= 34h
Displacement, higher			Byte 4	= 12h

Data in memory

Logical Address	Content (Hex)
8000:0000	8B
8000:0001	07
8000:0002	89
8000:0003	84

8000:0004	34
8000:0005	12
8000:0006	

14.

Address	10600h	10601h	20600h	20601h	30600h	30601h
Data	12h	34h	56h	78h	10h	20h

Given DS = 1000h, SS = 2000h, CS = 3000h, BP = 0500h, SI = 0100h. Now **deduce** what data will be stored in BX if the instruction MOV BX, [BP + SI] is executed.
⇒ PA = 2000 x 10h + (0500 + 0100)h = 20600h [SS will be used for BP],
So BX will store 7856h

15.

Address	589B6h	589B7h	589B8h	650FBh	650FCh	650FDh
Data	15h	74h	36h	C1h	DAh	FEh

Assume DS = 5555h, SS = 3200h, CS = 2000h, SI = 3467h, BP = 2766h, BX = FBACh.

a. Based on the above-given registers and table, deduce which data will be accessed by executing the instruction CALL [BX].
⇒ As this instruction is addressing program codes in memory, the instruction will access a physical location which is in the code segment. In order to access the code segment we need CS and IP values. The CS value is given and the IP value has to be taken from the table as the IP value is given in relative register. The location of the lower byte of IP value is calculated by DS*10+contents of the relative registers.
Physical location where lower byte of IP is stored = DS*10+BX
= 5555*10+FBACh
= 650FCh
As the IP is 16 bit we have to take the value of the next location (650FDh) too. The value of the IP will be FEDAh. So the instruction will access the location CS*10+IP = 2000*10 + FEDAh = 2FEDAh.

b. Explain with reasoning CALL [BX] falls under which addressing mode.
⇒ Addressing Program Codes with Relative Registers (No proper name is given in the slide).

c. Deduce the hex machine code of the instruction MOV [BP], BL.
⇒ 88 1E 66 27 (immediate add.)

16.

Address	576BDh	576BEh	576BFh	5A7B9h	5A7BAh	5A7BBh
Data	75h	42h	BAh	F2h	4Eh	AEh

Assume CS = 6666h, SS = A200h, DS = 4AC0h, SI = FBBAh, BP = 8877h, BX = CABDh.

a. Based on the above given registers and table, deduce which data will be accessed by executing the instruction CALL [BX].
⇒ As this instruction is addressing program codes in memory, the instruction will access a physical location which is in the code segment. In order to access code segment we need CS and IP values. The CS value is given and the IP value has to taken from the table as the IP value is given in relative register. The location of the lower byte of IP value is calculated by DS*10+contains of the relative registers.
Physical location where lower byte of IP is stored = DS*10+BX
= 4AC0*10+CABDh
= 576BDh
As the IP is 16 bit we have to take the value of the next location (576BEh) too. The value of the IP will be 4275h. So the instruction will access the location CS*10+IP = 6666*10 + 4275h = 6A8D5h.

b. Explain with reasoning CALL [BX] falls under which addressing mode.
⇒ Addressing Program Codes with Relative Registers (No proper name is given in the slide).

c. Deduce the hex machine code of the instruction MOV AX, [BP+2000h].
⇒ 8B 86 00 20h

Memory Banking

1. For the instruction, MOV 'Destination Register', [117X h], the A_0 and $\overline{BHE}$ values were both found to be 0. Assuming the memory is divided into even and odd banks, answer the following:
   a. Find a possible value of X. Explain why you chose this value.
      ⇒ A_0 and $\overline{BHE}$ values were both zero, thus this is an aligned MOV operation on a word sized register.
      Aligned move : X should be even (0/2/4/6 etc)
   b. Suggest a possible general-purpose register as a Destination for the MOV instruction above. Explain whether it'll be an 8-bit or a 16-bit register.
      ⇒ A_0 and $\overline{BHE}$ values were both zero, thus this is an aligned MOV operation on a word sized register.
      Possible destination register: AX, BX, CX, DX
2. Assuming the memory is divided into even and odd banks, an instruction requires one bus cycle for which A_0 = 1 and $\overline{BHE}$ = 0. If the instruction is a MOV instruction from a memory to a destination register,
   a. Deduce a possible value of the memory location.
      ⇒ Byte size move operation from odd memory bank
      Address should be odd (1/3/5/7 etc)
   b. Deduce a possible general-purpose register as a Destination for the MOV instruction above.
      ⇒ Byte size move operation from odd memory bank
      Any byte register AH, AL, BH, BL etc
3. An 8086 cpu has a RAM of 1MB split into two equal memory banks. The A_0 and $\overline{BHE}$ pins of the cpu are used to access the two memory banks. For each of the given instructions, determine the following information:
(i) Size of the data being transferred (byte/word)
(ii) Which portion of the data bus was used to transfer the data from source location to destination register
(iii) Value of A_0 and $\overline{BHE}$ required to make the transfer
(iv) Memory bank from which the data is fetched
(v) Total number of data cycles used to move data
(vi) The internal address of the selected memory bank from which the data is being transferred (19 bits address seen by the selected memory bank)
Assuming DS=0000h, consider the following instructions-
   a. MOV AH, [8086h]
      (i) Byte size
      (ii) D7-D0
      (iii) A_0 = 0, $\overline{BHE}$ = 1
      (iv) Even bank
      (v) Single cycle
      (vi) Address of even bank = floor(8086/2) = 4043H (19 bits address)
   b. MOV AL, [8086h]
      Same as above
   c. MOV AX, [8086h]
      (i) Word size
      (ii) D15-D0
      (iii) A_0 = 0, $\overline{BHE}$ = 0
      (iv) Both memory banks
      (v) Single cycle
      (vi) Address of even,odd bank = floor(8086/2) = 4043H (19 bits address)

d. MOV BH, [8085h]
(i) Byte size
(ii) D15-D8
(iii) $A_0 = 1$, $\overline{BHE} = 0$
(iv) Odd bank
(v) Single cycle
(vi) Address of odd bank = floor(8085/2) = 4042H (19 bits address)

e. MOV BL, [8085h]
Same as above

f. MOV BX, [8085h]
(i) Word size
(ii) D15-D0
(iii) 1st bus cycle: $A_0 = 1$, $\overline{BHE} = 0$, 2nd bus cycle: $A_0 = 0$, $\overline{BHE} = 1$
(iv) Odd bank first, even bank next.
(v) Two cycle
(vi) Address of odd bank = floor(8085/2) = 4042H
Address of even bank = floor((8085+1)/2) = 4043H

4. Why is memory banking done in the 8086 system? What would happen if memory banking is not implemented?
⇒ To access word sized data in a single bus cycle. Without banking, word sized data movement would require twice as much time.
5. What do you understand by aligned word and unaligned word? Explain with an example.
⇒ Aligned word: When the higher byte is contained in the odd bank and lower byte is contained in the even bank for a word sized move operation.
Unaligned word: When the higher byte is in the even bank and the lower byte is in the odd bank for a word sized move operation.

Pin Specification and Timing Diagram

1. Suppose you have an Intel 8086 which is operating at a Duty Cycle of 60% and for each clock pulse assume T_{off} = 40ns. The 8086 is now going to execute the instruction MOV [1235h], AX. Based on this, answer the following questions:

a. Estimate the frequency at which the 8086 is operating.
⇒ Duty Cycle = 60% = percent time of T_{on}
Percent time of T_{off} = 40% = 40ns
Cycle duration 100% = 100ns
Frequency = 1/100ns = 10^7 Hz = 10MHz

b. Calculate the total time for one Instruction Cycle of the given instruction.
⇒ The given instruction is move operation of an unaligned word sized data
2 bus cycles required to move the data
Clock cycles required = 2x4 = 8
Total time required = 8x100ns = 800ns

c. Calculate the values of the A_0 and BHE' pins during the execution of the given instruction.
⇒ On the first bus cycle, Odd bank is accessed. A_0 = 1 and BHE' = 0
On the second bus cycle, even bank is accessed. A_0 = 0 and BHE' = 1

2. The diagram below shows 1 MACHINE/BUS CYCLE of an 8086 microprocessor operating at a particular frequency.

a. Calculate the percentage of Duty Cycle of the waveform. Additionally, calculate the frequency (in Mhz) at which the 8086 is operating if each clock pulse is 'ON' for exactly 55.55ns.
⇒ T_{on}= ⅔ of full cycle = 66.67% duty cycle = 55.55ns
Full cycle duration, T = 55.55 x 100 / 66.67 = 83.32ns
Frequency = 1/83.32ns = 12 MHz

b. During a READ cycle, suppose an 8086 is operating at the frequency you got from (I) and is now reading from a Memory that takes 300ns to upload data on the data bus. Explain with reasoning, whether the microprocessor will insert a Wait State (Tw) or not.
⇒ During a read cycle, the $\overline{RD}$ pin issues a read command to the memory module during the T2 cycle. Whether the processor will enter Tw state is dependent on availability of data from memory by T3 state. As a result, if the data is not available within a single T state, the processor will enter Tw state.
Since a single T state = 83.32ns and the time for data to be available = 300ns, the processor will enter the waiting state for about 4 clock cycles.

3. Suppose an 8086 is operating in a way such that T_{ON} is 1/4th of the total time required for one clock pulse. Consider T_{ON} is 30ns. Now the 8086 is going to execute the instruction MOV AX, [2315h] i.e. 16 bits of data will be read from memory.

a. Calculate the frequency in MHz at which the 8086 is operating.
⇒ T_{on}= ¼ of full cycle = 25% duty cycle = 30ns

Full cycle duration = 30*4 = 120ns
Clock frequency = 1/120ns = 8.33MHz

b. Calculate the time required for 1 Machine / Bus Cycle.
⇒ Time required for 1 machine cycle = 4 clock cycle = 120*4 = 480ns

c. Deduce the total time required to execute the given instruction MOV AX, [2315h].
⇒ Unaligned data → Two bus cycles required = 480x2 = 960ns

d. Explain with proper reasoning the values of pins A_0 and BHE' during the execution of the given instruction MOV AX, [2315h].
⇒ On the first bus cycle, Odd bank is accessed. A_0 = 1 and BHE' = 0
On the second bus cycle, even bank is accessed. A_0 = 0 and BHE' = 1

e. Suppose from the start of the T3 clock pulse, the memory took 240 ns to finalize uploading data on the data bus. Explain with reason whether the processor will exert a Wait State or not.
⇒ Since two clock cycles were required, the processor will exert a wait state of one clock cycle.

4. A Microprocessor has a duty cycle of 40%.
   a. Calculate the frequency (in Mhz) at which the 8086 operates if each clock pulse is 'ON' for exactly 80 ns.
   ⇒ DIY
   b. Construct the timing diagram of a Write cycle.
   ⇒ Refer to slide

5. For the following instructions, mention the states of $\overline{RD}$, $\overline{WR}$, $M/\overline{IO}$, $\overline{BHE}$ pins during the T2 cycle.
   a. MOV AL, [34h]

$\overline{RD}$	$\overline{WR}$	$M/\overline{IO}$	$\overline{BHE}$
0 (read from memory)	1 (not writing)	1 (access memory)	1 (even address, byte)

   b. MOV [33h], BL

$\overline{RD}$	$\overline{WR}$	$M/\overline{IO}$	$\overline{BHE}$
1 (not reading)	0 (write to memory)	1 (access memory)	0 (odd address, byte)

   c. OUT 82h, AL

$\overline{RD}$	$\overline{WR}$	$M/\overline{IO}$	$\overline{BHE}$
1 (not reading)	0 (write to output)	0 (access IO)	1 (even address)

   d. IN AL, 82h

$\overline{RD}$	$\overline{WR}$	$M/\overline{IO}$	$\overline{BHE}$
1 (reading input)	1 (not writing)	0 (access IO)	1 (even address)

6. a. Suppose the time required to complete 1 instruction cycle for the instruction MOV AX, [1235h] is 200ns. And for each bus cycle TON = 0.2T where T represents the time required for 1 clock pulse. Now deduce the values of TON, TOFF, and the frequency at which this 8086 is operating.
⇒ 1 instruction cycle = 2 bus cycles for the given instruction
So, 2 bus cycles = 200ns; hence 1 bus cycle = 100ns; 1 clock pulse = 100/4 = 25ns
Ton = 0.2 x 25ns = 5ns; Toff = 25-5 = 20ns;
Frequency = 1/25ns = 40Mhz

b. Deduce the values of the Ao and BHE pins during 1 instruction cycle of the above given mov operation.
⇒ 1st bus cycle; Ao = 1, BHE' = 0 and 2nd bus cycle, Ao = 0, BHE' = 1

c. Illustrate the timing diagram for the given mov operation showcasing only the CLK, M/IO', and RD'/WR' pins [use either RD' or WR' depending on the operation].
⇒ Will use RD' pin

7. a. Suppose the time required to complete 1 instruction cycle for the instruction MOV [1235h], AX is 400ns. And for each bus cycle TON = 0.4T where T represents the time required for 1 clock pulse. Now deduce the values of TON, TOFF, and the frequency at which this 8086 is operating.
⇒ 1 instruction cycle = 2 bus cycles for the given instruction
So, 2 bus cycles = 400ns; hence 1 bus cycle = 200ns; 1 clock pulse = 200/4 = 50ns
Ton = 0.4 x 50ns = 20ns; Toff = 50-20 = 30ns;
Frequency = 1/50ns = 20Mhz

b. Deduce the values of the Ao and BHE pins during 1 instruction cycle of the above given mov operation.
⇒ 1st bus cycle; Ao = 1, BHE' = 0 and 2nd bus cycle, Ao = 0, BHE' = 1

c. Illustrate the timing diagram for the given mov operation showcasing only the CLK, M/IO', and RD'/WR' pins [use either RD' or WR' depending on the operation].
⇒ Will use WR' pin

Interrupts

1. Suppose you have to design a new Priority Interrupt Controller (PIC) named “X”. It should be able to handle 10 different interrupt requests at a certain time and should be able to address 16,384 types of interrupts. But keep in mind that the process of handling interrupt requests and priority handling would be the same as 8259A PIC.

a. Explain in brief how an 8259A PIC co-operates with the 8086 microprocessor to handle interrupts from multiple devices with just one interrupt request pin.
⇒ Self study

b. Deduce with reasoning how many data bus lines of the PIC “X” would be connected to the 8086 data bus?
⇒ PIC 8259 IC works with the 256 interrupt types of the 8086 processor that requires 8 data bits to access the 256 types ($2^8 = 256$). [when an interrupt is triggered, the 8259 IC sends “data” to the 8086 IC mentioning which type was triggered.]
So PIC X which is able to address 16384 types of interrupt should have,
Number of data bits = $log_2(16384)$ = 14

c. In cascading mode, deduce with reasoning the maximum number of I/O devices PIC “X” can handle. Also find the number of cascading pins for PIC “X”.
⇒ PIC X can handle 10 different interrupts at a time: IRR/ISR will have 10 bits (10 interrupt inputs per chip)
Thus, in the cascading mode, PIC X will be able to handle 10x10 = 100 I/O devices.

To address the 10 slave PIC X IC, we need,
Number of cascade pins (CAS) = $ceiling(log_2(10)) = 4$

There should be CAS3-CAS0

2. Assume the table is a portion of the current memory address space of an Intel 8086. The microprocessor currently has the following values in its registers: SS = 2000h, SP = 1124h, CS = 3000h, IP = 1450h. Now a signal arrives at the INTR pin of this 8086.

Addr	00117h	00116h	00115h	00114h	00276h	00277h	00248h	00279h
Data	45h	86h	22h	14h	12h	34h	56h	78h

a. If the 8086 decides to service the interrupt, then do the bits of the flag register change?
⇒ Only TF and IF can change (cleared). The rest of the values will remain the same and will be popped after the interrupt service routine finishes execution.

b. “Hardware interrupts can also be represented as Software interrupts” - Explain your views with regards to this statement. ⇒ DIY

c. If the signal is of Interrupt type 69, then deduce the values of CS and IP as the 8086 starts the service routine.
⇒ Type 69 interrupt: Interrupt vector starts at = 69x4 = 276 = 0114h
Thus, new value of IP after interrupt = (Low byte at 0114h, High byte at 0115h) = 2214h
New value of CS after interrupt = (low byte at 0116h, high byte at 0117h) = 4589h

d. Deduce the values of CS, IP, and SS after the interrupt service routine has been completed.
⇒ CS, IP, SS values will be the same as it was before the interrupt.

3.

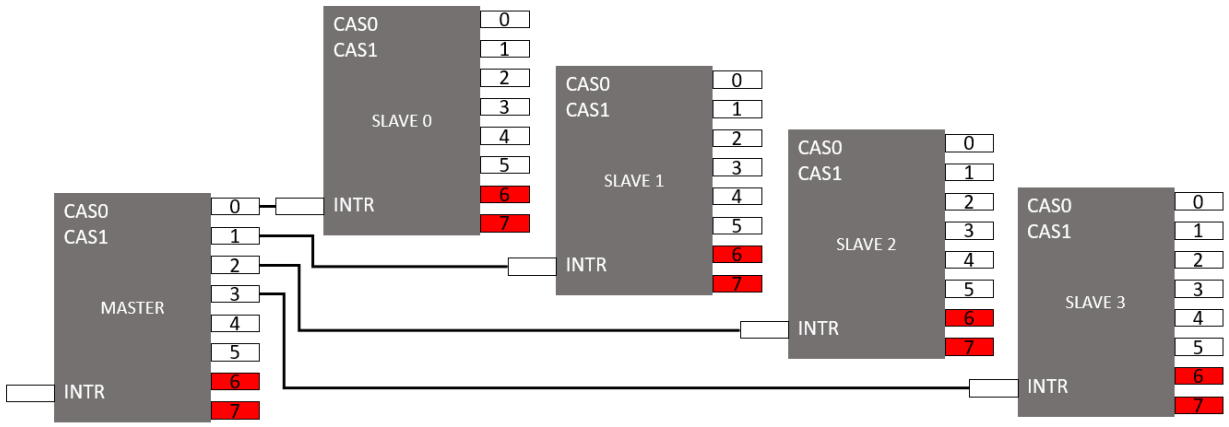
Address	0006A	0006B	0006C	0006D	0006Eh	0006F
Data	00h	08h	00h	03h	00h	05h

a. Use the given table to calculate the starting address of the ISR for interrupt type 27.
⇒ type 27 has interrupt vector at 27x4 = 108 = 6C
Thus, IP = 0300h, CS = 0500h.
Starting address for ISR = CS:IP = 0050h:0030h = 05300h

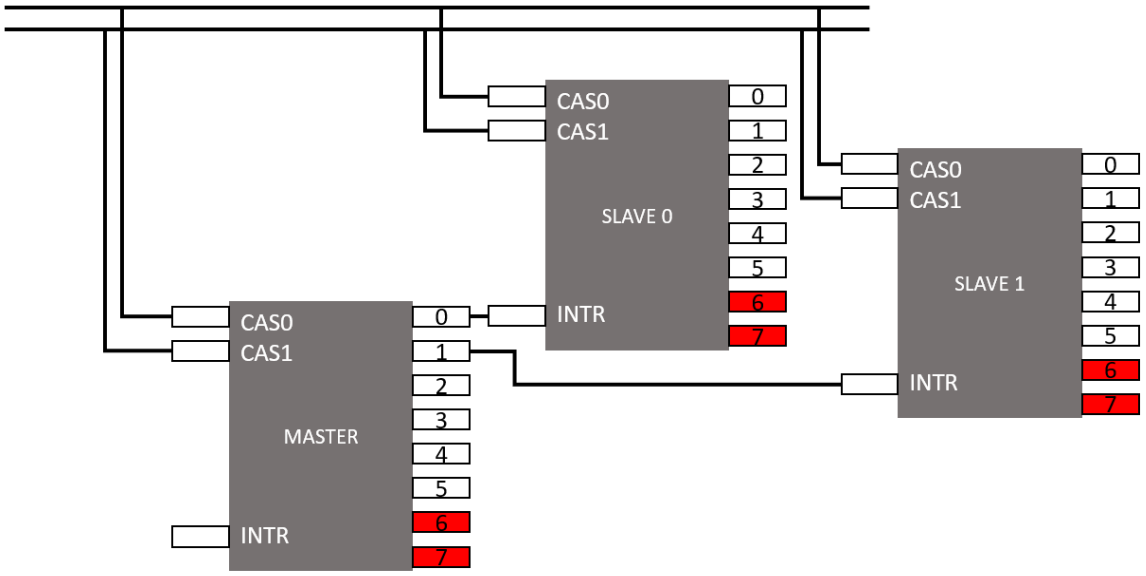
b. Suppose the above-mentioned ISR is a program consisting of nothing but 25 MOV operations, an IRET instruction at the end, and nothing else. The structure of each MOV operation in the ISR is "**MOV AL, [XXXX h]**", where XXXX is an Offset. **Calculate** the physical address where we can find the IRET instruction
⇒ MOV AL, [XXXXh] has 16 bits of displacement, thus machine code will be of 4 bytes
25 instructions, 4 bytes per instruction = 100 bytes total = 64h worth memory
Thus, location of IRET = 05300h + 64h + 1h = 05361h

4. Assume you have several faulty 8259 PICs where the IR6 and IR7 pins do not work, and CAS2 is always giving a low (0) signal.

a. In this scenario explain what will be the maximum number of interrupts that can be serviced through these faulty PICs?
⇒ CAS2 always low signal, so we can only use CAS1-CAS0 (2 cascade pins = 4 slave devices)
Each 8259 IC has IR0-IR5 working (total 6 working pins)
One master, 4 slaves, thus number of free interrupt pins of master = 6-4 = 2 (4 occupied by slaves)
4 slaves, 6 pins each = 4x6 = 24 interrupt pins
Total number of interrupts serviceable = 24 + 2 = 26



b. For a particular project, your microprocessor needs to provide service to 16 interrupts coming from I/O devices. If you use the same faulty PICs mentioned above then explain how many slave PICs will be required? Also using the proper diagram, illustrate how the master and slave PICs will be connected with each other.
⇒ Two slaves = 6x2 = 12 interrupts
Free pins of master = 6-2 = 4
Total interrupts = 12 + 4 = 16



5.

Address	000B6	000B7	000B8	000B9	000BA
Data	20h	30h	40h	50h	60h

a. Suppose, the interrupt pointer for Interrupt type "X" is 31434 h. Additionally, the corresponding CS values of the ISR for this interrupt are stored starting from the memory location 000B6 h. Now, deduce what value will be stored in the memory location 000B5.
⇒ CS lower byte in B6, higher byte in B7, CS = 3020h
We have, physical address = 31434h
Thus, IP = Physical address - CSx10 = 31434h - 30200h = 1224h
So the data stored in 000B4h and 000B5h are lower and higher bytes of IP respectively.
Thus, data at 00B5 = 12h

b. Using the result obtained from "A", deduce the value of "X".
⇒ The starting address of the interrupt vector = 000B4h = 180
Thus, interrupt type = 180/4 = 45

c. "All software interrupts are maskable"- Do you agree with this? Explain your answer ⇒ DIY

d. When servicing an interrupt explain why does the 8086 clear IF and TF? ⇒ DIY

6.

a. A student designed a new PIC called 'PIC 2.0' which takes 20 interrupt requests via its IRR0 – IRR19 pins. The size of its Interrupt Mask register (IMR), Interrupts Service Register (ISR), and Interrupt Request Register (IRR) is 20 bits. It also has 4 Cascading (CAS) pins.

   i. Assume the value of ISR0 - ISR4 is 10000, the value of IMR0 - IMR4 is 10101, and the value of IRR0 - IRR4 is 01110. Now, explain the order in which the interrupts IR0-IR4 will be serviced. Assume IR0 has the highest priority and IR19 has the least priority.
   ⇒ IMR0, IMR2, IMR4 are masked, they will be disabled.
   IRR1, IRR2, IRR3 have received interrupt signals, among which IRR2 is disabled. Between the remaining two, IRR1 will be serviced before IRR3
   ISR0 = 1, meaning IRR0 is already being serviced.
   Thus, the servicing order will be IRR0→ IRR1→IRR3

b. The student decided to cascade 20 PIC 2.0's together in a master-slave configuration. However, due to a design flaw in the PIC 2.0, she noticed that the master PIC 2.0 could not communicate with all the connected 20 slave PIC 2.0s. Now assuming all IRR pins are Unmasked.

i. Calculate the maximum number of interrupts that a master PIC 2.0 can handle in cascade mode. Show necessary calculations.

⇒ There are 4 CAS pins, thus $2^4 = 16$ slaves are supported by the master
Each slave has 20 pins ⇒ 20x16 = 320 interrupts
Remaining master pins = 20-16 = 4
Thus, total number of serviceable interrupts = 324

ii. Explain how you would correct the flaw in the design so that the master can communicate with all the 20 slave PIC 2.0s.

⇒ In order to communicate with all 20 slave PIC, number of CAS pins required =

$$ceiling(log_2(20)) = 5$$

iii. Explain 2 possible methods of stopping the servicing of any interrupts between the master and the 8086 without disconnecting the wires or disabling the INTR / INT / INTA' pins.

⇒ Using the interrupt flag can disable interrupts.

7. Use the given table to calculate the starting address of the ISR for interrupt type 29.

Address	00074	00075	00076	00077	00078	00079
Data	08h	00h	03h	00h	00h	05h

a. Assuming that the above ISR has only n "MOV AL, [XXXX h]" (XXXX are 4 hex bits offset). If the IRET instruction is found in the address 0003D, find the value of n?

⇒ Interrupt type 29, interrupt vector starting address = 29x4 = 116 = 74h
From table, we get IP = 0008h, CS = 0003h
Thus, interrupt service routine physical address = 00038h
MOV AL, [XXXXh] has 16 bit displacement value, machine code will have 4 bytes
Thus we have, 00038h + n x 4 + 1 = 0003Dh
⇒ n = 1
Only one MOV instruction will be stored in the location.

b. Assume you have several faulty 8259 PICs where the IR5 and IR6 pins do not work, and CAS2 is always giving a low (0) signal.

i. In this scenario explain what will be the maximum number of interrupts that can be serviced through these faulty PICs?

ii. Draw the block diagram for such a system connecting the master and slave PICs

⇒ Solved in 4(a)

8. Assume a hypothetical scenario where CS and IP values for the starting address of an Interrupt Service Routine (ISR) are 1230h and 2000h, respectively. Here each memory location can store a maximum of 1 byte data. Now the ISR consists of 15 lines of code and assume 3 bits are required to store each line of code. Now, deduce mathematically the address where the IRET instruction will be found.

⇒ Starting address of ISR = 1230x10 + 2000 = 14300; Total size of ISR = 15x3 = 45 bits
IRET will take 43rd, 44th, and 45th bit. So, Location = 14305h

9. Assume the CS value required to locate an ISR is CBCCh and is stored at memory locations 000AEh and 000AFh respectively. Additionally assume the Interrupt vector (IV) of that ISR is CCEF2h. Hence, deduce the Interrupt Type that caused the aforementioned ISR and the value that will be stored at 000ADh.
⇒ IP = CCEF2 - (CBCCx10) = 1232; 000AD will store 12h
000AC h = 172 d; So interrupt type = 172/4 = Type 43

10. Assume at T= 0 ns, an interrupt request is made at IRR = 4 of a PIC working in fixed priority mode. The duration required to serve the interrupt is 20 ns. However, at T=10 ns another interrupt request is made at IRR = 2 which requires 25 ns to be served. Finally, at T = 15 ns, the last interrupt request arrives at IRR = 7 which takes 5 ns to get served.

a. Explain how the PIC performs its task in this case. Your answer must refer to the changes in ISR and IRR at each stage. Assume the interrupt flag can never be reset to 0.
⇒ Starts IR4, pauses IR4 and starts IR2, finishes IR2 and IR4, lastly starts IR7

b. Deduce the time at which PIC starts serving the IRR 7 interrupt request.
⇒ After 45ns

c. Illustrate the Internal Block Diagram of an 8259A PIC.

Basic Input Output

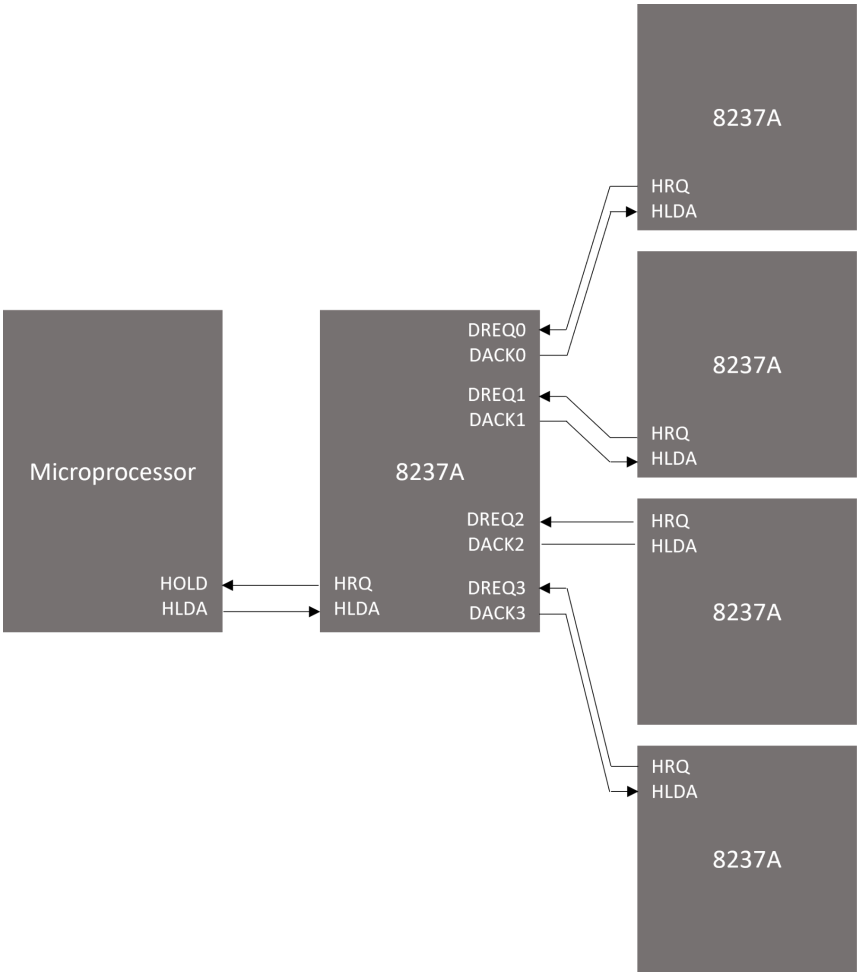
1. Explain what is variable addressing? Which register is used for variable addressing?
⇒ Variable addressing uses the DX register to store the address of input/output devices. The formats used for variable addressing are:
IN AX, DX / IN AL, DX / OUT DX, AL

2. Explain what is Programmed and Interrupt Driven I/O? Why do we need DMA?
⇒ Study from slide

3. Explain how data is transferred between the I/O and the memory of an Intel 8086 using the 8237A DMA controller.
⇒ Study from slide

4. (a) To adopt the utility of 15 channels, estimate the minimum number of secondary 8237 DMA controllers needed.
⇒ Each master controller can host 4 slave controllers as the 8237 DMA controller has 4 channels. Thus, for 15 channels in total, a system will require at least 4 slaves (secondary controller)

(b) Construct the DMA cascading diagram based on your answer in (a). Your diagram should show how the secondary controllers are connected with the primary controller using the appropriate pins.
⇒

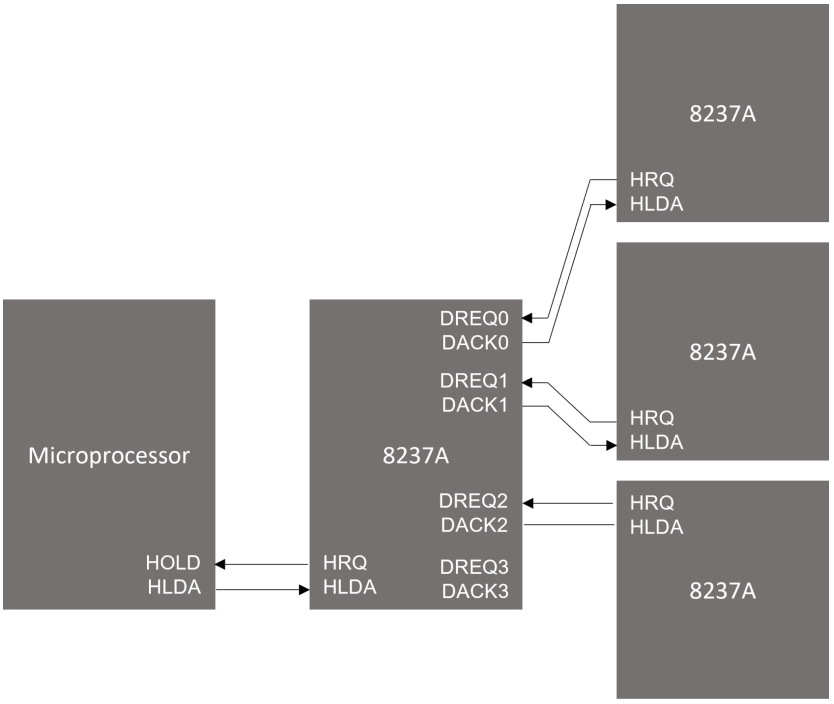


5. (a) To adopt the utility of 13 channels, estimate the minimum number of secondary 8237 DMA controllers needed.

⇒ Each master controller can host 4 slave controllers as the 8237 DMA controller has 4 channels. Thus, for 13 channels, we require 3 slaves (3x4=12) and one free channel of the master controller.

(b) Construct the DMA cascading block diagram based on your answer in (a). Your diagram should show how the secondary controllers are connected with the primary controller using the appropriate pins.

⇒



6. Describe the values of the given pins of an 8237 DMA Controller during a DMA write cycle:

i. $\overline{IOR}$ ii. $\overline{IOW}$ iii. $\overline{MEMW}$ iv. $\overline{MEMR}$

⇒ During a DMA Write cycle, the DMA will read data from a peripheral and write the data to memory. Thus the pin conditions during a write cycle will be:

i. $\overline{IOR}$ ⇒ Low, since reading from peripheral
ii. $\overline{IOW}$ ⇒ High
iii. $\overline{MEMW}$ ⇒ Low, since writing data to memory
iv. $\overline{MEMR}$ ⇒ High

7. Describe the values of the given pins of an 8237 DMA Controller during a DMA read cycle:

i. $\overline{IOR}$ ii. $\overline{IOW}$ iii. $\overline{MEMW}$ iv. $\overline{MEMR}$

⇒ DMA will read data from memory and write the data to a peripheral device.

i. $\overline{IOR}$ ⇒ High
ii. $\overline{IOW}$ ⇒ Low, since writing to peripheral
iii. $\overline{MEMW}$ ⇒ High
iv. $\overline{MEMR}$ ⇒ Low, since reading from memory

8. Describe the tasks of the given pins of the 8237 DMA Controller:
⇒ Study from slide

9. Intel. decided to create a new version of the 8237A DMA controller called “DMA 6” which supports 6 channels instead of the traditional 4 channels and has 20 address pins. Additionally, the size of the port addresses are now changed to 20 bits. A special register in 8086 named “QX” keeps the value of the 20 bits port address when using Variable Addressing. But for fixed addressing assume the size of the P8 byte remains the same.

(a) Deduce the maximum number of I/O devices the DMA 6 can handle.

⇒ Each channel of the device is capable of driving a specific I/O device. Thus the “DMA 6” chip will be able to handle 6 I/O devices at a time.

(b) Calculate the total size of the memory that the DMA 6 is capable of addressing

⇒ Number of address bits determine the addressing capability of the chip. Since there are 20 address pins, the DMA 6 chip will be able to address 2^{20} = 1MBytes of memory

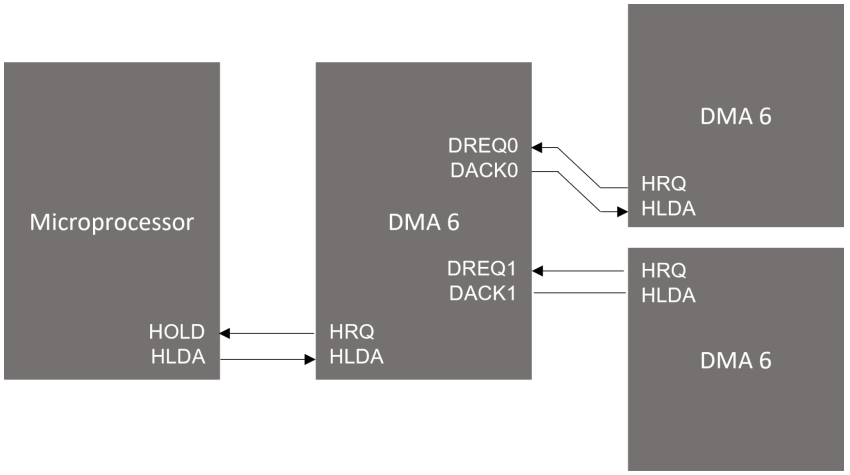
(c) Deduce the new range of addressing space for fixed addressing and variable addressing if Isolated I/O is used for I/O address mapping

⇒ For fixed addressing, P8 byte size value is used, so the fixed addressing method will allow only 256 memory locations (00000h - 000FFh)

For variable addressing, the special 20 bit port address register QX is used, so variable addressing method will allow using the entire 1Mbyte RAM (00000h - FFFFFh)

(d) Assume DMA 6 is in cascading mode. Illustrate using a diagram how 2 secondary DMA 6’s can be connected with one primary DMA 6. Also, explain the total process of how data transfer takes place between an Intel 8086 and the cascaded DMA 6’s.

⇒ For the data transfer process, refer to slide



10. Which mode does the 8086 microprocessor operate in when a DMA controller is used in the system and connected to the 8086 using HOLD and HLDA pins? (Minimum mode/Maximum mode)
⇒ Minimum mode

11. For data transfer the following scenarios, among programmed I/O, Interrupt based I/O and DMA based I/O, which one would you prefer and why?

a. Matrix keypad connected to a door locking mechanism

⇒ We can use programmed I/O as matrix keypads are not required to be very high speed/ instantaneous.

b. Power outage detection in computers

⇒ Interrupt based I/O should be used so that the CPU can handle these special scenarios immediately.

c. Transferring data from Hard disk to RAM.

⇒ Requires high speed transmission in bulk without affecting processing speeds, so DMA should be used.

12. How many I/O ports does the 8086 processor support in the isolated I/O method?

⇒ 65536 (2^16)

13. Compare the advantages and disadvantages of Memory mapped I/O vs Isolated I/O.

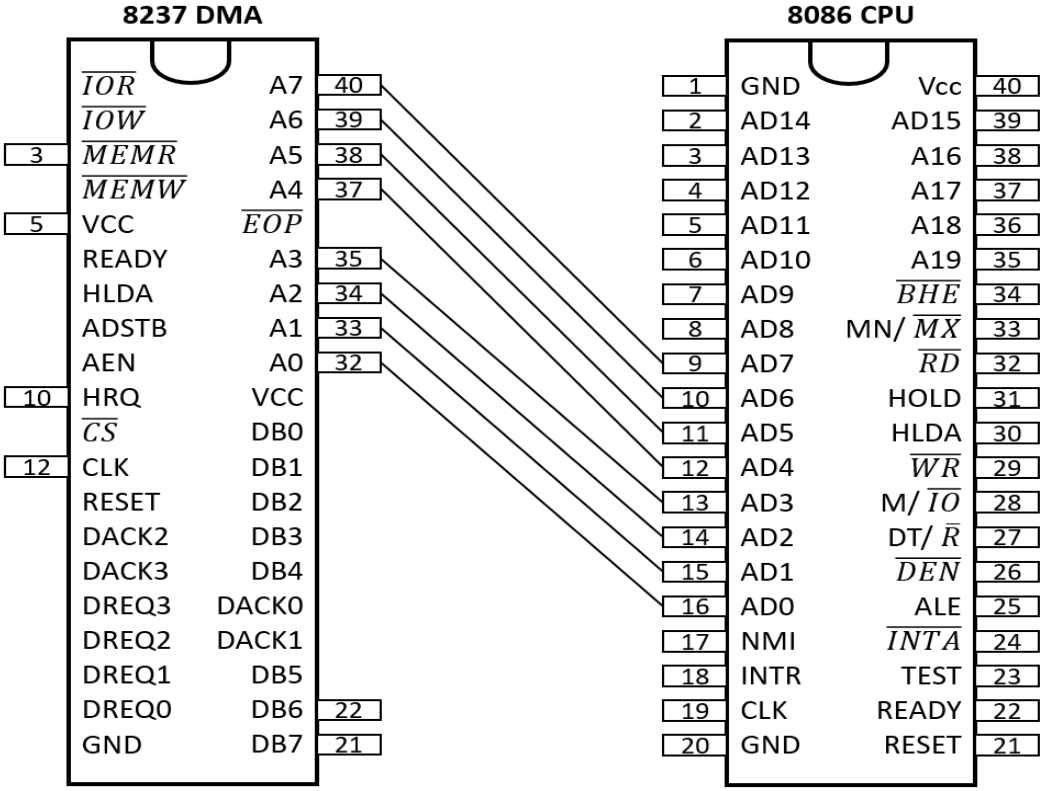
⇒ Refer to slides

14. A particular I/O device utilizes the MOV operation to take input from a temperature sensor. Is the device memory-mapped or isolated?

⇒ Using MOV operation implies memory mapped I/O

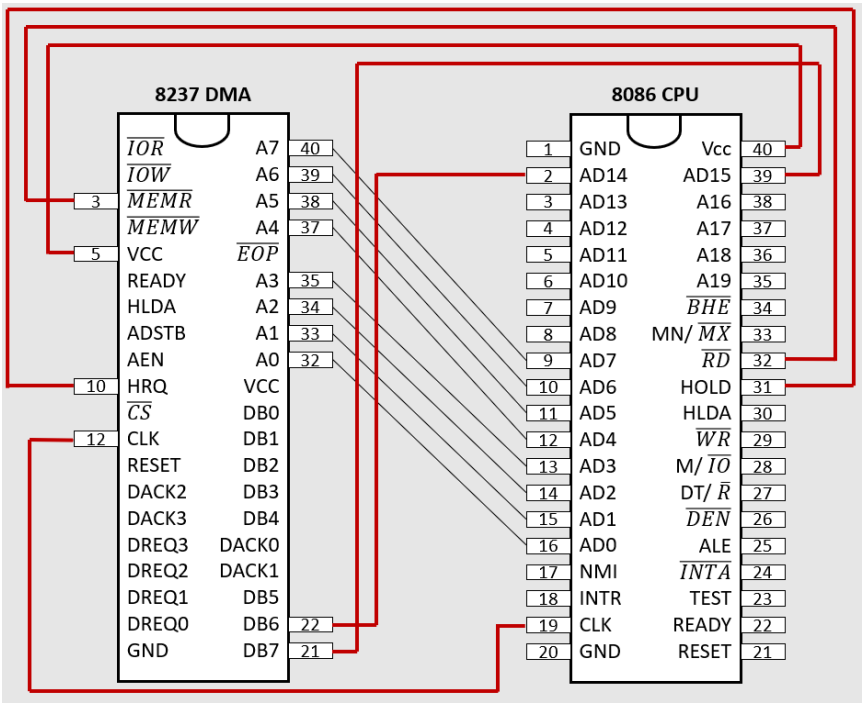
Using IN/OUT operation implies isolated I/O

15.



a. Based on your knowledge of the operation of the DMA chip, connect the six free pins of the 8237 IC to the appropriate pins of the 8086 IC. The address bus of the two ICs has been connected as a demonstration. Note that DB(0-7) pins of 8237 carry the higher byte of the 16-bit memory address. Assume both chips run on the same clock. Also if two pins have the same source, connect them with each other.

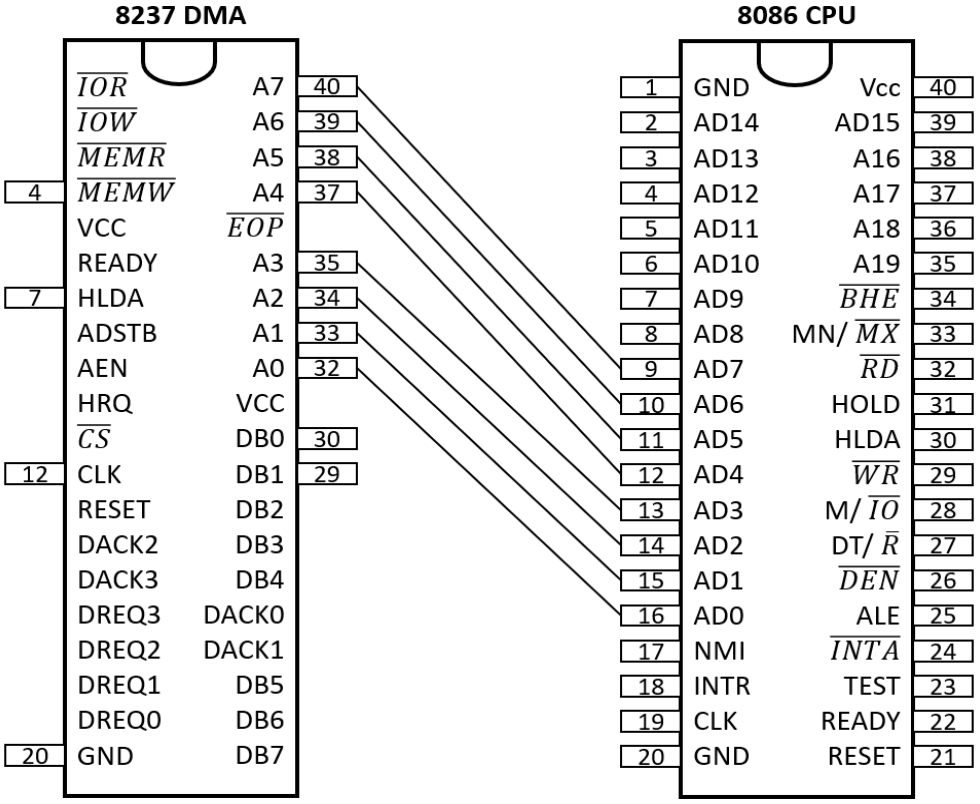
⇒



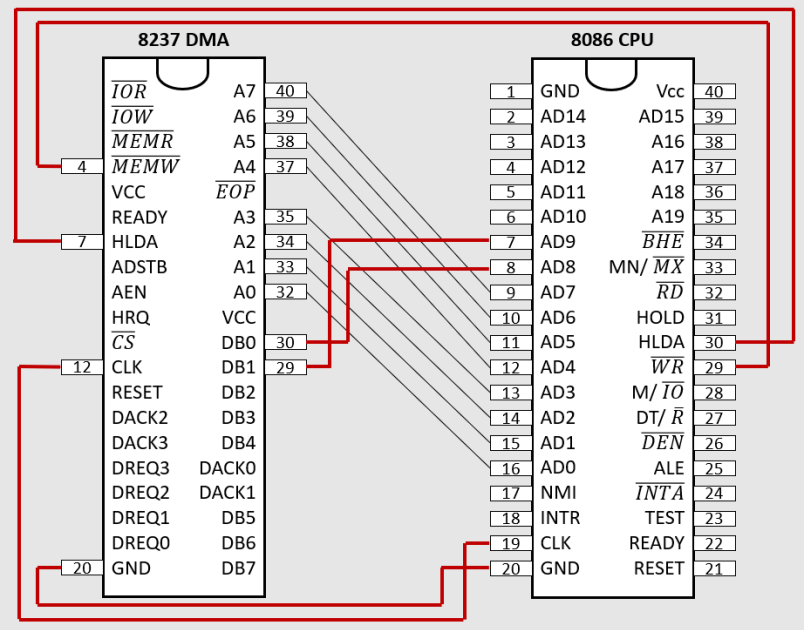
b. “An 8086-based system can support input-output devices in the memory mapped mode and isolated mode simultaneously.” Do you agree? Explain with reasoning.

⇒ Yes, memory mapped I/O uses the MOV operation, Isolated I/O uses IN/OUT operations, so we can use both in the same system.

16.



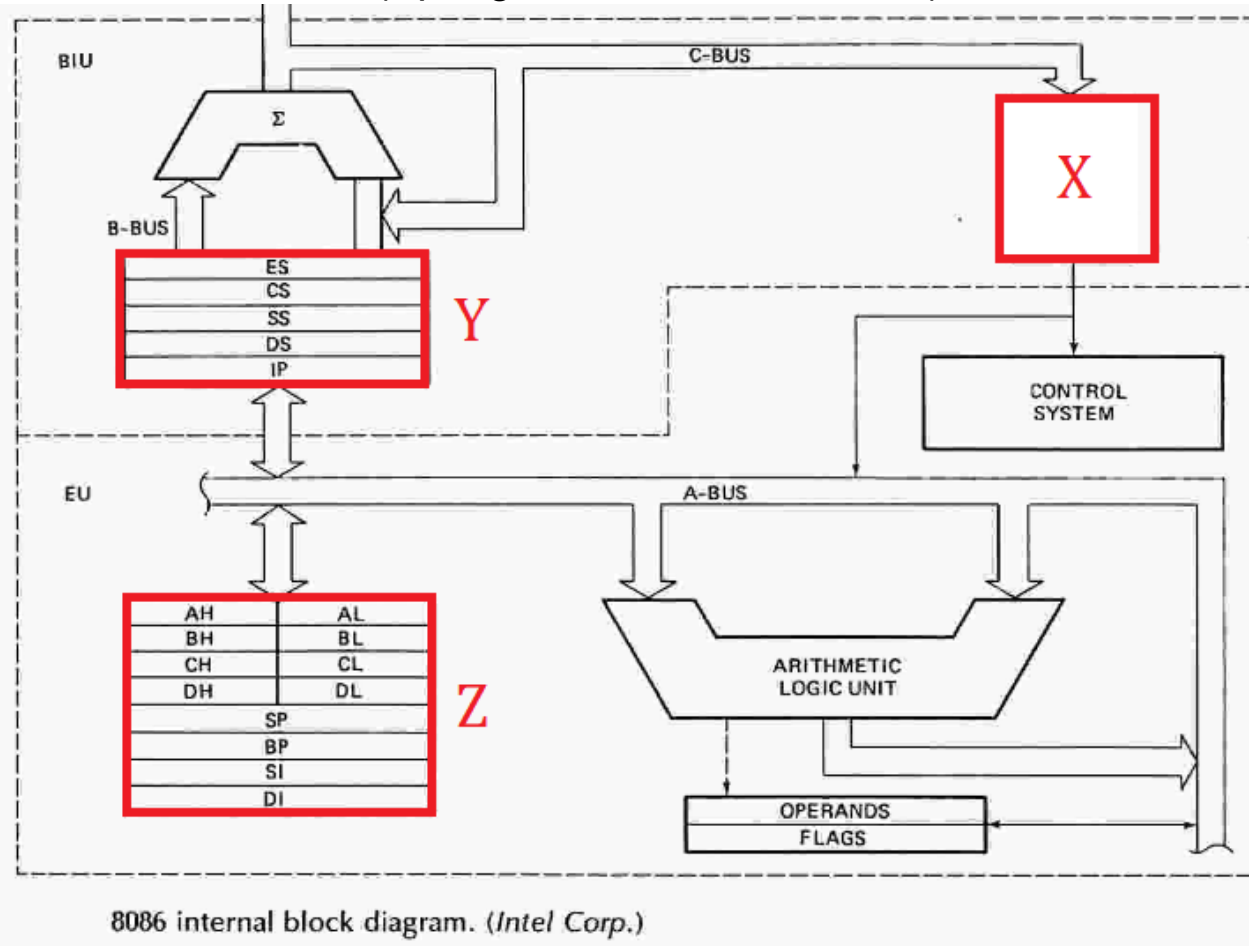
a. Based on your knowledge of the operation of the DMA chip, connect the six free pins of the 8237 IC to the appropriate pins of the 8086 IC. The address bus of the two ICs has been connected as a demonstration. Note that DB(0-7) pins of 8237 carry the higher byte of the 16-bit memory address. Assume both chips run on the same clock. Also if two pins have the same source, connect them with each other.
⇒



b. “By using memory mapped I/O, we can access a greater number of input/output ports compared to the isolated I/O method.” Do you agree? Explain with reasoning.
⇒ Yes, Memory mapped I/O uses the MOV operation, allowing us to use the addresses of the entire memory space (1MByte). Isolated I/O is restricted to 16 bit addressing and the IN/OUT operations, restricting us to 64KByte. Thus using memory mapped I/O we can access a greater number of I/O ports.

Quiz Questions

Q1 (Spring 2023 Section 5 Set-A)



8086 internal block diagram. (*Intel Corp.*)

⇒ 1. (a) X, Y, and Z are pre-fetch queue, segment registers, and General purpose+pointer/index registers, respectively.

(b)

2. 2kB = 2^11 = 2047 h

Ending address = 1B800 h + 2047 h =

3. Physical address = segment address * 10h + offset address

This physical address = CS * 10h + IP

CS is given. Calculate IP.

Q1 (Spring 2023 Section 7 Set-A)

Name : _____

ID : _____

A microprocessor has a 26 Bit Address Bus.

1. What is the memory capacity in megabytes?

   ⇒ Memory capacity = 2^{26} Bytes

   1 Kilobyte = 1024 byte = 2^{10} byte

   1 Megabyte = 1024 Kilobyte = $2^{10} \times 2^{10}$ byte = 2^{20} byte

   $2^{26} byte = 2^{26} / 2^{20}$ Megabyte = 2^{6} Megabyte = 64 Megabyte

2. What is the physical address (in hexadecimal) of the first memory location?

   ⇒ Physical address of first memory location

   Bin: 00 0000 0000 0000 0000 0000 0000

   Hex: 0000000 (7 Hex digits required for the address)

3. What is the physical address (in hexadecimal) of the final memory location?

   ⇒ Physical address of last memory location

   Bin: 11 1111 1111 1111 1111 1111 1111

   Hex: 3FFFFFF (7 Hex digits required for the address)

For each of the following registers, write down which functional unit it belongs to

4. SS	Bus Interface Unit
5. IP	Bus Interface Unit
6. SI	Execution Unit

The register values are given for the following instruction >>> MOV AL, [SI]
DS = **1000h**, CS = **3000h**, SS = **8A40h**, SI = **2000h**, DI = **030Fh,** IP = **1234h**, SP = **0010h**

7. What is the physical address of the source data?

   ⇒ In the instruction, source data is [SI]

   Logical address of the source data > DS:SI = 1000h:2000h

   Physical address of the source data = 1000h x 10h + 2000h = 12000h

8. What is the logical address of the current instruction?

   ⇒ The logical address of the current instruction is given by CS:IP = 3000h:1234h

9. Consider the command >>> PUSH AL. What is the physical address of the location where the value of AL is stored?

   ⇒ During the push command, the operand is stored at the stack memory location pointed by SS:SP

   Thus, the logical address of the push destination = 8A40h:0010h

   Physical address of the destination = 8A40h x 10h + 0010h = 8A410h

A certain data is stored at the location **FFFAh:FFCAh** of the RAM of the 8086 processor (overlapping segmentation)

10. What is the physical address of the mentioned data?

    ⇒ The physical address = FFFA0h + FFCAh = 10FF6Ah

    Memory segmentation of the 8086 processor is done in a wrap-around fashion, meaning that if a physical address overflows the 20 bit memory length, it just starts again from the starting address. So we just ignore the overflow.

    Thus, the physical address = 0FF6Ah

Considering overlapping segmentation, fill up the following table:

	Segment	Offset	Physical Address
11.	**6900h**	**0420h**	69420h
12.	**0100h**	EFFFh	**FFFFh**
13.	1F20h	**0109h**	**1F309h**
14.	Invalid	**ABC1h**	**C2105h**

Free marks will be provided for 15-20 since flags for SUB are not covered under the lectures.

Q1 (Spring 2023 Section 7 Set-B)

Name : _____

ID : _____

A microprocessor has a 28 Bit Address Bus.

1. What is the memory capacity in megabytes?

   ⇒ Memory capacity = 2^{28} Bytes

   1 Kilobyte = 1024 byte = 2^{10} byte

   1 Megabyte = 1024 Kilobyte = $2^{10} \times 2^{10}$ byte = 2^{20} byte

   $2^{28} byte = 2^{28} / 2^{20}$ Megabyte = 2^{8} Megabyte = 256 Megabyte

2. What is the physical address (in hexadecimal) of the first memory location?

   ⇒ Physical address of first memory location

   Bin: 0000 0000 0000 0000 0000 0000 0000

   Hex: 00000000 (7 Hex digits required for the address)

3. What is the physical address (in hexadecimal) of the final memory location?

   ⇒ Physical address of last memory location

   Bin: 1111 1111 1111 1111 1111 1111 1111

   Hex: FFFFFFF (7 Hex digits required for the address)

For each of the following registers, write down which functional unit it belongs to

4. DI	Execution Unit
5. ES	Bus Interface Unit
6. BX	Execution Unit

The register values are given for the following instruction >>> MOV AL, [DI]
DS = **2000h**, CS = **4000h**, SS = **8A40h**, SI = **2000h**, DI = **030Fh,** IP = **F234h**, SP = **0020h**

7. What is the physical address of the source data?

   ⇒ In the instruction, source data is [DI]

   Logical address of the source data > DS:DI = 2000h:030Fh

   Physical address of the source data = 2000h x 10h + 030Fh = 2030Fh

8. What is the logical address of the current instruction?

   The logical address of the current instruction is given by CS:IP = 4000h:F234h

9. Consider the command >>> PUSH AL. What is the physical address of the location where the value of AL is stored?

   ⇒ During the push command, the operand is stored at the stack memory location pointed by SS:SP

   Thus, the logical address of the push destination = 8A40h:0020h

   Physical address of the destination = 8A40h x 10h + 0020h = 8A420h

A certain data is stored at the location **FFEBh:FF1Dh** of the RAM of the 8086 processor (overlapping segmentation)

10. What is the physical address of the mentioned data?

    The physical address = FFEB0h + FF1Dh = 10FDCDh

    Memory segmentation of the 8086 processor is done in a wrap-around fashion, meaning that if a physical address overflows the 20 bit memory length, it just starts again from the starting address. So we just ignore the overflow.

    Thus, the physical address = 0FDCDh

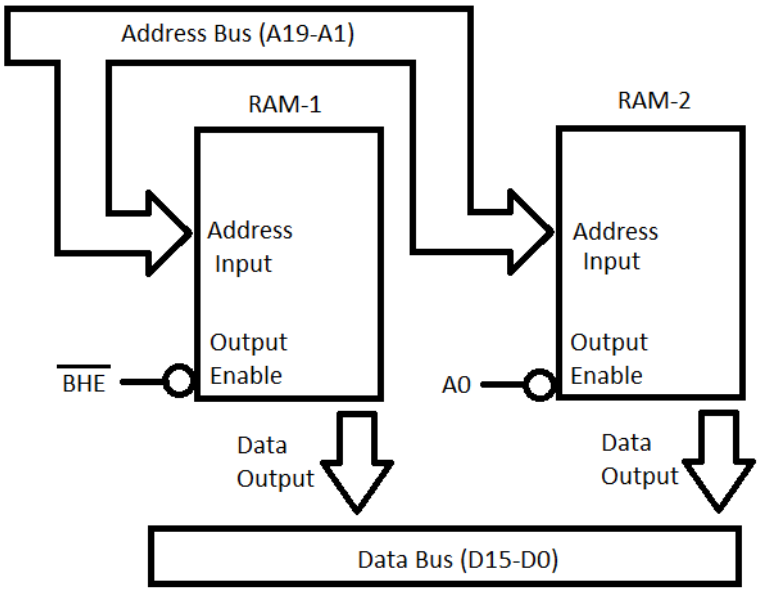
Considering overlapping segmentation, fill up the following table:

	Segment	Offset	Physical Address
11.	**4200h**	**0069h**	42069h
12.	**0200h**	DFFFh	**FFFFh**
13.	2F10h	**0209h**	**2F309h**
14.	972Dh	**AE31h**	**A2101h**

Free marks will be provided for 15-20 since flags for SUB are not covered under the lectures.

Q2 (Spring 2023 Section 7 Set-A)

Two individual RAM modules are connected to the 8086 processor in the following way to enable word size data transfer in a single bus-cycle:



1	For the given connection, identify whether RAM-1 is even bank or odd bank	Odd bank
2	What is the memory size (in kilobytes) of the RAM-1 memory module?	512 KBytes
3	Which bits of the data bus are the data output pins of the RAM-1 module connected to?	D15-D8
4	Which bits of the data bus are the data output pins of the RAM-2 module connected to?	D7-D0

Consider the following two scenarios:

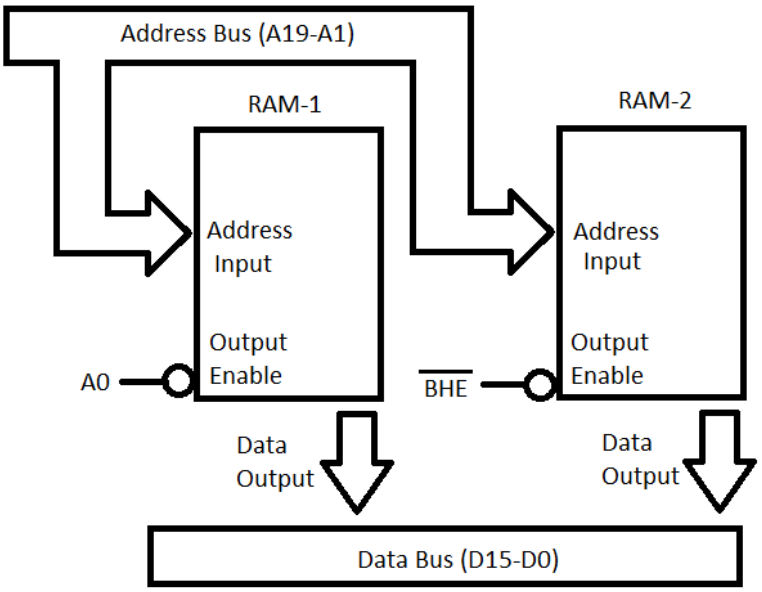
SCENARIO 1: DS = 0000H	**SCENARIO 2: DS = 0000H**
(a) MOV BX, *last_4_digits_of_your_ID* (b) MOV AL, [BX] (c) MOV AH, [BX+1]	(a) MOV BX, *last_4_digits_of_your_ID* (b) MOV AX, [BX]

5	For scenario 1, instruction (b), which memory bank does the data come from? ⇒ If last digit is even: data comes from even bank ⇒ If last digit is odd: data comes from odd bank
6	For scenario 1, instruction (b), what are the values of A0 and $\overline{BHE}$? ⇒ If last digit is even: even bank: A0 = 0 and $\overline{BHE}$ = 1 ⇒ If last digit is odd: odd bank: A0 = 1 and $\overline{BHE}$ = 0
7	For scenario 1, instruction (c), which memory bank does the data come from? ⇒ If last digit is even: data comes from odd bank ⇒ If last digit is odd: data comes from even bank

8	For scenario 1, instruction (c), what are the values of A0 and $\overline{BHE}$? ⇒ If last digit is even: odd bank: A0 = 1 and $\overline{BHE}$ = 0 ⇒ If last digit is odd: even bank: A0 = 0 and $\overline{BHE}$ = 1
9	After the execution of the given instructions, does the AX register contain the same data in both scenarios? (Yes/No) ⇒ Yes, since AH gets high byte and AL gets the low byte
10	Do scenario 1 and scenario 2 require the same number of bus cycles for the given instructions? (Yes/No) => If last digit is even: no (scenario 1 takes 3 bus cycles and scenario 2 takes 2 bus cycles since un-aligned) ⇒ If last digit is odd: yes (Both scenarios take 3 bus cycles since data is un-aligned

Q2 (Spring 2023 Section 7 Set-B)

Two individual RAM modules are connected to the 8086 processor in the following way to enable word size data transfer in a single bus-cycle:



1	For the given connection, identify whether RAM-1 is even bank or odd bank	Even bank
2	What is the memory size (in kilobytes) of the RAM-1 memory module?	512 KByte
3	Which bits of the data bus are the data output pins of the RAM-1 module connected to?	D7-D0
4	Which bits of the data bus are the data output pins of the RAM-2 module connected to?	D15-D8