Q1. Using 8085, explain the stack operation; where initial SP value is FFFEh. **03**

Q2. Suppose, Instruction Pointer (IP) of 8086 is moving in backward memory directions for executing following assembly language codes: **12**

```
MOV AX, FA1Ah
NEG AX
ADD AX, A1h
```

After the executions of all arithmetic and logical instructions in single-step debug mode with disabling interrupt option, what values would be there in AX and Flag (CF, PF, AF, ZF, SF, OF, IF, TF, and DF) registers?

Q1. Derive the machine code for the instruction: **IN AL, F1 h**. Also, show how the derived contents can be stored in different memory locations. **06**

Q2. Your given Student Id format is **20 . . XXXX**. Use last 4 (four) digits of your id as a hexa-decimal value (i.e., **XXXXh**) and consider the following table. **09**

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

Now, using the table derive the machine code contents of the following MOV instructions and show how many memory locations are required to store the derived machine codes:

i. `MOV AL, [XXXXh]`

ii. `MOV SS:[BX+DI+ XXXXh], DX`

iii. `MOV BX, XXXXh`

.....

Q1. Draw the details of READ Bus timing diagram showing all the necessary/required signals of 8086. You should consider that the read operation will be made from a memory address of 01111H and there are 4 WAIT states due to an NMI interrupt. **7**

Q2. Suppose, while debugging an assembly language program the values of the registers are: `Flag=FEB9h, IP=0102h, CS=0700h, SP=FFFAh`. Now, if `INT 00100001B` is requested, derive the memory addresses from where the new `IP` and `CS` can be retrieved; Also show the step-by-step changes in memory contents of stack segment along with corresponding `SP` values while handling the interrupt by the 8086 microprocessor. **8**