1 (6 pts) If ES=1132h and DS=1200h and BX=F106h, while executing
\[ 1300:4128 \text{ MOV AX, [ES:BX]} \]
   a. What is the Physical Address Location that is accessed for the data?
   \[ 11320 + F106 = 20426h \]
   b. What is the Logical Address Location that is accessed for the data?
   \[ 1132:F106h \]
   c. What is the Physical Address Location that is accessed for this instruction's
      opcode byte?
   \[ 17128h \]

2 (10 pts) Without using the Parity Flag information complete the following program so
there are Even Number of 1's in AX, the program Jumps to label EVEN otherwise it jumps
label ODD.

\[
\begin{align*}
\text{MOV AX, [SI]} \\
\text{MOV CL, 10h} \\
\text{MOV BL, 00h} \\
\text{; COMPLETE the dots in the following 4 lines first!} \\
\text{BACK: ROR .... AX, 1h} \\
\text{... JNC SKIP} \\
\text{... INC BL} \\
\text{... SKIP: DEC CL} \\
\text{JNZ BACK} \\
\text{; THEN COMPLETE THE rest of the PROGRAM BELOW}
\end{align*}
\]
3 (10 pts) Write a procedure called MINSIGNED to find the Smallest of all Signed Numbers in memory starting at VALUES. The last element in the array is ‘$’. The Signed Comparison instructions (JG, JL etc.) must be used in your program during signed comparison. The result should be in the AL register.

VALUES db 23h, 15h, 59h, 81h, 7fh, 13h, 31h, '$'
.code  ; smallest is 81h
MINSIGNED PROC
  MOV AX, @data
  MOV DS, AX
  mov si, offset message
  MOV AL, 7Fh
  BACKR: CMP [si], '$'
  JZ OUTR
  CMP [si], AL
  JG SKIP
  ; found new smaller
  MOV AL, [SI]
  SKIP: INC SI
  JMP BACKR

  outr: ; the result should be in AL
  mov ax, 4c00h
  int 21h
MINSIGNED endp

4. (7 pts) Complete the following program to divide the unsigned value in the AL register by the unsigned value in the BL register and store the quotient and remainder (result of the division) in the 29ABCh and 29ABDh memory locations respectively.

MOV AX, 1000h
MOV DS, AX
MOV AL, [1234h]
MOV BL, [5678h]
Mov AH, 00h
DIV BL
MOV CX, 2000h
MOV DS, CX
MOV [9ABCh], AL
MOV [9ABDh], AH
5. (5 pts) Find and store the Auxiliary Flag (AF) value into the D0 bit of the BL register after executing ANY operation. You MUST use the LAHF operation to find the AF. The LAHF instruction is defined as follows:

LAHF -- Load Flags into AH Register

Description | D7 D6 D5 D4 D3 D2 D1 D0
--- | --- | --- | --- | --- | --- | --- | --- | ---
Load: AH = flags | SF ZF xx AF xx PF xx CF

: any operation here
LAHF
AND AH, 10h
Jz sifir
MOV BL, 01h
Sifir:
MOV BL, 00h

6. (15 pts) Complete the code below to convert the BINARY NUMBER < 64h in memory location 01234H, into its DECIMAL equivalent value and store this number in AL register. For example 29h => 41d.

MOV AX, 0000h
MOV DS, AX
MOV AL, [1234h]
MOV CL, 0Ah
DIV CL
; quotient in AL, rem in AH
XCHG AL, AH
MOV CL, 4
ROL AL, CL
KOR AX, CL

7. (10 pts) Find how many times this program will execute the NOP instruction in the following code below:

MOV DS, AX
MOV CX, 20H
YZ: PUSH CX
MOV CX, 10H
BC: NOP
DEC CX
MOV AL, CL
XOR CH, AL
JNZ ABC
POP CX
DEC CX
LOOP XYZ
8. (8pts) a) The address range to be decoded for four 8K x 8 EPROMs start from B8000h. Fill in the memory map for this architecture. Write also their hex representations on the left column.

<table>
<thead>
<tr>
<th>Hex</th>
<th>A19</th>
<th>A18</th>
<th>A17</th>
<th>A16</th>
<th>A15</th>
<th>A14</th>
<th>A13</th>
<th>A12</th>
<th>A11</th>
<th>A10</th>
<th>A09</th>
<th>A08</th>
<th>A07</th>
<th>A06</th>
<th>A05</th>
<th>A04</th>
<th>A03</th>
<th>A02</th>
<th>A01</th>
<th>A00</th>
</tr>
</thead>
<tbody>
<tr>
<td>B8000H</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BA0000</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>BB0000</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>BC0000</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>BD0000</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>BE0000</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>BF0000</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>BF0000</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

b) (4pts) Write the necessary instructions to access the last possible memory location in the memory for this architecture in (a) and move that byte into the BL register.

```
MOV AX, BFFFh
MOV DS, AX
MOV SI, FFFPh
MOV BL, OFFSET SI
```
9. (15 pts)
Two BCD numbers < 10 are entered from the keyboard. The procedure is as follows:

a) Input these numbers from the keyboard using INT 16, AH=00h \(\text{(given below)}\) 

b) Convert each number to unpacked BCD’s and multiply them together then add multiplication.

c) Divide this number by 6, quotient in AL and keeping the remainder in AH.

d) Finally display the quotient using INT 10, AH=0Eh

An example flow and hints: for example **numbers 3 and 9** are entered from the keyboard, you must convert them into unpacked BCD’s first. When they are multiplied and AAM the result is stored in the AX register as 0207. Then this number is converted to division using AAD. Divided by 6, the result 4 should be displayed on the screen as \((3\times9)/6\) where 3 is in AH register (not displayed) as the remainder. Your code should be as minimal as possible. You must follow the given procedure and use AAM and AAD.

```
Mov ah, 00h
Int 16h
Mov cl, al
Mov ah, 00h
Int 16h
Mov ah, cl; complete your code below

Sub 3030h ; strip off ascii 30
Mul ah
AAM
Mov cl, 6
Aad
Div cl
Add al, 30h
Mov ah, 0eh
Int 10h
```