

**HACETTEPE UNIVERSITY**

**ELECTRICAL AND ELECTRONICS ENGINEERING**

**ELE338 MICROPROCESSOR ARCHITECTURE AND PROGRAMMING LAB.**

**PRELIMINARY WORK 3**

**PROCEDURE USAGE AND STACK OPERATIONS**

**2020-2021 SPRING**

**Student**

**Name: Egemen Can**

**Surname: Ayduğan**

**ID: 21728036**

**Date: 11.04.2021**

**1. Question**

**EMU 8086 CODE**

|  |
| --- |
| **.Model Small**  **.Stack 64**  **.Data**  **String1 DB 'The Offset is: ','$'**  **NextRow DB 0Ah,0DH,'$' ; To the next row**  **Numbers DW 1230h,1F05h,045Ah,0B0ACh,4708h ; My Values**  **.Code**    **Proc Main**    **Mov AX,Data**  **Mov DS,AX**    **LEA SI,Numbers ; I take the value in DATA Segement.**  **MOV AX,[SI] ; And put the value in AX Register.**  **Mov BL,00h ; This is my counter.(Highest-->Lowest)**  **Mov BH,05h ; Loop counter for retrieving numbers from memory**    **Tımes:**  **Call Foffset ; I call the Foffset.**    **ADD SI,2 ; I add 2 to SI because to move to the next number.**  **MOV AX,[SI] ; The reason I add 2 is because its number is word type.**  **; And put the new value in AX Register.**  **DEC BH ; Loop Counter**  **CMP BH,00h ; To finish the code when the loop is over.**  **JE Over**  **JMP Tımes ; Return to get a new number .**    **Mov AH,004Ch ; To stop program.**  **Int 21h**    **Endp Main**  **Proc Foffset**    **RETURN: ; To examine the next bit**  **Inc BL ; I ıncrease my counter to find offset.**  **MOV DI,0h ; To reset 4 bıts**  **MOV CX,4h ; To control 4 bits each time because of 4 BINARY = 1 HEX**    **TRY:**  **SHL AX,1d ; And ı shift the number 1 bit to the left.**  **JC HERE ; IF CF=1,it jumps to HERE**  **JCXZ COMP ; If CX =0,ıt jumps to COMP**  **DEC CX ; 4 Bit Counter**  **CMP CX,00h ; If the counter finish, CF=1**  **JNE TRY ; If CX does not 0,it jumps to TRY**    **COMP:**  **CMP DI,00h ; This is number controller.**  **; If the number is 0, the offset has been found.**  **JNE RETURN ; If number isn’t 0, it jumps to RETURN and starts to examine the next number**  **JE FINISH ; If the number is 0, it jumps to Fınısh.**    **HERE: ; The part that checks the 1 bit.**  **INC DI ; IF DI increases, this number is not 0.**  **DEC CX**  **CMP CX,00h ; If the counter finish, CF=1**  **JE COMP ; IF CF=1, it jumps to COMP to control number finally.**  **JMP TRY**      **FINISH:**  **Mov AH,09h**  **Mov DX,OFFSET String1 ; To show string on the screen**  **Int 21h**    **Add BL,48d ; I add 48 to the counter to show the number on the screen.**  **Mov DL,BL**    **Mov AH,02h ; Print the offset**  **Int 21h**    **Mov AH,09h**  **Mov DX,OFFSET NextRow ; To pass the next row.**  **Int 21H**    **Mov BL,0h**  **Mov DL,0h ; I have to make Registers 0 for the next number.**  **Mov AX,0h**  **Ret**  **Endp Foffset**  **Over:**  **Ends** |

**RESULTS**

|  |
| --- |
| **In order, it finds the offsets in the numbers.** |

**COMMENT**

|  |
| --- |
| I entered 5 numbers in the data section. I put my procedure in a loop that will cycle 5 times. In this loop, it will do the operation for each number, then move on to the next number and do the operation for that number as well. My code worked 5 times with procedure.  In the procedure part, I examined the number I entered bit by bit. And I grouped these bits in 4. If the bit is 0, I set it to the beginning of the loop, if bit is 1, I incremented my DI recorder by 1. After checking 4 bits, I checked my DI value with the CMP command. If my DI value is 0, I found my value 0 in the number. I code the offset of the number into the BL Register. Increments 1 for every 4 bit check. When 0 is found, it gives me the offset. Finally, I printed this on the screen. |

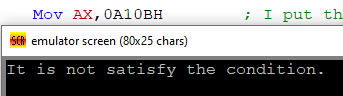
**2. Question**

**EMU8086 CODE**

|  |
| --- |
| **.Model Small**  **.Stack 64**  **.Data**  **String1 DB "It is not satisfy the condition.","$"**  **.Code**  **Proc Main**  **Mov AX,Data**  **Mov DS,AX**    **Mov AX,0A00BH ; I put the value in AX Register**  **Call Funct ; Call the procedure**  **Endp Main**  **Proc Funct**  **Push AX ; To saves it to use later**  **Rol AX,4 ; To checks the middle nibbles are zero**  **Cmp AH,00h**  **Je Exists ; If the middles are zero, it jumps to Exists**  **Jne NotExists ; If the middles are not zero, it jumps to NotExists**    **Exists:**  **Pop AX ; Pop the first value AX**  **Xchg AH,AL ; Exchanges the AL and the AH**  **Ror AL,4 ; Rotates the AL**  **Rol AH,4 ; Rotates the AH**  **Jmp Finish ; The reason for rotating 4 times is that the hex value is 4 bits.**    **NotExists:**  **Mov AH,09h**  **Mov DX,OFFSET String1 ; If the middles are not zero**  **Int 21h ; To print the screen "It is not satisfy the condition.".**  **Finish:**  **Endp Funct**  **End** |

**RESULTS**

|  |  |
| --- | --- |
|  |  |
|  |  |

****

**COMMENT**

|  |
| --- |
| I put the value in AX Register. And I called the procedure. My procedure part consists of 2 parts. The first part checks whether the number is exists or not exists. The second part does its operation if the number is exists.  In my second part, I put the AX value, which ı previously held with push, with pop. Then I swapped the AH and AL values with the XCHG command. Then I rotated the new AL value to the right and the new AH value to the left. And finally I found the value I wanted and finished the code. |

**2.Question(Bonus)**

**EMU8086 Code**

|  |
| --- |
| **.Model Small**  **.Stack 64**  **.Data**  **.Code**  **Proc Main**  **Mov AX,Data**  **Mov DS,AX**    **Mov AX,0B217h ; I put the value in AX Register**  **Call Funct ; Call the procedure**  **Endp Main**  **Proc Funct**    **Ror AH,4d ; Rotates the AH**  **Ror AX,4d ; Rotates the AX**  **Ror AL,4d ; Rotates the AL**    **Endp Funct**  **End** |

**RESULTS**

|  |
| --- |
|  |

**Comment**

I put the value in AX Register and ı call the procedure. In the procedure, I rotated the AH , AX and AL in order. And finally I found the value I wanted and finished the code