**Lab Report**

**Lab – 04**

**Course No: 3110**

**Course Name: Sessional Based on 3109 (Microprocessors and Assembly Language)**

**Submitted To:**

**Sadia Zaman Mishu**

**Assistant Professor,**

**Department of CSE, RUET**

**Submitted By:**

**Anika Tabassum Era**

**Roll: 1703176**

**Section: C**

**Experiment No: 04**

**Experiment Name: Write an assembly language program that will prompt the user to enter a HEX digit character (0-9 or A-F) and display it on next line in decimal and check whether this decimal number is odd or not. If odd then the program will repeat and if even then the program will terminate. If the user enters an illegal character, prompt the user to enter another character.**

**Code:**

**.MODEL SMALL**

**.STACK 100H**

**.DATA**

**MSG1 DB 0AH,0DH, 'Enter a hex digit: $'**

**MSG2 DB 0AH,0DH, 'In decimal it is: $'**

**MSG3 DB ' is an odd number $'**

**MSG4 DB ' is an even number $'**

**MSG5 DB ' is an illegal character $'**

**.CODE**

**MAIN PROC**

**AGAIN:**

**;INITIALIZING PROMPT USER**

**MOV AX, @DATA**

**MOV DS, AX**

**; FIRST PROMPT USER MSG**

**LEA DX, MSG1**

**MOV AH,9**

**INT 21H**

**;INPUT HEX DIGIT**

**MOV AH, 1**

**INT 21H**

**MOV BL,AL**

**CMP BL, 57D ; COMPARING BL WITH ASCII OF 9**

**JLE DIG ; LESS OR EQUAL**

**JG IS ; GREATER**

**;INVALID CHECK**

**;IS HEX?**

**IS:**

**CMP BL, 70D ; CHECK IF INPUT > F**

**JG INV**

**JLE HEX**

**INV:**

**;LINE FEED**

**MOV DL,0AH**

**MOV AH,2**

**INT 21H**

**;CARRIAGE RETURN**

**MOV DL,0DH**

**MOV AH,2**

**INT 21H**

**;PRINT**

**MOV DL, 0AH**

**INT 21H**

**MOV DL ,0DH**

**INT 21H**

**MOV DL, BL**

**MOV AH,2**

**INT 21H**

**MOV AX, @DATA**

**MOV DS, AX**

**MOV AH, 9**

**LEA DX, MSG5**

**INT 21H**

**MOV AH, 2**

**MOV DL, 0AH**

**INT 21H**

**MOV DL ,0DH**

**INT 21H**

**JMP AGAIN**

**HEX:**

**SUB BL,17D ;CONVERTING THE HEX CHARACTER IN DECIMAL**

**;LINE FEED**

**MOV DL,0AH**

**MOV AH,2**

**INT 21H**

**;CARRIAGE RETURN**

**MOV DL,0DH**

**MOV AH,2**

**INT 21H**

**;SECOND PROMPT USER MSG2**

**LEA DX, MSG2**

**MOV AH,9**

**INT 21H**

**MOV DL,49D ;PRINTING "1" BEFORE BL**

**MOV AH,2**

**INT 21H**

**; PRINT THE CONVERTED VALUE THAT IS STORED IN BL**

**MOV DL,BL**

**MOV AH,2**

**INT 21H**

**;LINE FEED**

**MOV DL,0AH**

**MOV AH,2**

**INT 21H**

**;CARRIAGE RETURN**

**MOV DL,0DH**

**MOV AH,2**

**INT 21H**

**MOV DL, 0AH**

**INT 21H**

**MOV DL ,0DH**

**INT 21H**

**MOV DL,49D ;PRINTING "1" BEFORE BL**

**MOV AH,2**

**INT 21H**

**MOV DL, BL**

**MOV AH,2**

**INT 21H**

**;EVEN ODD**

**TEST BL, 1 ;A CHECK TO EVEN ODD**

**JNE ODDD**

**MOV AX, @DATA**

**MOV DS, AX**

**MOV AH, 9**

**LEA DX, MSG4**

**INT 21H**

**JMP EXIT**

**ODDD:**

**MOV AX, @DATA**

**MOV DS, AX**

**MOV AH, 9**

**LEA DX, MSG3**

**INT 21H**

**MOV AH, 2**

**MOV DL, 0DH**

**INT 21H**

**MOV DL, 0AH**

**INT 21H**

**JMP AGAIN**

**DIG:**

**SUB BL, 0D**

**;LINE FEED**

**MOV DL,0AH**

**MOV AH,2**

**INT 21H**

**;CARRIAGE RETURN**

**MOV DL,0DH**

**MOV AH,2**

**INT 21H**

**;SECOND PROMPT USER MSG2**

**LEA DX, MSG2**

**MOV AH,9**

**INT 21H**

**;PRINTS THE CONVERTED NUMBER**

**MOV AH, 2**

**MOV DL, BL**

**INT 21H**

**;LINE FEED**

**MOV DL,0AH**

**MOV AH,2**

**INT 21H**

**;CARRIAGE RETURN**

**MOV DL,0DH**

**MOV AH,2**

**INT 21H**

**MOV DL, 0AH**

**INT 21H**

**MOV DL ,0DH**

**INT 21H**

**MOV DL, BL**

**MOV AH,2**

**INT 21H**

**;EVEN ODD**

**TEST BL, 1 ;A CHECK TO EVEN ODD**

**JNE ODD**

**MOV AX, @DATA**

**MOV DS, AX**

**MOV AH, 9**

**LEA DX, MSG4**

**INT 21H**

**JMP EXIT**

**ODD:**

**MOV AX, @DATA**

**MOV DS, AX**

**MOV AH, 9**

**LEA DX, MSG3**

**INT 21H**

**MOV AH, 2**

**MOV DL, 0DH**

**INT 21H**

**MOV DL, 0AH**

**INT 21H**

**JMP AGAIN**

**EXIT:**

**;DOS RETURN**

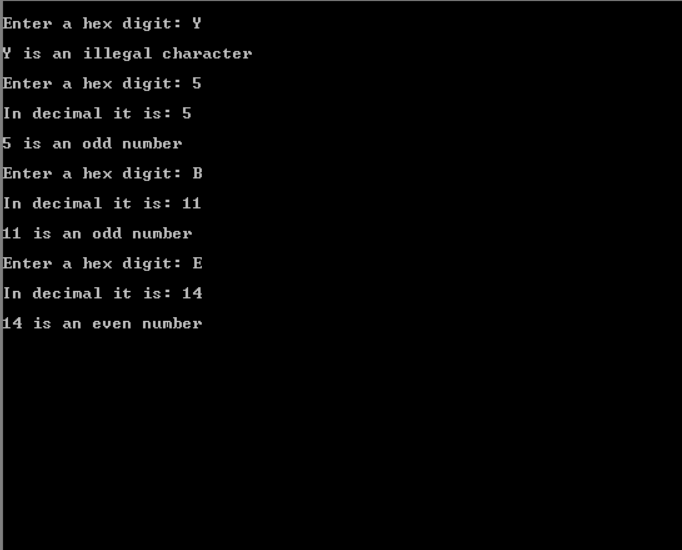
**MOV AH, 4CH**

**INT 21H**

**MAIN ENDP**

**END MAIN**

**Output:**

****

**Discussions:**

**The solution to the given problem statement was comprehensible yet, there was some difficulties while coding. It was at first tough to making the loop process functional, later on the hexadecimal-decimal conversion seemed puzzling.**

**Overall the functions are running without error.**