# AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY


**Department of Computer Science and Engineering**
Program: BSc in Computer Science and Engineering

Course Code: CSE 2214

Assignment No: 03

Date of Submission: 26 July 2025

Submitted by,
Name: Abrar Ahmad Chy
Student ID: 20230104063
Lab Section: B-1

**Question 01: Write a program that prompts the user to type a hex number of four hex digits or less, and outputs it in binary on the next line. If the user enters an illegal character, he or she should be prompted to begin again. Accept only uppercase letters. Your program may ignore any input beyond four characters.**

**Solution**:

```
.MODEL SMALL
.STACK 100H
.DATA
M1 DB 0AH,0DH,'TYPE A HEXA NUMBER (0 - FFFF) : $'
M2 DB 0AH,0DH,'IN BINARY IT IS : $'
M3 DB 0AH,0DH,'ILLEGAL HEXA DIGIT, TRY AGAIN : $'

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

START:
    MOV AH, 9
    LEA DX, M1
    INT 21H

    XOR BX, BX        ; clear BX to store 16-bit value
    MOV CL, 4         ; shift left by 4 bits (hex digit)

READ_CHAR:
    MOV AH, 1
    INT 21H           ; read char into AL

    CMP AL, 0DH
    JE SHOW_RESULT    ; if Enter pressed, go to show result

    CMP AL, '0'
    JB INVALID
    CMP AL, '9'
    JBE HEX_TO_NUM

    CMP AL, 'A'
```

```
    JB INVALID
    CMP AL, 'F'
    JA INVALID

    SUB AL, 37H       ; 'A'=65, 65-55=10 -> convert to 10-15
    JMP ADD_HEX

HEX_TO_NUM:
    AND AL, 0FH       ; convert '0'-'9' to 0-9

ADD_HEX:
    SHL BX, CL        ; shift 4 bits left to make space
    OR BL, AL         ; add new hex digit
    JMP READ_CHAR

INVALID:
    MOV AH, 9
    LEA DX, M3
    INT 21H
    JMP START

SHOW_RESULT:
    MOV AH, 9
    LEA DX, M2
    INT 21H

    MOV CX, 16        ; 16 bits
    MOV AH, 2

PRINT_BINARY:
    SHL BX, 1         ; MSB goes to CF
    JC PRINT_ONE
    MOV DL, '0'
    INT 21H
    JMP LOOP_NEXT

PRINT_ONE:
    MOV DL, '1'
    INT 21H
```

```
LOOP_NEXT:
    LOOP PRINT_BINARY

    MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

**Question 02:**

**Write a program that prompts the user to enter two unsigned hex numbers, 0 to FFFFh, and prints their sum in hex on the next line. If the user enters an illegal character, he or she should be prompted to begin again. Your program should be able to handle the possibility of unsigned overflow. Each input ends with a carriage return.**

**Solution**:

```
.MODEL SMALL
.STACK 100H

.DATA
M1 DB 0AH, 'TYPE A HEXA NUMBER 0 - FFFF : $'
M2 DB 0AH, 'THE SUM IN HEXA IS $'
COUNTER DB 4
NUM     DW ?

.CODE
MAIN PROC
    MOV AX, @DATA
    MOV DS, AX

    ; --- Prompt for first number ---
    MOV AH, 9
    LEA DX, M1
    INT 21H

    CALL READ        ; read first number
```

```
    MOV NUM, BX      ; store in NUM

    ; --- Prompt for second number ---
    MOV AH, 9
    LEA DX, M1
    INT 21H

    CALL READ        ; read second number (in BX)

    ; --- Show result message ---
    MOV AH, 9
    LEA DX, M2
    INT 21H

    ; --- Add and show carry ---
    ADD BX, NUM      ; BX = num1 + num2
    JC SHOWCY        ; if carry

    MOV AH, 2        ; no carry
    MOV DL, '0'
    INT 21H
    JMP NEXT

SHOWCY:
    MOV AH, 2
    MOV DL, '1'
    INT 21H

NEXT:
    MOV COUNTER, 4   ; reset counter for showing 4 hex digits
    CALL SHOW

    ; --- Exit ---
    MOV AH, 4CH
    INT 21H
MAIN ENDP


; READ procedure: reads a hex number (up to 4 digits)
READ PROC
    XOR BX, BX
    MOV CL, 4
```

```
    MOV AH, 1
    INT 21H

WHILE_:
    CMP AL, 0DH
    JE END_W

    CMP AL, '9'
    JG LETTER
    AND AL, 0FH
    JMP SHIFT

LETTER:
    SUB AL, 37H       ; Convert A–F to 10–15

SHIFT:
    SHL BX, CL
    OR BL, AL
    INT 21H
    JMP WHILE_

END_W:
    RET
READ ENDP

 SHOW procedure: displays BX as 4-digit hex
SHOW PROC
    MOV CL, 4

START:
    MOV DL, BH
    SHR DL, CL
    CMP DL, 9
    JG LETTER1
    ADD DL, 30H       ; Convert 0–9 to ASCII
    JMP SHOW1

LETTER1:
    ADD DL, 37H       ; Convert 10–15 to 'A'–'F'

SHOW1:
    MOV AH, 2
```

```
    INT 21H
    ROL BX, CL        ; Rotate left to get next nibble
    DEC COUNTER
    CMP COUNTER, 0
    JNE START
    RET
SHOW ENDP


END MAIN
```