COMPUTER ORGANIZATION AND ARCHITECTURE

SEMESTER II 2020/2021

SECR2033-09

GROUP PROJECT

NAME OF STUDENTS	MATRIC NUMBER
AUM JEEVAN A/L AUM NIRANGKAR	A20EC0017
SOH ZEN REN	A20EC0152
MOHAMAD HAZIQ ZIKRY BIN MOHAMMAD RAZAK	A20EC0079

LECTURER'S NAME: DR NUR HALIZA B. ABDUL WAHAB

SUBMISSION DATE: 3RD JULY 2021

VIDEO LINK: https://youtu.be/8qLuONeiv2c

Table Of Contents

1. Member responsibilities

Member Name	Responsibilities
AUM JEEVAN A/L AUM NIRANGKAR	Create program and add comments to the program
SOH ZEN REN	Design program and fix available errors
MOHAMAD HAZIQ ZIKRY BIN MOHAMMAD RAZAK	Create layout for program and edit report

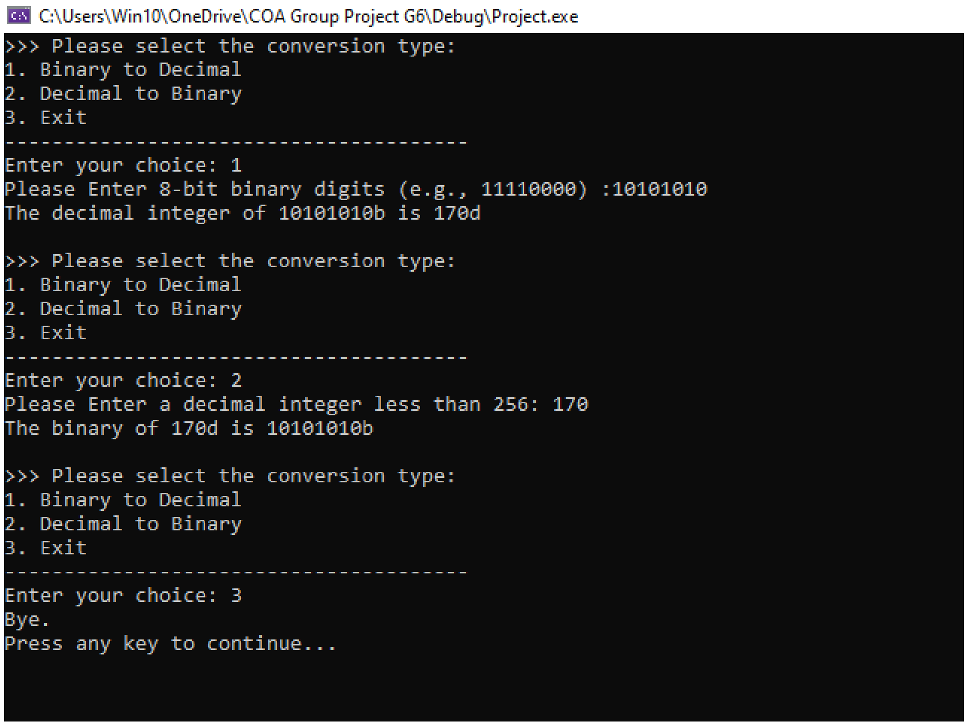
2. Coding and Explanation

```
TITLE COA GroupProject Group 6
INCLUDE Irvine32.inc
.data
        NumberLength dword ?
        binarynum byte 9 DUP(? )
        decimalnum dword ?
        str1 byte ">>> Please select the conversion type: ", 0
        str2 byte "1. Binary to Decimal", 0
        str3 byte "2. Decimal to Binary", 0
        str4 byte "3. Exit", 0
        str5 byte "---------------------------------------", 0
        prompt byte "Enter your choice: ", 0

        ; binary prompts
        binary_prompt byte "Please Enter 8-bit binary digits (e.g., 11110000) :", 0
        b2d_ans1 byte "The decimal integer of ", 0
        b2d_ans2 byte "b is ", 0

        ; decimal prompts
        decimal_prompt byte "Please Enter a decimal integer less than 256: ", 0
        d2b_ans1 byte "The binary of ", 0
        d2b_ans2 byte "d is ", 0

        ; exit prompts
        exit_prompt byte "Bye.", 0

        ; errormessage
        error_choice byte "ERROR.... Please select correct choice !", 0
        error_binary byte "ERROR.... Please enter 8-bit binary digits !", 0
        error_decimal byte "ERROR... Please enter decimal integer that less than 256 !", 0
        .code

; main menu

main PROC
        mainmenu :
                        mov edx, offset str1
                        call WriteString; Display Output Statement
                        call crlf
                        mov edx, offset str2
                        call WriteString                        ; Display option 1 in
main menu
                        call crlf
                        mov edx, offset str3
```

```
                    call WriteString                                        ; Display option 2 in
main menu
                    call crlf
                    mov edx, offset str4
                    call WriteString                                        ; Display option 3 in
main menu
                    call crlf
                    mov edx, offset str5
                    call WriteString                                        ; Display Output Line
                    call crlf

                    mov edx, offset prompt
                    call WriteString; Display "Enter Your Choice"
                    call ReadInt

                    CMP AX, 1
                    JE Bin2Dec
                    CMP AX, 2
                    JE Dec2Bin
                    CMP AX, 3
                    JE exitprog
                    JNE errormessage1

; -------------------------------------------------- ----------------------------------------
; JMP TO THIS ERROR IF THE INPUT CHOICE IS NOT DISCOVER
; ------------------------------------------------------------------------------------------
errormessage1:
        mov edx, offset error_choice
        call WriteString                    ;Display error_choice
        call crlf                           ; make a new line
        jmp mainmenu                        ; jump back to main menu
; ------------------------------------------------------------------------------------------
; JMP TO THIS ERROR IF THE INPUT BINARY IS NOT 8 BIT / NOT 0 & 1
; ------------------------------------------------------------------------------------------
errormessage2:
        mov edx, offset error_binary
        call WriteString                    ;Display error_binary
        call crlf                           ; make a new line
        jmp mainmenu                        ; jump back to main menu
; ---------------------------------------------------------- ----------------------------------
; JMP TO THIS ERROR IF THE INPUT DECIMAL IS NOT LESS TAHN 256
; ---------------------------------------------------------- ----------------------------------
errormessage3:
        mov edx, offset error_decimal
        call WriteString                    ;Display error_decimal
        call crlf                           ; make anew line
        jmp mainmenu                        ; jump back to main menu

; -----------------------------------
; binary to decimal function
; -----------------------------------
Bin2Dec:
        ; calling the prompt& input the binary number
        mov decimalnum, 0
        mov edx, offset binary_prompt
        call WriteString                    ;Display binary_prompt
        mov edx, offset binarynum           ; move register edx with the address of binarynum
        mov ecx, SIZEOF binarynum
        call ReadString                     ; Ask user to input a string
        mov NumberLength, eax               ; move the length of the string into NumberLength
        cmp NumberLength, 8d                ; compare the length with 8
        jne errormessage2                   ; if the length is not 8 then an error message will be
displayed

; converter part
mov esi, 7
mov ecx, 8
; binarynum[0], binarynum[1], binarynum[2]....binarynum[7] -> 8bit  , so esi->7
; 8,7,6,5,4,3,2,1,(0)->jump to bin2dec_ans , so ecx->8
; both ecx & esi will decrease once finish one loop
Converter1:
        mov Numberlength, 8d                ; always make NumebrLength equal 8 after a loop
        cmp ecx, 0                          ; compare register ecx with 0
        je Bin2Dec_ans                      ; if ecx equals 0 then jump to the output part
        cmp binarynum[esi], '1'             ; compare the character of the string with ‘1’
        je todosum                          ; if equal 1 then jump to the todosum part
        cmp binarynum[esi], '0'             ; compare the character of the string with ‘0’
        je increment                        ; if equal 0 then jump to the increment part
        jne errormessage2                   ; if there is character other than 1 or 0 then jump to
errormessage2 to remind the user input proper binary number.
        todosum :
                sub NumberLength, ecx       ; substract numberlength, 8 with the latest ecx to get
the exact position
                mov eax, 1d                 ; move the value of eax with 1
        whilePart:
                cmp NumberLength, 0         ; if numberlength is 0 then jmp to sumPart
                je sumPart
                mov ebx, 2d                 ; move ebx with 2
                mul ebx                     ; multiply eax with ebx which is 2
                dec NumberLength            ; decrease the numberlength by 1
                jmp whilePart               ; loop again
        sumPart :
                add decimalnum, eax            ; add the eax into decimalnum
                jmp increment                  ; jmp to increment
        increment :
                dec esi                        ; decrease esi
                dec ecx                        ; decrease ecx by 1
                jmp Converter1                 ; jmp back to converter1

; ANSWER OUTPUT
Bin2Dec_ans :
        mov edx, offset b2d_ans1
        call WriteString               ;Display b2d_ans1
        mov edx, offset binarynum
        call WriteString               ;Display binarynum
        mov edx, offset b2d_ans2
        call WriteString               ;Display b2d_ans2

        mov eax, decimalnum
        call WriteDec

        mov al, 'd'
        call WriteChar                 ; display character d after the decimal number is displayed
        call crlf
        call crlf

        jmp mainmenu                   ; jump back to main menu

; ------------------------------------
; decimal to binary function
; ------------------------------------

Dec2Bin:
        mov edx, offset decimal_prompt
        call WriteString                       ;display decimal_prompt
        call ReadDec                           ; ask user to input a decimal value
        mov decimalnum, eax                    ; move the value obtained into decimalnum
        cmp eax, 255d                          ; compare eax with 255
        ja errormessage3                       ; if the value of decimalnum is above than 255 then,
jump to errormessage3

;converter part
                mov esi, 7
                mov eax, decimalnum
                mov ecx, 8
; binarynum[0], binarynum[1], binarynum[2]....binarynum[7] -> 8bit  , so esi->7
; 8,7,6,5,4,3,2,1,(0)->jump to bin2dec_ans , so ecx->8
; both ecx & esi will decrease once finish one loop
```

```
Converter2 :
                    mov edx, 0                  ; clear the edx by moving 0 into the edx
                    mov ebx, 2d
                    div ebx                     ; divide eax with 2
                    cmp edx, 1                  ; compare edx with 1
                    je makingbinary1            ; if the edx is 1 then jump to makingbianry1
                    cmp edx, 0                  ; compare edx with 0
                    je makingbinary0            ; if the edx is 0 then jump to makingbinary 0
                    makingbinary1 :
                    mov binarynum[esi], '1'     ; move character 1 into the position of the binarynum
                    jmp increment2
                    makingbinary0 :
                    mov binarynum[esi], '0'     ; move character 0 into the position of the binarynum
                    jmp increment2
                    increment2 :
                    dec esi                     ; decrease esi by 1
                    loop Converter2             ; loop back to Converter2 until ecx is 0

; ANSWER OUTPUT
Dec2Bin_ans :
                    mov edx, offset d2b_ans1
                    call WriteString                ;Display d2b_ans1
                    mov eax, decimalnum
                    call WriteDec                   ;Display decimalnum
                    mov edx, offset d2b_ans2
                    call WriteString                ;Display d2b_ans2
                    mov edx, offset binarynum
                    call WriteString                ;Display binarynum

                    mov al, 'b'
                    call WriteChar          ; display character b after the binary number is displayed
                    call crlf
                    call crlf

                    jmp mainmenu            ; jump back to main menu

exitprog :
                         mov edx, offset exit_prompt
                         call WriteString           ;Display Exit Prompt
                         call crlf
                         call WaitMsg

                         exit
                         main ENDP
                         end main
```

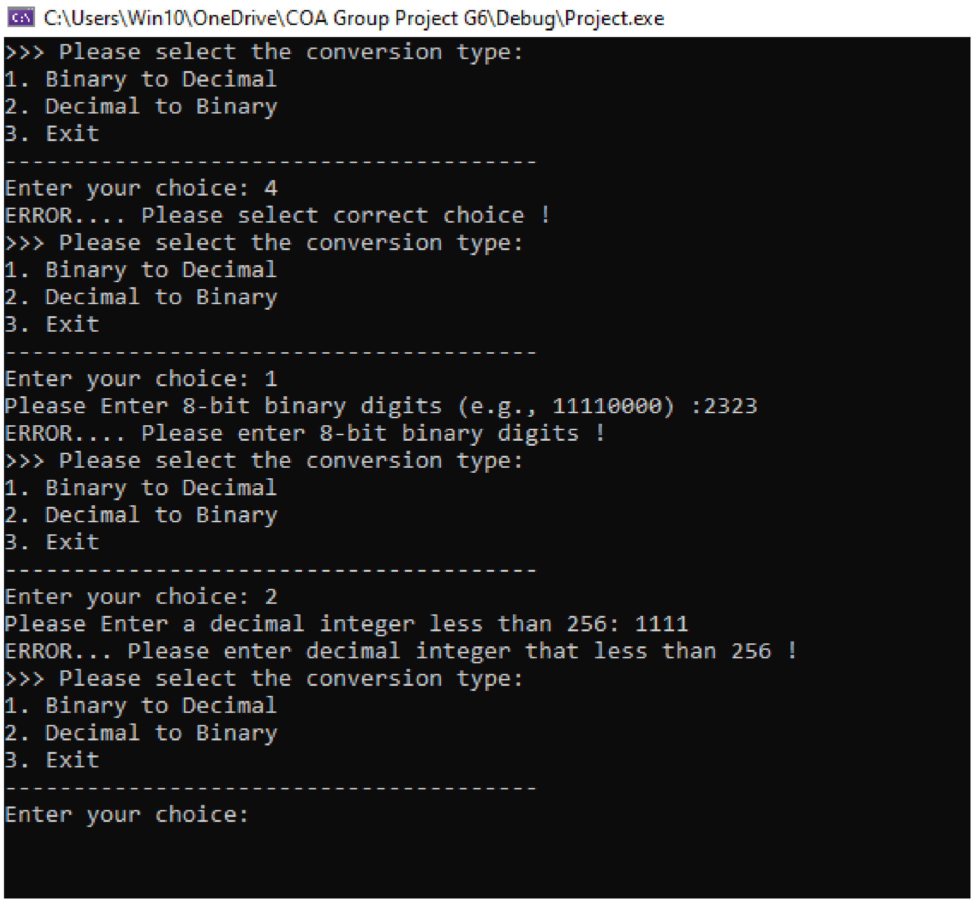
3. Example input & output



```
C:\Users\Win10\OneDrive\COA Group Project G6\Debug\Project.exe
>>> Please select the conversion type:
1. Binary to Decimal
2. Decimal to Binary
3. Exit
---------------------------------------
Enter your choice: 1
Please Enter 8-bit binary digits (e.g., 11110000) :10101010
The decimal integer of 10101010b is 170d

>>> Please select the conversion type:
1. Binary to Decimal
2. Decimal to Binary
3. Exit
---------------------------------------
Enter your choice: 2
Please Enter a decimal integer less than 256: 170
The binary of 170d is 10101010b

>>> Please select the conversion type:
1. Binary to Decimal
2. Decimal to Binary
3. Exit
---------------------------------------
Enter your choice: 3
Bye.
Press any key to continue...
```

figure 1

In *figure 1*, the assembly will accept input for option in menu. Then, if user chooses option 1, the assembly will prompt for user to enter an 8-bit binary digits and then converting it into a decimal integer. If user chooses option 2, the assembly will prompt for user to enter a decimal integer less than 256 and then convert it into a binary digit. Assembly will prompt the user if they want to try again after completing option 1 or 2. Finally, if user chooses option 3, program will end.



```
C:\Users\Win10\OneDrive\COA Group Project G6\Debug\Project.exe
>>> Please select the conversion type:
1. Binary to Decimal
2. Decimal to Binary
3. Exit
---------------------------------------
Enter your choice: 4
ERROR.... Please select correct choice !
>>> Please select the conversion type:
1. Binary to Decimal
2. Decimal to Binary
3. Exit
---------------------------------------
Enter your choice: 1
Please Enter 8-bit binary digits (e.g., 11110000) :2323
ERROR.... Please enter 8-bit binary digits !
>>> Please select the conversion type:
1. Binary to Decimal
2. Decimal to Binary
3. Exit
---------------------------------------
Enter your choice: 2
Please Enter a decimal integer less than 256: 1111
ERROR... Please enter decimal integer that less than 256 !
>>> Please select the conversion type:
1. Binary to Decimal
2. Decimal to Binary
3. Exit
---------------------------------------
Enter your choice:
```

figure 2

In *figure 2*, it is similar to figure 1 but if user enters a menu option that is not 1, 2 or 3, an error message will be displayed and jump to main menu for the user to input again. This also applies if user enters a non-8-bit binary digits or a decimal integer less than 256.

4. Discussion & Conclusion

This assembly program that we were asked to write is a program that allows the conversion of an 8-bit unsigned binary to its decimal and conversion of a positive decimal to its 8-bit binaries. Based on the question, the example output given does not display error feedback when the user enters the invalid input, but our group decided to include this feature into our code in order to improve the understanding of the user if they make a mistake. After the user enters an invalid input that is explained in *figure 2,* computer will prompt the user to reenter the input.

The concept that we use to convert the binary to decimal number is by looping the character inside the binary number which is made by string. First, the decimal number must be cleared with 0. Then, user will be asked to enter an eight-bit binary number that is stored as string. Next, the assembly will continue with the conversion part. For this part, value of register ESI is started with 7 and register ECX is started with 8. This is because inside an eight-bit string, the index of characters has 0 until 7 but the function needs to loop 8 times. Hence, ESI and ECX will decrease by 1 after finishing a loop. After that, we make a conditional jump by naming the target as Converter1. Inside, we compare register ECX with 0 so that the assembly will jump out from Converter1 to the output part once we finish counting 8 bits of binary number. Next, the assembly will determine whether the character of the string is '0' or '1'. If the character is '1', the exact position of the character must be known by subtracting the variable "NumberLength" that we have declared it with value of 8 by the current register ECX. By having the exact position, we able to know the number of times to loop for the multiplication of 2. For example, if the index is 4, the result should be 2x2x2x2 = 16 and the result is stored in the register EAX. After the loop, the assembly will jump to the sum part where register EAX will be added into the decimal number. Lastly, the register ECX and register ESI are decreased by 1 before jumping back to the target, Converter1. However, if the character is '0', there is nothing to do with number but just decrease the register ESI and register ECX by 1. This is because when we power the number,0 with any number, the result is still 0. Certainly, if user input the string with character other than 0 and 1, an error message will be displayed. After the 8 times of jumping, the assembly will jump to the output part where the result of decimal number will be known.

Similarly, we use back the concept of looping while converting decimal to binary but here we are checking for the decimal, not a string. First, the instruction to input a decimal number is displayed and user is asked to enter a decimal number. Next, the value 255 is compared with register EAX where the input decimal value is stored to ensure that it is below than 256. If the value is above 255, an error message will be displayed. Next, the assembly will continue with the conversion part. For this part, register ESI is started with 7 and register ECX is started with 8 which has the same reason as the previous converter part. Moving forward, a conditional jump is made to ensure that the register ECX is counted 8 times before jumping to the output part by naming the target as Converter2. Inside here, register EDX is cleared by moving value 0 into it so that there is space to fit the reminder value during the division. Then, the register EAX is divided by 2. To determine whether the binary number from the position is 0 or 1, comparison is used. If the reminder value is 1, move the character '1' into the position of the binary number or else move character '0' into the position of the binary number. After that, register ESI is decreased only since we are using loop for this function. The loop will stop once the register ECX is zero and jump to the output part where the result of binary number will be known.

In conclusion, this computer organization project had tasked us to write a simple code to convert an 8-bit unsigned binary to its decimal and converting a positive decimal to its 8-bit binaries. To tackle this question, we had broken the code into partitions that we could solve one by one as a group.

Converting binary to decimal and vice versa is an important concept to understand as the binary numbering system forms the basis for all computer and digital systems. Clearly, we have all gained numerous amounts of knowledge in letting us apply what we have learned in Computer Organization Architecture, whilst gaining new knowledge when studying to complete this project successfully.

5. References

Chau, C. F., & Fung, Y. F. (2011). A tool for self-learning assembly language programming and computer architecture: Design and evaluation. *Computer Applications in Engineering Education*, *19*(2), 286-293.

Detmer, R. C. (2001). *Introduction to 80x86 assembly language and computer architecture*. Jones & Bartlett Learning.

Null, L., & Lobur, J. (2014). *Essentials of Computer Organization and Architecture*. Jones & Bartlett Publishers.