|  |
| --- |
| **PRACTICE EXERCISES OF THE MICROPROCESSORS & MICROCONTROLLERS**  **Instructor: The Tung Than**  **Student's name: Nguyen Quoc Truong An**  **Student code: 21521810** |

**PRACTICE REPORT NO 5**

**LAB5: ADDITION OF TWO 32-BIT NUMBERS ON THE 8086 PROCESSOR**

1. **Student preparation**

- Students install the [emu8086 software](https://drive.google.com/file/d/1znmi3FVGWaBUFf-Gba-khSBhw7qz2FQL/view).

1. **Practice content / Exercise**

|  |
| --- |
| **1. Refer to the example:** |

o Students open File >> example >> add/subtract

o Run test, based on the instruction set describing its operation.

(Help >> Documentations and tutorials or Press F1)

o Know how to reuse their screen print code

**2. Write a program that adds 2 32-bit numbers.**

**3. Write a program to subtract 2 32-bit numbers.**

* **Chương trình đầy đủ hỗ trợ tính toán cộng/trừ 2 số 32-bit:**
* Input là 2 số đầu vào ở dạng **Hexadecimal** (ví dụ đầu vào có thể nhập các cặp số như sau: {ABC123, 4F}; {0EF34, 000E4}; {DEFCBA23, AB} …)
* Kết quả hiển thị trên màn hình Console gồm 4 dòng lần lượt: **Tổng, Cờ báo tràn tổng, Hiệu, Cờ báo tràn hiệu (KẾT QUẢ Ở DẠNG BÙ 2).**

.MODEL SMALL

.STACK 100h

.DATA

;=======================PHAN KHAI BAO CAC CHUOI===============================

MSG1 DB 10,13, "NHAP SO THU NHAT: $"

MSG2 DB 10,13, "NHAP SO THU HAI: $"

MSG3 DB 10,13, "TONG HAI SO LA: $"

MSG4 DB 10,13, "HIEU HAI SO LA: $"

MSG5 DB 10,13, "OVERLOAD FLAG: $"

MSG6 DB "h$" ; IN RA KI HIEU 'h' SAU CUNG KET QUA DANG HEX

;========================PHAN KHAI BAO CAC BIEN===============================

NUM1 DD ?, '$' ; LUU SO THU NHAT

NUM2 DD ?, '$' ; LUU SO THU HAI

RESULT DD ?, '$' ; LUU KET QUA SUM/SUBTRACT

OVFL\_FLAG DB ?, '$' ; LUU BIT OVERFLOW

NUM\_HIGH DW ?, '$' ; LUU 16 BIT CAO

NUM\_LOW DW ?, '$' ; LUU 16 BIT THAP

A DD ?, '$' ; CAC BIEN TAM 32 BIT

B DD ?, '$'

.CODE

;============================CHUONG TRINH CHINH===============================

MAIN PROC

;=========================PHAN KHOI TAO BAN DAU===============================

;LAY DIA CHI CUA VUNG NHO DATA VAO THANH GHI DOAN DS

MOV AX, @DATA

MOV DS, AX

;=======================PHAN NHAP 2 SO INPUT O DAND HEX=======================

MOV AH, 9h ; THONG BAO NHAP SO THU NHAT

LEA DX, MSG1

INT 21h

CALL READ\_NUM ; NHAP SO THU NHAT

MOV DX, NUM\_HIGH

MOV AX, NUM\_LOW

MOV [NUM1], AX ; LUU SO THU NHAT VAO BIEN NUM1

MOV [NUM1+2], DX ; 16 BIT THAP VAO [NUM1], 16 BIT CAO VAO [NUM1+2]

MOV AH, 9h ; THONG BAO NHAP SO THU HAI

LEA DX, MSG2

INT 21h

CALL READ\_NUM ; NHAP SO THU HAI

MOV DX, NUM\_HIGH

MOV AX, NUM\_LOW

MOV [NUM2], AX ; LUU SO THU NHAT VAO BIEN NUM2

MOV [NUM2+2], DX ; 16 BIT THAP VAO [NUM2], 16 BIT CAO VAO [NUM2+2]

;=====================PHAN THUC HIEN TINH TONG HAI SO=========================

CALL SUM ; GOI HAM THUC HIEN TINH TONG

;==========================PHAN IN KET QUA PHEP CONG==========================

MOV AH, 9h ;IN RA THONG BAO KET QUA ADD

LEA DX, MSG3

INT 21h

CALL PRINT\_RESULT ; GOI HAM IN RA KET QUA SUM

MOV AH, 9h ; IN RA THONG BAO OVERFLOW

LEA DX, MSG5

INT 21h

MOV DL, [OVFL\_FLAG] ; IN RA GIA TRI CO BAO TRAN PHEP CONG

ADD DL, 30h

MOV AH, 2h

INT 21h

;=====================PHAN THUC HIEN TINH HIEU HAI SO=========================

CALL SUBTRACT ; GOI HAM THUC HIEN TINH HIEU

;==========================PHAN IN KET QUA PHEP TRU===========================

MOV AH, 9h ; IN RA THONG BAO KET QUA SUB

LEA DX, MSG4

INT 21h

CALL PRINT\_RESULT ; GOI HAM IN RA KET QUA SUB

MOV AH, 9h ; IN RA THONG BAO OVERFLOW

LEA DX, MSG5

INT 21h

MOV DL, [OVFL\_FLAG] ; IN RA GIA TRI CO BAO TRAN PHEP TRU

ADD DL, 30h

MOV AH, 2h

INT 21h

;==============================THOAT CHUONG TRINH=============================

MOV AH, 4Ch ; NGAT THOAT KHOI CHUONG TRINH

INT 21h

MAIN ENDP

;=============================================================================

;=============================CAC CHUONG TRINH CON============================

;====================HAM DOC INPUT DUOC NHAP TU BAN PHIM======================

READ\_NUM PROC

MOV A, 0

MOV B, 0

XOR DX, DX ; RESET DX = 0

XOR AX, AX ; RESET AX = 0

MOV BX, 16 ; BX = 16 (HE HEX NEN CO SO NHAN TICH LUY KHI CHUYEN DOI SANG DEC LA 16)

READ\_KEY\_PRESSED: ; DOC KY TU NHAP TU BAN PHIM

MOV AH, 1h ; READ KY TU TU BAN PHIM, KY TU DUOC LUU TRONG AL

INT 21h

CMP AL, 0Dh ; NEU AL = 0Dh (MA ASCII CUA PHIM ENTER) => BAM PHIM ENTER

JE READ\_DONE ; NEU BAM PHIM ENTER THI HOAN THANH VIEC NHAP SO

CMP AL, 41h ; SO SANH VOI KY TU A (MA ASCII CUA 'A' = 41h) DE PHAN LOAI CHU HAY SO

JL NUMBER ; CHUYEN VE GIA TRI TUONG UNG VOI SO

JMP CHARACTER ; CHUYEN VE GIA TRI TUONG UNG VOI CHU

NUMBER: ; SO DUOC BAM - MA ASCII CUA '0' THI SE RA GIA TRI TUONG UNG

SUB AL, 30h ; VD: '9' - '0' = 9

JMP CONVERT\_NUM ; NHAY DEN HAM CHUYEN HEX SANG DECIMAL

CHARACTER: ; CHU DUOC BAM - 37h THI SE RA GIA TRI TUONG UNG

SUB AL, 37h ; VD: 'A' - 37h = 65 - 55 = 10

CONVERT\_NUM: ; THUC HIEN NHAN CAC GIA TRI VUA NHAP VOI BAC TUONG UNG

XOR AH, AH ; VD: TA NHAP 3 GIA TRI: A, 9, 3 => SO HEX = A\*16^2 + 9\*16^1 + 3\*16^0

CMP DX, 0

JNE OVERFLOW ; KIEM TRA TRAN THANH GHI NEU TA NHAP QUA 16 BIT, NEU CO NHAY TOI XU LI TRAN

MOV B, AX ; CHUYEN DOI NHU VD NAY: A, 9, 3 => SO HEX = A\*16^2 + 9\*16^1 + 3\*16^0

MOV AX, A

MUL BX

ADD AX, B

MOV A, AX

MOV NUM\_HIGH, DX ; LUU 16 BIT CAO

MOV NUM\_LOW, AX ; LUU 16 BIT THAP

JMP READ\_KEY\_PRESSED

OVERFLOW: ; XU LY TRAN

MOV A, AX

MOV CX, 4 ; SO LAN VONG LAP LOOP

MOV DX, NUM\_HIGH

MOV AX, NUM\_LOW

SHIFT:

SHL AX, 1 ; DICH TRAI 1 BIT, HE HEX NEN DICH 4 LAN

RCL DX, 1 ; OFF-BIT CUA PHEP DICH TRAI DUOC DUA VAO DX

LOOP SHIFT

ADD AX, A

MOV NUM\_HIGH, DX ; LUU 16 BIT CAO

MOV NUM\_LOW, AX ; LUU 16 BIT THAP

JMP READ\_KEY\_PRESSED

READ\_DONE: ; HOAN THANH VIEC DOC 1 SO TU BAN PHIM O DANG HEX

RET

READ\_NUM ENDP

;============================================================================

;==========================HAM THUC HIEN TINH TONG===========================

SUM PROC

CLC ; XOA BIT CF TU CAC LENH TRUOC

MOV AX, [NUM1] ; AX CHUA 16 BIT THAP CUA NUM1

MOV BX, [NUM1+2] ; BX CHUA 16 BIT CAO CUA NUM1

MOV CX, [NUM2] ; CX CHUA 16 BIT THAP CUA NUM2

MOV DX, [NUM2+2] ; DX CHUA 16 BIT THAP CUA NUM2

ADD AX, CX ; CONG 16 BIT THAP CUA 2 SO VOI NHAU, AX = AX + CX

MOV [RESULT], AX ; LUU 16 BIT THAP CUA KET QUA

ADC BX, DX ; CONG 16 BIT CAO CUNG VOI BIT CARRY FLAG, BX = BX + DX + CF

MOV [RESULT+2], BX ; LUU 16 BIT CAO CUA KET QUA

JO SET\_FLAG\_SUM ; XU LY TRAN SO PHEP CONG

MOV [OVFL\_FLAG], 0 ; OVFL\_FLAG = 0 NEU KHONG TRAN

JMP SUM\_DONE

SET\_FLAG\_SUM:

MOV [OVFL\_FLAG], 1 ; OVFL\_FLAG = 1 NEU TRAN

SUM\_DONE: ; HOAN THANH VIEC CONG

RET

SUM ENDP

;==========================================================================

;========================HAM THUC HIEN TINH HIEU===========================

SUBTRACT PROC

CLC ; XOA BIT CF TU CAC LENH TRUOC

MOV AX, [NUM1] ; AX CHUA 16 BIT THAP CUA NUM1

MOV BX, [NUM1+2] ; BX CHUA 16 BIT CAO CUA NUM1

MOV CX, [NUM2] ; CX CHUA 16 BIT THAP CUA NUM2

MOV DX, [NUM2+2] ; DX CHUA 16 BIT THAP CUA NUM2

SUB AX, CX ; TRU 16 BIT THAP CUA 2 SO VOI NHAU, AX = AX - CX

MOV [RESULT], AX ; LUU 16 BIT THAP CUA KET QUA

SBB BX, DX ; TRU 16 BIT CAO CUNG VOI BIT CARRY FLAG, BX = BX - DX - CF

MOV [RESULT+2], BX ; LUU 16 BIT CAO CUA KET QUA

JO SET\_FLAG\_SUB ; XU LY TRAN SO PHEP TRU

MOV [OVFL\_FLAG], 0 ; OVFL\_FLAG = 0 NEU KHONG TRAN

JMP SUB\_DONE

SET\_FLAG\_SUB:

MOV [OVFL\_FLAG], 1 ; OVFL\_FLAG = 1 NEU TRAN

SUB\_DONE: ; HOAN THANH VIEC TRU

RET

SUBTRACT ENDP

;==========================================================================

;========================HAM THUC HIEN IN KET QUA==========================

PRINT\_RESULT PROC

; THUC HIEN IN 16 BIT CAO CUA KET QUA

MOV BX, [RESULT+2] ; BX CHUA 16 BIT CAO CUA KET QUA

MOV CX, 4

PRINT\_HIGH: ; CHIA 16 BIT THANH 4 PHAN, MOI PHAN UNG VOI 1 SO TRONG HE HEX

MOV AX, 4 ; 16 BIT NEN TA CAN 4X4 VONG LAP ( = CX \* AX )

XOR DX, DX ; RESET DX = 0

SHIFT\_HIGH: ; DICH BIT

SHL BX, 1 ; DICH TRAI 1 BIT

RCL DX, 1 ; OFF-BIT CUA LENH DICH DUOC DUA VAO DX VOI TRONG SO TUONG UNG

DEC AX ; GIAM AX 1 DON VI (AX = AX - 1)

CMP AX, 0 ; KIEM TRA XEM CO LAP DU 4 LAN TRONG VONG LAP BE

JNE SHIFT\_HIGH ; VONG LAP BE AX = 4

CMP DX, 0Ah ; SO SANH GIA TRI VOI 0Ah UNG VOI 'A'

JGE PRINT\_CHAR\_HIGH

PRINT\_NUM\_HIGH: ; IN SO: '0' -> '9'

ADD DX, 30h

MOV AH, 2h

INT 21h

JMP PRINT\_ONE\_HIGH\_DONE

PRINT\_CHAR\_HIGH: ; IN CHU: 'A', 'B', 'C', 'D', 'E', 'F'

ADD DX, 37h

MOV AH, 2h

INT 21h

PRINT\_ONE\_HIGH\_DONE:

LOOP PRINT\_HIGH ; VONG LAP LON CX = 4

; THUC HIEN IN 16 BIT THAP CUA KET QUA

MOV BX, [RESULT] ; THUC HIEN TUONG TU NHU 16 BIT CAO ....

MOV CX, 4

PRINT\_LOW:

MOV AX, 4

XOR DX, DX

SHIFT\_LOW:

SHL BX, 1

RCL DX, 1

DEC AX

CMP AX, 0

JNE SHIFT\_LOW

CMP DX, 0Ah

JGE PRINT\_CHAR\_LOW

PRINT\_NUM\_LOW:

ADD DX, 30h

MOV AH, 2h

INT 21h

JMP PRINT\_ONE\_LOW\_DONE

PRINT\_CHAR\_LOW:

ADD DX, 37h

MOV AH, 2h

INT 21h

PRINT\_ONE\_LOW\_DONE:

LOOP PRINT\_LOW

MOV AH, 9h ; IN RA KI HIEU h (HEX)

LEA DX, MSG6

INT 21h

RET ; HOAN THANH VIEC IN SO 32 BIT DANG HEX (VD KET QUA IN: 1234ABCDh)

PRINT\_RESULT ENDP

;==========================================================================

END

1. **Report**
2. **Describe how sample code works.**

**Sample code:**

**A screenshot of a computer program

Description automatically generated with medium confidence**

**-Các lệnh được sử dụng trong ví dụ trên:**

**+MOV**: copy dữ liệu của toán operand2 lưu vào operand1

**+ADD**: operand1 = operand1 + operand2

**+SUB**: operand1 = operand1 - operand2

**+TEST:** thực hiện AND logic các bit giữa 2 toán hạng và set giá trị tương ứng vào các cờ ZF, SF, PF

**+JZ:** nhảy nếu ZF = 1

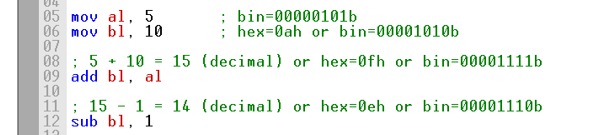
**+INT:** được sử dụng để gọi một ngắt mềm (của BIOS hoặc MSDOS) trong chương trình hợp ngữ. Khi một ngắt mềm được gọi thì hệ thống sẽ thực hiện chương trình con phục vụ ngắt tương ứng với nó.

**+SHL:** Dịch trái operand1, số bit dịch lưu trong operand2.

**+LOOP:** giảm CX 1 đơn vị, nhảy tới Label nếu CX khác 0

**+RET:** Trở về lại vị trí vừa thực hiện gọi thủ tục

**-Hoạt động của code:**



+Gán giá trị al = 5

+Gán giá trị bl = 10

+Cộng bl = bl + al = 5 + 10 = 15

+Trừ bl = bl – 1 = 15 – 1 = 14

A screenshot of a computer code

Description automatically generated with low confidence

+Gán cx = 8 (Số vòng lặp sẽ là 8 lần)

+Gán dl = ‘0’

+Kiểm tra bit đầu tiên của bl

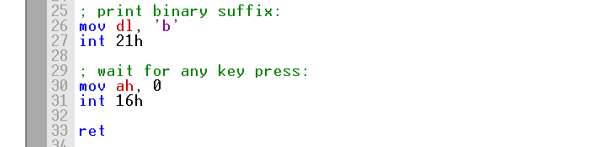
+Nếu ZF = 1 tức bit đầu tiên của bl = 0, nhảy tới nhãn zero

+Nếu ZF = 0, thực hiện gán dl = ‘1’

+Nhã zero: thực hiện interupt 21h (ah = 2 => ghi ký tự chứa trong dl ra màn hình)

+Dịch trái bl 1 bit

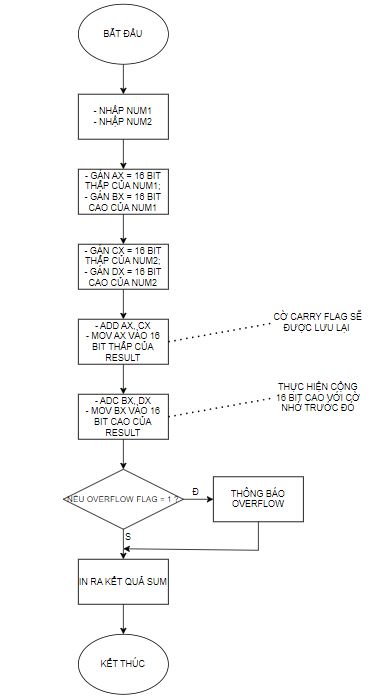
+Lặp lại các lệnh phía sau nhãn print nếu cx khác 0 (hệ thống tự giảm cx đi 1 đơn vị)



+In ký tự ‘b’ ra màn hình

+Chờ đợi user nhấn phím bất kì để thoát chương trình.

1. **Flowchart of the program algorithm to add two 32-bit numbers.**

****

1. **Explain how the algorithm works, accompanied by a video (send a Google Drive link) to demonstrate the circuit operation in case the instructor cannot run the design file**.

**\* Google Drive link:** [**https://drive.google.com/drive/folders/1p\_Y06qennIHkIlv12zh7QKBxUR5NCTJy?usp=sharing**](https://drive.google.com/drive/folders/1p_Y06qennIHkIlv12zh7QKBxUR5NCTJy?usp=sharing)

**\*\*\*Giải thích chi tiết code:**

**A picture containing text, screenshot, font

Description automatically generated** **A green text on a white background

Description automatically generated with low confidence**

**-Phần khai báo gồm:**

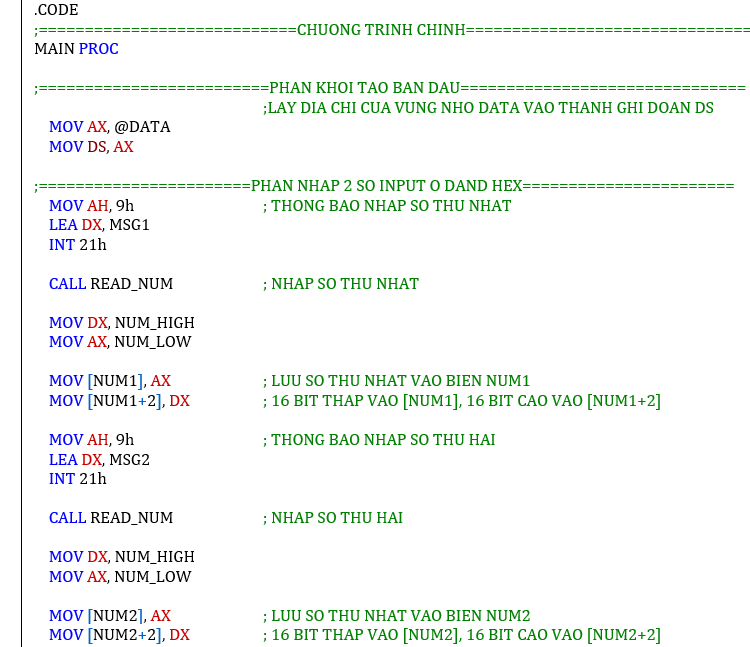
+ Các chuỗi thông báo nhập xuất, chuỗi hiển thị trên màn hình console

+ Các biến NUM1, NUM2 để chứa 2 số 32-bit

+ Biến OVFL\_FLOW chứa giá trị của cờ OF để báo tràn số

+ Các biến tạm NUM\_HIGH, NUM\_LOW, A, B phục vụ lưu trữ trong quá trình tính toán

**-Phần chương trình chính:**

****

+ Thông báo nhập số thứ nhất NUM1 dưới dạng **HEX**

+ Gọi hàm thực hiện nhập số và đọc chuyển đổi giá trị input được nhập vào

+ Lưu 16 bit thấp của số được nhập vào NUM\_LOW, 16 bit cao vào NUM\_HIGH

+ Gán lần lượt 16 bit thấp và 16 bit của của số vào 16 bit thấp và 16 bit cao của NUM1

+ Việc nhập số thứ hai được thực hiện tương tự như trên

**-Thực hiện tính cộng:**

**A picture containing text, screenshot, font

Description automatically generated**

+ Sau khi thực hiện xong việc nhập giá trị 2 số, gọi hàm SUM để tính tổng

+ Sau đó thực hiện in ra thông báo kết quả tính tổng

+ Gọi hàm PRINT\_RESULT để in ra kết quả SUM dưới dạng **HEX**

+ In ra thông báo OVERFLOW FLAG và in ra giá trị của cờ CF được lưu trong biến OVFL\_FLAG

**-Thực hiện tính trừ:**

A screenshot of a computer

Description automatically generated with low confidence

**+** Ta thực hiện tính trừ 2 số vừa nhập ở trên

**+** Gọi hàm SUBTRACT để thực hiện tính trừ

+ Sau đó thực hiện in ra thông báo kết quả tính trừ

+ Gọi hàm PRINT\_RESULT để in ra kết quả SUBTRACT dưới dạng **HEX**

+ In ra thông báo OVERFLOW FLAG và in ra giá trị của cờ CF được lưu trong biến OVFL\_FLAG

+Thực hiện gọi ngắt để thoát khỏi chương trình.

**-Các chương trình con:**

**-Chương trình nhập số 32-bit ở dạng HEX:**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with low confidence**

**+** Ta thực hiện gán A = 0, B = 0, DX = 0, AX = 0, BX = 16 (Hệ HEX ứng với cơ số 16)

**+** Thực hiện gọi ngắt nhập ký tự từ bàn phím

**+** Kiểm tra ký tự nhập có mã Ascii = 0Dh (13 tron hệ thập phân), tức là phím ENTER thì kết thúc việc nhập

**+** Kiểm tra nếu ký tự nhập có mã Ascii bé hơn 41h (mã Ascii của ký tự ‘A’), tức ký tự ta nhập là chữ số thì tiến hành đổi chữ số thành giá trị số tương ứng bằng các lấy mã Ascii của nó trừ đi 30h (mã Ascii của ký tự ‘0’). Ngược lại ký tự vừa được nhập là chữ (‘A’, ‘B’, ‘C’, ‘D’, ‘E’, ‘F’) thì tiến hành đổi ký tự chữ thành giá trị tương ứng bằng các lấy mã Ascii của nó trừ đi cho 37h sẽ ra giá trị tương ứng (‘A’-10, ‘B’-11, ‘C’-12, ‘D’-13, ‘E’-14, ‘F’-15).

**+** Thực hiện kiểm tra tràn từ thanh ghi 16-bit AX sang thanh ghi 16-bit DX trong quá trình nhập (vì số nhập vào có giá trị lên đến 32-bit), nếu DX khác 0 đồng nghĩa với việc đã nhập quá 16-bit, tiến hành nhảy tới nhãn xử lý tràn bit từ thanh ghi AX sang DX. Ngược lại tiếp tục nhập và xử lý giá trị nhập.

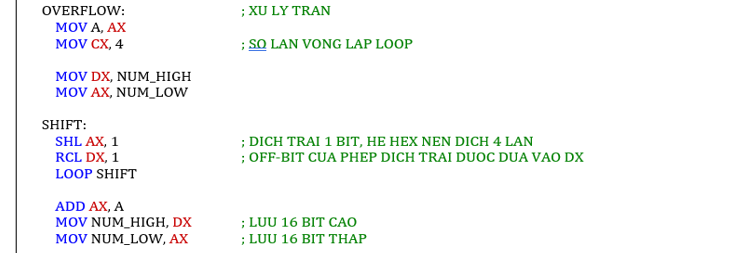
**+** Ta xử lý giá trị nhập vào bằng các thực hiện 4 lệnh sau để chuyển đổi các ký tự nhập vào ứng với trọng số tương ứng:

A white background with green text

Description automatically generated with low confidence

+Sau đó tiếp tục quay lại thực hiện nhập ký tự tiếp theo của số HEX cho đến khi nhận được phím ENTER

**+Ta xử lý khi nhập số quá 16-bit như sau:**

****

**+** Ta tiến hành lần lượt dịch trái AX 4 bit (dịch lần lượt từng bit)

**+** Trong lúc dịch từng bit, ta sử dụng lệnh **RCL** có chức năng **xoay** thanh ghi DX qua cờ **CF**, mà cờ **CF** được gán là bit vừa dịch trái từ thanh ghi AX ở lệnh **SHL AX, 1** như trên. **Như vậy ta đã dịch chuyển 4 bit cao của AX sang 4 bit thấp của DX.**

**+** Sau khi tiến hành dịch chuyển xong, ta cộng giá trị tương ứng của ký tự vừa được nhập vào AX (AX lúc này chứa 16 bit thấp của số cần nhập << 4)

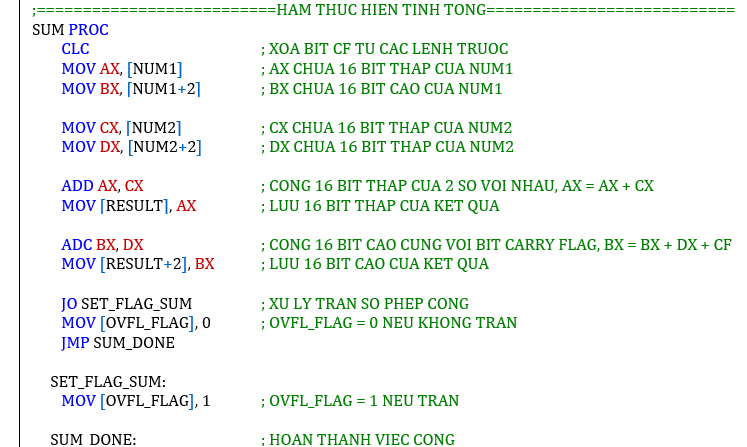
**+** Sau khi xử lý xong, tiến hành lưu giá trị nhập hiện tại vào biến NUM\_HIGH, NUM\_LOW ứng với 16 bit cao và 16 bit thấp của số đã được nhập hiện tại.

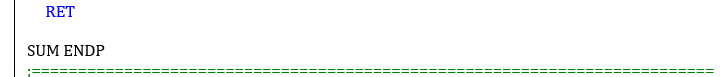
A green text on a white background

Description automatically generated with low confidence

+ Khi nhảy tới nhãn READ\_DONE thì hoàn thành xong việc nhập số 32-bit ở dạng HEX, kết quả lưu 16-bit cao ở NUM\_HIGH, 16-bit thấp ở NUM\_LOW.

**-Hàm thực hiện tính SUM:**

****

****

+ Ta thực hiện gán AX = 16-bit thấp của NUM1, BX = 16-bit cao của NUM1, CX = 16-bit thấp của NUM2, DX = 16-bit cao của NUM2

+ Thực hiện cộng 16-bit thấp

+ Thực hiện cộng 16-bit cao cờ nhớ CF

+ Gán giá trị OVFL\_FLAG = 1 nếu cờ OF = 1, OVFL\_FLAG = 0 nếu cờ OF = 0. Hoàn thanh việc tính cộng.

**-Hàm thực hiện tính SUBTRACT:**

****

+ Ta thực hiện gán AX = 16-bit thấp của NUM1, BX = 16-bit cao của NUM1, CX = 16-bit thấp của NUM2, DX = 16-bit cao của NUM2

+ Thực hiện trừ 16-bit thấp

+ Thực hiện trừ 16-bit cao cờ nhớ CF

+ Gán giá trị OVFL\_FLAG = 1 nếu cờ OF = 1, OVFL\_FLAG = 0 nếu cờ OF = 0. Hoàn thanh việc tính trừ.

**-Hàm thực hiện in kết quả tính toán 32-bit ở dạng HEX:**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

**A picture containing text, screenshot, font, software

Description automatically generated**

**+** Để thực hiện in ra số 32-bit ở dạng **HEX** ta cần tác 32-bit ra thành các phần 4-bit rồi chuyển 4-bit này thành giá trị tương ứng trong hệ Hexadecimal.

**+** Vì có 16-bit cao và 16-bit thấp nên ta lần lượt chuyển đổi và in 16-bit cao xong sau đó mới tới 16-bit thấp.

**-Ta tiến hành chuyển đổi và in ra 16-bit cao như sau:**

+ Gán BX = 16-bit cao của RESULT

+ Gán DX = 0

+ Gán CX = 4, AX = 4 ( 4 vòng lặp lớn ứng với CX và 4 ký tự trong hệ HEX, 4 vòng lặp nhỏ ứng với giá trị AX và mỗi ký tự trong hệ HEX có 4 bit, 4\*4 = 16 bit)

+ Ta tiến hành **dịch trái BX 1 bit**, xoay DX qua CX qua cờ **CF** bằng lệnh **RCL DX, 1.**

+ Sau khi lặp bước trên đủ 4 lần, DX chứa 4-bit của ký tự HEX tương ứng

+ Kiểm tra nếu giá trị DX, nếu DX >= 0Ah (Tức giá trị tương ứng của ‘A’ trong HEX) thì **cộng DX = DX + 37h** sau đó tiến hành in ký tự chữ. Ngược lại giá trị DX ứng với ký tự số thì **cộng DX = DX + 30h** sau đó tiến hành in ký tự số.

**-Việc chuyển đổi và in ra 16-bit thấp tương tự như in 16-bit cao được trình bày ở trên.**

1. **References**

[1] [Subtract two 32-bit numbers](https://projectsgeek.com/2011/05/program-to-subtract-two-32-bit-numbers.html)

[2] [Add two 32-bit numbers](https://ankurm.com/8086-assembly-program-to-add-two-32-bit-numbers/)

[3] [Họ vi xử lý Intel 80x86](https://vhoc.net/wp-content/uploads/2015/05/Vhoc.Netnghien-cuu-8086.pdf)