Name: Kulvir Singh
Register Number: 19BCE2074

# Microprocessor Lab Digital Assignment 1

## 1) 16 Bit Addition

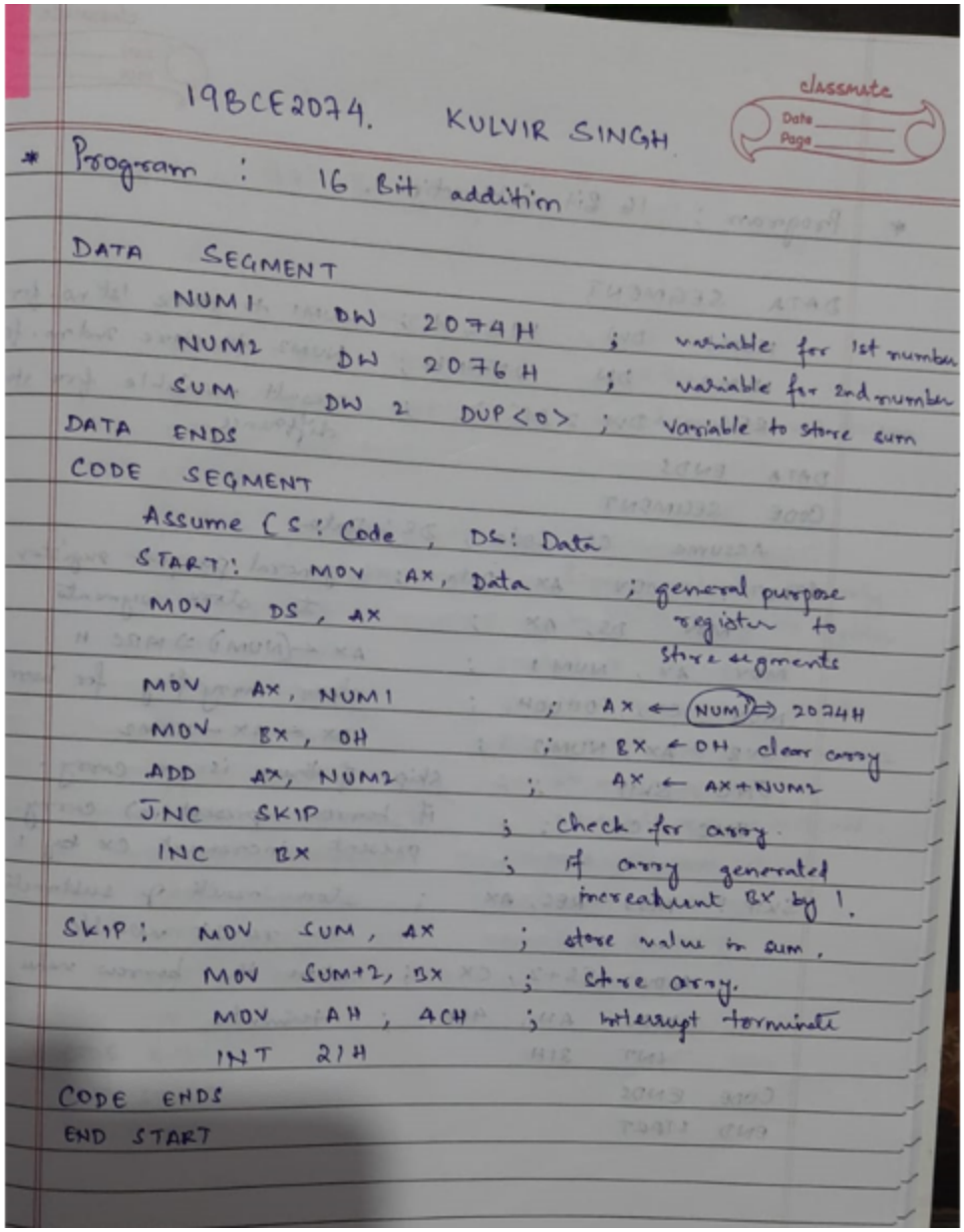
### Aim :

Write a program in 8086 Assembly Language to add 2 16 bit numbers and store the result in a variable

### Requirements :

8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

### Handwritten Program :

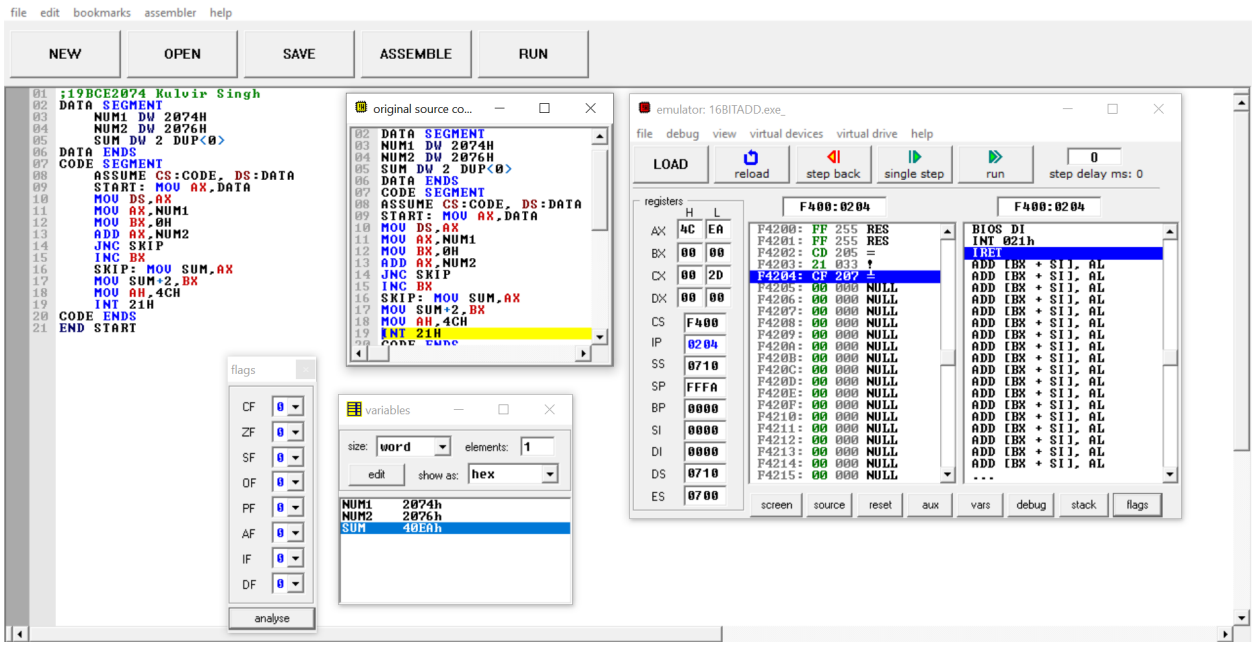


19BCE2074. KULVIR SINGH

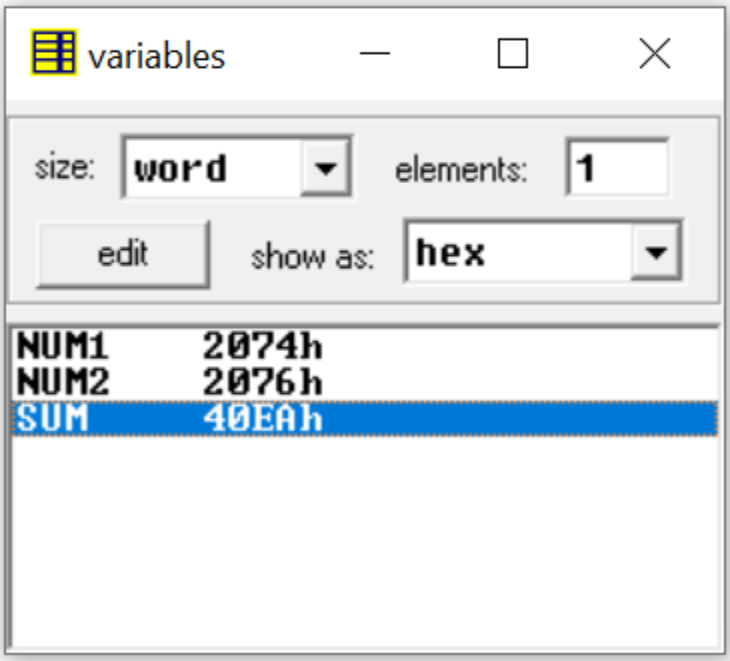
* Program : 16 Bit addition

```
DATA    SEGMENT
        NUM1    DW   2074H      ; variable for 1st number
        NUM2    DW   2076H      ; variable for 2nd number
        SUM     DW   2 DUP<0>   ; variable to store sum
DATA    ENDS
CODE    SEGMENT
        Assume CS: Code , DS: Data
        START:  MOV  AX, Data   ; general purpose
        MOV  DS , AX              register to
                                  store segments
        MOV  AX, NUM1           ; AX ← (NUM1) ⇒ 2074H
        MOV  BX , 0H            ; BX ← 0H clear carry
        ADD  AX, NUM2           ; AX ← AX+NUM2
        JNC  SKIP               ; check for carry
        INC  BX                 ; If carry generated
                                  increament BX by 1.
SKIP:   MOV  SUM , AX           ; store value in sum,
        MOV  SUM+2, BX          ; store carry.
        MOV  AH , 4CH           ; interrupt terminate
        INT  21H
CODE    ENDS
END START
```

**Screenshots :**


file edit bookmarks assembler help
NEW
OPEN
SAVE
ASSEMBLE
RUN
;19BCE2074 Kulvir Singh
DATA SEGMENT
NUM1 DW 2074H
NUM2 DW 2076H
SUM DW 2 DUP<0>
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START: MOV AX,DATA
MOV DS,AX
MOV AX,NUM1
MOV BX,0H
ADD AX,NUM2
JNC SKIP
INC BX
SKIP: MOV SUM,AX
MOV SUM+2,BX
MOV AH,4CH
INT 21H
CODE ENDS
END START
original source co...
emulator: 16BITADD.exe_
file debug view virtual devices virtual drive help
LOAD
reload
step back
single step
run
step delay ms: 0
registers
F400:0204
F400:0204
BIOS DI
INT 021h
IRET
flags
CF
ZF
SF
OF
PF
AF
IF
DF
analyse
variables
size: word
elements: 1
edit
show as: hex
NUM1 2074h
NUM2 2076h
SUM 40EAh
screen
source
reset
aux
vars
debug
stack
flags


flags
CF
ZF
SF
OF
PF
AF
IF
DF
analyse


variables
size: word
elements: 1
edit
show as: hex
NUM1 2074h
NUM2 2076h
SUM 40EAh

### Inference :

The program can successfully add two 16 bit numbers and stores the correct result in the variable.

## 2) 16 Bit Subtraction

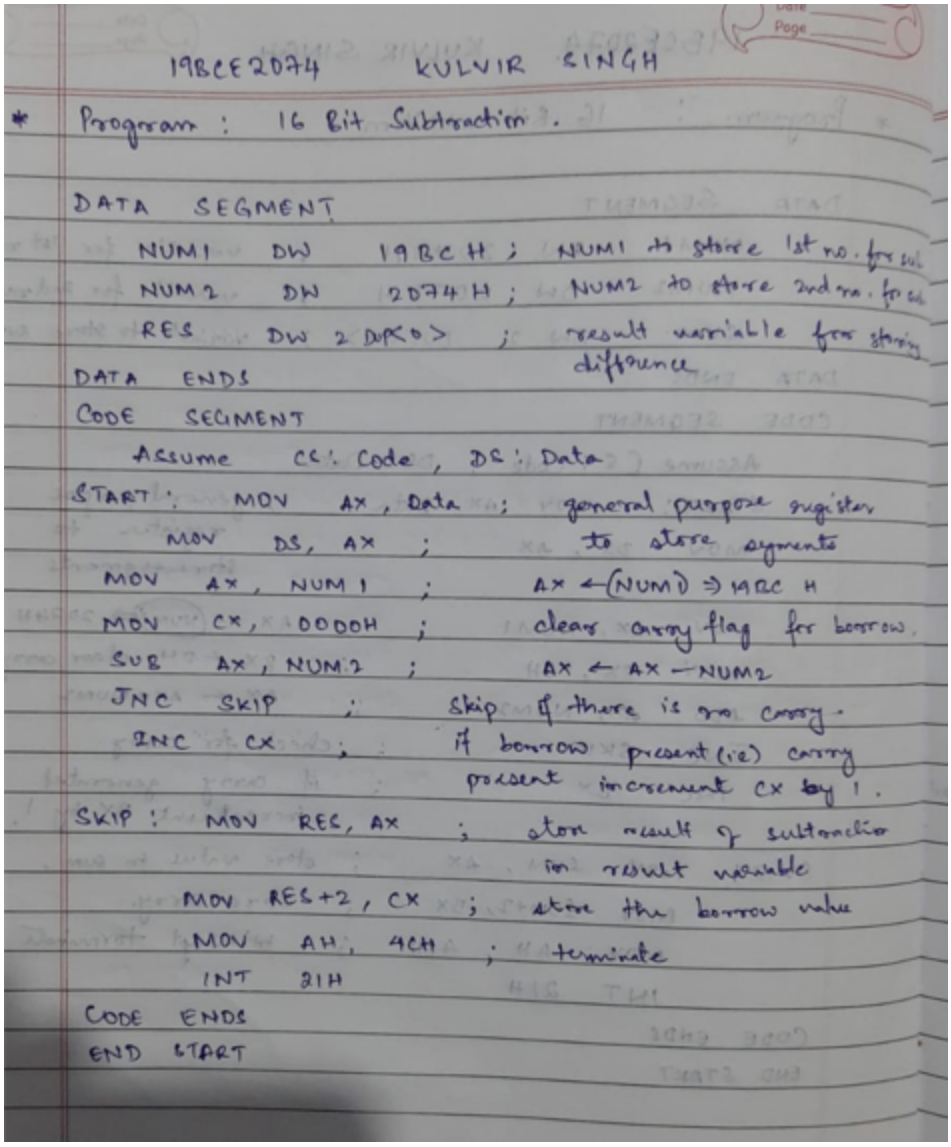
### Aim :

Write a program in 8086 Assembly Language to subtract 2 16 bit numbers and store the result in a variable

### Requirements :

8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

### Handwritten Program :

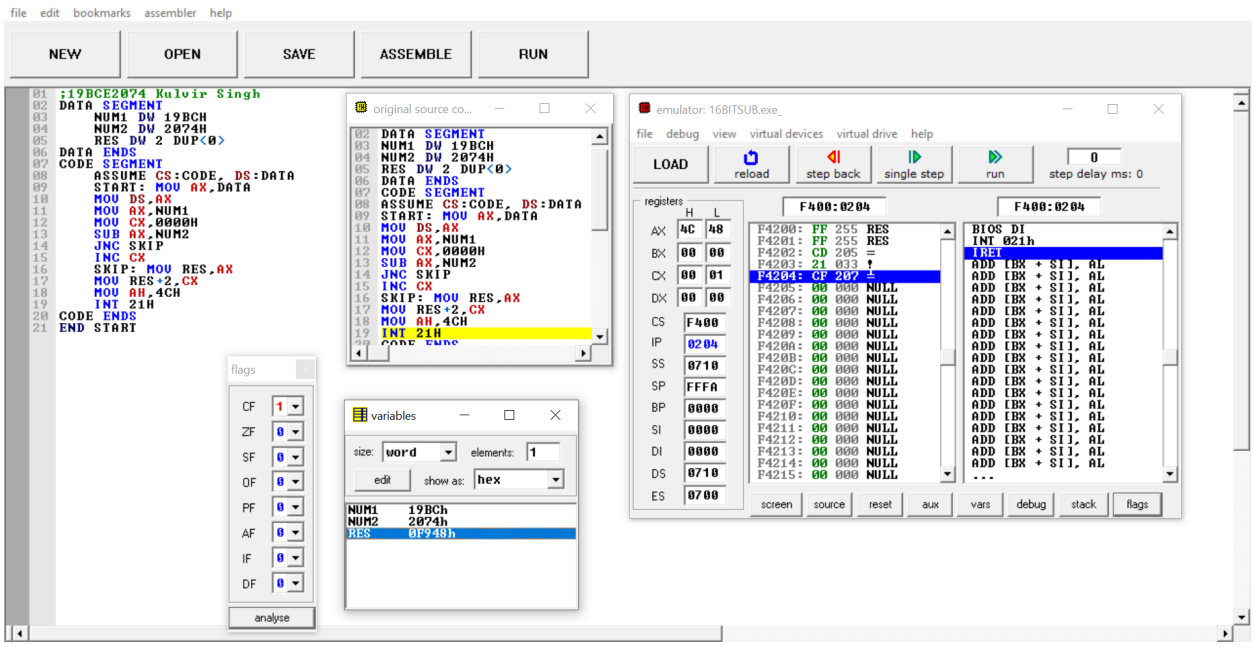


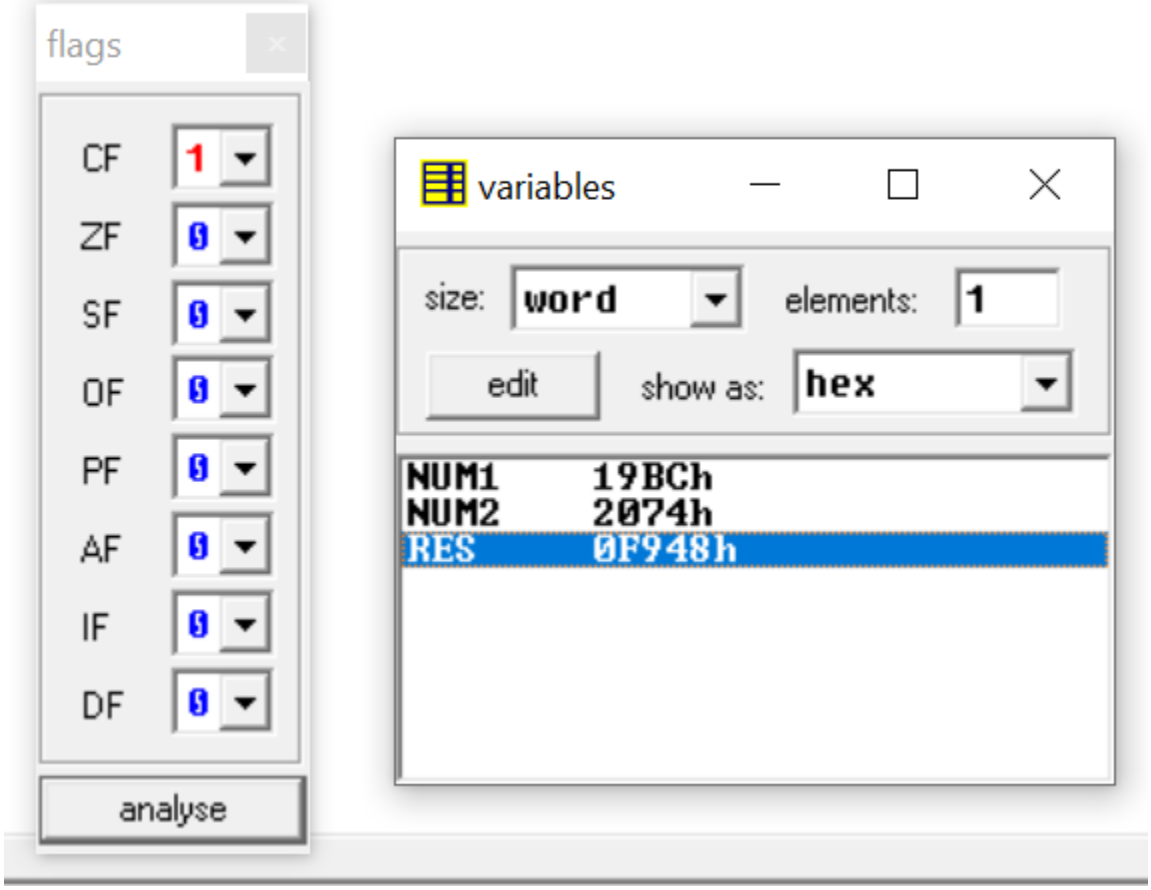
19BCE2074 KULVIR SINGH

* Program : 16 Bit Subtraction.

```
DATA SEGMENT
    NUM1  DW  19BCH ; NUM1 to store 1st no. for sub
    NUM2  DW  2074H ; NUM2 to store 2nd no. for sub
    RES   DW  2 DUP(0) ; result variable for storing
                         difference
DATA ENDS
CODE SEGMENT
    Assume CS: Code, DS: Data
START: MOV AX, Data ; general purpose register
    MOV DS, AX ;      to store segments
    MOV AX, NUM1 ;    AX ← (NUM1) ⇒ 19BC H
    MOV CX, 0000H ;   clear carry flag for borrow.
    SUB AX, NUM2 ;    AX ← AX − NUM2
    JNC SKIP ;        Skip if there is no carry.
    INC CX ;          if borrow present (i.e) carry
                      present increment CX by 1.
SKIP: MOV RES, AX ;   store result of subtraction
                      in result variable
    MOV RES+2, CX ;   store the borrow value
    MOV AH, 4CH ;     terminate
    INT 21H
CODE ENDS
END START
```

**Screenshots :**


file edit bookmarks assembler help
NEW
OPEN
SAVE
ASSEMBLE
RUN
;19BCE2074 Kulvir Singh
DATA SEGMENT
NUM1 DW 19BCH
NUM2 DW 2074H
RES DW 2 DUP(0)
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START: MOV AX,DATA
MOV DS,AX
MOV AX,NUM1
MOV CX,0000H
SUB AX,NUM2
JNC SKIP
INC CX
SKIP: MOV RES,AX
MOV RES+2,CX
MOV AH,4CH
INT 21H
CODE ENDS
END START
original source co...
emulator: 16BITSUB.exe_
file debug view virtual devices virtual drive help
LOAD
reload
step back
single step
run
step delay ms: 0
registers
F400:0204
BIOS DI
INT 021h
screen source reset aux vars debug stack flags
flags
CF
ZF
SF
OF
PF
AF
IF
DF
analyse
variables
size: word
elements: 1
edit
show as: hex
NUM1 19BCh
NUM2 2074h
RES 0F948h


flags
CF 1
ZF 0
SF 0
OF 0
PF 0
AF 0
IF 0
DF 0
analyse
variables
size: word
elements: 1
edit
show as: hex
NUM1 19BCh
NUM2 2074h
RES 0F948h

### Inference :

The program can successfully subtract two 16 bit numbers and store the correct result in the variable.

## 3) 32 Bit Addition

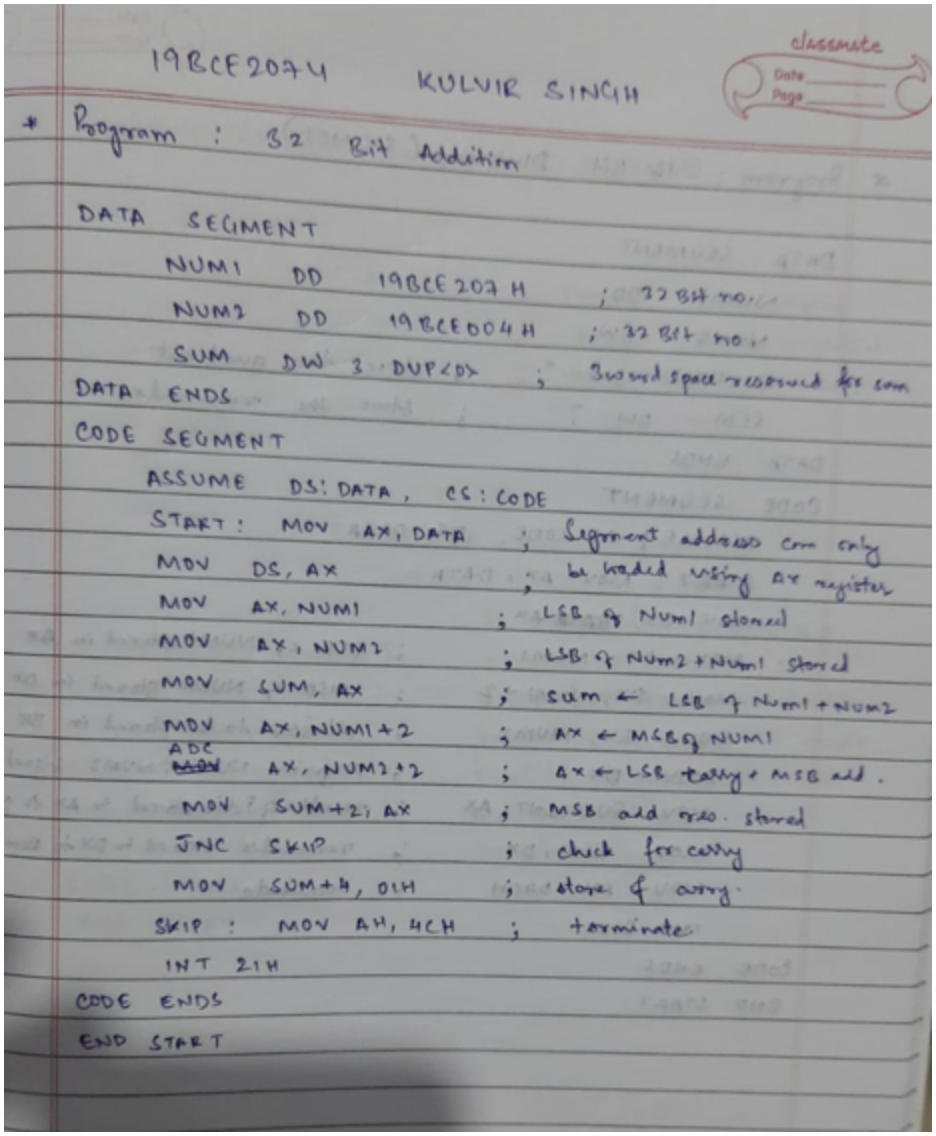
### Aim :

Write a program in 8086 Assembly Language to add 2 32 bit numbers and store the result in a variable

### Requirements :

8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

### Handwritten Program :



19BCE2074 KULVIR SINGH

Assignment

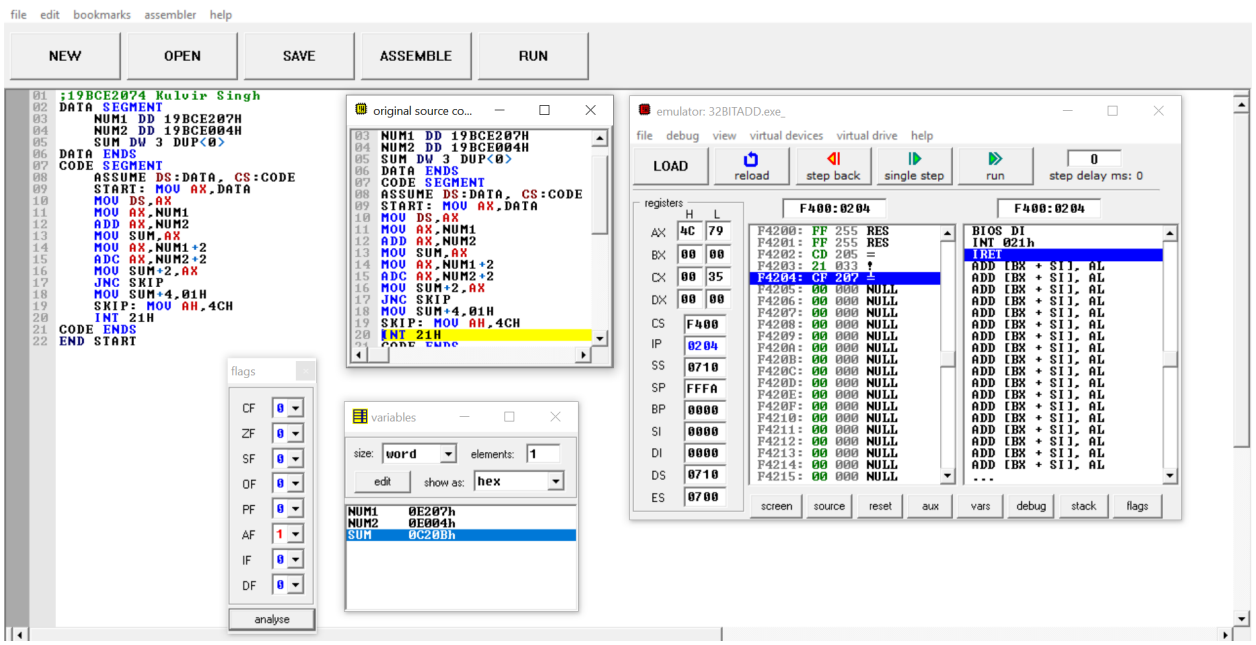
Date

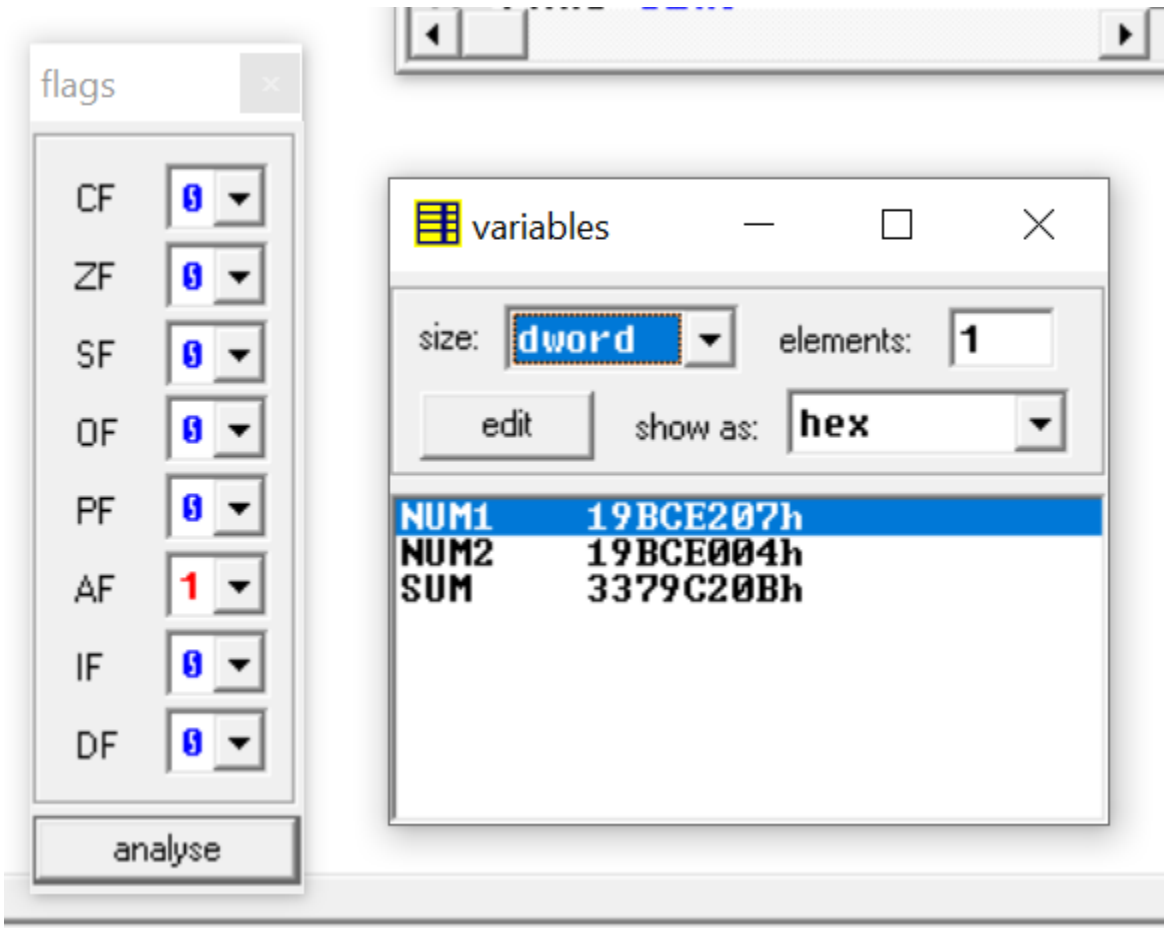
Page

* Program : 32 Bit Addition

```
DATA SEGMENT
    NUM1  DD  19BCE207H   ; 32 Bit no.
    NUM2  DD  19BCED04H   ; 32 Bit no.
    SUM   DW  3 DUP(0)    ; 3 word space reserved for sum
DATA ENDS
CODE SEGMENT
    ASSUME DS: DATA , CS: CODE
    START: MOV AX, DATA   ; Segment address can only
    MOV DS, AX            ; be loaded using AX register
    MOV AX, NUM1          ; LSB of Num1 stored
    MOV AX, NUM2          ; LSB of Num2 + Num1 stored
    MOV SUM, AX           ; sum ← LSB of Num1 + NUM2
    MOV AX, NUM1+2        ; AX ← MSB of NUM1
    ADC AX, NUM2+2        ; AX ← LSB carry + MSB add.
    MOV SUM+2, AX         ; MSB add res. stored
    JNC SKIP              ; check for carry
    MOV SUM+4, 01H        ; store of carry.
SKIP: MOV AH, 4CH         ; terminate
    INT 21H
CODE ENDS
END START
```

**Screenshots :**


file edit bookmarks assembler help
NEW
OPEN
SAVE
ASSEMBLE
RUN
;19BCE2074 Kulvir Singh
DATA SEGMENT
NUM1 DD 19BCE207H
NUM2 DD 19BCE004H
SUM DW 3 DUP(0)
DATA ENDS
CODE SEGMENT
ASSUME DS:DATA, CS:CODE
START: MOV AX,DATA
MOV DS,AX
MOV AX,NUM1
ADD AX,NUM2
MOV SUM,AX
MOV AX,NUM1+2
ADC AX,NUM2+2
MOV SUM+2,AX
JNC SKIP
MOV SUM+4,01H
SKIP: MOV AH,4CH
INT 21H
CODE ENDS
END START
original source co...
flags
CF
ZF
SF
OF
PF
AF
IF
DF
analyse
variables
size: word
elements: 1
edit
show as: hex
NUM1 0E207h
NUM2 0E004h
SUM 0C20Bh
emulator: 32BITADD.exe_
file debug view virtual devices virtual drive help
LOAD
reload
step back
single step
run
step delay ms: 0
registers
F400:0204
BIOS DI
INT 021h
IRET
screen
source
reset
aux
vars
debug
stack
flags


flags
CF
ZF
SF
OF
PF
AF
IF
DF
analyse
variables
size: dword
elements: 1
edit
show as: hex
NUM1 19BCE207h
NUM2 19BCE004h
SUM 3379C20Bh

### Inference :

The program can successfully add two 32 bit numbers and store the correct result in the variable.

## 4) 32 Bit Subtraction

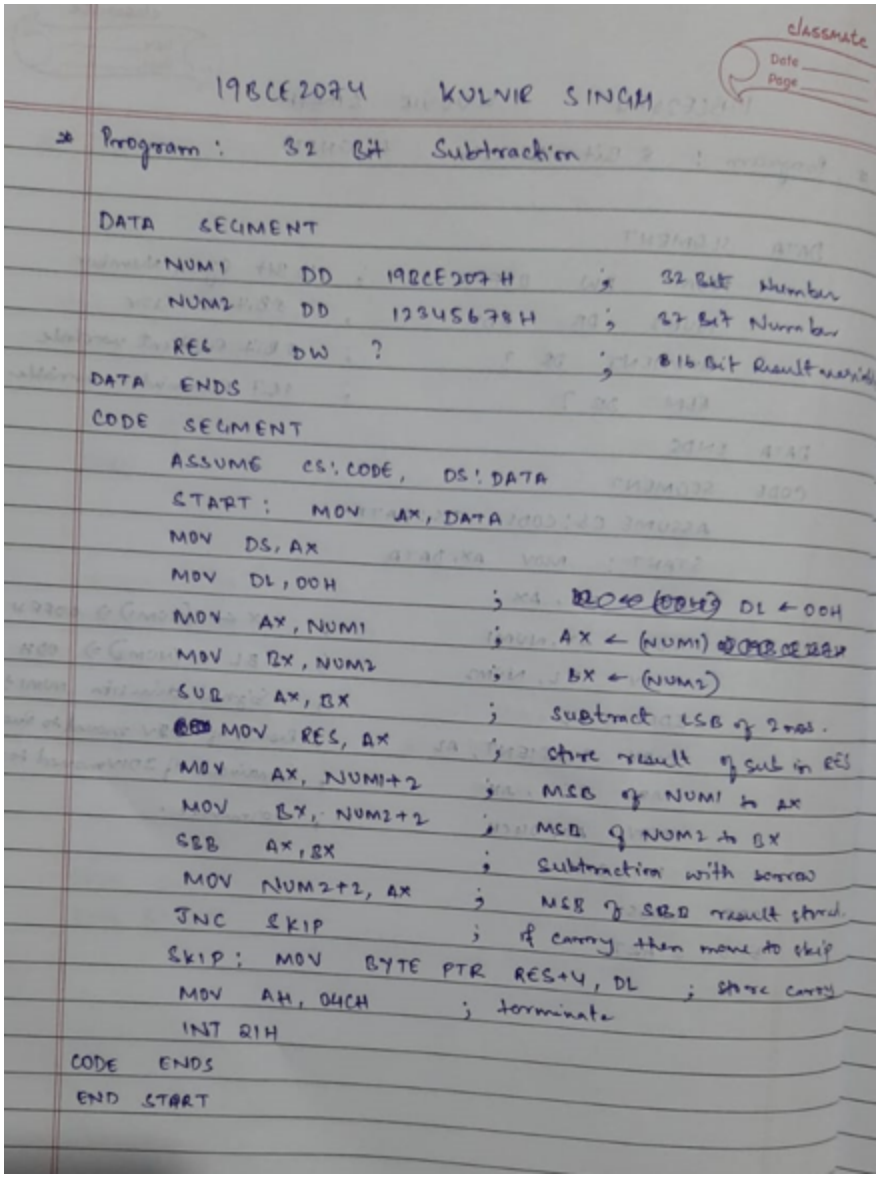
### Aim :

Write a program in 8086 Assembly Language to subtract 2 32 bit numbers and store the result in a variable

### Requirements :

8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

### Handwritten Program :

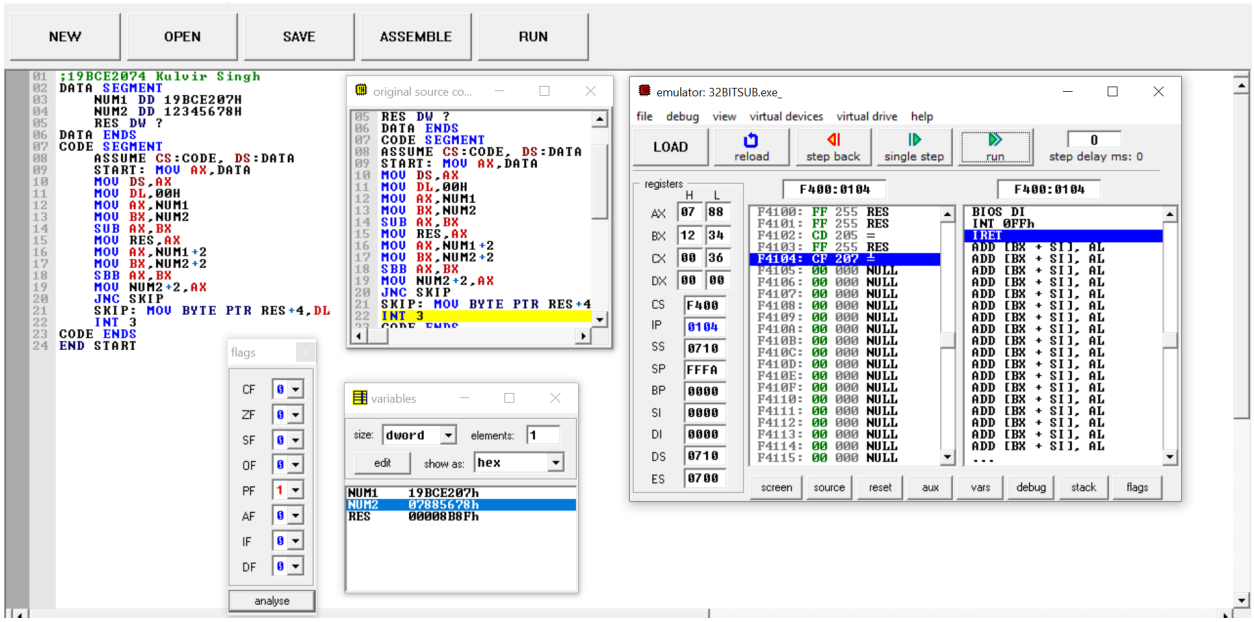


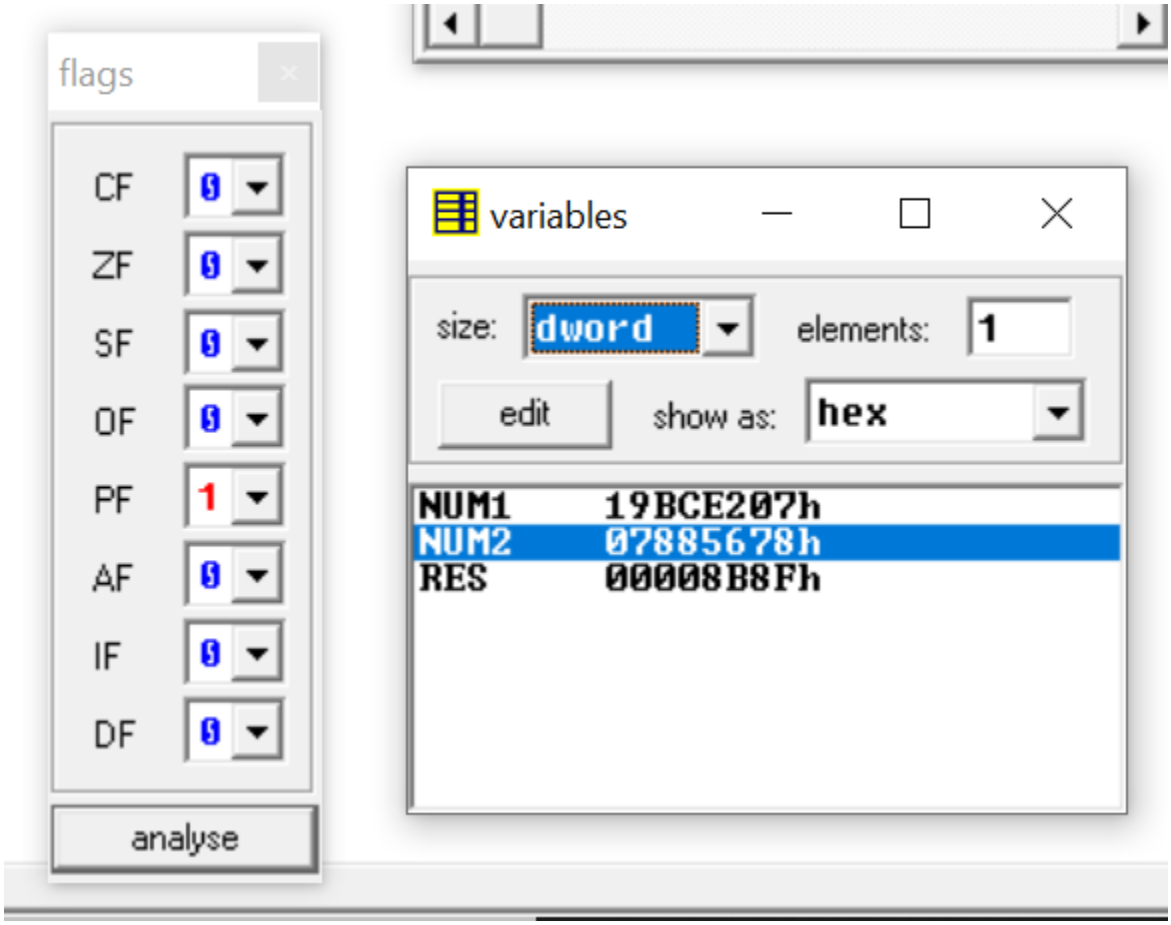
19BCE2074 KULVIR SINGH

Program : 32 Bit Subtraction

```
DATA SEGMENT
    NUM1 DD 19BCE207H      ; 32 Bit Number
    NUM2 DD 12345678H      ; 32 Bit Number
    RES  DW ?              ; 16 Bit Result word
DATA ENDS
CODE SEGMENT
    ASSUME CS: CODE, DS: DATA
    START: MOV AX, DATA
    MOV DS, AX
    MOV DL, 00H            ; DL ← 00H
    MOV AX, NUM1           ; AX ← (NUM1)
    MOV BX, NUM2           ; BX ← (NUM2)
    SUB AX, BX             ; Subtract LSB of 2 nos.
    MOV RES, AX            ; store result of sub in RES
    MOV AX, NUM1+2         ; MSB of NUM1 in AX
    MOV BX, NUM2+2         ; MSB of NUM2 in BX
    SBB AX, BX             ; Subtraction with borrow
    MOV NUM2+2, AX         ; MSB of SBB result stored
    JNC SKIP               ; if carry then move to skip
SKIP: MOV BYTE PTR RES+4, DL ; store carry
    MOV AH, 04CH           ; terminate
    INT 21H
CODE ENDS
END START
```

**Screenshots :**


NEW
OPEN
SAVE
ASSEMBLE
RUN
;19BCE2074 Kulvir Singh
DATA SEGMENT
NUM1 DD 19BCE207H
NUM2 DD 12345678H
RES DW ?
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START: MOV AX,DATA
MOV DS,AX
MOV DL,00H
MOV AX,NUM1
MOV BX,NUM2
SUB AX,BX
MOV RES,AX
MOV AX,NUM1+2
MOV BX,NUM2+2
SBB AX,BX
MOV NUM2+2,AX
JNC SKIP
SKIP: MOV BYTE PTR RES+4,DL
INT 3
CODE ENDS
END START
original source co...
emulator: 32BITSUB.exe_
file debug view virtual devices virtual drive help
LOAD
reload
step back
single step
run
step delay ms: 0
registers
F400:0104
F400:0104
flags
CF
ZF
SF
OF
PF
AF
IF
DF
analyse
variables
size: dword
elements: 1
edit
show as: hex
NUM1 19BCE207h
NUM2 07885678h
RES 00008B8Fh


flags
CF
ZF
SF
OF
PF
AF
IF
DF
analyse
variables
size: dword
elements: 1
edit
show as: hex
NUM1 19BCE207h
NUM2 07885678h
RES 00008B8Fh

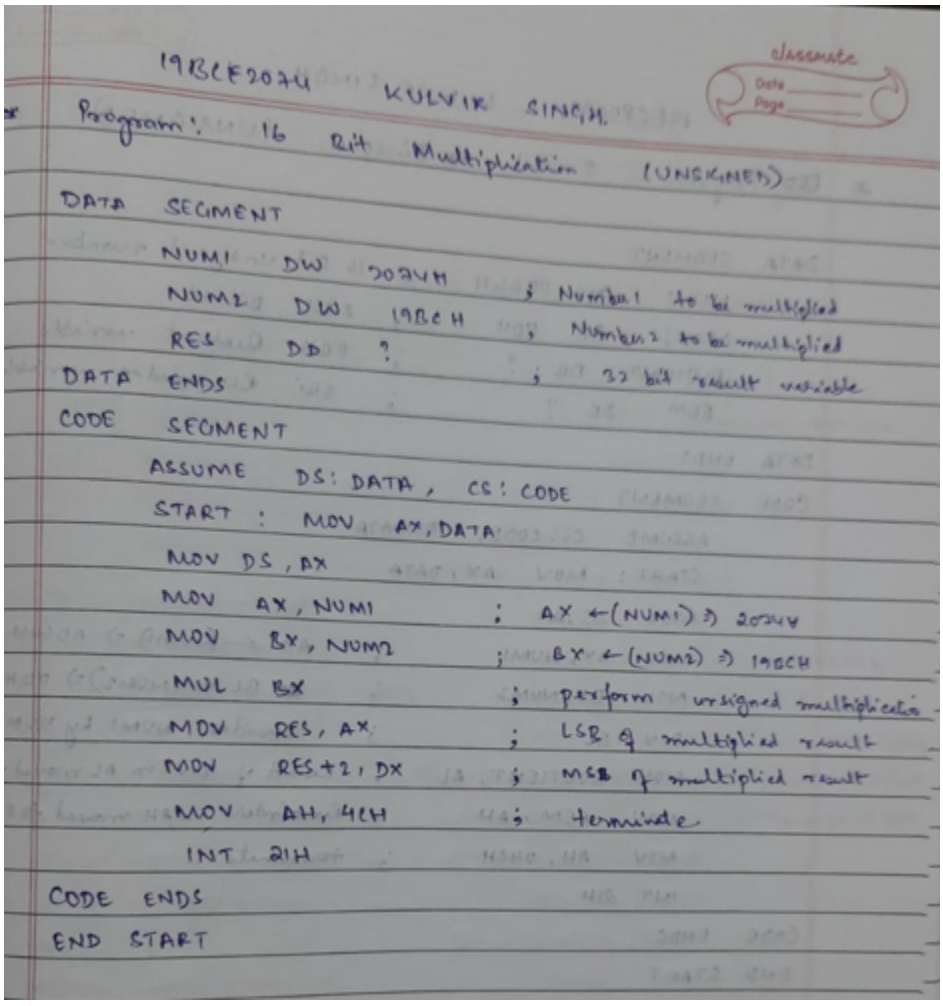
### Inference :
The program can successfully subtract two 32 bit numbers and store the correct result in the variable.

## 5) 16 Bit Multiplication UNSIGNED

### Aim :
Write a program in 8086 Assembly Language to multiply 2 16 bit UNSIGNED numbers and store the result in a variable

### Requirements :
8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

### Handwritten Program :

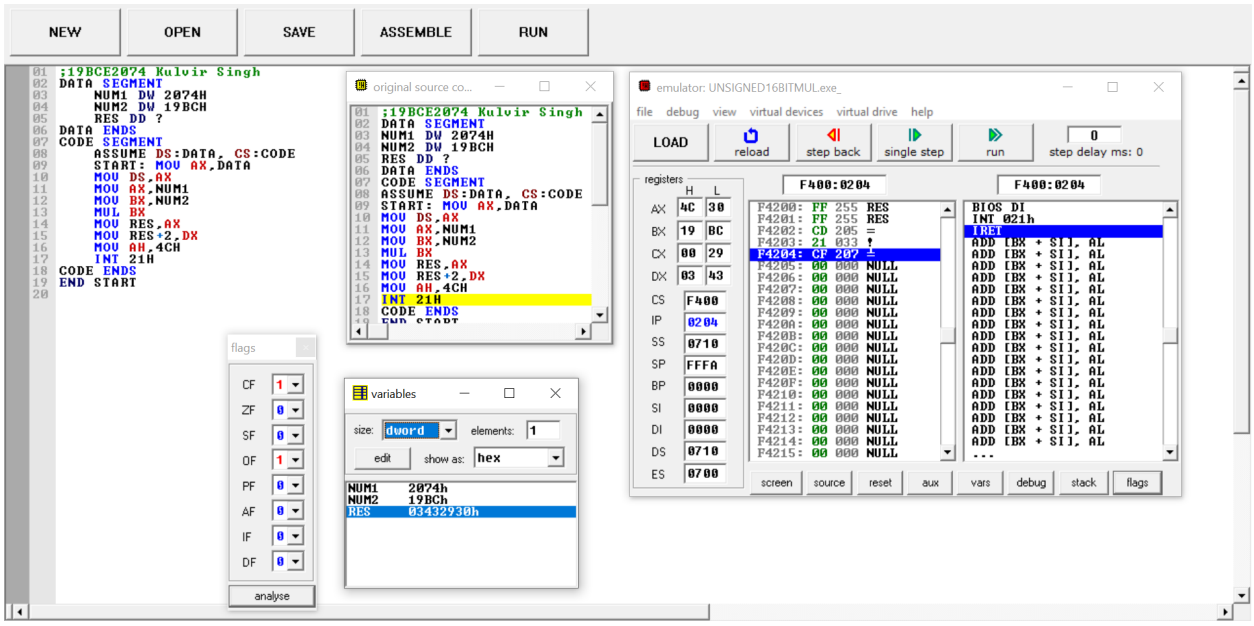


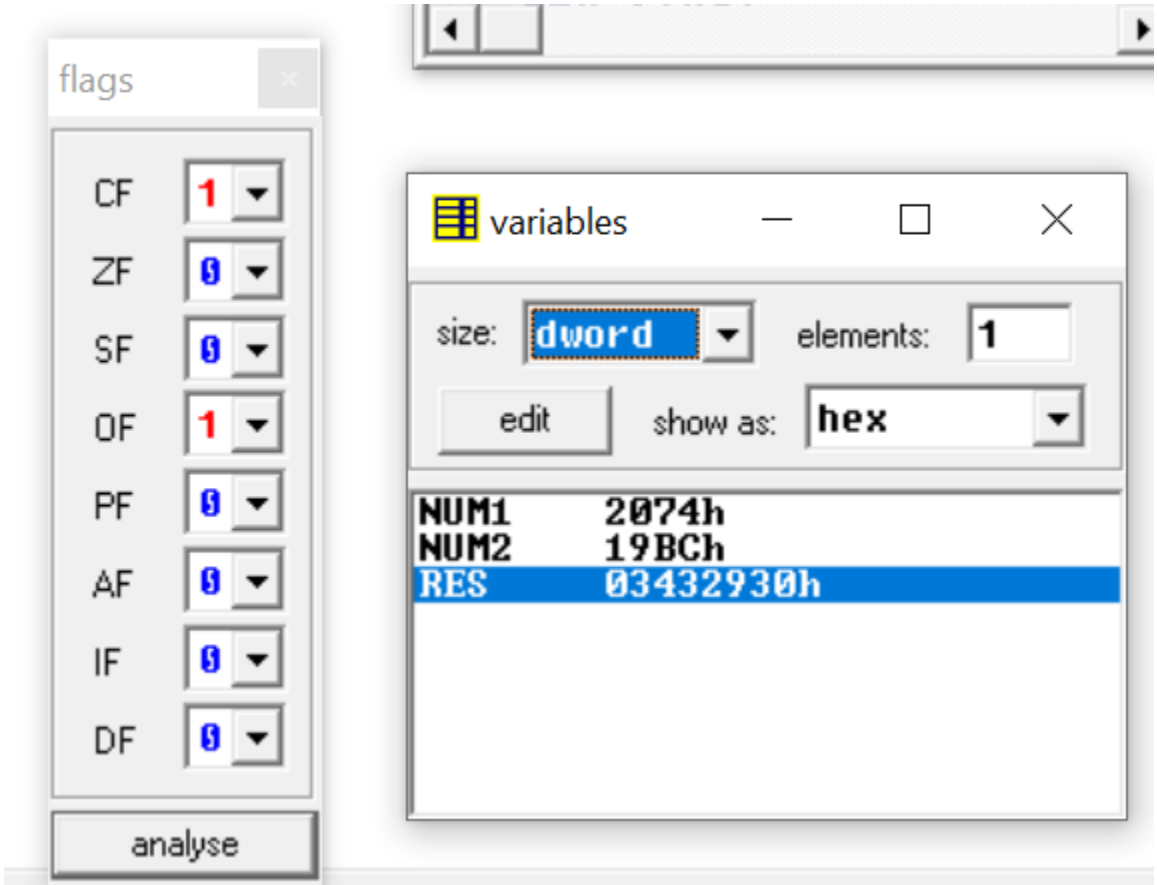
```
19BCE2074   KULVIR SINGH.

Program: 16 Bit Multiplication (UNSIGNED)

DATA SEGMENT
    NUM1 DW 2024H   ; Number1 to be multiplied
    NUM2 DW 190CH   ; Number2 to be multiplied
    RES  DD ?       ; 32 bit result variable
DATA ENDS
CODE SEGMENT
    ASSUME DS: DATA, CS: CODE
    START : MOV AX, DATA
    MOV DS, AX
    MOV AX, NUM1    ; AX ← (NUM1) ⇒ 2024H
    MOV BX, NUM2    ; BX ← (NUM2) ⇒ 190CH
    MUL BX          ; perform unsigned multiplicatio
    MOV RES, AX     ; LSB of multiplied result
    MOV RES+2, DX   ; MSB of multiplied result
    MOV AH, 4CH     ; terminate
    INT 21H
CODE ENDS
END START
```

## Screenshots :



## Inference :

The program can successfully multiply two 16 bit UNSIGNED numbers and store the correct result in the variable.

# 6) 16 Bit Multiplication SIGNED

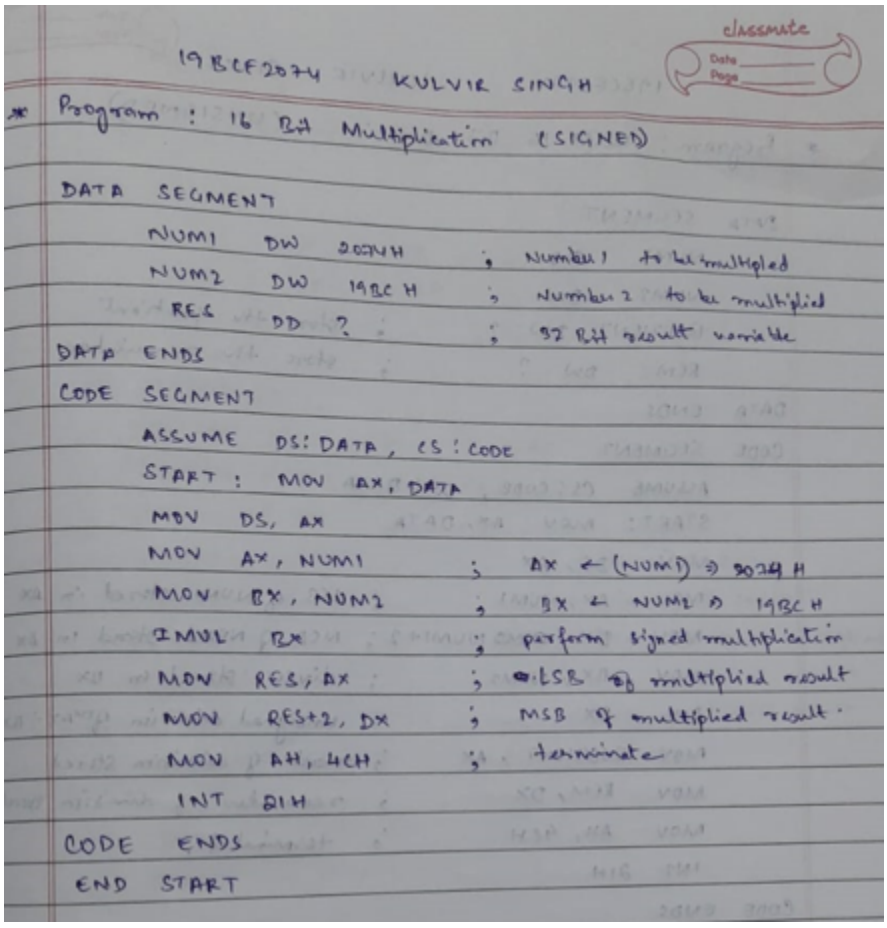
## Aim :

Write a program in 8086 Assembly Language to multiply 2 16 bit SIGNED numbers and store the result in a variable

## Requirements :

8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

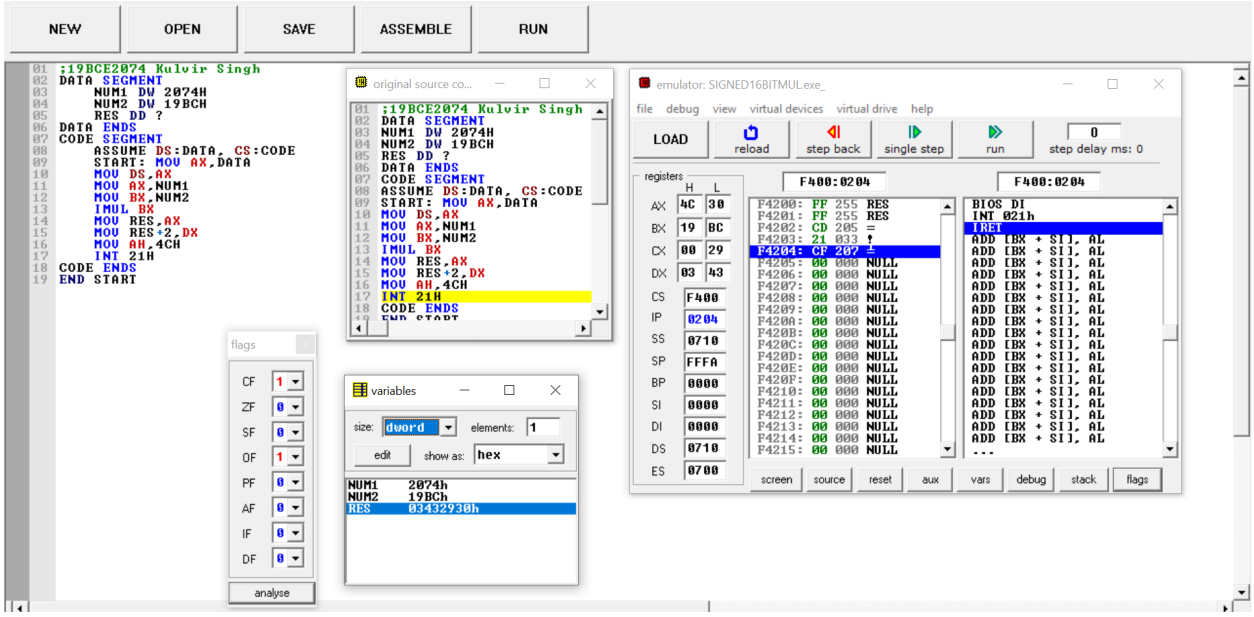
## Handwritten Program :

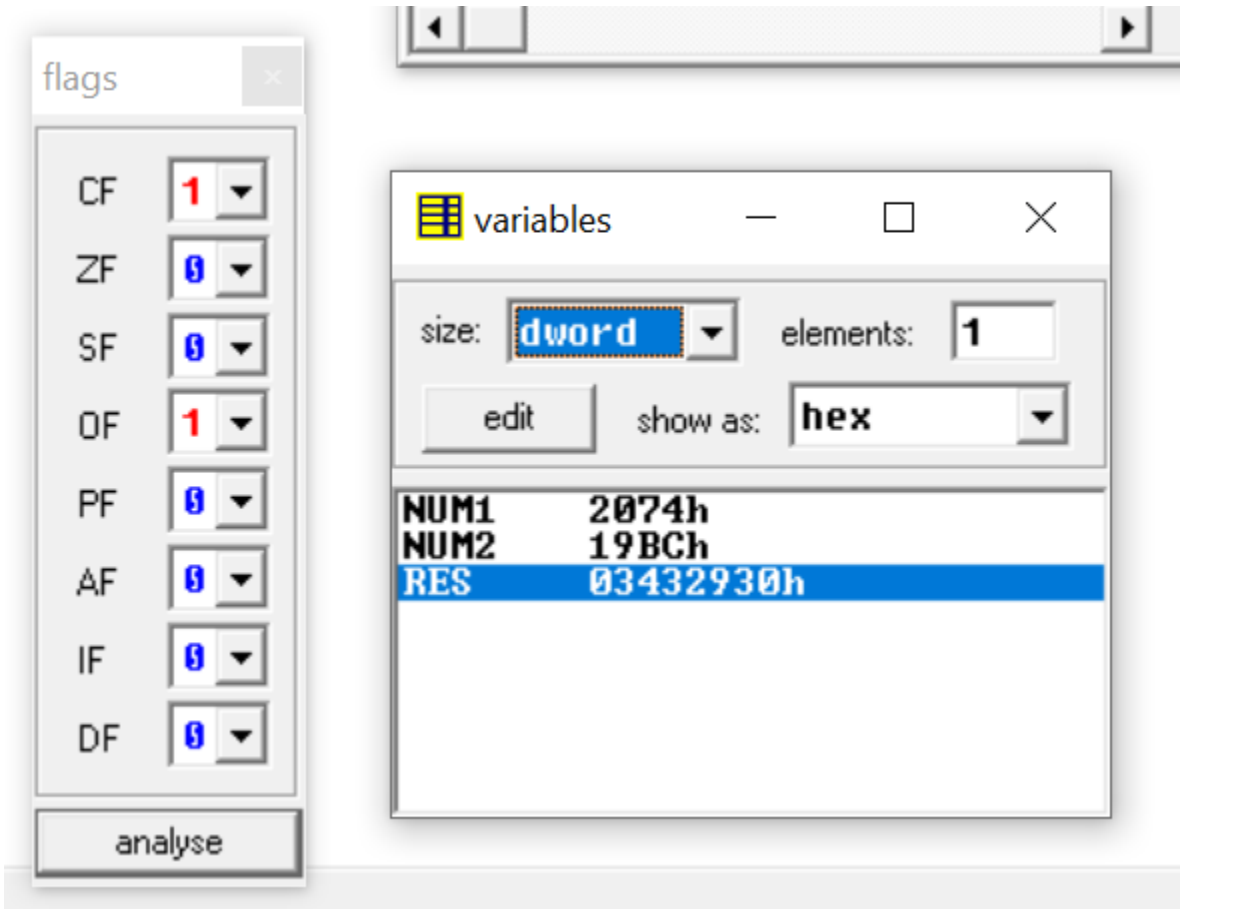


```
19BCE2074  KULVIR SINGH                    classmate
                                           Date
                                           Page
* Program : 16 Bit Multiplication (SIGNED)

DATA SEGMENT
     NUM1  DW  3074H   ; Number 1 to be multiplied
     NUM2  DW  19BCH   ; Number 2 to be multiplied
     RES   DD  ?       ; 32 Bit result variable
DATA ENDS
CODE SEGMENT
     ASSUME DS: DATA, CS: CODE
     START: MOV AX, DATA
     MOV DS, AX
     MOV AX, NUM1      ; AX ← (NUM1) ⇒ 3074 H
     MOV BX, NUM2      ; BX ← NUM2 ⇒ 19BC H
     IMUL BX           ; perform signed multiplication
     MOV RES, AX       ; LSB of multiplied result
     MOV RES+2, DX     ; MSB of multiplied result.
     MOV AH, 4CH       ; terminate
     INT 21H
CODE ENDS
END START
```

## Screenshots :



## Inference :

The program can successfully multiply two 16 bit SIGNED numbers and store the correct result in the variable.

# 7) 16 Bit Division UNSIGNED

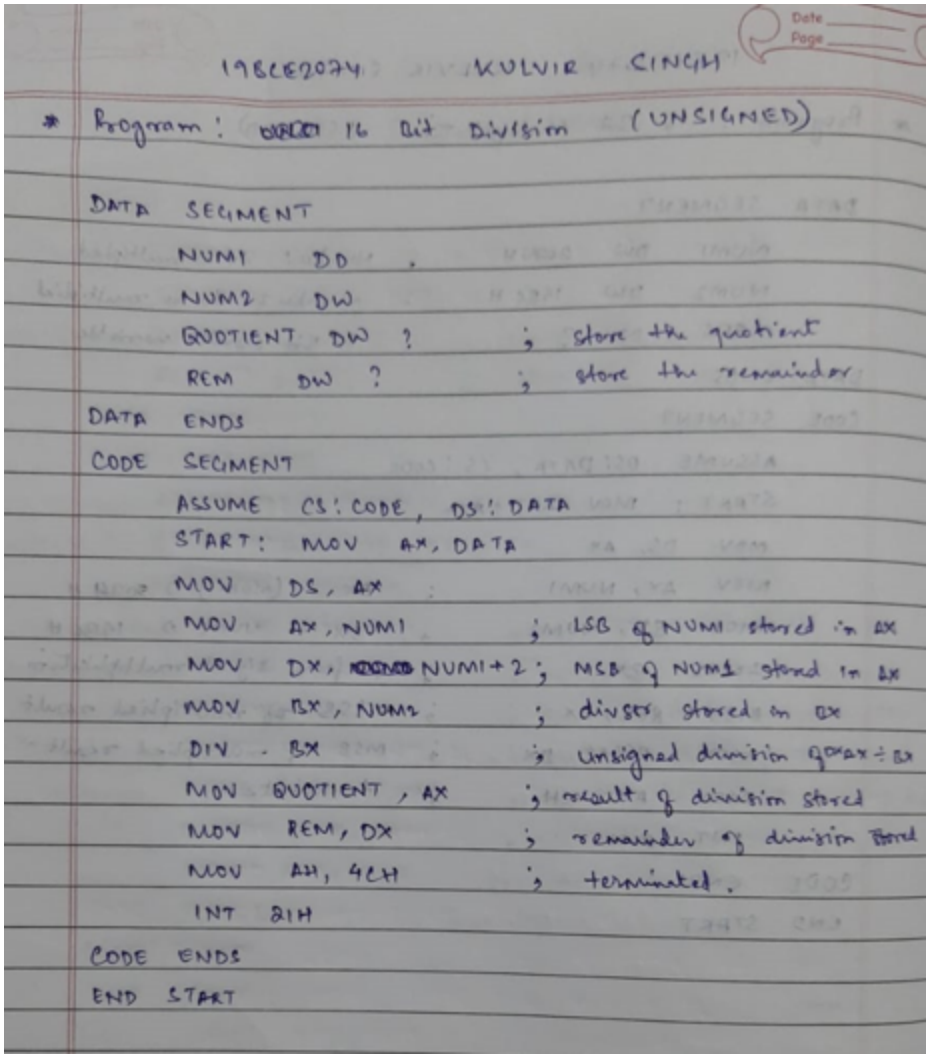
## Aim :

Write a program in 8086 Assembly Language to divide 2 16 bit UNSIGNED numbers and store the result in a variable

## Requirements :

8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

## Handwritten Program :

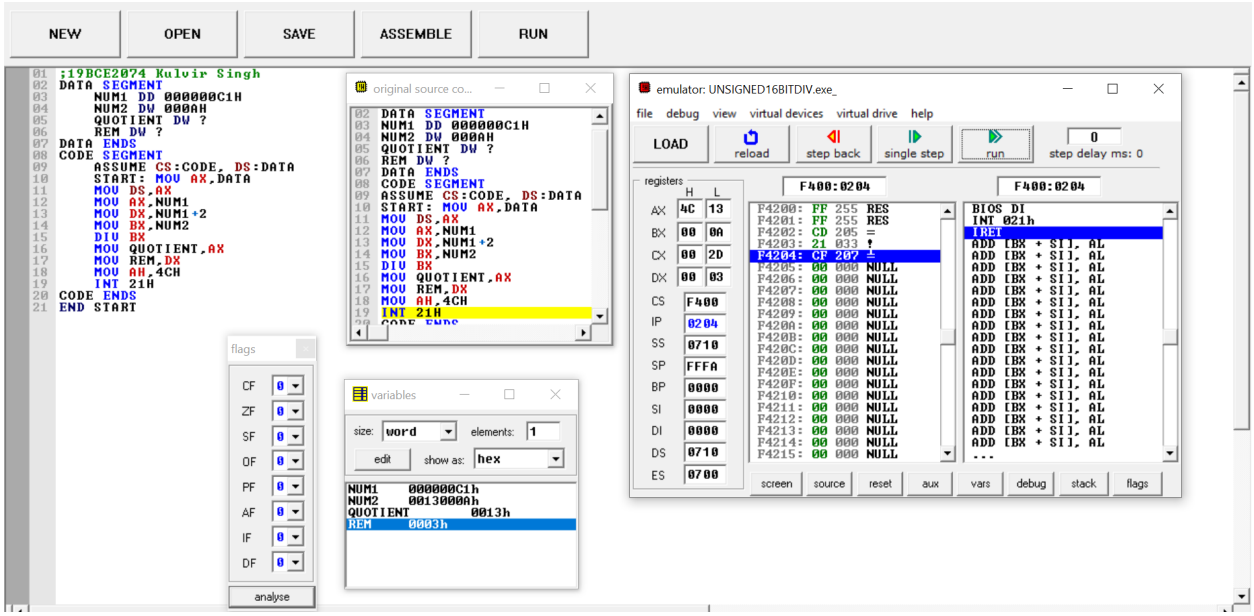


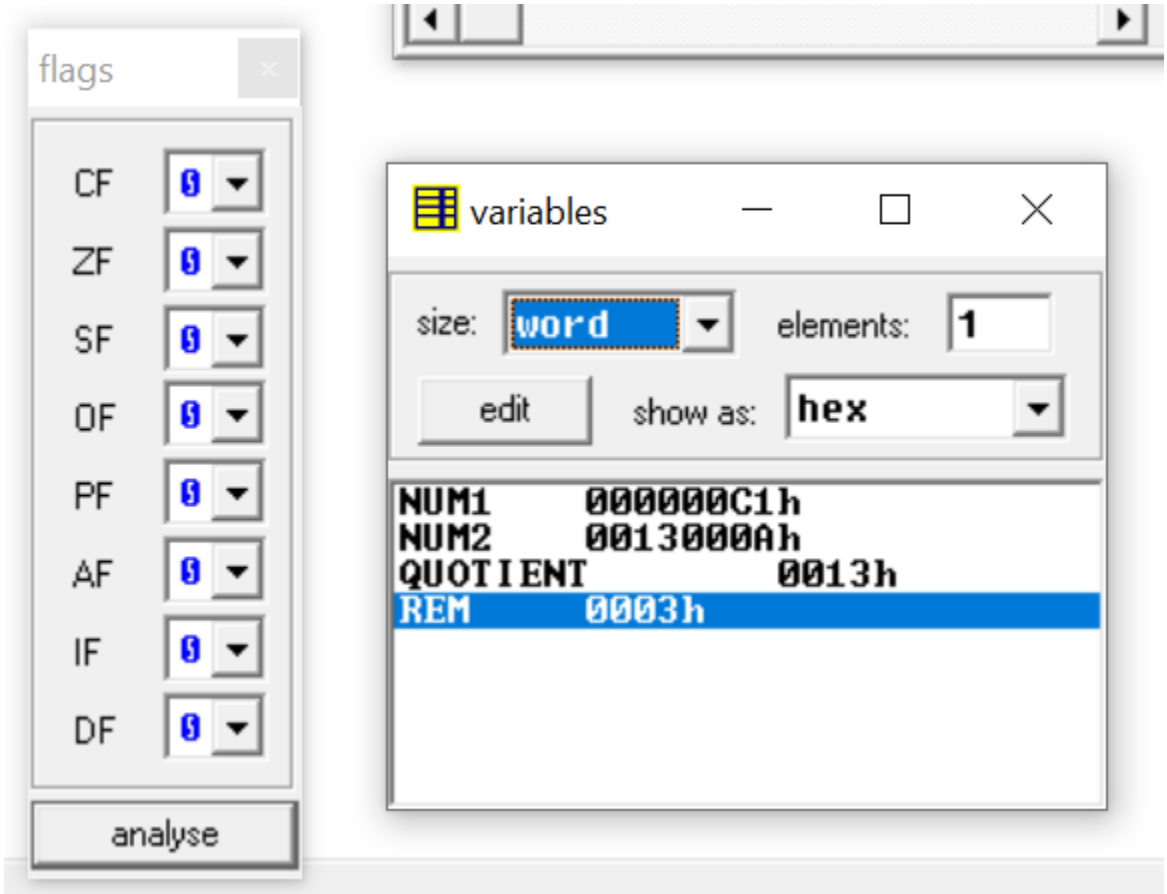
19BCE2074 KULVIR CINGH

* Program : 16 Bit Division (UNSIGNED)

```
DATA  SEGMENT
      NUM1   DD  .
      NUM2   DW
      QUOTIENT  DW  ?        ; store the quotient
      REM    DW  ?           ; store the remainder
DATA  ENDS
CODE  SEGMENT
      ASSUME  CS: CODE , DS: DATA
      START:  MOV  AX, DATA
      MOV   DS, AX
      MOV   AX, NUM1         ; LSB of NUM1 stored in AX
      MOV   DX, NUM1+2       ; MSB of NUM1 stored in AX
      MOV   BX, NUM2         ; divisor stored in BX
      DIV   BX               ; unsigned division of AX ÷ BX
      MOV   QUOTIENT , AX    ; result of division stored
      MOV   REM, DX          ; remainder of division stored
      MOV   AH, 4CH          ; terminated.
      INT   21H
CODE  ENDS
END   START
```

**Screenshots :**




## Inference :

The program can successfully divide two 16 bit UNSIGNED numbers and store the correct result in the variable.

# 8) 16 Bit Division SIGNED

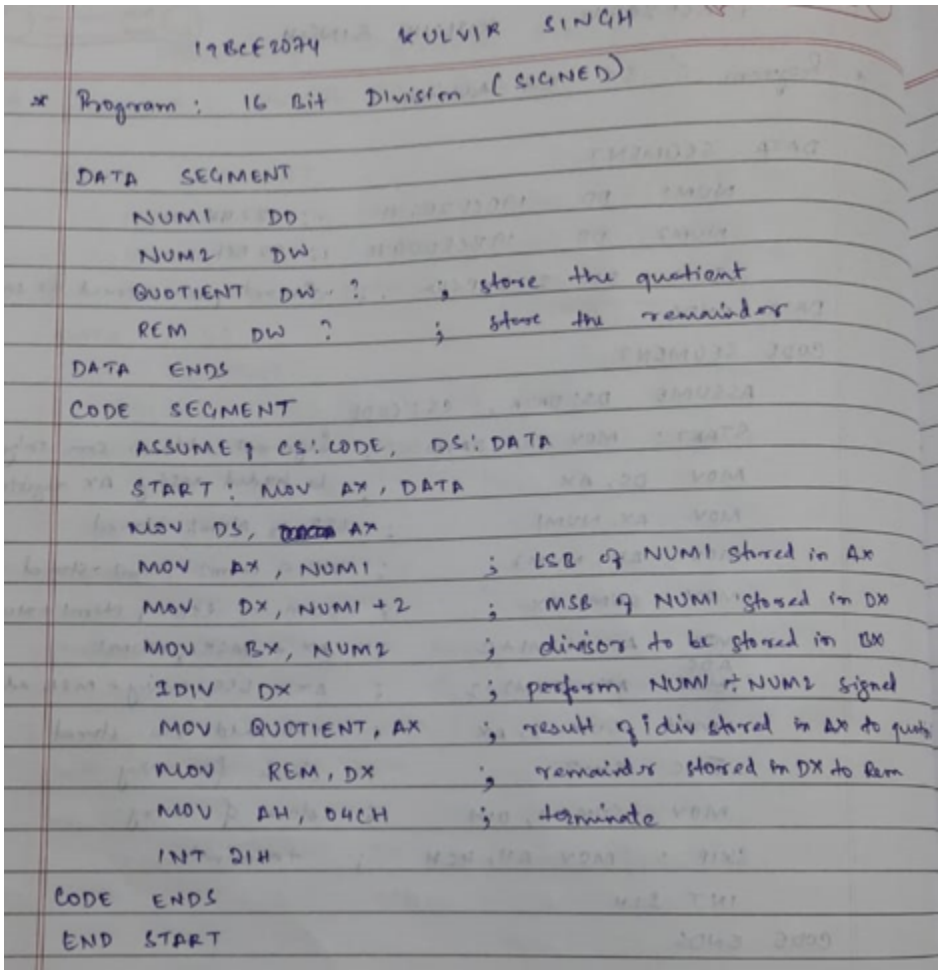
## Aim :

Write a program in 8086 Assembly Language to divide 2 16 bit SIGNED numbers and store the result in a variable

## Requirements :

8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

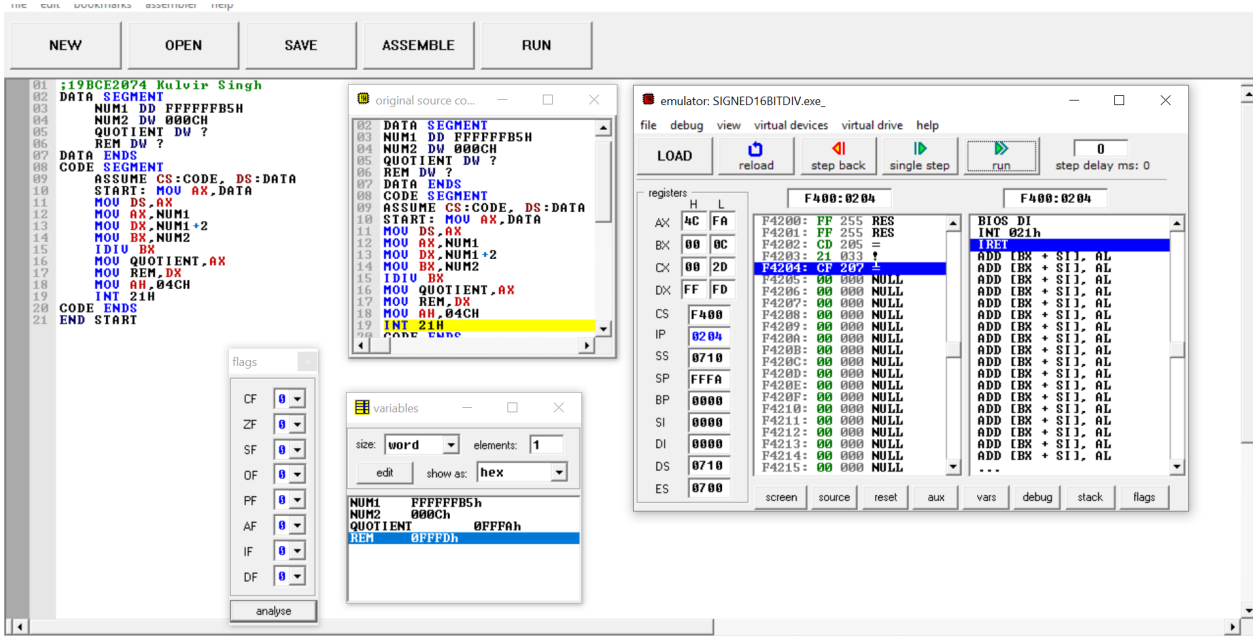
## Handwritten Program :

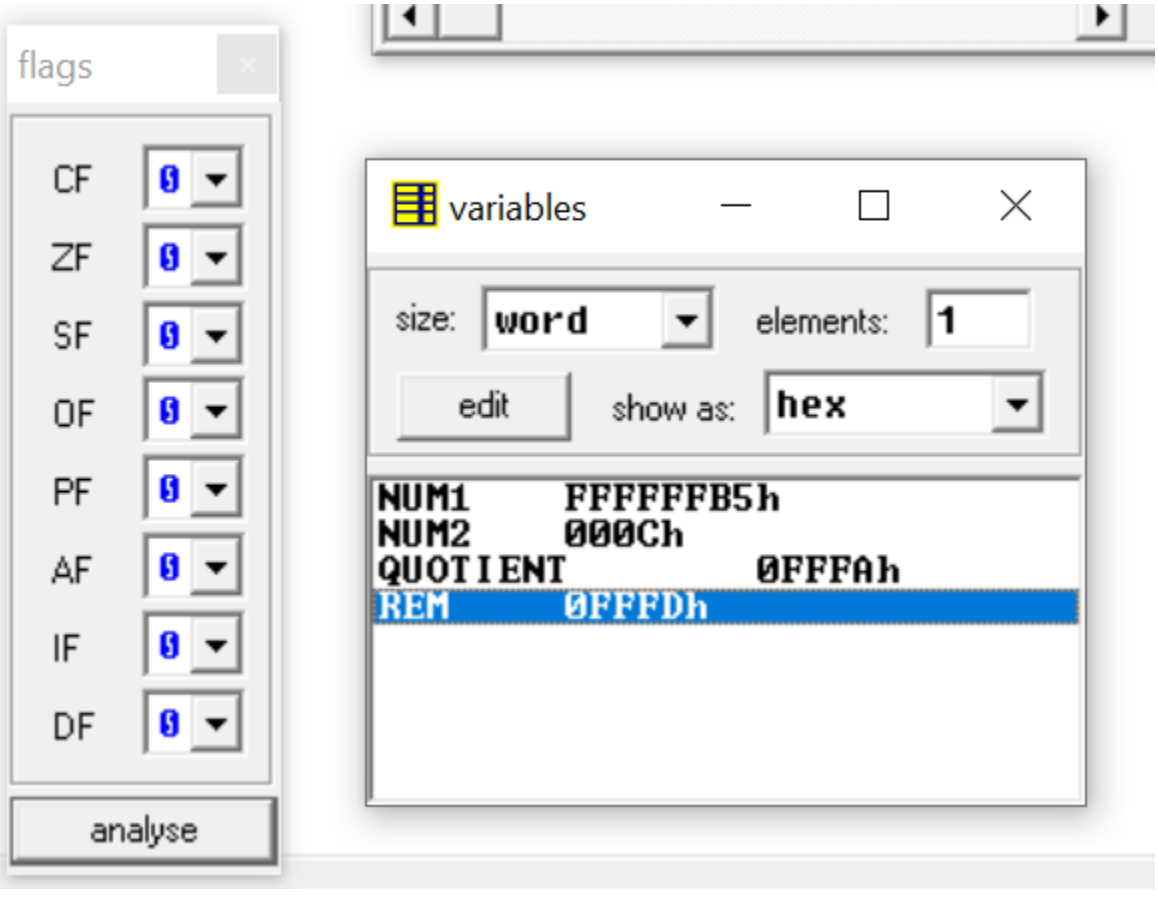


```
19BCE2074   KULVIR SINGH
* Program : 16 Bit Division (SIGNED)

DATA SEGMENT
    NUM1 DD
    NUM2 DW
    QUOTIENT DW ?    ; store the quotient
    REM DW ?         ; store the remainder
DATA ENDS
CODE SEGMENT
    ASSUME CS: CODE, DS: DATA
    START: MOV AX, DATA
    MOV DS, AX
    MOV AX, NUM1     ; LSB of NUM1 stored in AX
    MOV DX, NUM1+2   ; MSB of NUM1 stored in DX
    MOV BX, NUM2     ; divisor to be stored in BX
    IDIV DX          ; perform NUM1 ÷ NUM2 signed
    MOV QUOTIENT, AX ; result of idiv stored in AX to quot
    MOV REM, DX      ; remainder stored in DX to Rem
    MOV AH, 04CH     ; terminate
    INT 21H
CODE ENDS
END START
```

## Screenshots :


NEW
OPEN
SAVE
ASSEMBLE
RUN
;19BCE2074 Kulvir Singh
DATA SEGMENT
NUM1 DD FFFFFFB5H
NUM2 DW 000CH
QUOTIENT DW ?
REM DW ?
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START: MOV AX,DATA
MOV DS,AX
MOV AX,NUM1
MOV DX,NUM1+2
MOV BX,NUM2
IDIV BX
MOV QUOTIENT,AX
MOV REM,DX
MOV AH,04CH
INT 21H
CODE ENDS
END START
original source co...
emulator: SIGNED16BITDIV.exe_
file debug view virtual devices virtual drive help
LOAD
reload
step back
single step
run
step delay ms: 0
registers
F400:0204
F400:0204
BIOS DI
INT 021h
IRET
screen source reset aux vars debug stack flags
flags
CF ZF SF OF PF AF IF DF
analyse
variables
size: word
elements: 1
edit
show as: hex
NUM1 FFFFFFB5h
NUM2 000Ch
QUOTIENT 0FFFAh
REM 0FFFDh


flags
CF
ZF
SF
OF
PF
AF
IF
DF
analyse
variables
size: word
elements: 1
edit
show as: hex
NUM1 FFFFFFB5h
NUM2 000Ch
QUOTIENT 0FFFAh
REM 0FFFDh

## Inference :

The program can successfully divide two 16 bit SIGNED numbers and store the correct result in the variable.

# 9) 8 Bit Division UNSIGNED

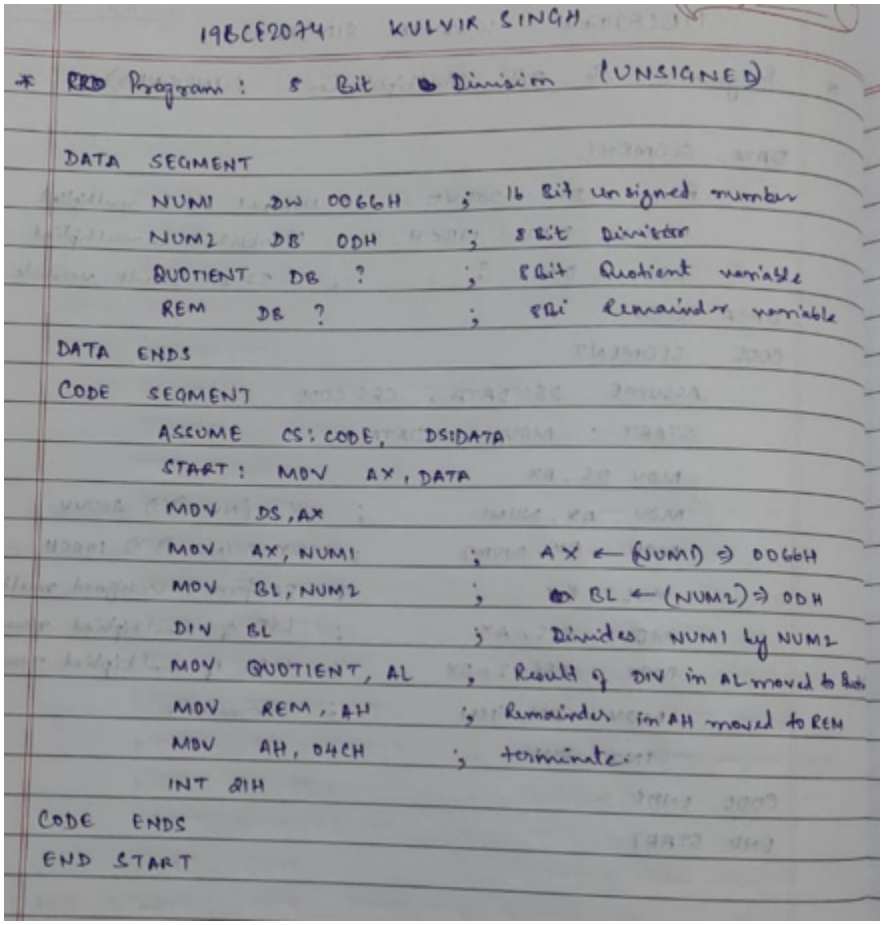
## Aim :

Write a program in 8086 Assembly Language to divide 2 8 bit UNSIGNED numbers and store the result in a variable

## Requirements :

8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

## Handwritten Program :

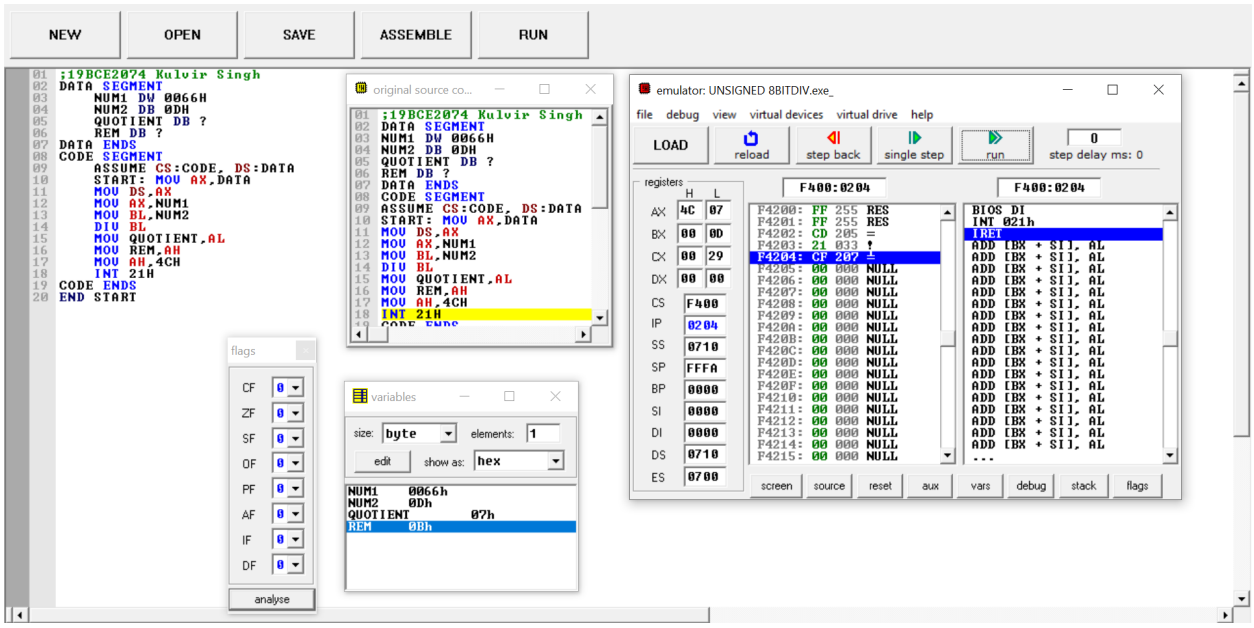


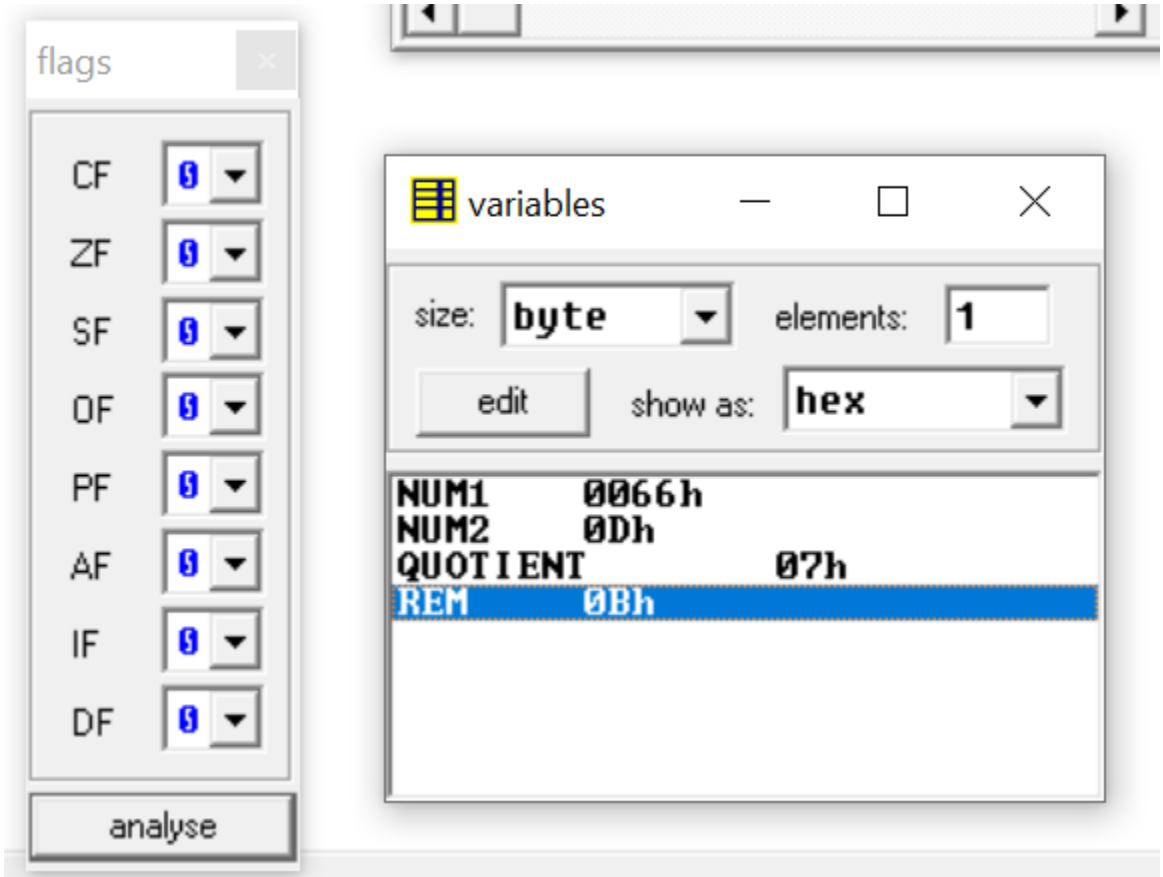
```
19BCE2074 KULVIR SINGH

* Program : 8 Bit Division (UNSIGNED)

DATA SEGMENT
    NUM1 DW 0066H  ; 16 Bit unsigned number
    NUM2 DB 0DH    ; 8 Bit Divisor
    QUOTIENT DB ?  ; 8 Bit Quotient variable
    REM DB ?       ; 8 Bit Remainder variable
DATA ENDS
CODE SEGMENT
    ASSUME CS: CODE, DS:DATA
    START: MOV AX, DATA
    MOV DS, AX
    MOV AX, NUM1   ; AX ← (NUM1) ⇒ 0066H
    MOV BL, NUM2   ; BL ← (NUM2) ⇒ 0DH
    DIV BL         ; Divides NUM1 by NUM2
    MOV QUOTIENT, AL ; Result of DIV in AL moved to
    MOV REM, AH    ; Remainder in AH moved to REM
    MOV AH, 04CH   ; terminate.
    INT 21H
CODE ENDS
END START
```

## Screenshots :



## Inference :

The program can successfully divide two 8 bit UNSIGNED numbers and store the correct result in the variable.

# 10) 8 Bit Division SIGNED

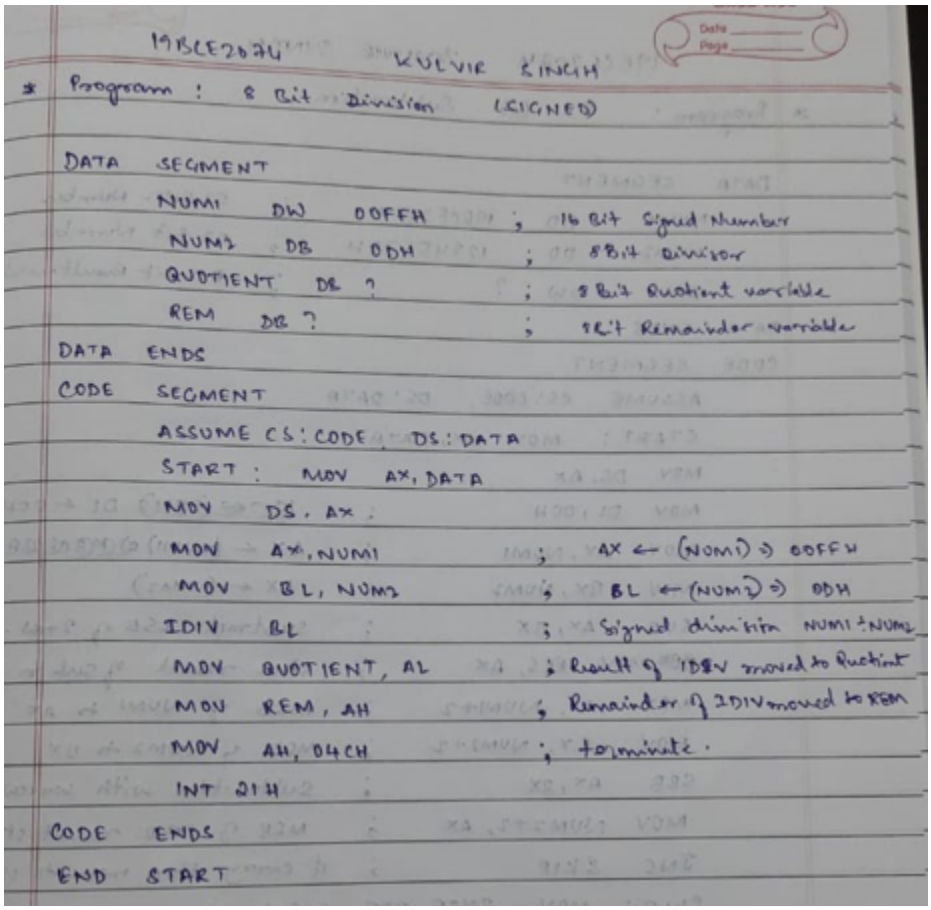
## Aim :

Write a program in 8086 Assembly Language to divide 2 8 bit SIGNED numbers and store the result in a variable

## Requirements :

8086 EMU - An emulator to run the 8086 Assembly Language Code
Operating System - Any valid operating system that can execute the emulator

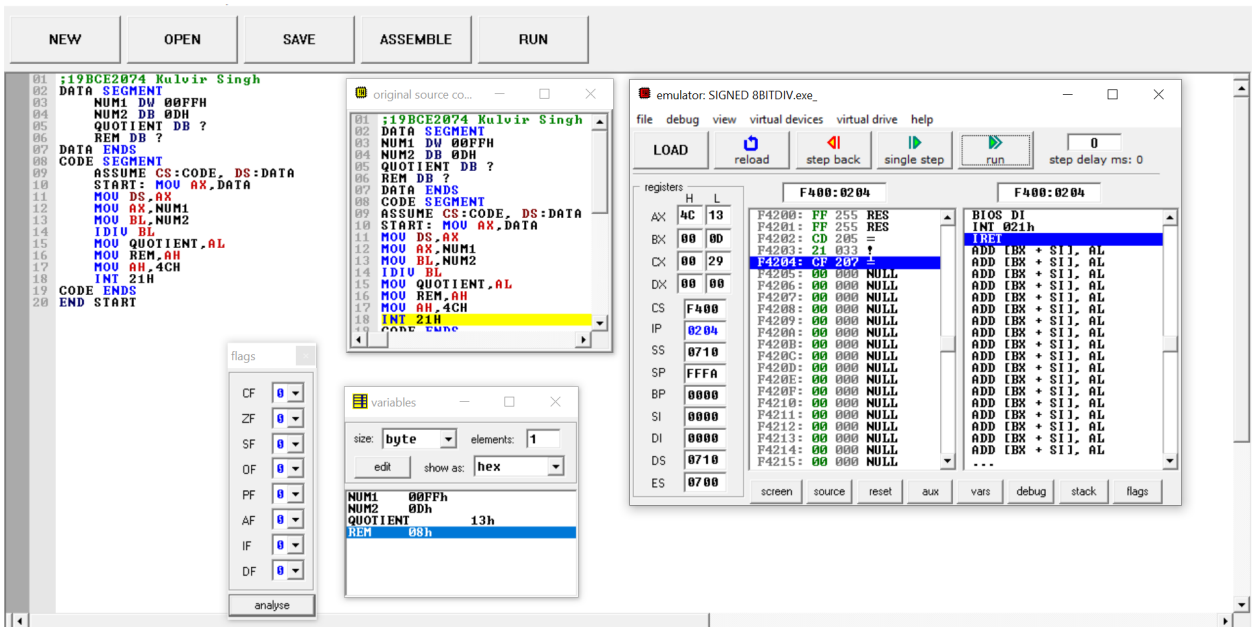
## Handwritten Program :

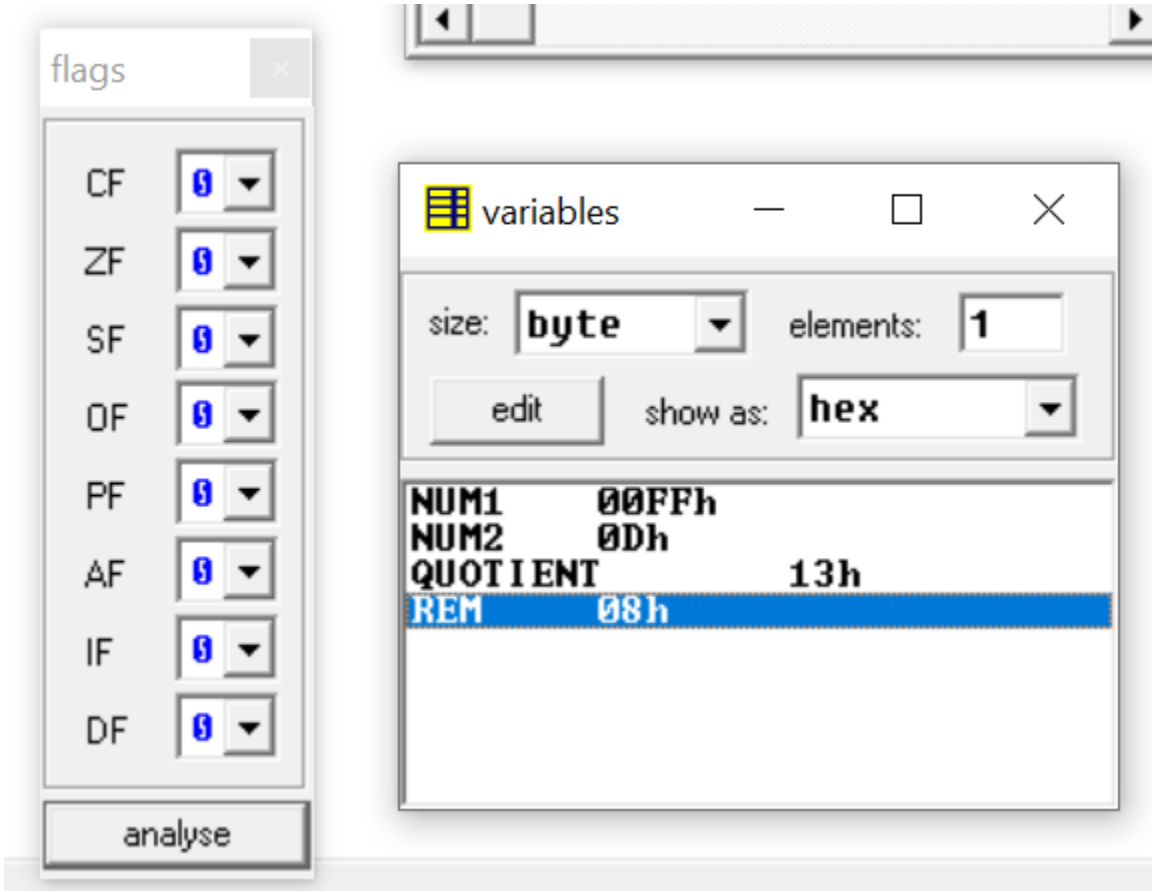


```
19BCE2074  KULVIR RINCHH
* Program : 8 Bit Division (SIGNED)

DATA  SEGMENT
      NUM1  DW  00FFH  ; 16 Bit Signed Number
      NUM2  DB  0DH    ; 8 Bit Divisor
      QUOTIENT  DB ?   ; 8 Bit Quotient variable
      REM   DB ?       ; 8 Bit Remainder variable
DATA  ENDS
CODE  SEGMENT
      ASSUME CS: CODE, DS: DATA
      START : MOV  AX, DATA
      MOV   DS, AX
      MOV   AX, NUM1   ; AX ← (NUM1) ⇒ 00FFH
      MOV   BL, NUM2   ; BL ← (NUM2) ⇒ 0DH
      IDIV  BL         ; Signed division NUM1 ÷ NUM2
      MOV   QUOTIENT, AL ; Result of IDIV moved to Quotient
      MOV   REM, AH    ; Remainder of IDIV moved to REM
      MOV   AH, 04CH   ; to terminate.
      INT 21H
CODE  ENDS
END   START
```

## Screenshots :



## Inference :

The program can successfully divide two 8 bit SIGNED numbers and store the correct result in the variable.