U18CO018

Shubham Shekhaliya

MIT

Assignment-9

1-> Program to multiply signed 16-bit numbers.

**Code:-**

**model small**

**.8086**

**.data**

**a dw 0004H**

**b dw -0002H**

**c dw 0**

**d dw 0**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov ax, a**

**mov bx, b**

**imul bx**

**mov c, ax**

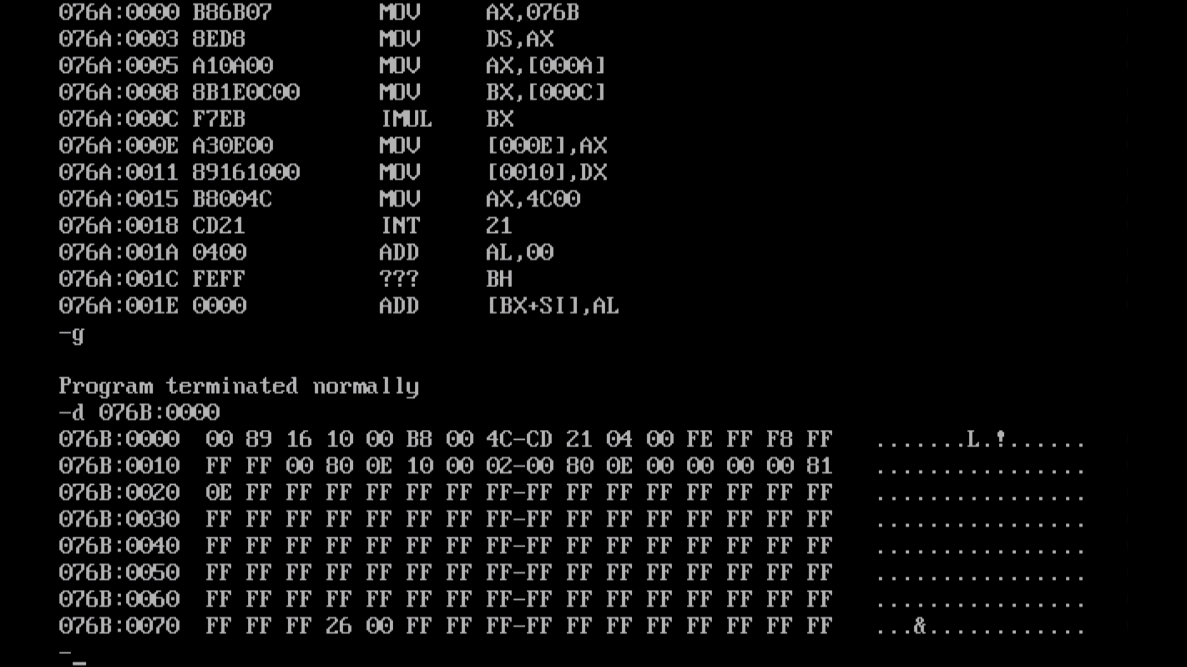
**mov d, dx**

**mov ax, 4C00H**

**int 21h**

**end**

**Output:-**

****

2-> Program to multiply unsigned 16-bit numbers.

**Code:-**

**model small**

**.8086**

**.data**

**a dw 0004H**

**b dw 0FFFEH**

**c dw 0**

**d dw 0**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov ax, a**

**mov bx, b**

**mul bx**

**mov c, ax**

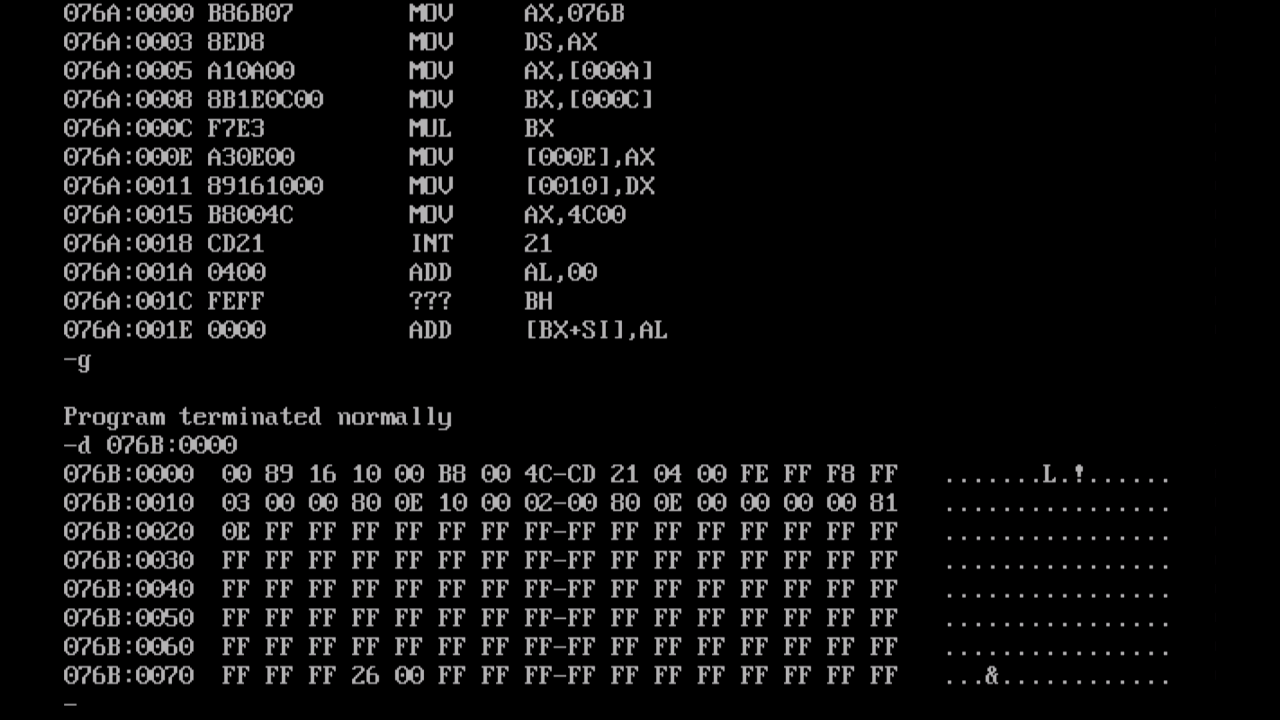
**mov d, dx**

**mov ax, 4C00H**

**int 21h**

**end**

**Output:-**

****

3-> Program for division of unsigned 8-bit numbers.

**Code:-**

**model small**

**.8086**

**.data**

**a db 28H**

**b db 03H**

**c dw ?**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov ax,0000H**

**mov bx,0000H**

**mov al,a**

**mov bl,b**

**div bl**

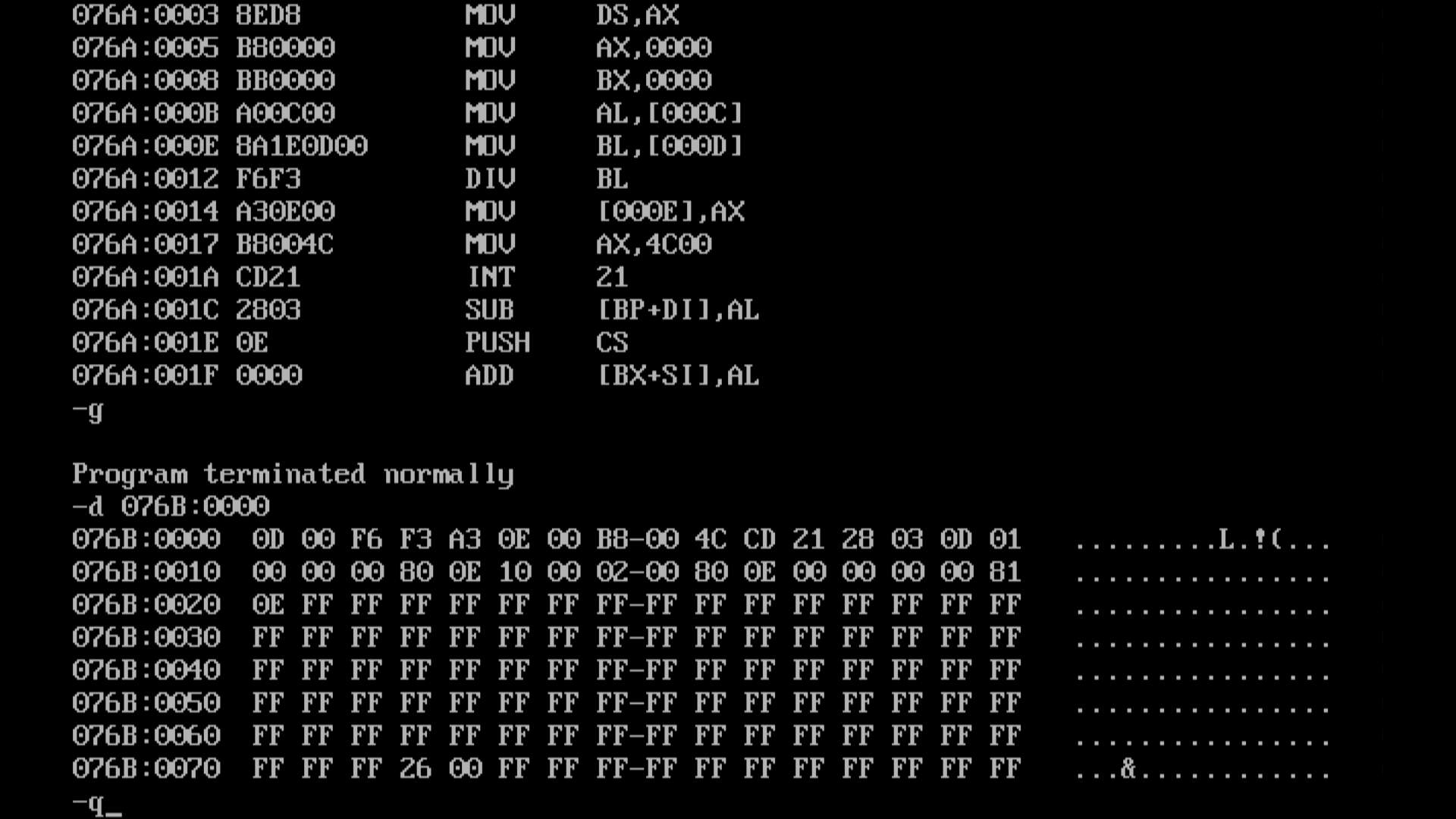
**mov c,ax**

**mov ax, 4C00H**

**int 21h**

**end**

**Output:-**



4-> Program for division of unsigned 16-bit numbers.

**Code:-**

**model small**

**.8086**

**.data**

**a dw 0188H**

**b dw 0012H**

**c dw ?**

**d dw ?**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov ax,0000H**

**mov bx,0000H**

**mov dx,0000H**

**mov ax,a**

**mov bx,b**

**div bx**

**mov c,ax**

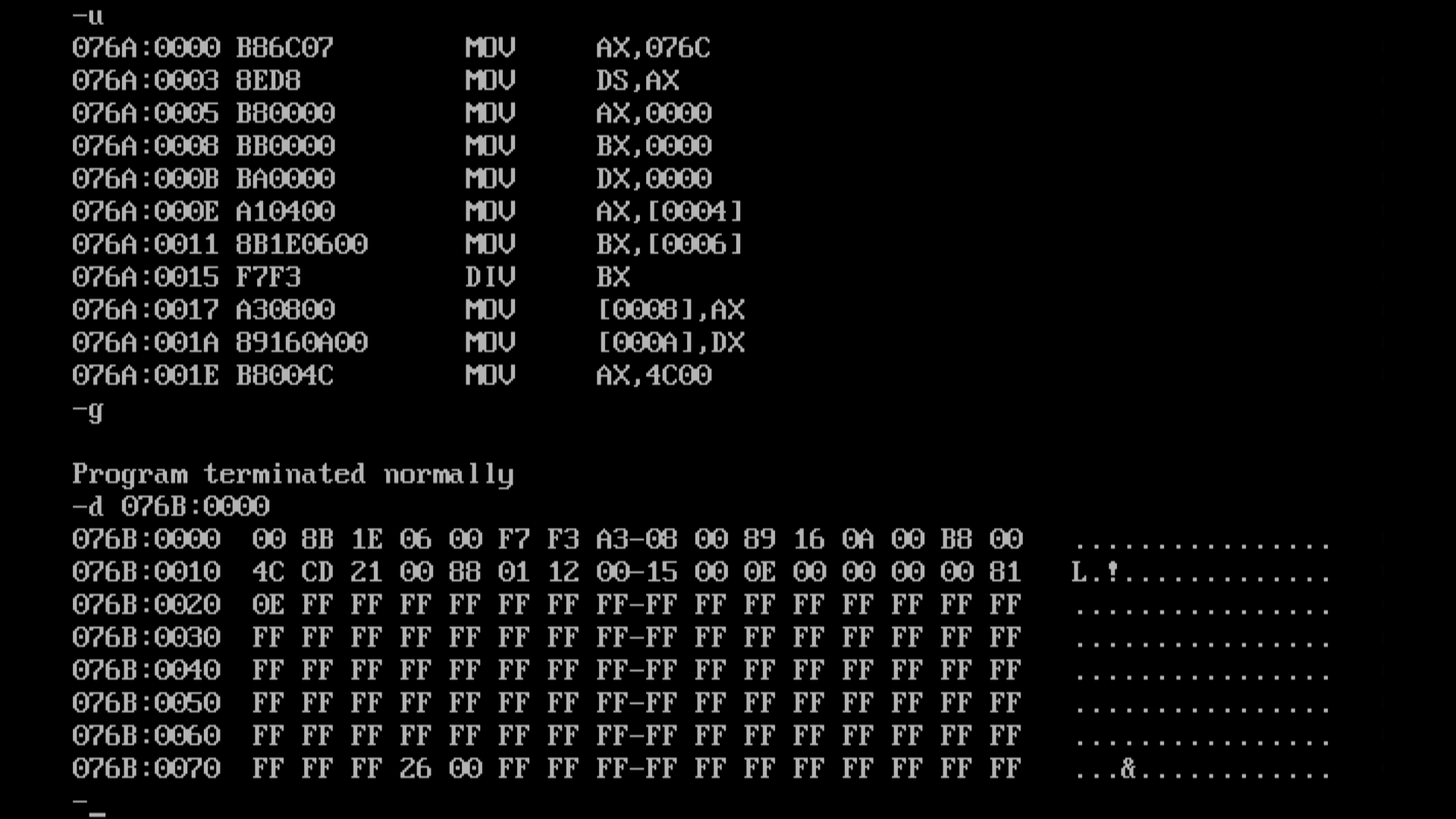
**mov d,dx**

**mov ax, 4C00H**

**int 21h**

**end**

**Output:-**



5-> Program for division of signed 8-bit numbers.

**Code:-**

**model small**

**.8086**

**.data**

**a db 28H**

**b db -03H**

**c dw ?**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov ax,0000H**

**mov bx,0000H**

**mov al,a**

**mov bl,b**

**idiv bl**

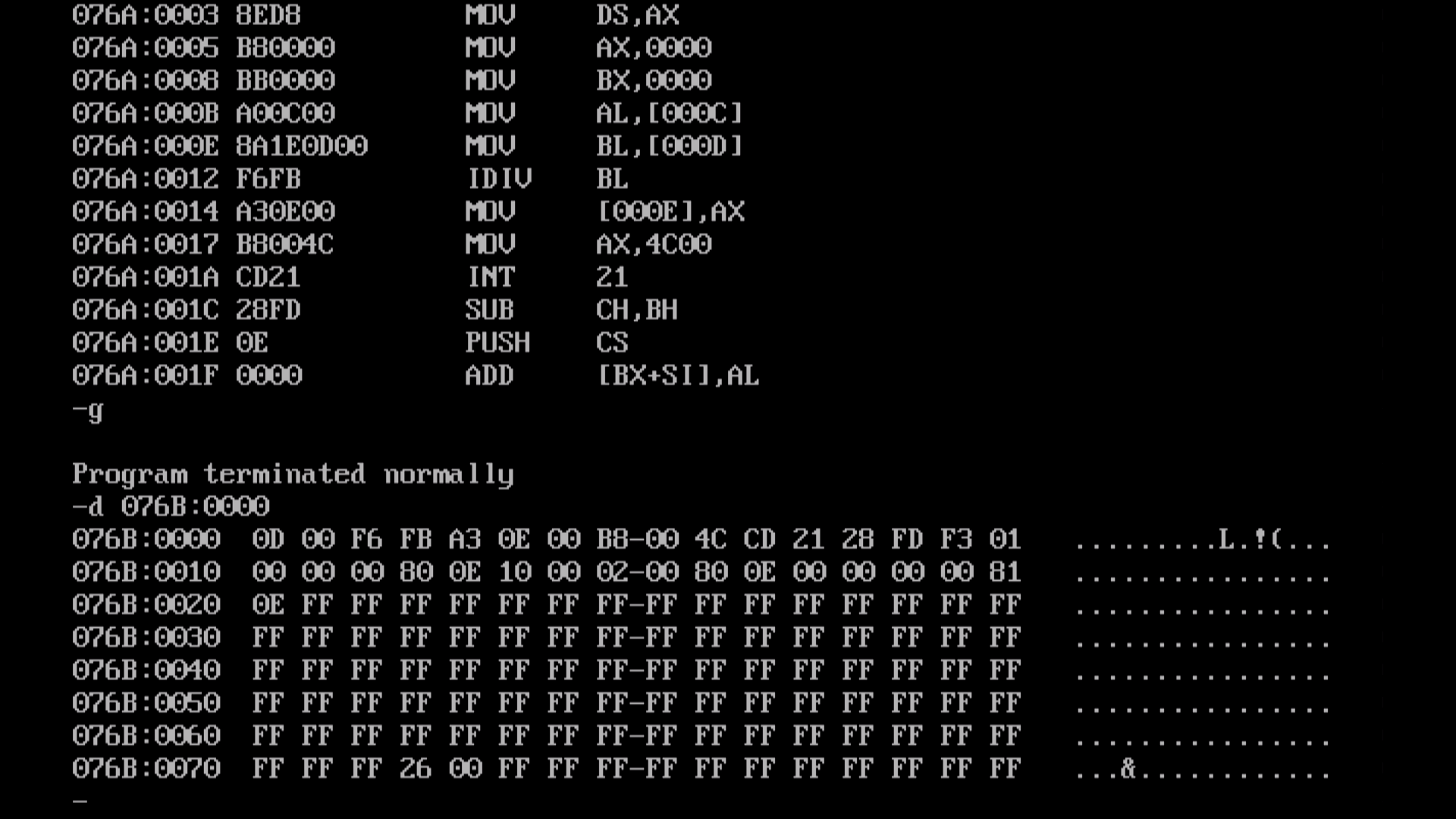
**mov c,ax**

**mov ax, 4C00H**

**int 21h**

**end**

**Output:-**



6-> Program for division of signed 16-bit numbers.

**Code:-**

**model small**

**.8086**

**.data**

**a dw 0188H**

**b dw -0012H**

**c dw ?**

**d dw ?**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov ax,0000H**

**mov bx,0000H**

**mov dx,0000H**

**mov ax,a**

**mov bx,b**

**idiv bx**

**mov c,ax**

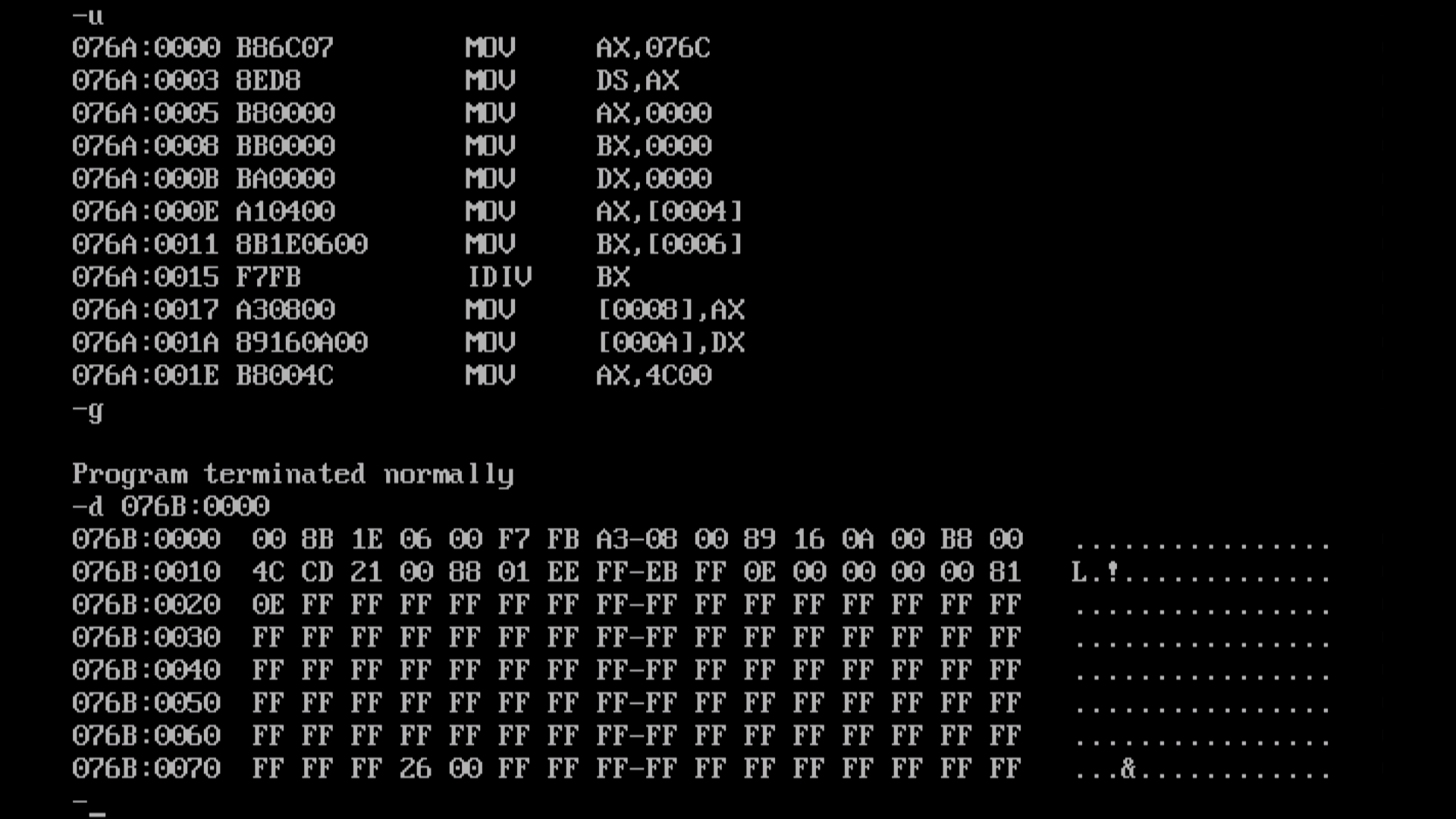
**mov d,dx**

**mov ax, 4C00H**

**int 21h**

**end**

**Output:-**



7-> Program for data transfer using different addressing modes.

**Code:-**

**model small**

**.8086**

**.data**

**s1 db "UVWXYZ$"**

**s2 db "ABCDEF$"**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov ah,15h**

**mov bh,al**

**mov di,offset s1**

**mov bl,[di]**

**mov ax,[0012h]**

**mov si, offset s2**

**mov dl,[si+2]**

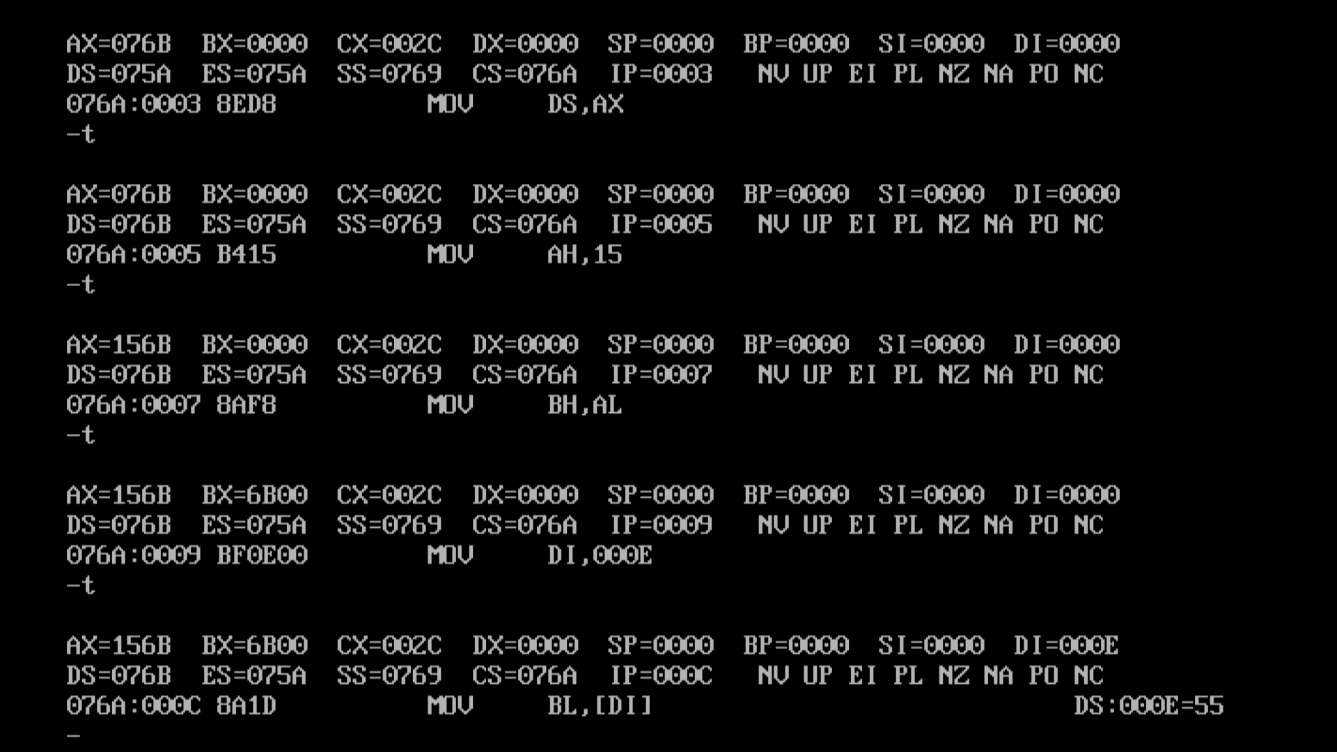
**in ax , 50H**

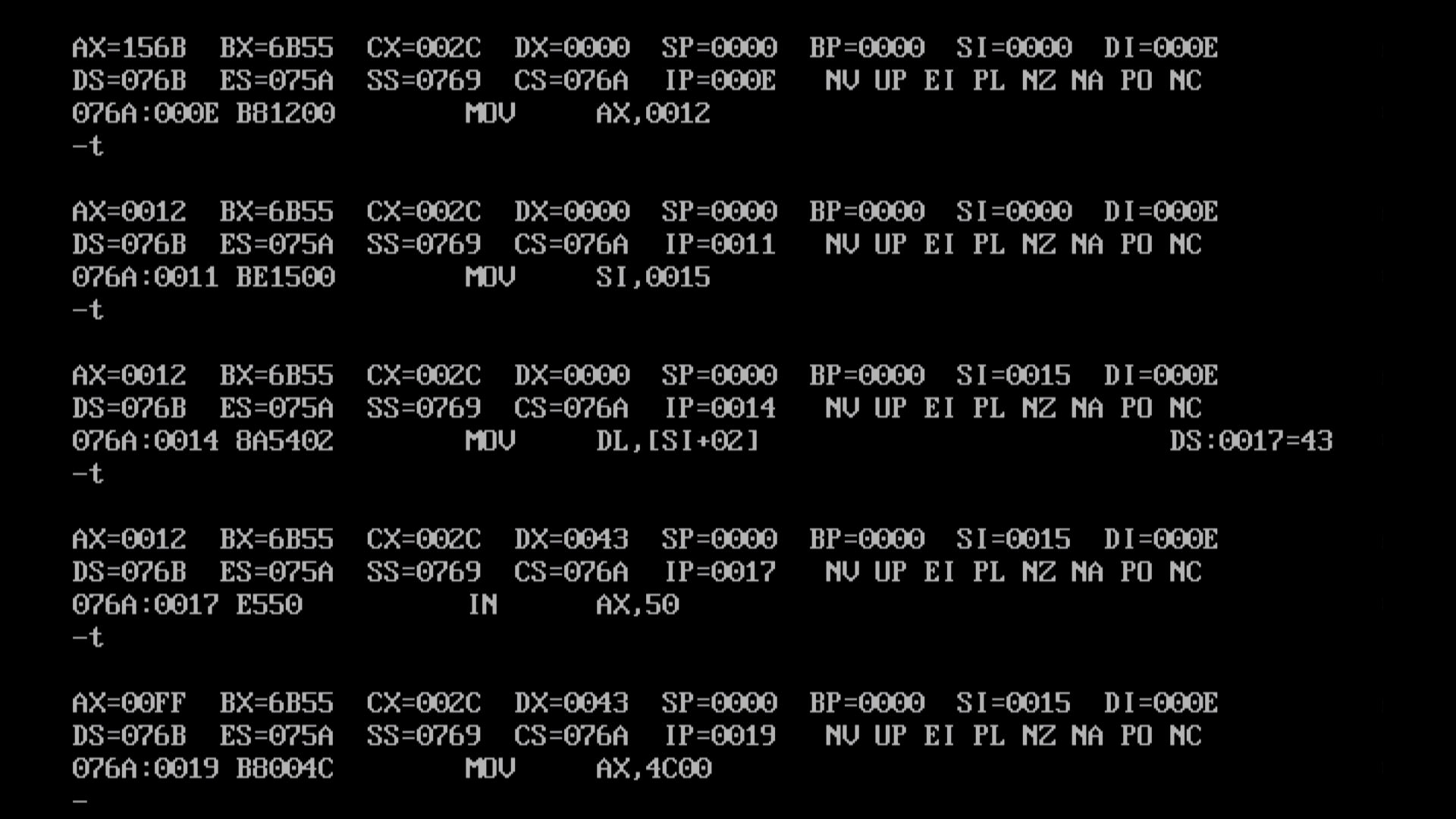
**mov ax, 4C00H**

**int 21h**

**end**

**Output :-**

****

****

8-> Program to move data from source to destination using indirect addressing mode (Block Move without overlap).

**Code:-**

**model small**

**.8086**

**.data**

**s1 db "UVWXYZ$"**

**s2 db 6 dup(0)**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov si,offset s1**

**mov di,offset s2**

**mov cx,0006h**

**up: mov al,[si]**

**mov [di],al**

**inc si**

**inc di**

**dec cx**

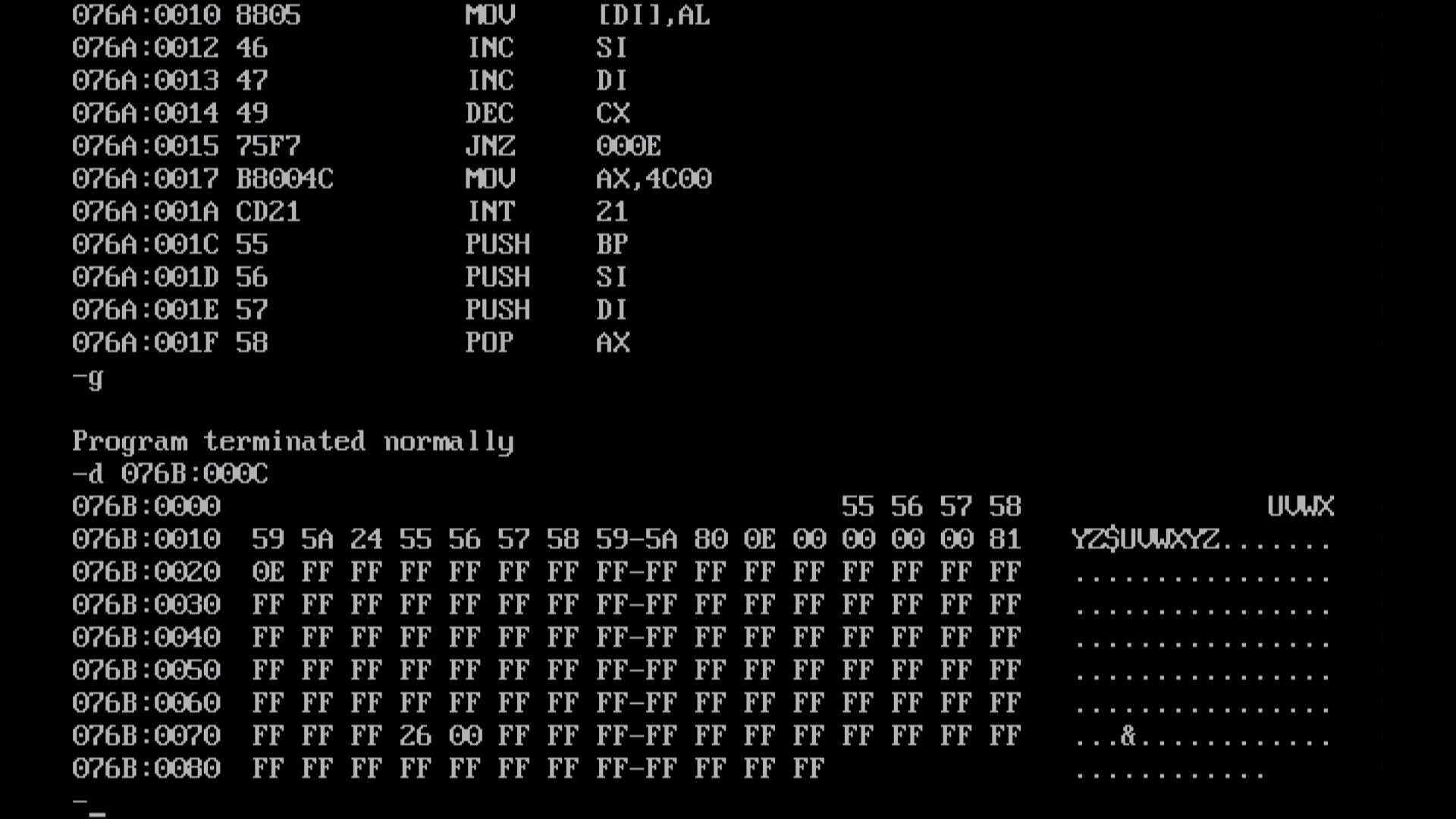
**jnz up**

**mov ax, 4C00H**

**int 21h**

**end**

**Output :-**



9-> Program to move a block of data from source to destination (With overlap in either direction.

**Code:-**

**model small**

**.8086**

**.data**

**s1 db "UVWXYZ$"**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov si,offset s1**

**mov cx,0006h**

**mov bl,05h**

**mov di,offset [s1+3]**

**up: inc si**

**inc di**

**dec bl**

**jnz up**

**go: mov al,[si]**

**mov [di],al**

**dec si**

**dec di**

**dec cx**

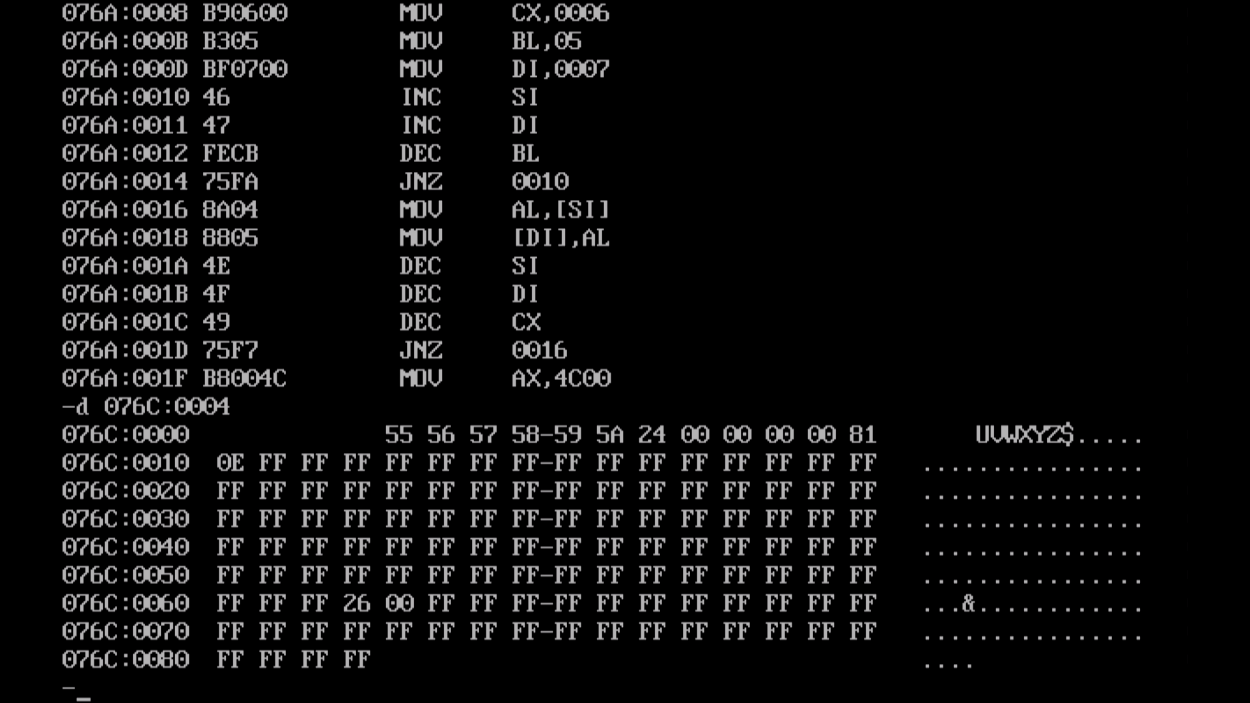
**jnz go**

**mov ax, 4C00H**

**int 21h**

**end**

**Before :-**

****

**After :-**

****

10-> Program to interchange two blocks of data.

**Code:-**

**model small**

**.8086**

**.data**

**s1 db "UVWXYZ$"**

**s2 db "123456$"**

**.code**

**mov ax, @data**

**mov ds, ax**

**mov si,offset s1**

**mov di,offset s2**

**mov cx,0006h**

**up: mov al,[si]**

**mov bl,[di]**

**mov [si],bl**

**mov [di],al**

**inc si**

**inc di**

**dec cx**

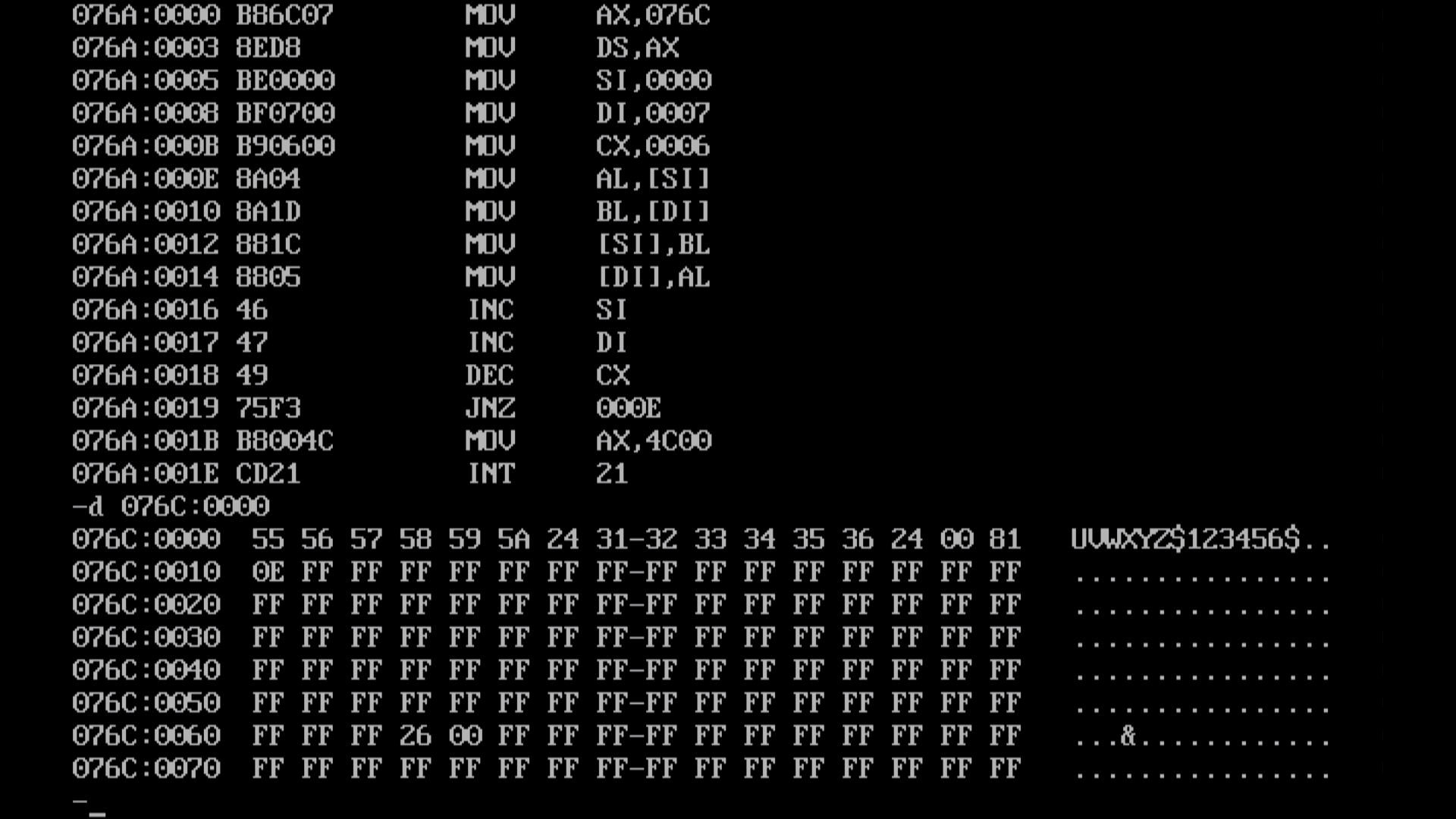
**jnz up**

**mov ax, 4C00H**

**int 21h**

**end**

**Before:-**

****

**After:-**

