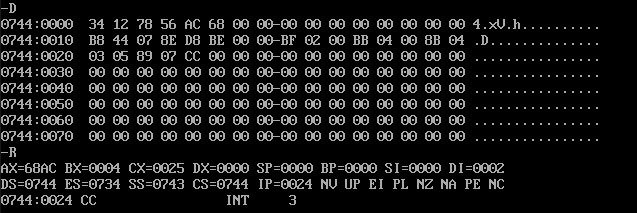
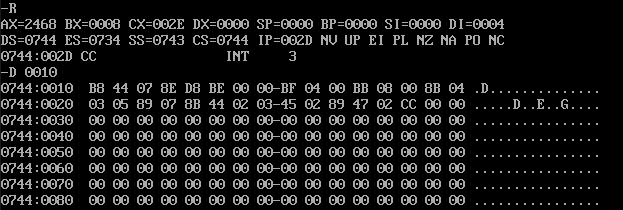
**Microprocessor Fundamentals and Programming**

**Assignment 2 CE092 : Nevil Parmar**

1. **Add 2 16-bit numbers. The 16-bit numbers are stored into the data segment.**

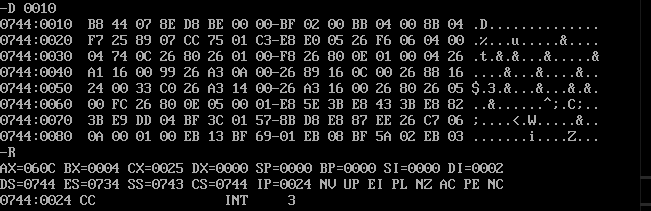
* **Code:**
* data segment
* n1 dw 1234h
* n2 dw 5678h
* result dw ?
* data ends
* code segment
* assume cs:code,ds:data
* mov ax,data
* mov ds,ax
* lea si,n1
* lea di,n2
* lea bx,result
* mov ax,[si]
* add ax,[di]
* mov [bx],ax
* int 03
* code ends
* end
* **Output:**
* ****

1. **Add 2 32-bit numbers stored in the data segment.**

* **Code:**
* data segment
* n1 dd 12345678h
* n2 dd 12342345h
* result dd ?
* data ends
* code segment
* assume cs:code,ds:data
* mov ax,data
* mov ds,ax
* lea si,n1
* lea di,n2
* lea bx,result
* mov ax,[si]
* add ax,[di]
* mov [bx],ax
* mov ax,[si+2]
* add ax,[di+2]
* mov [bx+2],ax
* int 03
* code ends
* end
* **Output:**
* ****

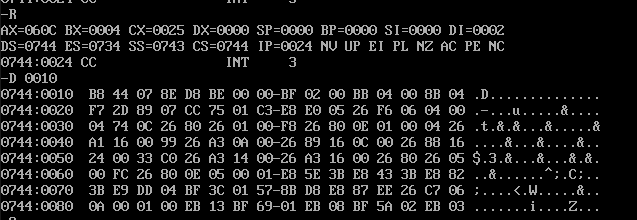
1. **Program to multiply two unsigned 16-bit numbers.**

* **Code:**
* data segment
* n1 dw 12h
* n2 dw 56h
* result dd ?
* data ends
* code segment
* assume cs:code,ds:data
* mov ax,data
* mov ds,ax
* lea si,n1
* lea di,n2
* lea bx,result
* mov ax,[si]
* mul [di]
* mov [bx],ax
* int 03
* code ends
* end
* **Output:**

****

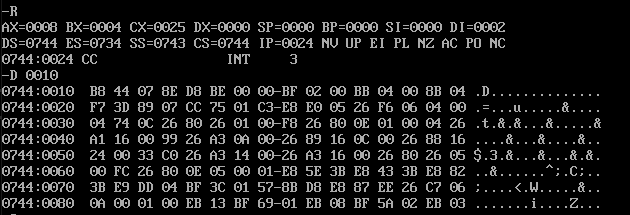
1. **Program to multiply signed 16-bit numbers.**

* **Code:**
* data segment
* n1 dw -12h
* n2 dw -56h
* result dd ?
* data ends
* code segment
* assume cs:code,ds:data
* mov ax,data
* mov ds,ax
* lea si,n1
* lea di,n2
* lea bx,result
* mov ax,[si]
* imul [di]
* mov [bx],ax
* int 03
* code ends
* end
* **Output:**

****

1. **Program to divide a 16-bit unsigned/signed number by 16-bit number. The numbers are stored into the data segment.**

* **Code:**
* data segment
* n1 dw 20h
* n2 dw 4h
* result dw ?
* data ends
* code segment
* assume cs:code,ds:data
* mov ax,data
* mov ds,ax
* lea si,n1
* lea di,n2
* lea bx,result
* mov ax,[si]
* idiv [di]
* mov [bx],ax
* int 03
* code ends
* end
* **Output:**

****

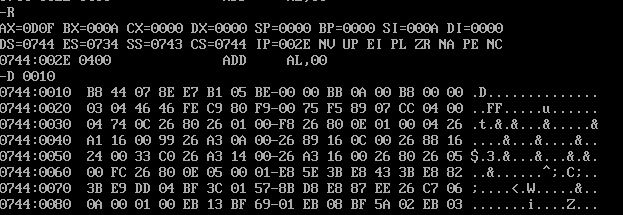
1. **Program to copy an array of bytes/words from the variable “SOURCE” to variable “DEST” which are defined in data segment.**

* **Code:**
* data segment
* arr1 db 1,2,3,4,5
* arr2 db 0,0,0,0
* data ends
* code segment
* assume cs:code,ds:data
* MOV AX,DATA
* MOV DS,AX
* MOV CL,5
* LEA BX,arr1
* LEA SI,arr2
* L:  MOV CH,[BX]
* MOV [SI],CH
* INC BX
* INC SI
* DEC CL
* CMP CL,00
* JNZ L
* INT 3
* code ends
* end
* **Output:**

****

1. **To sum an array of numbers stored in the data segment.**

* **Code:**
* data segment
* arr1 dw 0201h,0202h,0203h,0404h,0305h
* result dw ?
* data ends
* code segment
* assume cs:code,ds:data
* MOV AX,DATA
* MOV DS,AX
* MOV CL,5
* LEA SI,arr1
* LEA BX,result
* MOV AX,0000h
* L:  ADD AX,[SI]
* INC SI
* INC SI
* DEC CL
* CMP CL,00
* JNZ L
* MOV [BX],AX
* INT 3
* code ends
* end
* **Output:**

****

1. **Program to separate even and odd numbers from an array of words.**

* **Code:**
* data segment
* arr db 1,2,3,4
* oddarr db 10 dup(?)
* evenarr db 10 dup(?)
* data ends
* code segment assume cs:code,ds:data
* mov ax,data
* mov ds,ax
* lea si,oddarr
* lea di,evenarr
* lea bx,arr
* mov cl,4
* mov dh,2
* find:
* mov ah,0000h
* mov al,[bx]
* mov dl,al
* div dh
* cmp ah,00
* je even
* mov [si],dl
* inc si
* inc bx
* dec cl
* cmp cl,00
* jnz find
* even:
* mov [di],dl
* inc di
* dec cl
* cmp cl,00
* jnz find
* int 03
* code ends
* end
* **Output:**

