Fast**FAST- National University of Computer & Emerging Sciences, Karachi.  
Department of Computer Science**

**Quiz No, II Fall 2018**

**November 3, 2018**

1. Encode/Decode the following Instructions: MOV BX, [SI+8], SUB AX, BX, 89 7C FE, Push X1

**10001011 01 011 100 = 1B5C**

**0010 1001 11 011 000 = 29C8**

**Mov (SI+EF), DI**

**FF202AH**

1. Using string primitives Copy 100 doublewords from source to target using REP.

**.data**

**Val1 DWORD 100 DUP(?)**

**Val2 DWORD 100 DUP(?)**

**.code**

**cld**

**mov ecx,LENGTHOF Val1**

**mov esi,OFFSET Val1**

**mov edi,OFFSET Val2**

**rep movsd**

1. Use MOVSD to delete the first element of the following doubleword array. All subsequent array values must be moved one position forward toward the beginning of the array:

**array DWORD 10,10,20,30,40,50,60,70,80,90,100**

**.code**

**cld**

**mov ecx,(LENGTHOF array) - 1**

**mov esi, OFFSET array+4**

**mov edi, OFFSET array**

**rep movsd**

1. This program compares two strings (source and destination). It displays a message indicating whether the lexical value of the source string is EQUAL than the destination string.

**.data**

**source BYTE "FAST "**

**destination BYTE "FURIOUS"**

**message1 BYTE “Source is equal to destination”,0**

**message2 BYTE “Source is not equal to destination”,0**

**.code**

**Main proc**

**cld**

**mov esi, OFFSET source**

**mov edi, OFFSET destination**

**mov ecx, LENGTHOF source**

**repe cmpsb**

**jz \_equal**

**mov edx,OFFSET message2**

**jmp quit**

**\_equal:**

**mov edx,OFFSET message1**

**quit:**

**call WriteString**

**exit**

**main ENDP**

**END main**

1. Discuss the steps taken by the CPU when the INT instruction is invoked by a program.

* **Step1:** The operand of the INT instruction is multiplied by 4 to locate the matching interrupt vector table entry.
* **Step2:** The CPU pushes the flags and a 32-bit segment/offset return address on the stack, disables hardware interrupts, and executes a far call to the address stored at location (10h \* 4) in the interrupt vector table (F000:F065).
* **Step3**: The interrupt handler at F000:F065 executes until it reaches an IRET (interrupt return) instruction**.**
* **Step 4**: The IRET instruction pops the flags and the return address off the stack, causing the processor to resume execution immediately following the INT 10h instruction in the calling program.

1. What is an interrupt. Discuss some frequently used interrupts.

Interrupt is a temporary method creating a temporary halt during program execution and allows peripheral devices to access the microprocessor.

Some of the common interrupts are:

* + ***INT 10h Video Services:*** Procedures that display routines that control the cursor position, write text in color, scroll the screen, and display video graphics.
  + ***INT 16h Keyboard Services:*** Procedures that read the keyboard and check its status.
  + ***INT 17h Printer Services:*** Procedures that initialize, print, and return the printer status.
  + ***INT 21h MS-DOS Services***: Procedures that provide input-output, file handling, and memory management.
  + Also known as MS-DOS function calls.

1. Write string to file or device.

**.data**

**message "HELLO WORLD",0**

**.code**

**mov ah,40h**

**mov bx,1**

**mov cx, LENGTHOF message**

**mov dx, OFFSET message**

**int 21h**

1. Read single character from standard input

**.data**

**character BYTE ?**

**.code**

**mov ah, 01h**

**int 21h**

**mov character, al**

1. Using String primitives, multiply each element of a doubleword array by a constant value.

**.data**

**array DWORD 11,12,13,14,15,16,17,18,19,20**

**constant DWORD 2**

**.code**

**cld**

**mov esi,OFFSET array**

**mov edi,esi**

**mov ecx,LENGTHOF array**

**L1: lodsd**

**mul constant**

**stosd**

**loop L1**

1. Consider:

Int myarray[20], productaray[20].

Call MultiplyPro

Pass the elements of array to a function, the function finds N X M products where N is greater than M

**INCLUDE Irvine32.inc**

**.data**

**myArray DWORD 3,2,7,5,6,2,7,6,8,6,4,9,0,6,11,19,14,12,1,3**

**productArray DWORD 20 DUP(?)**

**value DWORD ?**

**v DWORD ?**

**.code**

**main PROC**

**mov eax ,0**

**mov edi , OFFSET myArray**

**mov esi ,edi**

**mov ebx , OFFSET productArray**

**add esi ,4**

**call MultiplyPro**

**exit**

**main ENDP**

**MultiplyPro PROC**

**mov ecx , 20**

**L1:**

**mov v , ecx**

**mov eax ,DWORD PTR [edi]**

**mov ecx , 19**

**L2:**

**mov edx , DWORD PTR [esi]**

**cmp eax , edx**

**jb q**

**mov value , eax**

**mul edx**

**mov DWORD PTR [ebx],eax**

**call WriteDec**

**mov eax,value**

**jmp p**

**q:**

**mov DWORD PTR [ebx],0**

**p: add ebx , 4**

**add esi , 4**

**Loop L2**

**mov ecx,v**

**add edi , 4**

**Loop L1**

**ret 8**

**MultiplyPRO ENDP**

**END main**