CSC 3210

Spring 22

TTM

Set A

Solution

1. Consider the following code. What is the content of AX and BX register after executing the code. Explain your answer. (2 points for correct answer, 5 points for explanation)

.data

Array WORD 1122h, 3344h, 5566h

.code

Mov esi, 0

Mov edi, OFFSET Array+2

Mov ax, [edi]

Add esi, 4

Mov bx, Array[esi]

Answer:

First store the array items in Little Endian order. Below each box contains 8-bit or 1 byte data.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 22 | 11 | 44 | 33 | 66 | 55 |  |  |

Esi contains 0. At the last line of the code, esi is used as array index

Mov edi, OFFSET Array+2

Array23 makes a 2-byte jump from the beginning of the array and reach 44.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 22 | 11 | 44 | 33 | 66 | 55 |  |  |

Offset operator returns the address of 44 and then mov operation stores that in edi register.

Mov ax, [edi]

Here you are dereferencing edi register using [edi] and you want to store the value in AX register. Because your destination is 16-bit (2 bytes) long, [edi] will extract 2 bytes from 44.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 22 | 11 | 44 | 33 | 66 | 55 |  |  |

Because computer is little endian, it will read the data in reverse order : 33 44

Ax contains 3344

Add esi, 4

Nowe esi contains 4.

Mov bx, [Array+esi]

Array+esi is actuall Array+4 as Esi contains 4. Array+4 will do a 4-byte jump from the beginning of the array and take you to the 44.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 22 | 11 | 44 | 33 | 66 | 55 |  |  |

As the destination in mov operation is bx (2-byte long), array+4 extracts 2-bytes from 66.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 22 | 11 | 44 | 33 | 66 | 55 |  |  |

Because computer is little endian, it will read the data in reverse order: 55 66

bx contains 5566

1. Translate the following code into assembly code(masm). You can use Mov,Loop, Add, Sub instructions to complete the translation (8 points).

int var1 = 0

int var2 = 10;

for (i = 7; i > 0; i--){

for (j=10; j > 0; j--){

var1 = var1 + var2

}

}

Solution:

You can store the data as either byte, word or dword

.data

var1 DWORD 0

var2 DWORD 10

var3 DWORD 15

.code

mov ecx, 7 ; initialize outer loop counter

L1:

mov edx, ecx ; keep a back up of ecx

mov ecx, 10 ; initialize inner loop counter

L2:

mov eax, var2

add var1, eax

Loop L2

mov ecx, edx ; restore outer loop counter

Loop L1