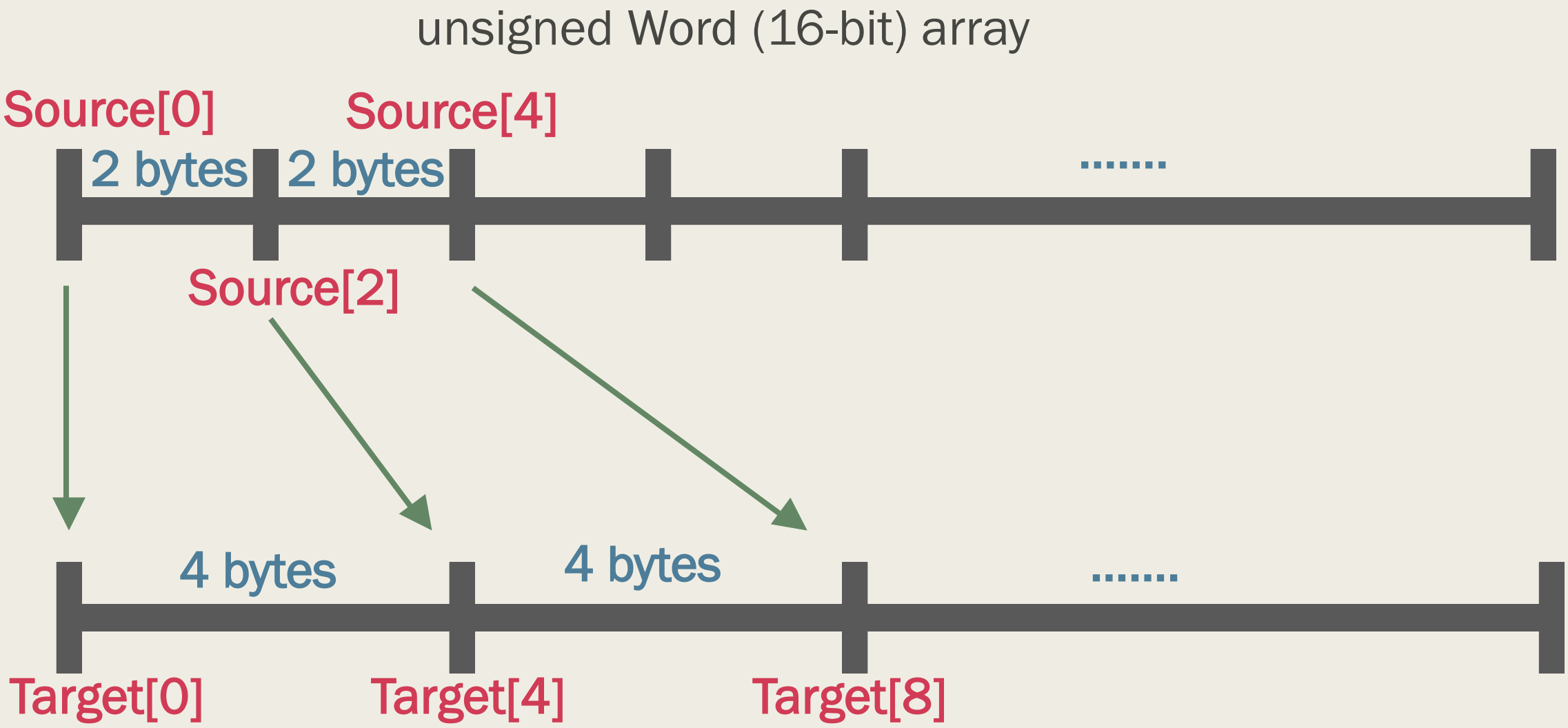
Assembly Language

CH04 - Exercise & Homework

Exercise 4.

Copying a Word Array to a DoubleWord array

Write a program that uses a loop to copy all the elements from an unsigned Word (16-bit) array into an unsigned doubleword (32-bit) array.


unsigned Word (16-bit) array
Source[0]
Source[4]
2 bytes
2 bytes
.....
Source[2]
4 bytes
4 bytes
.....
Target[0]
Target[4]
Target[8]
unsigned doubleword (32-bit) array

```
sourceNum = 0;
targetNum = 0;

for( i = LENGTHOF(sourceArray);  i>0 ; i- - )
{
        targetArray[targetNum ] = sourceArray[sourceNum ];
        sourceNum++;
        targetNum++;
}
```

```
.386
.model flat,stdcall
.stack 4096
ExitProcess proto,dwExitCode:dword
INCLUDE Irvine32.inc

; textbook Page.95
```

```
.data
;設一個unsigned Word (16-bit) array
source word 1,2,3,4,5,6,7,8,9,10
;value name type value

;source的長度
count = LENGTHOF source

; unsigned doubleword (32-bit) array
target dword count DUP(?)
; value name type number of value uninitialized
```

```
.data
;設一個unsigned Word (16-bit) array
source word 1,2,3,4,5,6,7,8,9,10
;value name type value

;source的長度
count = LENGTHOF source

; unsigned doubleword (32-bit) array
target dword count DUP(?)
; value name type number of value uninitialized
```

```
.code
main proc
        mov ecx, count        ;set loop count
        mov esi,0             ; source, esi = 0(offset of byteVal)
        mov edi,0             ; target, edi = 0(offset of byteVal)
L1:                           ; label
      movzx  eax, source[esi]  ; eax = source[esi]
      mov  target[edi], eax ; target[edi] = eax = source[esi]
      add  esi, 2         ; 1 word => 2 bytes, esi += 2
      add  edi, 4         ; 1 doubleword => 4 bytes, esi += 4
      loop L1                   ; count = count – 1
```

```
        invoke ExitProcess,0        ; exit process
main endp                           ; end of main procedure

end main                            ; end of the program
```

Homework Assignment 01

due at 10/14 23:59

Fibonacci number

submission format

Write a program that uses a loop to calculate the first seven values of the Fibonacci number sequence, describe by the following formula:

$$Fib(1) = 1, Fib(2) = 1, Fib(n) = Fib(n - 1) + Fib(n - 2)$$

*Place each values in the EAX register
(Or try to display them. See next page)

How to display computing result on screen?

- With Irvine's library
  - **WriteString** procedure

```
WriteString PROC

  Writes a null-terminated string to standard output.

Call args:  EDX = points to string

Return arg: None

Example:

.data
prompt BYTE "Enter your name: ",0

.code
       mov  edx,OFFSET prompt
       call WriteString
```

  - ***WriteInt*** *procedure*

```
WriteInt PROC

  Writes a signed 32-bit decimal number to standard output in decimal format
with a leading sign and no leading zeros.

Call args:  EAX = signed number to write

Return arg: None

Example:

      mov  eax,216543
      call WriteInt

Output: +216543

NOTES: To write an unsigned integer, use the WriteDec procedure.
       To write in hexadecimal, use the WriteHex procedure.
       To write in binary, use the WriteBin procedure.
       Use the mShow macro to display the contents of an 8-, 16-,
or 32-bit variable or register in any combination of hexadecimal,
signed decimal, unsigned decimal, or binary formats.
```

Submission Format

- Turn in your report **in group**
- Observe the changes of registers and memory while executing. **Screenshot when needed.**
- Pack (archive) the following two files
  - *xxxx.asm*
  - *Put your screenshots, your source code with comments, and your feedback about the assignment in the file. (file format: .doc/.docx/.pdf)*
- Upload to Tronclass