Control Structures: Examples and Sample Problems

ICS312
Machine-Level and Systems Programming

Henri Casanova (henric@hawaii.edu)
Signed Integers: SF and OF???

- Example: \( a = 80h \) (-128d), \( b = 23h \) (+35d) \( a < b \)
  - \( a - b = a + (-b) = 80h + DDh = 15Dh \)
  - dropping the 1, we get 5Dh (+93d), which is erroneously positive!
  - So, SF=0 and OF=1

- Example: \( a = F3h \) (-13d), \( b = 23h \) (+35d) \( a < b \)
  - \( a - b = a + (-b) = F3h + DDh = D0h \) (-48d)
  - D0h is negative and we have no overflow (in range)
  - So, SF=1 and OF=0

- Example: \( a = F3h \) (-13d), \( b = 82h \) (-126d) \( a > b \)
  - \( a - b = a + (-b) = F3h + 7Eh = 171h \)
  - dropping the 1, we get 71h (+113d), which is positive and we have no overflow
  - So, SF=0 and OF=0

- Example: \( a = 70h \) (112d), \( b = D8h \) (-40d) \( a > b \)
  - \( a - b = a + (-b) = 70h + 28h = 98h \), which is erroneously negative
  - So, SF=1 and OF=1
Mystery Code

What does this code print? (all signed)

```assembly
mov     ebx, 12
mov     eax, 1
cmp     ebx, 10
jle     end_label
dec     eax
mov     eax, ebx
jz      end_label
add     eax, 3
end_label:  call    print_int
```
Mystery Code

What does this code print? (all signed)

```assembly
mov    ebx, 12
mov    eax, 1
cmp    ebx, 10
jle    end      ; doesn’t branch
dec    eax       ; eax = 0, ZF = 0
mov    eax, ebx ; eax = 12
jz     end       ; branches
add    eax, 3

end:      call    print_int ; prints 12
```
Computing the Sum of an Array

- Let’s write a (fragment of a) program that computes the sum of an array
- Let us assume that the array is “declared” in the `.bss` segment as:
  - `array    resd 20 ; An array of 20 double words`
- And let us assume that its elements have been set to some values
- We want to compute the numerical sum of all its elements into register ebx
Computing the Sum of an Array

mov ebx, 0 ; ebx = 0 (sum)
mov ecx, 0 ; ecx = 0

(main loop)

; Compute address of current element
mov eax, array ; eax points to 1st element
mov edx, ecx ; edx = ecx

(loop index)
imul edx, 4 ; edx = 4 * ecx
add eax, edx ; eax = array + 4 * ecx

; Increment the sum
add ebx, [eax] ; sum += element

; Move to the next element
Computing the Sum of an Array

; SHORTER/SIMPLER VERSION

mov ebx, 0 ; ebx = 0 (sum)
mov ecx, 0 ; ecx = 0 (loop index)
mov eax, array ; eax = array

main_loop:
; Increment the sum
add ebx, [eax] ; sum += element
; Move to the next element
add eax, 4 ; eax += 4
inc ecx ; ecx ++
; Done?
cmp ecx, 20 ; compare ecx to