Subprograms: Examples and Sample Problems

ICS312
Machine-Level and
Systems Programming

Henri Casanova (henric@hawaii.edu)



Assuming that ESP=00001000h

00001000h

increasing addresses



Assuming that ESP=00001000h

push dword 1; ESP = 00000FFCh

↑ 00001000h 00000FFFh 00000FFEh 00000FFDh 00000FFCh

increasing addresses



Assuming that ESP=00001000h

push dword 1; ESP = 00000FFCh

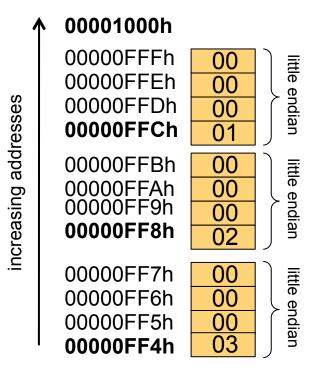
push dword 2 ; ESP = 00000FF8h

\$\ \bigcirc 00001000h \\ 00000FFFh \\ 00000FFDh \\ 00000FFCh \\ 00000FFAh \\ 00000FFAh \\ 00000FF8h \\ 000000FF8h \\ 00000FF8h \\ 00000FFRAH \\ 0000



Assuming that ESP=00001000h

push dword 1 ; ESP = 00000FFChpush dword 2 ; ESP = 00000FF8hpush dword 3 ; ESP = 00000FF4h





Assuming that ESP=00001000h

push dword 1 ; ESP = 00000FFChpush dword 2 ; ESP = 00000FF8h

push dword 3; ESP = 00000FF4h

pop eax ; EAX = 3pop ebx ; EBX = 2pop ecx ; ECX = 1

00001000h 00000FFFh 00 little endian 00000FFEh 00 addresses 00000FFDh 00 00000FFCh 00000FFBh 00 00 00000FFAh increasing 00000FF9h 00000FF8h 00000FF7h little endian 00000FF6h 00 00000FF5h 00000FF4h

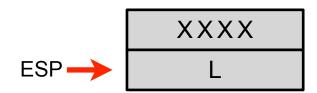
M

```
dd 42, 43, 44, 45, 56
L
         dword L
push
call
         func
add
       esp, 4
call
       print int
. . .
func:
      push
               ebp
               ebp, esp
      mov
      push
              [ebp+8]
      push
      call
               reference
      add
               esp, 8
      add
               eax, 10
               ebp
      pop
      ret
reference:
      push
               ebp
                ebp, esp
      mov
               eax, [ebp+12]
      mov
      add
                eax, [ebp+8]
                eax, [eax]
      mov
                ebp
      pop
      ret
```

```
ESP XXXX
```

M

```
dd 42, 43, 44, 45, 56
L
         dword L
push
call
         func
add
       esp, 4
call
       print int
. . .
func:
      push
               ebp
               ebp, esp
      mov
      push
              [ebp+8]
      push
      call
                reference
      add
               esp, 8
      add
               eax, 10
               ebp
      pop
      ret
reference:
      push
               ebp
                ebp, esp
      mov
               eax, [ebp+12]
      mov
                eax, [ebp+8]
      add
                eax, [eax]
      mov
                ebp
      pop
      ret
```



```
dd 42, 43, 44, 45, 56
L
                                                                  XXXX
        dword L
push
call
        func
                                                  ESP -
                                                                 ret @
        esp, 4
add
       print int
call
. . .
func:
     push
               ebp
               ebp, esp
     mov
     push
              [ebp+8]
     push
      call
               reference
      add
               esp, 8
      add
               eax, 10
               ebp
     pop
      ret
reference:
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
               eax, [ebp+8]
      add
               eax, [eax]
     mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                  XXXX
        dword L
push
call
        func
                                                                  ret @
add
        esp, 4
call
       print int
                                                              saved ebp
. . .
func:
               ebp
     push
               ebp, esp
     mov
     push
               [ebp+8]
     push
      call
               reference
      add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
      mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                  XXXX
        dword L
push
call
        func
                                                                 ret @
add
        esp, 4
call
       print int
                                                              saved ebp
. . .
func:
     push
               ebp
               ebp, esp
     mov
     push
              [ebp+8]
     push
      call
               reference
      add
               esp, 8
      add
               eax, 10
               ebp
     pop
      ret
reference:
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
      mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                 XXXX
        dword L
push
call
        func
                                                                 ret @
add
        esp, 4
call
       print int
                                                             saved ebp
                                                  EBP -
. . .
func:
                                                  ESP-
     push
               ebp
               ebp, esp
     mov
     push
              [ebp+8]
     push
     call
               reference
     add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                 XXXX
        dword L
push
call
        func
                                                                 ret @
add
        esp, 4
call
       print int
                                                              saved ebp
                                                  EBP —
. . .
func:
     push
               ebp
                                                  ESP-
               ebp, esp
     mov
     push
              [ebp+8]
     push
     call
               reference
     add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
      ret
```

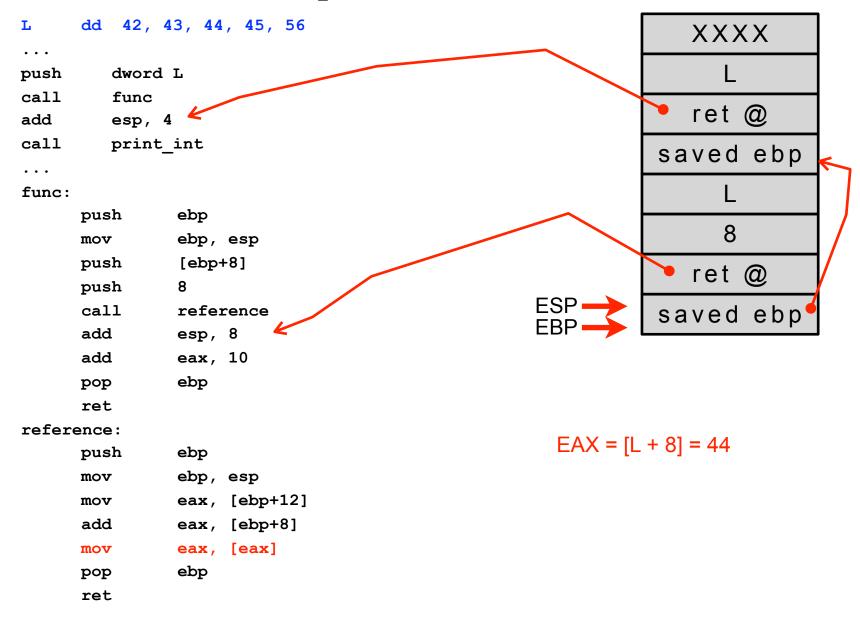
```
dd 42, 43, 44, 45, 56
L
                                                                 XXXX
        dword L
push
call
        func
                                                                 ret @
add
        esp, 4
call
       print int
                                                              saved ebp
                                                  EBP -
. . .
func:
     push
               ebp
                                                                    8
               ebp, esp
     mov
     push
               [ebp+8]
                                                                 ret @
                                                  ESP -
     push
     call
               reference
               esp, 8
     add
               eax, 10
      add
               ebp
     pop
      ret
reference:
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                XXXX
        dword L
push
call
        func
                                                                ret @
add
        esp, 4
call
       print int
                                                             saved ebp
                                                 EBP -
. . .
func:
     push
               ebp
                                                                   8
               ebp, esp
     mov
     push
               [ebp+8]
                                                                ret @
     push
                                                             saved ebp
     call
               reference
     add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                 XXXX
        dword L
push
call
        func
                                                                ret @
add
        esp, 4
call
       print int
                                                             saved ebp
. . .
func:
     push
               ebp
                                                                    8
               ebp, esp
     mov
     push
               [ebp+8]
                                                                ret @
     push
                                                             saved ebp
     call
               reference
     add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                XXXX
        dword L
push
call
        func
                                                                ret @
add
        esp, 4
call
       print int
                                                             saved ebp
. . .
func:
     push
               ebp
                                                                   8
               ebp, esp
     mov
     push
               [ebp+8]
                                                                ret @
     push
                                                             saved ebp
     call
               reference
     add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
                                                    EAX = L
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                 XXXX
        dword L
push
call
        func
                                                                ret @
add
        esp, 4
call
       print int
                                                             saved ebp
. . .
func:
     push
               ebp
                                                                    8
               ebp, esp
     mov
     push
               [ebp+8]
                                                                ret @
     push
                                                             saved ebp
     call
               reference
     add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
                                                    EAX = L + 8
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
               eax, [ebp+8]
      add
               eax, [eax]
     mov
               ebp
     pop
      ret
```



```
dd 42, 43, 44, 45, 56
L
                                                                XXXX
        dword L
push
call
        func
                                                                ret @
add
        esp, 4
call
       print int
                                                             saved ebp
                                                 EBP-
. . .
func:
     push
               ebp
                                                                   8
               ebp, esp
     mov
     push
              [ebp+8]
                                                                ret @
                                                 ESP -
     push
     call
               reference
     add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
                                                    EAX = 44
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
     ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                 XXXX
        dword L
push
call
        func
                                                                 ret @
        esp, 4
add
call
       print int
                                                              saved ebp
                                                  EBP-
. . .
func:
     push
               ebp
                                                  ESP-
               ebp, esp
     mov
     push
              [ebp+8]
     push
     call
               reference
     add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
                                                    EAX = 44
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
      ret
```

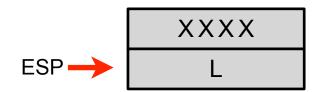
```
dd 42, 43, 44, 45, 56
L
                                                                 XXXX
        dword L
push
call
        func
                                                                 ret @
add
        esp, 4
                                                  EBP -
call
       print int
                                                              saved ebp
. . .
func:
     push
               ebp
               ebp, esp
     mov
     push
              [ebp+8]
     push
     call
               reference
     add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
                                                    EAX = 44
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                  XXXX
        dword L
push
call
         func
                                                                  ret @
add
        esp, 4
                                                  EBP-
call
       print int
                                                              saved ebp
. . .
func:
     push
               ebp
               ebp, esp
     mov
     push
               [ebp+8]
     push
      call
               reference
      add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
                                                     EAX = 44 + 10 = 54
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
      mov
               ebp
     pop
      ret
```

```
dd 42, 43, 44, 45, 56
L
                                                                  XXXX
        dword L
push
call
        func
                                                  ESP -
                                                                 ret @
        esp, 4
add
       print int
call
. . .
func:
     push
               ebp
               ebp, esp
     mov
     push
              [ebp+8]
     push
      call
               reference
      add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
                                                    EAX = 54
     push
               ebp
               ebp, esp
     mov
               eax, [ebp+12]
     mov
      add
               eax, [ebp+8]
               eax, [eax]
     mov
               ebp
     pop
      ret
```

M

```
dd 42, 43, 44, 45, 56
L
         dword L
push
call
         func
add
       esp, 4
call
       print int
. . .
func:
      push
                ebp
                ebp, esp
      mov
      push
              [ebp+8]
      push
      call
                reference
      add
                esp, 8
                eax, 10
      add
                ebp
      pop
      ret
reference:
      push
                ebp
                ebp, esp
      mov
                eax, [ebp+12]
      mov
      add
                eax, [ebp+8]
                eax, [eax]
      mov
                ebp
      pop
      ret
```



$$EAX = 54$$

```
dd 42, 43, 44, 45, 56
L
         dword L
push
call
         func
       esp, 4
add
call
       print int
. . .
func:
     push
               ebp
               ebp, esp
      mov
     push
              [ebp+8]
     push
      call
               reference
      add
               esp, 8
               eax, 10
      add
               ebp
     pop
      ret
reference:
     push
               ebp
               ebp, esp
      mov
               eax, [ebp+12]
      mov
      add
               eax, [ebp+8]
               eax, [eax]
      mov
               ebp
     pop
      ret
```

$$EAX = 54$$

M

```
dd 42, 43, 44, 45, 56
L
                                                                   XXXX
         dword L
push
call
         func
add
       esp, 4
call
       print int
. . .
func:
               ebp
      push
               ebp, esp
      mov
     push
              [ebp+8]
     push
      call
               reference
      add
               esp, 8
      add
               eax, 10
               ebp
      pop
      ret
reference:
                                                      prints "54"
      push
               ebp
               ebp, esp
      mov
               eax, [ebp+12]
      mov
      add
               eax, [ebp+8]
               eax, [eax]
      mov
               ebp
      pop
      ret
```



func:

ret

Practice

What things are wrong with the following program?

```
push
        ebx
        30
push
        func
call
add
        esp, 4
        print_int
call
call
        print_nl
push
        ebp
        ebp, esp
mov
        eax, [ebp+8]
mov
        eax, [ebp+4]
add
```



Practice (Solution)

What 5 things are wrong with the following program?

```
push
                  ebx
                  dword 30
         push
         call
                  func
                  esp, 8
         add
        call
                  print_int
                  print_nl
         call
func:
         push
                  ebp
                  ebp, esp
         mov
                  eax, [ebp+12]
         mov
        add
                  eax, [ebp+8]
                  ebp
         pop
        ret
```



Practice

What does the stack look like?

```
push
                   ebx
                   dword 30
         push
         call
                   func
                                                HERE?
         add
                   esp, 8
                   print_int
         call
                   print_nl
         call
func:
         push
                   ebp
                   ebp, esp
         mov
                                                HERE?
                   eax, [ebp+12]
         mov
                   eax, [ebp+8]
         add
                   ebp
         pop
         ret
```



Practice (Solution)

What does the stack look like?

push	ebx
push	dword 30
call	func
	<
add	esp, 8
call	print_int
call	print_nl

XXXXXX EBX 30

- -

ret

func: push ebp
mov ebp, esp
<----mov eax, [ebp+12]
add eax, [ebp+8]
pop ebp

EBX
30
Return @
EBP



Local Variables Example

Inside the body of the subprogram, parameters are referenced as:

□ [EBP+8]: 1st parameter

□ [EBP+12]: 2nd parameter

Inside the body of the subprogram, local variables are referenced as:

□ [EBP-4]: 1st local variable

□ [EBP-8]: 2nd local variable

□ [EBP-12]: 3rd local variable

EBP+12 2nd parameter
EBP+8 1st parameter
EBP+4 return address
EBP saved EBP
EBP-4 1st local var
EBP-8 2nd local var
EBP-12 3rd local var

Very important you have this picture in mind; you should be able to redraw it



Let's write the assembly code equivalent to the following C/Java function

```
int f(int num) { // computes Fibonacci numbers
  int x, sum;
  if (num == 0) return 0;
  if (num == 1) return 1;
  x = f(num-1);
  sum = x + f(num-2)
  return sum;
}
```

Let's write a "straight" translation, without optimizing variables away, just for demonstration purposes

Ŋ.

A Full Example (main program)

```
segment .data
                  db
                        "Enter n: ", 0
     msg1
     msg2
                  db
                        "The result is: ", 0
     ...; declaration of asm main and setup
                                 ; eax = address of msg1
              eax, msg1
     mov
     call
              print string
                                 ; print msg1
              read int
                                 ; get an integer from the keyboard (in EAX)
     call
     push
              eax
                                 ; put the integer on the stack (parameter #1)
     call
                                 ; call f
               ebx
                                 ; remove the parameter from the stack
     pop
              ebx, eax
                                 ; save the value returned by f
     mov
                                 ; eax = address of msg2
              eax, msg2
     mov
     call
              print string
                                 ; print msg2
              eax, ebx
                                 ; eax = sum
     mov
     call
              print int
                                 ; print the sum
                                 ; print a new line
     call
              print nl
```

...; clean up

%include "asm io.inc"

19

A Full Example (function f)

```
FUNCTION: f
    Takes one parameter: an integer
    eax = return value
segment .text
    enter 8,0
                  ; num in [ebp+8]
                  ; local var x in [ebp-4],
                  ; local var sum in [ebp-8]
    push ebx
                  ; save ebx
    push
          ecx
                  ; save ecx
    push edx
                  ; save edx
          eax, [ebp+8]; eax = num
    mov
          eax. 2 : eax -= 2
    sub
                  ; if not <0, goto next
         next
    ins
    add
         eax, 2
                           : eax += 2
          end
    jmp
next:
          eax, [ebp+8]; eax = num
    mov
                       : eax -= 1
    add
          eax. -1
```

```
push eax
                    ; put (num -1) on stack
    call f
                    ; call f (recursively)
                    ; remove (num-1) from stack
    add
          esp, 4
           [ebp-4], eax ; put the returned value in x
    mov
           eax, [ebp+8]; eax = num
    mov
          eax. -2
    add
                    : eax -= 2
    push eax
                    ; put (num -2) on stack
    call f
                    ; call f (recursively),
                    : the return value is in eax
    add
          esp, 4; remove (num-1) from stack
    add
          eax, [ebp-4]; eax += x
end:
                    ; restore ebx
          edx
    pop
    pop
          ecx
                    ; restore ecx
          ebx
                    ; restore edx
    pop
    leave
                    ; clean up the stack
    ret
                    ; return
```