Intro to Operating Systems

CS461 / ECE422 – UIUC SPRING 2016

By Gene Shiue

Outline

x86 ISA

Registers

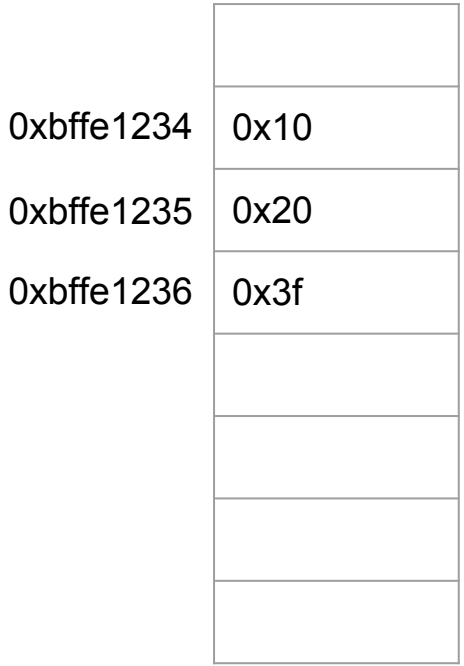
Assembly Instructions

Stack

Stack Frame

32-bit x86 ISA

- 1 byte = 8 bits
- char -> 1 byte
- integer -> 4 bytes
- word -> 2 bytes (in gdb, word -> 4 bytes)
- long -> 4 bytes
- Memory address -> 4 bytes
- Pointer -> ?
- Registers -> 4 bytes
- Each memory location -> 1 byte

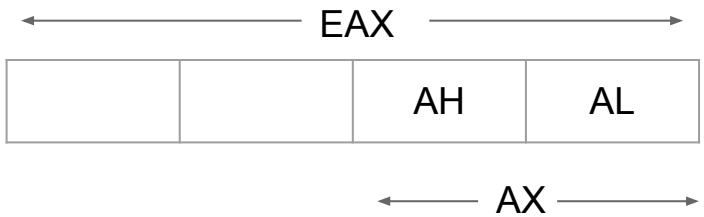


Registers

General Purpose: **EAX, EBX, ECX, EDX**, EDI, ESI

Special:

- EIP: Instruction pointer
- ESP: Stack pointer
- EBP: Base pointer

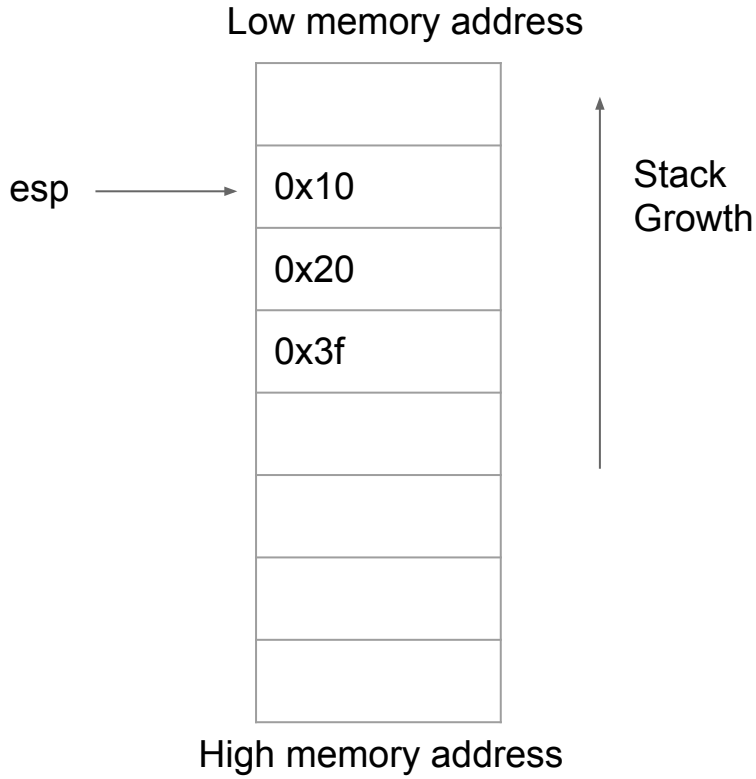


Assembly Instructions

push, pop, jmp, call, mov, lea, xor, cmp, dec, inc, int, leave, ret, and a lot more!

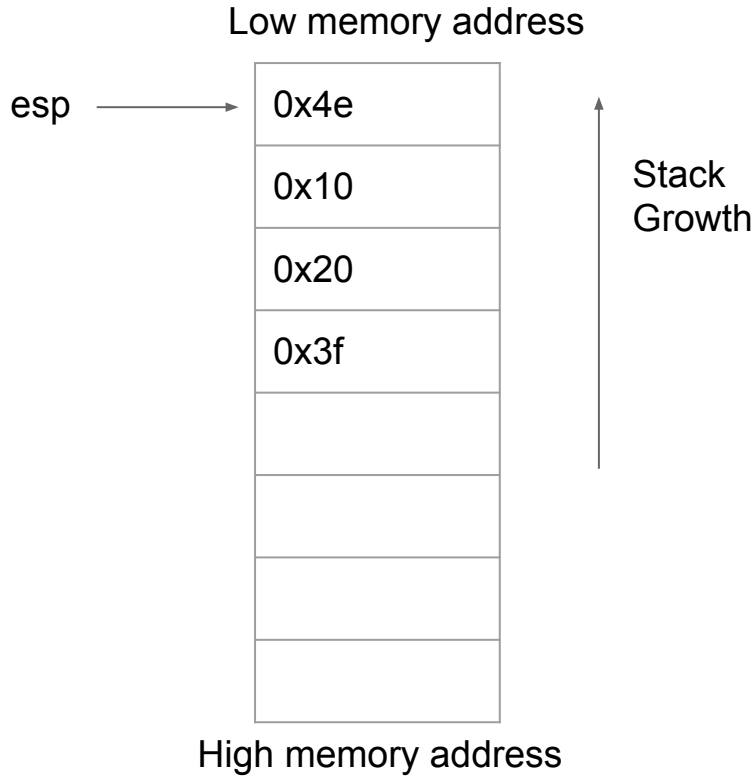
```
cmp    $0xffffff83,%eax
jne    <label>

call   foo

mov    0x8(%ebp),%eax

lea    -0x10(%ebp),%eax

xor    %ecx,%ecx
```

Stack


Low memory address
esp
0x10
0x20
0x3f
Stack
Growth
High memory address

Stack

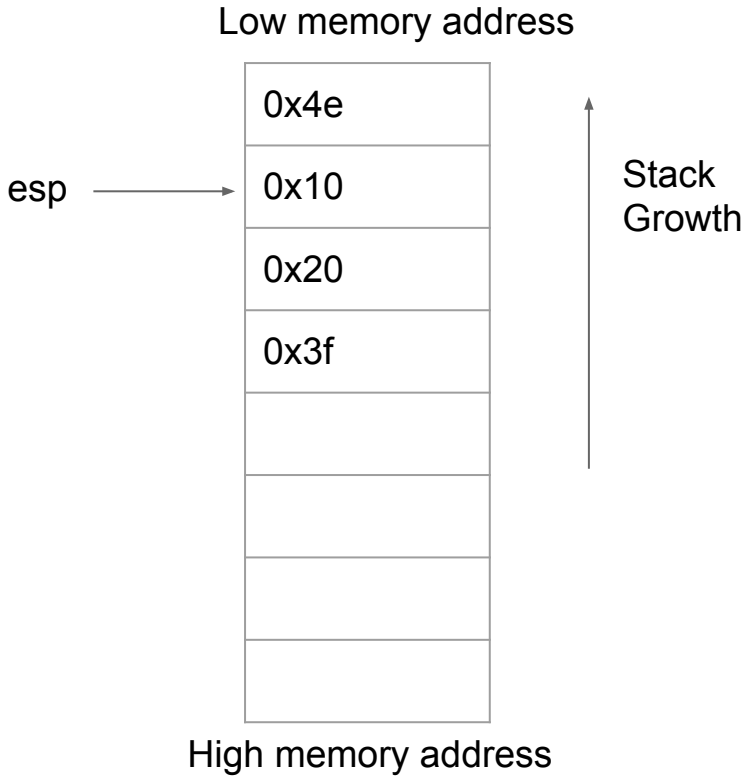
push $0x4e



Stack

push $0x4e

pop %eax (eax contains 0x4e)

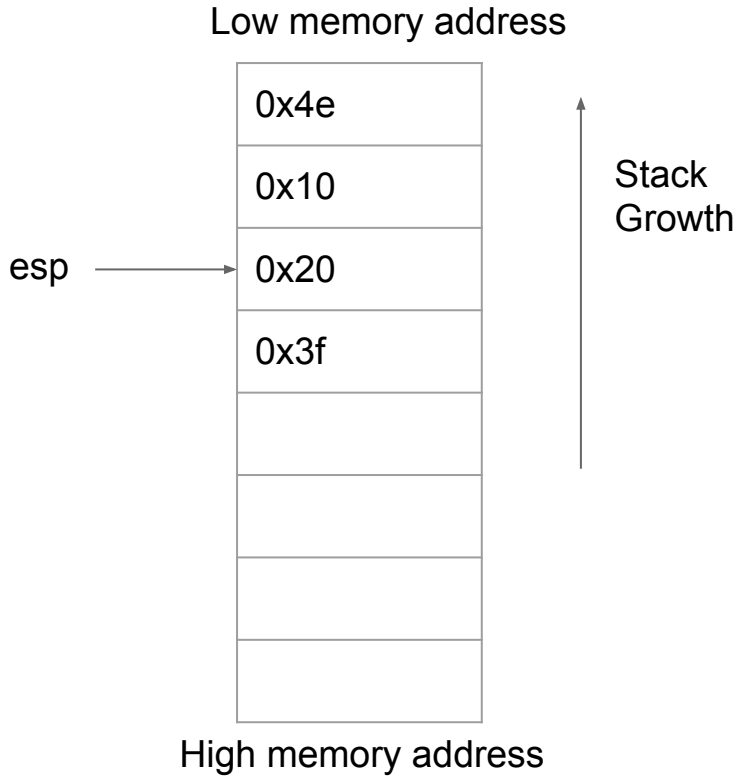


Stack

push $0x4e

pop %eax (eax contains 0x4e)

pop %ebx (ebx contains 0x10)

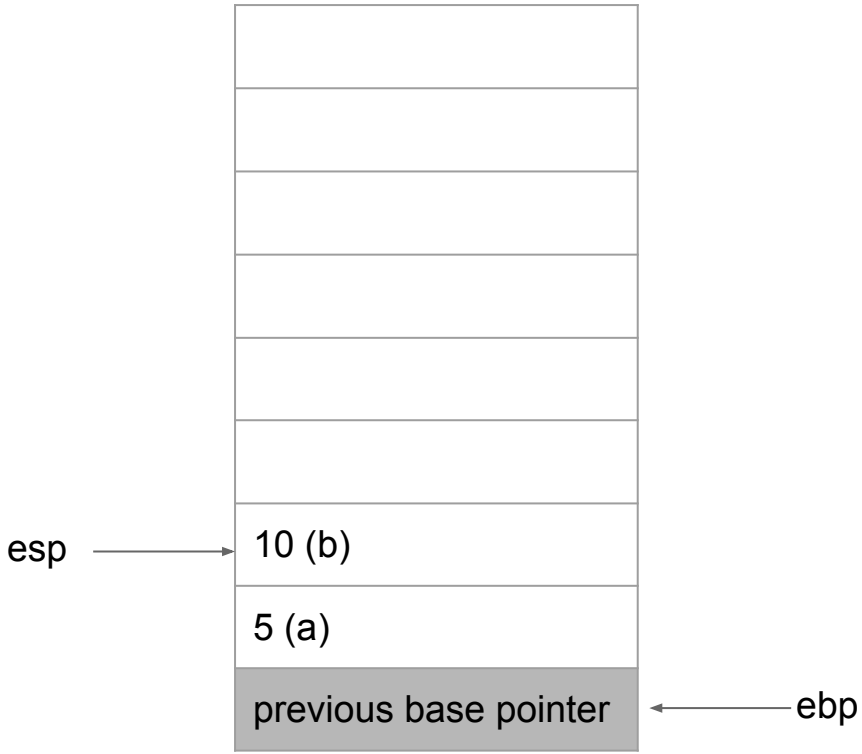


Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*

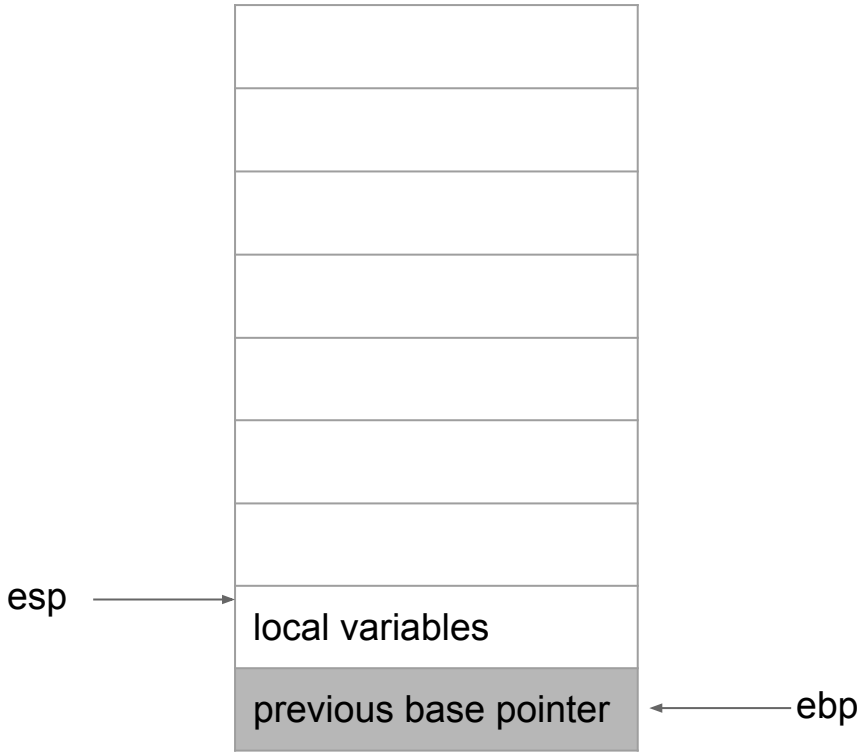
ex:
int a = 5; (push $5)
int b = 10; (push $10)



Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*

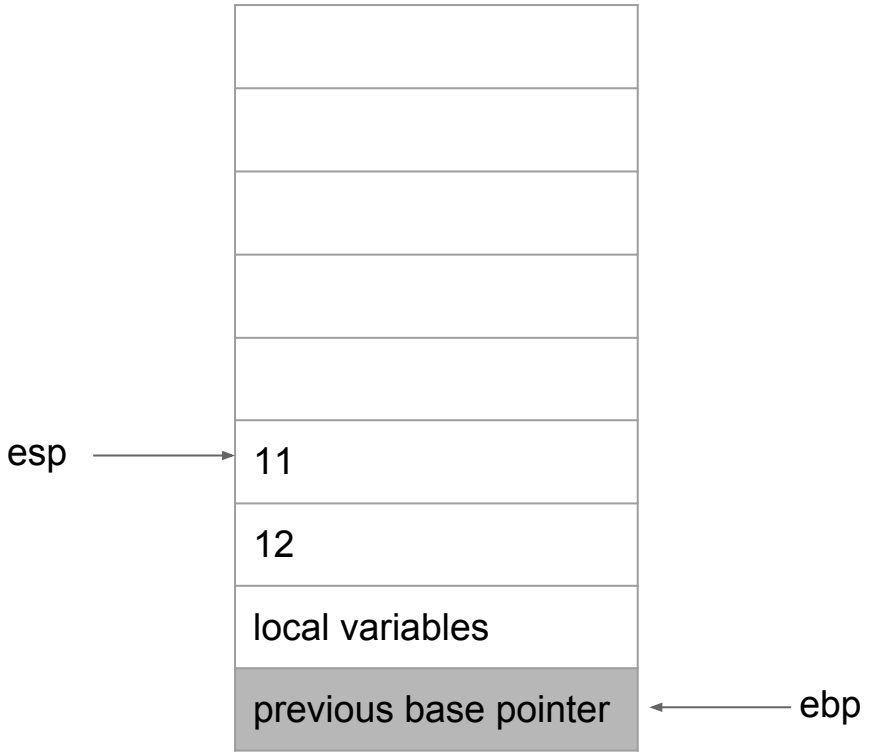


Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*

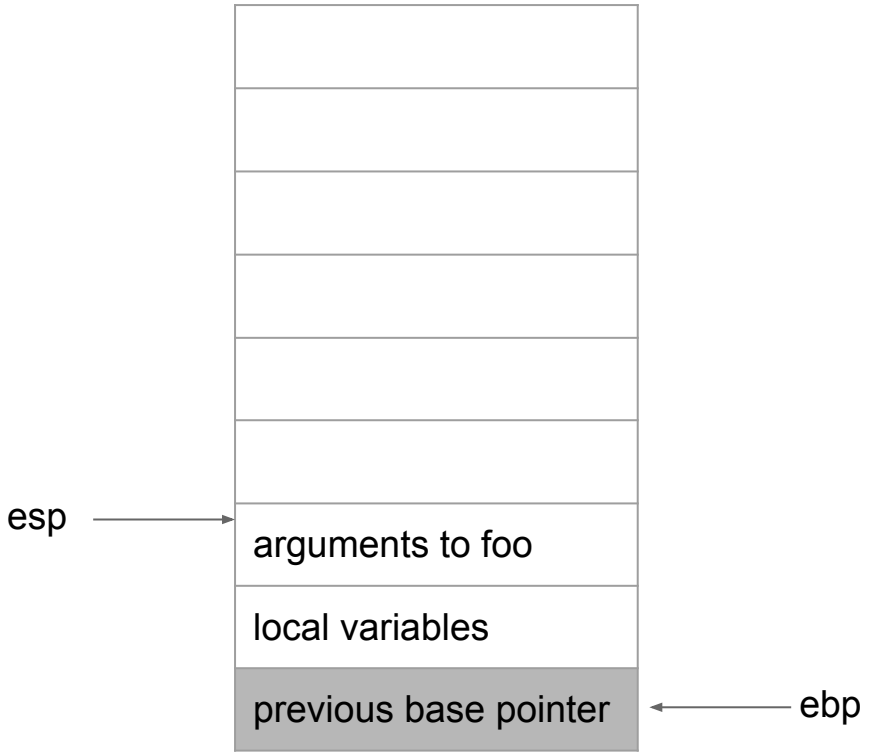
ex:
foo takes two integers,
in main: foo(11,12); (push $12, push $11)



Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*



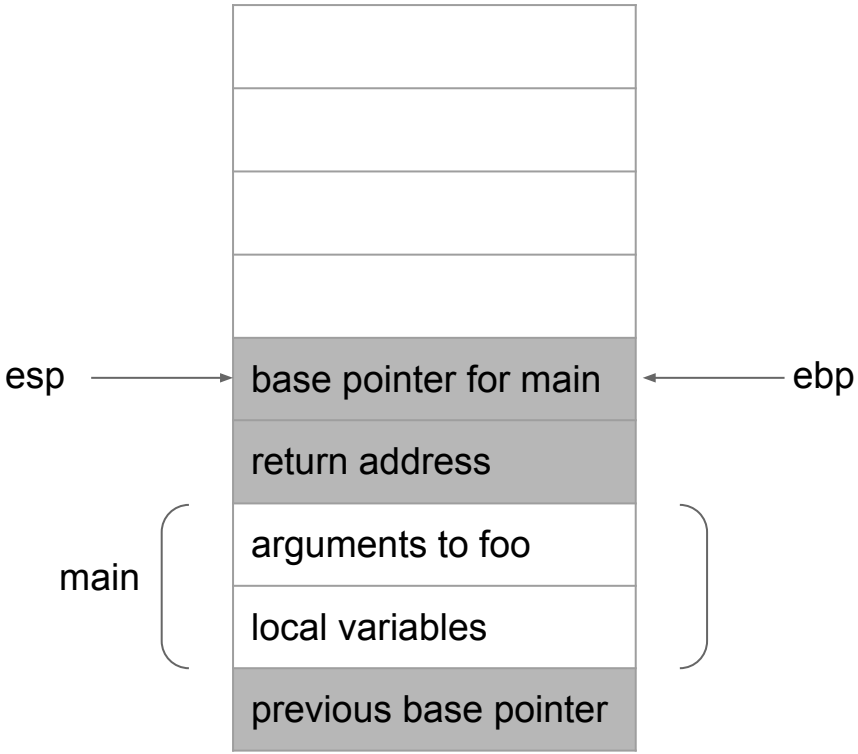
Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*

assembly:

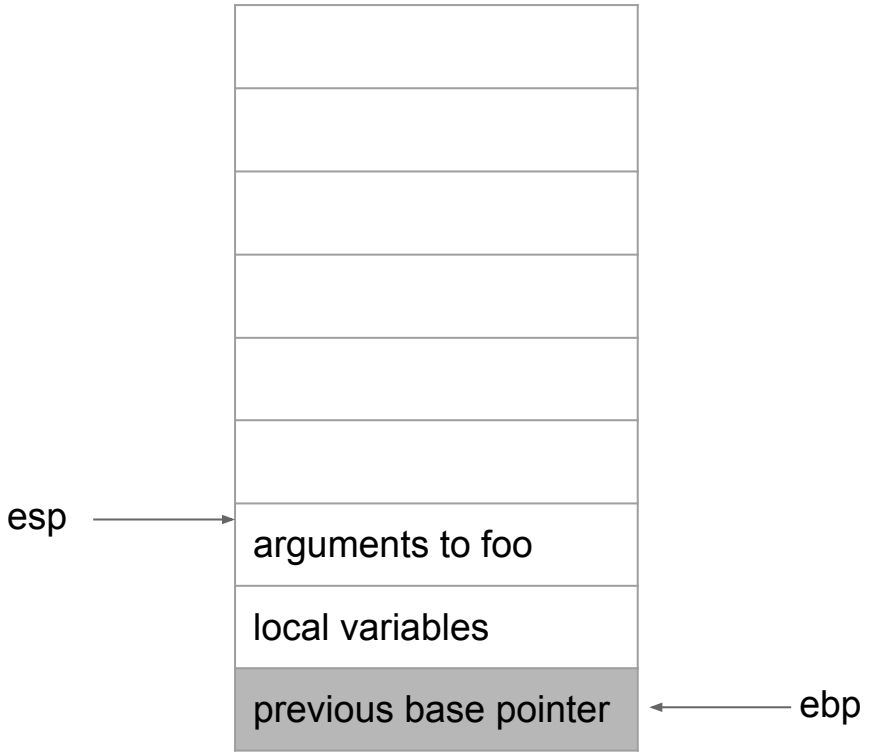
```
call foo
...
foo:
push $ebp
mov $esp,$ebp
...
```



Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*



Stack Frame

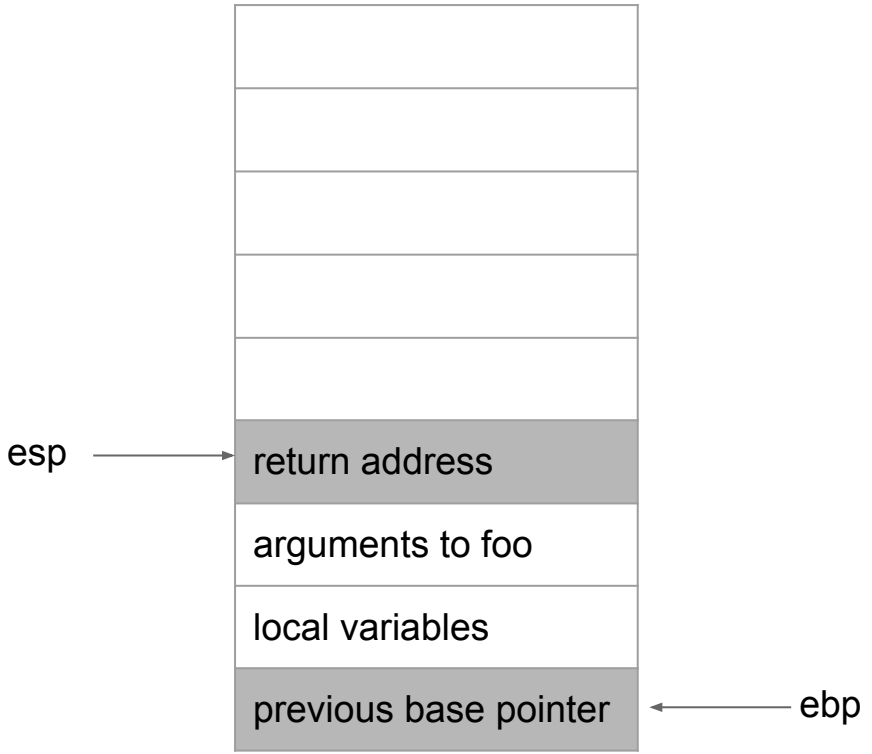
example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*

assembly: call foo

(push %eip;

address of foo's first instruction -> eip)

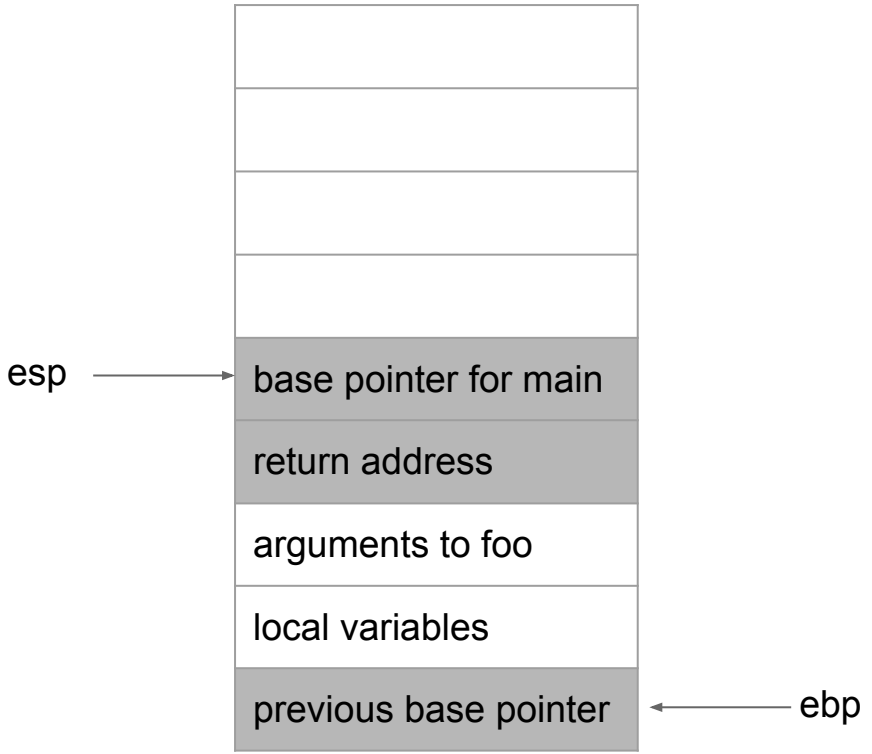


Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*

assembly: push %ebp

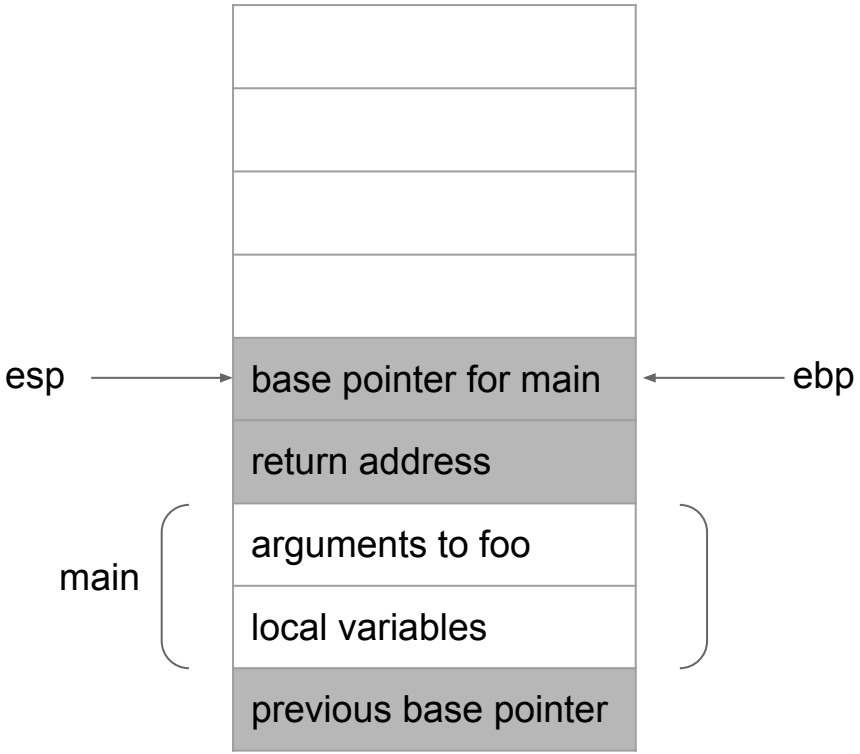


Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*

assembly: mov %esp,%ebp



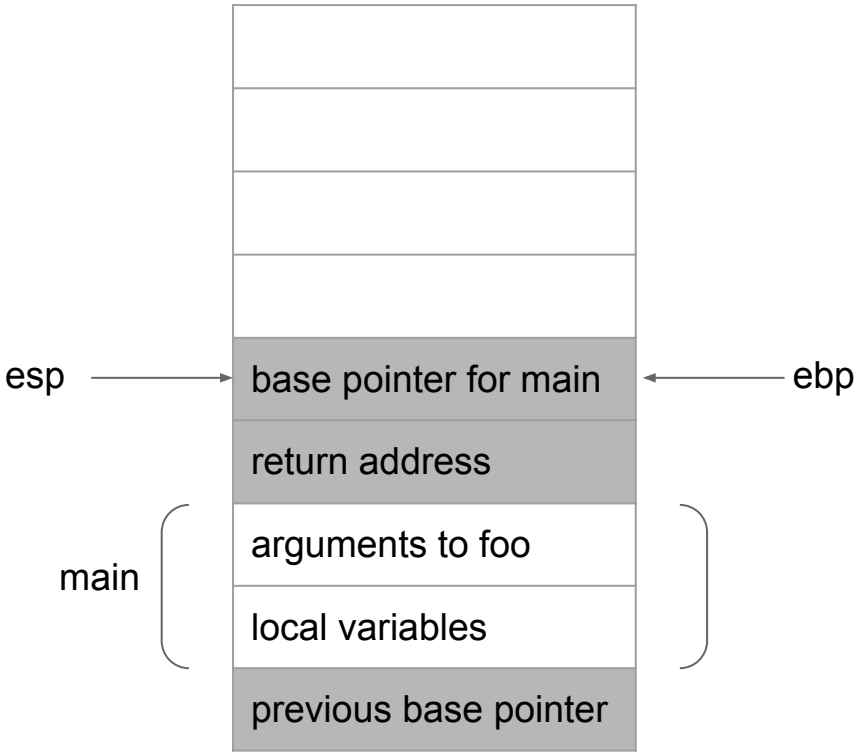
Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*

assembly:

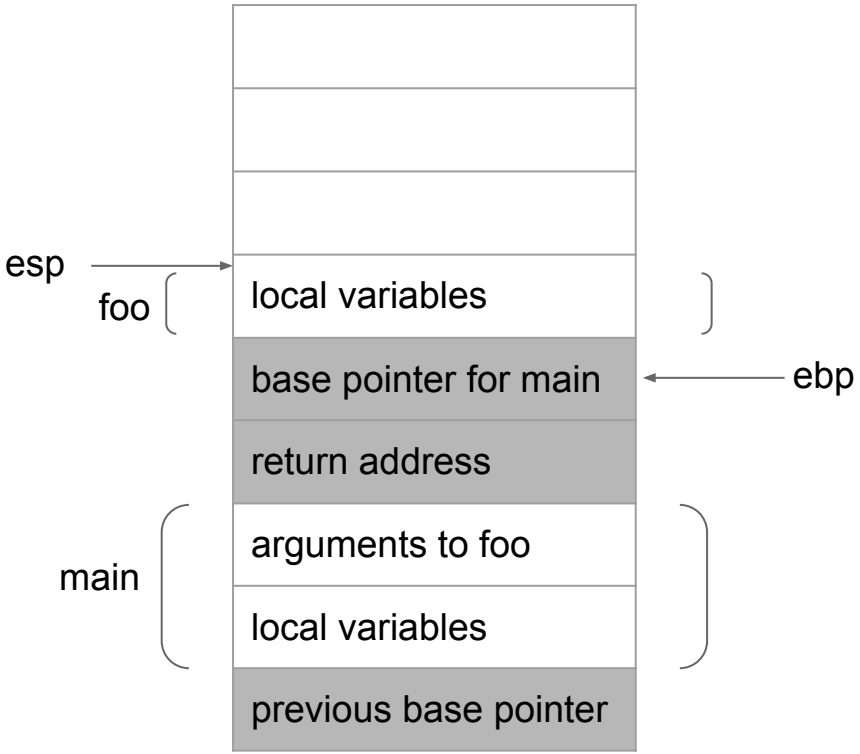
```
call foo
...
foo:
push $ebp
mov $esp,$ebp
...
```



Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*
4. Do stuff in *foo*

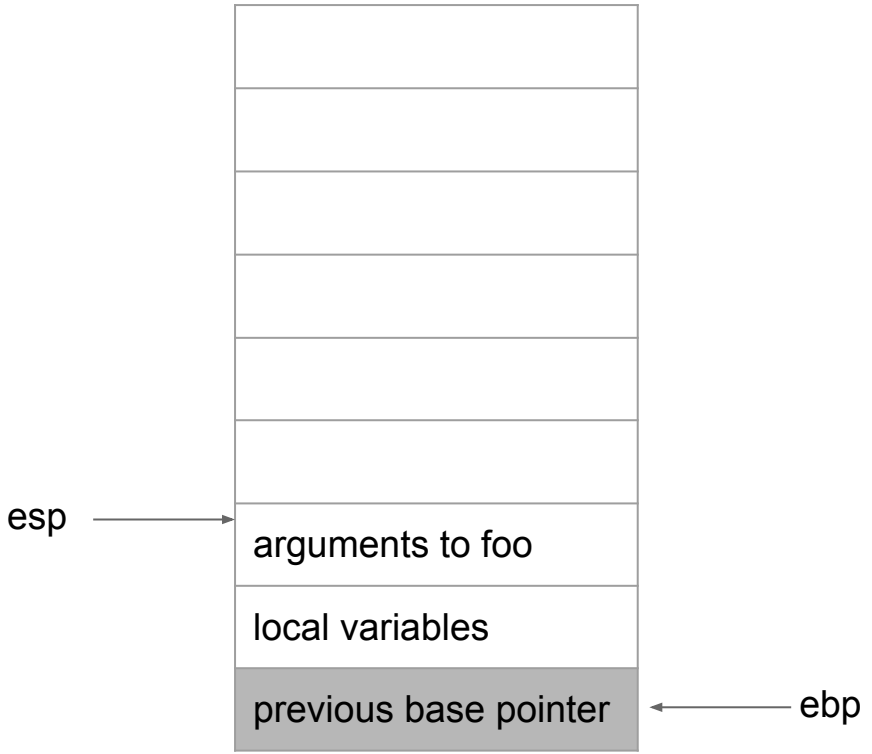


Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*
4. Do stuff in *foo*
5. Return to *main*

assembly:
leave
ret

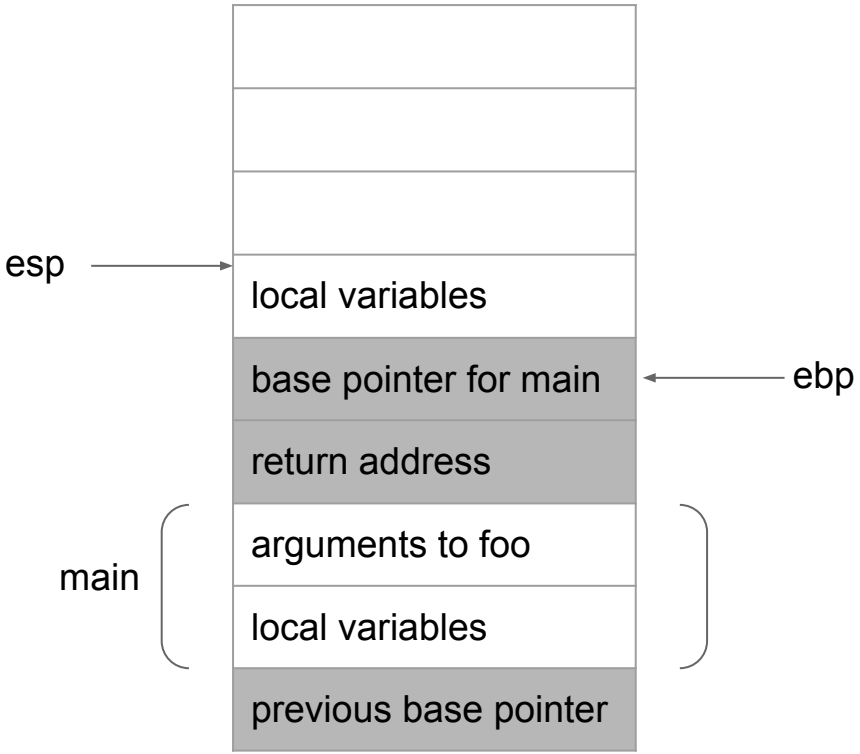


Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*
4. Do stuff in *foo*
5. Return to *main*

assembly:
leave
ret



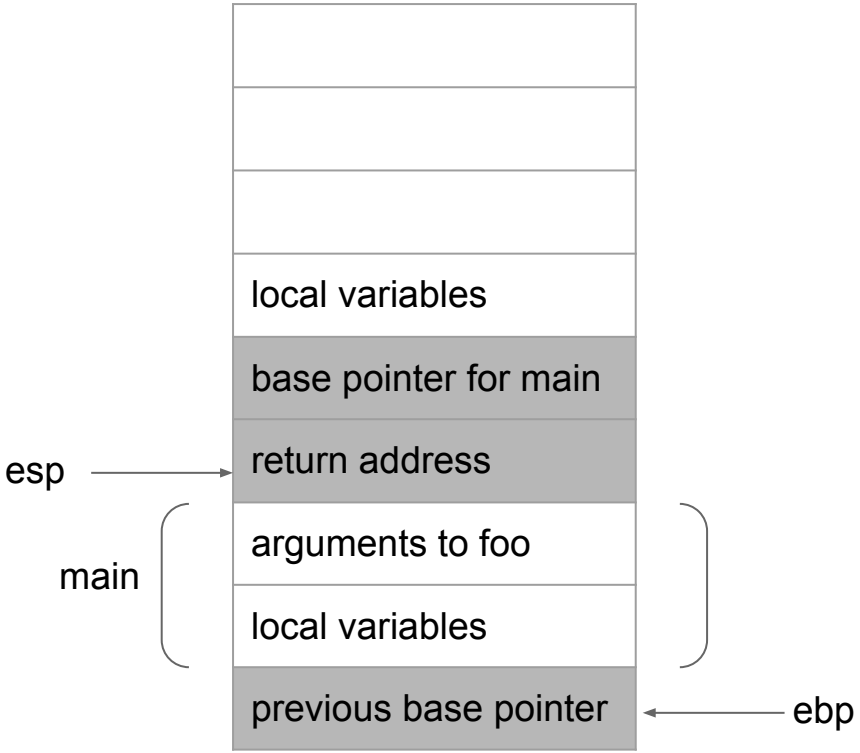
Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*
4. Do stuff in *foo*
5. Return to *main*

assembly: leave

(mov %ebp, %esp;
  pop %ebp)



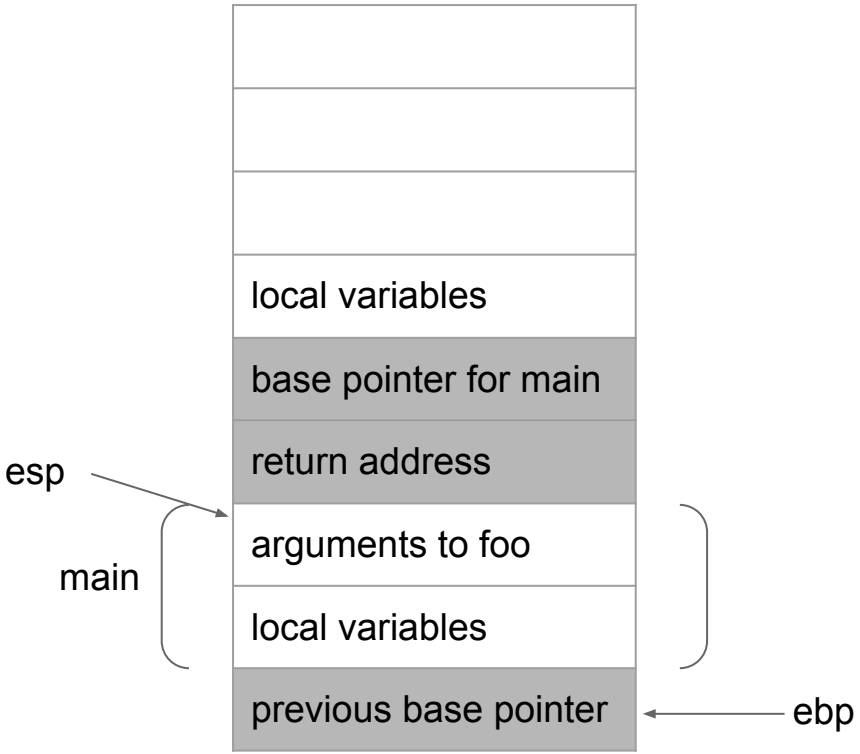
Stack Frame

example: *main* calls *foo*

1. Do stuff in *main*
2. Set up arguments to call *foo*
3. Set up stack frame for *foo*
4. Do stuff in *foo*
5. Return to *main*

assembly: ret

(pop %eip)



More Info?

https://courses.engr.illinois.edu/ece391/notes/student-notes.pdf

(Notes Set 0 and 1)

Next Week

MP1 Checkpoint 1