

Understand
procedures

Get to know
stack in
memory

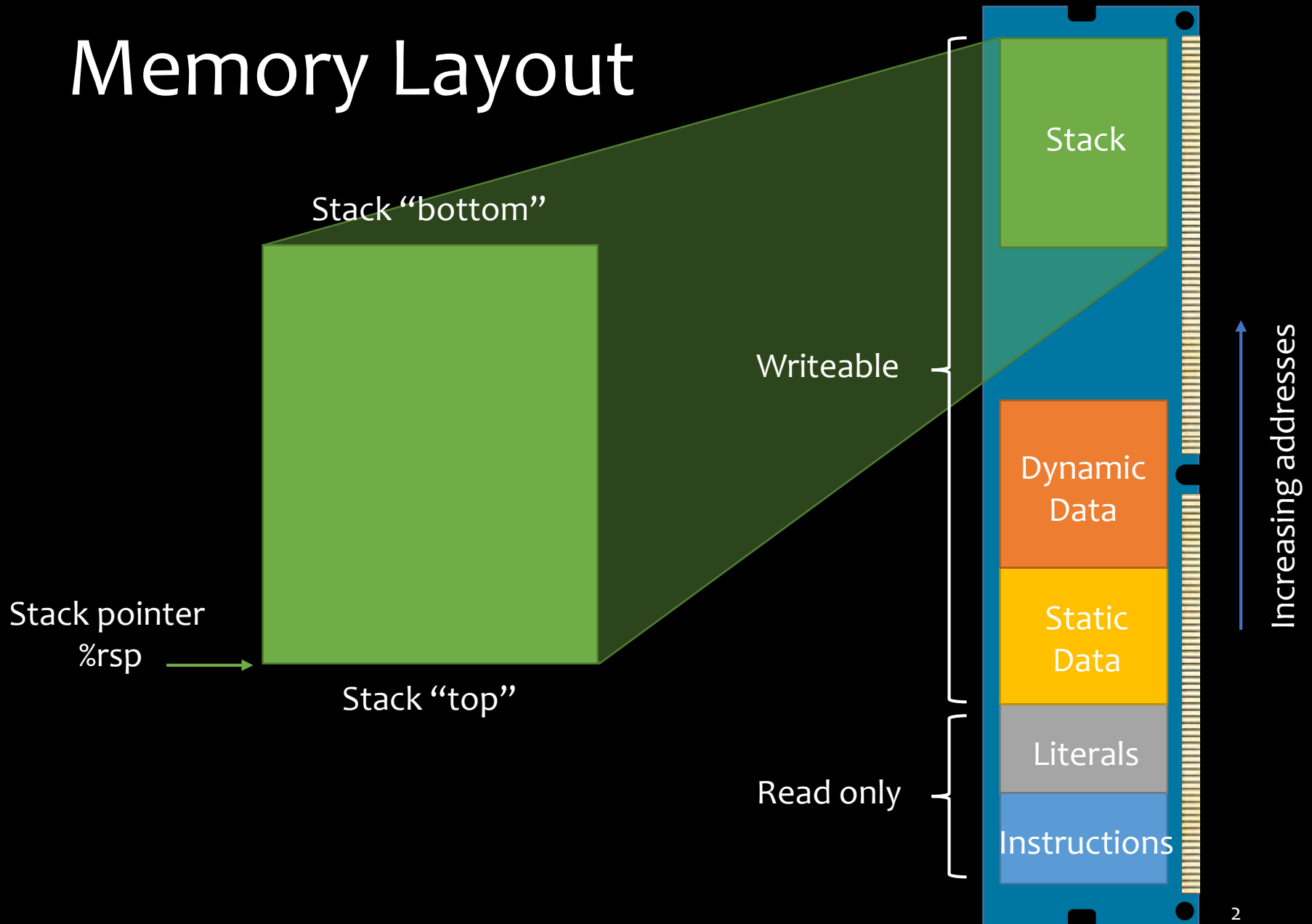
Using stack for
procedure
calling



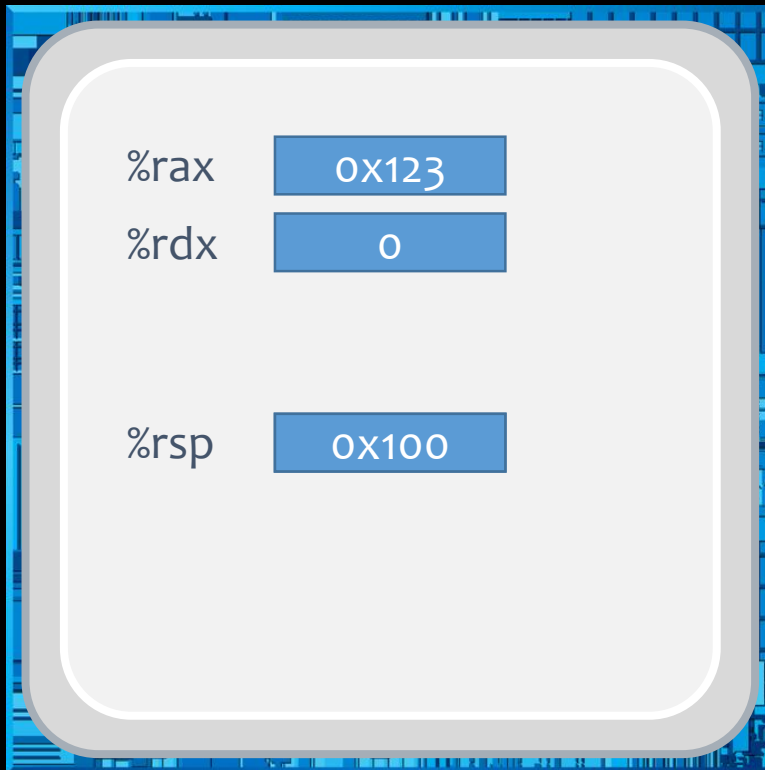
x86 Procedures

- ① Stack
- ② Control/Data Transfer
- ③ Local Storage
- ④ Recursive Procedures

Memory Layout

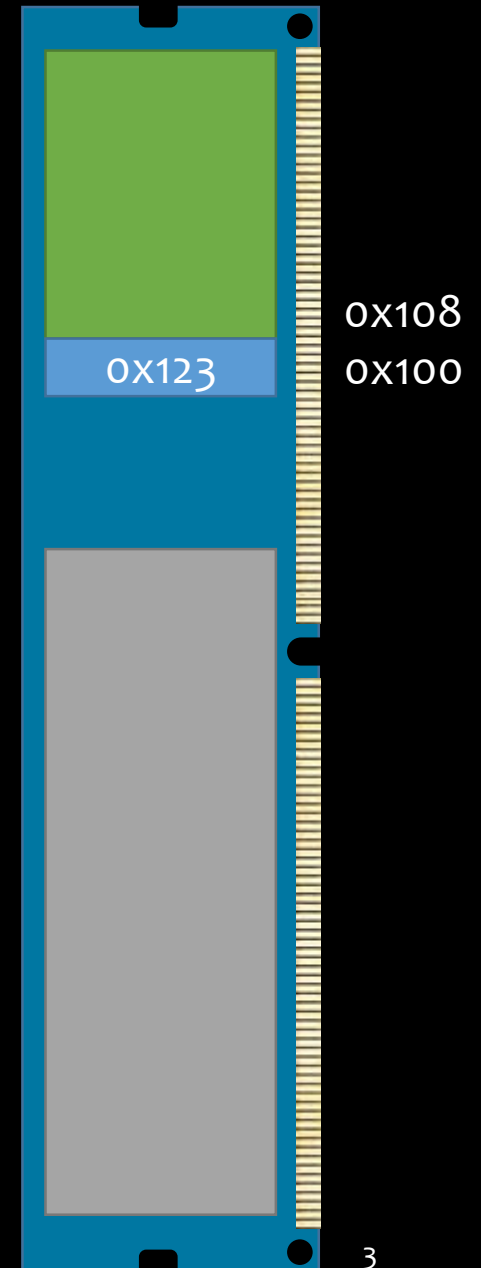


Stack Push

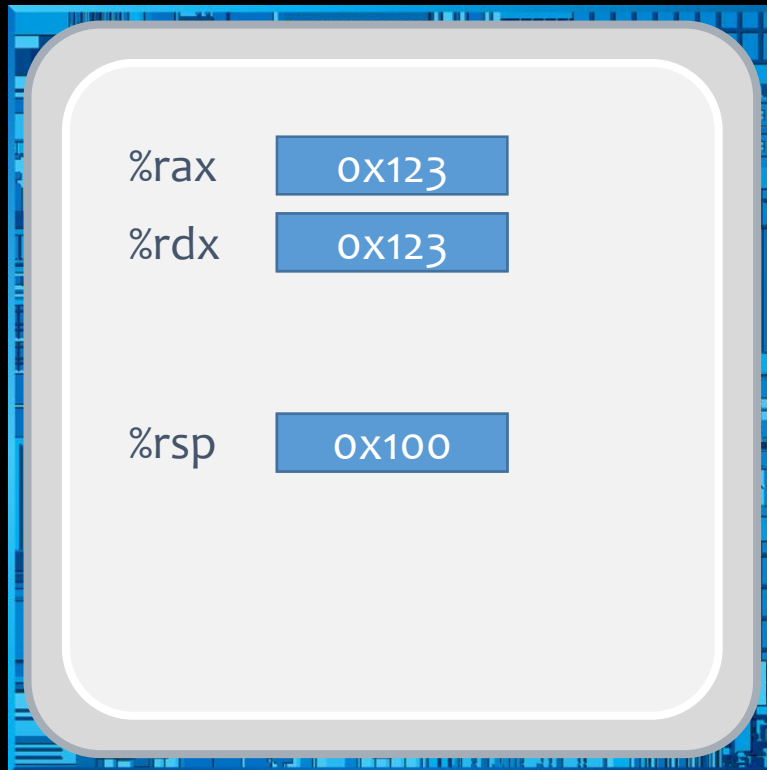


pushq %rax

PUSH S

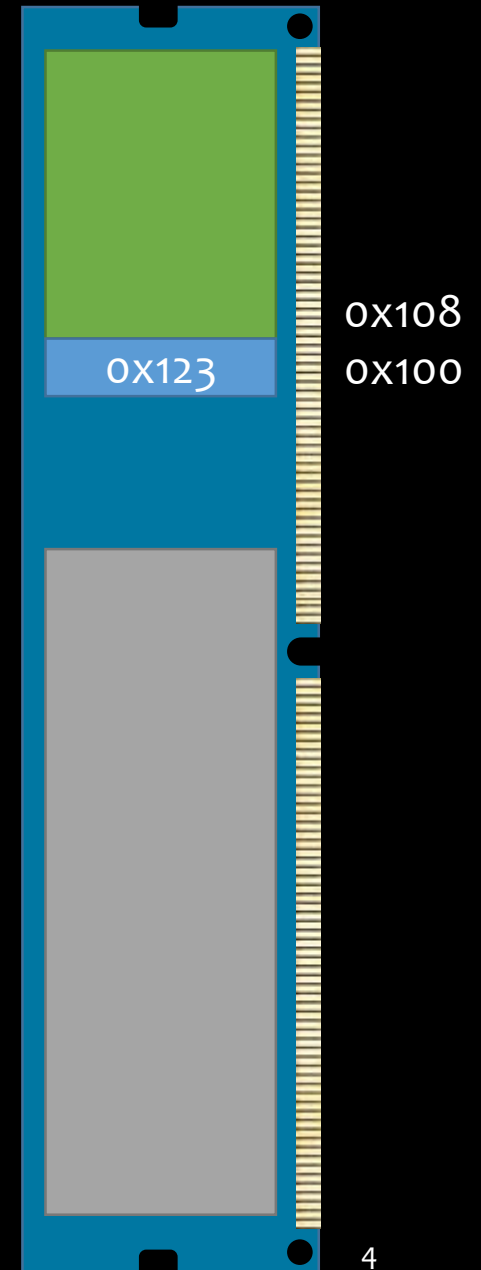


Stack Pop



`popq %rdx`

POP D



Procedure Call

CALL Label

Caller

Callee

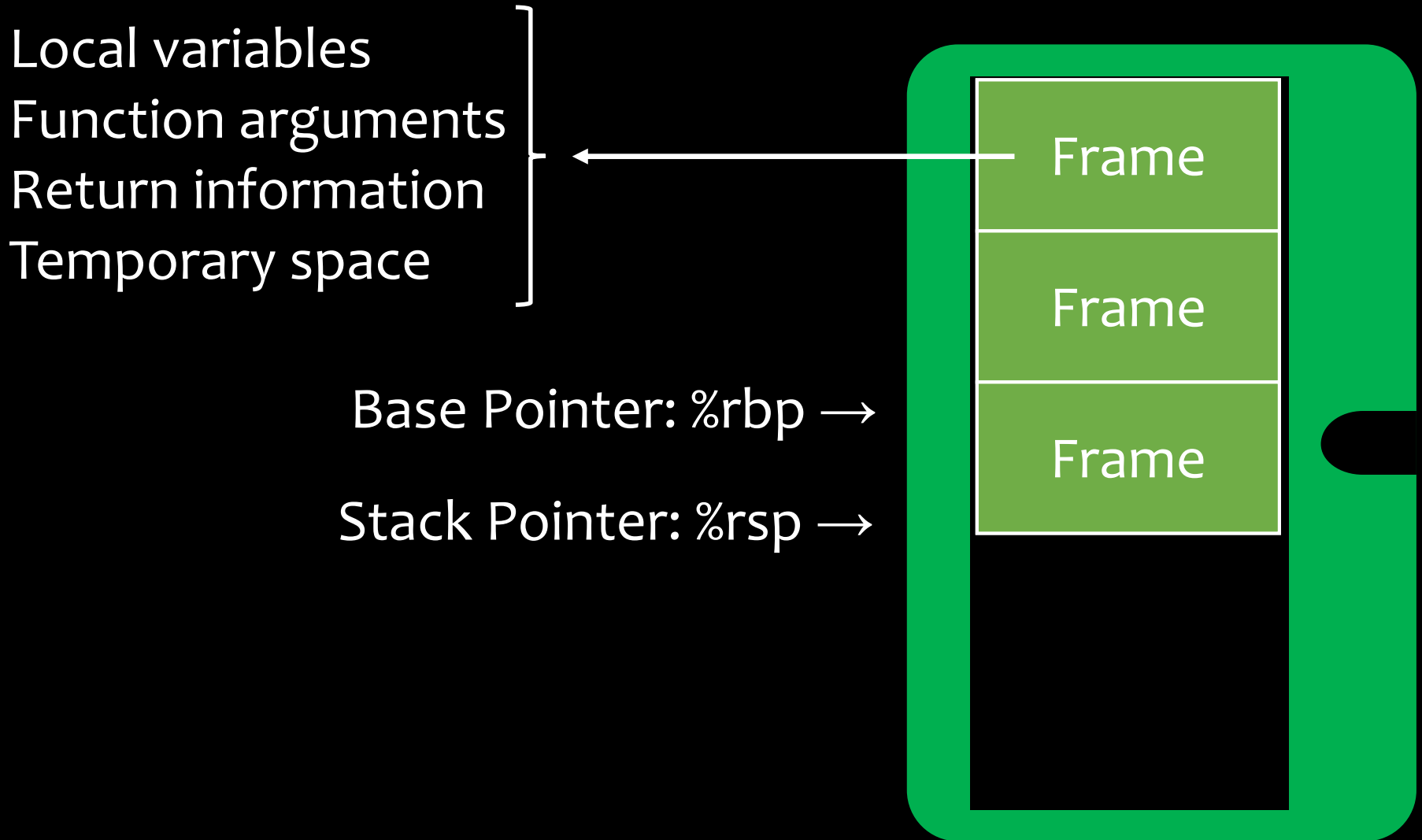
```
...  
<set up args>  
call  
<clean up args>  
<find return val>  
...
```

```
<create local vars>  
...  
<set up return val>  
<destroy local vars>  
ret
```

where to find return val

where to find args
“return address”

Stack Frame

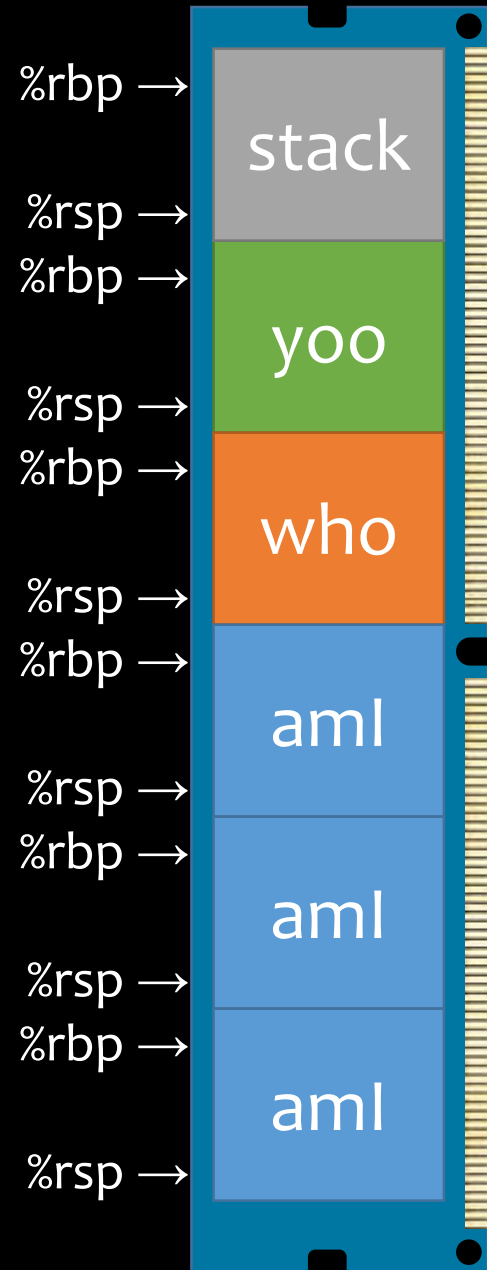
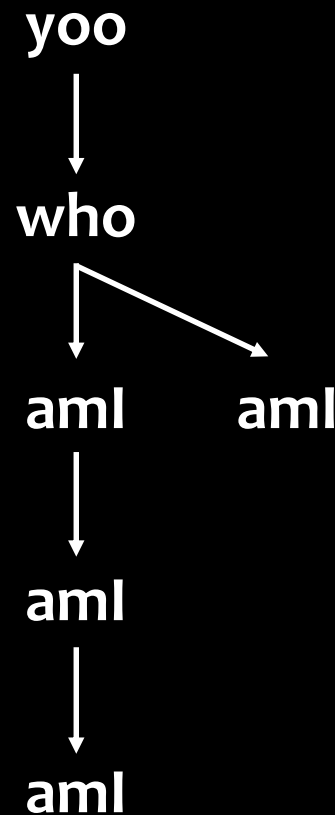


Call Chain Example

```
yoo(...)  
{  
  ...  
  who();  
  ...  
}
```

```
amI(...)  
{  
  ...  
  amI();  
  ...  
}
```

```
who(...)  
{  
  ...  
  amI();  
  ...  
  amI();  
  ...  
}
```



Register Saving Convention

%rax Return value

%rbx Callee saved

%rcx Argument #4

%rdx Argument #3

%rsi Argument #2

%rdi Argument #1

%rsp Stack pointer

%rbp Callee saved

%r8 Argument #5

%r9 Argument #6

%r10 Caller saved

%r11 Caller saved

%r12 Callee saved

%r13 Callee saved

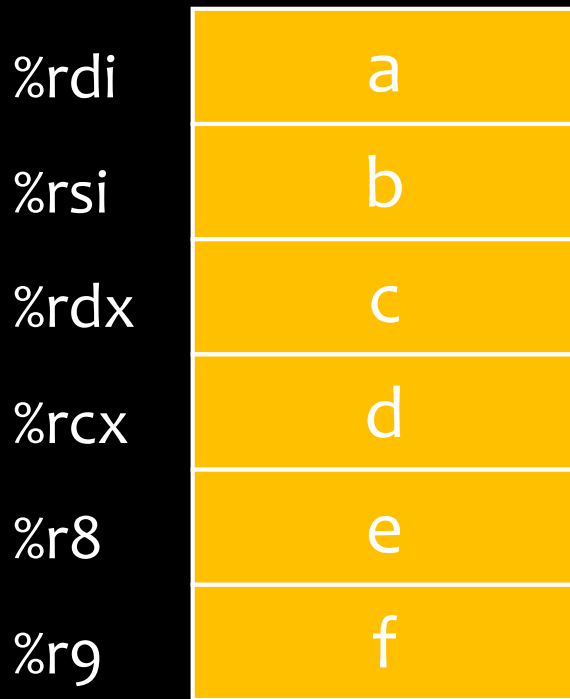
%r14 Callee saved

%r15 Callee saved


```

long myfunc(long a, long b, long c, long d,
            long e, long f, long g, long h)
{
    long xx = a * b * c * d * e * f * g * h;
    long yy = a + b + c + d + e + f + g + h;
    long zz = utilfunc(xx, yy, xx % yy);
    return zz + 20;
}

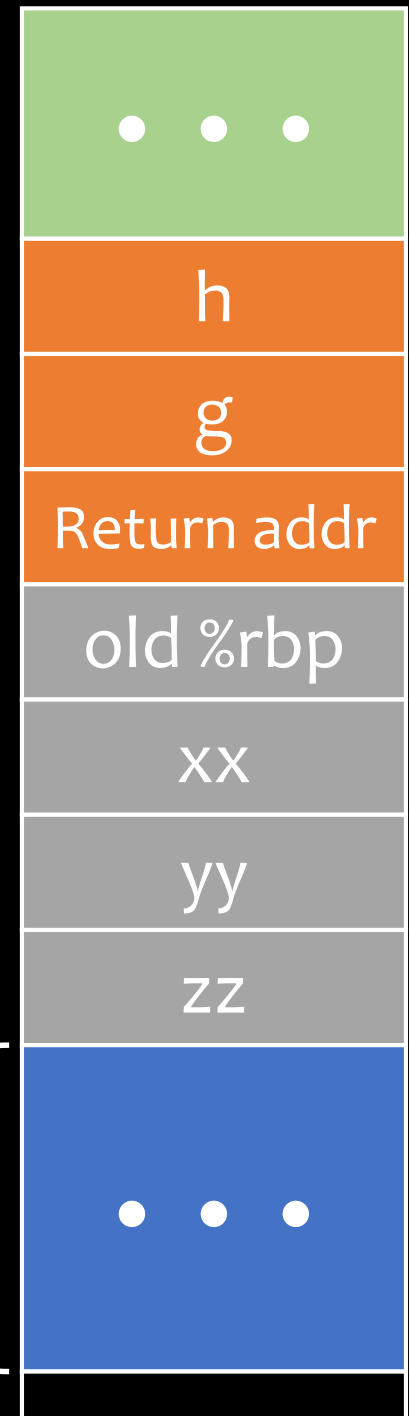
```



%rbp→

%rsp→

“red zone”



Functions can access
memory up to 128 bytes

x86-64

```
long utilfunc(long a, long b, long c)
{
    long xx = a + 2;
    long yy = b + 3;
    long zz = c + 4;
    long sum = xx + yy + zz;

    return xx * yy * zz + sum;
}
```

%rdi

a

%rsi

b

%rdx

c

References to stack frame → %rsp
%rbp now available for general-purpose
use

x86-64

%rbp →

%rsp →

Return
addr

old %rbp

xx

yy

zz

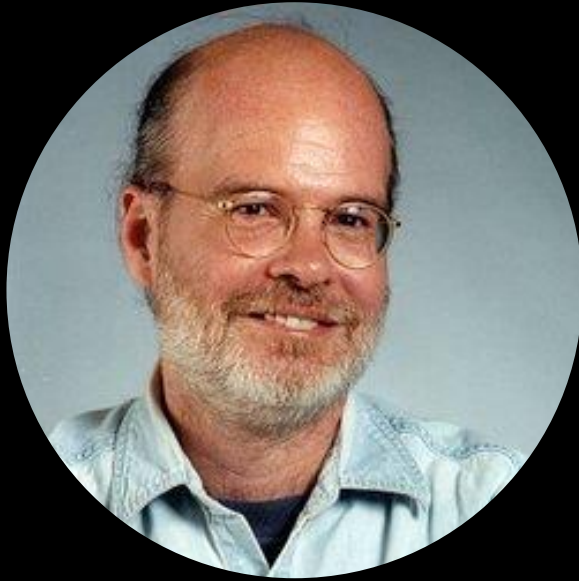
sum

...

“red zone”
128 bytes

Summary

- Memory Layout
- Stack
 - PUSH
 - POP
- Stack frame
- Procedure Call
 - CALL
 - RET
- Register Saving Convention
- Red zone



Charles Petzold

American programmer, Microsoft MVP

“ Programming in machine code is like eating with a toothpick.

”