Laboratory 7 Notes
X86 Stack

- Certain instructions implicitly modify the stack pointer (**push**, **pop, call, ret**)

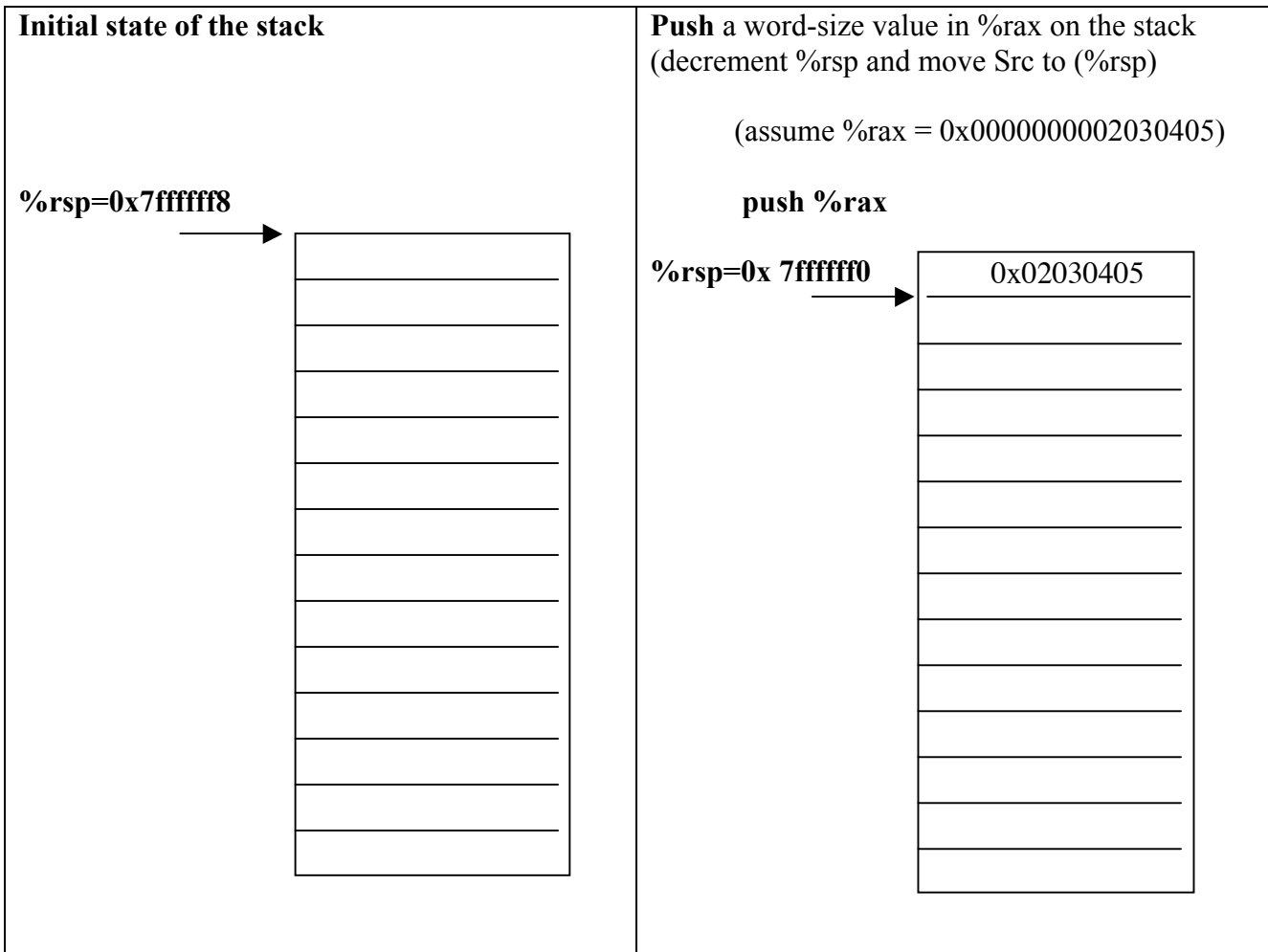- *%rsp (stack pointer)* always holds a pointer into the current stack frame

push *src*   1. Make space on the stack by decrementing %rsp:
$$\%rsp \leftarrow \%rsp - 8$$

2. Move *src* to the stack:
$$(\%rsp) \leftarrow src$$

Initial state of the stack	Push a word-size value in %rax on the stack (decrement %rsp and move Src to (%rsp)
	(assume %rax = 0x0000000002030405)
%rsp=0x7ffffff8	**push %rax**
	%rsp=0x 7ffffff0 0x02030405

pop *dest*   1. Move contents of top of stack to the *dest*

$$dest \leftarrow (\%rsp)$$

2. Release space on the stack by incrementing %rsp.

$$\%rsp \leftarrow \%rsp + 8$$

Initial State of Stack	Pop a word-size value from the stack.
	Pop %rbx
	(%rbx gets 0x0000000002030405)
	%rsp=0x7ffffff8

%rsp=0x7ffffff0

0x02030405

0x02030405

call *function*   1. Pushes the *return address* on stack (return address is the address of the instruction *following* the function call)

%rsp ← %rsp - 8
(%rsp) ← %rip (already updated for next instruction)

2. Puts the starting address of the *function* in %rip:
%rip ← starting address of *function*

ret   1. Pops the return address from the top of the stack into %rip (to resume execution of the *calling* function).

%rip ← (%rsp)
%rsp ← %rsp + 8

Conventions for drawing stack diagrams

To record the contents of the stack to understand how the stack is used, using the following notation:

- We use the model of memory where the stack has low addresses at the bottom and high at the top. Each row in the stack represents an 8-byte value. The initial **%rsp** with a subscript of **0** is pointing to the top of the current stack frame

stack frame of calling function	
%rsp$_0$ ---→ stack frame of current function	ret addr to calling program
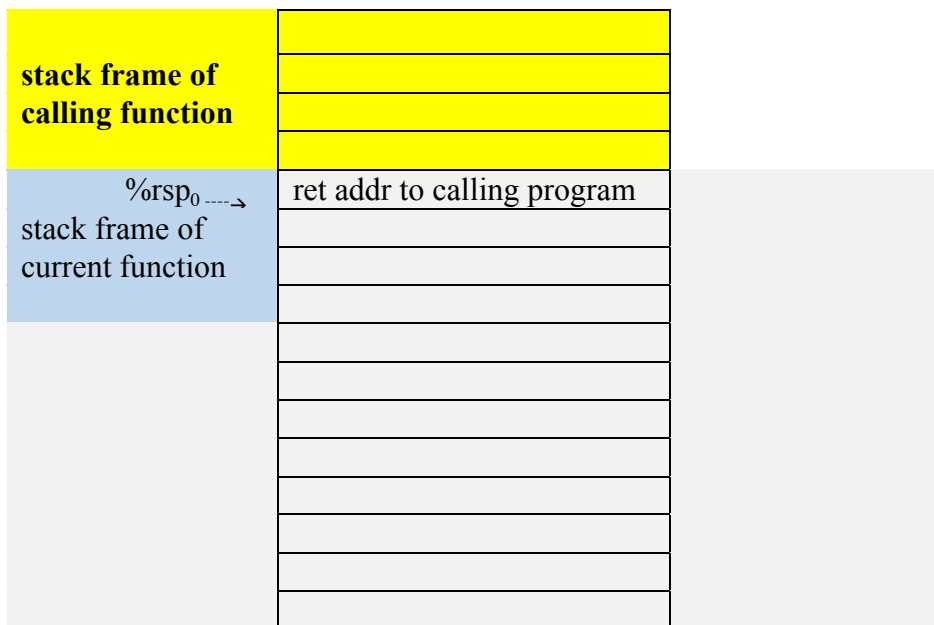
- Trace the effect on the stack of executing each instruction in the program by moving the position of the **%rsp** when it changes, (incrementing the subscript for each new value), and by recording new values on the stack as they are stored there.

- When the stack starts to empty, continue with the same notation, except use the right hand side of the stack diagram to indicate the changes.

- Also record changes to relevant registers.