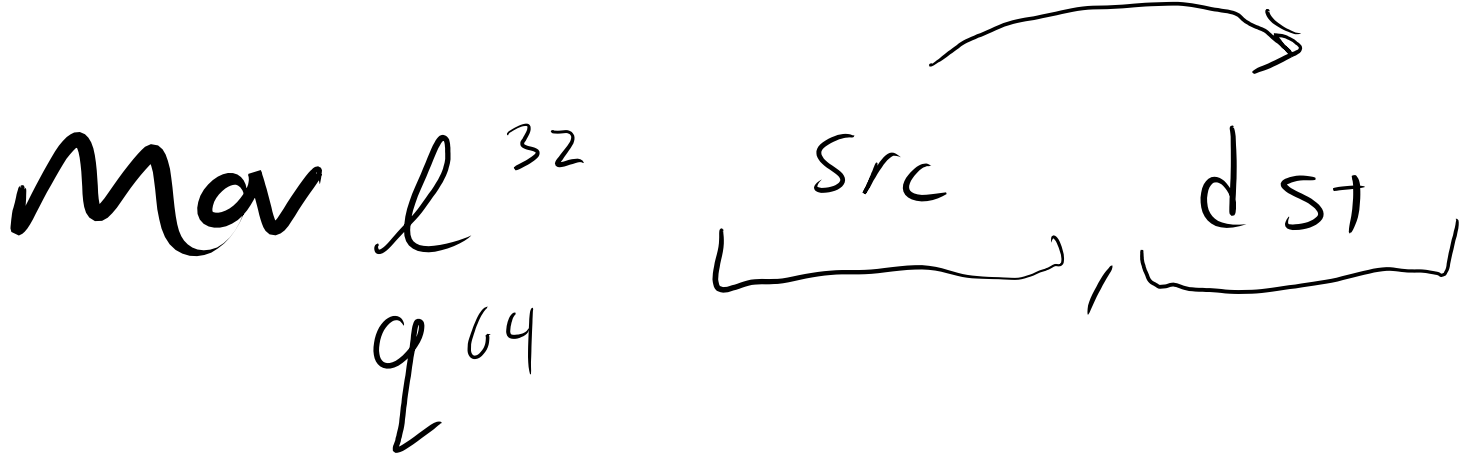


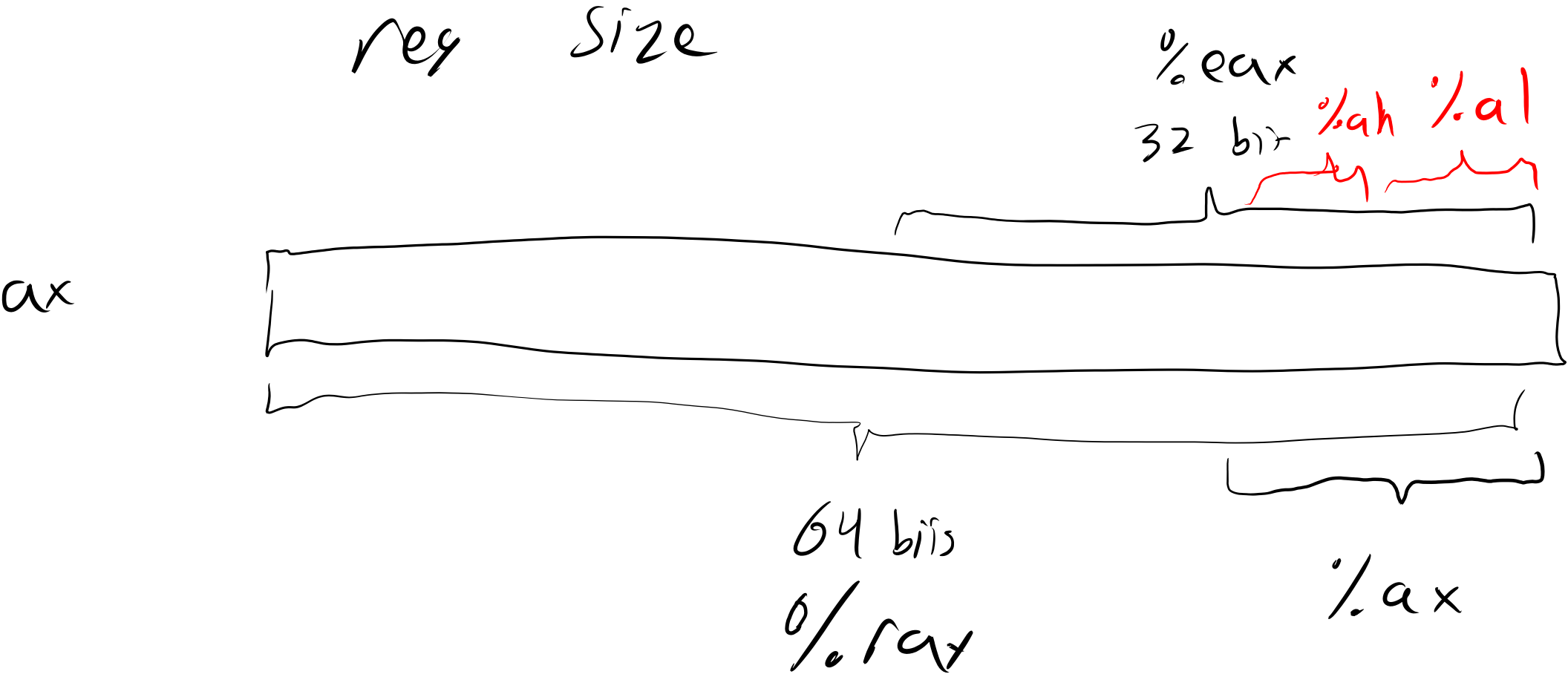
mem → reg

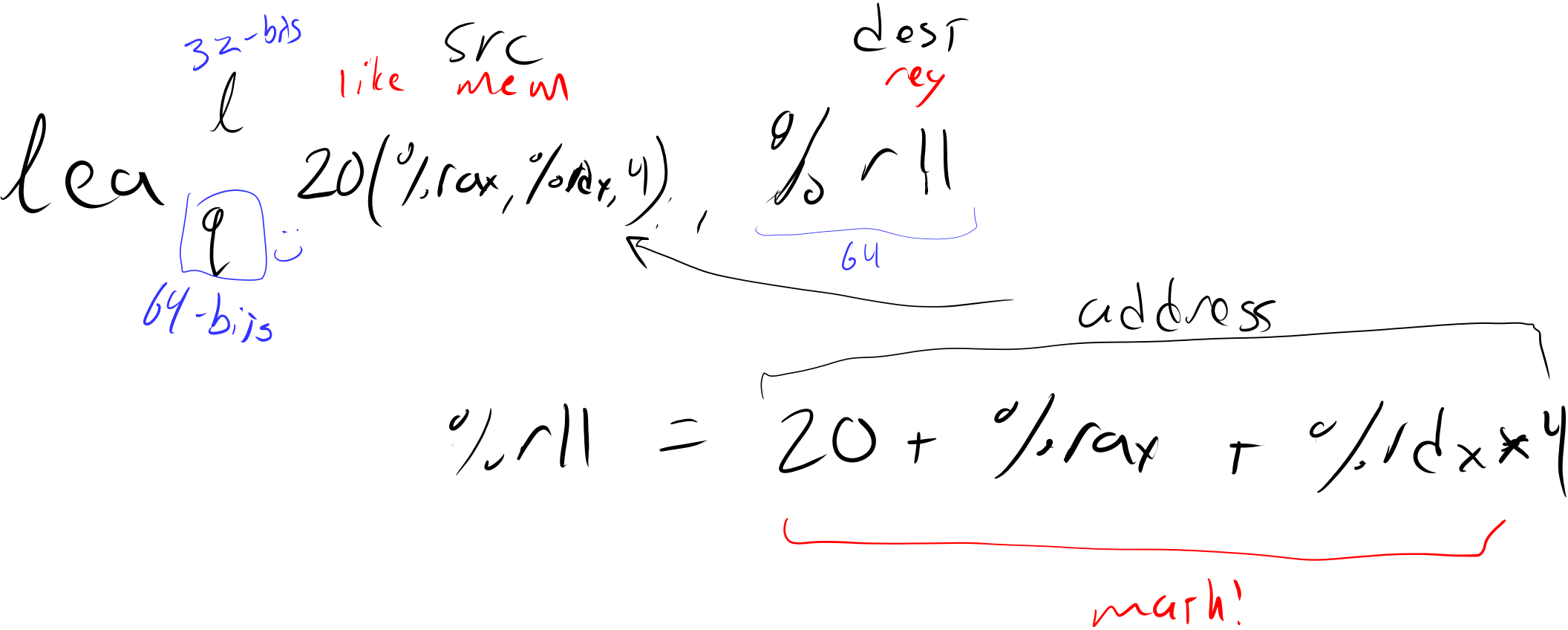
reg → mem

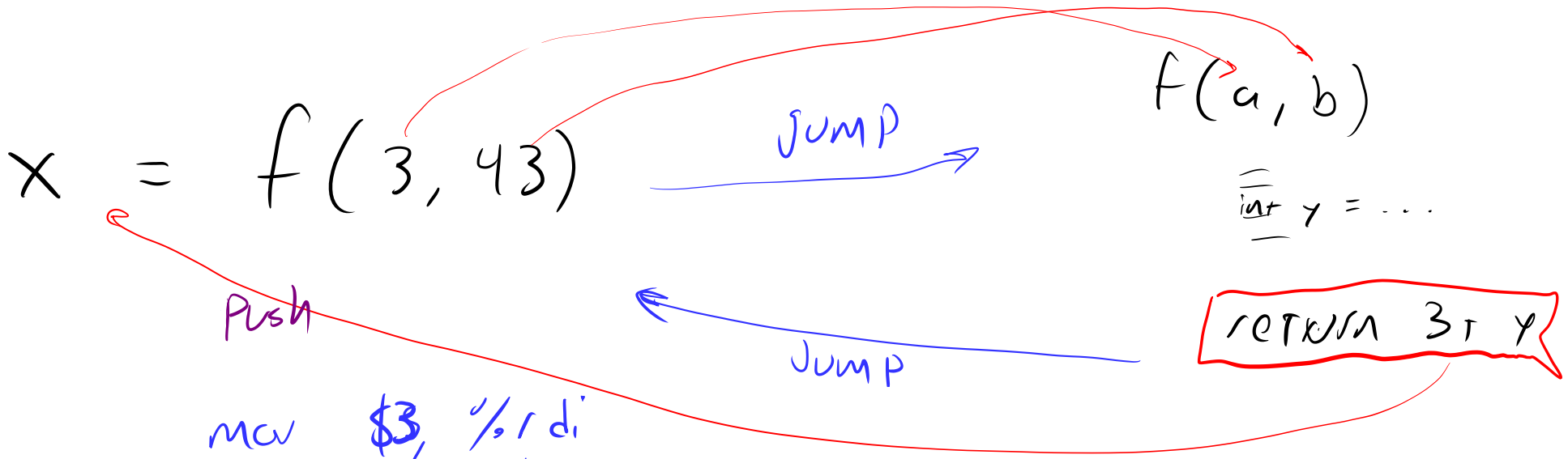
~~mem → mem~~

imm → reg

~~reg → imm~~


reg
size
%eax
32 bits
%ah
%al
ax
64 bits
%rax
%ax


32-bits
l
like
src
mem
dest
reg
lea
q
20(%rax,%rdx,4)
%r11
64
64-bits
address
%r11 = 20 + %rax + %rdx*4
math!



mov $3, %rdi
mov $43, %rsi
push ret addr
call f jump to f

f:

pop

Calling convention

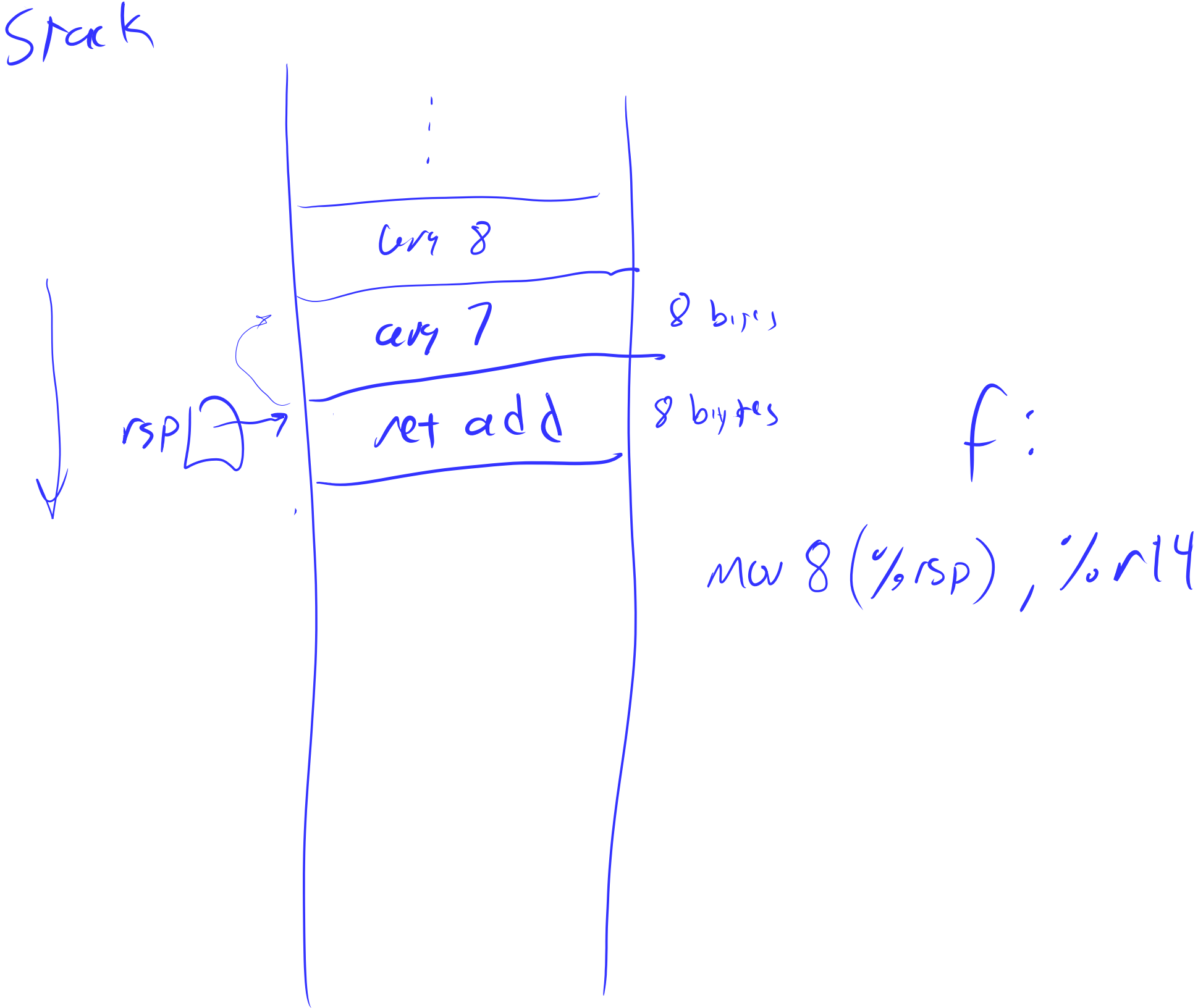
3 args

arg: 6 in reg %rdi, %rsi, %rdx, %rcx, %r8, %r9

7th & beyond push backwards

ret: %rax

ret pop ret addr
go to popped place


Stack
arg 8
arg 7
8 bytes
rsp
ret add
8 bytes
f:
mov 8(%rsp), %r14

src dst

add ——→ , ——→

Same limitation as move

dst += src

and it sets the condition codes

cmp src , dst

dst - src discard result

set condition codes

test is like and version of cmp

Jmp addr → always does

Jle addr → if cc <=

Je addr → if cc ==

jne

signed

g

ge

ne

a

b

ae

be

unsigned

above

below