P127-131

- 压入和弹出栈数据
  - pushq S
  - popq D

比如将一个四字值压入栈中，首先要将栈指针-8，然后新值吸入栈顶，因此 pushq %rbp 的行为等价于如下两条命令：

subq $8, %rsp

movq %rbp, (%rsp)

同样，弹出一个栈顶元素，读出数据，则需要在 mov 执行完后，再将栈顶的指针+8

movq (%rsp), %rax

addq $8, %rsp

- 加载有效地址
  - 指令：leaq S，D
  - 形式是从内存读取到寄存器，但实际上并未引用内存。目的操作数 D 必须为一个寄存器、

- 整数运算操作

指令		效果	描述
leaq	S, D	$D \leftarrow \&S$	加载有效地址
INC	D	$D \leftarrow D+1$	加1
DEC	D	$D \leftarrow D-1$	减1
NEG	D	$D \leftarrow -D$	取负
NOT	D	$D \leftarrow \sim D$	取补
ADD	S, D	$D \leftarrow D+S$	加
SUB	S, D	$D \leftarrow D-S$	减
IMUL	S, D	$D \leftarrow D*S$	乘
XOR	S, D	$D \leftarrow D \hat{} S$	异或
OR	S, D	$D \leftarrow D \mid S$	或
AND	S, D	$D \leftarrow D \& S$	与
SAL	k, D	$D \leftarrow D << k$	左移
SHL	k, D	$D \leftarrow D << k$	左移（等同于SAL）
SAR	k, D	$D \leftarrow D >>_A k$	算术右移
SHR	k, D	$D \leftarrow D >>_L k$	逻辑右移

整数算术操作。加载有效地址(leaq)指令通常用来执行简单的算术操作。其余的指令是更加标准的一元或二元操作。我们用$>>_A$和$>>_L$来分别表示算术右移和逻辑右移。注意，这里的操作顺序与 ATT 格式的汇编代码中的相反

- 一元和二元操作
  - 一元：只有一个操作数，既是源又是目的，比如 incq(%rsp) ：栈顶元素+1
  - 二元：第一个操作数可以是立即数、寄存器或内存位置，第二个操作数可以是寄存器或内存位置。需要注意的是，若第二个操作数是内存地址，则需要读出到寄存器，执行操作后，再写回内存。

- 移位操作
  - SAL k,D 或者 SHL k,D 表示 D 左移 k 位
  - SAR k,D 表示算术右移（填符号位）
  - SHR k,D 表示逻辑右移（填 0）