第三章作业

3.59

用 x，y 表示 p 分析：

$$
\begin{aligned}
\because p &= x \times y \\
&= (2^{64} \times x_h + x_l) \times (2^{64} \times y_h + y_l) \\
&= 2^{128} * x_h \times y_h + 2^{64} \times (x_h \times y_l + x_l \times y_h) + x_l \times y_l \\
\therefore p_l &= x_l * y_l \\
p_h &= x_h \times y_l + x_l \times y_h + (x_l \times y_l)_h
\end{aligned}
$$

反汇编代码分析：

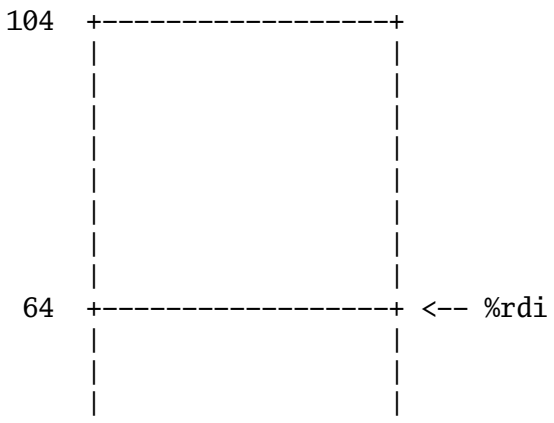
```
# dest in %rdi, x in %rsi, y in %rdx
store_prod:
  movq %rdx, %rax     # %rax=y
  cqto                # expand sign bit of %rax(y) to
   %rdx
  # %rdx=y_h

  movq %rsi, %rcx     # %rcx=x
  sarq $63, %rcx      # %rcx=x >> 63
  # %rcx=x_h

  imulq %rax, %rcx    # %rcx=x_h*y_l
  imulq %rsi, %rdx    # %rdx=y_h*x_l
  addq %rdx, %rcx     # %rcx=x_h*y_l+x_l*y_h
  mulq %rsi
  # %rdx:%rax=x_l*y_l,%rax=(x_l*y_l)_l=p_l
  addq %rcx, %rdx
  # %rdx=x_h*y_l+x_l*y_h+(x_l*y_l)_h=p_h

  movq %rax, (%rdi)
  movq %rdx, 8(%rdi)
  # *dest=p_h:p_l
  ret
```

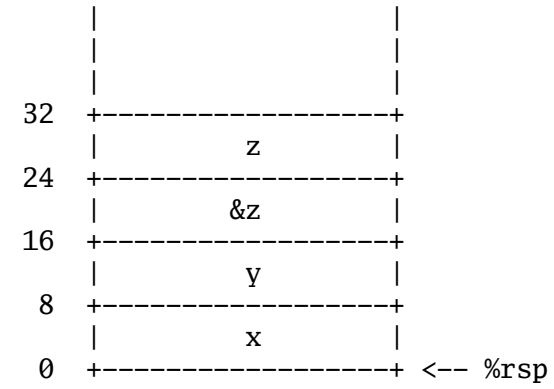
3.63

```
long switch_prob(long x, long n) {
  long result = x;
  switch (n) {
    case 60:
    case 62:
    result = x * 8;
    break;
    case 63:
    result = x / 8;
    break;
    case 64:
    result = x * 15;
    x = result;
    case 65:
    x *= x;
    default:
    result = x + 75;
  }
  return result;
}
```

3.67

A





B

传入了一个%rsp+64 的指针。

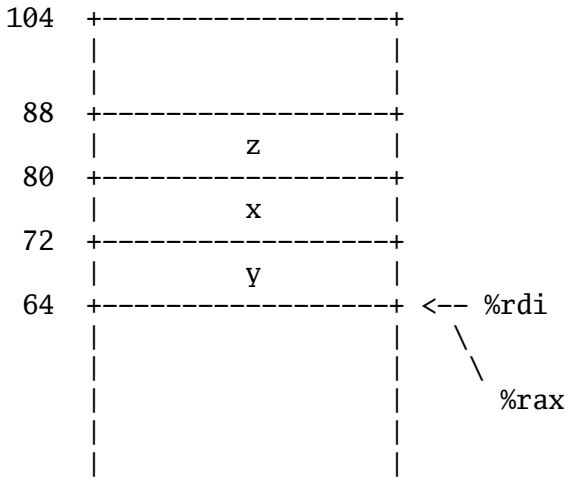
C

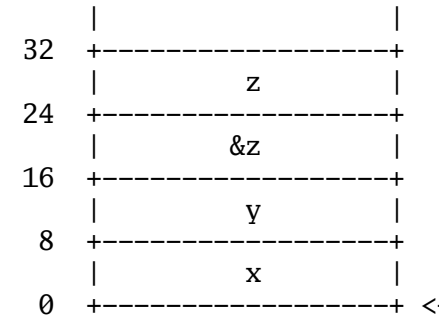
通过%rsp 指针加上偏移量。

D

通过访问%rsp+64 指针加上偏移量。

E





eval 通过访问%rsp 加上偏移量访问 r。

F

如果把结构作为参数，那么实际传递的会是一个空的位置指针，函数把数据存储在这个位置上，同时返回值也是这个指针。

3.71

```
#include <assert.h>
#include <stdio.h>

#define BUFFER_SIZE

int good_echo() {
  char buffer[BUFFER_SIZE];
  if (fgets(buffer, BUFFER_SIZE, stdin) == NULL) {
    return EOF;
  }
  puts(buffer);
}
```