P159-163

1. Switch 语句

   a. 根据整数索引值进行多重分支。

   b. 该语句提高了代码可读性，且通过使用跳转表，这种数据结构使得实现更加高效

      i. 跳转表：一个数组，确定跳转指令的目标

      ii. 如图是一个 switch 的 demo：

```
void switch_eg(long x, long n,
               long *dest)
{
    long val = x;

    switch (n) {

    case 100:
        val *= 13;
        break;

    case 102:
        val += 10;
        /* Fall through */

    case 103:
        val += 11;
        break;

    case 104:
    case 106:
        val *= val;
        break;

    default:
        val = 0;
    }
    *dest = val;
}
```

a）switch语句

```
1  void switch_eg_impl(long x, long n,
2                      long *dest)
3  {
4      /* Table of code pointers */
5      static void *jt[7] = {
6          &&loc_A, &&loc_def, &&loc_B,
7          &&loc_C, &&loc_D, &&loc_def,
8          &&loc_D
9      };
10     unsigned long index = n - 100;
11     long val;
12
13     if (index > 6)
14         goto loc_def;
15     /* Multiway branch */
16     goto *jt[index];
17
18  loc_A:    /* Case 100 */
19     val = x * 13;
20     goto done;
21  loc_B:    /* Case 102 */
22     x = x + 10;
23     /* Fall through */
24  loc_C:     /* Case 103 */
25     val = x + 11;
26     goto done;
27  loc_D:    /* Cases 104, 106 */
28     val = x * x;
29     goto done;
30  loc_def:  /* Default case */
31     val = 0;
32  done:
33     *dest = val;
34 }
```

b）翻译到扩展的C语言

图 3-22 switch 语句示例以及翻译到扩展的 C 语言。该翻译给出了跳转表 jt 的结构，以及如何访问它。作为对 C 语言的扩展，GCC 支持这样的表

对应的汇编代码：

```
    void switch_eg(long x, long n, long *dest)
    x in %rdi, n in %rsi, dest in %rdx
1   switch_eg:
2     subq    $100, %rsi              Compute index = n-100
3     cmpq    $6, %rsi                Compare index:6
4     ja      .L8                     If >, goto loc_def
5     jmp     *.L4(,%rsi,8)           Goto *jt[index]
6   .L3:                            loc_A:
7     leaq    (%rdi,%rdi,2), %rax     3*x
8     leaq    (%rdi,%rax,4), %rdi     val = 13*x
9     jmp     .L2                     Goto done
10  .L5:                            loc_B:
11    addq    $10, %rdi               x = x + 10
12  .L6:                            loc_C:
13    addq    $11, %rdi               val = x + 11
14    jmp     .L2                     Goto done
15  .L7:                            loc_D:
16    imulq   %rdi, %rdi              val = x * x
17    jmp     .L2                     Goto done
18  .L8:                            loc_def:
19    movl    $0, %edi                val = 0
20  .L2:                            done:
21    movq    %rdi, (%rdx)            *dest = val
22    ret                             Return
```

不同代码块实现了 switch 语句的不同分支，类似的，汇编代码块计算了寄存器%rdi 的值，跳转到函数结尾处由 L2 表示的位置。