第四章作业

4.47

A

```c
#include <stdio.h>

void bubble(long *data , long count) {
  long *i, *last;
  for (last = data + count - 1; last > data; --last)
    for (i = data; i < last; ++i)
      if (*(i + 1) < *i) {
        long t = *(i + 1);
        *(i + 1) = *i;
        *i = t;
      }
}

int main() {
  long a[4] = {4, 2, 3, 1};
  bubble(a, 4);
  for (int i = 0; i < 4; ++i) printf("%ld ", a[i]);
  return 0;
}
```

B

由于 Y86-64 中无乘法和 leaq 指令，所以 bubble 函数改为传入两个指针 begin 和 end，表示数组的头和尾（均被包含在数组中）。

```
.pos 0
  irmovq stack, %rsp
  call main
  halt

# Array of 4 elements
.align 8
begin:
  .quad 0x0000000000000002
  .quad 0x0000000000000004
  .quad 0x0000000000000001
end:
  .quad 0x0000000000000003
```

```
14
15 main:
16   irmovq begin, %rdi
17   irmovq end, %rsi
18   call bubble
19   ret
20
21 # void bubble(long *begin, long *end)
22 # begin in %rdi, end in %rsi
23 bubble:
24   jmp L2
25 L4:
26   mrmovq 8(%rax), %r9
27   mrmovq (%rax), %r10
28   rrmovq %r9, %r8
29   subq %r10, %r8
30   jge L3
31   rmmovq %r10, 8(%rax)
32   rmmovq %r9, (%rax)
33 L3:
34   irmovq $8, %r8
35   addq %r8, %rax
36   jmp L5
37 L6:
38   rrmovq %rdi, %rax
39 L5:
40   rrmovq %rsi, %r8
41   subq %rax, %r8
42   jg L4
43   irmovq $8, %r8
44   subq %r8, %rsi
45 L2:
46   rrmovq %rsi, %r8
47   subq %rdi, %r8
48   jg L6
49   ret
50
51
52 .pos 0x200
53 stack:
```

```
7778 hitics/chap4 git:(master) ✗ » ./sim/misc/yas 4.47.ys
7779 hitics/chap4 git:(master) ✗ » ./sim/misc/yis 4.47.yo
Stopped in 111 steps at PC = 0x13.  Status 'HLT', CC Z=1 S=0 O=0
Changes to registers:
%rax:   0x0000000000000000      0x0000000000000020
%rsp:   0x0000000000000000      0x0000000000000200
%rsi:   0x0000000000000000      0x0000000000000018
%rdi:   0x0000000000000000      0x0000000000000018
%r9:    0x0000000000000000      0x0000000000000002
%r10:   0x0000000000000000      0x0000000000000001

Changes to memory:
0x0018: 0x0000000000000002      0x0000000000000001
0x0020: 0x0000000000000004      0x0000000000000002
0x0028: 0x0000000000000001      0x0000000000000003
0x0030: 0x0000000000000003      0x0000000000000004
0x01f0: 0x0000000000000000      0x0000000000000055
0x01f8: 0x0000000000000000      0x0000000000000013
```

Figure 1: 运行结果

4.51

阶段	$iaddq$ **V**, **rB**
取值	$icode : ifun \leftarrow M_1[PC]$ $rA : rB \leftarrow M_1[PC]$ $valC \leftarrow M_8[PC + 2]$ $valP \leftarrow PC + 10$
译码	$valB \leftarrow R[rB]$
执行	$valE \leftarrow valB + valC$ $Set\ \ CC$
访问	
写回	$R[rB] \leftarrow valE$
更新 PC	$PC \leftarrow valP$

4.59

4.47 的版本最优。

4.47 版本中的循环部分如下：

```
1 L4:
```

```
  mrmovq 8(%rax), %r9
  mrmovq (%rax), %r10
  rrmovq %r9, %r8
  subq %r10, %r8
  jge L3
  rmmovq %r10, 8(%rax)
  rmmovq %r9, (%rax)
```

单个循环最多有 7 条指令。

4.48 版本中的循环部分如下：

```
L4:
  mrmovq 8(%rax), %r9
  mrmovq (%rax), %r10
  rrmovq %r9, %r8
  subq %r10, %r8
  cmovl %r9, %r11
  cmovl %r10, %r9
  cmovl %r11, %r10
  rmmovq %r9, 8(%rax)
  rmmovq %r10, (%rax)
```

单个循环最多有 9 条指令。

4.49 版本中的循环部分如下：

```
L4:
  mrmovq 8(%rax), %r9
  mrmovq (%rax), %r10
  rrmovq %r9, %r8
  rrmovq %r10, %r11
  xorq %r9, %r10
  subq %r11, %r8
  cmovge %r11, %r9
  xorq %r10, %r9
  xorq %r9, %r10
  rmmovq %r9, 8(%rax)
  rmmovq %r10, (%rax)
```

单个循环最多有 11 条指令。

4.47 的版本的指令最少，单次循环耗时较少。