**第四章**

**4.47**

(1)

1. **void** bubble\_p(**long** \*data,**long** count){
2. **long** \*i,\*last;
3. **for**(last = data+count -1;last > data;last--){
4. **for**(i = data;i < last;i++){
5. **if**(\*(i+1) < \*i){
6. **long** t = \*(t+1);
7. \*(t+1) = \*i;
8. \*i = t;
9. }
10. }
11. }
12. }

(2)

1. .pos 0
2. irmovq stack,%rsp
3. call main
4. halt
6. .align 8
7. data:
8. .quad 0x0000000000000004
9. .quad 0x0000000000000003
10. .quad 0x0000000000000002
12. data\_end:
13. .quad 0x0000000000000001
15. main:
16. irmovq data,%rdi
17. call ysBubbleP
18. ret
20. ysBubbleP:
21. jmp L2
22. L4:
23. mrmovq 8(%rax),%r9
24. mrmovq (%rax),%r10
25. rrmovq %r9,%r8
26. subq %r10,%r8
27. jge L3
28. rmmovq %r10, 8(%rax)
29. rmmovq %r9,(%rax)
30. L3:
31. irrmovq %8,%r8
32. addq %r8,%rax
33. jmp L5
34. L6:
35. rrmovq %rdi,%rax
36. L5:
37. rrmovq %rsi,%r8
38. subq %rax,%r8
39. jg L4
40. irmovq $8,%r8
41. subq %r8,%rsi
42. L2:
43. rrmovq %rsi,%r8
44. subq %rdi, %r8
45. jg L6
46. ret
48. .pos 0x200
49. stack:

**4.51**

|  |  |
| --- | --- |
| 取指 | icode:ifun = M1[PC]; rA:rB = M1[PC+1]; valC = M8[PC+2]; valP = PC + 10; |
| 译码 | **valB = R[rB]** |
| 执行 | **ValE = valB + valC; set CC** |
| 访存 | **/** |
| 写回 | **R[rb] = valE** |
| 更新PC | **PC = valP** |

**4.55**

1.预测错误情况：

将(E\_icode == IJXX && !e\_Cnd)

改为(E\_icode == IJXX && E\_ifun != UNCOND && e\_Cnd)

2.修改跳转，对所有的jxx操作

将M\_icode == IJXX && !M\_Cnd : M\_valA;

改为M\_icode == IJXX && M\_ifun != UNCOND && M\_Cnd : M\_valA;

**4.59**

4.47的性能更好.

50%的jge指令会跳转成功，对于跳转成功的循环会执行9条指令，失败的会执行5条指令，平均每个循环执行7条指令。

4.48无jge，每个循环执行9条，

4．49无jge，每个循环执行11条。