# 4.47：

A：

void bubble\_b(long \*data, long count)

{

long i, last;

for (last = count - 1; last > 0; last--)

{

for (i = 0; i < last; i++)

{

if (\*(data + i + 1)< \*(data + i))

{

long t = \*(data + i + 1);

\*(data + i + 1) = \*(data + i);

\*(data + i) = t;

}

}

}

}

B：

# Execution begins at address 0

.pos 0

irmovq stack, %rsp # Set up stack pointer

call main # Execute main program

halt # Terminate program

# Array of 8 elements

.align 8

data:

.quad 0xa000a000a000

.quad 0x0b000b000b00

.quad 0x00c000c000c0

.quad 0x000d000d000d

main: irmovq data,%rdi

irmovq $4,%rsi

call bubble\_p # bubble\_p(data, 8)

ret

# void bubble\_p(long \*data, long count)

# data in %rdi, count in %rsi

bubble\_p:

rrmovq %rsi, %r9

irmovq $1, %r11

subq %r11, %r9

jmp L2

L4:

rrmovq %rdi, %rdx

rrmovq %rax, %rcx

irmovq $8, %r10

loop:

subq %r11, %rcx

jl end\_loop

addq %r10, %rdx

jmp loop

end\_loop:

mrmovq (%rdx), %r8

rrmovq %rdx, %rsi

addq %r10, %rsi

mrmovq (%rsi), %rcx

rrmovq %rcx, %r10

subq %r8, %r10

jge L3

rmmovq %r8, (%rsi)

rmmovq %rcx, (%rdx)

L3:

addq %r11, %rax

jmp L5

L6:

xorq %rax, %rax

L5:

rrmovq %rax, %r10

subq %r9, %r10

jl L4

subq %r11, %r9

L2:

jg L6

ret

# Stack starts here and grows to lower addresses

.pos 0x200

stack:

# 4.51：

iaddq V, rB

取址:

​ 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

访存:

写回:

​ R[rB] <-- valE

PC 更新:

​ PC <-- valP

# 5.15：

void inner(vec\_ptr u, vec\_ptr v, data\_t \*dest)

{

long i;

long length = vec\_length(u);

long limit = length - 5;

data\_t \*udata = get\_ver\_start(u);

data\_t \*vdata = get\_vec\_start(v);

data\_t sum0 = (data\_t)0;

data\_t sum1 = (data\_t)0;

data\_t sum2 = (data\_t)0;

data\_t sum3 = (data\_t)0;

data\_t sum4 = (data\_t)0;

data\_t sum5 = (data\_t)0;

for (i = 0; i < limit; i += 6)

{

sum0 = sum0 + udata[i] \* vdata[i];

sum1 = sum1 + udata[i + 1] \* vdata[i + 1];

sum2 = sum2 + udata[i + 2] \* vdata[i + 2];

sum3 = sum3 + udata[i + 3] \* vdata[i + 3];

sum4 = sum4 + udata[i + 4] \* vdata[i + 4];

sum5 = sum5 + udata[i + 5] \* vdata[i + 5];

}

for (; i < length; i++)

{

sum0 = sum0 + udata[i] \* vdata[i];

}

\*dest = sum0 + sum1 + sum2 + sum3 + sum4 + sum5;

}

原因：只有两个加载单元，一个时钟周期只能加载两个值，CPE 最低只能到 1.00。

# 5.19：

void psum1(float a[], float p[], long n)

{

long i;

float tmp, tmp1, tmp2, tmp3 = 0;

for (i = 0; i < n - 3; i += 4)

{

tmp = tmp3 + a[i];

tmp1 = tmp + a[i + 1];

tmp2 = tmp1 + a[i + 2];

tmp3 = tmp2 + a[i + 3];

p[i] = tmp;

p[i + 1] = tmp1;

p[i + 2] = tmp2;

p[i + 3] = tmp3;

}

for (; i < n; i++)

{

tmp3 += a[i];

p[i] = tmp3;

}

}