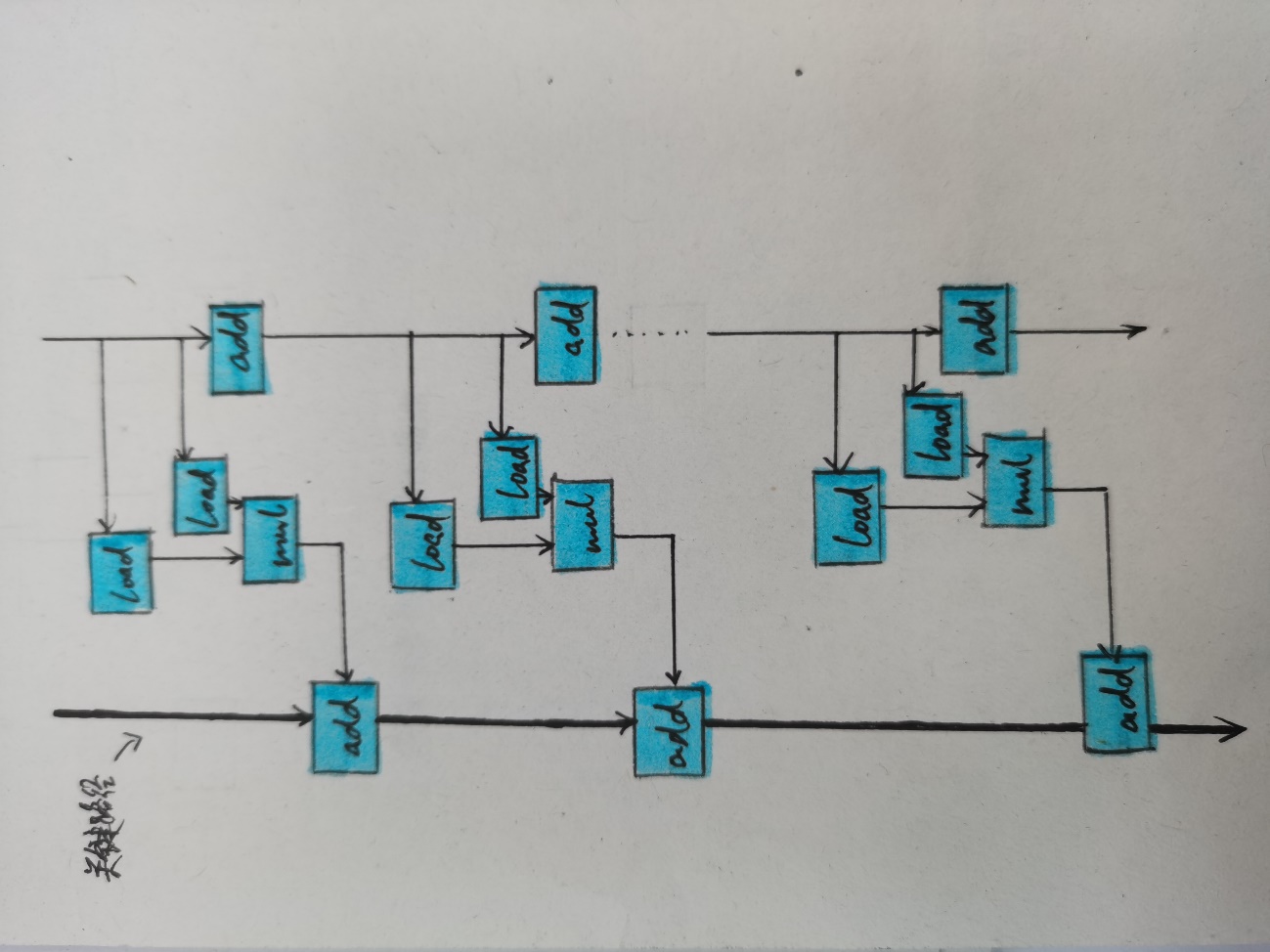
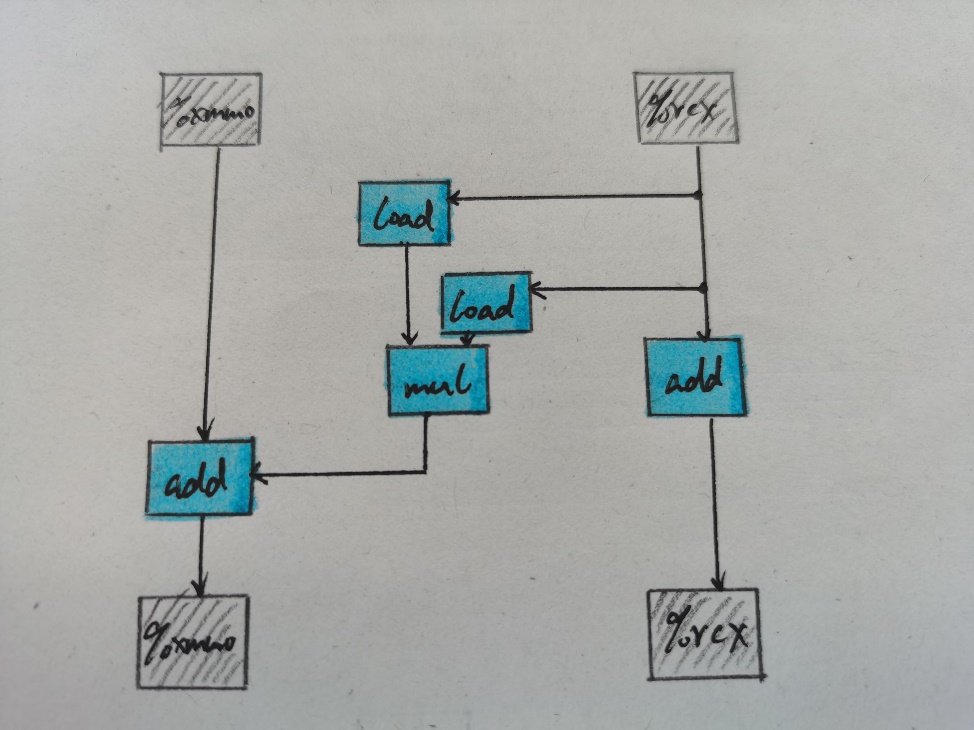
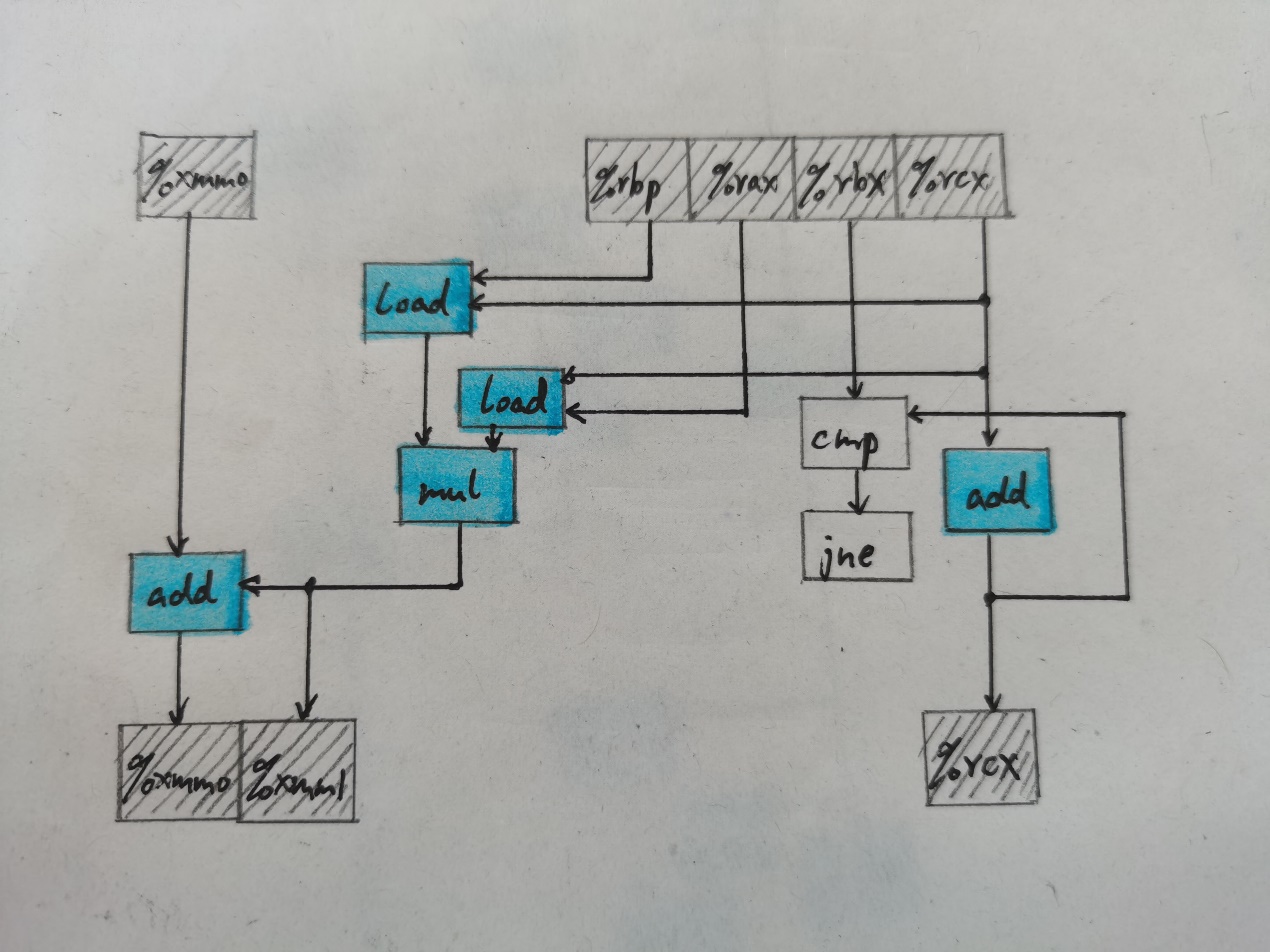
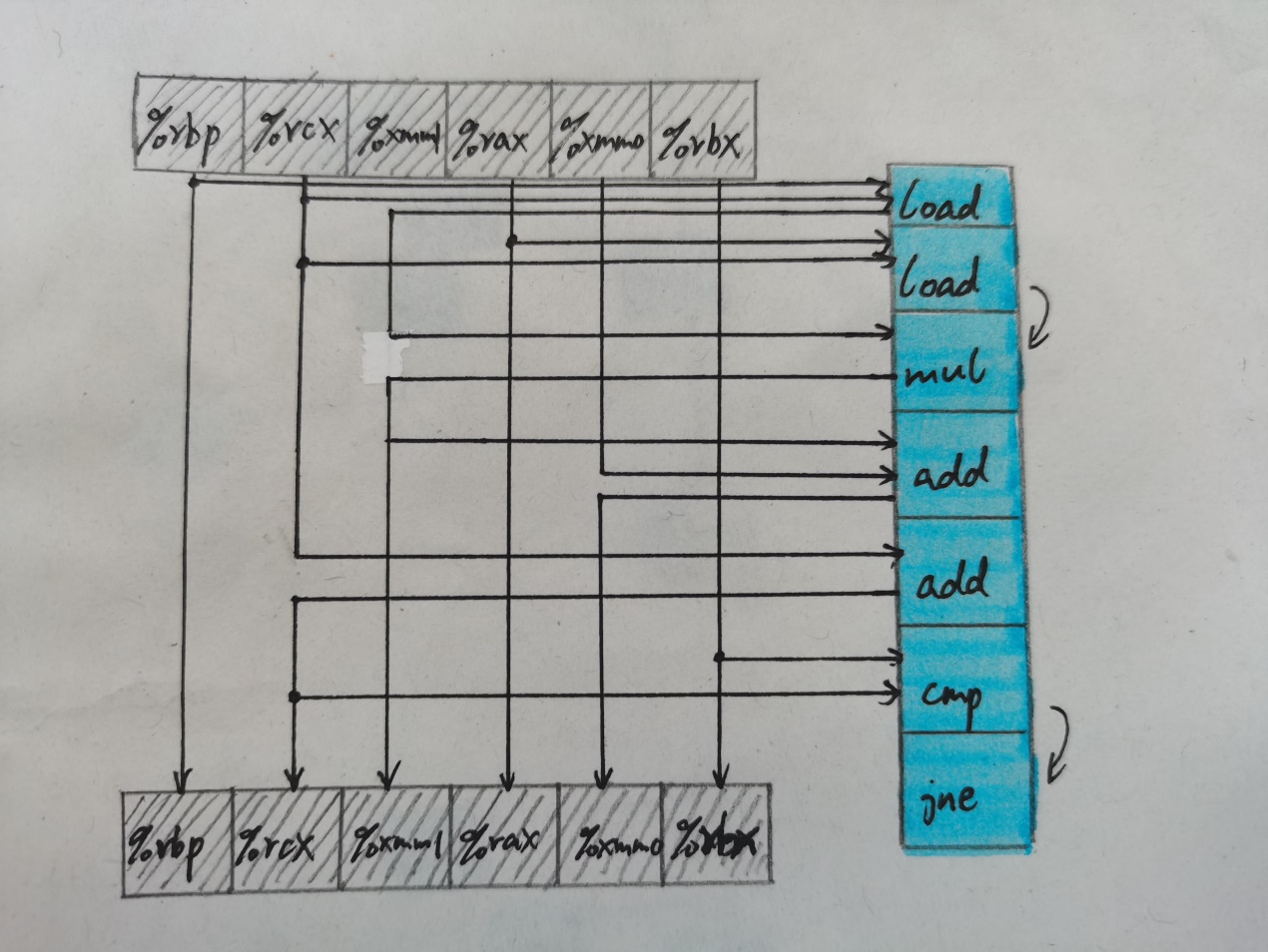
5.13

A.



B.浮点数加法延迟为3个周期，故CPE下界为3.00

C.整数加法延迟为1个周期，故CPE下界为1.00

D.浮点数乘法不在关键路径上，关键路径上为浮点数加法，而浮点数加法延迟为3个周期，导致CPE下界为3.00

5.15

按6×6循环展开：

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\_vec\_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)hjm

{

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;

}

限制因素可能是寄存器溢出，或者分支预测错误后受到处罚。

5.17

void \*my\_memset(void \*s, int c, size\_t n)

{

size\_t K = sizeof(unsigned long);

size\_t cnt = 0;

unsigned long x;

unsigned char \*px = (unsigned char \*)&x;

while(cnt < K)

{

\*px++ = (unsigned char) c;

cnt++;

}

unsigned char \*schar = s;

for (cnt = 0; (size\_t)schar % K != 0 && cnt < n; cnt++)

{

\*schar++ = (unsigned char)c;

}

size\_t limit = n - K + 1;

for (; cnt < limit && (int)limit > 0; cnt += K)

{

\*(unsigned long \*)schar = x;

schar += K;

}

for (; cnt < n; cnt++)

{

\*schar++ = (unsigned char)c;

}

return s;

}

5.19

3\*1a循环展开，重新结合的代码如下：

void faster\_psum\_3\_1a(float a[], float p[], long n)

{

long i;

float val = 0;

float tmp1, tmp2, tmp3;

for (i = 0; i < n - 2; i += 3)

{

tmp1 = val + a[i];

tmp2 = tmp1 + a[i+1];

tmp3 = tmp2 + a[i+2];

p[i] = tmp1;

p[i+1] = tmp2;

p[i+2] = tmp3;

val = tmp3;

}

for (; i < n; i++)

{

val += a[i];

p[i] = val;

}

}