# Pg316

## 3.

1)

$$512MB / 64MB = 8$$

每个内存条需要8个DRAM芯片.

2)

$$2GB = 2 \times 1024MB = 2048MB$$

$$2GB / 512MB = 4$$

需要4个内存条

3)

4GB = $2^{12}$MB = $2^{22}$KB = $2^{32}$B

64M × 8B = $2^{19}$K = $2^{29}$B

主存地址共32位. 其中低29位用作DRAM芯片内地址, 片内地址有29位, 第1~3位用于选择芯片的位平面（设地址最低位为第1位）,第4~16位为片内列地址，第17~29位为片内行地址.

## 5.

磁盘旋转一圈所需的时间为

$$1000ms / 7200RPM / 60 \approx 8.33ms$$

故平均等待时间为

$$8.33ms / 2 = 4.165ms$$

一个4KB数据块的传输时间为

$$1000ms \times 4KB / 40MB \approx 0.0976ms$$

一个4KB数据块的平均读取/写入时间为

$$2ms + 10ms + 0.0976ms + 4.165ms \approx 16.27ms$$

程序对数据块的处理时间为

$$1000ms \times 20000 \times 1 / 500MHz = 0.04ms$$

对该数据块操作的总时长为

$$16.27ms + 0.04ms = 32.58ms$$

每秒可以完成该操作的次数为

$$1000ms \ / \ 32.58ms \ \approx \ 30$$

## 8.

1)

1GB = $2^{30}$B

cache_size = 64KB = $2^{16}$B

block_size = 128B = $2^7$

主存地址有30位,记作$M_{29}M_{28}M_{27}M_{26}...M_3M_2M_1M_0$, 其中, $M_0$~$M_6$为块内偏移量, $M_7$~$M_{15}$为cache行号, $M_{16}$~$M_{29}$为cache标记位.

2)

由于采用直接直接映射法和直写法, 每个cache行的储空间分配为: 有效位:1, 标记位:14, 数据:128×8.

cache行大小为

$$1 \ + \ 14 \ + \ 128 \ \times \ 8 \ = \ 1039 \ bit$$

cahe总容量为

$$1039bit \ \times \ 512 \ = \ 531968bit$$

## 12.

1)

该程序段对数组x[], y[]均按空间存放顺序访问, 空间局部性较好. 由于每个数组元素在该程序段中只被访问一次, 没有时间局部性. 由于没有给出cache容量、块大小、映射/回写方式, 无法判断命中率高低.

2)

block_size = $2^4$B = 4 × sizeof(float)

cache_size = $2^5$B

=> cache_line = cache_size / block_size = 2

0x8049040 % cache_size = 1000 0000 0100 1001 0000 0100 0000 % 0000 0000 0000 0000 0000 0001 0000 = 0

每个块能够存放4个`float`变量, 根据数组x[]首地址`0x8049040`对应的cache行号为0, x[0]~x[3]映射到cache_line[0], x[4]~x[7]映射到cache_line[1], 由于数组y[]的存储空间紧跟在x[]之后, y[0]~y[3]映射到cache_line[0], y[4]~y[7]映射到cache_line[1].

由于该程序段每次执行`sum += x[i] * y[i]`语句时, x[i], y[i]总是映射到cache同一行，导致冲突, 即每次访问都需要加载x[i]和y[i]所在的块, 故命中率为0.

3)

cache改为2路组相联后, `block_size = 8B`, `cache_line = 4`, cache分为2组，每组2行, 每行能够存放2个float变量.

x[0],x[1]映射到cache_line[0][], x[2],x[3]映射到cache_line[1][], x[4],x[5]映射到cache_line[0][], x[6],x[7]映射到cache_line[1][];

y[0],y[1]映射到cache_line[0][], y[2],y[3]映射到cache_line[1][], y[4],y[5]映射到cache_line[0][], y[6],y[7]映射到cache_line[1][];

由于该程序段每次执行`sum += x[i] * y[i]`语句时加载x[i]和y[i]所在的块时可以将之映射到同行不同组的cache中, 故命中率为50%.

4)

数组x[]改为拥有12个元素后, x[0]~x[3]映射到cache_line[0], x[4]~x[7]映射到cache_line[1], x[8]~x[11]映射到cache_line[0], 由于数组y[]的存储空间紧跟在x[]之后, y[0]~y[3]映射到cache_line[1], y[4]~y[7]映射到cache_line[0].

该程序段每次执行`sum += x[i] * y[i]`语句时, x[i], y[i]总是映射到cache不同行, 故命中率为75%.

## 23.

1)

指令I对应的汇编形式为

```
addl (%edx, %ecx, 4), %eax
```

寻址方式为基址加比例变址加偏移量.

2)

当`i = 50`时, 取指令操作过程中MMU得到的指令的线性地址是`laddr = 0x8049c08 + 0 = 0x8049c08`.

操作数的线性地址为`0x0 + 0x804d000+0x50 * 4 = 0x804d0c8`.

3)

手写循环语句S对应的指令序列:

```
movl    $0x0, %ecx
.FOR
cmpl    %ebx, %ecx
jge     .EndFOR
addl    (%edx, %ecx, 4), %eax
addl    $0x1, %ecx
jmp     .FOR
.EndFOR
```

GCC生成的目标代码:

```
movl   $0xa,-0x4(%ebp)      //this is N = 10
movl   $0x0,-0x8(%ebp)      //this is i
jmp    11f3 <main+0x46>
lea    0x84(%eax),%edx
mov    -0x8(%ebp),%ecx
mov    (%edx,%ecx,4),%ecx
lea    0x64(%eax),%edx
mov    (%edx),%edx
add    %edx,%ecx            //this is sum += a[i]
lea    0x64(%eax),%edx
mov    %ecx,(%edx)
addl   $0x1,-0x8(%ebp)      //i++
cmpl   $0x9,-0x8(%ebp)
```

```
jle    11d1 <main+0x24>
```

注意到GCC对无优化要求生成的目标代码中将N, i, sum变量存放在栈中.

4)

PE = PG = 1

5)

指令I在第一次执行时, 取指令时不会发生缺页异常. 因为指令I不在页面起始处, 在执行指令I前面的指令发生缺页时, 会将指令I一并载入内存. 但是取数组a[]的第一个元素a[0]时可能发生缺页异常. 因为a[]的地址和指令I的地址不在同一页中, 可能数组a[]所在的页之前从未被访问过, 造成缺页.

如果发生缺页异常, 由页大小为4KB, 页故障的线性地址为`0x804d000`, 该地址会保存在`CR2`寄存器中.

6)

指令I的线性地址是`0x8049c08`.

由于页大小为4KB($2^{12}$B), 线性地址的低12位表示页内偏移量, 高20位为虚页号, 虚页号的高10位为页目录索引, 低10位为页表项索引. 故指令I所在页的虚页号是`0000 1000 0000 0100 1001`, 页目录索引为`0000 1000 00`, 页表索引为`00 0100 1001`, 页内偏移量为`1100 0000 1000`.

第一次执行指令I时, `P = 1`, `R/W =0`, `U/S = 1`, `A = 1`, `D = 0`.

7)

类似5）中的回答，指令I在第一次执行时, 取指令时不会发生TLB缺失. 因为指令I不在页面起始处, 在执行指令I前面的指令发生TLB缺失时, 会将指令I所在页的页表项载入TLB.

但是取数组a[]的第一个元素a[0]时可能发生缺页异常. 因为a[]的地址和指令I的地址不在同一页中, 可能数组a[]所在的页之前从未被访问过, 造成TLB缺失.

虚页号的低2位为TLB组索引, 高18位为TLB标记.

指令I地址`0x8049c08`对应的TLB组索引为`0x01`,TLB标记为`0x2012`, TLB表第1组中有对应标记的表项, TLB命中, 页框号为`0x028b0`, 主存地址为`0x28b0c08`.

7)

cache_size = 8KB = $2^{13}$B

block_size = 32B = $2^{5}$B

group_line = 2

group_num = cache_size / (block_size * group_line) = $2^{7}$

主存地址低5位为块内偏移量, 中间7位为组索引, 高20位为标记位.

指令I的线性地址为`0x8049c08`, 组索引为`1100 000`, 块内偏移量为`0 1000`, 注意到指令I地址不在主存块的起始位置, 故在第一次执行指令I时, 指令I的地址已经在之前的指令执行时装入对应的cache行中, 不会发生cache缺失. 指令I映射到cache的第`1100 000`组中（第96组）

9)

`N = 2000`时, 数组a[]的大小为`4B * 2000 = 8000B = 7.8125KB`, 占用`7.8125 / 4 = 1.953125`个页. 由于a[]的起始地址为`0x804d000`, 位于虚页号是`0x804d`的第`0x000`个字节处. 故数组a[]将占用2个页面, 虚页号为`0x804d`, `0x804e`.

数组元素a[1200]相对于a[0]的偏移量为`1200 * 4B = 4800B`, 即`4800 / 1024 / 4 = 1.171875`个页, 故a[1200]在a[0]所在页的下一页.