3.

(1) 512/64 = 8片

(2) $2\times2^{20}/512 = 4$个

(3) 因为按字节编址，所以主存地址为32位。

∵ 传输宽度为64位，且共有8片芯片，每片有8位个位平面

∴ 可以同时读 64位 数据。

∵ 每个内存条容量为 512MB = $2^9\times2^{20}\times2^{11}$ b

∴ 内存条内存储单元的地址有 29位。

∵ 有8片芯片，∴ 3位用于选择芯片

其余 26位，13为DRAM芯片内的行地址

13位为列地址。

假设内存地址为 $A_{31}A_{30}\cdots A_0$。

则 $A_{28}\sim A_3$ 为芯片内地址

$A_{28}\sim A_{16}$ 为行地址

$A_{15}\sim A_3$ 为列地址

$A_2\sim A_0$ 表示芯片号。

5. 7200/60 = 120 转/秒。

120/1000 = 0.12 转/毫秒

∴ 转一圈需 $\frac{1}{0.12}\approx 8.33$ ms

半圈，即平均旋转等待时间为 4.17ms

一个数据块的传输时间为 $4/(400\times2^{10}/1000) = 0.0977$ ms

∴ 数据块的平均读取或写回的时间为

2 + 10 + 4.17 + 0.0977 = 16.27 ms

数据块的处理时间为

$(20000/(500\times2^{20}))\times1000 = 0.381$ ms

∴ 完成一次的时间为 $16.27\times2 + 0.381 = 32.92$ ms

每秒可执行 1000/32.92 ≈ 30次

7. (a) int a[N][M]

(1)
```
for(j=0; j<M; j++)
  for(i=0; i<N; i++)
    sum+= a[i][j];
```

(2)
```
for(j=0; j<M; j++)
  for(i=0; i<N; i++)
  { sum+= a[i][j];
    sum*= a[i][j];
  }
```

(3)
```
for(i=0; i<N; i++)
  for(j=0; j<M; j++)
    sum+= a[i][j];
```

(4)
```
for(i=0; i<N; i++)
  for(j=0; j<M; j++)
  { sum+= a[i][j];
    sum*= a[i][j];
  }
```

8. (1) ∵ 主存地址空间大小为 1GB = 2^{30} B

∴ 主存地址30位

$64\times2^{10}/128 = 2^9$ 行

∴ cache行号占9位，块大小为 128B = 2^7 B，块内地址占7位

∴ 标记位为14位。

14位	9位	7位
标记	行号	块内地址

(2) 因为是直接映射，所以不考虑替换算法，所以无用于替换的控制位；因为是全写方式，所以无修改位。

∴ 每个cache行包含 1位有效位、14位标记位、128B数据

总容量为 $2^9\times(1+14+128\times8)$ bit = 531968 bit

12. (1)

x和y都按存储顺序访问，所以空间局部性都好。

每个元素都只被访问了一次，所以无时间局部性。

因为无cache具体信息，所以无法推断命中率。

(2) cache共有32/16 = 2行

x存放在主存中以H开始的单元中，所以

x[0]~x[3]在第4块，x[4]~x[7]在第5块

y[0]~y[3]在第6块，y[4]~y[7]在第7块

x[0]~x[3]和y[0]~y[3]映射在第0行

x[4]~x[7]和y[4]~y[7]映射在第1行

∴访问x[i]时无法访问到y[i]，命中率为0

(3) cache共有32/8 = 4行

∴共两组，每组两行

∴x[0]~x[1]在第8块…

y[0]~y[1]在第12块…

第8块在第0组，第9块在第1组第0行。

10 0 | 11 1 |

∴x[i]、y[i]可以放在同一组的不同行中

∴命中率为50%。

(4) x[8]~x[11]在第6块，y[0]~y[3]在第7块

y[4]~y[7]在第8块。

y[0]~y[3]映射在第1行，y[4]~y[7]在第0行

∴命中率为75%

访问x[0]时，无命中，将x[0]~x[3]装入cache的第0行

访问y[0]时，无命中，将y[0]~y[3]装入cache的第1行。

访问x[1]~x[3]和y[1]~y[3]时都命中

∴为75%。

19. 256/32 = 8行

(1) 直接映射：

① s=64

访问元素为a[0]、a[64]，共10000次。

两者相差64×4 = 256B。

∴缺失率为100%

② s=63

访问元素为a[0]，a[63]，a[126]

a[0]在第0块，a[63]在第7块

a[126]在第15块

分别在第0、7、7行。

∴缺失率为$\frac{2}{3}$

(2) 2路组相联映射。

cache有2^2组，每组1行。

① s=64

a[0]在第0块，a[64]在第8块。

∴会放在同一组的不同行上。

∴缺失率为1‰

② s=63

a[0]在第0组，a[63]和a[126]在同一组

∴缺失率为3‰

21.

(1) 页面大小为 $128B = 2^7 B$

∴ 低7位为页内偏移量

高9位为虚拟页号

由图可知有4组

∴ 虚拟页号中低2位为TLB组索引

高7位为TLB标记.

(2) 物理地址中低7位为页内偏移量.

高5位为物理页号

(3) 块大小为 $4B = 2^2 B$

共16行 = 四行

∴ 低2位为块内地址

中间4位为行索引，高6位为标记.

(4)

067AH = 0000 0110 0111 1010 B

① 虚拟页号为 00000110，映射到TLB第0组，在TLB中找到了，但有效位为0，所以缺失.

② 在慢表中找虚页号为 0CH 的，其有效位为1，得到页框号 19H = 11001B

③ 由上面两步可得物理地址为 11001 11010B

得到 cache 行号为14行，有效位为1，

标记为 33H = 110011B，与物理地址前6位相同.

∴ cache命中.

块内地址为10，取出字节2中的内容 4AH

∴ 变量的值是 $74_{(10)}$

23.

(1)
```
movl   $0, %ecx
.Loop
   cmpl   %ebx, %ecx
   jge    .EXIT
   addl   (%edx, %ecx, 4), %eax
   incl   %ecx
   jmp    .Loop
.EXIT
```

(2) 在执行到程序P时，已经是保护模式并采用分页虚拟管理方式，所以 PE = 1 (保护模式)

PG = 1 (启用分页)

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

寻址方式是 基址+比例变址+偏移量

(4) 表2在P305

① 取指时，MMU根据CS段寄存器对应的描述符cache中的信息进行逻辑地址到线性地址的转换，因为是用户程序，所以看用户代码段的信息.

访问段时对当前特权级的最低等级要求 DPL = 当前特权级 CPL = 3，所以可以访问.

MMU将CS对应的段描述符中的基地址与指令地址相加，得到线性地址为 0 + 0x8048c08 = 0x8049c08

② 取数时，MMU先根据DS对应的段描述符得到DPL，若 DPL ≤ CPL 才可转换地址，此时 DPL = CPL.

所以MMU将DS对应的段描述符中的基地址与操作数有效地址相加得到线性地址

0 + R[edx] + R[ecx] × 4 = 0x8049300 + 50 × 4 = 0x8049c8

(4) 续 图在P307

指令I的线性地址为 0x8049c08

低12位：1100 0000 1000 为页内偏移量

高20位：0000 1000 0000 0100 1001 为虚页号

其中高10位：0000 1000 00 为页目录索引

低10位：00 0100 1001 为页表索引

页目录项的地址为：页目录表首地址 + 页目录索引 << 2

= 0x3d000000 + 0x080

= 0x3d000080

页表项地址 = 页表首地址 + 页表索引 << 2

= 0x5c800000 + 0x124

= 0x5c800124

指令I的物理地址 = 页表项地址 + 页内偏移量

= 0x5c800124 + 0xc08

= 0x5c8d2c

第一次执行指令I时，对应页表项中：

P=1（页在主存中）

R/W=0（只读）

U/S=1（可访问）

A=1（该页已被访问）

D=0（代码页不会被修改）

(5) 不会发生缺页异常，因为指令I不在页面起始处，若发生缺页异常，页故障的线性地址（可能是）0x804a000 它正好位于一个页面的起始位置（页大小为4KB，所以页起始地址的后12位为0），该地址保存在控制寄存器 CR2 中。

(6) 不会，因为指令I不在页面起始处。

因为有16个表项，2^2路组相联方式。

所以共有 $2^4/2^2 = 4$ 组

∴ 20位虚拟页号中 高18位为TLB标记，低2位为TLB组索引

第一次执行指令I时，组索引为01，标记为 02024H，在TLB中找到相应表项且有效位为1，得到页框为028B0H，再与页内偏移量相加得到主存地址：0x28b0c08。

(7) cache共有 $8\times2^{10}/32 = 256$ 行

有 256/2 = 128 组。

所以主存地址划分为：高20位为标记，中间7位为组索引，低5位为块内地址。

指令I主存地址为：0x28b0c08

∴ 在cache的 $96H = 96_{(10)}$ 组。

块内地址为 01000，显然不在主存块的起始位置，所以不会发生cache缺失。

(8) 数组a共占 8000B，线性地址为段基址 + 首地址 = 0x8049300，

其中 a[0]~a[831] 在一个页面中

其余 2000-832 = 1168 个元素需要 $1168\times4/4K\times2^{10}$

= 1.14 个页面。

所以占用了3个页面。

虚页号分别为 0000 1000 0000 0100 1001

0000 1000 0000 0101 1010

0000 1000 0000 0101 1011

a[1200] 在第2个页中。