一、

- 某计算机的存储系统由Cache和主存组成。若所访问的字在Cache中，则存取它需要10ns；将所访问的字从主存装入Cache需要60ns。假定Cache的命中率为0.9，计算该存储系统访问一个字的平均存取时间。
  - 平均存取时间： 10ns×0.9 + (60ns+10ns)×(1-0.9) = 16ns

二、

- 假设一4路组相联Cache，数据存储空间大小为64KB，块大小为16字节，主存地址32位，主存一个字包含4个字节，Cache采用写回策略，每个数据块包括1位有效位，Cache每个字用1位脏位来表示是否被修改。
  - (1)CPU如何解释主存地址（主存地址格式）
  - (2)计算实现该Cache所需总存储容量

二、

Cache与主存组数一样，地址沿组方向增长

- 假设一4路组相联Cache，数据存储空间大小为64KB，块大小为16字节，主存地址32位，主存一个字包含4个字节，Cache采用写回策略，每个数据块包括1位有效位，Cache每个字用1位脏位来表示是否被修改。
  - (1)CPU如何解释主存地址（主存地址格式）
    - 主存容量：2^{32} = 4G Bytes
    - Cache容量：64K Bytes
    - 块(Block)大小：16 Bytes
    - Way：4 ways (Cache每组含4个Block)
    - Cache组数：64KB/(16B×4)=2^{10} = 1024组
    - 主存每组块数 = 4G Bytes/(16Bytes×1024组) = 2^{18}块/组
    - 主存地址：32位，高18位为组内块地址，中间10位为组地址，低4位为块内地址
    - Cache的Tag：23位 = 1位有效位+4位脏位(4个字)+18位组内块地址

18	10	4
组内块地址 (tag)	组地址	块内偏移

二、

- 假设一4路组相联Cache，数据存储空间大小为64KB，块大小为16字节，主存地址32位，主存一个字包含4个字节，Cache采用写回策略，每个数据块包括1位有效位，Cache每个字用1位脏位来表示是否被修改。
  - (2)计算实现该Cache所需总存储容量
    - Cache的Tag：23位 = 1位有效位+4位脏位(4个字)+18位组内块地址
    - 每Cache行组成：23位Tag+128位数据(16 Bytes)
    - 实现Cache的总存储容量：(23 + 128)×(64K/16) = 604K bits = 75.5K Bytes

18	10	4
组内块地址 (tag)	组地址	块内偏移

三、

- 计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍（与“Cache速度比主存快10倍”的区别？），采用LRU替换算法，假定块替换的时间忽略不计。
  - (1)计算上述取数过程的命中率
  - (2)计算采用Cache后的加速比

三、

- 计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计。
  - 主存容量32K字，Cache容量4K字
  - 块大小＝64字
  - 组内Cache块数＝4块
  - 组数＝4K/(64×4)＝16
  - 主存块数＝32K/64＝512
  - 主存块/组＝512/16＝32

Cache与主存组数一样，地址沿组方向增长

三、

- 计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计。
  - 4352个主存字分配在68个主存块中（4352/64=68）
    - 主存块0：主存字0~63
    - 主存块1：主存字64~127
    - ……
    - 主存块63：主存字4032~4095
    - 主存块64：主存字4096~4159
    - 主存块65：主存字4160~4223
    - 主存块66：主存字4224~4287
    - 主存块67：主存字4288~4351

三、

- 计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计。
  - 主存块⇔ Cache组
  - 68个主存块
  - 16个Cache组
  - 每Cache组4块

Cache组	主存块
0	0，16，32，48，64
1	1，17，33，49，65
2	2，18，34，50，66
3	3，19，35，51，67
4	4，20，36，52
…	…
13	13，29，45，61
14	14，30，46，62
15	15，31，47，63

三、

- **初始：**

组	块0	块1	块2	块3
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				

三、

➢ **第1次循环：块0~块63**

组	块0	块1	块2	块3
0	0	16	32	48
1	1	17	33	49
2	2	18	34	50
3	3	19	35	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23		55
8	8	24		56
9	9	25		57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

块64~67?

三、

Note：LRU算法替换最长时间不用的块

➢ **第1次循环：块64~块67**

组	块0	块1	块2	块3
0	0➔**64**	16	32	48
1	1➔**65**	17	33	49
2	2➔**66**	18	34	50
3	3➔**67**	19	35	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

块0~3?

三、

➢ **第2次循环：块0~块3**

组	块0	块1	块2	块3
0	64	16➔**0**	32	48
1	65	17➔**1**	33	49
2	66	18➔**2**	34	50
3	67	19➔**3**	35	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

三、

➢ **第2次循环：块4~块15**

组	块0	块1	块2	块3
0	64	16➔**0**	32	48
1	65	17➔**1**	33	49
2	66	18➔**2**	34	50
3	67	19➔**3**	35	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

三、

➢ **第2次循环：块16~块19**

组	块0	块1	块2	块3
0	64	16➔**0**	32➔**16**	48
1	65	17➔**1**	33➔**17**	49
2	66	18➔**2**	34➔**18**	50
3	67	19➔**3**	35➔**19**	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

三、

➢ **第2次循环：块20~块31**

组	块0	块1	块2	块3
0	64	16➔**0**	32➔**16**	48
1	65	17➔**1**	33➔**17**	49
2	66	18➔**2**	34➔**18**	50
3	67	19➔**3**	35➔**19**	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

三、

➢ **第2次循环：块32~块35**

组	块0	块1	块2	块3
0	64	16➔**0**	32➔**16**	48➔**32**
1	65	17➔**1**	33➔**17**	49➔**33**
2	66	18➔**2**	34➔**18**	50➔**34**
3	67	19➔**3**	35➔**19**	51➔**35**
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

三、

➢ **第2次循环：块36~块47**

组	块0	块1	块2	块3
0	64	16➔**0**	32➔**16**	48➔**32**
1	65	17➔**1**	33➔**17**	49➔**33**
2	66	18➔**2**	34➔**18**	50➔**34**
3	67	19➔**3**	35➔**19**	51➔**35**
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

三、

➢ **第2次循环：块48~块51**

组	块0	块1	块2	块3
0	64➔**48**	16➔**0**	32➔**16**	48➔**32**
1	65➔**49**	17➔**1**	33➔**17**	49➔**33**
2	66➔**50**	18➔**2**	34➔**18**	50➔**34**
3	67➔**51**	19➔**3**	35➔**19**	51➔**35**
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

三、

➢ **第2次循环：块52~块63**

组	块0	块1	块2	块3
0	64➔**48**	16➔**0**	32➔**16**	48➔**32**
1	65➔**49**	17➔**1**	33➔**17**	49➔**33**
2	66➔**50**	18➔**2**	34➔**18**	50➔**34**
3	67➔**51**	19➔**3**	35➔**19**	51➔**35**
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

三、

➢ **第2次循环：块64~块67**

组	块0	块1	块2	块3
0	64➔**48**	16➔0➔**64**	32➔**16**	48➔**32**
1	65➔**49**	17➔1➔**65**	33➔**17**	49➔**33**
2	66➔**50**	18➔2➔**66**	34➔**18**	50➔**34**
3	67➔**51**	19➔3➔**67**	35➔**19**	51➔**35**
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

三、

- 规律：
  - 第1次循环 (单位：块)
    - 主存块0~63：未命中
    - 主存块64~67：未命中，替换
    - 因“块替换的时间忽略不计” ➔等价于仅存在未命中情形
    - 未命中次数：68
  - 第2次循环~第10次循环 (单位：块)
    - 映射到组0~3的20个主存块：未命中，替换
    - 主存块：0~3，16~19，32~35，48~51，64~67
    - 其余48个主存块：全部命中
    - 未命中次数：20×9
    - 命中次数：48×9

三、

- (1)计算上述取数过程的命中率
  - 块未命中 vs. 块命中(单位：块)
    - 未命中次数：68＋20×9＝248
    - 命中次数：48×9＝432
  - 若块未命中
    - 1次字未命中，63次字命中
  - 若块命中
    - 64次字命中
  - CPU读存储器总计(单位：字)
    - Cache未命中次数：1×248 = 248
    - Cache命中次数：63×248＋64×432＝43272
  - 命中率：
    - 43272/(248+43272)=99.43%

三、

- (2)计算采用Cache后的加速比
  - 设Cache一次访问(字)时间为T
  - 则主存一次访问(字)时间为10T(Cache速度是主存速度的10倍)
  - 未命中时访问(字)时间为10T+T=11T
  - 加速比=全部主存访问时间/(Cache未命中访问时间+Cache命中访问时间)=(10T×(248+43272))/(11T×248+T×43272)=9.46

四、

- 考虑一个Cache，其存取时间为2.5ns，行大小为64字节，命中率H=0.95。主存使用块传送方式，第一个字（4字节）存取时间为50ns，其后每个字存取时间为5ns。
  - (1)出现一次Cache缺失的存取时间是多少？假设此时Cache等待，直到该行从主存传送到Cache，然后再从Cache读取
  - (2)假设行大小增大到128字节，命中率提升到0.97，是否会降低平均存取时间

四、

- 考虑一个Cache，其存取时间为2.5ns，行大小为64字节，命中率H=0.95。主存使用块传送方式，第一个字（4字节）存取时间为50ns，其后每个字存取时间为5ns。
  - (1)出现一次Cache缺失的存取时间是多少？假设此时Cache等待，直到该行从主存传送到Cache，然后再从Cache读取
    - 行大小为64字节=16字
    - 出现一次Cache缺失的存取时间为：50ns+15×5ns+2.5ns = 127.5ns

四、

- 考虑一个Cache，其存取时间为2.5ns，行大小为64字节，命中率H=0.95。主存使用块传送方式，第一个字（4字节）存取时间为50ns，其后每个字存取时间为5ns。
  - (2)假设行大小增大到128字节，命中率提升到0.97，是否会降低平均存取时间
    - 原条件下，平均存取时间为$T = H \times T_c + (1-H) \times T_m = 0.95 \times 2.5ns + 0.05 \times 127.5ns = 8.75ns$
    - 行大小增加到128字节=32字后，出现一次Cache缺失的存取时间为50ns+31×5ns+2.5ns = 207.5ns，平均存取时间为$T = 0.97 \times 2.5ns + 0.03 \times 207.5ns = 8.65ns$
    - 可见平均存取时间降低了

块大小与缺失率的关系：
#一般而言，增加块大小将降低缺失率(因为空间局部性)，但块大小达到一定程度时，缺失率会随块大小的继续增加而上升(因为块数量下降带来块替换的增加)；
#单纯增加块大小带来缺失代价(缺失损失)的增大。

五、

- 给定一个32位的虚拟地址空间和一个24位的物理地址，对于下面不同的分页大小P，请确定虚拟页号（VPN）、虚拟页内偏移量（VPO）、物理页号（PPN）和物理页内偏移量（PPO）的位数。

P	#VPN位数	#VPO位数	#PPN位数	#PPO位数
1KB	22	10		
2KB	21	11		
4KB	20	12		
8KB	19	13		

五、

- 给定一个32位的虚拟地址空间和一个24位的物理地址，对于下面不同的分页大小P，请确定虚拟页号（VPN）、虚拟页内偏移量（VPO）、物理页号（PPN）和物理页内偏移量（PPO）的位数。

P	#VPN位数	#VPO位数	#PPN位数	#PPO位数
1KB	22	10	14	10
2KB	21	11	13	11
4KB	20	12	12	12
8KB	19	13	11	13

六、

- 假定一个计算机系统中有一个TLB和一个L1 Data Cache。该系统按字节编址，虚拟地址16位，物理地址12位；页大小为128字节，TLB采用4路组相联映射，共有16个页表项；L1 Data Cache采用直接映射方式，块大小为4字节，共16行。在系统运行到某一时刻。TLB、页表和L1 Data Cache中的部分内容（用十六进制表示）如下图所示。

组号	标记	实页号	有效位	标记	实页号	有效位	标记	实页号	有效位	标记	实页号	有效位
0	03	—	0	09	1D	1	00	—	0	07	10	1
1	13	2D	1	02	—	0	04	—	0	0A	—	0
2	02	—	0	08	—	0	06	—	0	03	—	0
3	07	—	0	63	12	1	0A	34	1	72	—	0

(a)TLB内容(4路组相联，4组，16个页表项)

六、

虚页号	实页号	有效位
000	08	1
001	03	1
002	14	1
003	02	1
004	—	0
005	16	1
006	—	0
007	07	1
008	13	1
009	17	1
00A	09	1
00B	—	0
00C	19	1
00D	—	0
00E	11	1
00F	0D	1

行索引	标记	有效位	字节3	字节 2	字节 1	字节 0
0	19	1	12	56	C9	AC
1	—	0	—	—	—	—
2	1B	1	03	45	12	CD
3	—	0	—	—	—	—
4	32	1	23	34	C2	2A
5	0D	1	46	67	23	3D
6	—	0	—	—	—	—
7	10	1	12	54	65	DC
8	24	1	23	62	12	3A
9	—	0	—	—	—	—
A	2D	1	43	62	23	C3
B	—	0	—	—	—	—
C	12	1	76	83	21	35
D	16	1	A3	F4	23	11
E	33	1	2D	4A	45	55
F	—	0	—	—	—	—

➢ **(b)部分页表内容(前16项)**　**(C)L1 Data Cache内容(直接映射，16行，块大小4字节)**

六、

- 请回答下列问题：
  - (1)虚拟地址中哪几位表示虚拟页号、哪几位表示页内偏移量？虚拟页号中哪几位表示TLB标记？哪几位表示TLB组索引？
  - (2)物理地址中哪几位表示物理页号、哪几位表示页内偏移量？在访问Cache时，物理地址如何划分成标记字段、行索引字段和块内地址字段？
  - (3)CPU从地址067AH中取出的值是多少？要求对CPU读取地址067AH中内容的过程进行详细说明。

六、

- 请回答下列问题：
  - (1)虚拟地址中哪几位表示虚拟页号、哪几位表示页内偏移量？虚拟页号中哪几位表示TLB标记？哪几位表示TLB组索引？
    - TLB分为4组，所以TLB组索引为2位
    - 16位虚拟地址中低7位为页内偏移量，高9位为虚页号；虚页号中高7位为TLB标记，低2位为TLB组索引

<table>
<tr><td>7</td><td>2</td><td>7</td></tr>
<tr><td>TLB标记</td><td>TLB组索引</td><td rowspan="2">页内偏移</td></tr>
<tr><td colspan="2">虚页号</td></tr>
</table>

六、

- 请回答下列问题：
  - (2)物理地址中哪几位表示物理页号、哪几位表示页内偏移量？在访问Cache时，物理地址如何划分成标记字段、行索引字段和块内地址字段？
    - Cache有16行，所以Cache行索引为4位
    - 12位物理地址中低7位为页内偏移量，高5位为物理页号。12位物理(主存)地址中，低2位为块内地址，中间4位为Cache行索引，高6位为标记

5	7	
物理页号	页内偏移	
6	4	2
Cache标记	Cache行索引	块内地址

六、

7	2	7
TLB标记	TLB组索引	页内偏移
虚页号		

❖ 请回答下列问题：

➢ (3)CPU从地址067AH中取出的值是多少？要求对CPU读取地址067AH中内容的过程进行详细说明。

- 地址067AH=0000 0110 0111 1010B，所以，虚页号为0000 0110 0B，映射到TLB的第00组
- 将0000 011B=03H与TLB第0组的四个标记比较，虽然和其中一个相等，但对应的有效位为0，其余都不相等，所以TLB缺失，需要访问主存中的页表
- 直接查看0000 0110 0B=00CH处的页表项，有效位为1，取出物理页号19H=1100 1B，和页内偏移111 1010B拼接成物理地址：1100 1111 1010B
- 根据中间4位1110直接找到Cache第14行(即第E行)，有效位为1，且标记为33H=11 0011B，正好等于物理地址高6位，故命中
- 根据物理地址最低两位10，取出字节2中的内容4AH=0100 1010B

6	4	2
Cache标记	Cache行索引	块内地址