

第九章习题2

copyright ICS小班18班lhy

一、单项选择题(36分)

- () 1.对于虚拟存储系统,一次访存过程中,下列命中组合不可能发生的是_____.
- A.TLB 未命中, Cache 未命中, Page 未命中
 - B.TLB 未命中, Cache 命中, Page 命中
 - C.TLB 命中, Cache 未命中, Page 命中
 - D.TLB 命中, Cache 命中, Page 未命中

- () 2.有程序段如下:

```
int foo( ) {  
    char str1[20], *str2;  
    str2 = (char*)malloc(20*sizeof(char)); free(str2);  
}
```

下列说法中正确的是_____.

- A.str1 和 str2 指向的内存都是分配在栈空间内的
- B.str1 和 str2 指向的内存都是分配在堆空间内的
- C.str1 指向的内存是分配在栈空间内的,str2 指向的内存是分配在堆空间内的
- D.str1 指向的内存是分配在堆空间内的,str2 指向的内存是分配在栈空间内的

- () 3.假设有一台 64 位的计算机的物理页块大小是 8KB,采用三级页表进行虚拟地址寻址,它的虚拟地址的 VPO(Virtual Page Offset,虚拟页偏移)有 13 位,问它的虚拟地址的 VPN(Virtual Page Number,虚拟页号码)有多少位?

- A. 20
- B. 27
- C. 30
- D. 33

- () 4.进程 P1 通过 fork()函数产生一个子进程 P2.假设执行 fork()函数之前,进程 P1 占用了 53 个(用户态的)物理页,则 fork 函数之后,进程 P1 和进程 P2 共占用_____个(用户态的)物理页;假设执行 fork()函数之前进程 P1 中有一个可读写的物理页,则执行 fork()函数之后,进程 P1 对该物理页的页表项权限为_____.

- A. 53,读写
- B. 53,只读
- C. 106,读写
- D. 106,只读

- () 5.动态管理器分配策略中,最适合“最佳适配算法”的空白区组织方式是:

- A.按大小递减顺序排列
- B.按大小递增顺序排列
- C.按地址由小到大排列
- D.按地址由大到小排列

()6.Intel 的 IA32 体系结构采用二级页表,称第一级页表为页目录(Page Directory),第二级页表为页表(Page Table).页面的大小为 4KB,页表项 4 字节.以下给出了页目录与若干页表中的部分内容,例如,页目录中的第 1 个项索引到的是页表 3,页表 1 中的第 3 个项索引到的是物理地址中的第 5 个页.则十六进制逻辑地址 8052CB 经过地址转换后形成的物理地址应为十进制的_____.

- A. 21195 B. 29387 C. 21126 D. 47195

页目录		页表1		页表2		页表3	
VPN	页表号	VPN	页号	VPN	页号	VPN	页号
1	3	3	5	2	1	2	9
2	1	4	2	4	4	3	8
3	2	5	7	8	6	5	3

()7.已知某系统页面长 8KB,页表项 4 字节,采用多层分页策略映射 64 位虚拟地址空间.若限定最高层页表占 1 页,则它可以采用多少层的分页策略?

- A.3 层 B.4 层 C.5 层 D.6 层

()8.在 C 语言中实现 Mark-and-Sweep 算法时,可以基于以下哪个假设:(宿主机为 32 位机器)

- A.所有指针指向一个块的起始地址
B.所有指针数据都是 4 字节对齐
C.只需要扫描数据类型为指针的堆中的数据空间
D.只需要扫描所有长度为 4 字节的堆中的数据空间

()9.在 Core i7 中,以下哪个页表项属于 4 级页表项,不属于 1 级页表项:

- A.G 位(Global Bit)
B.D 位(Dirty Bit)
C.XD 位(Disable or enable instruction fetch)
D.U/S 位(User or supervisor mode access permission)

()10.在 Core i7 中,关于虚拟地址和物理地址的说法,不正确的是:

- A.VPO = CI + CO B.PPN = TLBT + TLBI
C.VPN1 = VPN2 = VPN3 = VPN4 D.TLBT + TLBI = VPN

()11.下列与虚拟内存有关的说法中哪些是不对的?

- A.操作系统为每个进程提供一个独立的页表,用于将其虚拟地址空间映射到物理地址空间.
B.MMU 使用页表进行地址翻译时,虚拟地址的虚拟页面偏移与物理地址的物理页面偏移是相同的.
C.若某个进程的工作集大小超出了物理内存的大小,则可能出现抖动现象.
D.动态内存分配管理,采用双向链表组织空闲块,使得首次适配的分配与释放均是空闲块数量的线性时间.

()12.假定整型变量 A 的虚拟地址空间为 0x12345cf0,另一整型变量 B 的虚拟地址 0x12345d98,假定一个 page 的长度为 0x1000 byte,A 的物理地址数值和 B 的物理地址数值关系应该为:

- A.A 的物理地址数值始终大于 B 的物理地址数值
B.A 的物理地址数值始终小于 B 的物理地址数值
C.A 的物理地址数值和 B 的物理地址数值大小取决于动态内存分配策略
D.无法判定两个物理地址值的大小

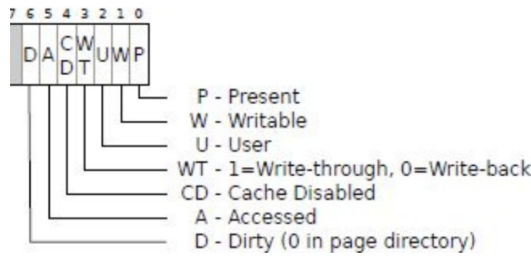
二、非选择题(64分)

13.(24分)

I. (1)在进行地址翻译的过程中,操作系统需要借助页表(Page Table)的帮助.考虑一个 32 位的系统,页大小是 4KB,页表项 (Page Table Entry)大小是 4 字节(Byte),如果不使用多级页表,常驻内存的页表一共需要(1)_____页.

(2)考虑右图已经显示的物理内存分配情况,在二级页表的情况下,已经显示的区域的页表需要占据(2)_____页.

II. IA32 体系采用小端法和二级页表.其中两级页表大小相同,页大小均为 4KB, 结构也相同.TLB 采用直接映射.TLB 和页表每一项的后 7 位含义如下图所示.为简便起见,假设 TLB 和页表每一项的后 8~12 位都是 0 且不会被改变.注意后 7 位值为“27”则表示可读写.



VP0	已分配页
...	
VP1023	
VP1024	
...	
VP2047	未分配页
Gap	
1023 unallocated pages	
VP10239	
VP10240	已分配页
...	
VP11263	
...	

当系统运行到某一时刻时,TLB 内容如下:

索引	TLB 标记	内容	有效位
0	0x04013	0x3312D027	1
1	0x01000	0x24833020	0
2	0x005AE	0x00055004	1
3	0x00402	0x24AEE020	0
4	0x0AA00	0x0005505C	0
5	0x0000A	0x29DEE000	1
6	0x1AE82	0x00A23027	1
7	0x28DFC	0x00023000	0

一级页表的基地址为 0x0C23B00,物理内存中的部分内容如下:

地址	内容	地址	内容	地址	内容	地址	内容
00023000	E0	00023001	BE	00023002	EF	00023003	BE
00023120	83	00023121	C8	00023122	FD	00023123	12
00023200	23	00023201	FD	00023202	BC	00023203	DE
00023320	33	00023321	29	00023322	E5	00023323	D2
0005545C	97	0005545D	C2	0005545E	7B	0005545F	45
00055464	97	00055465	D2	00055466	7B	00055467	45
0C23B020	27	0C23B021	EB	0C23B022	AE	0C23B023	24
0C23B040	27	0C23B041	40	0C23B042	DE	0C23B043	29
0C23B080	05	0C23B081	5D	0C23B082	05	0C23B083	00
2314D200	23	2314D201	12	2314D202	DC	2314D203	0F
2314D220	A9	2314D221	45	2314D222	13	2314D223	D2
29DE404C	27	29DE404D	42	29DE404E	BA	29DE404F	00
29DE4400	D0	29DE4401	5C	29DE4402	B4	29DE4403	2A

此刻,系统先后试图对两个已经缓存在 cache 中的内存地址进行写操作,请分析完成写之后系统的状态(写的地址和上面的内存地址无交集),完成下面的填空.若不需要某次访问或者缺少所需信息,请填“\”.

第一次向地址 0xD7416560 写入内容,TLB 索引为:(3)_____,完成写之后该项 TLB 内容为:

(4)_____,二级页表页表项地址为:(5)_____,物理地址为:(6)_____.

第二次向地址 0x0401369B 写入内容,TLB 索引为:(7)_____,完成写之后该项 TLB 内容为:

(8)_____,二级页表页表项地址为:(9)_____,物理地址为:(10)_____.

III. 下面的程序使用 fork 逼近系统的进程数量上限,请仔细阅读程序并填空.

```
#include <stdio.h>
#include <sys/wait.h>
#include <unistd.h>
#define N 4
int main() {
    volatile int pid, cnt = 1; for (int i =
    0; i < N; i++) {
        if ((pid = fork()) > 0) {
            cnt++;
        }
        while (wait(NULL) > 0); return 0;
    }
```

整个过程中,变量 cnt 最大值是(11)_____.假设所有的数据都已经存在于内存中,pid 和 cnt 在同一个物理页中.从第一个进程开始执行 for 语句开始,此过程对于 cnt 的操作至少会导致页表中(12)_____次虚拟页对应的物理页被修改.

14.(18分) 为了提升虚拟内存地址的转换效率,降低遍历两级页表结构所带来的地址转换开销,英特尔处理器中引入了大页 TLB,即一个 TLB 项可以涵盖整个 4MB 对齐的地址空间(针对 32 位模式).只要设置页目录页中页目录项(PDE)的大页标志位,即可让 MMU 识别这是一个大页 PDE,并加载到大页 TLB 项中.大页 PDE 中记录的物理内存页面号必须是 4MB 对齐的,并且整个连续的 4MB 内存均可统一通过该大页 PDE 进行地址转换.

在 32 位的 Linux 系统中,为了方便访问物理内存,内核将地址 0~768MB 间的物理内存映射到虚拟内存地址 3GB~3GB+768MB 上,并通过大页 PDE 进行该区间的地址转换.任何 0~768MB 的物理内存地址可以直接通过加 3G(0xC0000000)的方式得到其虚拟内存地址.在内核中,除了该区间的内存外,其他地址的内存通常都通过普通的两级页表结构来进行地址转换.

假设在我们使用的处理器中有 2 个大页 TLB 项,其当前状态如下:

索引号	TLB标记	页面号	有效位
0	0xC48	0x04800	1
1	0xC9C	0x09C00	1

有 4 个普通 TLB 项,当前的状态如下:

索引号	TLB标记	页面号	有效位
0	0xF8034	0x04812	1
1	0xF8033	0x09812	1
2	0xF4427	0x12137	1
3	0xF44AE	0x17343	1

当前页活跃的目录页(PD)中的部分 PDE 的内容如下:

PDE索引	页面号	其他标志	大页位	存在位
786	0x04800	...	1	1
807	0x09C00	...	1	1
977	0x09C33	...	0	1
992	0x09078	...	0	1

注:普通页面大小为4KB,并且4KB对齐.每个页面的页面号为其页面起始物理地址除以4096得到.大页由连续1024个4KB小页组成,且4MB对齐.

I.分析下面的指令序列,

```
movl $0xC48012024, %ebx
movl $128, (%ebx)
movl $0xF8034000, %ecx
movl $36(%ecx), %eax
```

请问,执行完上述指令后,eax 寄存器中的内容是(1)_____;在执行上述指令过程中,共发生了(2)____次 TLB miss? 同时会发生(3)____次 page fault? 注:不能确定时填写"--".

II.请判断下列页面号对应的页面中,哪些一定是页表页? 哪些不是? 哪些不确定?

页面号	是否为页表页(是/不是/不确定)
0x04800	(4)_____
0x09C33	(5)_____
0x09812	(6)_____

III.下列虚拟地址中哪一个对应着够将虚拟内存地址 0xF4427048 映射到物理内存地址 0x14321048 的页表项((7)_____)?

(A)0x09C33027 (B)0xC9C3309C
(C)0xC9C33027 (D)0x09C3309C

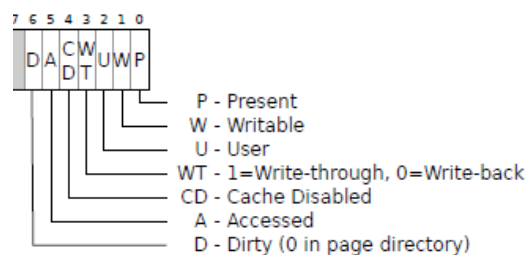
通过上述虚拟地址,利用 movl 指令修改对应的页表项,完成上述映射,在此过程中,是否会产生 TLB miss? ((8)_____)(回答:会/不会/不确定)

修改页表项后,是否可以立即直接使用下面的指令序列将物理内存地址 0x14321048 开始的一个 32 位整数清零? 为什么?

```
movl $0xF4427048, %ebx movl $0, (%ebx)
```

答: (9)_____.

15.(12分)Intel 的 IA32 体系结构采用小端法和二级页表.其中两级页表的大小相同,页大小为 4KB.一级页表和二级页表的表项结构相同,其中页表项后六位的含义如下.已知一级页表的地址为 0x0c23b000,物理内存中的部分内容如下图所示.



地址	内容	地址	内容	地址	内容	地址	内容
00023000	E0	00023001	BE	00023002	EF	00023003	BE
00023120	83	00023121	C8	00023122	FD	00023123	12
00023200	23	00023201	FD	00023202	BC	00023203	DE
00023320	33	00023321	29	00023322	E5	00023323	D2
00023FF8	29	00023FF9	FF	00023FFA	DE	00023FFB	BC
00055004	03	00055005	D0	00055006	74	00055007	89
0005545C	97	0005545D	C2	0005545E	7B	0005545F	45
00055460	97	00055461	D2	00055462	7B	00055463	45
00055464	97	00055465	E2	00055466	7B	00055467	45
0C23B020	55	0C23B021	EB	0C23B022	AE	0C23B023	24
0C23B040	55	0C23B041	AB	0C23B042	2A	0C23B043	01
0C23B080	05	0C23B081	5D	0C23B082	05	0C23B083	00
0C23B09D	05	0C23B09E	D3	0C23B09F	F2	0C23B0A0	0F
0C23B274	05	0C23B275	3D	0C23B276	02	0C23B277	00
0C23B9FC	25	0C23B9FD	D2	0C23B9FE	14	0C23B9FF	23
2314D200	23	2314D201	12	2314D202	DC	2314D203	0F
2314D220	A9	2314D221	45	2314D222	13	2314D223	D2
2314D4A0	BD	2314D4A1	BC	2314D4A2	88	2314D4A3	D3
2314D890	00	2314D891	2D	2314D892	B3	2314D893	00
24AEE001	07	24AEE002	A0	24AEE003	37	24AEE004	C2
24AEE520	D1	24AEE521	DA	24AEE522	8C	24AEE523	B5
29DE2504	02	29DE2505	AD	29DE2506	FF	29DE2507	56
29DE4400	D0	29DE4401	5C	29DE4402	B4	29DE4403	2A
29DE9402	00	29DE9403	20	29DE9404	73	29DE9405	D4
29DEE500	B0	29DEE501	CD	29DEE502	23	29DEE503	1A

采用直接映射,TLB的内容如下所示.

索引	TLB标记	内容	有效位
0	0x08001	2314d220	1
1	0x01000	24aee520	0
2	0x005AE	00055004	0
3	0x016BA	0c23b09d	1
4	0x0AA00	0005545c	1
5	0x0000A	29dee500	0
6	0x5AE82	00023320	1
7	0x28DFC	00023000	1

I.某用户态进程试图写入虚拟地址:0x080016ba.该访问的最后结果是(1)_____.

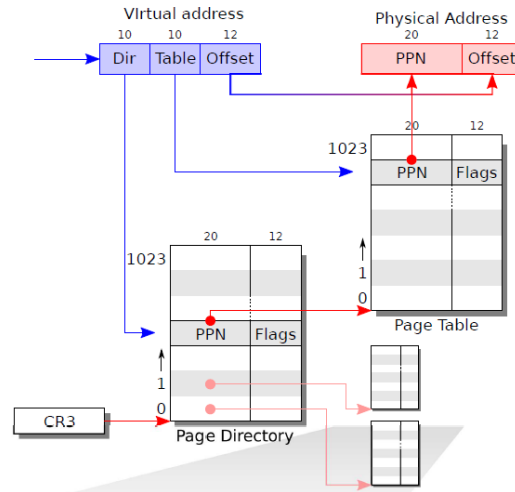
- A.该进程成功写入,未触发异常
- B.该进程触发了一个缺页异常
- C.该进程触发了一个非法访问异常

II.下面描述了具体的访问过程,请填空.如果某个空在访问过程中已不可用,请填入“--”

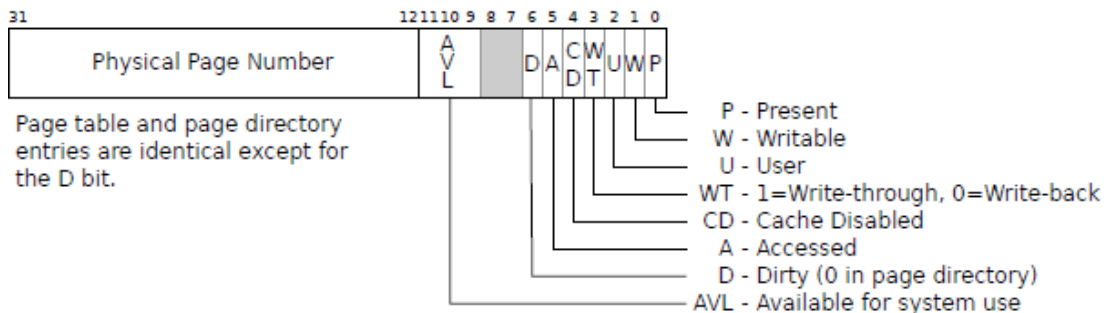
(i)TLB 的索引为 (2)_____,访问为(3)_____ (a) 命中 (b) 不命中.

(ii)一级页表表项地址为(4)_____.二级页表表项地址为(5)_____.最后物理地址为(6)_____.

16.(10分)Intel 的 IA32 体系结构采用二级页表,称第一级页表为页目录(Page Directory),第二级页表为页表(Page Table).其虚拟地址到物理地址的翻译方式如下图.先根据 CR3 找到页目录地址,然后依据偏移 Dir 找到一个页目录项,页目录项的高 20 位(PPN)为二级页表地址;在二级页表中根据偏移 Table 找到页表项,页表项中的高 20 位(PPN)即为物理地址的高 20 位,将这 20 位与虚拟地址的低 12 位拼在一起形成完整的物理地址.



页目录和页表均有 1024 项,每一项为 4 字节,含义如下:



页目录和页表由操作系统维护,通常情况下只能在内核态下访问,为了给用户提供一个访问页表项和页目录项内容的接口,假设操作系统中已经执行过如下代码段:

```
#define UVPT 0xef400000
#define PDX(la) (((unsigned int) (la)) >> 22) & 0x3FF
.....
kern_pgdir[PDX(UVPT)] = PADDR(kern_pgdir) | PTE_U | PTE_P;
```

其中 kern_pgdir 是操作系统维护的页目录数组,共 1024 项,每一项的类型为 unsigned int. PADDR(kern_pgdir) 用于获得 kern_pgdir 的物理地址,页目录在物理内存中正好占一页,所以 kern_pgdir 的物理地址是 4KB 对齐的. PTE_U 和 PTE_P 代表了这个页目录项的权限,即用户态可访问(只读). 可以看到,这条语句将页目录的第 PDX(UVPT) 项指向了页目录自身. 利用这一点,对于给定的虚拟地址 va, 可以获得 va 对应的页目录项和页表项内容,分别对应于函数 get_pde 和 get_pte, 请完成这两个函数:

```
#define UVPT 0xef400000
// get_pde(va):获取虚拟地址 va 对应的一级页表(页目录)中的页目录项内容
unsigned int get_pde(unsigned int va) {
    unsigned int pdx = (va >> (1)) & (2);
    unsigned int addr = UVPT + ((3)) + pdx * 4;
    return *((unsigned int *) (addr));
}

// get_pte(va):获取虚拟地址 va 对应的二级页表中的页表项内容
unsigned int get_pte(unsigned int va) {
    unsigned int PGNUM = va >> (4);
    unsigned int addr = (5) + PGNUM;
    return *((unsigned int *) (addr));
}
```


参考答案

一、选择题

- 1.D.考察 TLB,Cache,页式虚拟存储器基本性质.
- 2.C.考察 malloc 函数是显式地分配和释放堆存储器
- 3.C.本题考查对页表组成的理解.页块大小为 8KB,即 2^{13} byte.在 64 位机器上,一个页表条目为 8byte.故共有页表条目 2^{10} 项,故每一级页表可以表示 10 位地址.因此三级页表存储,共需要 $10 \times 3 = 30$ 位.
- 4.B.fork 使用之后,只生成新的 mm_struct 等结构,不会将原来的数据也拷贝一份,而是采用了 copy-on-write 的策略,因而占用的物理内存不会是原来的两倍,同时这两个进程的页表项权限都只有读.
- 5.B.最佳适应算法的空白区是按小大递增顺利链接在一起.
- 6.B. $4KB = 2^{12}$,所以页内地址有 12 位. $4KB/4B = 1K$,所以页目录和每个页表中的页表项数为 1K 个.因此,在 32 位的虚拟地址中,最低的 12 位为页内地址(Offset),最高的 10 位为页目录的虚拟地址(Dir),中间 10 位为页表的虚拟地址(Table).十六进制逻辑地址 8052CB 转换为二进制后为 10 000000101 001011001011,Dir 为 10,即 10 进制的 2,在表中对应到页表 1.Table 为 101,即 10 进制的 5,在表中对应到物理页面 7.因此,物理地址应为 7 的二进制 111 和 Offset 的拼合,即 111001011001011,转换为十进制为 29387,答案为 B.
- 7.C.由题意,64 位虚拟地址的虚拟空间大小为 2^{64} .页面长为 8KB,页表项 4 字节,所以一个页面可存放 2K 个表项.由于最高层页表占 1 页,也就是说其页表项个数最多为 2K 个,每一项对应一页,每页又可存放 2K 个页表项,依次类推可知,采用的分页层数为 5 层.
- 8.B.
- 9.B.
- 10.B. ($VPN = TLBT + TLBI$).
- 11.D.首次释放不一定是空闲块数量的线性时间.
12. B.考察物理地址和虚拟地址的映射关系.同一个 page.

二、非选择题

13.答案

I:1024, 5

II:第一次试图向地址 0xD7416560 写入内容,TLB 索引为 6,完成写之后该项 TLB 内容是 0x00A23067,二级页表页表项地址为 \,物理地址是 0x00A23560.第二次试图向地址 0x0401369B 写入内容,TLB 索引为 3,完成写之后该项 TLB 内容是 0x00BA4067 二级页表页表项所在地址为 0x29DE404C,物理地址是 0x00BA469B.

III:5,15

14.答案:

I:(1)128;(2)0;(3)0;

两个虚拟地址映射的是同一块物理内存;因此读出的就是写入的;此过程中全部 TLB 命中,因而既无 TLB miss,也不会有 page fault.

II:不确定;因为是大页,一定不是当前页目录项对应的页表页,但不一定该页面不会用作其他页目录项对应的页表页.

是;当前页目录项(977)对应的页表页.

不确定;任何页面都可能用作页表页.

III:B;虚拟地址对应的页表页的页面号(0x09C33)已知,通过其地址直接加 3G(即 0xC0000000),即可得到当前页表页的基地址(0xC9C33000),在加上对应的第 0x27 乘以 4 到页内偏移.

不会;因为地址 0xC9C33000 在大页映射范围内,已经被大页 TLB 项覆盖到了,会直接命中.

不能直接修改.因为 TLB 项中的内容和页表中的内容不一致,需要将对应的 TLB 项设置为失效,然后通过 TLB miss 重新加载页表结构中新的地址映射关系,之后才能访问对应的虚拟地址.

15.答案:

I. (c);

II: 1, (b); 0x0C23B080; 0x00055004(或"--");8974D6BA(或"--")

说明:联合考察虚存、高速缓存(通过 TLB 考察)、大端法和小端法的知识.

16.答案:

```
#define UVPT 0xef400000
unsigned int get_pde(unsigned int va) {
    unsigned int pdx = (va >> 22) & 0x3ff;
    unsigned int addr = UVPT + (UVPT >> 10) + pdx * 4;
    return *((unsigned int *)(addr));
}

unsigned int get_pte(unsigned int va) {
    unsigned int PGNUM = va >> 12;
    unsigned int addr = UVPT + PGNUM * 4;
    return *((unsigned int *)(addr));
}
```

要获取页目录项,需要让 $\text{addr} = \text{UVPT}[31:22] \mid \text{UVPT}[31:22] \mid \text{PDX} \mid 00$,即最高的 10 位为 UVPT 的高 10 位,中间的 10 位也是 UVPT 的高 10 位,接下来 10 位是 PDX(va 的高 10 位),最后两位为 0. 这样,根据翻译机制,第一步取出 addr 的高 10 位 UVPT[31:22],由于页目录中 UVPT[31:22]指向的是页目录本身,所以这一步找到的二级页表仍然为页目录本身.第二步取出中间 10 位 UVPT[31:22],重复刚才的过程,这一步找到的物理页仍然为页目录本身.最后加上偏移量 PDX|00 就可以找到页目录中的第 PDX 项.

要获取页表项,让 $\text{addr} = \text{UVPT}[31:22] \mid \text{PDX} \mid \text{PTX} \mid 00$,其中 PTX 是 va 的中间 10 位.这样,第一步翻译后得到的二级页表仍然指向页目录本身,第二步翻译后得到的物理页指向的是 va 对应的二级页表.最后用偏移量 PTX|00 即可获得二级页表中的第 PTX 项.