Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Fall Term 2012

SYSTEMS PROGRAMMING AND COMPUTER ARCHITECTURE
Assignment 9: Virtual Memory

Assigned on: **29th Nov 2012**
Due by: **6th Dec 2012**

Question 1

A virtual memory system is described by a number of parameters:

	Description
N	Number of addresses in virtual address space ($N = 2^n$)
M	Number of addresses in physical address space ($M = 2^m$)
P	Page size ($P = 2^p$) (in bytes)

A virtual address consists of:	
VPO	Virtual page offset (bytes)
VPN	Virtual page number

A physical address consists of:	
PPO	Physical page offset (bytes)
PPN	Physical page number

For a memory system with $n = 30$ and $m = 22$, determine the number of bits in the VPN, VPO, PPN, and PPO for the following page sizes:

P	# VPN bits	# VPO bits	# PPN bits	# PPO bits
512 Bytes				
1 KB				
2 KB				

Question 2

Consider a small memory system with a TLB and a L1 data cache. We make the following assumptions to simplify the question:

- The memory is byte addressable, each access is always to **a single byte**.
- Virtual addresses are 14 bits wide.
- Physical addresses are 12 bits wide.
- The page size is 64 bytes.
- The TLB is two-way set associative with 8 total entries.
- The L1 (data) cache is physically addressed, direct mapped, and has a 4-byte block size; there are 16 sets.

At some point in the execution of a program, the page table (only the top 16 entries are shown), the TLB, and the cache have these contents:

TLB: (all values are hexadecimal)

Set	Tag	PPN	Valid	Tag	PPN	Valid
0	05		0	12	42	1
1	02	20	1	04	32	1
2	01	22	1	07		0
3	01	22	1	02		0

Page table: (all values are hexadecimal)

VPN	PPN	Valid
00		
01	02	1
02	03	1
03		
04		
05		
06	22	1
07	22	1

VPN	PPN	Valid
08		
09	20	1
0a		
0b		
0c		
0d	04	1
0e		
0f		

Cache: (all values are hexadecimal)

Index	Tag	Valid	Block[0]	Block[1]	Block[2]	Block[3]
00	00	1	de	ad	fa	ce
01	31	0				
02	24	1	02	13	e1	de
03	22	1	22	23	e2	2e
04	21	0				
05	22	0				
06	18	0				
07	22	1	9a	01	00	de
08	20	0				
09	22	1	83	1a	09	ce
0a	20	1	0d	1f	f1	d0
0b	3a	0				
0c	3f	0				
0d	24	1	be	fb	57	02
0e	23	0				
0f	22	1	cf	7a	9b	a0

Blank entries indicate that contents of this block are not relevant.

For a given virtual address, indicate the VPN, the TLB index, the TLB tag, if there is a page fault and (if appropriate) the PPN.

To help you answer these questions, start with the virtual address in binary. Then show the physical address (in binary) and indicate the byte offset, the cache index and tag, if there is a cache hit, and the byte value retrieved from the cache (if appropriate). If there is a cache miss, enter "—" for "Cache byte returned". If there is a page fault, enter "—" for "PPN" and leave parts (c) and (d) blank.

a) Virtual address is 0x0268

(a) Virtual address (in binary)

13	*12*	*11*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*

(b) Address translation

Parameter	Value
VPN	
TLB index	
TLB tag	
TLB hit (y/n)	
Page fault (y/n)	
PPN	

(c) Physical address (in binary)

11	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*

(d) Physical memory reference

Parameter	Value
Byte offset	
Cache index	
Cache tag	
Cache hit (y/n)	
Cache byte returned	

b) Virtual address is 0x0197

(a) Virtual address (in binary)

13	*12*	*11*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*

(b) Address translation

Parameter	Value
VPN	
TLB index	
TLB tag	
TLB hit (y/n)	
Page fault (y/n)	
PPN	

(c) Physical address (in binary)

11	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*

(d) Physical memory reference

Parameter	Value
Byte offset	
Cache index	
Cache tag	
Cache hit (y/n)	
Cache byte returned	

c) Virtual address is 0x035e

(a) Virtual address (in binary)

13	*12*	*11*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*

(b) Address translation

Parameter	Value
VPN	
TLB index	
TLB tag	
TLB hit (y/n)	
Page fault (y/n)	
PPN	

(c) Physical address (in binary)

11	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*

(d) Physical memory reference

Parameter	Value
Byte offset	
Cache index	
Cache tag	
Cache hit (y/n)	
Cache byte returned	

d) Virtual address is 0x021a

(a) Virtual address (in binary)

13	*12*	*11*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*

(b) Address translation

Parameter	Value
VPN	
TLB index	
TLB tag	
TLB hit (y/n)	
Page fault (y/n)	
PPN	

(c) Physical address (in binary)

11	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*

(d) Physical memory reference

Parameter	Value
Byte offset	
Cache index	
Cache tag	
Cache hit (y/n)	
Cache byte returned	

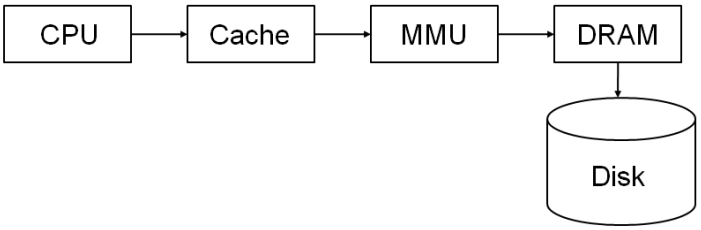

Figure 1: Schema of the memory system in Question 4.

Question 3

Consider a direct mapped cache of size 1K bytes. Assume that the cache starts empty and that local variables and computations take place completely within the registers (only the matrix is in memory). Remember that `sizeof(int) == 4`.

```
1  int sum_nth_array_elements_twice(int m[128], int n) {
2
3      int i;
4      int tot = 0;
5
6      for (i = 0; i < 128; i += n) {
7              tot += m[i];
8      }
9      for (i = 0; i < 128; i += n) {
10             tot += m[i];
11     }
12     return tot;
13 }
```

a) What is the cache miss rate if the block size is 4 bytes and n is 1?

b) What is the cache miss rate if the block size is 16 bytes and n is 2?

Question 4

Assume the memory system in Figure 1 has the following implementation:

- Assume there is only one process running on this computer.
- The memory is byte addressable, each access is always to a single byte.
- Virtual addresses are 32-bit long.
- The cache has a total of 2^{10} lines and is four-way set associative. The cache has 8-byte data blocks and is write-through.
- The main memory is 2^{26} bytes large.
- A page has a size of 2^{12} bytes.

a) Does this system cache virtual or physical addresses?

b) How many bytes can the cache hold?

c) Consider the contents of memory location 0x05a0861. In how many different locations in the cache might this data be stored (provide the set number as well)? In how many different locations in the cache will the data actually be found at any given moment?

d) How many tag bits are used? How many blocks in virtual memory compete for one set in the cache?

e) Explain how the 32-bit addresses produced by the CPU are processed by the cache. Indicate which bits are used to select data from the data block, which bits are used to select the appropriate set, and which bits are used to compare the tag field in order to determine if there is a cache hit.

f) How many virtual pages are available in the page table that the MMU uses? How many bits does each page table entry have (at least)? Assume a one-level page table.

g) At any given time, what is the greatest number of page-table entries that can have their valid bit set to 1?

h) Explain how the 32-bit addresses produced by the cache after a "miss" are processed by the page table. Indicate which bits are used to form the page offset and which bits are used to form the virtual page number. Explain what a TLB is and where it would be located in the figure.

Hand In Instructions

This is a paper exercise. If you want your solution to be revised please hand it in during your exercise class on the due date.