ZHEJIANG UNIVERSITY COLLEGE OF INFORMATION SCIENCE AND ELECTRONICS ENGINEERING

HOMEWORK #7

Computer Organization and Design

Name: Student ID:

Major: Electronic Science and Technology

Date: 2024 年 12 月 20 日

Problem 1.

Virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. This exercise shows how this table must be updated as addresses are accessed. The following table is a stream of virtual addresses as seen on a system. Assume 4 KiB pages, a 4-entry fully associative TLB, and true LRU replacement. If pages must be brought in from disk, increment the next largest page number.

4095, 31272, 15789, 15000, 7193, 4096, 8912

TLB

Valid	Tag	Physical Page Number
1	11	12
1	7	4
1	3	6
0	4	9

Page Table

Valid	Physical page or in disk
1	5
0	Disk
0	Disk
1	6
1	9
1	11
0	Disk
1	4
0	Disk
0	Disk
1	3
1	12

a) Given the address stream in the table, and the initial TLB and page table states shown above, show the final state of the system. Also list for each reference if it is a hit in the TLB, a hit in the page table, or a page fault.
b) Repeat Exercise a), but this time use 16 KiB pages instead of 4 KiB pages. What would be some of the advantages of having a larger page size? What are some of the disadvantages?
c) Show the final contents of the TLB if it is 2-way set associative. Also show the contents of the TLB if it is direct mapped. Discuss the importance of having a TLB to high performance. How would virtual memory accesses be handled if there were no TLB?

There are several parameters that impact the overall size of the page table. Listed below are several key page table parameters.

Virtual address size	Page size	Page table entry size
32 bits	4 KiB	8 bytes

d) Given the parameters in the table above, calculate the total page table size for a system running 5 applications that utilize half of the memory available.
e) Given the parameters in the table above, calculate the total page table size for a system running 5 applications that utilize half of the memory available, given a two-level page table approach with 256 entries. Assume each entry of the main page table is 6 bytes. Calculate the minimum and maximum amount of memory required.
f) A cache designer wants to increase the size of a 4 KiB virtually indexed, physically tagged cache. Given the page size listed in the table above, is it possible to make a 16 KiB direct-mapped cache, assuming 2 words per block? How would the designer increase the data size of the cache?

Answer： a)

the tag of the address stream in the table is 0,7,3,3,1,1,2

- address 4095 with tag 0：hit in the page table, change TLB
- address 31272 with tag 7：hit in TLB
- address 15789 with tag 3：hit in TLB
- address 15000 with tag 3：hit in TLB
- address 7192 with tag 1：page fault, change page table, change TLB
- address 4096 with tag 1：hit in TLB
- address 8912 with tag 2：page fault, change page table, change TLB

表 1　TLB

Valid	Tag	Physical Page Number
1	1	13
1	7	4
1	3	6
1	2	14

表 2　Page Table

Valid	Physical page or in disk
1	5
1	13
1	14
1	6
1	9
1	11
0	Disk
1	4
0	Disk
0	Disk
1	3
1	12

b)

use 16 KiB pages instead of 4 KiB pages.

the tag of the address stream in the table is 0,1,0,0,0,0,0

- address 4095 with tag 0：hit in the page table, change TLB
- address 31272 with tag 1：page fault, change page table, change TLB
- address 15789 with tag 0：hit in TLB
- address 15000 with tag 0：hit in TLB
- address 7192 with tag 0：hit in TLB
- address 4096 with tag 0：hit in TLB
- address 8912 with tag 0：hit in TLB

表 3　TLB

Valid	Tag	Physical Page Number
1	1	13
1	7	4
1	3	6
1	0	5

表 4　Page Table

Valid	Physical page or in disk
1	5
1	13
0	disk
1	6
1	9
1	11
0	Disk
1	4
0	Disk
0	Disk
1	3
1	12

Advantages of a Larger Page Size

- Reduced Page Table Size: With larger pages, the number of pages needed to cover the same amount of memory is reduced, which in turn reduces the size of the page table.
- Improved TLB Efficiency: Larger pages mean fewer entries are needed in the Translation Lookaside Buffer (TLB), which can improve TLB hit rates and overall memory access performance.
- Reduced I/O Overhead: Larger pages can reduce the frequency of page faults and the associated I/O operations, as more data is brought into memory with each page.

Disadvantages of a Larger Page Size

- Increased Internal Fragmentation: Larger pages can lead to more wasted memory within each page, as not all memory allocated within a page may be used.
- Higher Memory Overhead for Small Processes: Small processes may not fully utilize the larger pages, leading to inefficient use of memory.
- Longer Page Transfer Times: Larger pages take more time to transfer between disk and memory, which can increase the latency of page faults.

c)

for 2-way set associative cache, the index of the address stream in the table is 0,1,1,1,1,1,0 the tag of the address stream in the table is 0,3,1,1,0,0,1

表 5 TLB

Valid	Tag	Physical Page Number
1	0	5
1	2	14
1	1	13
1	7	4

for direct-mapped cache, the index of the address stream in the table is 0,3,3,3,1,1,2 the tag of the address stream in the table is 0,1,0,0,0,0,0

表 6 TLB

Valid	Tag	Physical Page Number
1	0	5
1	1	13
1	2	14
1	3	6

Importance of Having a TLB to High Performance

- Reduced Memory Access Time: The Translation Lookaside Buffer (TLB) is a cache that stores recent translations of virtual memory to physical memory addresses. By having a TLB, the system can quickly translate virtual addresses to physical addresses without having to access the page table in memory, significantly reducing memory access time.

- Improved CPU Efficiency: With a TLB, the CPU can spend less time on address translation and more time on executing instructions, leading to better overall performance and efficiency.
- Higher TLB Hit Rates: A well-designed TLB can achieve high hit rates, meaning that most memory accesses can be translated quickly. This is crucial for maintaining high performance, especially in systems with large and complex memory hierarchies.

Handling Virtual Memory Accesses Without a TLB

- Increased Memory Access Latency: Without a TLB, every virtual memory access would require a lookup in the page table, which is stored in main memory. This would significantly increase the latency of memory accesses, as accessing main memory is much slower than accessing a TLB.
- Higher CPU Overhead: The CPU would need to spend more time performing address translations, which would reduce the time available for executing instructions and decrease overall system performance.
- More Frequent Page Table Accesses: Without the TLB, the page table would need to be accessed for every memory operation, leading to increased memory traffic and potential bottlenecks in the memory subsystem.

d)

Each application uses half of the virtual memory, i.e.,

$$\text{Memory used per application} = \frac{2^{32}}{2} = 2^{31}\,\text{bytes}.$$

The number of pages required per application is:

$$\text{Pages per application} = \frac{2^{31}}{2^{12}} = 2^{19}\,\text{pages}.$$

The page table size for one application is:

$$\text{Page table size (1 application)} = 2^{19} \times 8\,\text{bytes} = 2^{22}\,\text{bytes} = 4\,\text{MiB}.$$

Final Answer: The total page table size is **20** MiB.

e)

1. Number of Pages:

- Virtual address space = 2^{32} bytes
- Page size = 2^{12} bytes

- Number of pages = $\frac{2^{32}}{2^{12}} = 2^{20}$ pages

2. Memory Utilization:

- Each application utilizes half of the memory available.
- Therefore, each application uses $\frac{2^{32}}{2} = 2^{31}$ bytes of memory.
- Number of pages used by each application = $\frac{2^{31}}{2^{12}} = 2^{19}$ pages

3. Two-Level Page Table:

- Each level-1 page table entry points to a level-2 page table.
- Number of level-1 entries = 256
- Each level-1 entry size = 6 bytes
- Number of level-2 entries = 2^{11} entries
- Each level-2 entry size = 8 bytes

4. Page Table Size for One Application:

- Level-1 page table size = 256 × 6 bytes = 1536 bytes
- Level-2 page table size = $2^{19} \times 8$ bytes = 2^{22} bytes
- Total page table size for one application = 1536 bytes + 2^{22} bytes = 4,097.5 KB

5.Minimum and Maximum Memory Required: if all the applications are not shell memory, then the maximum memory required is 5*4,097.5 KB=20,487.5 KB

if all the applications are shell same memory, then the minimum memory required is 4,097.5 KB

f)

for a 16 KiB direct-mapped cache, assuming 2 words per block, the cache size is 16 KiB, the block size is 8 bytes, and the number of blocks is 16 KiB / 8 bytes = 2024 blocks.

the cache offset is 3 bits, the cache index is 11 bits. but the page offset bits is 12 bits. 11bits+3bits=14bits > 12bits,

so it is impossible to designer such a cache because the cache index + cache offset is greater than the page offset, which may lead to address translation errors.

Increasing Data Size of the Cache:

- Set-Associative Cache: One way to increase the data size of the cache while maintaining the virtually indexed, physically tagged configuration is to use a set-associative cache. For example, a 2-way set-associative cache would halve the number of index bits required.
- Larger Page Size: Another approach is to increase the page size, which would

increase the number of page offset bits, allowing for a larger cache size.

Problem 2.

Consider a system with 40-bit virtual addresses, 36-bit physical addresses, and 64 KiB (2^{16} bytes) pages. The system uses a page map to translate virtual addresses to physical addresses; each page map entry includes dirty (D) and resident (R) bits.
a) Assuming a flat page map, what is the size of each page map entry, and how many entries does the page map have?

Size of page map entry in bits: _____

Number of entries in the page map: _____

b) If changed the system to use 16 KiB (2^{14} bytes) pages instead of 64 KiB pages, how would the number of entries in the page map change? Please give the ratio of the new size to the old size.

(# entries with 16 KiB pages) / (# entries with 64 KiB pages): _____

c) Assume we still use 64 KiB pages. The contents of the page map and TLB are shown to the right. The page map uses an LRU replacement policy and the LRU page (shown below) will be chosen for replacement. For each of these four accesses, compute its corresponding physical address and indicate whether the access causes a TLB miss and/or a page fault. Assume each access starts with the TLB and Page Map state shown to the right.

TLB

VPN (tag)	V	D	PPN
0x0	1	0	0xBE7A
0x3	0	0	0x7
0x5	1	1	0xFF
0x2	1	0	0x900

Fill in table below

	Virtual Addr	PPN (in hex)	PhysAddr (in Hex)	TLB Miss? (Y/N)	Page Fault? (Y/N)
1.	0x06004				
2.	0x30286				
3.	0x68030				
4.	0x4BEEF				

Page Map

VPN	R	D	PPN	
0	1	0	0xBE7A	
1	0	0	---	
2	1	0	0x900	
3	1	0	0x8	⇦ LRU PAGE
4	0	0	---	
5	1	1	0xFF	
6	1	0	0x70	

Answer : a)

Size of page map entry in bits:

- Physical Address Size: 36 bits
- page size: 64 KiB (2^{16} bytes)
- page offset bits: 16 bits
- thus, the size of the page map entry = 36 - 16 + 2 = 22 bits

Number of entries in the page map:

- Virtual Address Size: 40 bits
- Page Size: 64 KiB (2^{16} bytes)
- Number of Entries: $\frac{2^{40}}{2^{16}} = 2^{24}$ entries

b)

Number of entries in the new page map:

- Virtual Address Size: 40 bits
- Page Size: 16 KiB (2^{14} bytes)
- Number of Entries: $\frac{2^{40}}{2^{14}} = 2^{26}$ entries
- thus, the ratio of the number of entries in the new page map to the old page map is $\frac{2^{26}}{2^{24}} = 4$

c)

Virtual Addr	PPN (in hex)	PhysAddr (in Hex)	TLB Miss? (Y/N)	Page Fault? (Y/N)
0x06004	0xBE7A	0xBE7A6004	N	N
0x30286	0x8	0x80286	Y	N
0x68030	0x70	0x708030	Y	N
0x4BEEF	-	-	Y	Y