Discussion 6: Paging, Caches

March 8, 2024

Contents

1 Paging

1.1 Concept Check

1. True or False: The page table base pointer contains the virtual address of the page table.

   F, physical

2. True or False: If a user process accesses an address that generates a page fault, the OS will terminate this process for access violation.

   F. maybe demand paging or other scenario.

3. What are some advantages of having a larger page size? What about its disadvantages?

   It decrease the pte size and the page table size. internal fragmentation.

1.2 Little Page Translation

The following table shows the 4 entries in the page table. Recall that the valid bit is 1 if the page is resident in physical memory and 0 if the page is on disk or hasn't been allocated.

Valid Bit	Physical Page Number
0	7
1	9
0	3
1	2

If there are 1024 bytes per page, what is the physical address corresponding to the virtual address 0xF74?

0xF74 = 111101110100, phy page 2, phy adder = 1011 0111 0100 = 0xB74

1.3 Demand Pages

An up-and-coming big data startup has just hired you to help design their new memory system for a byte-addressable system. Suppose the virtual and physical memory address space is 32 bits with a 4KB page size.

1. Suppose you know that there will only be 4 processes running at the same time, each with a Resident Set Size (RSS) of 512MB and a working set size of 256KB. What is the minimum amount of TLB entries that your system would need to support to be able to map/cache the working set size for one process? What happens if you have more entries? What about if you have fewer entries?

256 / 4 = 64 entries in TLB. the performance would increase since the TLB is bigger. the TLB would miss more, the performance would suffer.

2. Suppose you run some benchmarks on the system and you see that the system is utilizing over 99% of its paging disk IO capacity, but only 10% of its CPU. What is a combination of the of disk space and memory size that can cause this to occur? Assume you have TLB entries equal to the answer from the previous part.

It is likely thrashing, since the meomory may be too small to hold the working set of 4 processes, namely, 1MB.

3. Out of increasing the size of the TLB, adding more disk space, and adding more memory, which one would lead to the largest performance increase and why?

Adding more memory so that paging in is less often. Because cpu interact with memory directly rather than disk.

2 Caches

2.1 Translation Trivia

1. Consider a machine with a physical memory of 8 GB, a page size of 8 KB, and a page table entry size of 4 bytes. How many levels of page tables would be required to map a 46-bit virtual address space if every page table fits into a single page?

   there can be 2K entries per page, resulting in 11 bits. 8kB = 13 bits. (46 - 13) / 11 = 3 level page tables.

2. List the fields of a page table entry (PTE) in your scheme.

   33 - 13 = 20 bits for PPN, the remaining 12 bits could be used as information bits

3. Without a cache or TLB, how many memory operations are required to read or write a single 32-bit word?

   look up pagetable + read memory = 3 +1 = 4

4. With a TLB, how many memory operations can this be reduced to? Best-case scenario? Worst-case scenario?

   1; 4

5. Consider a machine with a page size of 1024 bytes. There are 8KB of physical memory and 8KB of virtual memory. The TLB is a fully associative cache with space for 4 entries that is currently empty. Assume that the physical page number is always one more than the virtual page number. This is a sequence of memory address accesses for a program we are writing: 0x294, 0xA76, 0x5A4, 0x923, 0xCFF, 0xA12, 0xF9F, 0x392, 0x341.

   Here is the current state of the page table.

Valid Bit	Physical Page Number
0	NULL
1	2
0	NULL
0	4
0	5
1	6
1	7
0	NULL

0x294 = 000 10 1001 0100
0xA76 = 010 10 0111 0110
0x5A4 = 001 01 1010 0100
0x923 = 010 01 0010 0011
0xCFF = 011 00 11111111
0xA12 = 010 10 0001 0010
0xF9F = 011 11 1001 1111
0x392 = 000 11 1001 0010
0x341 = 000 11 0100 0001

How many TLB hits and page faults are there? What are the contents of the TLB at the end of the sequence?

hits: 11111 = 5
page faults: 1 1 1 = 3

TLB
tag PPN
0 1
2 3
1 2
3 4

2.2 AMAT Calculations

Assume you are building a memory scheme with single level page tables. Each main memory access takes 50 ns and each TLB access takes 10 ns.

1. Assuming no page faults (i.e. all virtual memory is resident,) what TLB hit rate is required for an AMAT of 61 ns?

   x * 60 + (1 - x) * 110 = 61.
   x = 49 / 50 = 0.98

2. Assuming a TLB hit rate of 50%, how does the AMAT of this scenario compare to no TLB?

   amat = 85 , without tlb = 100

3. To improve your system, you add a two level paging scheme and a cache. The cache has a 90% hit rate with a lookup time of 20 ns. Additionally, the TLB hit rate is now improved to 95%. What is the average time to read a location from memory?

mem access = 0.9 * 20 + 0.1 * 70 = 25

amat = 0.95 * (10 + 25) + 0.05 * (10 + 3 * 25) = 37.5

2.3 Associativity Analysis

A big data startup has just hired you to help design their new memory system for a byte-addressable system. Suppose the virtual and physical memory address space is 32 bits with a 4KB page size.

1. First, you create a direct mapped cache and a fully associative cache of the same size that uses an LRU replacement policy. You run a few tests and realize that the fully associative cache performs much worse than the direct mapped cache does. What's a possible access pattern that could cause this to happen?

repeatedly access X + 1 blocks. where X is the blocks cache can held.

2. Instead, your boss tells you to build a 8KB 2-way set associative cache with 64 byte cache blocks. How would you split a given virtual address into its tag, index, and offset numbers?

20bits(tag) 6bits(index)
6bits(offset)

2.4 Replacement Roulette

Assume your program has the following memory access pattern.

A	B	C	D	A	B	D	C	B	A

1. How many misses will you get with FIFO?

   7

2. How many misses will you get with LRU?

   8

3. How many misses will you get with MIN?

   5

4. If we increase the cache size, are we always guaranteed to get better cache performance? Explain for FIFO, LRU, and MIN.

   FIFO not. others are guaranteed.

2.5 On the Clock

1. Suppose that we have a 10-10-12 virtual address split using a two-level paging scheme. Assume that the physical address is 32-bit as well and PTE is 4 bytes. Show the format of a PTE complete with bits required to support the clock algorithm.

   20bits(PPN) dirty, use, writable, and valid bits

2. Assume that physical memory can hold at most four pages. The program you run has the following memory access pattern.

t	1	2	3	4	5	6	7	8	9	10	11	12
Page	A	B	C	A	C	D	B	D	A	E	B	F

   What pages remain in memory at the end of the following sequence of page table operations? What are the use bits set to for each of these pages?

E1 B0 F1 D0