Homework 5 Deadline: None

Fall 2022 (This is optional homework. You do not need to submit it.)

Q1. Assume we have a cache hierarchy where we have a 4-way set-associative cache. The virtual address space is 1GB, the page size is 2KB, and the cache block size is 4 bytes. Each way contains two blocks. We need 4 bits of flags per page entry. The physical address space is 512MB.

Answer the following questions. (For each part, you need to show your work and how you calculated this number.)

a) What is the maximum size of the cache which enables us to have a virtually-indexed physically tagged cache with no aliasing?
b) Using this cache size, how many bits are needed for the tag?
c) How many virtual pages does this memory have?
d) What is the size of the flat (one-level) page table (per process) in this system?
e) If we use a 2-level page table design with 6 bits for the first level and the rest for the second level, what would be the maximum size of this two-level page table?
f) If the 6 most significant bits of the virtual address always stay zero (i.e., the process never uses those addresses), what would be the size of the 2-level page table?

SOLUTION

a)

For no aliasing, need the VIPT index to be contained within the page offset.

index width + block offset width <= page offset width

log2(sets) + log2(blocks) <= log2(page size)

sets = $\frac{cache\ size}{line\ size * associativity} = \frac{cache\ size}{8\ B * 4\ ways}$

blocks = 2

page size = 2 kB

log2($\frac{cache\ size}{8\ B * 4\ ways}$**) + log2(2) <= log2(2kB)**

Cache Size = 32 kB

b)

tag size = physical address width - index width - block offset width

= log2(physical address space) - log2($\frac{cache\ size}{line\ size * associativity}$**) - log2(blocks)**

log2(512 * 2^20) - log2($\frac{32\,B * 2^{10}}{8\,B * 4\,ways}$) - log2(2)
tag size = 29 - 10 - 1 = 18 bits

c)
Number of virtual pages = $\frac{Virtual\ Address\ Space}{Page\ Size}$
= $\frac{1\,GB}{2\,kB}$ = $\frac{1 * 2^{30}}{2 * 2^{10}}$ = 524288

d)
Size of flat page table = page table entry size * number of virtual pages
page table entry size = PPN width + # Flag Bits = 18 bits (from part b) + 4 bits = 22 bits
number of pages = 524288
Size of flat page table = 22 bits * 524288 = ~1 MB (or ~2 MB)

e)
1st level table size = 2^number of bits * entry size = 2^6 * 22 bits = 176 B

2nd level table size = 2^number of bits * entry size
number of bits = Virtual Address bits - page offset bits - 1st level bits
= log2(1 GB) - log2(2kB) - 6
=30 - 11 - 6 = 13 bits
2nd level table size = 2^13 * 22 bits = ~22 kB (or ~23 kB)

page table size = 1st level table size + 2^6 * 2nd level table size
= 176 B + 2^6 * 22 kB = ~1 MB (or ~2 MB)

f)
1st level table size = 176 B (same as part e)

2nd level table size = 2^number of bits * entry size
number of bits = Virtual Address bits - page offset bits - 1st level bits - 6
= log2(1 GB) - log2(2kB) - 6 - 6
= 30 - 11 - 6 - 6 = 7 bits
2nd level table size = 2^7 * 22 bits = 352 B

page table size = 1st level table size + 2^6 * 2nd level table size
= 176 B + 2^6 * 352 B = ~22 kB (or ~23 kB)

Q2. For each configuration shown below, state how many bits are needed for the metrics defined in the following.

a. 32-bit operating system, 4-KB pages, 1 GB of RAM

b. 32-bit operating system, 16-KB pages, 2 GB of RAM

c. 64-bit operating system, 16-KB pages, 16 GB of RAM

1. Virtual address
2. Physical address
3. Virtual page number
4. Physical page number
5. Page Offset

SOLUTION
a)
1. 32 bits
2. log2(1 GB) = 30 bits
3. Virtual Address bits - Page Offset Bits = 32 - log2(4 kB) = 32 - 12 = 20 bits
4. Physical Address bits - Page Offset Bits = 30 bits - 12 bits = 18 bits
5. Page Offset Bits = 12 bits (calculated in a3)

b)
1. 32 bits
2. log2(2 GB) = 31 bits
3. Virtual Address bits - Page Offset Bits = 32 - log2(16 kB) = 32 - 14 = 18 bits
4. Physical Address bits - Page Offset Bits = 31 bits - 14 bits = 17 bits
5. Page Offset Bits = 14 bits (calculated in b3)

c)
1. 64 bits
2. log2(16 GB) = 34 bits
3. Virtual Address bits - Page Offset Bits = 64 - log2(16 kB) = 64 - 14 = 50 bits
4. Physical Address bits - Page Offset Bits = 34 bits - 14 bits = 20 bits
5. Page Offset Bits = 14 bits (calculated in c3)