Gasser Ahmed

[gasser18@vt.edu](mailto:gasser18@vt.edu)

ECE 5484, Homework 5

1. a. Blocks of main memory = 264 / 64 = 264 / 26 = 258 blocks

b. Since cache has 2048 = 211 blocks, therefore size of the block = 11 bits

Since each block contains 64 = 26 bytes, therefore size of the offset = 6 bits

Therefore, size of the tag = 64 – (11 + 6) = 64 – 17 = 47 bits

c. 0x00000000000163FA

= 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001 0110 0011 1111 1010

Tag = 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 000

Block = 10110001111

Offset = 111010

Therefore, 10110001111 = 0x58F = Cache block 1423

1. a. Blocks of main memory = 224 / 64 = 224 / 26 = 218 blocks

b. Since each block contains 64 = 26 bytes, therefore size of the offset = 6 bits

Therefore, size of the tag = 24 – 6 = 18 bits

c. Since it is associative cache, it can map anywhere.

1. a. Since number of sets = 64 / 2 = 32 = 25 sets, therefore size of set = 5 bits

Since each block contains 4 = 22 bytes, therefore size of the offset = 2 bits

Therefore, size of the tag = 21 – (5+2) = 14 bits

b. Since number of sets = 64 / 4 = 16 = 24 sets, therefore size of set = 4 bits

Since each block contains 4 = 22 bytes, therefore size of the offset = 2 bits

Therefore, size of the tag = 21 – (4+2) = 15 bit

1. a. 220 / 28 = 212 pages

b. 216 / 28 = 28 page frames

c. 220 / 28 = 212 entries

1. a. 7ns + 15ns = 22ns

b. EAT for Cache = Ratio for hit \* (time for hit) + Ratio for miss \* (time for miss)

= 0.97 \* (15ns) + 0.03 \* (15ns + 30ns) = 15.9ns

EAT for TLB= Ratio for hit TLB \* (time for hit TLB + EAT Cache)

= 0.95 \* (7ns + 15.9ns) = 21.755ns

1. a. Since there are 32MB, or 25x220 = 225 addresses

Therefore, we need 25 bits for a virtual address.

b. Since there are 4MB, or 22x220 = 222

Therefore, we need 22 bits for physical address.

c. Since there are 225/211 pages in virtual memory, therefore the page table can have 214 entries.

d. 0x37F = 00000000000000 01101111111,

Since the first 14 bits are the page, therefore the remaining bits (11 bits) are the offset

Therefore, replace the first 14 bits (00000000000000) by (00000000000001) (page 0 maps to

frame 1), to get the physical address 0000000000000101101111111, or 0xB7F

e. 0x1203 = 00000000000010 01000000011

Since the first 14 bits are the frame, therefore the frame is 2.

Since virtual page 4 maps to frame 2, so we will replace the first 14 bits (00000000000010)

with (00000000000100), to get the virtual address 0000000000010001000000011, or 0x2203