# CS 356 Cache Exercises

**Redekopp**

**Name: ____Solutions_____**

**Score: _____**

**Note: Attach all work to receive full credit**

1.) (18 pts.) A processor has a 32-bit memory address space (i.e. 32-bit addresses). The memory is broken into blocks of 32 **bytes** each. The computer also has a cache capable of storing 16K **bytes**.

a.) How many blocks can the cache store?
16Kbytes / 32 bytes = $2^{14}$ / $2^5$ = $2^9$ = 512 blocks

b.) Assuming the cache uses **direct-mapping**, how many bits are there in each of the TAG, BLOCK, and BYTE OFFSET fields fields of the address. Show your calculations.

| Tag | Block | Byte Offset |
|---|---|---|
| A31-A14<br>18-bits | A13-A5<br>9-bits | A4-A0<br>5-bits |
| Remaining Bits | 512 blocks = $2^9$<br>=> 9 address bits | 32 bytes per block |

c.) Assuming the cache uses a **4-way set-associative mapping**, how many bits are there in each of the TAG, SET and BYTE OFFSET fields of the address. Show your calculations.

| Tag | Set | Byte Offset |
|---|---|---|
| **A31-A12**<br>**20-bits** | **A11-A5**<br>**7-bits** | **A4-A0**<br>**5-bits** |
| Remaining Bits | 512 blocks / 4-ways=<br>$2^7$ blocks in a set<br>=> 7 address bits | 32 bytes per block |

2.) A processor has a 36-bit memory address space (i.e. 36-bit addresses). The memory is broken into blocks of 64 **bytes**. The computer also has a cache capable of storing **1 Megabyte**.

a.) How many blocks can the cache store?
1M / 64 bytes = $2^{20}$ / $2^6$ = $2^{14}$ = 16K (16,384) blocks

b.) Assuming the cache uses **direct-mapping**, how many bits are there in each of the TAG, BLOCK, and BYTE OFFSET fields of the address. Show your calculations.

| Tag | Block | Byte Offset |
|---|---|---|
| **A35-A20**<br>**16-bits** | **A19-A6**<br>**14-bits** | **A5-A0**<br>**6-bits** |
| Remaining Bits | 16K blocks = $2^{14}$<br>=> 14 address bits | 64 bytes per block |

c.) Assuming the cache uses an **8-way set-associative mapping**, how many bits are there in each of the TAG, SET and BYTE OFFSET fields of the address. Show your calculations.

| Tag | Set | Byte Offset |
|---|---|---|
| **A35-A17**<br>**19-bits** | **A16-A6**<br>**11-bits** | **A5-A0**<br>**6-bits** |
| Remaining Bits | 16K blocks / 8-ways=<br>$2^{11}$ blocks in a set<br>=> 11 address bits | 64 bytes per block |

3.) Exercises 5.3.3 in CO&D, 4th Ed. page 550. Exercises 5.3.3 in CO&D, 4th Ed. page 550. You are asked to optimize a cache capable of storing 8 bytes total for the given references. There are three **direct-mapped** cache designs possible by varying the block size: C1 has one-byte blocks, C2 has two-byte blocks, and C3 has four-byte blocks. In terms of miss rate, which cache design is best? If the miss stall time is 25 cycles, and C1 has an access time of 2 cycles, C2 takes 3 cycles, and C3 takes 5 cycles, which is the best cache design? (Every access, hit or miss, requires an access to the cache)

Do this exercise only for access pattern a below. To make this easier for you, the address sequence has been converted to binary below. First do the address bit breakdown, then complete the table to answer the question.

| Address Sequence a | LS 8-bits of address |
|---|---|
| 1 | 0000 0001 |
| 134 | 1000 0110 |
| 212 | 1101 0100 |
| 1 | 0000 0001 |
| 135 | 1000 0111 |
| 213 | 1101 0101 |
| 162 | 1010 0010 |
| 161 | 1010 0001 |
| 2 | 0000 0010 |
| 44 | 0010 1100 |
| 41 | 0010 1001 |
| 221 | 1101 1101 |

| Cache | Tag | Block | Byte Offset |
|---|---|---|---|
| C1 | An-A3 | A2-A0 | None (no need) |
| C2 | An-A3 | A2-A1 | A0 |
| C3 | An-A3 | A2 | A1-A0 |

| Address Sequence | LS 8-bits of address | C1 Block | H/M | C2 Block | H/M | C3 Block | H/M |
|---|---|---|---|---|---|---|---|
| 1 | 0000 0001 | 1 | M | 0 | M | 0 | M |
| 134 | 1000 0110 | 6 | M | 3 | M | 1 | M |
| 212 | 1101 0100 | 4 | M | 2 | M | 1 | M |
| 1 | 0000 0001 | 1 | H | 0 | H | 0 | H |
| 135 | 1000 0111 | 7 | M | 3 | H | 1 | M |
| 213 | 1101 0101 | 5 | M | 2 | H | 1 | M |
| 162 | 1010 0010 | 2 | M | 1 | M | 0 | M |
| 161 | 1010 0001 | 1 | M | 0 | M | 0 | H |
| 2 | 0000 0010 | 2 | M | 1 | M | 0 | M |
| 44 | 0010 1100 | 4 | M | 2 | M | 1 | M |
| 41 | 0010 1001 | 1 | M | 0 | M | 0 | M |
| 221 | 1101 1101 | 5 | M | 2 | M | 1 | M |
| Miss Rate | | 11/12 | | | 9/12 | | 10/12 |
| Time | | 2*12 + 11*25 | 299 | 3*12+9*25 | 251 | 5*12+10*25 | 310 |

4.) Exercises 5.3.5 in CO&D, 4th Ed. page 551. Consider a direct-mapped cache of 64 KB and a block size of 8 bytes. Generate a series of read requests (address sequence) that will have a lower miss rate on a 2 KB 2-way set associative cache with the same block size than this 64 KB direct mapped cache? Then generate a series of read requests (address sequence) that will have a lower or equal miss rate in the 64KB direct-mapped cache. To make answers more standard, use address that are as small as possible to make your case (i.e. make unneeded tag bits of your address 0 [you should just need to change a few of the LS tag bits] and make the byte offset portion [i.e. LSBs] 0 wherever possible).

Use a sequence of 4 addresses for each answer. Write out your address to 20-bits. (i.e. 0x00800)

| Cache | Tag | Block or Set | Byte Offset |
|---|---|---|---|
| 64 KB, direct, 2 w/block | An-A16 | A15-A3 | A2-A0 |
| 2 KB, 2-way SA, 2 w/block | An-A3 | A9-A3 | A2-A0 |

To have the 2KB, 2-way SA cache have a lower miss rate, we want to find addresses that will map to the same block in the larger direct mapped cache but to different sets in the 2 KB cache.

| Address | Block in 64 KB | H/M | Set in 2 KB | H/M |
|---|---|---|---|---|
| 0x00000 | Block 0 | M | Set 0 – Way 0 | M |
| 0x10000 | Block 0 | M | Set 0 – Way 1 | M |
| 0x00000 | Block 0 | M | Set 0 – Way 0 | H |

To have the 64KB, direct mapped cache have a lower miss rate, we want to find addresses that will map to different blocks in the larger direct mapped cache but to the same set in the 2 KB cache.

| Address | Block in 64 KB | H/M | Set in 2 KB | H/M |
|---|---|---|---|---|
| 0x00000 | Block 0 | M | Set 0 – Way 0 | M |
| 0x00400 | Block 0x80 = 128 | M | Set 0 – Way 1 | M |
| 0x00800 | Block 0x100 = 256 | M | Set 0 – Way 0 | M / evict 00000 |
| 0x00000 | Block 0 | H | Set 0 | M |