Test 3 Review Problem Solutions

1. 32 KB data ⇒ 8 K words = 2^{13} words
Block size is 2 words. $2^{13}/2 = 2^{12}$ blocks

4-way set associative cache means there are 4 blocks per set.

$$\frac{2^{12} \text{ blocks}}{4 \text{ block/set}} = 2^{\textcircled{10}} \text{sets} = 1024 \text{ sets}$$

- Index is <u>10 bits long</u>.
- Block size is 2 words, which is 8 bytes = 2^3 bytes so offset is <u>3 bits long</u>.
- Tag is $32 - (10 + 3) =$ <u>19 bits long</u>.

Each block contains 2 words, or 64 bits, of data. Each block also has a 19 bit tag and one valid bit. Therefore, the total cache size is:

$$2^{12} \times (64 + 19 + 1) = 2^{12} \times 84$$
$$\simeq 344 \text{ Kbits}$$

or, a ~43 KB cache for 32 KB of data.

2. First, let's "remove" the offset portion. A block size of 4 words requires a 4 bit offset.

$$\frac{1208}{16} = 75$$

Note: this is equivalent to right shifting by 4 because $16 = 2^4$.

This result, 75, is the decimal representation of the tag and index bits of the address.

To isolate the index, we can mod this number by the number of sets.

$75 \% 256 = \boxed{75}$

In this case, because the tag is 0, we already have the index.

3. Clock period = 1 ns
Miss penalty = 25 clock cycles
Miss rate = .05
Hit time = 1 clock cycle

Average Memory Access Time = Hit time + (Miss Rate × Miss Penalty)

= 1 clock cycle + (.05 × 25 clock cycles)
= (1 + 1.25) clock cycles
= 2.25 clock cycles * 1 ns/cycles
= $\boxed{2.25 \text{ ns}}$

4. 8 K blocks = 2^{13} blocks
4 words = 16 bytes = 2^4 bytes

a. Direct-mapped: every set contains one block so there are $\boxed{2^{13} \text{ sets.}}$

2^{13} sets $\Rightarrow$ 13 index bits
2^4 bytes $\Rightarrow$ 4 offset bits

$32 - (13 + 4) = \boxed{15 \text{ tag bits}}$

b. Two-way set associative: every set contains 2 blocks so there are $\boxed{2^{12} \text{ sets}}$

2^{12} sets $\Rightarrow$ 12 index bits

2^4 bytes $\Rightarrow$ 4 offset bits

$32-(12+4) = \boxed{16 \text{ tag bits}}$

c. Four-way set associative: every set contains 4 blocks so there are $\boxed{2^{11} \text{ sets.}}$

2^{11} sets $\Rightarrow$ 11 index bits

2^4 bytes $\Rightarrow$ 4 offset bits

$32-(11+4) = \boxed{17 \text{ tag bits}}$

d. Fully-associative: there is one set containing all of the blocks. $\boxed{1 \text{ set}}$

2^0 sets = 1 set $\Rightarrow$ 0 index bits

2^4 bytes $\Rightarrow$ 4 offset bits

$32-4 = \boxed{28 \text{ tag bits}}$

5a. Tag: 31-10 Index: 9-4 Offset: 3-0

Block size: 2^4 bytes = 16 bytes = 4 words

Cache entries: 2^6 sets = 64 sets

Reference	Binary	Tag	Index	Offset
0	$\underbrace{0000\ 00}_{tag}\underbrace{00\ 0000}_{index}\underbrace{0000}_{offset}$	0	0	0
4	0000 0000 0000 0100	0	0	4
16	0000 0000 0001 0000	0	1	0
132	0000 0000 1000 0100	0	8	4
232	0000 0000 1110 1000	0	14	8
160	0000 0000 1010 0000	0	10	0
1024	0000 0100 0000 0000	1	0	0
30	0000 0000 0001 1110	0	1	14
140	0000 0000 1000 1100	0	8	12
3100	0000 1100 0001 1100	3	1	12
180	0000 0000 1011 0100	0	11	4
2180	0000 1000 1000 0100	2	8	4

Ref	Result	Block Replaced?
0	MISS	NO
4	Hit	NO
16	MISS	NO
132	MISS	NO
232	MISS	NO
160	MISS	NO
1024	MISS	yes
30	Hit	NO
140	Hit	NO
3100	MISS	yes
180	MISS	NO
2180	MISS	yes

$\boxed{\text{3 Blocks Replaced}}$

$\boxed{\text{3/12 hit ratio}}$

Final State of cache:

<index, tag, data>
<0, 1, 0>
<1, 3, 12>
<8, 2, 4>
<14, 0, 8>
<10, 0, 0>
<11, 0, 4>

b. Tag: 31-12 Index: 11-5 Offset: 4-0

Block size: 2^5 bytes = 32 bytes = 8 wor

Cache entries: 2^7 sets = 128 sets

Reference	Tag	Index	Offset
0	0	0	0
4	0	0	4
16	0	0	16
132	0	4	4
232	0	7	8
160	0	5	0
1024	0	32	0
30	0	0	30
140	0	4	12
3100	0	96	28
180	0	5	20
2180	0	68	4

Ref	Result	Block Replaced?
0	Miss	NO
4	Hit	NO
16	Hit	NO
132	Miss	NO
232	Miss	NO
160	Miss	NO
1024	MISS	NO
30	Hit	NO
140	Hit	NO
3100	Miss	NO
180	Hit	NO
2180	MISS	NO

NO Blocks Replaced

5/12 hit ratio

Final State of Cache:

<index, tag, data>

<0, 0, 30>
<4, 0, 12>
<7, 0, 8>
<5, 0, 20>
<32, 0, 0>
<96, 0, 28>
<68, 0, 4>

6. Write-through, no write allocate

R/W	Addr	Tag	Index	Offset	Result	Memref	Update Cache
W	1840	3	19	0	Miss	yes	NO
R	1844	3	19	4	Miss	yes	yes
R	304	0	19	0	Miss	yes	yes
W	308	0	19	4	Hit	yes	yes
R	820	1	19	4	Miss	yes	yes
R	824	1	19	8	Hit	NO	NO

Write-back, write allocate

R/W	Addr	Tag	Index	Offset	Result	MemRef	Update Cache
W	1840	3	19	0	MISS	yes	yes
R	1844	3	19	4	Hit	NO	NO
R	304	0	19	0	MISS	yes	yes
W	308	0	19	4	Hit	NO	yes
R	820	1	19	4	MISS	yes (2)	yes
R	824	1	19	8	Hit	NO	NO

t. a.

Page size is 4KB.

$$4\text{ KB} \simeq 4096 \text{ Bytes} = 2^{\overset{\boxed{12}}{}} \text{ Bytes}$$

$\downarrow$ 12 page offset bits

Address	Binary
4095	1111 1111 1111
31272	111 1010 0010 1000
15789	11 1101 1010 1101
7193	1 1100 0001 1001
4096	1 0000 0000 0000
8912	10 0010 1101 0000

Take 31272 as an example. The binary representation is:

$$\underbrace{111}_{\text{virtual page number}} \; \underbrace{1010\ 0010\ 1000}_{\text{page offset}}$$

So, the virtual page number is 7, and the page offset is 2600.

The TLB is a four-entry fully-associative cache. Generally speaking, this is a 4-way set-associative cache with one set.

$$1 \text{ set} = 2^{\overset{\boxed{0}}{}} \text{ sets}$$

$\downarrow$ 0 index bits

Therefore, our 32-bit virtual address is divided into 20 tag bits and 12 page offset bits.

Virtual address	Virtual Page #	Page Offset	TLB Tag	TLB Result	Page Table Result	Physica Page #
4095	0	4095	0	Miss	Hit	5
31272	7	2600	7	Hit	—	4
15789	3	3501	3	Hit	—	6
7193	1	3097	1	Miss	Miss	13
4096	1	0	1	Hit	—	13
8912	2	720	2	Miss	Miss	14

Take address 4095 as an example. The TLB tag is 0. Our caches starting state is

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

Since tag 0 is not in the cache, we have a miss. Using the virtual page #, 0, as an index into the page table, we see that we have a hit and the physical page number is 5. This is brought into the cache and replaces the last entry, which is invalid.

Take address 7193 as an example. The TLB tag is 1. Since tag 1 is not in the cache, we have a miss. Using the virtual page #, 1, as an index into the page table, we see that we have a page fault. We can pull the page from disk and assign it the next higher page number, 13. This translation is also placed into the TLB, replacing the least recently used entry (the one containing tag 11).

Final State of TLB and Page Table:

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

Valid	Physical Page # or 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

Address	Binary
9452	10 0100 1110 1100
30964	111 1000 1111 0100
19136	100 1010 1100 0000
46502	1011 0101 1010 0110
38110	1001 0100 1101 1110
16653	100 0001 0000 1101
48480	1011 1101 0110 0000

Virtual Address	Virtual Page #	Page Offset	TLB Tag	TLB Result	Page Table Result	Physical Page #
9452	2	1260	2	Miss	Miss	13
30964	7	2292	7	Hit	—	4
19136	4	2752	4	Miss	Hit	9
46502	11	1446	11	Miss	Hit	12
38110	9	1246	9	Miss	Miss	14
66653	4	269	4	Hit	—	9
48480	11	3424	11	Hit	—	12

Here, we're assuming the initial tag 11 entry was less recently used than the initial tag 3 entry.

Final State of TLB and Page Table:

Valid	Tag	Physical Page #
1	4	9
1	7	4
1	11	12
1	9	14

Valid	Physical Page # or Disk
1	5
0	Disk
1	13
1	6
1	9
1	11
0	Disk
1	4
0	Disk
1	14
1	3
1	12