Assignment4

Problem 1.

For a direct-mapped cache design with a 32-bit address, the following bits of the address are used to access the cache.

Tag	Index	Offset
31–10	9–5	4–0

1. What is the cache block size (in words)?
2. How many entries does the cache have?
3. What is the ratio between total bits required for such a cache implementation over the data storage bits? (Assume the cache has valid bit, dirty bit and reference bit)

Starting from power on, the following byte-addressed cache references are recorded.

Address											
0	4	16	132	232	160	1024	30	140	3100	180	2180

4. How many blocks are replaced?
5. What is the hit ratio?
6. List the final state of the cache, with each valid entry represented as a record of <index, tag, data>.

A: 1.cache block size is $2^5 = 32$ bytes = 8 words.

2.\# of entries = $2^5 = 32$

3.ratio = $\frac{8\times32+3+22}{8\times32} = 1.098$.

4.four blocks, at address 1024, 30, 3100, 2180 respectively.

Address	Binary address	Tag(decimal)	Index(decimal)	Hit
0	00000000_00000000_00000000_00000000	0	0	N
4	00000000_00000000_00000000_00000100	0	0	Y
16	00000000_00000000_00000000_00010000	0	0	Y
132	00000000_00000000_00000000_10000100	0	4	N
232	00000000_00000000_00000000_11101000	0	7	N
160	00000000_00000000_00000000_10100000	0	5	N
1024	00000000_00000000_00000100_00000000	1	0	N
30	00000000_00000000_00000000_00011110	0	0	N
140	00000000_00000000_00000000_10001100	0	4	Y
3100	00000000_00000000_00001100_00011100	3	0	N
180	00000000_00000000_00000000_10110100	0	5	Y
2180	00000000_00000000_00001000_10000100	2	4	N

5.$4/12 \approx 33.33\%$

6.<index,tag,data>: <0,3,MEM[3100]>, <4,2,MEM[2180]>, <5,0,MEM[180]>, <7,0,MEM[232]>

index	tag	data
0	3	MEM[3100]
4	2	MEM[2180]
5	0	MEM[180]
7	0	MEM[232]

Problem 2.

Caches are important to providing a high-performance memory hierarchy to processors. Below is a list of 32-bit memory address references, given as word addresses

3, 180, 43, 2, 191, 88, 190, 14, 181

The is exercise examines the impact of different cache designs, specifically comparing associative caches to the direct-mapped caches. The address stream is shown above.

1. For a three-way set associative cache with two-word blocks and a total size of 24 words, which bits represent index and which bits represent tag in 32-bit memory address (e.g. Index: 9-5 Tag: 31-10) ?
2. Using the given sequence, show the final cache contents for a three-way set associative cache with two-word blocks and a total size of 24 words. Use LRU replacement. For each reference identify the index bits, the tag bits, the block of set bits, and if it is a hit or a miss.
3. For a fully associative cache with one-word blocks and a total size of 8 words, which bits represent index and which bits represent tag in 32-bit memory address (e.g. Index: 9-5 Tag: 31-10) ?

A:

1.two-word block => Offset is one bit, and $24/2/3 = 4 = 2^2$, so Index: 2-1, thus Tag: 31-3.

2.

Address	Binary Address	Index bits	Tag bits	Block of set bits	Hit
3	00000000_00000000_00000000_00000011	01	00000000_00000000_00000000_00000	0	N
180	00000000_00000000_00000000_10110100	10	00000000_00000000_00000000_10110	0	N
43	00000000_00000000_00000000_00101011	01	00000000_00000000_00000000_00101	1	N
2	00000000_00000000_00000000_00000010	01	00000000_00000000_00000000_00000	0	Y
191	00000000_00000000_00000000_10111111	11	00000000_00000000_00000000_10111	0	N
88	00000000_00000000_00000000_01011000	00	00000000_00000000_00000000_01011	0	N
190	00000000_00000000_00000000_10111110	11	00000000_00000000_00000000_10111	0	Y
14	00000000_00000000_00000000_00001110	11	00000000_00000000_00000000_00001	1	N
181	00000000_00000000_00000000_10110101	10	00000000_00000000_00000000_10110	0	Y

addr	set[00]_block[0]	set[00]_block[1]	set[00]_block[2]	set[01]_block[0]	set[01]_block[1]	set[01]_block[2]	set[10]_block[0]	set[10]_block[1]	set[10]_block[2]	set[11]_block[0]	set[11]_block[1]	set[11]_block[2]
3				MEM[3]								
180				MEM[3]			MEM[8]					
43				MEM[3]	MEM[43]		MEM[8]					
2				MEM[2]	MEM[43]		MEM[8]					
191				MEM[2]	MEM[43]		MEM[8]			MEM[191]		
88	MEM[88]			MEM[2]	MEM[43]		MEM[8]			MEM[191]		
190	MEM[88]			MEM[2]	MEM[43]		MEM[8]			MEM[190]		
14	MEM[88]			MEM[2]	MEM[43]		MEM[8]			MEM[191]	MEM[14]	
181	MEM[88]			MEM[2]	MEM[43]		MEM[181]			MEM[191]	MEM[14]	

3.one-word block so the offset is 0 bit. And index is 0 bit in fully associative cache, so tag is 31-0 bits.

4. Using the given sequence, show the final cache contents for a fully associative cache with one-word blocks and a total size of 8 words. Use LRU replacement. For each reference identify the index bits, the tag bits, and if it is a hit or a miss.

Multilevel caching is an important technique to overcome the limited amount of space that a first level cache can provide while still maintaining its speed. Consider a processor with the following parameters:

Base CPI, No Memory Stalls	Processor Speed	Main Memory Access Time	First Level Cache MissRate per Instruction	Second Level Cache, Direct-Mapped Speed	Global Miss Rate with Second Level Cache, Direct-Mapped	Second Level Cache, Eight-Way Set Associative Speed	Global Miss Rate with Second Level Cache, Eight-Way Set Associative
1.5	2 GHz	100 ns	7%	12 cycles	3.5%	28 cycles	1.5%

5. Calculate the CPI for the processor in the table using: 1) only a first level cache, 2) a second level direct-mapped cache, and 3) a second level eight-way set associative cache.

4.

addr/tag	Hit	Cache
3	N	MEM[3]
180	N	MEM[3],MEM[180]
43	N	MEM[3],MEM[180],MEM[43]
2	N	MEM[3],MEM[180],MEM[43],MEM[2]
191	N	MEM[3],MEM[180],MEM[43],MEM[2],MEM[191]
88	N	MEM[3],MEM[180],MEM[43],MEM[2],MEM[191],MEM[88]
190	N	MEM[3],MEM[180],MEM[43],MEM[2],MEM[191],MEM[88],MEM[190]
14	N	MEM[3],MEM[180],MEM[43],MEM[2],MEM[191],MEM[88],MEM[190],MEM[14]
181	N	MEM[181],MEM[180],MEM[43],MEM[2],MEM[191],MEM[88],MEM[190],MEM[14]

5. (1)$7\% \times (100/0.5) + 1.5 = 15.5$. (2) $1.5 \times 93\% + 3.5\% \times (12 + 1.5) + 3.5\% \times (1.5 + 12 + 100/0.5) = 9.34$. (3) $1.5 \times 93\% + 5.5\% \times (28 + 1.5) + 1.5\% \times (1.5 + 28 + 100/0.5) = 6.46$

Problem 3.

This Exercise examines the single error correcting, double error detecting (SEC/DED) Hamming code.

1. What is the minimum number of parity bits required to protect a 128-bit word using the SEC/DED code?
2. Consider a SEC code that protects 8 bit words with 4 parity bits. If we read the value 0x375, is there an error? If so, correct the error.

A:1. 8.

$2.0x375 = (001101110101)_2$

	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	result
code	0	0	1	1	0	1	1	1	0	1	0	1	
p1	x		x		x		x		x		x		0
p2		x	x			x	x			x	x		0
p4				x	x	x	x					x	0
p8								x	x	x	x	x	1

error at position 8. So, the correct code is $(001101100101)_2 = 0x365$.

Problem 4.

Virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. This exercise shows how this table must be updated as addresses are accessed. The following data constitutes a stream of virtual addresses as seen on a system. Assume page size is 4 KiB, a 4-entry fully associative TLB, and true LRU replacement. If pages must be brought in from disk, increment the next largest page number (i.e. If the current maximum physical page number is 12, then the next physical page brought in from disk has a page number of 13).

4669, 2227, 13916, 34587, 12608

TLB:

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

Page table:

Valid	Physical Page or in Disk
1	5
0	Disk
0	Disk
1	6
1	9
1	11
0	Disk
1	4
0	Disk
0	Disk
1	3
1	12

1. Which bits represent virtual page number and which bits represent page offset in 32-bit virtual memory address?
2. Given the address stream shown, and the initial TLB and page table states provided above, show the final state of the system. Also list for each reference if it is a hit in the TLB, a hit in the page table, or a page fault.
3. Show the final contents of the TLB if it is 2-way set associative. **The page table states are provided above and the initial TLB is shown below.** Discuss the importance of having a TLB to high performance. How would virtual memory accesses be handled if there were no TLB?

TLB:

Valid	Tag	Physical Page	Index
1	2	9	0
0	0	5	0
1	5	12	1
1	3	4	1

A:1. page size is 4KiB, so the offset is 12 bits from 11-0 in 32-bit address. Thus, VPN is 32-12=20 bits from 31-12 in 32-bit address.

2.

Vaddr(decimal)	Vaddr(binary)	VPN and tag	Hit
4669	0001 0010 0011 1101	1	page fault
2227	1000 1011 0011	0	hit in page table
13916	0011 0110 0101 1100	3	hit in TLB
34587	1000 0111 0001 1011	8	page fault
12608	0011 0001 0100 0000	3	hit in TLB

TLB

Valid	Tag	PPN
1	0	5
1	8	14
1	3	6
1	1	13

Page Table

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

3.From the TLB shown, we find there are two sets, so index is one bit at 12^{th} bit in Vaddr.

Vaddr(decimal)	Vaddr(binary)	VPN	Tag	Index	Hit
4669	0001 0010 0011 1101	1	0	1	page fault
2227	1000 1011 0011	0	0	0	hit in page table
13916	0011 0110 0101 1100	3	1	0	hit in page table
34587	1000 0111 0001 1011	8	4	1	page fault
12608	0011 0001 0100 0000	3	1	1	hit in TLB

TLB

Valid	Tag	Physical Page	Index
1	1	6	0
1	0	5	0
1	0	13	1
1	1	14	1

Page Table

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

TLB is implemented in CPU which reduces plenty of time to access memory. In fact, we often access a continuous piece of data at an address, and TLB can record the physical pages we have recently visited, so that we no longer need to repeatedly access memory. And this can improve the efficiency and rationality of CPU.