EECS 112 & CSE 132, FALL 2017

Homework 4

Due date: December 1, 2017

Student ID:									Name:

1. Below is a list of 64-bit memory address references, given as word addresses:
0x03, 0xb4, 0x2b, 0x02, 0xbf, 0x58, 0xbe, 0x0e, 0xb5, 0x2c, 0xba, 0xfd
Consider a direct-mapped cache with 16 one-word blocks. Assume that cache is initially empty.

1.a. Write down a sequence of characters showing the result of each reference. ‘H’ for a hit, ‘P’ for a miss without having to replace a block already in cache, ‘R’ for a miss requiring replacement. For instance if first four references are miss on empty cache, second four are hits, and third four references are misses requiring replacement, the character sequence will be ‘PPPPHHHHRRRR’

Cache Size: 16 → 4 bit addressable index = 1 bit hex, 1 word block → 0 offset

address		index	tag	
0x03	P1	0		PPPP PPPR PPPP
0xB4	P2	1		
0x2B	P3	2	0	P4
0x02	P4	3	0	P1
0xBF	P5	4	B	P2
0x58	P6	5	B	P8
0xBE	P7	6		
0x0E	R1	7		
0xB5	P8	8	5	P6
0x2C	P9	9		
0xBA	P10	A	B	P10
0xFD	P11	B	2	P3
		C	2	P9
		D	F	P11
		E	~~B~~ 0	P7, R1
		F	B	P5

Note: I realized after #5 that only tag needs to be written to check.

1.b. Repeat 1.a. for a direct mapped cache with two-word blocks and the same total ~~number of blocks as 1.a.~~ data capacity

address	Result	Content after reference
0x03 0000 0011	P	<1, mem(0x02)> <1, mem(0x03)> <1,0>
0xB4 1011 0100	P	<2, mem(0xB4)><2, mem(0xB5)> <2,B>
0x2B 0010 1011	P	<5, mem(0x2A)><5, mem(0x2B)> <5,2>
0x02 0000 0010	H	<1, mem(0x02)><1, mem(0x03)> <1,0>
0xBF 1011 1111	P	<7, mem(0xBE)><7, mem(0xBF)> <7,B>
0x58 0101 1000	P	<4, mem(0x58)><4, mem(0x59)> <4,5>
0xBE 1011 1110	H	<7, mem(0xBE)><7, mem(0xBF)> <7,B>
0x0E 0000 1110	R	<7, mem(0x0E)><7, mem(0x0F)> <7,0>
0xB5 1011 0101	H	<2, mem(0xB4)><2, mem(0xB5)> <2,B>
0x2C 0010 1100	P	<6, mem(0x2C)><6, mem(0x2D)> <6,2>
0xBA 1011 1010	R	<5, mem(0xBA)><5, mem(0xBB)> <5,B>
0xFD 1111 1101	R	<5, mem(0xFC)><5, mem(0xFD)> <0,F>

1.c. Repeat 1.a. for a direct mapped cache with four-word blocks and the same total number of blocks as 1.a.

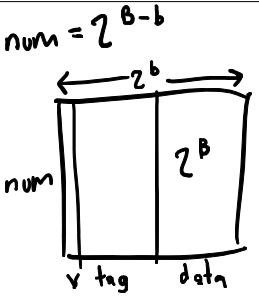
Reference	Result	Content after reference
0x03 0000 0011	P	<0,mem(0x00)><0,mem(0x01)><0,mem(0x02)><0,mem(0x03)> <0,0>
0xB4 1011 0100	P	<1,mem(0xB4)><1,mem(0xB5)><1,mem(0xB6)><1,mem(0xB7)> <1,B>
0x2B 0010 1011	P	<3,mem(0x28)><1,mem(0x29)><1,mem(0x2A)><1,mem(0x2B)> <3,2>
0x07 0000 0010	H	<0,mem(0x00)><0,mem(0x01)><0,mem(0x02)><0,mem(0x03)> <0,0>
0xBF 1011 1111	R	<3,mem(0xBC)><3,mem(0xBD)><3,mem(0xBE)><3,mem(0xBF)> <3,B>
0x58 0101 1000	P	<2,mem(0x58)><2,mem(0x59)><2,mem(0x5A)><2,mem(0x5B)> <2,5>
0xBE 1011 1110	H	<3,mem(0xBC)><3,mem(0xBD)><3,mem(0xBE)><3,mem(0xBF)> <3,B>
0x0E 0000 1110	R	<3,mem(0x0C)><3,mem(0x0D)><3,mem(0x0E)><3,mem(0x0F)> <3,0>
0xB5 1011 0101	R	<1,mem(0xB4)><1,mem(0xB5)><1,mem(0xB6)><1,mem(0xB7)> <1,B>
0x2C 0010 1100	R	<3,mem(0x2C)><3,mem(0x2D)><3,mem(0x2E)><3,mem(0x2F)> <3,7>
0xBA 1011 1010	R	<2,mem(0xB8)><2,mem(0xB9)><2,mem(0xBA)><2,mem(0xBB)> <2,B>
0xFD 1111 1101	R	<3,mem(0xFC)><3,mem(0xFD)><3,mem(0xFE)><3,mem(0xFF)> <3,F>

1.d. Which cache choice is optimum for the given access sequence?

Direct Mapped Cache with two-word blocks.

(Less Replacements)

2. Consider a direct mapped cache with a total data capacity of 2^B bytes, and a block size of 2^b bytes. Assume that the input byte address is A bits. Calculate the total number of bits required to fabricate this cache. You should consider both valid and dirty bit.



$tag = A - (B-b) - b = A - B$

$Total = 2^{B-b}(1 + A - B + 1 \cdot 2^3) = 2^{B-b}(9 + A - B)$

$Total = 2^{B-b}(9 - A - B)$

3. Cache block size (B) can affect both miss rate and miss latency. Find the block size that minimizes the total miss latency given the following miss rates (same for instruction and data cache) for various block sizes if:

8: 4% 16: 3% 32: 2% 64: 1.5% 128: 1%

3.1. Miss penalty is 20 ×B cycles

	miss latency	cycles
8: 4%	4% · 20 · 8	6.4
16: 3%	3% · 20 · 16	9.6
32: 2%	2% · 20 · 32	12.8
64: 1.5%	1.5% · 20 · 64	19.2
128: 1%	1% · 20 · 128	25.6

Best: B = 8

3.2. Miss penalty is 20 +B cycles

8	4% · 28	1.12
16	3% · 36	1.08
32	2% · 52	1.04
64	1.5% · 84	1.26
128	1% · 148	1.48

Best:

B = 32

3.3 Miss penalty is constant

8	4% · K	.04 K
16	3% · K	.03 K
32	2% · K	.02K
64	1.5% · K	.015K
128	1% · K	.01 K

Best:

B = 128

4.Assume that main memory accesses take 70 ns and that 36% of all instructions access data memory. The following table shows data for L1 caches attached to each of two processors, P1 and P2.

	L1 Size	L1 Miss Rate	L1 Hit Time
P1	2 KiB	8.0%	0.66 ns
P2	4 KiB	6.0%	0.90 ns

4.1. Assuming that the L1 hit time determines the cycle times for P1 and P2, what are their respective clock rates?

$f_{P1} = 1/t_{L1\,hit} = 1/0.66\,ns = 1.515\ GHz$

$f_{P2} = 1/t_{L1\,hit} = 1/0.90\,ns = 1.111\ GHz$

4.2. What is the Average Memory Access Time for P1 and P2 (in cycles)?

$AMAT = c_{hit} + Rate_{miss} \cdot c_{miss}$

$AMAT_{P1} = 1 + 0.08 \cdot \lceil 70/.667 \rceil = 1 + 0.08 \cdot 107 = 9.56\ cycles$

$AMAT_{P2} = 1 + 0.06 \cdot \lceil 70/.907 \rceil = 1 + 0.06 \cdot 78 = 5.68\ cycles$

5. Solve problem 1.b replacing the direct-mapped cache with the following cache designs (total number of blocks and block size is the same as problem 1.b). Each valid entry should look like <i,{tag1,lru1}, …,{tag_n,lru_n}>, where n is the number of blocks you can put in a row, and lru is the true LRU counter.

5.a. 2-way set associative cache

Reference	Result	Content after reference
0x03 0000 0011	P	<1,{0,0} { , }>
0xB4 1011 0100	P	<2,{B,0} { , }>
0x2B 0010 1011	P	<5,{2,0} { , }>
0x02 0000 0010	H	<1,{0,0} { , }>
0xBF 1011 1111	P	<7,{B,0} { , }>
0x58 0101 1000	P	<4,{5,0} { , }>
0xBE 1011 1110	H	<7,{B,0} { , }>
0x0E 0000 1110	P	<7,{B,1} {0,0}>
0xB5 1011 0101	H	<2,{B,0} { , }>
0x2C 0010 1100	P	<6,{2,0} { , }>
0xBA 1011 1010	P	<5,{2,1} {B,0}>
0xFD 1111 1101	P	<6,{2,1} {F,0}>

5.b. 4-way set associative cache

Reference	Result	Content after reference
0x03 0000 0011	P	<1,{0,0}{ , }{ , }{ , }>
0xB4 1011 0100	P	<2,{B,0}{ , }{ , }{ , }>
0x2B 0010 1011	P	<5,{2,0}{ , }{ , }{ , }>
0x02 0000 0010	H	<1,{0,0}{ , }{ , }{ , }>
0xBF 1011 1111	P	<7,{B,0}{ , }{ , }{ , }>
0x58 0101 1000	P	<4,{5,0}{ , }{ , }{ , }>
0xBE 1011 1110	H	<7,{B,0}{ , }{ , }{ , }>
0x0E 0000 1110	P	<7,{B,1}{0,0}{ , }{ , }>
0xB5 1011 0101	H	<2,{B,0}{ , }{ , }{ , }>
0x2C 0010 1100	P	<6,{2,0}{ , }{ , }{ , }>
0xBA 1011 1010	P	<5,{2,0}{B,0}{ , }{ , }>
0xFD 1111 1101	P	<6,{2,1}{F,0}{ , }{ , }>

5.c. For this specific example, what is the advantage of 4-way compared to 2-way?

While there is no immediate benefit, 4-way has more slots to hold more values (increases temporal locality and hit ratio). In the 2-way, if values with tag = 5, 6, or 7, then the LRU would be ejected and replacement is needed (also ↓ hit ratio)