CS2700 Homework 2

Phillip Janowski (pajmc2@mail.umsl.edu)

October 5, 2018

Question 1.

11.1

(a) $ABC + \overline{A} * \overline{B} * \overline{C}$

ABC	$\overline{A} * \overline{B} * \overline{C}$	ABC	$\overline{A} * \overline{B} * \overline{C}$	$ABC + \overline{A} * \overline{B} * \overline{C}$
000	111	0	1	1
001	110	0	0	0
010	101	0	0	0
011	100	0	0	0
100	011	0	0	0
101	010	0	0	0
110	001	0	0	0
111	000	0	0	1

(b) $ABC + A\overline{B} * \overline{C} + \overline{A} * \overline{B} * \overline{C}$

ABC	$\overline{A} * \overline{B} * \overline{C}$	ABC	$A\overline{B} * \overline{C}$	$A * \overline{B} * \overline{C}$	$ABC + A\overline{B} * \overline{C} + \overline{A} * \overline{B} * \overline{C}$
000	111	0	1	0	1
001	110	0	0	0	0
010	101	0	0	0	0
011	100	0	0	0	0
100	011	0	0	1	1
101	010	0	0	0	0
110	001	0	0	0	0
111	000	1	0	0	1

(c) $A(B\overline{C} + \overline{B}C)$

ABC	$\overline{A} * \overline{B} * \overline{C}$	$B\overline{C}$	$\overline{B}C$	$A(B\overline{C} + \overline{B}C)$
000	111	0	0	0
001	110	0	1	0
010	101	1	0	0
011	100	0	0	0
100	011	0	0	0
101	010	0	1	1
110	001	1	0	1
111	000	0	0	0

(d) $(A + B)(A + C)(\overline{A} + \overline{B})$

ABC	$\overline{A} * \overline{B} * \overline{C}$	$A + B$	$A + C$	$\overline{A} + \overline{B}$	$(A + B)(A + C)(\overline{A} + \overline{B})$
000	111	0	0	1	0
001	110	0	1	1	0
010	101	1	0	1	0
011	100	1	1	1	1
100	011	1	1	1	1
101	010	1	1	1	1
110	001	1	1	0	0
111	000	1	1	0	0

Question 2.

11.3

(a) $F = \overline{V + A + L} = \overline{V} * \overline{A} * \overline{L}$

VAL	$\overline{V} * \overline{A} * \overline{L}$	F
000	111	1
001	110	0
010	101	0
011	100	0
100	011	0
101	010	0
110	001	0
111	000	0

(b) $F = \overline{A} + \overline{B} + \overline{C} + \overline{D} = \overline{ABCD}$

$ABCD$	$\overline{A} * \overline{B} * \overline{C} * \overline{D}$	F
0000	1111	1
0001	1110	0
0010	1101	0
0011	1100	0
0100	1011	0
0101	1010	0
0110	1001	0
0111	1000	0
1000	0111	0
1001	0110	0
1010	0101	0
1011	0100	0
1100	0011	0
1101	0010	0
1110	0001	0
1111	0000	0

Question 3.

3.3 32 bit instruction, 8bit opcode, 24bit address

(a) $2^24 = 2^20*2^4 = 1$ megabyte $*16 = 16$ megabytes of addressable memory

(b) 1. 32bit address on a 16bit bus will take 3 cycles, 1 for the address and two for the data
2. 16 bit address on 16 bit bus will take 4 cycles, 2 for the address and two for data

(c) 24bits for the program counter and 32 bits for the instruction register

Question 4.

3.7

(a) A factor of 2

(b) 50 transfers, 25 of one byte, 25 of two bytes 75 transfers, $25 + 25 * 2$ A difference factor of 1.5

Question 5.

4.2

(a) 8bit tag, 14bit set, 2bit word

0x1111111 → 000100010001000100010001

Tag	Set	Word
00010001	00010001000100	01
11	444	1

0x666666 → 011001100110011001100110

Tag	Set	Word
01100110	01100110011001	10
66	1999	2

0xBBBBBB → 101110111011101110111011101

Tag	Set	Word
10111011	1011101110111	01
BB	2EEE	3

(b) 22bit Tag 2bit Word

0x1111111 → 000100010001000100010001

Tag	Word
0001000100010001000100	01
44444	1

0x666666 → 011001100110011001100110

Tag	Word
0110011001100110011001	10
19999	2

0xBBBBBB → 10111011101110111011101

Tag	Word
101110111011101110111	01
2EEEE	3

(c) 9 bit Tag 13 bit Set 2 bit Word

0x1111111 → 000100010001000100010001

Tag	Set	Word
000100010	0010001000100	01
22	444	1

0x666666 → 011001100110011001100110

Tag	Set	Word
011001100	1100110011001	10
CC	1999	2

0xBBBBBB → 10111011101110111011101

Tag	Set	Word
101110111	011101110111	01
17	EEE	3

Question 6.

4.4

(a) 8 bit Tag 14 bit Line 2 bit Word
Address length = 24 bits
Number of addressable units = $2^{2}4 = 16MB$
Block Size $2^{2} = 4$ words
Blocks in Memory = $2^{2}4 * 2^{-2} = 4MB$
Lines in Cache = $2^{1}4$
Tag Size = 8 bits

(b) 22 bit Tag 2 bit Word

Question 7.

4.8

$2^1$6 bytes 8 byte block size 32 Lines

(a) $\log_2 8 \rightarrow 3$ bit offset
$\log_2 32 \rightarrow 5$ bit line
$16 - 5 - 3 = 8$ bit tag

(b)

Tag	Line	Offset	Line Number
00010001	00011	011	Line 3
11000011	00110	100	Line 6
11010000	00011	101	Line 3
10101010	10101	010	Line 21

(c) 00011010|00011|010
will remain until address = 00011010|00011|111

(d) 8 byte address $2^8 = 256$ bytes
item Tags are used to identify items that can have the same place in the cache with two different addresses in memory