

## Solutions to Written Homework 01

**Assigned:** Wed 23 Sep 2009

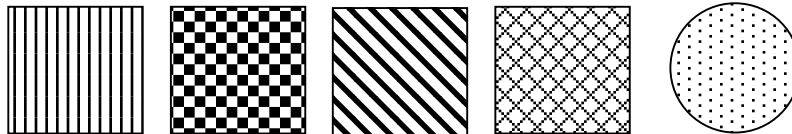
**Due:** Wed 30 Sep 2009

**Instructions:**

- The assignment is due at the *beginning* of class on the due date specified. Late assignments will be penalized 50%, as stated in the course information sheet. Late assignments *will not be accepted* after the solutions have been distributed.
- We expect that you will study with friends and often work out problem solutions together; however, you must write up your own solutions, in your own words. Cheating will not be tolerated.
- We expect your homework to be neat, organized, and legible. If your handwriting is unreadable, please type your solutions. Use 8.5in by 11in loose-leaf or printer paper, and please do not hand in sheets that have been ripped from spiral bound notebooks.
- Problem 1 requires graph paper. Printable graph paper may be found at the following URL:

<http://www.printfreegraphpaper.com/gp/c-i-14.pdf>

**Problem 1** [24 pts, (16,8)]: Patterns.

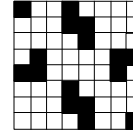
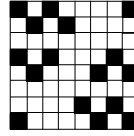
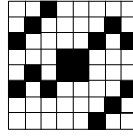
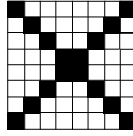


Patterns, like the ones above, are available in most drawing programs for filling regions. A pattern is often defined by an  $8 \times 8$  array of bits. In each of the following two examples, the  $8 \times 8$  array of bits on the left corresponds to the pattern on the right.



The 0s represent white, and the 1s represent black. Each row is an 8-bit binary number. As we know, a 4-bit binary number can be expressed as a single hex-digit, so an 8-bit binary number can be expressed with two hex-digits. Designers specify a pattern by giving eight 2-hex-digit numbers, one 2-hex-digit number per row. The two patterns given above are encoded as “11, 11, 11, 11, 11, 11, 11, 11” and “33, 33, CC, CC, 33, 33, CC, CC.”

- i. For each of the following patterns, give the eight 2-hex-digit encoding.



*Solution:* The solution for each of the images is obtained as follows. First, look at the first row of a selected image. Take the first 4 bits and convert them to hex. Then convert the last four bits to hex. Write the 2 resulting hex digits. If you obtain 0 and 0, write 00, not 0. Repeat for each row. The results are:

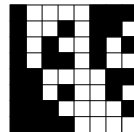
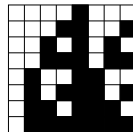
- 81, 42, 24, 18, 18, 24, 42, 81
- 20, 42, 85, 18, 58, A1, 02, 04
- A1, 50, 00, A2, 45, 00, 0A, 85
- 90, 18, 08, 43, C2, 10, 18, 09

- ii. Use graph paper to show the pattern described by each of the following sequences of eight 2-hex-digit numbers. (See the instructions above for a link to printable graph paper.)

08, 19, 2A, 3B, 4C, 5D, 6E, 7F

87, 96, A5, B4, C3, D2, E1, F0

*Solution:* The solution is obtained as follows. Take the first 2 digit hexadecimal number. Convert each of the digits to binary. You may need to insert 0 as most significant bits to make each of the 2 resulting binary numbers 4 bits. Write the resulting binary string of 8 bits as the first row. Repeat with the remaining hex numbers, each number becomes a separate row. At the end color the ones of the resulting matrix in black.



**Problem 2** [32 pts, (8,8,8,8)]: Negative numbers and two's complement.

In each of the parts below, show your work.

- i. Give the 8-bit two's complement representations of the following integers: 27, 83, -75, -69.

*Solution:*

$$27 = 16 + 8 + 2 + 1 = 00011011$$

$$83 = 64 + 16 + 2 + 1 = 01010011$$

$$-75 \implies 75 = 64 + 8 + 2 + 1 = 01001011 \implies 10110101$$

$$-69 \implies 69 = 64 + 4 + 1 = 01000101 \implies 10111011$$

- ii. Give the integer (in standard base-10 notation) which is represented by each of the following 8-bit two's complement numbers: 10110011, 01100011, 01101111, 10001111.

Solution:

$$10110011 \implies 01001101 = 1 + 4 + 8 + 64 = 77 \implies -77$$

$$01100011 = 1 + 2 + 32 + 64 = 99$$

$$01101111 = 1 + 2 + 4 + 8 + 32 + 64 = 111$$

$$10001111 \implies 01110001 = 1 + 16 + 32 + 64 = 113 \implies -113$$

- iii. Compute the following sums and differences using 8-bit two's complement representations, as shown in class and described in the text:  $-69 + 27$ ,  $83 - 75$ , and  $-75 - 69$ . In each case, indicate whether the calculation results in an overflow. If it does not result in an overflow, then verify that your answer is correct by converting the result back to standard base-10 notation.

*Note:* Use the two's complement representations from part i above.

Solution:

$$\begin{array}{r} 1\ 0\ 1\ 1\ 1\ 0\ 1\ 1 = -69 \\ +\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1 = 27 \\ \hline 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0 = -42 \end{array}$$

$$\begin{array}{r} 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1 = 83 \\ +\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1 = -75 \\ \hline (1)\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0 = 8 \end{array}$$

$$\begin{array}{r} 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1 = -75 \\ +\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 1 = -69 \\ \hline (1)\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 0 \neq -144 \end{array}$$

The only calculation that resulted in an overflow is the last one, because  $-144$  is outside the range of 8-bit two's complement.

- iv. What range of numbers can be represented in (a) 6-bit two's complement and (b) 18-bit two's complement? What are the *minimum* number of bits necessary to represent (c) 2010 and (d)  $-97$  in two's complement?

Solution:

(a)  $-32$  to  $31$ : 5 bits to represent the magnitude of the number and 1 bit to represent its sign, where  $2^5 = 32$ . Since 0 only has one representation (00 0000), we have one extra bit pattern  $10\ 0000$  – which we use for  $-32$ . In general, the range of an  $n$ -bit two's complement representation is  $-2^{n-1}$  to  $2^{n-1} - 1$ , inclusive.

(b)  $-131,072$  to  $131,071$ : (Similarly, where  $2^{17} = 131,072$ .)

(c) 12 bits. Because  $2^{10} - 1 = 1023 < 2010 \leq 2047 = 2^{11} - 1$ , we need 11 bits for the magnitude, and then one bit for the sign.

(d) 8 bits. For the same reasoning as (c), because  $-2^7 = -128 \leq -97 < -64 = -2^6$ .

**Problem 3** [20, (4 pts each)]: Multiplication.

Perform the following multiplications in binary. For each problem part, you must (1) convert each decimal number to binary, (2) perform the multiplication in binary, and (3) convert the binary result back to decimal. You must show your work.

*Note:* For consistency, place the binary representation of the left multiplicand in the top row of your multiplication and place the binary representation of the right multiplicand on the bottom row of your multiplication. Thus, “ $4 \times 7$ ” would be

$$\begin{array}{r} 1\ 0\ 0 \\ \times 1\ 1\ 1 \\ \hline \end{array}$$

while “ $7 \times 4$ ” would be

$$\begin{array}{r} 1\ 1\ 1 \\ \times 1\ 0\ 0 \\ \hline \end{array}$$

by this convention.

**i.**  $51 \times 10$

Solution: “ $51 \times 10 = 510$ ” would be

$$\begin{array}{r} 1\ 1\ 0\ 0\ 1\ 1 \\ \times 1\ 0\ 1\ 0 \\ \hline 0\ 0\ 0\ 0\ 0\ 0 \\ 1\ 1\ 0\ 0\ 1\ 1 \\ 0\ 0\ 0\ 0\ 0\ 0 \\ 1\ 1\ 0\ 0\ 1\ 1 \\ \hline 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0 \end{array}$$

$$11111110 \implies 2 + 4 + 8 + 16 + 32 + 64 + 128 + 256 \implies 510$$

**ii.**  $51 \times 7$

Solution: “ $51 \times 7 = 357$ ” would be

$$\begin{array}{r} 1\ 1\ 0\ 0\ 1\ 1 \\ \times 1\ 1\ 1 \\ \hline 1\ 1\ 0\ 0\ 1\ 1 \\ 1\ 1\ 0\ 0\ 1\ 1 \\ 1\ 1\ 0\ 0\ 1\ 1 \\ \hline 1\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1 \end{array}$$

$$101100101 \implies 1 + 4 + 32 + 64 + 256 \implies 357$$



- i. Construct the truth table for a 3-input, 1-output majority gate. Label your input bits  $A$ ,  $B$ , and  $C$ , and label your output bit  $M$ .

Solution:

$A$	$B$	$C$	$M$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

- ii. Using the DNF construction described in class, construct the *Boolean formula* corresponding to a 3-input, 1-output majority gate.

Solution: The disjunctive normal form (DNF) of  $M$  from the truth table is:

$$M = (\neg A \wedge B \wedge C) \vee (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge \neg C) \vee (A \wedge B \wedge C).$$

If you look carefully at the truth table and circuit for the full adder, as described in class and in the text, you will note that the carry-out bit  $C_o$  is exactly the majority vote of the three input bits  $A$ ,  $B$ , and  $C_i$ . However, the circuit (and corresponding Boolean formula) that implements the majority function for the carry-out bit  $C_o$  is quite different than your DNF construction above.

- iii. Give the Boolean formula that corresponds to the circuit (as shown in class and in the text) for the full adder carry-out bit. Use  $A$ ,  $B$ , and  $C$  as variables in your Boolean formula (i.e., replace  $C_i$  from the circuit diagram with simply  $C$ ). Note that  $A \oplus B \equiv (A \wedge \neg B) \vee (\neg A \wedge B)$ , as can easily be seen from the DNF construction applied to the XOR truth table.

Solution: The Boolean formula that corresponds to the circuit for the carry-out bit of the full adder is:

$$C_o = ((A \oplus B) \wedge C) \vee (A \wedge B).$$

- iv. Using the Laws of Logic, show that your Boolean formulae from parts ii and iii above are logically equivalent. In other words, start with your Boolean formula from part ii (obtained via the DNF construction) and apply the Laws of Logic until you derive your Boolean formula from part iii.

Solution: We start with the Boolean formula from part ii and apply distributivity, complement ( $C \vee \neg C = T$ ), and identity ( $(A \wedge B) \wedge T = A \wedge B$ ):

$$\begin{aligned} M &= (\neg A \wedge B \wedge C) \vee (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge \neg C) \vee (A \wedge B \wedge C) \\ &= (((\neg A \wedge B) \vee (A \wedge \neg B)) \wedge C) \vee ((A \wedge B) \wedge (\neg C \vee C)) \\ &= ((A \oplus B) \wedge C) \vee ((A \wedge B) \wedge T) \\ &= ((A \oplus B) \wedge C) \vee (A \wedge B) \\ &= C_o \end{aligned}$$

v. The simplest Boolean formula (and corresponding circuit) for the majority function is

$$M = (A \wedge B) \vee (A \wedge C) \vee (B \wedge C).$$

Again, using the Laws of Logic, show that your Boolean formula from part ii (obtained via the DNF construction) is logically equivalent to the formula given above.

*Hint:* The formula you obtain via the DNF construction in part ii will contain four terms (variables or their negations that are ANDed together), and one of these terms is  $(A \wedge B \wedge C)$ . Replicate this term three times, obtaining

$$(term_1) \vee (term_2) \vee (term_3) \vee (A \wedge B \wedge C) \vee (A \wedge B \wedge C) \vee (A \wedge B \wedge C).$$

(Why does this not change the function computed by the formula?) Now group each of your other terms together with a single copy of  $(A \wedge B \wedge C)$

$$\left[ (term_1) \vee (A \wedge B \wedge C) \right] \vee \left[ (term_2) \vee (A \wedge B \wedge C) \right] \vee \left[ (term_3) \vee (A \wedge B \wedge C) \right]$$

and apply the Laws of Logic to each of your groupings.

Solution: We start with the Boolean formula from part ii which also contains the replicated term and apply commutativity, distributivity, complement, and identity:

$$\begin{aligned} M &= (\neg A \wedge B \wedge C) \vee (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge \neg C) \vee (A \wedge B \wedge C) \\ &= (\neg A \wedge B \wedge C) \vee (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge \neg C) \vee (A \wedge B \wedge C) \\ &\quad \vee (A \wedge B \wedge C) \vee (A \wedge B \wedge C) \\ &= \left[ (\neg A \wedge B \wedge C) \vee (A \wedge B \wedge C) \right] \vee \left[ (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge C) \right] \\ &\quad \vee \left[ (A \wedge B \wedge \neg C) \vee (A \wedge B \wedge C) \right] \\ &= ((\neg A \vee A) \wedge (B \wedge C)) \vee ((\neg B \vee B) \wedge (A \wedge C)) \vee ((\neg C \vee C) \wedge (A \wedge B)) \\ &= (T \wedge (B \wedge C)) \vee (T \wedge (A \wedge C)) \vee (T \wedge (A \wedge B)) \\ &= (B \wedge C) \vee (A \wedge C) \vee (A \wedge B) \\ &= (A \wedge B) \vee (A \wedge C) \vee (B \wedge C). \end{aligned}$$