

CSE 140 Homework Two

October 30, 2014

*Only **Problem Set Part B** will be graded. Turn in only **Problem Set Part B** which will be due on November 12, 2014 (Wednesday) at 4:30pm.*

1 Problem Set Part A

- Roth&Kinney, 6th Ed: 3.6
- Roth&Kinney, 6th Ed: 3.9
- Roth&Kinney, 6th Ed: 3.10
- Roth&Kinney, 6th Ed: 3.12
- Roth&Kinney, 6th Ed: 3.15
- Roth&Kinney, 6th Ed: 3.16
- Roth&Kinney, 6th Ed: 3.18
- Roth&Kinney, 6th Ed: 3.19
- Roth&Kinney, 6th Ed: 3.20
- Roth&Kinney, 6th Ed: 3.21
- Roth&Kinney, 6th Ed: 3.29
- Roth&Kinney, 6th Ed: 3.30
- Roth&Kinney, 6th Ed: 3.33
- Roth&Kinney, 6th Ed: 3.34
- Roth&Kinney, 6th Ed: 3.36
- Roth&Kinney, 6th Ed: 3.37

- Roth&Kinney, 6th Ed: 4.5
- Roth&Kinney, 6th Ed: 4.6
- Roth&Kinney, 6th Ed: 4.8
- Roth&Kinney, 6th Ed: 4.10
- Roth&Kinney, 6th Ed: 4.13
- Roth&Kinney, 6th Ed: 4.15
- Roth&Kinney, 6th Ed: 4.18
- Roth&Kinney, 6th Ed: 4.26
- Roth&Kinney, 6th Ed: 4.29
- Roth&Kinney, 6th Ed: 4.32
- Roth&Kinney, 6th Ed: 4.39
- Roth&Kinney, 6th Ed: 4.41
- Roth&Kinney, 6th Ed: 5.5
- Roth&Kinney, 6th Ed: 5.8
- Roth&Kinney, 6th Ed: 5.10
- Roth&Kinney, 6th Ed: 5.11
- Roth&Kinney, 6th Ed: 5.12
- Roth&Kinney, 6th Ed: 5.13
- Roth&Kinney, 6th Ed: 5.20
- Roth&Kinney, 6th Ed: 5.28
- Roth&Kinney, 6th Ed: 5.30
- Roth&Kinney, 6th Ed: 5.32
- Roth&Kinney, 6th Ed: 5.33
- Roth&Kinney, 6th Ed: 5.34
- Roth&Kinney, 6th Ed: 5.42

2 Problem Set Part B

1 (Boolean Algebra)

Fill in the following proof and the name of the theorem (if any) applied at the last step. Also, when the Consensus Theorem is applied, please underline the terms you used to generate or eliminate another term. A list of Boolean axioms and theorems can be found in the back of your homework.

Part (A)

$$a'c'd + bd + abc' + bc$$

=

Consensus Theorem

=

Absorption

$$= a'c'd + bcd + bc'd + bc + ab$$

Simplification Theorem 1

=

Simplification Theorem 1

$$= a'c'd + bc + ab$$

Part (B)

$$(b + c')(a' + b + d')(a + b' + c + d)$$

=

Distributivity

=

Distributed Consensus Theorem

$$= ab + bc + bd + a'b'c' + b'c'd'$$

Distributivity

=

Consensus Theorem

=

Consensus Theorem

=

Simplification Theorem 1

=

Absorption

$$= a'b'c' + ac'd' + bd + bc$$

2 (Implication and Inhibition)

Implication and Inhibition had never been popular gates at CircuitBoard High. Both of them had a slight bodily asymmetry, for which they were often made fun of by the other gates, and which made it difficult for them to commute by foot between classes. Furthermore, neither of them were particularly associative – Inhibition was a shy and inhibited boy, while Implication was always implicating herself in altercations with the other gates.

One day, the two happened to meet on the silicon playground. They realized that their respective strengths and shortcomings were complementary. They continued meeting up with each other after school every day. It did not take long for their friendship to bloom into an adorable young romance.

Eventually, the PROM rolled around. Seeing that they had each unexpectedly managed to acquire a willing date, they decided to enroll in a dance class together. Their instructor assured them that together, they would be able to perform all of the same dance moves that the other gates could do – especially NANDrew and NORa, whom everyone expected to win PROM King and Queen.

In this problem, you will be helping Implication and Inhibition to learn all of the essential steps of the Universal Circuit Synthesis Dance. Really, you will be proving that together, the two-input Inhibition and Implication gates can be composed to form any other two-input boolean function. Observe that Implication and Inhibition can be expressed as follows:

$$\begin{array}{llll} \text{Definition of Implication:} & x \rightarrow y & = & x' + y \quad \text{“}x \text{ implies } y\text{”} \\ \text{Definition of Inhibition:} & x/y & = & xy' \quad \text{“}y \text{ inhibits } x\text{”, or “}x \text{ unless } y\text{”} \end{array}$$

(Part A) In order to get started, the couple has to learn how to perform the most basic moves in the dance. Help them to learn the following three steps (WITHOUT the help of inverters or any other gates besides Implication and Inhibition):

- i. Given two arbitrary inputs x and y , how can Implication and/or Inhibition output a “1”? How would they output a “0”? (This behavior should NOT depend on the particular values of x and y .)
- ii. Given an arbitrary input x , how can Implication/Inhibition produce x' ? (Hint: the result of part A.i may prove useful.)
- iii. Given two arbitrary inputs x and y , how can Implication/Inhibition produce $x \bullet y$? $x + y$? (Hint: the result of part A.ii may prove useful.)

(Part B) Now that the couple can simulate the behavior of the AND, OR, and Inverter gates, they are fully capable of representing any boolean function. (To celebrate this milestone, they made their relationship universally known on gatebook.com.) However, it turns out that if we want to implement a generic boolean function (implemented using AND, OR, and Inverters) with Implication and Inhibition, applying the conventions in Part A is not actually an optimal strategy. Instead, the following rules provide a good guideline:

Rule 1: If F has the form $F = A + B$, where A and B are sub-expressions, then F can be decomposed into $A' \rightarrow B$ (or equivalently $B' \rightarrow A$).

Rule 2: If F has the form $A \bullet B$, then F can be decomposed into A/B' (or equivalently B/A').

Any boolean function which is in standard form can be recursively decomposed in this manner until each of the sub-expressions are single variables, possibly with inverters. As you know from Part A.ii, implementing an inverter using Implication and Inhibition is expensive in terms of both gate-count and delay. Fortunately, some functions conveniently decompose in such a way that there are no bottom-level inverters. The XOR function is such a function.

Using the two rules described, decompose the XOR function, $x \oplus y = xy' + x'y$, into an Implication/Inhibition function. (You may use the table provided to guide your decomposition, or you may do an original derivation, so long as we can clearly follow the steps of your derivation.)

Definition of $\oplus$	$xy' + x'y$
Rule 1	
DeMorgan's Laws	
Rule 1	
Rule 2	$(x \rightarrow y) \rightarrow (y/x)$

(Part C) How can the Implication/Inhibition expression of XOR in part B be modified to instead produce the XNOR function, without changing the number of gates used (though Implication gates may be exchanged with Inhibition gates, and vice versa) and without introducing any inverters?

(Part D) *On the night of the PROM, Implication and Inhibition arrived and began dancing. Everyone was shocked to see them at all, let alone performing as well as the other gates. Delay for delay, transistor for transistor, they were able to keep up with all of the other gates during the Universal Synthesis Dance (although NANDrew and NORa were still slightly more fluent).*

Finally, it came time for the Full Subtractor Swing. Even for NANDrew and NORa, this one was tough, and neither of them were able to efficiently perform it without getting help from the inverter. Implication and Inhibition looked at each other. Simultaneously, they realized they could implement the subtractor without the use of a single inverter. They were about to make PROM history.

The dancers are supposed to synthesize the circuit for the following difference and borrow-out equations:

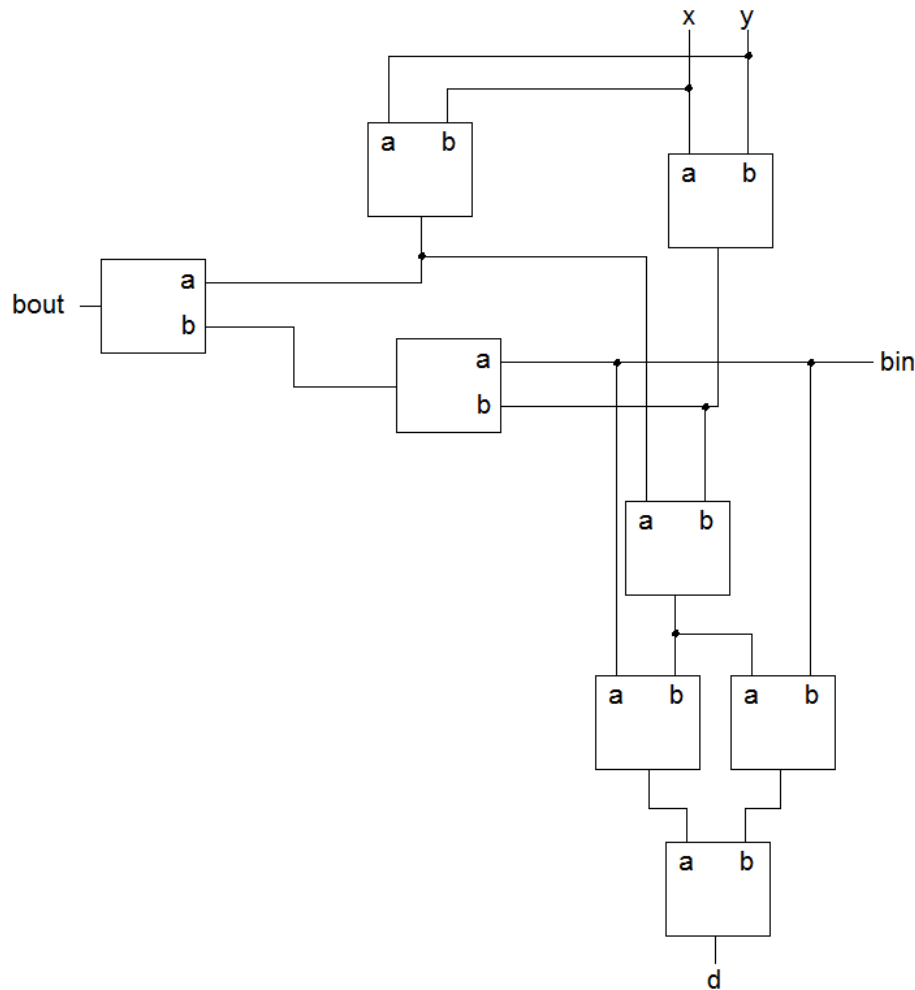
$$d_i = x_i \oplus y_i \oplus b_i = x_i \odot y_i \odot b_i$$

$$b_{i+1} = x'_i y_i + b_i(x'_i + y_i)$$

Convert these boolean expressions to ones that use ONLY Implication and Inhibition. We ask that you produce an implementation of the b_{i+1} function that uses no more than 4 gates, so as to facilitate the correct completion of the subtractor in the next part of this question.

(Part E) As you remember from class, it is possible for NANDrew and NORa to optimize their implementation of a Full Adder to use only 8 gates. Analogously, it is possible for Implication and Inhibition to optimize their implementation of the equations generated in Part D in a way that uses 8 Implication and/or Inhibition gates as well. This is done by observing that there are shared sub-expressions between the d_i and b_{i+1} equations, and that thus the portion of the circuit that computes those expressions can be shared between the equations. (If such sub-expressions do not appear in the equations you derived in Part D, keep in mind that in the d_i equation, x_i , y_i , and b_i can be permuted in any order, since XOR and XNOR are both commutative and associative. Furthermore, in the b_{i+1} equation, the b_i and y_i can be exchanged without sacrificing correctness.)

Show how Implication and Inhibition can use the results of Part D, as well as the ideas in the previous paragraph, to complete the schematic of the Full Subtractor presented below, and thereby go down in PROM history as the first couple to synthesize the Full Subtractor Swing with no inverter gates, by specifying whether each gate represents the function $a \rightarrow b$, $b \rightarrow a$, a/b , or b/a .



3 (Karnaugh Maps)

After the bang up job you did on using the new adder chip on Excess-3 addition, your future in hardware design is bright (on the other hand, the intern's attempts to smear you eliminated all his future prospects). With this in mind your boss assigns you a new task related to decimal encoding systems.

Straight out of the most secretive research labs, the **6311** decimal encoding system is causing quite a stir. A four bit binary code with bit weights 6, 3, 1, and 1 enables all decimal numbers to be encoded. One slight flaw is that there are multiple codes for certain digits (e.g., 1 can be coded as 0001 or 0010); however, this is fixed by declaring "Standard" numbers as using the rightmost 1 if a single one's place is sufficient (0001 is standard whereas 0010 is not). This system has no useful arithmetic properties, leading to highly obfuscated addition and subtraction correction behavior, making it an ideal addition to your company's new encryption chips.

In order to integrate this encoding system into your new chips, your boss has decided that a 6311 9's complement logic board is needed. Knowing you are an expert in decimal encoding systems, you have been assigned to design the boolean function to perform this task. As a reminder, 9's complement is a base-10 complementing system such that each digit is subtracted from 9. Hence the 9's complement of the number 43 is 56 and the 9's complement of 8 is 1.

A table showing the standard 6311 codes for the digits 0-9 as well as their 9's complement is at the end of this exam.

(Part A) As a first task, you are to fill in the following four Karnaugh maps for the 9's complement ($y = y_3y_2y_1y_0$) of a 6311 digit $x = x_3x_2x_1x_0$. Put a 0 for every maxterm, 1 for every minterm, and X for every *don't care*. Also please circle the *prime implicants*.

y_3

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00				
01				
11				
10				

y_2

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00				
01				
11				
10				

y_1

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00				
01				
11				
10				

y_0

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00				
01				
11				
10				

(Part B) Using your newly created Karnaugh maps, please write the *minimal* sum-of-products and product-of-sums forms for each variable.

Output variable	Sum-of-Products	Product-of-sums
y_3		
y_2		
y_1		
y_0		

(Part C) One senior engineer thinks it might be useful to accept non-standard 6311 encoded numbers as *input*. A non-standard number can have a 1 in either of the last two digits. For example, 4 can be represented by either a 0101 or 0110. Given you still want to output the “standard” 6311 digit (with rightmost bit having preference), fill in the following Karnaugh maps with the change this will cause, circle the *prime implicants*, and write the minimal sum-of-products form for each variable.

y_3

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00				
01				
11				
10				

y_2

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00				
01				
11				
10				

y_1

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00				
01				
11				
10				

y_0

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00				
01				
11				
10				

Output variable	Sum-of-Products
y_3	
y_2	
y_1	
y_0	

The list of Boolean algebra **axioms** follows:

Axiom 1 (Closure property):

- (a) B is closed with respect to the operator $+$;
- (b) B is also closed with respect to the operator $\cdot$;

Axiom 2 (Identity element):

- (a) B has an identity element i with respect to $\cdot$, such that $B \cdot i = B$;
- (b) B also has an identity element j with respect to $+$, such that $B + j = B$;

Axiom 3 (Commutativity Property):

- (a) B is commutative with respect to $+$;
- (b) B is also commutative with respect to $\cdot$;

Axiom 4 (Distributivity Property):

- (a) The operator $\cdot$ is distributive over $+$;
- (b) Similarly, the operator $+$ is distributive over $\cdot$;

Axiom 5 (Complement Element): For every $x \in B$, there exists an element $x' \in B$, the complement of x , such that:

- (a) $x + x' = i$, where i is the identity element with respect to the $\cdot$ operator, and
- (b) $x \cdot x' = j$, where j is the identity element with respect to the $+$ operator.

Axiom 6 (Cardinality Bound): There are at least two elements $x, y \in B$ such that $x \neq y$.

A list of useful Boolean algebra **theorems** that you may need follows; the duals also hold, but they are not expected to be needed:

Consensus Theorem $XY + X'Z + YZ = XY + X'Z$

Distributed Consensus Theorem $XY + X'Z = (X + Z)(X' + Y)$

Simplification Theorem 1 $XY + XY' = X$

Simplification Theorem 2 $X + XY = X$

You can also use any of the following theorems if you find them useful:

Theorem 1 (Idempotency):

- (a) $x + x = x$ and
- (b) $x \cdot x = x$.

Theorem 2 (Identity Absorption):

- (a) $x + i = i$, where i is the identity element with respect to the $\cdot$ operator, and
- (b) $x \cdot j = j$, where j is the identity element with respect to the $+$ operator.

Theorem 3 (Absorption, also known as Simplification Theorem 2):

- (a) $yx + x = x$ and
- (b) $(y + x)x = x$.

Theorem 4 (Involution): $(x')' = x$

Theorem 5 (Associativity):

- (a) $x + (y + z) = (x + y) + z$ and
- (b) $x(yz) = (xy)z$.

Theorem 6 (De Morgan's law):

- (a) $(x + y)' = x'y'$ and
- (b) $(xy)' = x' + y'$.