CSE 20, Fall 2020 - Homework 3

Due: Monday 10/26 at 11 am PDT

Instructions

Upload a single file to Gradescope for each group. All group members' names and PIDs should be on each page of the submission. Your assignments in this class will be evaluated not only on the correctness of your answers, but on your ability to present your ideas clearly and logically. You should always explain how you arrived at your conclusions, using mathematically sound reasoning. Whether you use formal proof techniques or write a more informal argument for why something is true, your answers should always be well-supported. Your goal should be to convince the reader that your results and methods are sound.

Reading Rosen **Section 1.1** Definition 5, Definition 6, Example 11, **Section 1.4** Definitions 1 (p. 40) and 2 (p. 42), Table 2 (p 47), Examples 21-22 (pp. 47-48), **Section 2.1** Definitions 8 and 9 (p. 123). **Section 1.5** Example 1 (p. 57) and Example 4 (p. 59)

Key Concepts Conditionals, Quantifiers and Predicates

Problem 1 (20 points)

a) Convert the following compound propositions into an equivalent statement using only the operators AND ($\wedge$), OR ($\vee$) and NOT ($\neg$)
   1) $(p \leftrightarrow q) \rightarrow r$
   2) $p \rightarrow (q \rightarrow (r \rightarrow (s \rightarrow t)))$

b) Express the negations of the following statements such that all negation symbols ($\neg$) immediately precede the predicate (i.e. write the negations as: $\neg(\exists x P(x)) \equiv \forall x(\neg P(x))$). Express your answer using equivalent statement using only the operators AND ($\wedge$), OR ($\vee$) and NOT ($\neg$).
   1) $\forall x \exists y P(x,y) \ \vee \ \exists x \forall y Q(x,y)$
   2) $\forall x \exists y (P(x,y) \ \rightarrow Q(x,y))$

Solutions:

a) We will use the following identities for this solution:

$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

$p \rightarrow q \equiv \neg p \vee q$

As needed, we will also use De-Morgan's law to simplify our final answers (this is not a requirement).

1) $$\begin{aligned}(p \leftrightarrow q) \rightarrow r &\equiv \neg(p \leftrightarrow q) \vee r \\ &\equiv \neg((p \rightarrow q) \wedge (q \rightarrow p)) \vee r \\ &\equiv \neg((\neg p \vee q) \wedge (\neg q \vee p)) \vee r\end{aligned}$$

This is an acceptable final solution. If you wish to further simplify this, you could use De-Morgan's law on the negation:

$$\begin{aligned}\neg((\neg p \vee q) \wedge (\neg q \vee p)) \vee r &\equiv \neg(\neg p \vee q) \vee \neg(\neg q \vee p) \vee r \\ &\equiv (p \wedge \neg q) \vee (q \wedge \neg p) \vee r\end{aligned}$$

2) Here, we will work from the outermost parentheses, and work inwards:

$$\begin{aligned}p \rightarrow (q \rightarrow (r \rightarrow (s \rightarrow t))) &\equiv \neg p \vee (q \rightarrow (r \rightarrow (s \rightarrow t))) \\ &\equiv \neg p \vee (\neg q \vee (r \rightarrow (s \rightarrow t))) \\ &\equiv \neg p \vee (\neg q \vee (\neg r \vee (s \rightarrow t))) \\ &\equiv \neg p \vee (\neg q \vee (\neg r \vee (\neg s \vee t)))\end{aligned}$$

And since these are all OR operators individually, we can use the associative property to write:

$$p \rightarrow (q \rightarrow (r \rightarrow (s \rightarrow t))) \equiv \neg p \vee \neg q \vee \neg r \vee \neg s \vee t$$

b) This question will involve the concept of negation of quantifiers and De-Morgan's law

1) $\neg(\forall x \exists y P(x,y) \vee \exists x \forall y Q(x,y)) \equiv \neg(\forall x \exists y P(x,y)) \wedge \neg(\exists x \forall y Q(x,y))$
$\equiv \exists x \forall y \neg P(x,y) \wedge \forall x \exists y \neg Q(x,y))$

2) $\neg(\forall x \exists y (P(x,y) \rightarrow Q(x,y))) \equiv \exists x \forall y \neg (P(x,y) \rightarrow Q(x,y))$
$\equiv \exists x \forall y \neg (\neg P(x,y) \vee Q(x,y))$
$\equiv \exists x \forall y (\neg(\neg P(x,y)) \wedge \neg Q(x,y))$
$\equiv \exists x \forall y (P(x,y) \wedge \neg Q(x,y))$

Problem 2 (20 points)

In this question, you'll define and work with predicates defined over the set of integers A = {-1,0,1}. This will be your domain.

Recall the definition: **Statements involving predicates and quantifiers are logically equivalent means they have the same truth value no matter which predicates (domains and functions) are substituted in.**

For full credit, your solution for each part below needs to include all of the following:

1) Precise definition of predicate(s)
2) Evaluation of left-hand-side quantified statement, with explanations referring to definition of specific predicate and logical structure of statement
3) Evaluation of right-hand-side quantified statement, with explanations referring to definition of specific predicate and logical structure of statement
4) Conclusion, with explanations

a) Using an input-output definition table, define a predicate P over A so that it can be used to prove that $\exists y \forall x P(x,y)$ is not equivalent to $\forall x \exists y P(x,y)$

b) Using an input-output definition table, define a predicate Q over A so that it can be used to prove that $\forall x \exists y Q(x,y)$ is not equivalent to $\exists x \forall y Q(x,y)$

c) Determine which of the following are true logical equivalences, and informally justify your reasoning (the predicates P and Q here are independent of the previous 2 parts of the question):

1) $\exists x P(x) \vee \exists x Q(x) \equiv \exists x (P(x) \vee Q(x))$
2) $\forall x P(x) \vee \forall x Q(x) \equiv \forall x (P(x) \vee Q(x))$

Solutions:

As a broad guideline for this question, there is no one correct definition of a predicate. You can come up with any suitable definition that allows you to solve the question.

a) In this part, we have to prove that the order of the quantifiers matters
$\exists y \forall x P(x,y) \neq \forall x \exists y P(x,y)$

1) Let us define the predicate $P(x,y)$ as: $x^2 = y$
2) Let us consider the left-hand side: $\exists y \forall x P(x,y)$

This is clearly **False**. To prove this, we exhaustively check each possible value for y:

$y = -1$ is false since $x = 0$ yields $x^2 = 0 \neq -1$

$y = 0$ is false since $x = 1$ yields $x^2 = 1 \neq 0$

$y = 1$ is false since $x = 0$ yields $x^2 = 0 \neq 1$

3) Let us consider the right-hand side: $\forall x \exists y P(x, y)$
   We will show that this is true, by exhaustively checking all the values of x

   $x = -1$ is true since $y = 1$ yields $x^2 = 1 = y$

   $x = 0$ is true since $y = 0$ yields $x^2 = 0 = y$

   $x = 1$ is true since $y = 1$ yields $x^2 = 1 = y$
4) In conclusion, we have seen that the left-hand side of the proposition evaluates to **False** whereas the right-hand side evaluates to **True**. Hence, we have shown with a counter-example that the two propositions are unequal. In general, we prove that the order of quantifiers is important in an expression.

b) In this part, we have to prove that the order of variables also matters
   $\forall x \exists y Q(x, y) \neq \exists x \forall y Q(x, y)$
   1) Let us define the predicate $Q(x, y)$ as: $x = y$
   2) Let us consider the left-hand side: $\forall x \exists y Q(x, y)$
      This is **True**. To prove this, we exhaustively check each possible value for x:

      $x = -1$ is true since $y = -1$ yields $x = y$

      $x = 0$ is true since $y = 0$ yields $x = y$

      $x = 1$ is true since $y = 1$ yields $x = y$

   3) Let us consider the right-hand side: $\exists x \forall y Q(x, y)$
      We will show that this is **False**, by exhaustively checking all the values of x

      $x = -1$ is false since $y = 0, 1$ will both yield $x \neq y$, and hence, at $x = -1$, this will not be universally true in the domain A.

      $x = 0$ is false since $y = -1, 1$ will both yield $x \neq y$, and hence, at $x = 0$ as well, this will not be universally true in the domain A.

      $x = 1$ is false since $y = 0, -1$ will both yield $x \neq y$, and hence, at $x = 1$, this will not be universally true in the domain A.
   4) In conclusion, we have seen that the left-hand side of the proposition evaluates to **True** whereas the right-hand side evaluates to **False**. Hence, we have shown with a counter-example that the two propositions are unequal. In general, we prove that the quantifier in front of each variable is also important.

c) This part will only require an informal proof, as stated in the question.
   1) $\exists x P(x) \vee \exists x Q(x) \equiv \exists x (P(x) \vee Q(x))$

This **is** a logical equivalence. Informally, the existential quantifier acts similar to disjunction, and disjunction is associative and commutative. Remember, grouping the "ORs ($\vee$)" and rearranging them does not affect the output truth value.

Another way of analyzing this is that the left-hand-side quantified statement is made true when at least one of the predicates P and Q have a witness for them being true. At the same time, the right-hand-side quantified statement is made true when there's a witness for either P and Q being true. These are two ways of describing the same condition.

2) $\forall x P(x) \vee \forall x Q(x) \equiv \forall x (P(x) \vee Q(x))$

This **is not** a logical equivalence, since the universal quantifier is not associative. The right-hand side of the proposition evaluates to true when at least one of the predicates P and Q evaluate to true for all x, but neither has to individually evaluate to true for all x. However, the left-hand side evaluates to true only when one of P or Q evaluates to true for all x, which is not the same as the right-hand side.

Problem 3 (20 points)

Let's say you are designing a file management system defined using the following statements:

1. If the file system is not locked, then new messages will be queued.
2. If the file system is not locked, then the system is functioning normally, and conversely.
3. If new messages are not queued, then they will be sent to the message buffer.
4. If the file system is not locked, then new messages will be sent to the message buffer.
5. New messages will not be sent to the message buffer.

a) Defining appropriate propositional variables, translate each statement into a compound proposition.
b) Determine if the system defined here is consistent.

Solutions:

a) Let's define the system variables as:
A: "The file system is locked"
B: "The system is functioning normally"
C: "New messages are queued"
D: "New messages will be sent to the buffer"

Therefore, we can define the system as:
1) If the file system is not locked, then new messages will be queued: $\neg A \rightarrow C$
2) If the file system is not locked, then the system is functioning normally, and conversely: $\neg A \leftrightarrow B$

3) If new messages are not queued, then they will be sent to the message buffer: $\neg C \rightarrow D$
4) If the file system is not locked, then new messages will be sent to the message buffer: $\neg A \rightarrow D$
5) New messages will not be sent to the message buffer: $\neg D$

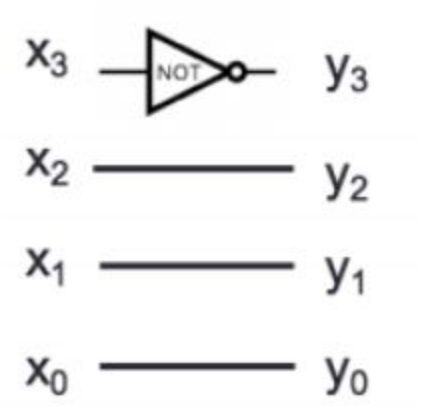
b) For this part, you need to just provide one set of truth value assignments to each of the 4 propositional variables that make each of the 5 statements true. The solution will, however, list out the entire truth table:

A	B	C	D	$\neg A \rightarrow C$	$\neg A \leftrightarrow B$	$\neg C \rightarrow D$	$\neg A \rightarrow D$	$\neg D$
F	F	F	F	F	F	F	F	T
F	F	F	T	F	F	T	T	F
F	F	T	F	T	F	T	F	T
F	F	T	T	T	F	T	T	F
F	T	F	F	F	T	F	F	T
F	T	F	T	F	T	T	T	F
F	T	T	F	T	T	T	F	T
F	T	T	T	T	T	T	T	F
T	F	F	F	T	T	F	T	T
T	F	F	T	T	T	T	T	F
T	**F**	**T**	**F**	**T**	**T**	**T**	**T**	**T**
T	F	T	T	T	T	T	T	F
T	T	F	F	T	F	F	T	T
T	T	F	T	T	F	T	T	F
T	T	T	F	T	F	T	T	T
T	T	T	T	T	F	T	T	F

We observe that $A \equiv T,\ B \equiv F,\ C \equiv T,\ D \equiv F$ makes all the propositions true, and hence the system **is consistent**.

Problem 4 (20 points)

Consider the following circuit with four inputs x_0, x_1, x_2, x_3 and four outputs y_0, y_1, y_2, y_3:



a) Does this logic circuit implement the operation of taking a number represented in width-4 sign-magnitude representation and producing the width-4 sign-magnitude representation of (−1) times this number? If yes, explain why using the definition of sign-magnitude representation and consider all possible inputs. If no, provide specific example input values, trace the circuit to compute its output for this example, and use the definition to calculate the two numbers being represented and explain why these numbers support your conclusion.

b) Does this logic circuit implement the operation of taking a number represented in width-4 sign-magnitude representation and producing the width-4 2s complement representation of (−1) times this number? If yes, explain why using the definition of 2s complement representation and consider all possible inputs. If no, provide specific example input values, trace the circuit to compute its output for this example, and use the definition to calculate the two numbers being represented and explain why these numbers support your conclusion.

Solutions:

a) Yes, this circuit implements this operation. To see this, we can organize our calculations in a table: the columns labelled with x's are all possible inputs; the columns labelled with y's are calculated by tracing the circuit using the definition of the inverter gate; the other intermediate columns use the definition of sign-magnitude representation. Namely, x_3 is the sign bit (1 means multiply the magnitude by −1 and 0 means multiply the magnitude by 1) and the magnitude of the number being represented is calculated as $x_2 \cdot 2^2 + x_1 \cdot 2^1 + x_0 \cdot 2^0$

x_3	x_2	x_1	x_0	$[x_3x_2x_1x_0]_{s,4}$	y_3	y_2	y_1	y_0	$[y_3y_2y_1y_0]_{s,4}$	$Out = (-1)Input?$
1	1	1	1	−7	0	1	1	1	7	Yes
1	1	1	0	−6	0	1	1	0	6	Yes
1	1	0	1	−5	0	1	0	1	5	Yes
1	1	0	0	−4	0	1	0	0	4	Yes
1	0	1	1	−3	0	0	1	1	3	Yes
1	0	1	0	−2	0	0	1	0	2	Yes
1	0	0	1	−1	0	0	0	1	1	Yes
1	0	0	0	0	0	0	0	0	0	Yes
0	1	1	1	7	1	1	1	1	−7	Yes
0	1	1	0	6	1	1	1	0	−6	Yes
0	1	0	1	5	1	1	0	1	−5	Yes
0	1	0	0	4	1	1	0	0	−4	Yes
0	0	1	1	3	1	0	1	1	−3	Yes
0	0	1	0	2	1	0	1	0	−2	Yes
0	0	0	1	1	1	0	0	1	−1	Yes
0	0	0	0	0	1	0	0	0	0	Yes

b) No, this circuit does not implement this operation. For an example, consider the input to the circuit $x_3 = 1,\ x_2 = 0,\ x_1 = 0,\ x_0 = 0.$ The outputs of the circuit are $y_3 = 0,\ y_2 = 0,\ y_1 = 0,\ y_0 = 0$. Using the definition of 2s complement, the input represents the value $(1000)_{2c,4} = 1 \cdot (-2^3) + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = -8$ and the output represents the value $(0000)_{2c,4} = 0 \cdot (-2^3) + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 0$. Since $-8 \neq 0$, we have found an example to demonstrate that the circuit does not implement the operation of multiplying by −1 when the strings are interpreted in 2s complement width 4.

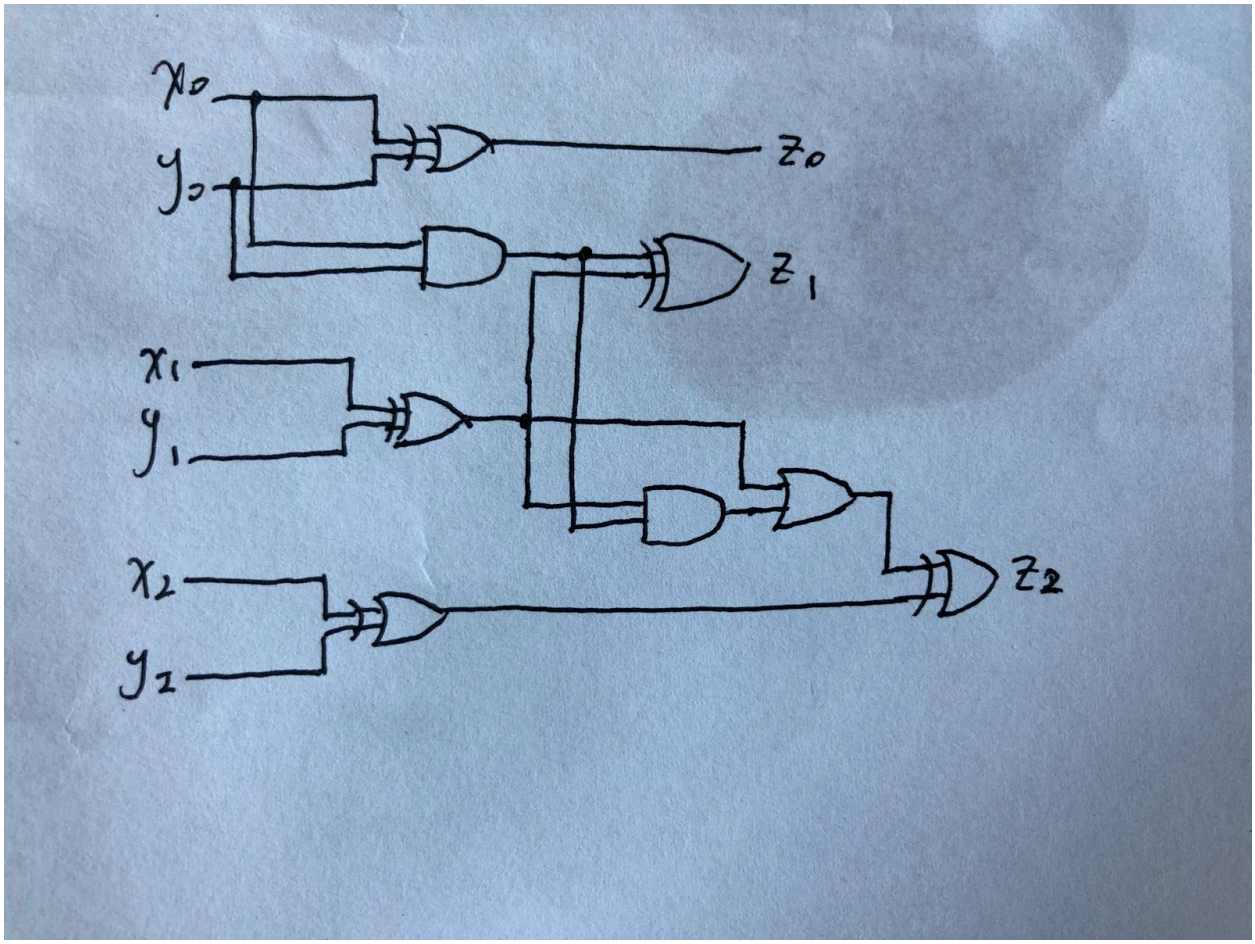
Problem 5 (20 points)

In this problem, we consider building a circuit to implement the summation of 2 width-3 signed numbers. Mathematically, this problem can be formulated as follows:
Given 2 inputs $(x_2x_1x_0)_2,\ (y_2y_1y_0)_2$, implement a circuit to calculate the summation $(z_2z_1z_0)_2$
Assume all negative numbers are using 2s complement representation.

a) Design the circuit and draw it.
b) Verify the correctness of your design. You can do this by representing $z_2,\ z_1, z_0$ as a function of $x_2,\ x_1,\ x_0,\ y_2,\ y_1,\ y_0$ and calculate the entire truth table.

Solutions:

a) The circuit can take the same form as the one we used for width-3 unsigned number summation. In short, we have the following:
$z_0 = x_0 \oplus y_0,$
$z_1 = x_1 \oplus y_1 \oplus (x_0 \wedge y_0),$
$z_2 = x_2 \oplus y_2 \oplus c_1 = x_2 \oplus y_2 \oplus ((x_1 \wedge y_1) \vee ((x_1 \oplus y_1) \wedge c_0)) = x_2 \oplus y_2 \oplus ((x_1 \wedge y_1) \vee ((x_1 \oplus y_1) \wedge (x_0 \wedge y_0)))$



b) There're various ways to verify the correctness. The most straightforward (yet most tedious) way is to calculate the truth table. Another way, which requires some thinking, is to verify that the circuit can take exactly the same form as the circuit we used for unsigned 3-bit summation. Recall that for a signed integer, the left-most digit has a negative weight. Other than that, the binary representation for signed integers is completely the same as that for unsigned integers. Thus, the circuit to calculate $z_1 \& z_0$ can be the same as the circuit we used to calculate these 2 digits in the unsigned case. As for the leftmost digit, if there's no carry from second-from-left digit, then $z_2 = x_2 \oplus y_2$. If there's carry, then we should add $2^2 = 4$ to the summation result. If we directly adopt the circuit for the leftmost digit from unsigned summation, then we are actually adding $-2^2 = -4$ (the negative sign comes from the negative weight for the leftmost digit) to the result. Though 4 and -4 seem to differ a lot, recall that with only 3 bits, we can only represent $2^3 = 8$ values, so any 2 numbers with a difference of 8's multiplication should share exactly the same binary representation. Thus, we can justify that we can directly borrow the circuit for calculating signed 3-bit summation.

Problem 6 (10 points)

In this problem, you're supposed to use only and, or, xor, and negation gates to build up circuits that satisfy specific requirements.

- Please refer to Rosen p.25 for definitions of tautology and contradiction.

a) Suppose there's only one input signal. Design a circuit made up of only one gate that represents a contradiction. If you think such a circuit doesn't exist, please provide some reasoning.

b) Suppose there's only one input signal. Design a circuit made up of only one gate that represents a tautology. If you think such a circuit doesn't exist, please provide some reasoning.

Solutions:

a) Use an 'xor' gate and connect the input signal to both input pins of the gate. The output for the gate should always represent 'false', which means a contradiction.

b) Such a circuit doesn't exist. Since we have only one input signal (instead of two), we must connect the signal to both input pins of the gate that we choose. If you enumerate all gates allowed to use in this problem, you'll find that no one yields a tautology result.