Challenge Problems: Submission Guidelines and Samples
MATH2603: Discrete Mathematics

This document contains important information about the expectations for submissions of Challenge Problems. I suggest that you read these guideline prior to starting your first challenge problem. Then keep this document close at hand while completing your first few submissions.

General Guidelines

What will I submit?

Your work must be neatly typed up using a system that supports mathematical notation. For example, you can use MS Word and its equation editor; or you can write your work in a Jupyter notebook using Markdown and LaTeX; or you can create a pdf using LaTeX. Once it is written up, the work must be saved as a PDF file and then uploaded as a PDF to the area on Blackboard where the original assignment is located. Remember that the work is not actually submitted until you upload the file and click the "Submit" button. Grading and feedback will take place entirely on Blackboard. The following are not allowed: Submissions outside Blackboard (for example through email); files that are not in PDF form; and work that contains any handwriting. If you need to make a diagram or picture, I am happy to suggest some easy-to-learn software that will be suitable for your needs.

Crucially, **you should produce a self-contained document.** This means that a student in another section could understand your solution even if they had never seen the prompt.

How long should my submission be?

There are no length requirements for any Challenge Problem submissions. Your solution should be as long as it needs to be in order to give relevant background/restate the problem; answer any and all questions required by the prompt present your solution and accompanying explanations (in complete English sentences) to all questions required by the prompt; and make any conclusions you think are warranted. For most assignments, this will take 1-2 pages of typed work.

Who is my audience?

The first question you should ask yourself (or your instructor/boss) before creating a written document is "Who is my audience?" The answer to this question should inform the language that you use, the quantity and depth of background material you include, and the way you explain your work.

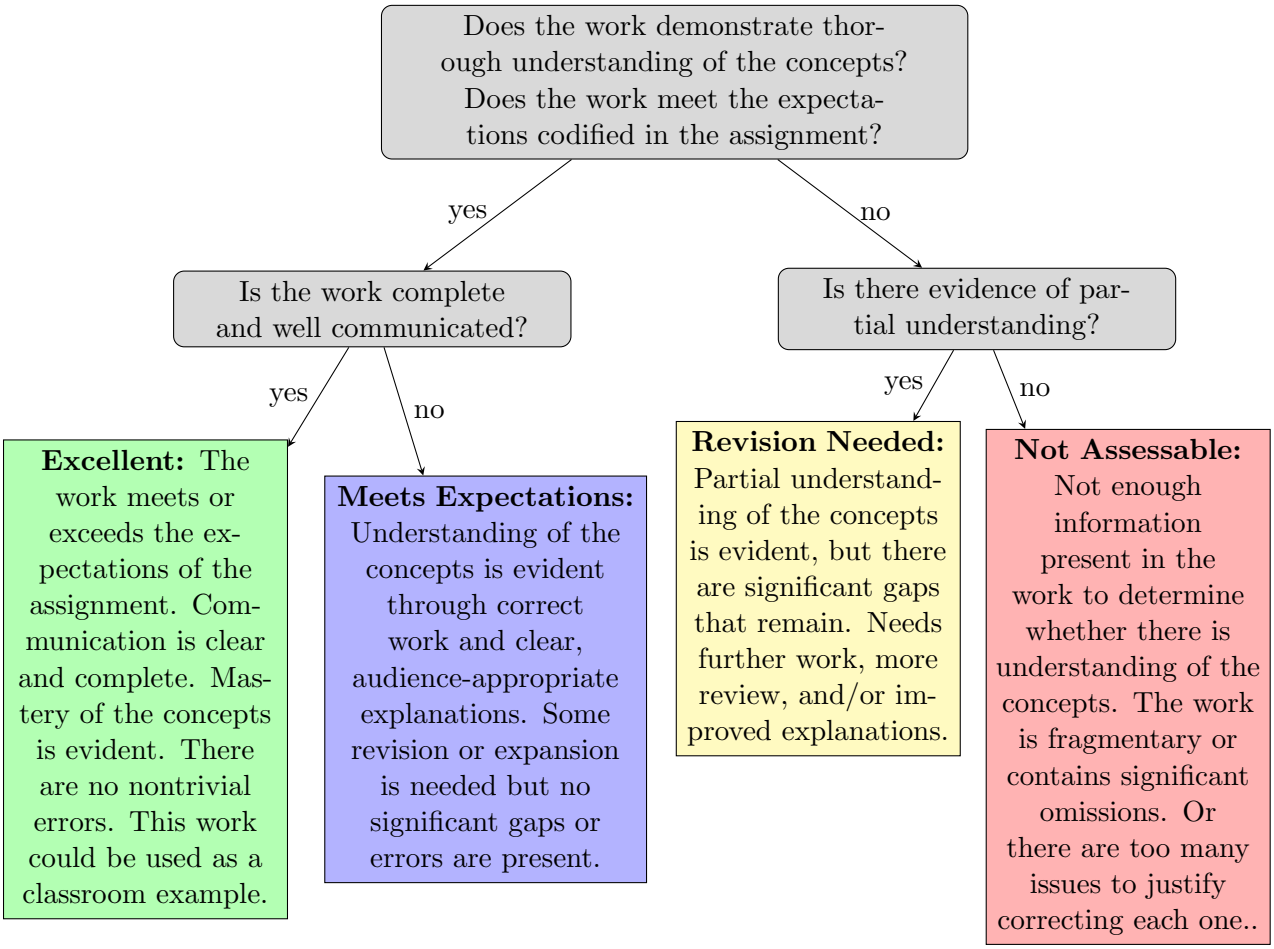
The intended audience for your solution is a classmate. This means you may assume your reader is familiar with (but may need a gentle reminder of) any relevant definitions or material covered in the book/class.

As an example, suppose a Challenge Problem asked you to evaluate a triple integral of a function over a 3-dimensional region, and your solution involves the use of the Divergence Theorem. It would **not be appropriate** for you to explain what a triple integral is, or what a parameterization of a surface is. On the other hand, it would be completely appropriate to remind the reader what the

the Divergence Theorem says; to explain how and why it applies in this problem; to explain how you came up with your surface parameterization; and to explain the main steps in your calculations. You would not need to explain, however, that an antiderivative of $\sin(x)$ is $-\cos(x)$.

How will my work be evaluated?

All Challenge Problems will be evaluated using the EMRN rubric from the syllabus (reproduced below). This rubric classifies the work with marks of **E** ("Excellent"), **M** ("Meets Expectations"), **R** ("Needs Revision"), or **N** ("Not Assessable"). Additionally, each Challenge Problem prompt will have specific explanations that explain how this rubric applies to the problem at hand. See the sample Challenge Problem below for an example.



How will I receive feedback?

As mentioned earlier, all solutions will be submitted via Blackboard. You will also receive feedback via Blackboard. Since Blackboard uses a points system, the conversion between your score (out of 3) on Blackboard, and the rubric discussed above is given by the following table

Blackboard Score	Rubric Score
3	**E**xcellent
2	**M**eets Expectations
1	**R**evision Needed
0	**N**ot Assessable

You will also receive extensive written feedback in the Blackboard Grade center. This feedback will address any and all issues with your solution, including mathematical correctness, presentation, typesetting/formatting, quality of explanations, etc.

Can I see an example?

Of course. The remainder of this document consists of a **sample Challenge Problem** from the summer section of Discrete mathematics, followed by four **sample solutions** representing each of the possible rubric items.

As a *warning*, the Challenge Problems for the summer section were a bit longer and more involved that the challenge problems this semester. You should take this into account when you browse the sample solutions.

Samples Table of Contents

Challenge Problem 1: Boolean Algebras

MATH2603: Discrete Mathematics

Overview: In this Challenge Problem, you'll have a chance to explore the connections between mathematical logic, boolean algebras, and computer circuits/architecture by building an addition circuit for a new type of computer!

Instructions: First, read Chapter 14 in the textbook (Boolean Algebras). This material may seem very familiar; much of it is equivalent to propositional logic that we discuss in Chapter 1. Next, complete the exercises below. Your solutions should be complete, clear, and correct. Your solutions should also be accompanied by explanations written in complete sentences that justify your work.

Instructions for submitting your work, and information on how the EMRN rubric will be applied to evaluate this challenge problem, are at the end of the assignment.

Background: Almost all modern computers store information in *bits* (short for binary digit); a bit is a digital version of a boolean varaible, so we will use x or y to refer to bits. Just like a boolean variable, a bit can have values of 0 or 1.

Computers have digital logic gates capable of performing the logical operations of `AND`, `OR`, and `NOT`. These gates, and associated notation, are discussed in the text. You can also find the symbols themselves on Wikipedia. By combining these logic gates in an appropriate way, one can make a *digital adder*: a digital circuit that takes as input two bits x and y, and outputs the sum $x + y$ (using binary addition) in the form of two bits s (for sum) and c (for carry). An input/output table for a digital adder is shown below. A digital circuit that performs this operation, along with additional details, can be found in the §14.4 of the text.

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

The Problem: Casper is an undergraduate at UArk and has been doing cutting edge research computer architectures based on the ternary number system: these computers use ternary digits which can have three values (0, 1, or 2).

1. Casper wants to make an adder for his ternary number system, but he's a bit stuck. Help him out by **making an input/output table for a ternary adder**. Just like the binary adder above, a ternary adder should have two inputs (X and Y) and two outputs (Carry and Sum). This time, there should be three possible values for each input.

2. Casper's research is woefully underfunded, so they wants to build their ternary adder using binary-encoded signals so that two bits are used for each ternary digit. Let the ternary digit X be encoded in the binary digits x_1x_0 so that $00 = (0)_3$, $01 = (1)_3$, and $10 = (2)_3$. Similarly encode Y as bits y_1y_0 and S as s_1s_0. Since the carry digit only takes values 0 and 1, we can encode C using a single bit (which you may also call C).

Design a binary circuit that implements the ternary adder that Casper needs to continue their research. A good first step would be to make an input/output table using binary (as opposed to the ternary table above).

Submitting your work: Your work must be neatly typed up using a system that supports mathematical notation. For example, you can use MS Word and its equation editor; or you can write your work in a Jupyter notebook using Markdown and LaTeX. Once it is written up, the work must be saved as a PDF file and then uploaded as a PDF to the area on Blackboard where the original assignment is located. Remember that the work is not actually submitted until you upload the file and click the Submit button. Grading and feedback will take place entirely on Blackboard. The following are not allowed: Submissions outside Blackboard (for example through email); files that are not in PDF form; and work that contains any handwriting, though you may *draw a diagram* neatly by hand, scan it, and include it in your submission.

Evaluation: Like all Challenge Problems, your work will be evaluated using the EMRN rubric. Please see the statement of this rubric in the syllabus for an explanation of how it is used. When applied to this Challenge Problem, the following criteria help to assign the grade:

- **E:** The solution consists of a clear, correct, and complete solution that includes several paragraphs explaining the binary circuit which implements the ternary adder. The solution contains no major errors (computation, logic, syntax, or semantic); it is also exceptionally clear and the writeup is professional in its look and style. The solution would be at home in a professional lecture or publication.
- **M:** The solution consists of a clear, correct, and complete solution that includes several paragraphs explaining the binary circuit which implements the ternary adder. The solution contains no major errors (computation, logic, syntax, or semantic) and is neatly and professionally written up.
- **R:** The solution contains at least one, but not several, major errors (computation, logic, syntax, and/or semantic) that require revision. An "R" may also be given for writeups that do not expend sufficient effort to produce a good-looking writeup.
- **N:** The solution has several significant errors; or the solution is missing large portions of the solution; or the solution is for a significantly altered version of the problem; or the submission is excessively cluttered, messy, difficult to read, or handwritten.

Student Last Name

Professor Wigglesworth

MATH2603: Discrete Mathematics

4 July 2020

Advanced Exploration 1: Boolean Algebras

The Problem: Casper is an undergraduate at UArk and has been doing cutting edge research computer architectures based on the ternary number system: these computers use ternary digits which can have three values (0, 1, or 2).

1. Casper wants to make an adder for his ternary number system, but he's a bit stuck. Help him out by **making an input/output table for a ternary adder**. Just like the binary adder above, a ternary adder should have two inputs (X and Y) and two outputs (Carry and Sum). This time, there should be three possible values for each input.

X	Y	C	S
0	0	0	0
0	1	0	1
0	2	0	2
1	0	0	1
1	1	0	2
1	2	1	0
2	0	0	2
2	1	1	0
2	2	1	1

   a. In order for Casper to make the ternary adder, he must first list all possible combinations of the two inputs (X and Y) with the three ternary values (0, 1, or 2). The nine possible pairs would then fill the first two columns of the input/output table as shown. Next, Casper would need to use Boolean addition on the inputs to find the values for the outputs, carry and sum. The math is difficult here because with ternary digits there is no number 3 or 4, but, rather the value 3 and 4 are written in ternary as 10 and 11, respectively, with the first digit in the carry column (C) and the second digit in the sum column (S). With this knowledge, Casper can begin adding each X and Y pair to finish making his ternary adder which in the end should look similar to the input/output table above.

2. Casper's research is woefully underfunded, so they want to build their ternary adder using binary-encoded signals so that two bits are used for each ternary digit. Let the ternary digit X be encoded in the binary digits x_1x_0 so that $00 = (0)_3$, $01 = (1)_3$, and $10 = (2)_3$. Similarly encode Y as bits y_1y_0 and S as s_1s_0. Since the carry digit only takes values 0 and 1, we can encode C using a single bit (which you may also call C).

   Design a binary circuit that implements the ternary adder that Casper needs to continue their research. A good first step would be to make an input/output table using binary (as opposed to the ternary table above).

X		Y		C	S	
x_1	x_0	y_1	y_0		s_1	s_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
1	0	0	0	0	1	0
1	0	0	1	1	0	0
1	0	1	0	1	0	1

   a. Given that Casper's research is underfunded and only has the resources to make a binary circuit, he must first translate the ternary input/output table from before into binary. This process is quite tedious because every number must transfer over correctly or the resulting circuit will not function correctly. To begin translating the table, Casper would need to take each data entry from the previous ternary table and transpose them into two bit binary digits, which only includes the values 0 and 1. Be careful when doing this because there is no new addition or math occurring here, rather each ternary digit is being written in binary. For example, the digit 2 in the previous ternary X column is now written as 10, with the 1 residing in the x_1 column and the 0 in the x_0 column. Once Casper's new input/output table is finished using the binary-encoder he should have four inputs (x_1, x_0, y_1, y_0) and three outputs (C, s_1, s_0). The next step in creating the binary circuit is to find the minterm of each row where the output column is 1. To write out a minterm correctly, each input value has to be written as either a Boolean

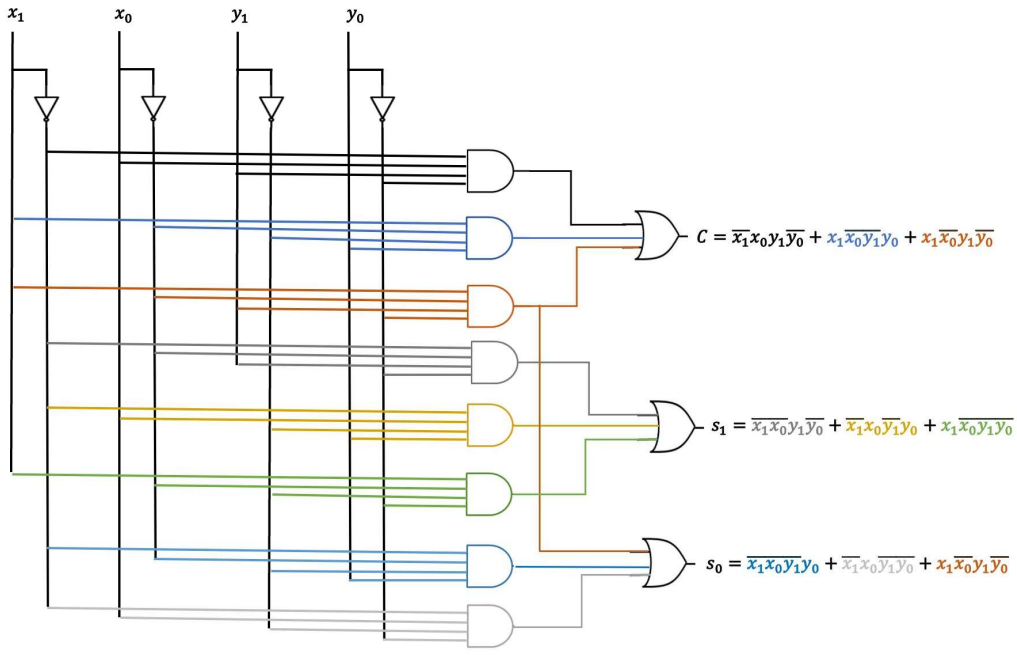
variable (x_1, x_0, y_1, y_0) when the table cell value is 1 or its complement ($\overline{x_1}$, $\overline{x_0}$, $\overline{y_1}$, $\overline{y_0}$) when the table cell value is 0. The minterm will then be shown as the product of the four respective inputs. The Boolean function will then be the sum of the minterms that made up each column. Casper's functions should end up being the following:

$$C = \overline{x_1}x_0y_1\overline{y_0} + x_1\overline{x_0}\overline{y_1}y_0 + x_1\overline{x_0}y_1\overline{y_0}$$

$$s_1 = \overline{x_1}\overline{x_0}y_1\overline{y_0} + \overline{x_1}x_0\overline{y_1}y_0 + x_1\overline{x_0}y_1y_0$$

$$s_0 = \overline{x_1}\overline{x_0}\overline{y_1}y_0 + \overline{x_1}x_0\overline{y_1}\overline{y_0} + x_1\overline{x_0}y_1\overline{y_0}.$$

Now that the Boolean functions have been found it will be easy to create the binary circuit by placing an inverter where a complement is needed, an AND gate where the inputs form the minterm by multiplication, and an OR gate to add the minterms together. When finished, Casper's binary circuit should look similar to the one shown below, allowing him to continue his cutting edge research with the ternary number system despite his original lack of funds.



Student Last Name
Math 2603
Advanced Exploration 1: Boolean Algebras
Month and Day, 2020

1.

X	Y	C	S
0	0	0	0
0	1	0	1
0	2	0	2
1	0	0	1
1	1	0	2
1	2	1	0
2	0	0	2
2	1	1	0
2	2	1	1

Begin by placing 0 in the first row of column Y, 1 in the second row of column Y, and 2 in the third row of column Y. Continue this pattern for the remainder of rows in column Y. Then place 0 in the first three rows of column X, 1 in rows four through six in column X, and 2 in rows seven through nine. This will provide you with all the possible combinations of ternary digits. Then you must calculate the ternary sum of X and Y for each row. Take row eight for example. The ternary sum of 2+1=10. Therefore, place a 1 in row eight of column C to represent the carry, and 0 in row eight of column S to represent the sum. Use this same process to find all the values of C and S. Thus, the table above is an input/output table for a ternary adder.

2.

Now we must build the ternary adder using binary encoded signals so that two bits are used for each ternary digit. Let ternary digit X be encoded in binary digits x_1x_0 so $00= (0)_3$, $01= (1)_3$, and $10= (2)_3$. Use the same method to encode Y as bits y_1y_0 and S as s_1s_0. Because the carry digit takes only values 0 and 1, we can encode C using a single bit, and again call it C. Using the X and Y from part 1, we get the table below.

x_1	x_0	y_1	y_0
0	0	0	0
0	0	0	1
0	0	1	0
0	1	0	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	0	1
1	0	1	0

To find the outputs, we can first determine the ternary output, and convert it back to binary. The process is illustrated in the figure below.

x_1	x_0	y_1	y_0
0	0	0	0
0	0	0	1
0	0	1	0
0	1	0	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	0	1
1	0	1	0

Ternary Output
0+0=00
0+1=01
0+2=02
1+0=01
1+1=02
1+2=10
2+0=02
2+1=10
2+2=11

Binary Output
000
001
010
001
010
100
010
100
101

We can then compile the input/output table using binary.

x_1	x_0	y_1	y_0	C	S_1	S_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
1	0	0	0	0	1	0
1	0	0	1	1	0	0
1	0	1	0	1	0	1

Using this input/output table, we can design a binary circuit that implements the ternary adder. We will do so by first using the information in the table above to create a Boolean expression that defines the function. Note, a function with three outputs is equal to three functions with one output.

Let us begin by representing S_0 as a Boolean function. We must first identify each row where S_0 is equal to 1. S_0 is equal to 1 for row two, row four, and row nine. We can now take the sum of the minterms of these rows. Thus, we have:

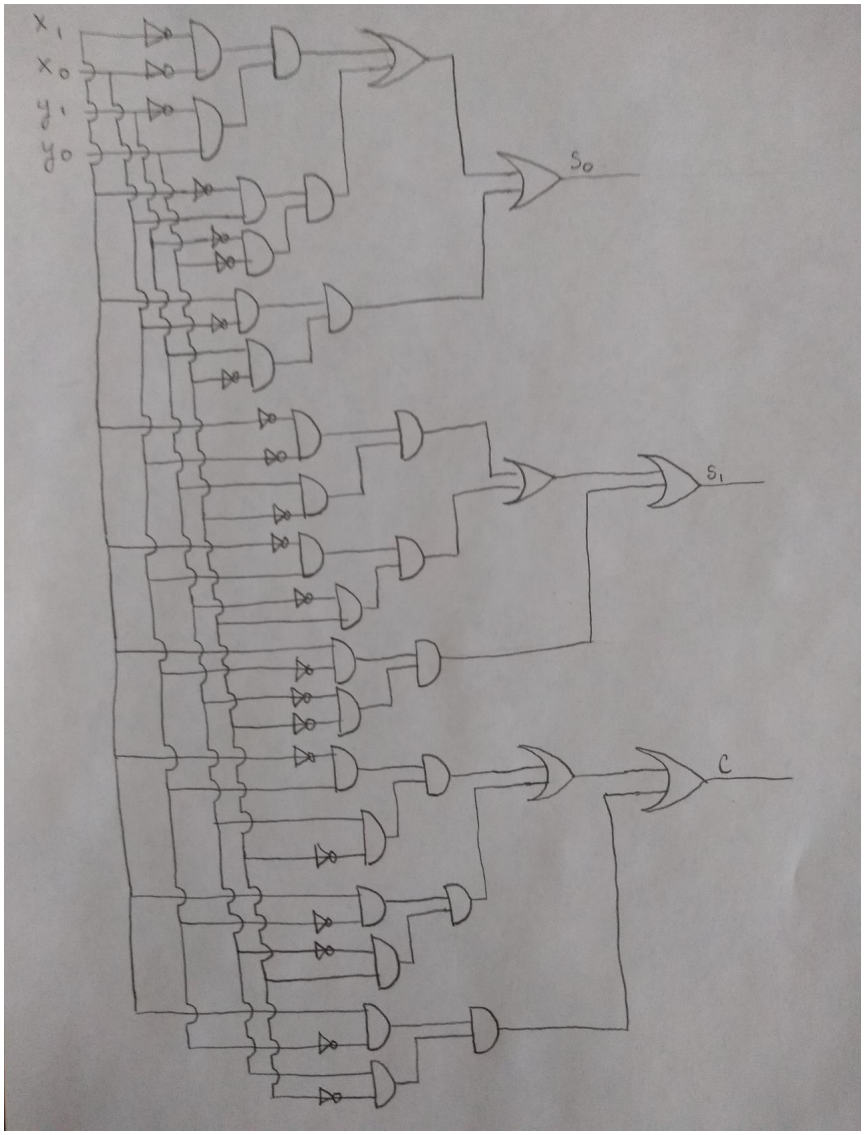
$S_0 = \bar{x}_1\bar{x}_0\bar{y}_1y_0 + \bar{x}_1x_0\bar{y}_1\bar{y}_0 + x_1\bar{x}_0y_1\bar{y}_0$

Apply the same method for S_1 and C we get:

$S_1 = \bar{x}_1\bar{x}_0y_1\bar{y}_0 + \bar{x}_1x_0\bar{y}_1y_0 + x_1\bar{x}_0\bar{y}_1\bar{y}_0$

$C = \bar{x}_1x_0y_1\bar{y}_0 + x_1\bar{x}_0\bar{y}_1y_0 + x_1\bar{x}_0y_1\bar{y}_0$

We now have the function we need to construct the binary circuit below. The gates in the circuit receive the Boolean input values x_1, x_0, y_1, y_0 and produce the outputs S_0, S_1, and C. In other words, the gates implement the above Boolean function. Hence, we have designed a binary circuit that implements the ternary adder that Casper needs to continue his research.



Advanced Exploration 1: Boolean Algebras

Another Student with a Name

1. Input / Output Table for a Ternary Adder:

Table 1:

X	Y	C	S
0	0	0	0
0	1	0	1
0	2	0	2
1	0	0	1
1	1	0	2
1	2	1	0
2	0	0	2
2	1	1	0
2	2	1	1

2.

The ternary number system comprises of three possible values for the digits, 0, 1, and 2. A ternary adder is an digital circuit that takes two bits, in this case X and Y, as inputs and outputs the sum of the two bits into two new bits named S for sum and C for carry. From this, an input/ output table can be generated, like the one shown in Table 1, that lists all the possible combinations of the three values for the bits and the values for S and C that are outputted.

To input this information into a binary circuit, the information from the ternary adder must be transformed into binary encoded signals where two bits will be used for each ternary digit. A new input/ output table can be constructed with these new values where X is encoded as the binary digits x_1x_0 so that 00 = $(0)_3$, 01 = $(1)_3$, and 10 = $(2)_3$. The same is done for Y and S where Y is encoded as the bits y_1y_0, and S is encoded as the bits s_1s_0. The carry digit, C, only takes 0 and 1 for its values so it only needs to be encoded as a single bit C. The input/ output table that represents the ternary adder converted into the binary number system is shown below in Table 2.

Table 2:

X		Y			S	
x_1	x_0	y_1	y_0	C	s_1	s_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
1	0	0	0	0	1	0
1	0	0	1	1	0	0
1	0	1	0	1	0	1

After converting the values from the ternary adder into the binary number system, Boolean expressions need to be determined to design a corresponding binary circuit. To do this, Karnaugh maps, or K-maps can be utilized. Using the values from the Table 2, a K-map can be used for each of the outputs to write Boolean expressions that are equivalent to the those represented in the table. The calculated Boolean expressions and their K-maps are shown below.

For C:

x_1x_0 \ y_1y_0	00	01	10
00	0	0	0
01	0	0	1
10	0	1	1

$$F = x_1\overline{x_0}\overline{y_1}y_0 + y_1\overline{y_0}$$

For s_1:

x_1x_0 \ y_1y_0	00	01	10
00	0	0	1
01	0	1	0
10	1	0	0

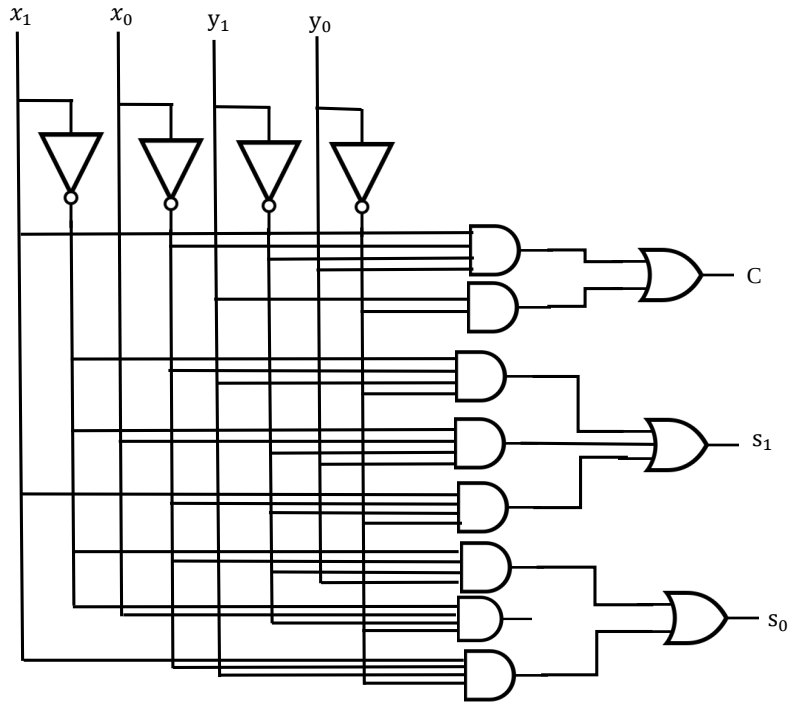
$$F = \overline{x_1}\overline{x_0}y_1\overline{y_0} + \overline{x_1}x_0\overline{y_1}y_0 + x_1\overline{x_0}\overline{y_1}\overline{y_0}$$

For s_0:

x_1x_0 \ y_1y_0	00	01	10
00	0	1	0
01	1	0	0
10	0	0	1

$$F = \overline{x_1}\overline{x_0}\overline{y_1}y_0 + \overline{x_1}x_0\overline{y_1}\overline{y_0} + x_1\overline{x_0}y_1\overline{y_0}$$

From this information, a binary circuit can be designed using AND, OR, and inverter gates to execute the Boolean expressions and achieve the desired results. A binary circuit that implements the ternary adder is shown below.



Advanced Exploration 1: Boolean Algebras

1)

X	Y	C	S
0	0	0	0
0	1	0	1
0	2	0	2
1	0	0	1
1	1	0	2
1	2	1	0
2	0	0	2
2	1	1	0
2	2	1	1

2)

A		B		Carry	Sum	
a1	a0	b1	b0	c(out)	s1	s0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
1	0	0	0	0	1	0
1	0	0	1	1	0	0
1	0	1	0	1	0	1

C(out) = a0b1 + a0b1 + a1b0

s1 = a0b0 + $\overline{a1}\,\overline{a0}$b1 + a1$\overline{b1}\,\overline{b0}$

s2 = a1b1 + $\overline{a1}\,\overline{a0}$b0 + a0$\overline{b1}\,\overline{b0}$