ITM 207: Truth Table Conversions

EXPRESSION TO TRUTH TABLE, TRUTH TABLE TO EXPRESSION

Note: Please ensure you are comfortable with the boolean operators and the structure of truth tables prior to studying this tip sheet.You can review those topics in the Boolean Basics Tip Sheet.

Expression to Truth Table

Tip: Before constructing a truth table from an expression, it is always a good idea to simplify it unless the question specifies not to do so. It will shorten the expression and make calculating the outputs easier. To learn some tips on simplifying expressions, check out our Simplifying Expressions Tip Sheet.

Step 1: Create a table frame that includes all the input variables

Let’s create a truth table for the following expression:

$$X = (A' + C)(B + A)$$

First, we create a frame for all the input variables. Whole terms can also be included in the truth table, but we will look at that in the end and consider its benefits. For now, let’s only focus on the letter variables:

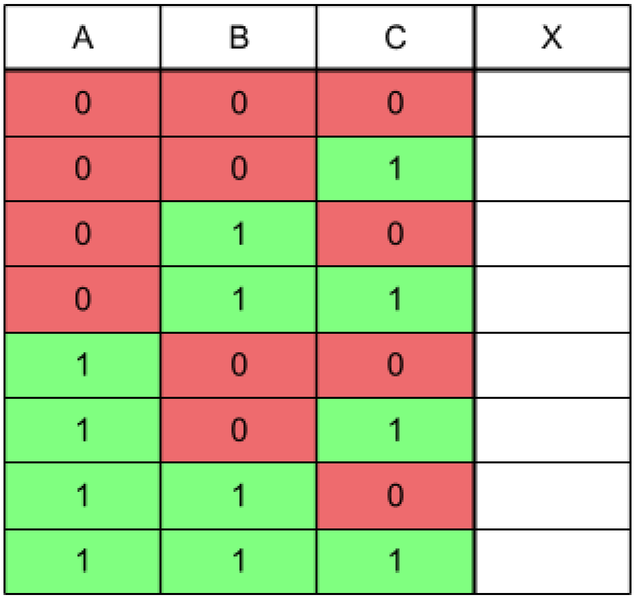
A	B	C	X

A, B, and C are the input variables, and X is the output variable.

Step 2: Fill out all possible combinations of inputs

Since we are working with boolean data, a variable can only be either 0 or 1. The tricky part lies in how we should fill out the table to ensure all possible input scenarios are accounted for. The way to do this is to consider each row of inputs as a binary number. Each number is unique and all we have to do is to keep counting up until we hit the highest possible number with the given amount of digits.

In this example, there are 3 variables and thus 3 digits. The smallest number is 000 and the largest is 111, leaving us with a total of 8 unique combinations:



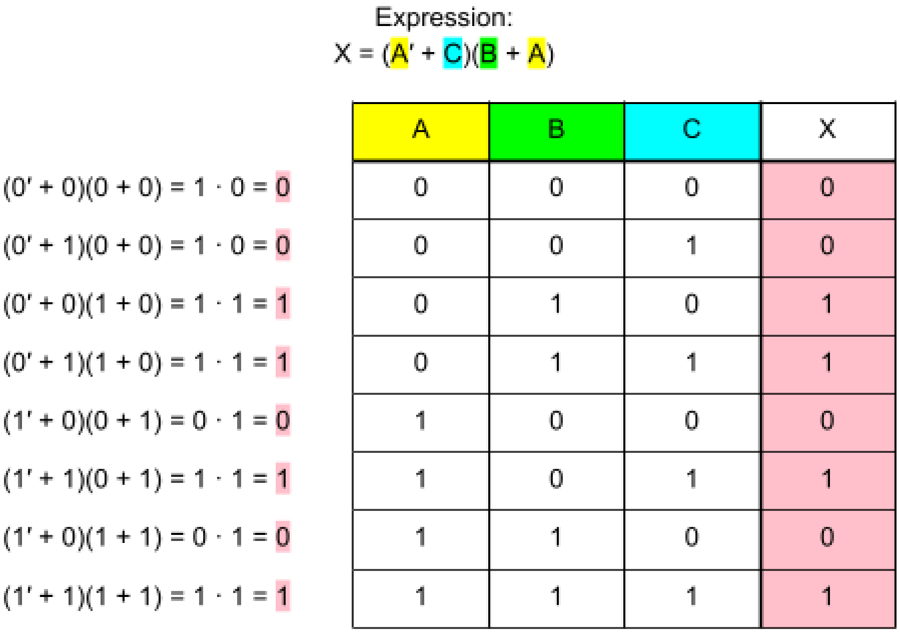
A	B	C	X
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

There are some patterns here that can help us fill out truth tables quicker:

- The number of unique combinations can be found by calculating 2^n, where n is the number of different input variables. In this example, it would be 2^3, giving us 8 combinations.
  - You may have recognized this formula as the same formula used to find the number of different things a binary number can represent given a limited number of digits. This is exactly what is going on. The number of variables are the number of digits we can work with and the things being represented are the different possible combinations.
- If we look at the input columns from right to left, we can see patterns in the way that 0 and 1 alternate. The numbers flip every row under C, every two rows under B, and every four rows under A. You can use these patterns to fill out the table column by column until the bottom row has all 1s.
  - But what if there is a fourth variable? How many rows will it take for the number to flip? Well, given the pattern of 1, 2, 4… the next number would be 8. These again relate back to the powers of 2. If you think back to the positional notation of binary numbers, the first digit has the value of 2^0, the second has 2^1, and the third 2^2. C is the first digit, B is the second, and A is the third. The value they each can represent in a binary number is 1, 2, and 4, which are also the numbers for how many rows until each flip.

Step 3: Substitute the expression with the combinations and solve for outputs

Now, we just go row by row replacing the variables in the expression with the inputs and solve:

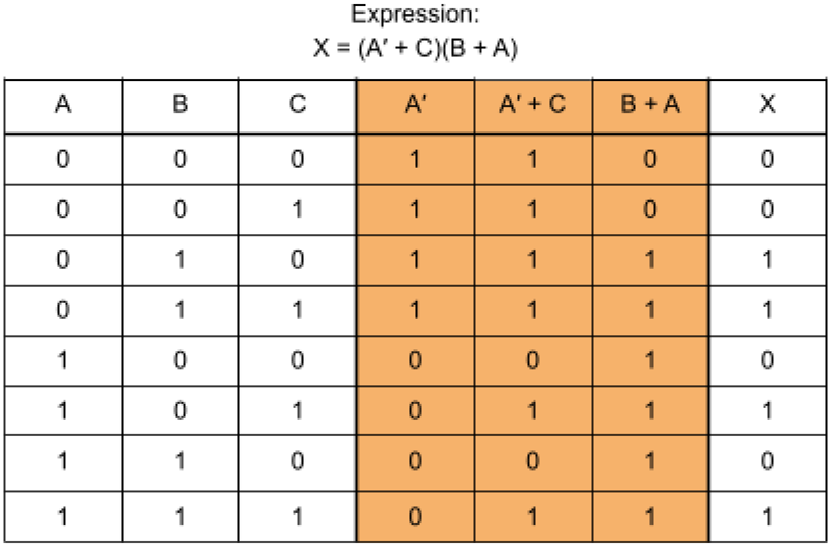


Expression:

$X = (A' + C)(B + A)$

	A	B	C	X
$(0' + 0)(0 + 0) = 1 \cdot 0 = 0$	0	0	0	0
$(0' + 1)(0 + 0) = 1 \cdot 0 = 0$	0	0	1	0
$(0' + 0)(1 + 0) = 1 \cdot 1 = 1$	0	1	0	1
$(0' + 1)(1 + 0) = 1 \cdot 1 = 1$	0	1	1	1
$(1' + 0)(0 + 1) = 0 \cdot 1 = 0$	1	0	0	0
$(1' + 1)(0 + 1) = 1 \cdot 1 = 1$	1	0	1	1
$(1' + 0)(1 + 1) = 0 \cdot 1 = 0$	1	1	0	0
$(1' + 1)(1 + 1) = 1 \cdot 1 = 1$	1	1	1	1

As mentioned earlier, you can also include terms in the truth table. This can reduce the amount of equations you have to write down and help keep track of longer expressions. However, the amount of calculations you have to do for the output stays the same:



Expression:

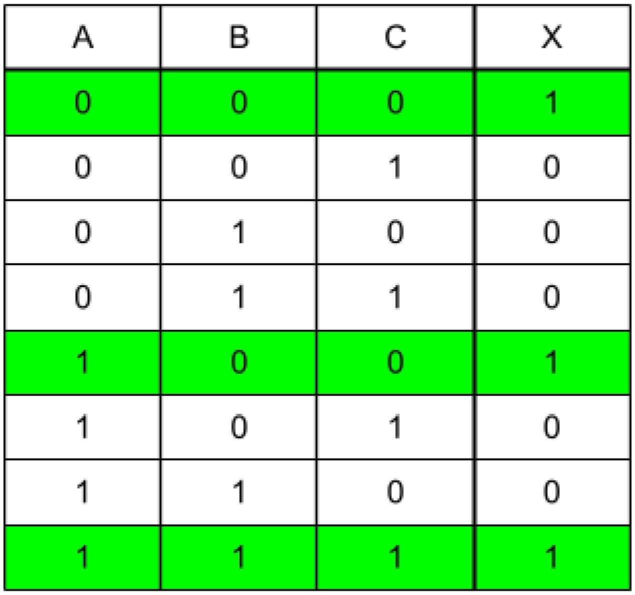
$X = (A' + C)(B + A)$

A	B	C	A′	A′ + C	B + A	X
0	0	0	1	1	0	0
0	0	1	1	1	0	0
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	0	0	1	0
1	0	1	0	1	1	1
1	1	0	0	0	1	0
1	1	1	0	1	1	1

Truth Table to Expression

Step 1: Identify rows where the output is 1

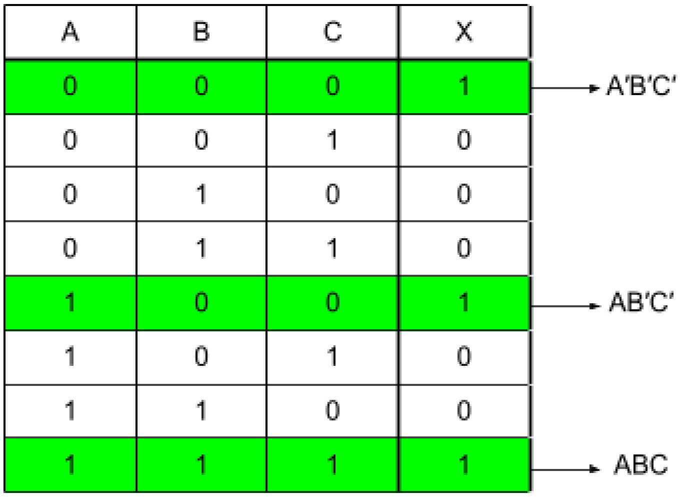
In this step, we are essentially asking the question: which combination of inputs yield an output of True?



A	B	C	X
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Step 2: Write an expression for each row by multiplying the inputs:

Here, we ask the question: what are the components of each selected combination? For example, the middle row means: for the output to be True, A has to be True, B has to be False, and C also has to be False, hence the expression of A **AND NOT** B **AND NOT** C (AB′C′):



A	B	C	X	
0	0	0	1	→ A′B′C′
0	0	1	0	
0	1	0	0	
0	1	1	0	
1	0	0	1	→ AB′C′
1	0	1	0	
1	1	0	0	
1	1	1	1	→ ABC

Step 3: Sum all the expressions

Now we connect all the expressions with addition:

$$X = A'B'C' + AB'C' + ABC$$

The addition represents **OR** gates. The combined expression is essentially saying: For the output to be 1, we need either A′B′C′, AB′C′, or ABC. This is true because as we found out in step 1 and 2, either one of the smaller expressions can yield an output of 1.

This would be the final step in finding the expression of a truth table, but depending on what the question asks or what you are going to do with the expression, you may want to simplify it. You can take a look at our Simplifying Expressions Tip Sheet for help on that topic.

Note: For conversions between truth tables and circuit diagrams, it is much easier to first convert to an expression then to the desired format. Refer to our Circuit Diagram Conversions Tip Sheet to review how to perform conversions between circuit diagrams and expressions!