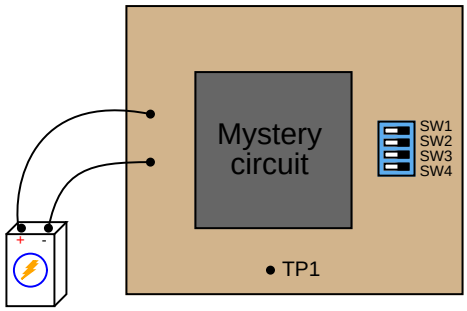

More Boolean Algebra

Questions

Question 1

Suppose you were faced with the task of writing a Boolean expression for a logic circuit, the internals of which are unknown to you. The circuit has four inputs – each one set by the position of its own micro-switch – and one output. By experimenting with all the possible input switch combinations, and using a logic probe to "read" the output state (at test point TP1), you were able to write the following truth table describing the circuit's behavior:



SW1	SW2	SW3	SW4	TP1
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

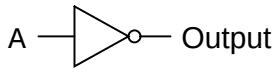
Based on this truth table "description" of the circuit, write an appropriate Boolean expression for this circuit.

file 01304

Question 2

Although it is seldom done, it is possible to express a truth table in verbal form, by describing what conditions must be met in order to generate a "high" output.

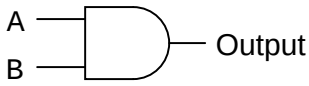
Take for example this simple truth table, for an inverter circuit:



A	Output
0	1
1	0

For this truth table, we could say that the output goes high when A is low. A different way of saying this would be to state that "the output is *true* when $\overline{A}$ is *true*."

Let's look at another example, this time of an AND gate:

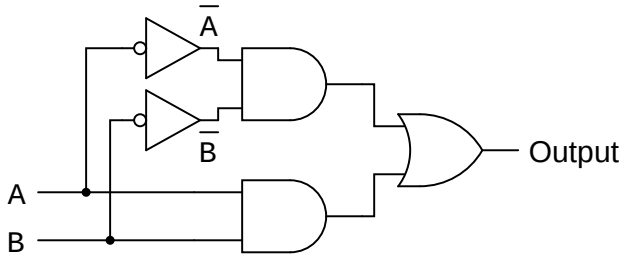


A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

For this truth table, we could say that the output goes high when A and B are both high. A different way of saying this would be to state that "the output is true when A is true and B is true." To use a half-Boolean, half-verbal description:

A AND B

Examine this logic gate circuit and corresponding truth table:



A	B	Output
0	0	1
0	1	0
1	0	0
1	1	1

Express the functionality of this truth table in words. What Boolean conditions must be satisfied ("true") in order for the output to assume a high state?

file 01326

Question 3

Develop a verbal description of this truth table, specifying what conditions must be met ("*true*" in a Boolean sense) in order for the output to assume a high state:

A	B	C	Output
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Do the same for this truth table as well:

A	B	C	Output
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

file 01327

Question 4

Sum-of-Product Boolean expressions all follow the same general form. As such, their equivalent logic gate circuits likewise follow a common form. Translate each of these SOP expressions into its equivalent logic gate circuit:

$$AB + A\overline{B}$$

$$A\overline{B} + \overline{A}B$$

$$ABC + \overline{A}B\overline{C} + AB\overline{C}$$

file 01325

Question 5

Write a Boolean SOP expression for this truth table, then simplify that expression as much as possible, and draw a logic gate circuit equivalent to that simplified expression:

A	B	C	Output
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

file 02822

Question 6

Write an SOP expression for this truth table, and then draw a gate circuit diagram corresponding to that SOP expression:

A	B	C	Output
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Finally, simplify this expression using Boolean algebra, and draw a simplified gate circuit based on this new (reduced) Boolean expression.

file 01333

Question 7

Examine the following truth table:

A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0

We know that this table represents the function of a NAND gate. But suppose we wished to generate a Boolean expression for this gate as though we didn't know what it already was, and we chose to generate an SOP expression based on all the "high" output conditions in the truth table:

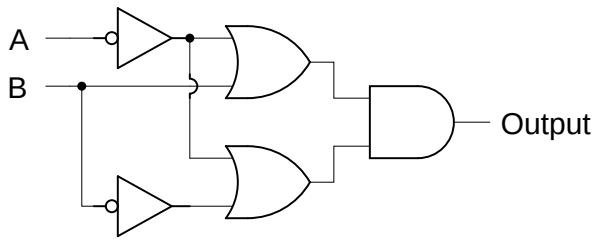
$$\overline{A}\,\overline{B} + \overline{A}B + A\overline{B}$$

Seems like a lot of work for just one gate, doesn't it? The fact that this truth table's output is mostly 1's causes us to have to write a relatively lengthy SOP expression. Wouldn't it be easier if we had a technique to generate a Boolean expression from the single *zero* output condition in this table? If we had such a technique, our resulting Boolean expression would have a lot fewer terms in it!

We know that a Negative-OR gate has the exact same functionality as a NAND gate. We also know that a Negative-OR gate's Boolean representation is $\overline{A} + \overline{B}$. If there is such a thing as a technique for deriving Boolean expressions from the "0" outputs of a truth table, this instance ought to fit it!

Now, examine the following truth table and logic gate circuit:

A	B	Output
0	0	1
0	1	1
1	0	0
1	1	0



Derive a Boolean expression from the gate circuit shown here, and then compare that expression with the truth table shown for this circuit. Do you see a pattern that would suggest a rule for deriving a Boolean expression directly from the truth table in this example (and the previous example)?

Hint: the rule involves *Product-of-Sums* form.

file 01338

Question 8

Product-of-Sum Boolean expressions all follow the same general form. As such, their equivalent logic gate circuits likewise follow a common form. Translate each of these POS expressions into its equivalent logic gate circuit:

$$(A + B)(\overline{A} + \overline{B})$$

$$(\overline{A} + \overline{B})(\overline{A} + B)$$

$$(A + B + C)(\overline{A} + B + \overline{C})(A + B + \overline{C})$$

file 02825

Question 9

Product-of-Sum Boolean expressions all follow the same general form. As such, their equivalent logic gate circuits likewise follow a common form. Translate each of these POS expressions into its equivalent logic gate circuit:

$$(A + B)(A + \overline{B})$$

$$(A + \overline{B})(\overline{A} + B)$$

$$(A + B + C)(\overline{A} + B + \overline{C})(A + B + \overline{C})$$

file 01336

Question 10

Inspect each of these Boolean expressions, and determine whether each one is a *sum of products*, or a *product of sums*:

$$(B + \overline{C} + D)(\overline{A} + B)$$

$$A\overline{B}\,\overline{C} + \overline{A}BC$$

$$(X + \overline{Y} + \overline{Z})(\overline{Y} + Z)(\overline{X} + Y)$$

$$\overline{M}\,\overline{N}\,\overline{O} + MN\overline{O} + M\overline{N}O$$

$$(X + \overline{Y + Z})(\overline{Y + \overline{Z}})$$

$$\overline{ABC} + A\overline{B}C$$

file 01324

Question 11

Examine this truth table and then write both SOP and POS Boolean expressions describing the Output:

A	B	C	Output
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Which of those Boolean expressions is simpler for this particular truth table? Which will be easier to reduce to simplest form (for the purpose of creating a gate circuit to implement it)?

file 02823

Question 12

Write a Boolean expression for this truth table, then simplify that expression as much as possible, and draw a logic gate circuit equivalent to that simplified expression:

A	B	C	Output
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

file 01341

Question 13

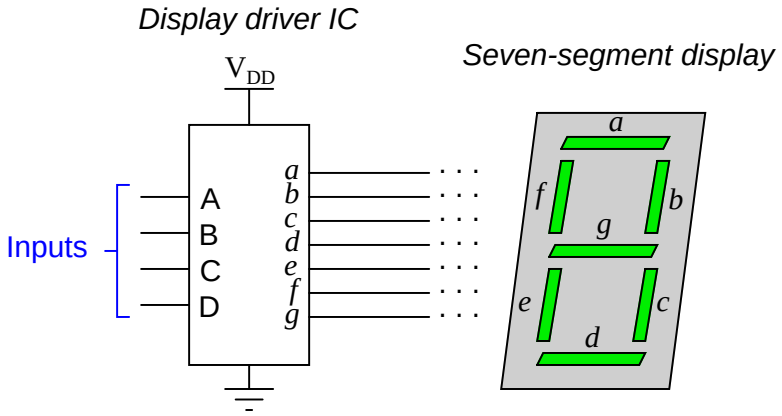
Write a Boolean expression for this truth table, then simplify that expression as much as possible, and draw a logic gate circuit equivalent to that simplified expression:

A	B	C	Output
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

file 02826

Question 14

A *seven segment decoder* is a digital circuit designed to drive a very common type of digital display device: a set of LED (or LCD) segments that render numerals 0 through 9 at the command of a four-bit code:



The behavior of the display driver IC may be represented by a truth table with seven outputs: one for each segment of the seven-segment display (a through g). In the following table, a "1" output represents an active display segment, while a "0" output represents an inactive segment:

D	C	B	A	a	b	c	d	e	f	g	Display
0	0	0	0	1	1	1	1	1	1	0	"0"
0	0	0	1	0	1	1	0	0	0	0	"1"
0	0	1	0	1	1	0	1	1	0	1	"2"
0	0	1	1	1	1	1	1	0	0	1	"3"
0	1	0	0	0	1	1	0	0	1	1	"4"
0	1	0	1	1	0	1	1	0	1	1	"5"
0	1	1	0	1	0	1	1	1	1	1	"6"
0	1	1	1	1	1	1	0	0	0	0	"7"
1	0	0	0	1	1	1	1	1	1	1	"8"
1	0	0	1	1	1	1	1	0	1	1	"9"

Write the unsimplified SOP or POS expressions (choose the most appropriate form) for outputs a, b, c, and e.

file 02824