CIS1910 Discrete Structures in Computing (I)

Winter 2019, Lab 4 Notes

Here are recommended practice exercises. Many have been covered in Lab 4.
Note that the examples and exercises listed in blue come from the following textbook:
"Discrete Mathematics and Its Applications," by Rosen, Mc Graw Hill, 7th Edition

A. BOOLEAN ALGEBRA AND CIRCUIT DESIGN

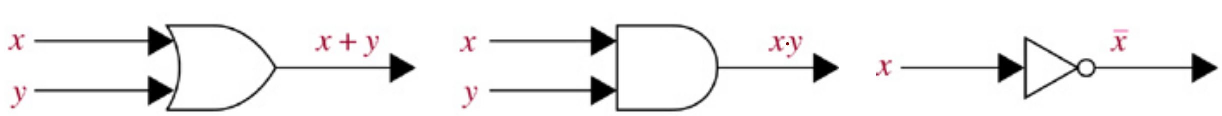
Consider the Boolean algebra $(\{0,1\},+,\cdot,{}^{-})$, as seen in class (slide 2.11). The Boolean operations $+$, $\cdot$ and $^{-}$ can then be defined by the tables below. In this lab, the symbol + will read "or" instead of "plus", the symbol · will read "and" instead of "dot", and the symbol ¯ will read "not" instead of "bar". Moreover, we will give · a higher precedence than +.

x	y	x+y
0	0	0
0	1	1
1	0	1
1	1	1

x	y	x·y
0	0	0
0	1	0
1	0	0
1	1	1

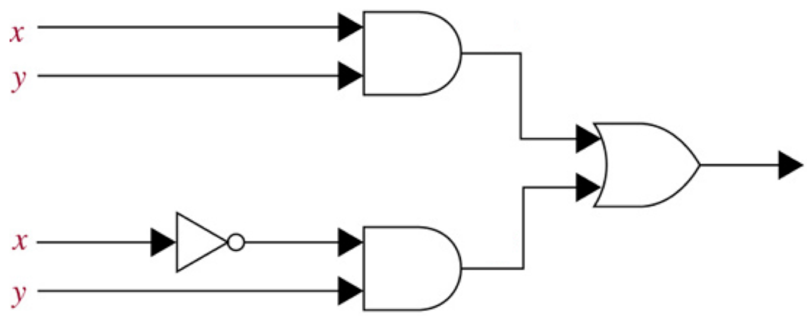
x	$\overline{x}$
0	1
1	0

This Boolean algebra is at the basis of ***circuit design***. A computer is made up of a number of circuits. The basic elements of circuits are ***gates***. Typically, there are one or more inputs to a gate, and only one output. Gate inputs are driven by voltages having two nominal values (e.g., 0V and 5V); these values are represented by the symbols 0 and 1 respectively. The output of a gate also provides two nominal values of voltage only. Here are common gates:

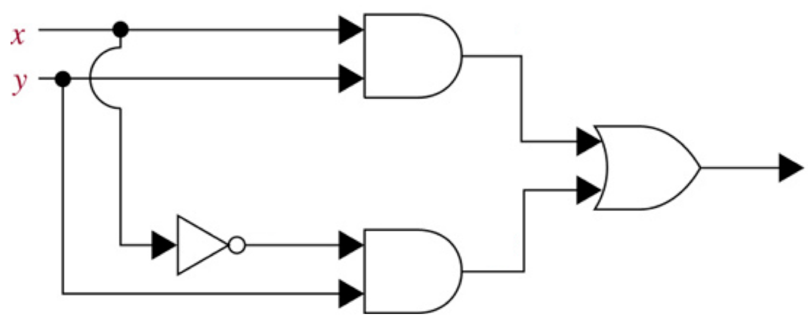


OR gate　　**AND** gate　　**INVERTER**

1. Consider the circuit below:



It can also be represented as follows:



What is the output to this circuit?

2. Exercises 1, 3 and 5 from Section 12.3 of the textbook

3. ***(a)*** Example 1 p. 823
(b) Draw the tables that correspond to these circuits.

4. ***(a)*** Construct the circuit that produces the output x·y+x·z+y·z.
(b) Draw the table that corresponds to this circuit.
(c) Example 2 p. 825

5. Example 3 p. 825

6. Consider the Boolean functions F and G defined by the table below.

x	*y*	*z*	*F(x,y,z)*	*G(x,y,z)*
0	0	0	0	1
0	0	1	0	1
0	1	0	0	0
0	1	1	0	1
1	0	0	0	1
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

(a) Show that F(x,y,z) can be expressed as a sum of minterms of degree 3. This sum is called the ***sum-of-products expansion*** of F.

(b) Construct the circuit that produces the output F(x,y,z).

(c) Note that the sum-of-products expansion of F involves 6 operations: 1 sum, 4 products, 1 complementation. ***Minimize*** F, i.e., find an expression for F(x,y,z) that involves a minimum number of operations.

(d) Construct the circuit that produces the output $x\cdot z$.

(e) Show that G(x,y,z) can be expressed as a product of maxterms of degree 3 (this product is called the ***product-of-sums expansion*** of G), and minimize G.

7. ***(a)*** For any positive integer n and for any Boolean function F of degree n, it is possible to find an expression for $F(x_1,x_2,\ldots,x_n)$ that involves no other Boolean operations than those in the set $\{+,\cdot,\bar{\ }\}$. Explain. We say that $\{+,\cdot,\bar{\ }\}$ is ***functionally complete***.

(b) Let ↓ be the Boolean operation defined by $x\downarrow y=\overline{x+y}$, for any elements x and y of B. This operation is called the ***NOR*** operation. Show that for any x and y of B we have:
$\overline{x}=x\downarrow x$ and $x\cdot y=(x\downarrow x)\downarrow(y\downarrow y)$ and $x+y=(x\downarrow y)\downarrow(x\downarrow y)$.

(c) Is $\{\downarrow\}$ functionally complete?

(d) The Boolean expression $x\cdot\overline{y}+z$ involves three distinct Boolean operations: +, · and ‾. Find an equivalent expression that involves ↓ only.

B. PROPOSITIONAL LOGIC

A ***propositional expression*** is a finite sequence of symbols. The accepted symbols are T (which denotes a proposition that is true), F (which denotes a proposition that is false), p, q, r, etc. (which denote propositional variables), $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, etc. (which denote propositional operations) and brackets. The sequence should make sense, i.e., it should become a proposition once specific propositions are considered. For example, T, p, $\neg q$, $p\vee F$, $q\wedge\neg p$, $p\rightarrow[(\neg q)\vee r]$ are propositional expressions, while $\vee pq$, $p\wedge\vee q\neg$ and $)p\vee(q]$ are not. Note that a truth table can be attached to any propositional expression.

1. Section 1.1 of the textbook

(a) Examples 1, 2, 3, 4, 5, 6

(b) Because conditional statements play such an essential role in mathematical reasoning, a variety of terminology is used to express $p\rightarrow q$. For example:

"if p, then q"	"q if p"	"p is sufficient for q"
"if p, q"	"q when p"	"a sufficient condition for q is p"
"p implies q"	"q unless not p"	"q is necessary for p"
"p only if q"	"q follows from p"	"a necessary condition for p is q"

Examples 7 and 10

2. Section 1.2

(a) Examples 1 and 2

(b) Translating sentences in natural language (such as English) into propositional expressions is an essential part of hardware and software ***system specification***. System and software engineers take requirements in natural language and produce precise and unambiguous specifications that can be used as the basis for system development.

Example 3