

Section 7.2: Logic Networks

April 13, 2009

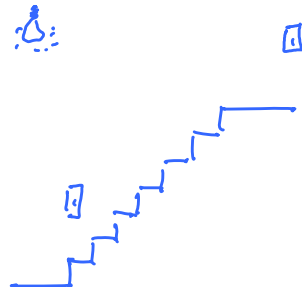
Abstract

We examine the relationship between the abstract structure of a Boolean algebra and the practical problem of creating logic networks for solving problems. There is a fundamental equivalence between Truth Functions, Boolean Expressions, and Logic Networks which allows us to pass from one to the other.

1 An Example Application, and Fundamental Parallels

Example: Two light switches, one light!

The problem is as follows: A light at the bottom of some stairs is controlled by two light switches, one at each end of the stairs. The two switches should be able to control the light independently. How do we wire the light?



- A Truth Function

$f(s_1, s_2) = L$

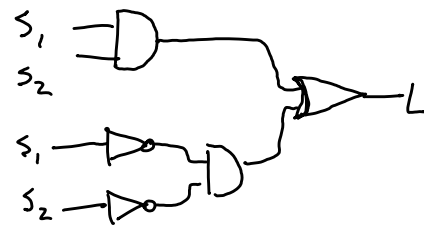
s_1	s_2	L
1	1	1
1	0	0
0	1	0
0	0	1

s_1	s_2	L
on	on	on
on	off	off
off	on	off
off	off	on

- A Boolean Expression

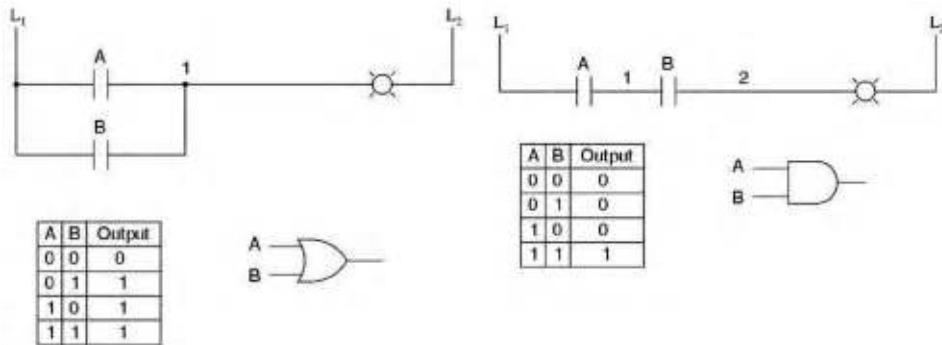
$$s_1 \cdot s_2 + s_1' \cdot s_2' = f(s_1, s_2)$$
$$s_1 \cdot s_2 + (s_1 + s_2)' = f(s_1, s_2)$$

• A Logic Network (Basic Mechanics and Conventions)

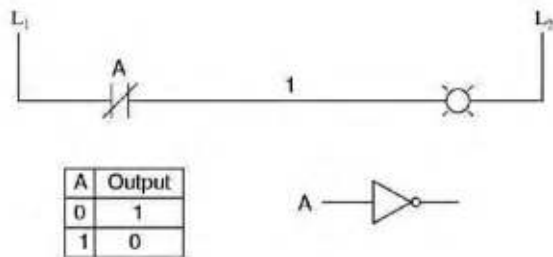


– Input or output lines are not tied together except by passing through gates:

- * OR gate
- * AND gate



* NOT gate



- Lines can be split to serve as input to more than one device.
- There are not loops with output of a gate serving as input to the same gate (feedback).
- There are no delay elements.

2 Applications

2.1 Converting Truth Tables to Boolean Expressions (Canonical Sum-of-Products Form)

Example: Practice 11, p. 558

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
1	1	1	1
1	1	0	0
1	0	1	1
1	0	0	1
0	1	1	0
0	1	0	0
0	0	1	1
0	0	0	1

$$f(x_1, x_2, x_3) =$$

$$x_1 \cdot x_2 \cdot x_3 +$$

$$x_1 \cdot x_2' \cdot x_3 +$$

$$x_1 \cdot x_2' \cdot x_3' +$$

$$x_1' \cdot x_2' \cdot x_3 +$$

$$x_1' \cdot x_2' \cdot x_3'$$

Example: Exercise 11, p. 568

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
1	1	1	0
1	1	0	0
1	0	1	1
1	0	0	1
0	1	1	0
0	1	0	1
0	0	1	0
0	0	0	0

$$= x_1 x_2' x_3 +$$

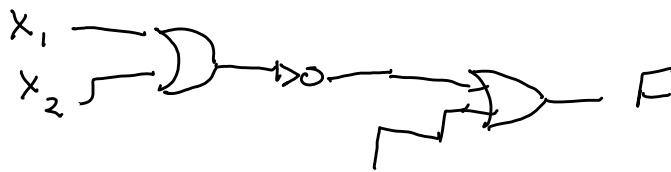
$$x_1 x_2' x_3' +$$

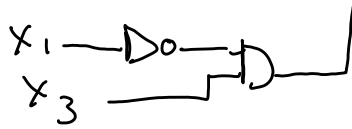
$$x_1' x_2 x_3'$$

2.2 Converting Boolean Expressions to Logic Networks

Example: Exercise 1b, p. 566

$$(x_1 + x_2)' + x_1' x_3$$





2.3 Converting Logic Networks to Truth Functions or Boolean Expressions

Example: Exercise 2, p. 567

X_1	X_2	$f(X_1, X_2)$
1	1	1
1	0	1
0	1	0
0	0	1

$$= X_1 X_2 + X_2'$$

2.4 Simplifying Canonical Form

We can use properties of Boolean algebra to simplify the canonical form, creating a much simpler logic network as a result.

Example: Practice 11, p. 558

$$f(x_1, x_2, x_3) = x_1 x_2 x_3 + x_2' = x_2' + x_1 x_2 x_3$$

$$= (x_2' + x_1) \cdot (x_2' + x_2) \cdot (x_2' + x_3)$$

$$= (x_2' + x_1) (x_2' + x_3) = x_2' + x_1 x_3$$

$$= x_2' \left[x_1 x_3 + x_1 x_3' + x_1' x_3 + x_1' x_3' \right]$$

$$= x_2'$$

2.5 Adding Binary numbers

Half-Adders and Full-Adders

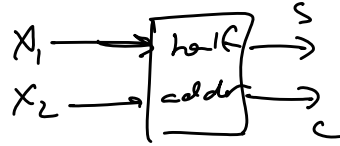
$$\begin{array}{r} 1 \\ + 1 \\ \hline 10 \end{array}$$

X_1	X_2	C	S	$X_1 + X_2$	$(X_1 X_2)'$	Δ
1	1	1	0	1	0	0
1	0	0	1	1	1	1
0	1	0	1	1	1	1
0	0	0	0	0	1	0

Half-Adder: Adds two binary digits.

$$s = x_1'x_2 + x_1x_2'$$

$$c = x_1x_2$$



Note, however, that the half-adder doesn't implement s in this way:
instead,

$$s = (x_1 + x_2) \cdot (x_1x_2)'$$

Questions:

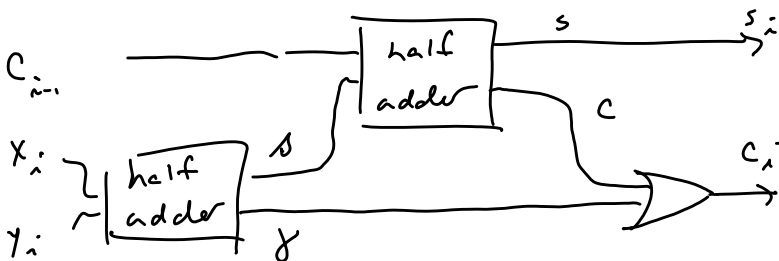
- 1 How? *Checked that it works via the truth function.*
- 2 Why? *efficient!*

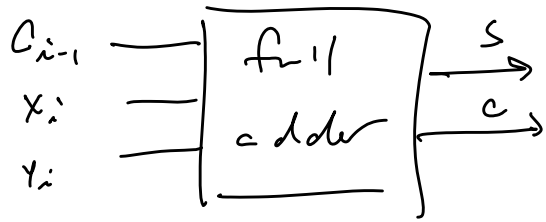
Full-Adder: Adds two digits plus the carry digit (made up of two half-adders, essentially!).

- Give c_{i-1}, x_i, y_i
- Simply use a half-adder to add the carry digit c_{i-1} to the sum digit s of a half-adder of x_i, y_i , to get s_i .
- To get the carry digit c_i , compare carry digits of both half-adders, to see if either gives a 1 (in which case $c_i = 1$).

Example: Practice 12, p. 563

C_{i-1}	x_i	y_i	s_i	c_i	half-adder in $x_i + y_i$		half-adder on $s + c_{i-1}$		
					s	c	s	c	$c + y$
0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	1	0	0
0	1	0	1	0	1	0	1	0	0
0	1	1	0	1	0	1	0	0	1
1	0	0	1	0	0	0	1	0	0
1	0	1	0	1	1	0	0	1	1
1	1	0	0	1	1	0	0	1	1
1	1	1	1	1	0	1	1	0	1





$$\begin{array}{r}
 1 \\
 0 1 \\
 \hline
 1 0 0
 \end{array}$$

