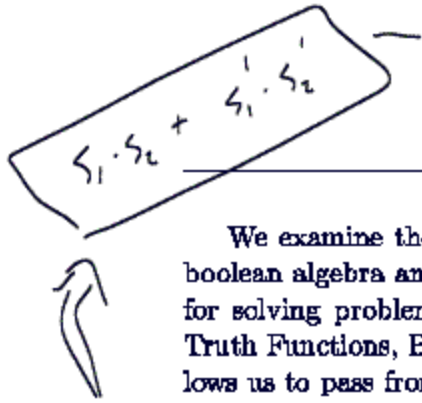


Section 7.2: Logic Networks

March 30, 2005



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.

| S_1 | S_2 | L |
|-------|-------|-----|
| On | On | On |
| On | Off | Off |
| Off | On | Off |
| Off | Off | On |

$$(S_1 \wedge S_2) \vee (S_1' \wedge S_2')$$

Truth table or function

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 1 |

1 An Example Application, and Fundamental Parallels:

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
- A Boolean Expression
- 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. *Improves readability*
- 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)

Practice 11, p. 485

Exercise 10, p. 495

2.2 Converting Boolean Expressions to Logic Networks

Exercise 1b, p. 493

2.3 Converting Logic Networks to Truth Functions or Boolean Expressions

Exercise 2, p. 493

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.

2.5 Adding Binary numbers

Half-Adders and Full-Adders

Half-Adder: Adds two binary digits.

CSF

$$\begin{cases} s = x_1'x_2 + x_1x_2' \\ c = x_1x_2 \end{cases}$$

| x_1 | x_2 | $s(x_1, x_2)$ | $c(x_1, x_2)$ | s_2 |
|-------|-------|---------------|---------------|-------|
| 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 |

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?
2. Why?

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

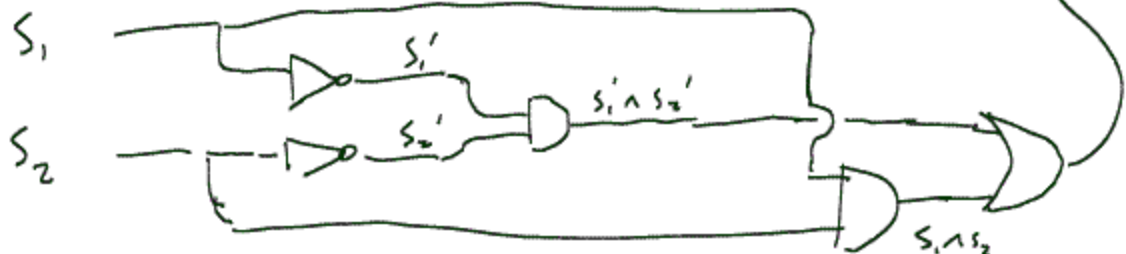
p 490

- 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 $c_i = 1$).

The light switch logic network:

$$(s_1 \wedge s_2) \vee (s_1' \wedge s_2')$$

3



Practice
11

| 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 |

"Canonical sum of products"

$$x_1 x_2 x_3 + x_1 x_2' x_3 + x_1 x_2' x_3' + x_1' x_2' x_3 + x_1' x_2' x_3'$$

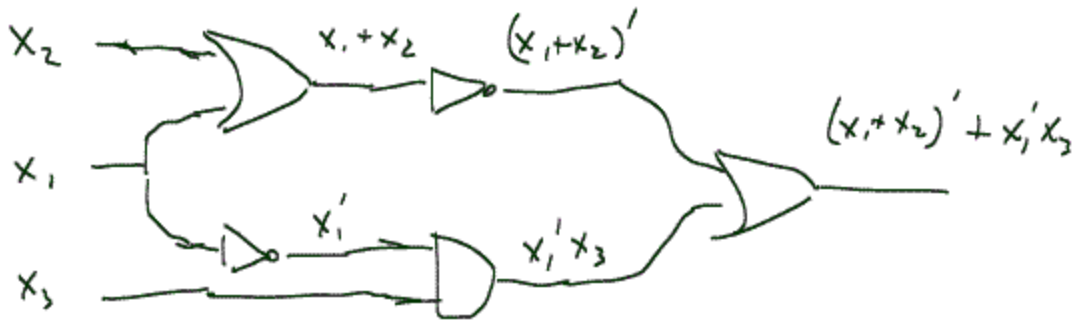
#10 p 495

| 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 |

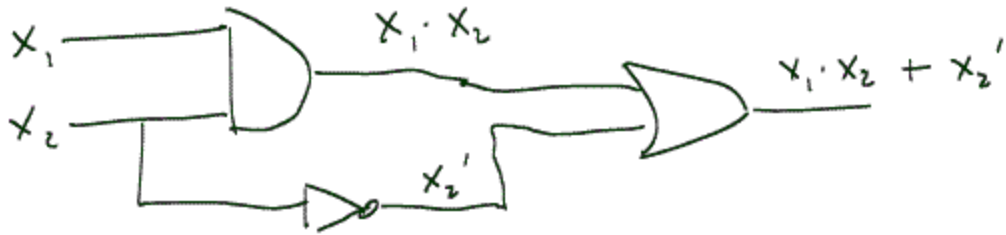
$$x_1 x_2 x_3 + x_1 x_2' x_3' + x_1' x_2 x_3'$$

#15, p 493

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



#2



| x_1 | x_2 | $f(x_1, x_2)$ |
|-------|-------|---------------|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

$$x_1 \cdot x_2 + x_2'$$

Let's simplify practice 11 i

$$x_1 x_2 x_3 + x_1 x_2' x_3 + x_1 x_2' x_3' + x_1' x_2 x_3 + x_1' x_2' x_3'$$

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

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

$$CSP = x_1 x_3 + x_1 x_2' x_3' + x_1' x_2'$$

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

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

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

Truth table for full adder:

| x_1 | x_2 | c | s | c_2 |
|-------|-------|-----|-----|-------|
| 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 |