SCHAUM'S OUTLINE OF

THEORY AND PROBLEMS OF BOOLEAN ALGEBRA and SWITCHING CIRCUITS

●

BY

ELLIOTT MENDELSON, Ph.D.

Professor of Mathematics
Queens College
City University of New York

●

McGRAW-HILL

New York St. Louis San Francisco Auckland Bogotá Caracas Lisbon London Madrid Mexico City Milan Montreal New Delhi San Juan Singapore Sydney Tokyo Toronto

4.4 LOGIC CIRCUITS

The processing of information is one of the most important roles of the modern digital computer. For this purpose, special devices are available.

An *and-gate* operates on two or more inputs $A_1, \ldots, A_n$ and produces their conjunction $A_1 \& A_2 \& \ldots \& A_n$. An and-gate is denoted $\textcircled{\&}$.

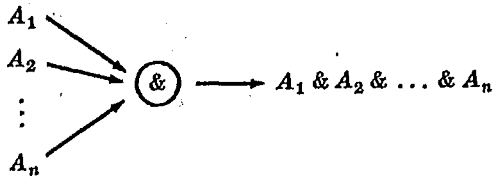


Fig. 4-19

More precisely, each input A_i has the form of a physical quantity (say, voltage level), of which we choose to distinguish two states, denoted 0 and 1. The state 1 occurs if A_i is true, and the state 0 if A_i is false. The output of the and-gate is likewise in two possible states, 0 and 1: it is 1 if and only if $A_1 \& A_2 \& \ldots \& A_n$ is true, and it is 0 if and only if $A_1 \& A_2 \& \ldots \& A_n$ is false. Often the state of an input or output is taken to be 1 if it is transmitting current and 0 if not. Arithmetically, the output of an and-gate is the *product* of the inputs.

Another common element of a logic circuit is an or-gate $\textcircled{\vee}$. If the inputs are $A_1, \ldots, A_n$ ($n \geqq 2$), then the output is $A_1 \vee A_2 \vee \cdots \vee A_n$.

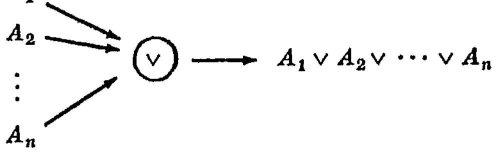


Fig. 4-20

Thus the output is 1 if and only if the output of at least one A_i is 1. Arithmetically, the output is the maximum of the inputs.

An *inverter* $\bigcirc\!\!\!\!\neg$ is a device which has one input A and produces as its output $\neg A$. Thus the output is 1 if the input is 0 and the output is 0 if the input is 1.

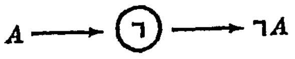


Fig. 4-21

A *logic circuit* is defined as a circuit constructed from various inputs by means of and-gates, or-gates, inverters, and possibly also other devices for performing truth-functional operations.

The actual electronic (or mechanical) devices used to construct and-gates, or-gates, and inverters vary with the state of technology. For this reason, it is most convenient to ignore (as far as possible) questions of hardware (diodes, transistors, vacuum tubes, etc.). This also holds for our treatment of switching circuits. Readers interested in the physical realization of switching and logic circuits can consult [53] and [13].

Example 4.8.

To construct a logic circuit producing the output $A_1 \leftrightarrow A_2$, notice that $A_1 \leftrightarrow A_2$ is logically equivalent to $(A_1 \& A_2) \vee (\neg A_1 \& \neg A_2)$ (Fig. 4-22) as well as to $(A_1 \& A_2) \vee \neg(A_1 \vee A_2)$ (Fig. 4-23). Clearly the second logic circuit is simpler.

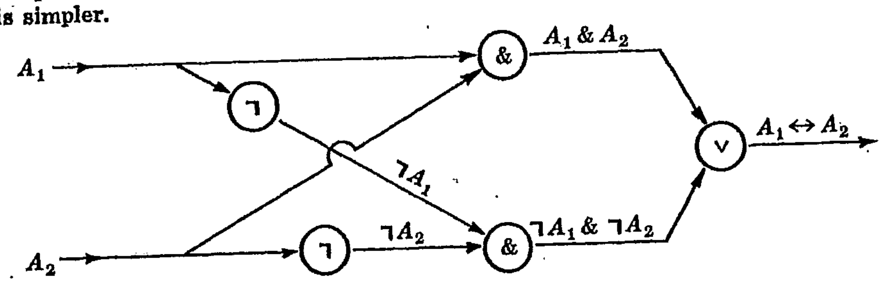


Fig. 4-22

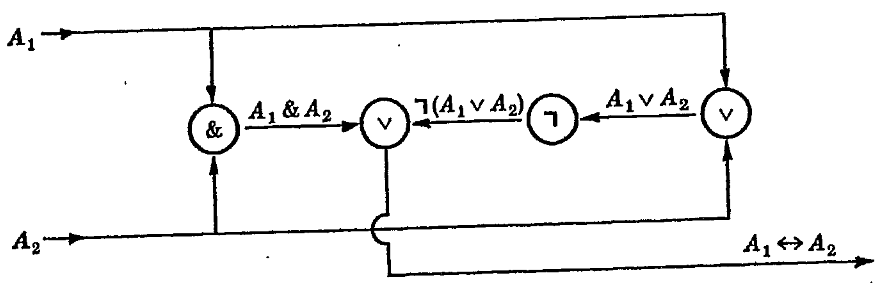


Fig. 4-23

Example 4.9.

Construct a logic circuit producing

$$(A_1 \& \neg A_2) \vee \neg A_1 \vee (A_2 \& A_3) \tag{1}$$

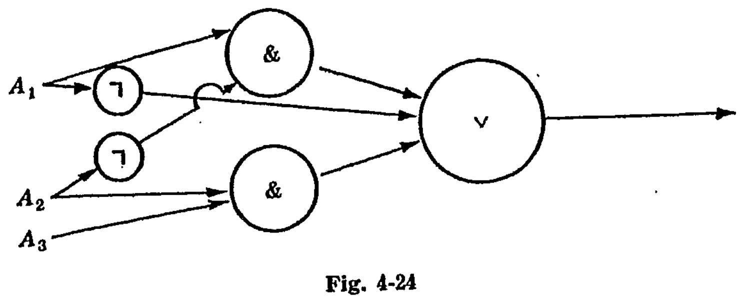


Fig. 4-24

Notice that, instead of a logic circuit, one could construct a series-parallel switching circuit through which current flows if and only if (*1*) is true (see Fig. 4-25).

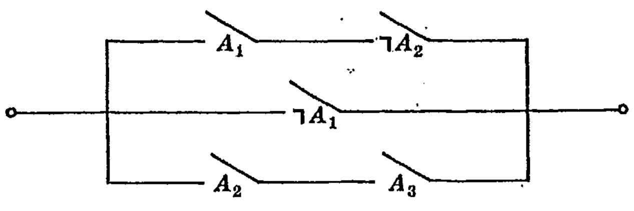


Fig. 4-25

Example 4.9 indicates that the same effect can be obtained by logic circuits as by series-parallel switching circuits. Indeed, connection in series corresponds to an and-gate, while connection in parallel corresponds to an or-gate.

4.5 THE BINARY NUMBER SYSTEM

We are accustomed to using the decimal number system. Thus 34,062 stands for the number $2+6\cdot 10+0\cdot 10^2+4\cdot 10^3+3\cdot 10^4$. In general, any positive integer can be represented in one and only one way in the form

$$a_0 + a_1\cdot 10 + a_2\cdot 10^2 + \cdots + a_k\cdot 10^k$$

where $0 \leq a_i \leq 9$ for $0 \leq i \leq k$ and $a_k > 0$. This number is denoted $a_k a_{k-1}\cdots a_2 a_1 a_0$ in standard decimal notation.

However, for any integer $r > 1$, every positive integer n can be represented uniquely in the form

$$a_0 + a_1\cdot r + a_2\cdot r^2 + \cdots + a_m\cdot r^m$$

where $0 \leq a_i \leq r-1$ for $0 \leq i \leq m$ and $a_m > 0$. This can be proved by induction on n.

In particular, every positive integer can be represented in binary notation:

$$a_0 + a_1\cdot 2 + a_2\cdot 2^2 + \cdots + a_m\cdot 2^m$$

where $0 \leq a_i \leq 1$ for $0 \leq i \leq m$ and $a_m = 1$.

Example 4.10.

The number 23 (in decimal notation) has the binary representation 10111, i.e. $2^4 + 2^2 + 2 + 1$. The decimal number 101 has the binary representation 1100101, i.e. $2^6 + 2^5 + 2^2 + 1$.

A procedure for finding the binary representation of a number n is to find the highest power 2^m which is $\leq n$, subtract 2^m from n, then find the highest power 2^j which is $\leq n - 2^m$, etc.

Further examples:

Decimal Notation	Binary Notation	Decimal Notation	Binary Notation
1	1	11	1011
2	10	16	10000
3	11	35	100011
4	100	52	110100
5	101	117	1110101
6	110		
7	111		
8	1000		

4.6 MULTIPLE OUTPUT LOGIC CIRCUITS

Occurrence of the same logic circuit as part of other logic circuits suggests the use of logic circuits with more than one output.

Example 4.11.

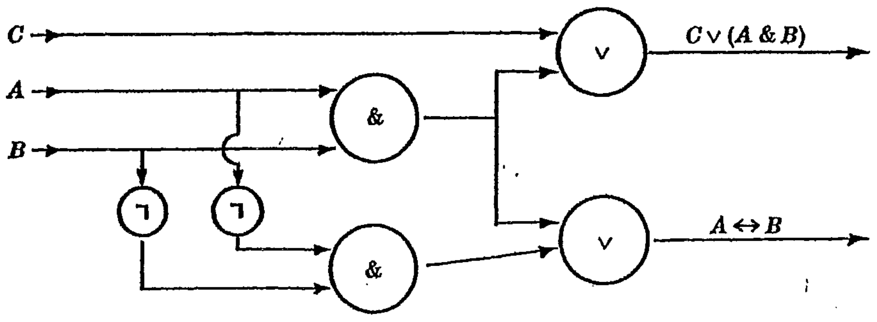


Fig. 4-26

Example 4.12.

Two numbers in binary notation are added in the same way as numbers in decimal notation.

Binary notation:	100101	Decimal notation:	37
	10111		23
	111100		60

If we just consider the addition of one digit numbers, 0 and 1, we have the following values for the *sum digit* s and the *carry digit* c.

A	B	s	c
1	1	0	1
0	1	1	0
1	0	1	0
0	0	0	0

Thus s corresponds to the exclusive-or (which we shall denote $A \oplus B$), while c corresponds to the conjunction.

If we wished to construct a separate logic circuit for s we would obtain

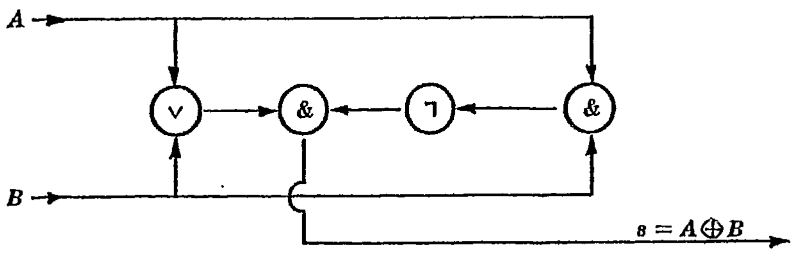


Fig. 4-27

Similarly we can construct a logic circuit for c:

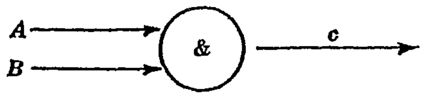


Fig. 4-28

However, we can combine these two circuits into a single multiple output circuit:

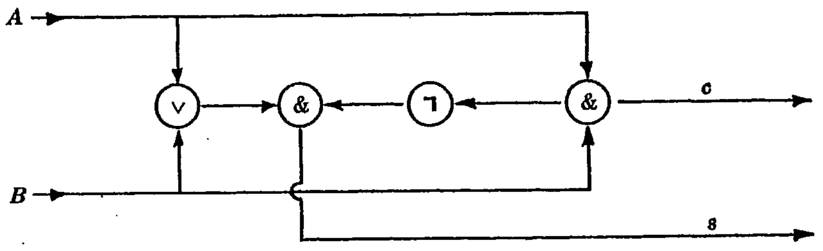


Fig. 4-29

The circuit in Fig. 4-29 is called a *half-adder.*

If we wish to add two single-digit numbers A and B†, while taking into account a carry-over C from a previous addition, we obtain the table

A	B	C	s	c
1	1	1	1	1
0	1	1	0	1
1	0	1	0	1
0	0	1	1	0
1	1	0	0	1
0	1	0	1	0
1	0	0	1	0
0	0	0	0	0

Thus s corresponds to the statement form

$$(C \,\&\, \neg(A \oplus B)) \vee ((A \oplus B) \,\&\, \neg C)$$

which is logically equivalent to $(A \oplus B) \oplus C$. The carry-over c corresponds to the statement form

$$(A \,\&\, B) \vee (C \,\&\, (A \oplus B))$$

We can use the circuit constructed for $A \oplus B$ in Fig. 4-27 to obtain the following diagram corresponding to the above statement.

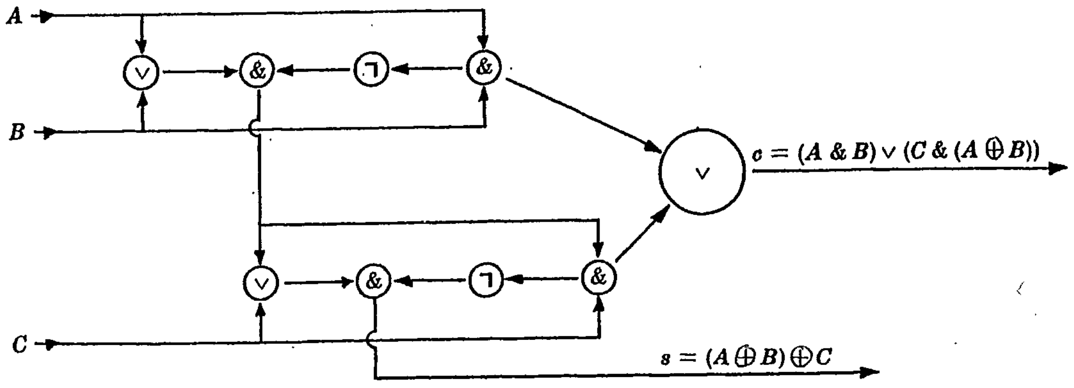
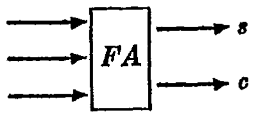
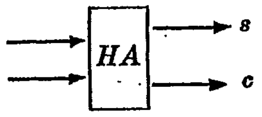


Fig. 4-30

The circuit of Fig. 4-30 is called a *full adder.*

†Actually, A is the proposition that the first number is 1, and B is the proposition that the second number is 1.

We can construct a circuit for adding two three-digit binary numbers $A_2A_1A_0$ and $B_2B_1B_0$. Let  represent a full-adder, and let  represent a half-adder. Then the sum is represented in Fig. 4-31 by $cs_2s_1s_0$.

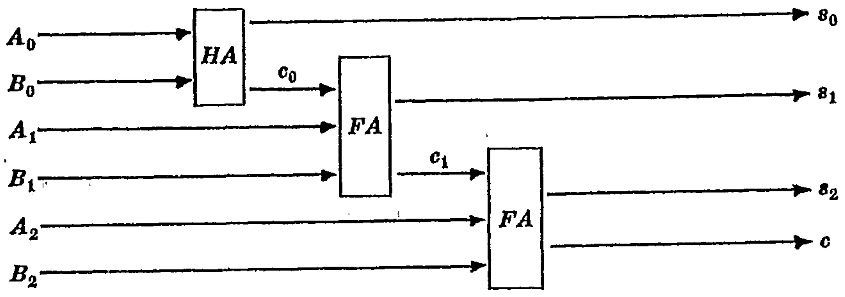


Fig. 4-31

LOGIC CIRCUITS

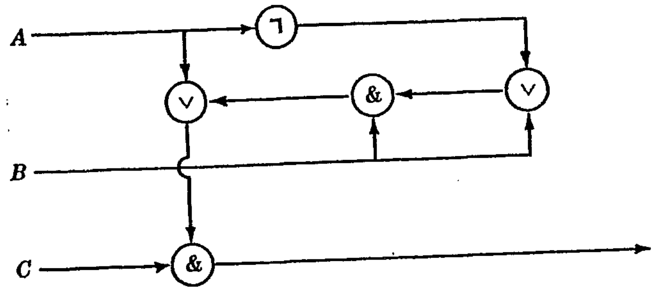
4.34. Construct logic circuits corresponding to the following statement forms.

(*a*) $(A \& \neg B) \vee (B \& (C \vee \neg A))$

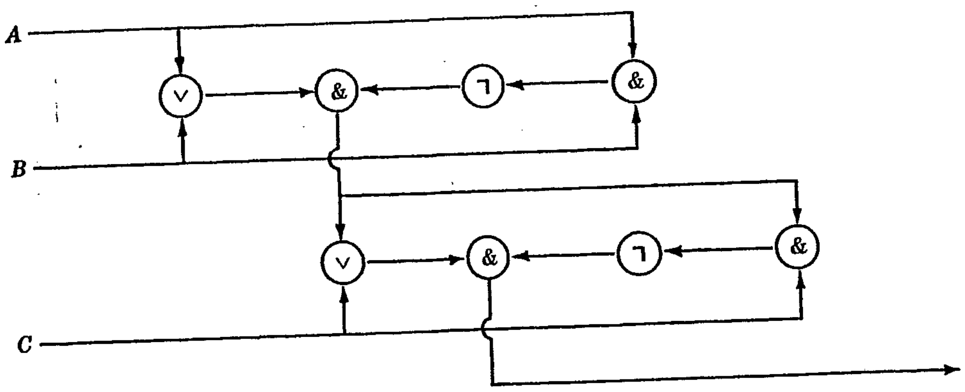
(*b*) $(A \rightarrow B) \vee \neg C$

4.35. Write down statement forms corresponding to the following logic circuits.

(*a*)



(*b*)



BINARY NUMBER SYSTEM

4.36. Write the binary notation for the following numbers given in decimal notation: 35, 74, 155, 320.

4.37. Write the decimal notation for the following numbers given in binary notation: 10110, 111011, 10001101.

4.38. Write the ternary notation (base 3) for the numbers given in Problem 4.36.

4.39. Write the decimal notation for the following numbers in ternary notation: 12011, 222110, 10110.

4.40. Solve Problem 4.36 for base 6 and base 8, instead of base 2.

4.41. Do the following additions in the binary system (and check by going over to the decimal system).

(*a*)
```
   11101
 + 1011
```
(*b*)
```
   11000
 + 101110
```

4.42. Do the following multiplications in the binary system (and check by going over to the decimal system).

(*a*)
```
   11101
 × 1011
```
(*b*)
```
   11000
 × 1010
```

4.43. (*a*) Let a non-negative integer less than 10 be given in binary notation: $a_3a_2a_1a_0$. Letting A_i stand for "a_i is 1", construct a logic circuit producing the statement that the given integer is a perfect square.

(*b*) Same as (*a*), except that the resulting proposition states that the given integer is even.

(*c*) Same as (*a*), except that the resulting proposition states that the given integer is a perfect cube.

4.44. (*a*) Using half-adders and full adders, draw a logic circuit which carries out the addition of two four-digit binary numbers.

(*b*) Same as (*a*), except that three two-digit numbers are to be added.