Digital Circuits

Logic and Design

Ronald C. Emery

Library of Congress Cataloging in Publication Data

Emery, Ronald C.

Digital circuits.

(Electrical engineering and electronics ; 25)
Bibliography: p.
Includes index.
1. Digital electronics. 2. Logic circuits. I. Title.
II. Series.
TK7868.DE45 1985 621.3819'5835 85-1561
ISBN 0-8247-7397-7

MARCEL DEKKER, INC.
270 Madison Avenue, New York, New York 10016

Current printing (last digit):
10 9 8 7 6 5 4 3 2 1

PRINTED IN THE UNITED STATES OF AMERICA

Preface

This textbook is intended to introduce the student of electronics to the fundamentals of digital circuits, both combinational and sequential, in a reasonable and systematic manner. The text material requires only algebra as a mathematical background. Useful applications for digital circuit design are stressed throughout the text.

This book is designed for a one-semester course in digital systems theory and design in junior colleges, technical institutes, and university technology and engineering programs. It is also intended for working technicians and technologists who require more knowledge of digital systems.

The text proceeds from basic logic concepts to circuits and designs. Chapter 1 explains the arithmetical foundation of the digital world. Chapter 2 discusses the concepts of Boolean algebra and their application to electronic circuitry. Chapter 3 covers minimization techniques of combinational digital circuits. Chapter 4 deals with the application of combinational circuit concepts to practical problems. Chapter 5 introduces sequential circuits and their characteristics and limitations. Chapter 6 examines methods of developing general sequential circuitry.

Each chapter consists of an introduction, text material, examples, a summary, and problems. A basic understanding of transistor circuits is helpful but not necessary. An appendix includes eight experiments that both professors and students may find useful.

I am indebted to all who have contributed to the preparation of this book. In particular, I am grateful to those students who have provided constructive comments over the years, to colleagues who have used the text in note form and who have furnished insights for improvements, and most importantly to the secretaries who have typed, revised, re-revised, and have been forced to decode my handwriting.

Ronald C. Emery

Contents

1
Numbers

INTRODUCTION

It is probable that mankind has been fascinated with numbers and their uses since the dawn of civilization. The ability to count, to add and to subtract were among the first of reasoning man's accomplishments. The origin of the decimal system is most likely due to the fact that our two hands contain 10 digits. A computer or control system has no fingers and operates only on "ON-OFF" type information. This feature requires that we know the binary system (the ON-OFF system). In this chapter we shall examine the binary number system and its application as well as the octal, hexadecimal, BCD and Gray Code systems.

THE BINARY SYSTEM

The first step in the discussion of logic design is the realization that the binary number system is a useful tool in the design of logic systems and in the case of the digital computer is an inherent feature. Therefore, our first task will be to define the binary system.

The binary system permits only two symbols, namely, 0 and 1, compared to the 10 symbols of the decimal system. The decimal symbols are 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9. Using the two permitted symbols of the binary system, we create all numbers, both integer and fractional. We do this by assigning values to the binary digits depending on where they appear in the sequence of symbols. For example, let us look at an integer string of binary symbols such as

10110.0

The point is the binary point. Starting immediately to its left, we assign to that position the value $A_0 \times 2^0$, where A_0 may be either a 0 or 1. It is 0 in the example. This position is called the "least significant bit" (LSB) position. The second position to the left of the binary point is assigned the value $A_1 \times 2^1$, and as before A_1 may be a 0 or 1. A_1 in the example is 1 so the 2^1 position has a decimal value of $1 \times 2^1 = 1 \times 2 = 2$.

We continue this process and arrive at the conclusion that 10110.0 can be written as

$$1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

$$= 16 + 0 + 4 + 2 + 0 = 22 \text{ as a decimal.}$$

If we want to get mathematical we would state that the integer part of the binary number expressed generally is:

$$N_I = A_n \times 2^n + A_{n-1} \times 2^{n-1} + \cdots + A_3 \times 2^3 + A_2 \times 2^2 + A_1 \times 2^1 + A_0 \times 2^0 \qquad \text{(Def. 1.1)}$$

Note how this corresponds to the decimal system.

$$A_n \times 10^n + A_{n-1} \times 10^{n-1} + \cdots + A_2 \times 10^2 + A_1 \times 10^1 + A_0 \times 10^0$$

We may have binary symbols to the right of the binary point as in the following example:

Example 1.1

Problem: Determine the value of the fractional binary number 0.1011 and find the expression for any fractional binary number.

Solution: Beginning immediately to the right of the binary point, we assign to that position the value $A_{-1} \times 2^{-1}$ where A_{-1} is either a 0 or 1. In the example it is a 1; thus the position has a value of $1 \times 1/2 = 0.5$. The second position to the right of the binary point is assigned the value $A_{-2} \times 2^{-2}$ and as before A_{-2} may be a 0 or 1. In the example A_{-2} is a 0 so $0 \times 2^{-2} = 0$.

We continue this process and come to the conclusion that 0.1011 can be written as

$$0.\left(1 \times \frac{1}{2} + 0 \times \frac{1}{2^2} + 1 \times \frac{1}{2^3} + 1 \times \frac{1}{2^4}\right)$$

$$= 0.5 + 0 + 0.125 + 0.0625$$

$$= 0.6875$$

The general expression for any fractional binary number is:

$$N_F = 0.A_{-1} \times 2^{-1} + A_{-2} \times 2^{-2} + \cdots + A_{-m} \times 2^{-m} + A_{-(m+1)} \times 2^{(m+1)} \qquad \text{(Def. 1.2)}$$

We now have the ability to convert any binary number, integer, fractional or mixed, to a decimal number.

Let's try our newly gained knowledge on some examples.

Example 1.2

Problem: Convert the following binary quantities to decimal.

(a) 110010.0
(b) 0.001101
(c) 111.101
(d) 101.111
(e) 0.11001
(f) 1101.0

Solution:

(a) $110010.0 = 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
$= 32 + 16 + 2 = 50$

(b) $0.001101 = 0 \times 0.5 + 0 \times 0.25 + 1 \times 0.125 + 1 \times 0.0625 + 0 \times 0.03125 + 1 \times 0.015625$
$= 0.203125$

(c) $N_I = 111.0 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 4 + 2 + 1 = 7$
$N_F = 0.101 = 1 \times 0.5 + 0 \times 0.25 + 1 \times 0.125 = 0.625$
$N_I + N_F = 7.0 + 0.625 = 7.625$

(d) $N_I = 101.0 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 4 + 0 + 1 = 5.0$
$N_F = 0.111 = 1 \times 0.5 + 1 \times 0.25 + 1 \times 0.125 = 0.875$
$N_I + N_F = 5.0 + 0.875 = 5.875$

(e) $0.11001 = 1 \times 0.5 + 1 \times 0.25 + 0 \times 0.125 + 0.625 + 1 \times 0.03125 = 0.5 + 0.25 + 0.03125 = 0.78125$

(f) $1101.0 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 8 + 4 + 1 = 13$

Integer Decimal to Integer Binary

It is sometimes (though not often) necessary to convert from the decimal system to the binary system. To accomplish this feat we effectively reverse mathematically the procedure of the preceding section.

The process is a mechanical one and is best shown by example.

Example 1.3

Problem: Convert 6 in decimal to a binary number.

Solution:

(a) Divide the decimal number by 2 and record the quotient and the remainder. The resulting remainder of this division is the least significant bit of the result.

$$2 \overline{)\,6}$$
$$3 \quad \text{remainder 0 (LSB)}$$

(b) Do another division by 2; record the quotient and remainder.

$$2 \overline{)\,6}$$
$$3 \quad \text{Remainder } 0 \text{ (LSB)}$$
$$1 \quad \text{Remainder } 1$$

(c) Do another division by 2; record the quotient and remainder. If the quotient is zero, the problem is complete.

$$2 \overline{)\,6}$$
$$3 \quad \text{Remainder } 0 \text{ (LSB)}$$
$$1 \quad \text{Remainder } 1$$
$$0 \quad \text{Quotient is zero, Remainder } 1$$

(d) The binary equivalent of the decimal number 6 is 110.

Example 1.4

Problem: Convert 135 in the decimal to a binary number.

Solution:

2	135	
	67	1
	33	1
	16	1
	8	0
	4	0
	2	0
	1	0
	0	1

The binary equivalent of decimal 135 is 10000111.

Decimal Fraction to Binary Fraction Conversion

Occasionally it may be useful to convert a fraction in decimal form to a binary fraction. That process is shown in the following example.

Example 1.5

Problem: Convert the decimal 0.25 to a binary fraction.

Solution:

(a) Multiply the decimal number by 2 and record the integer portion and fractional portion. The resulting integer of the first multiplication is the "most significant bit" (MSB) of the binary fraction.

Integer		
		0.25
0	X	2
		0.50

(b) Do another multiplication by 2; record the integer and fractional parts. If the fractional portion is zero, the problem is complete.

		0.25
	X	2
(MSB) 0		0.50
	X	2
1		1.00

Fractional part is zero

The binary equivalent of decimal 0.25 is 0.01.

Example 1.6

Problem: Convert the decimal 0.86 to binary.

Solution:

$$
\begin{array}{rr}
 & 0.86 \\
\times & 2 \\
\hline
 & 1.72 \\
\times & 2 \\
\hline
 & 1.44 \\
\times & 2 \\
\hline
 & 0.88 \\
\times & 2 \\
\hline
 & 1.76 \\
\times & 2 \\
\hline
 & 1.52 \\
\times & 2 \\
\hline
 & 1.04
\end{array}
$$

Note that the integer part (if it is one) is not used in the next multiplication.

This appears as if it could continue indefinitely. Where is it reasonable to stop the process? The answer to that question is not clear-cut. It depends to a large extent upon the application of the conversion.

Technically, it can be shown that it requires 3.32 binary bits to represent a decimal digit. Therefore, we conclude that the conversion can be reasonably halted after 3 to 4 bit per digit conversion. Considerable error is introduced if we stop converting too soon. For example, in Example 1.6 if we say that 0.110 is equivalent to 0.86, then we are saying that 0.75 (the true equivalent of 0.110) is approximately equal to 0.86 when, in fact, the error is nearly 12%. If we use 0.1101, the error is about 6%. If we use 0.110111 as the binary equivalent to 0.86, the resulting error will be less than 1%.

Mixed Number Conversion

The conversion of mixed numbers, either decimal-to-binary or binary-to-decimal, is a two-part process. The procedure for the conversion of a mixed binary number to decimal is shown in the following example.

Example 1.7

Problem: Convert the binary number 110.011 to decimal.

Solution: Convert the integer part (N_1) according to the rules of Def. 1.1.

$1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 6.$

Then convert the fractional part (N_F) according to the rules of Def. 1.2.

$$0.\left(0 \times \frac{1}{2} + 1 \times \frac{1}{4} + 1 \times \frac{1}{8}\right) = 0 + 0.25 + 0.125 = 0.375$$

The decimal equivalent of 110.011 is 6.375. The procedure for the conversion of a mixed decimal number to binary is as in the following example:

Example 1.8

Problem: Convert 14.68 decimal to binary.

Solution: Use the methods in Example 1.3 to convert the integer part.

2	14	
	7	0
	3	1
	1	1
	0	1

The integer part is 1110.0.
Convert the fractional part using the methods in Example 1.6.

	0.68
	× 2
1	1.36
	× 2
0	0.72
	× 2
1	1.44
	× 2
0	0.88
	× 2
1	1.76
	× 2
1	1.52
	× 2
1	1.04

The fractional part is 0.1010111.
The conversion of 14.68 decimal to binary is 1110.1010111.

Arithmetic with Binary Numbers

Although we will have little need to perform arithmetic operations using binary numbers, it will be instructive to examine the fundamentals of binary addition and subtraction. Digital computers do all of their arithmetical computations using binary numbers so it will be well if we have some familiarity with these operations.

Binary Addition

The first operation of importance to explore is that of addition. The addition rules for the binary number system are very simple. They are as follows:

$$\begin{array}{cccc} 0 & 0 & 1 & 1 \\ +0 & +1 & +0 & +1 \\ \hline 0 & 1 & 1 & 10 \end{array}$$

By successively adding 1 to binary numbers, Table 1.1 shows binary-decimal conversions up to decimal 15. These rules allow us to count in binary and thus build such a binary table.

Table 1.1

Binary	Decimal
0	0
1	1
10	2
11	3
100	4
101	5
110	6
111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

To add binary numbers we use the addition facts and apply the concept of "carry" from decimal arithmetic.

Example 1.9

Problem: Add binary 110 and binary 110.

Solution:

$$\begin{array}{rr} 110 & (6) \\ +\ 110 & +\ (6) \\ \hline 1100 & (12) \end{array}$$

We start with the least significant bit and say "0 + 0 = 0 with no carry," then "1 + 1 = 10; write down the 0 and carry the 1," then "1 + 1 = 10 and carry-on of 1 = 11; write down the 1 and carry the 1," then "1 + 0 = 1 with no carry and we are done."

Adding three or more numbers causes no real hardship, just an extension of the rules.

Example 1.10

Problem: Add 1010, 101 and 10011.

Solution:

$$\begin{array}{rr} 101 & (5) \\ 1010 & (10) \\ +\ 10011 & +\ (19) \\ \hline 100010 & (34) \end{array}$$

At this point, we will digress to introduce an electronic component that performs binary addition. This component is a medium-scale integrated (MSI) circuit chip. These chips contain large numbers of transistors. However, that fact is not important here. We are interested in how the chip operates.

Typical of this type of adder chip is the 54283. The 54283 is called a 4-Bit Binary Full Adder, where the term Full Adder means that the circuitry will accept a carry bit from some previous circuit. The chip has 16 pins as shown in Fig. 1.1. Nine of the leads are input leads, five are output leads, one is the power input (+5Vdc) and the remaining pin is the ground pin. Of the nine input leads, four are used to input a 4-bit binary number; these are A_1, A_2, A_3 and A_4. Four others are used to input a second 4-bit binary number; these are B_1, B_2, B_3 and B_4. The ninth input pin is the input for the carry-in from a previous adder; if there is no previous adder, this pin is

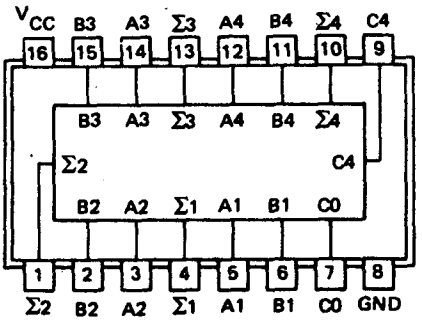

Fig. 1.1 4-Bit Binary Full Adder 54283. (Courtesy of Texas Instruments, Inc.)

grounded. For the five output pins, four are the sum outputs Σ_1, Σ_2, Σ_3 and Σ_4. The fifth pin is the carry-out pin for the sum. In these circuits +5Vdc is used to represent a binary 1, and ground is used to represent a binary 0.

Example 1.11

Problem: Use the 54283 to add 1101 and 0101.

Solution: There is no carry-in so pin 7 (Co) is grounded. Pin 8 (Gnd) is grounded. Pin 16 (+Vcc) is placed at +5Vdc. Let the binary number 1101 be number A, then

A_4	A_3	A_2	A_1
1	1	0	1
+5V	+5V	Gnd.	+5V
Pin 12	Pin 14	Pin 3	Pin 5

and

B_4	B_3	B_2	B_1
0	1	0	1
Gnd.	+5V	Gnd.	+5V
Pin 11	Pin 15	Pin 2	Pin 6

$$\begin{array}{r} 1101 \\ +\ 0101 \\ \hline 10010 \end{array}$$