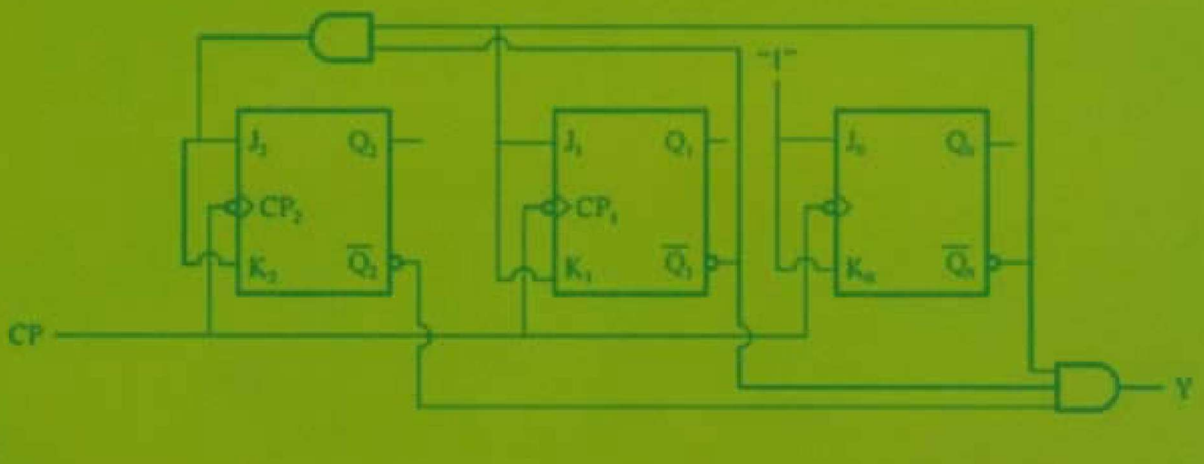
世纪出版 • 普通高等教育“十三五”规划教材

Digital Circuits

数字电路

Edited by Shangzhi Xin

忻尚芝 编著



上海科学技术出版社
Shanghai Scientific & Technical Publishers

世纪出版·普通高等教育"十三五"规划教材

Digital Circuits

数 字 电 路

Edited by Shangzhi Xin

忻尚芝 编著

上海科学技术出版社

Shanghai Scientific & Technical Publishers

图书在版编目(**CIP**)数据

数字电路＝Digital Circuits/忻尚芝编著. —上海：
上海科学技术出版社，2017.8
ISBN 978-7-5478-3645-3

Ⅰ.①数… Ⅱ.①忻… Ⅲ.①数字电路 Ⅳ.①TN79

中国版本图书馆 CIP 数据核字(2017)第 170762 号

Digital Circuits
数字电路
Edited by Shangzhi Xin
忻尚芝 编著

上海世纪出版股份有限公司
上 海 科 学 技 术 出 版 社 出版
(上海钦州南路 71 号 邮政编码 200235)

上海世纪出版股份有限公司发行中心发行
200001 上海福建中路 193 号 www.ewen.co
印刷

开本 787×1092 1/16 印张 6.5
字数：150 千字
2017 年 8 月第 1 版 2017 年 8 月第 1 次印刷
ISBN 978-7-5478-3645-3/TM・59
定价：28.00 元

Synopsis

This book has nine chapters. It includes number systems, codes, Boolean algebra, the Karnaugh map, logic gates, logic description method, combinatorial logic circuit, ICs of combinational logic circuits, flipflops and sequential logic circuits. The purpose of learning this book is to analyse and design the combinatorial circuit and sequential circuit.

This book is written for university students as a textbook of digital circuit course. It is also suitable to use as a reference book for electrical technicians.

Preface

This book, *digital circuits*, is a textbook for electrical major students in universities. There are enough selected examples for each chapter to help students to understand the principles and methods as well as the analysis and design processes. At the end of each chapter there are corresponding problems to improve the skills of digital circuit analysis and design. After learning this book students will have the ability to solve practical problems of electrical engineering.

The prerequisites of digital circuit course are general physics and electrical circuit. The suggested semester schedule of teaching is four hours per week. The lab of this course will have sixteen hours for eight experiments.

The financial support for published this book was the textbook development project of University of Shanghai for Science and Technology (USST). Andrew W. Ni from University of Michigan — Ann Arbor helped to review this book. The editor and staff at Shanghai Science & Technical Publishers did editorial work professionally to make this book publish successfully. I would express my heartfelt gratitude.

The book contains teaching preparation notes and experience summaries of the author who has been teaching digital circuit course for more than twenty years. But there are still some places in this book need to be improved. All of the information and suggestions could improve the content description, chapter arrangement, example choice, problem selection, even the experiment content and quantity are especially valuable. Welcome the corrections, comments, and suggestions from the readers and users of this book. Contact email address is xinsz@usst. edu. cn.

Shangzhi Xin

2017. 5

Contents

Chapter 1 Number Systems

In this chapter we will introduce number systems, decimal, binary, octal, and hexadecimal numbers. We will also study the ways to convert numbers between them.

We use the decimal number system daily. We are very familiar with it. But the digital electronic circuits use the binary number system mainly. Also they use octal number system and hexadecimal number system. We will learn the number systems and their conversions. Shown in Fig. 1. 1.

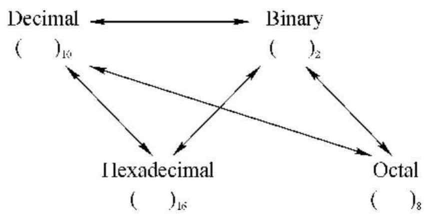


Fig. 1. 1 Number system and their conversion

1. 1 Decimal Number System

In the decimal number system there are ten different digits, from zero to nine. It is a base-ten system.

The decimal number 3794 means

$$(3794)_{10} = 3\times10^3 + 7\times10^2 + 9\times10^1 + 4\times10^0$$

Similarly the decimal number 503. 98 means

$$(503.98)_{10} = 5\times10^2 + 0\times10^1 + 3\times10^0 + 9\times10^{-1} + 8\times10^{-2}$$

1. 2 Binary Number System

It is a base-two system. There are two different digits, zero and one, in binary number system.

The binary number 110101 means

$$(110101)_2 = 1\times 2^5 + 1\times 2^4 + 0\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0 = (53)_{10}$$

And the binary number 111. 01 means

$$(111.01)_2 = 1\times 2^2 + 1\times 2^1 + 1\times 2^0 + 0\times 2^{-1} + 1\times 2^{-2} = (7.25)_{10}$$

1.3 Octal Number System

Octal number system is a base-eight system. There are eight different digits, from zero to seven.

$$(743)_8 = 7\times 8^2 + 4\times 8^1 + 3\times 8^0 = (483)_{10}$$

$$(701.35)_8 = 7\times 8^2 + 0\times 8^1 + 1\times 8^0 + 3\times 8^{-1} + 5\times 8^{-2} = (449.453125)_{10}$$

1.4 Hexadecimal Number System

Hexadecimal number system is a base-sixteen system. There are sixteen different digits, zero through fifteen. They are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

$$(3F0)_{16} = 3\times 16^2 + 15\times 16^1 + 0\times 16^0 = (1008)_{10}$$

$$\begin{aligned}(39A.BC)_{16} &= 3\times 16^2 + 9\times 16^1 + 10\times 16^0 + 11\times 16^{-1} + 12\times 16^{-2}\\ &= (922.734375)_{10}\end{aligned}$$

Summary sections 1. 1～1. 4, we have the general formula of binary, octal, and hexadecimal numbers to decimal number conversion

$$(\quad)_{10} = \sum_{i=-\infty}^{\infty} (\text{column value})_i \times (\text{base-}N)^i$$

1.5 Decimal to Binary, Octal, and Hexadecimal Conversion

We will introduce the comparision method.

Example 1. 1: $(53)_{10} = (?)_2$

Solution: We have known the values of $(\text{base-}N)^i$ for each column.

… 64 32 16 8 4 2 1

1

$53-32=21$

1

$21-16=5$

1

$(5-4)=1$

1

Then $$(53)_{10}=(110101)_2$$

Example 1.2: $(7.25)_{10}=(?)_2$

Solution: … 8 4 2 1. 0.5 0.25 0.125 …

1 1 1 1

Then $$(7.25)_{10}=(111.01)_2$$

Example 1.3: $(483)_{10}=(?)_8$

Solution: Similarly we write out the values of $(\text{base-}N)^i$, that is 8^i.

… 512 64 8 1

7

$483-7\times 64=35$

4

$35-4\times 8=3$

3

Then $$(483)_{10}=(743)_8$$

Example 1.4: $(449.453125)_{10}=(?)_8$

Solution: … 512 64 8 1. 0.125 0.015625 0.001953125 …

7 1 3 5

Then $$(449.453125)_{10}=(701.35)_8$$

Example 1.5: $(1008)_{10}=(?)_{16}$

Solution: … 4096 256 16 1

3

$1008-3\times 256=240$

F

Then $(1008)_{10}=(3F0)_{16}$

Example 1.6: $(922.734375)_{10}=(?)_{16}$

Solution: … 4096 256 16 1. 0.0625 0.00390625 0.000244140625 …

3 9 A B C

Then $(922.734375)_{10}=(39A.BC)_{16}$

1.6 Binary, Octal, and Hexadecimal Conversion

We can convert binary numbers to octal or hexadecimal numbers, and octal or hexadecimal numbers to binary numbers. But it is hard to convert between octal number and hexadecimal number directly. Binary or decimal must be used as a bridge.

Example 1.7: $(10111110)_2=(?)_8$

Solution: We find that three bits binary digits represent one bit octal digit. Starting at the binary point, mark out each group of three bits, and add leading zeros as needed.

$$(10111110)_2 = (010\ 111\ 110)_2 = (276)_8$$

Example 1.8: $(1110.1011)_2=(?)_8$

Solution: Similar for the right part of binary point, mark out each group of three bits and add tail zeros as needed.

$$(1110.1011)_2 = (001\ 110.101\ 100)_2 = (16.54)_8$$

Example 1.9: $(1110100110)_2=(?)_{16}$

Solution: We know that there are four bits binary digits to represent one bit hexadecimal digit. We will mark out each group of four bits binary digits.

$$(1110100110)_2 = (0011\ 1010\ 0110)_2 = (3A6)_{16}$$

Example 1.10: $(1100.011011)_2=(?)_{16}$

Solution: $(1100.011011)_2=(1100.0110\ 1100)_2=(C.6C)_{16}$

Example 1.11: $(30.62)_8=(?)_2$

Solution: Conversion from octal back to binary is just to write one bit octal as three bits binary. Each one octal digit will hold three bits binary places. We can only omit the leading zeros left of point and the tail zeros right of point at the end.

$$(30.62)_8 = (011\ 000.110\ 010)_2 = (11000.11001)_2$$

Example 1.12: $(4C3.A)_{16} = (?)_2$

Solution: Similar with the way of example 1.11.

$$(4C3.A)_{16} = (0100\ 1100\ 0011.1010)_2 = (10011000011.101)_2$$

Problems

1.1 Convert the following numbers.

$(10110)_2 = \_\_\_\_\_\__{10}$ $(0111.11)_2 = \_\_\_\_\_\__{10}$

$(110001)_2 = \_\_\_\_\_\__{10}$ $(10101.011)_2 = \_\_\_\_\_\__{10}$

$(254)_8 = \_\_\_\_\_\__{10}$ $(37.56)_8 = \_\_\_\_\_\__{10}$

$(76540)_8 = \_\_\_\_\_\__{10}$ $(572.70)_8 = \_\_\_\_\_\__{10}$

$(AE0)_{16} = \_\_\_\_\_\__{10}$ $(103.2)_{16} = \_\_\_\_\_\__{10}$

$(4CB0)_{16} = \_\_\_\_\_\__{10}$ $(A45D.0BC)_{16} = \_\_\_\_\_\__{10}$

1.2 Express the following decimal numbers in binary, octal, and hexadecimal numbers (the convertion $\varepsilon \leqslant 2^{-4}$).

(1) 438 (2) 359.78

1.3 Convert the binary numbers to octal and hexadecimal numbers.

(1) $(1010011)_2$ (2) $(111.01011)_2$

1.4 Convert the octal numbers to binary numbers.

(1) $(700)_8$ (2) $(337.01)_8$

1.5 Convert the hexadecimal numbers to binary numbers.

(1) $(AF5)_{16}$ (2) $(102.B)_{16}$

1.6 Express the following decimal numbers in binary-coded decimal (BCD) form.

(1) $(17)_{10}$ (2) $(35.0312)_{10}$

1.7 Express the following BCD in decimal number.

(1) $(10010101)_{BCD}$ (2) $(001000010010.10010001)_{BCD}$

Chapter 2 Codes

When numbers、letters、or words are represented by a special group of "0" and "1" which can be easily identified in digital system. We say that they are been encoded. And we call these symbols of "0" and "1" codes.

2.1 Binary-Coded-Decimal Code

In nature we use decimal number system. Digital systems use binary number system for operation. For this reason, ten codes of 4-bit binary number are represented ten decimal numbers 0 to 9. These codes are Binary-Coded-Decimal code (BCD code), shown in Table 2.1. Normally they are 8421 BCD codes for the binary weights of the four bits being 2^3、2^2、2^1 and 2^0.

Table 2.1 Some commonly used BCD codes

Decimal Number	Weighted Code			Unweighted Code	
	8421 Code	2421 Code	5421 Code	Excess-three Code	Three-cyclic Code
0	0000	0000	0000	0011	0010
1	0001	0001	0001	0100	0110
2	0010	0010	0010	0101	0111
3	0011	0011	0011	0110	0101
4	0100	0100	0100	0111	0100
5	0101	1011	1000	1000	1100
6	0110	1100	1001	1001	1101
7	0111	1101	1010	1010	1111
8	1000	1110	1011	1011	1110
9	1001	1111	1100	1100	1010

In Table 2. 1 we find that three-cyclic code changes only one bit between two successive numbers. The 9 reset to 0 also only changes one bit number. Three-cyclic code is same to the Gray code rejected three codes at the beginning and the end. We will study Gray Code in section 2. 2.

2. 2 The Gray Code

Counting is a basic operation in digital system. The Gray Code is developed to represent a sequence of numbers. There is only one bit change between two successive numbers in the sequence. The Gray Code can reduce the error、increase the life of hardware and find the mistake of changing one more bit once time. The Gray Code can be any bit.

To convert binary number to Gray code, start on the most significant bit (MSB) and use it as the MSB of Gray code. Then compare the MSB with the next bit. If they are different, then the next bit of Gray Code is 1. If they are same, then the Gray code is 0. Three-bit and four-bit binary numbers and their Gray code are shown in Table 2. 2 and Table 2. 3.

Table 2. 2 3-bit Gray Code

MSB		LSB	MSB		LSB
B_2	B_1	B_0	G_2	G_1	G_0
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

Table 2. 3 4-bit Gray Code

MSB			LSB	MSB			LSB
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

The process of three-bit binary number converted to Gray code is shown in Fig. 2. 1.

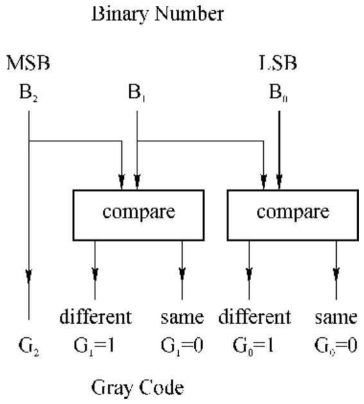


Fig. 2. 1 Converting binary number to Gray code

To convert Gray code to binary number. The process is shown in Fig. 2. 2.

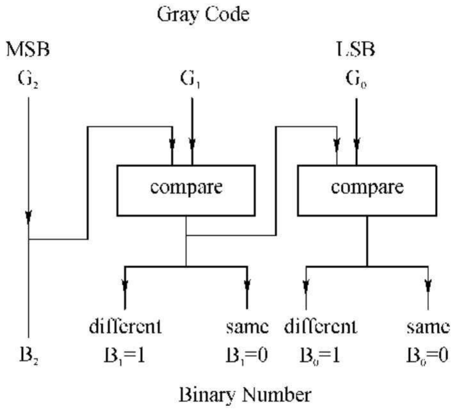


Fig. 2. 2 Converting Gray code to binary number

2. 3 ASCII Code

A digital system, for example a computer, must be able to handle nonnumerical information besides numerical data. The most important and widely used code is the American Standard Code for Information Interchange (ASCII). It is a seven-bit binary code and has $2^7 = 128$ codes. This is enough to represent all of the standard keyboard characters. Table 2. 4 shows a list of the ASCII code.

Table 2.4 ASCII Codes

Character	$B_6B_5B_4B_3B_2B_1B_0$	Character	$B_6B_5B_4B_3B_2B_1B_0$	Character	$B_6B_5B_4B_3B_2B_1B_0$	Character	$B_6B_5B_4B_3B_2B_1B_0$
NUL (null)	0 0 0 0 0 0 0	SP (space)	0 1 0 0 0 0 0	@	1 0 0 0 0 0 0	′	1 1 0 0 0 0 0
SOH(start of head)	0 0 0 0 0 0 1	!	0 1 0 0 0 0 1	A	1 0 0 0 0 0 1	a	1 1 0 0 0 0 1
STX (start of text)	0 0 0 0 0 1 0	"	0 1 0 0 0 1 0	B	1 0 0 0 0 1 0	b	1 1 0 0 0 1 0
ETX (end of text)	0 0 0 0 0 1 1	#	0 1 0 0 0 1 1	C	1 0 0 0 0 1 1	c	1 1 0 0 0 1 1
EOT (end of transmit)	0 0 0 0 1 0 0	$	0 1 0 0 1 0 0	D	1 0 0 0 1 0 0	d	1 1 0 0 1 0 0
ENQ (enquiry)	0 0 0 0 1 0 1	%	0 1 0 0 1 0 1	E	1 0 0 0 1 0 1	e	1 1 0 0 1 0 1
ACK (acknowledge)	0 0 0 0 1 1 0	&	0 1 0 0 1 1 0	F	1 0 0 0 1 1 0	f	1 1 0 0 1 1 0
BEL (bell)	0 0 0 0 1 1 1	`	0 1 0 0 1 1 1	G	1 0 0 0 1 1 1	g	1 1 0 0 1 1 1
BS (backspace)	0 0 0 1 0 0 0	(	0 1 0 1 0 0 0	H	1 0 0 1 0 0 0	h	1 1 0 1 0 0 0
HT (horizontal tab)	0 0 0 1 0 0 1	)	0 1 0 1 0 0 1	I	1 0 0 1 0 0 1	i	1 1 0 1 0 0 1
LF (line feed)	0 0 0 1 0 1 0	*	0 1 0 1 0 1 0	J	1 0 0 1 0 1 0	j	1 1 0 1 0 1 0
VT (vertical tab)	0 0 0 1 0 1 1	+	0 1 0 1 0 1 1	K	1 0 0 1 0 1 1	k	1 1 0 1 0 1 1
FF (form feed)	0 0 0 1 1 0 0	,	0 1 0 1 1 0 0	L	1 0 0 1 1 0 0	l	1 1 0 1 1 0 0
CR (carriage return)	0 0 0 1 1 0 1	-	0 1 0 1 1 0 1	M	1 0 0 1 1 0 1	m	1 1 0 1 1 0 1
SO (shift out)	0 0 0 1 1 1 0	.	0 1 0 1 1 1 0	N	1 0 0 1 1 1 0	n	1 1 0 1 1 1 0
SI (shift in)	0 0 0 1 1 1 1	/	0 1 0 1 1 1 1	O	1 0 0 1 1 1 1	o	1 1 0 1 1 1 1
DLE (data link escape)	0 0 1 0 0 0 0	0	0 1 1 0 0 0 0	P	1 0 1 0 0 0 0	p	1 1 1 0 0 0 0
DC1 (divice control 1)	0 0 1 0 0 0 1	1	0 1 1 0 0 0 1	Q	1 0 1 0 0 0 1	q	1 1 1 0 0 0 1
DC2 (divice control 2)	0 0 1 0 0 1 0	2	0 1 1 0 0 1 0	R	1 0 1 0 0 1 0	r	1 1 1 0 0 1 0
DC3 (divice control 3)	0 0 1 0 0 1 1	3	0 1 1 0 0 1 1	S	1 0 1 0 0 1 1	s	1 1 1 0 0 1 1
DC4 (divice control 4)	0 0 1 0 1 0 0	4	0 1 1 0 1 0 0	T	1 0 1 0 1 0 0	t	1 1 1 0 1 0 0
NAK (negative acknowledge)	0 0 1 0 1 0 1	5	0 1 1 0 1 0 1	U	1 0 1 0 1 0 1	u	1 1 1 0 1 0 1
SYN (synchronous idle)	0 0 1 0 1 1 0	6	0 1 1 0 1 1 0	V	1 0 1 0 1 1 0	v	1 1 1 0 1 1 0
ETB (end transmission block)	0 0 1 0 1 1 1	7	0 1 1 0 1 1 1	W	1 0 1 0 1 1 1	w	1 1 1 0 1 1 1
CAN (cancel)	0 0 1 1 0 0 0	8	0 1 1 1 0 0 0	X	1 0 1 1 0 0 0	x	1 1 1 1 0 0 0
EM (end of medium)	0 0 1 1 0 0 1	9	0 1 1 1 0 0 1	Y	1 0 1 1 0 0 1	y	1 1 1 1 0 0 1
SUB (substitute)	0 0 1 1 0 1 0	:	0 1 1 1 0 1 0	Z	1 0 1 1 0 1 0	z	1 1 1 1 0 1 0
ESC (escape)	0 0 1 1 0 1 1	;	0 1 1 1 0 1 1	[	1 0 1 1 0 1 1	{	1 1 1 1 0 1 1
FS (file separator)	0 0 1 1 1 0 0	<	0 1 1 1 1 0 0	\	1 0 1 1 1 0 0			1 1 1 1 1 0 0
GS (group separator)	0 0 1 1 1 0 1	=	0 1 1 1 1 0 1	]	1 0 1 1 1 0 1	}	1 1 1 1 1 0 1
RS (record separator)	0 0 1 1 1 1 0	>	0 1 1 1 1 1 0	∧	1 0 1 1 1 1 0	~	1 1 1 1 1 1 0
US (unit separator)	0 0 1 1 1 1 1	?	0 1 1 1 1 1 1	—	1 0 1 1 1 1 1	DEL (delete)	1 1 1 1 1 1 1

Problems

2.1 Encode decimal numbers in BCD codes.

(a) 8 (b) 17

(c) 739 (d) 6938.24

2.2 Convert following BCD codes to decimal numbers.

(a) 10010111 (b) 000110000110

(c) 10000111.1001 (d) 00001001.10000000

2.3 How many bits are required to represent the decimal numbers from 0 to 99 using (a) BCD code? (b) straight binary code?

2.4 Encode binary numbers in Gray codes.

(a) 10110 (b) 10101010

2.5 Convert following Gray codes to binary numbers.

(a) 00101101 (b) 11111111

2.6 Encode the message in ASCII code, using binary representation.

(a) Y = 52 + 7 (b) "STOP"

2.7 What is the message of ASCII code stored in computer.

01011000

01001001

01001110

2.8 Construct a table showing the decimal, binary, BCD, Gray, and ASCII representations of decimal numbers from 0 to 15.