
Digital and Microprocessor Technology

PATRICK J. O'CONNOR

Professor
DeVry Institute of Technology
Chicago, Illinois

PRENTICE HALL, INC. Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging in Publication Data

O'Connor, Patrick.

Digital and microprocessor technology.

Includes index.

1. Electronic digital computers—Circuits. 2. Logic circuits. 3. Switching circuits. I. Title.
TK7888.4.O26 1983 621.3819/5835 82-11199
ISBN 0-13-212514-5

Editorial/production supervision
and interior design by *Shari Ingerman*
Manufacturing buyer: *Gordon Osbourne*
Cover design by *Edsal Enterprises*

© 1983 by Prentice-Hall, Inc., Englewood Cliffs, N.J. 07632

All rights reserved. No part of this book
may be reproduced in any form or
by any means without permission in writing
from the publisher.

Printed in the United States of America

10 9 8 7 6 5 4 3

ISBN 0-13-212514-5

PRENTICE-HALL INTERNATIONAL, INC., *London*
PRENTICE-HALL OF AUSTRALIA PTY. LIMITED, *Sydney*
EDITORIA PRENTICE-HALL DO BRAZIL, LTDA., *Rio de Janeiro*
PRENTICE-HALL CANADA INC., *Toronto*
PRENTICE-HALL OF INDIA PRIVATE LIMITED, *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*
PRENTICE-HALL OF SOUTHEAST ASIA PTE. LTD., *Singapore*
WHITEHALL BOOKS LIMITED, *Wellington, New Zealand*

Preface

This book is based loosely on a series of lectures given at the DeVry Institute of Technology to three classes:

T-3 Digital Systems I: 16-week beginning digital electronics course for technicians

T-4 Digital Systems II: 16-week intermediate digital systems course

T-5 Digital Systems III: 15-week final digital interface/microprocessor course

These courses are taken in the students' final three trimesters at DeVry.

Since the goal of the technician program is not to produce designers, but individuals competent enough in the basic principles to service existing equipment, the emphasis in these courses is threefold:

1. To present the vocabulary, so that the student can comprehend and use the "buzzwords" of the field with a real understanding of the underlying concepts which these words embody.
2. To explain the principles of operation of components used in existing circuits and systems.
3. To describe safe handling procedures and failure modes for the logic devices likely to be encountered in the field.

The reader should not need to begin at a reading or math level much beyond high school. The average technician student in these courses has a

high school diploma or G.E.D., but is unlikely to have any junior-college courses or a mathematical background much past basic algebra. Any mathematical manipulations presented here will be no more complicated than freshman high school algebra, although "logic" algebra may be different from "arithmetic" algebra in nature.

A familiarity with basic electricity and transistor operation will be helpful, but not indispensable, to understanding the ideas presented here. Although the students at DeVry begin digital with a background of a year's basic electronics courses, the same students could start the course from "ground zero" without any real problems.

Patrick O'Connor
Chicago, Illinois

Contents

Preface

ix

part I

ASYNCHRONOUS CIRCUITS

1

1 Switch Circuits and Logic

3

- 1.1 Logic States 1 and 0 3
- 1.2 Switch Circuit Representation for 1 and 0 as Inputs and Outputs 4
- 1.3 Switch Circuit Representation of AND Logic as a Series Circuit of Normally Open Switches 6
- 1.4 Switch Circuit Representation of OR Logic as a Parallel Circuit of Normally Open Switches 6
- 1.5 Switch Circuit Representation of NOT Logic as a Switch with Normally Closed Contacts 6
- 1.6 Bipolar Transistors Used as Switches 7
- 1.7 Field-Effect Transistors Used as Switches 8

2 Introduction to Gates

11

- 2.1 The Truth Table, Gate Symbol, and Boolean Expression 11
- 2.2 Truth Tables for AND and OR Switch Circuits 13

- 2.3 AND Gate, Showing Logic Multiplication 17
- 2.4 OR Gate, Showing Logic Addition 18
- 2.5 NOT Gate, Showing That $\bar{0} = 1$, $\bar{1} = 0$, and $\bar{\bar{A}} = A$ 20

3 Logic Functions Derived from Basic Gates

26

- 3.1 The NAND Gate 26
- 3.2 The NOR Gate 27
- 3.3 DeMorgan's Theorem: Inverted Inputs versus Outputs 28
- 3.4 NAND and NOR Implementation of Other Gates 30
- 3.5 The EXCLUSIVE OR Gate (Equal/Unequal Compare) 34
- 3.6 Enabling and Inhibiting Signals Going Through Gates 36
- 3.7 The EXCLUSIVE OR (XOR) Showing That It Is a Controlled Inverter instead of Being Enabled and Disabled 39

4 Combinational Logic

43

- 4.1 Equation Simplification Using Identities 43
- 4.2 The Commutative Law for AND and OR Logic 52
- 4.3 The Effect of Redundancy on AND and OR Logic 53
- 4.4 The Associative Law for AND and OR Logic 54
- 4.5 The Effect of Double Inversion on Logic 54
- 4.6 Gate Representation of Boolean Expressions 55
- 4.7 Simplification of Boolean Expressions, Showing Consequent Simplification of the Gate Diagram 56
- 4.8 Other Simplification Rules 61

5 Families of Logic

64

- 5.1 Diode Logic AND and OR gates 64
- 5.2 Diode-Transistor Logic 71
- 5.3 Resistor-Transistor Logic 74
- 5.4 Transistor-Transistor Logic 77
- 5.5 MOS Logic 93
- 5.6 Emitter-Coupled Logic 100
- 5.7 Integrated Injection Logic 101
- 5.8 Characteristics of Different Logic Families 106

6 Applications of Gates

123

- 6.1 AND gates (Decoders) 123
- 6.2 Encoders 132
- 6.3 Inverters 138
- 6.4 Multiplexers 144
- 6.5 Using Multiplexers to Replace Random Logic 150

7	More Applications of Logic Gates	155
7.1	Arithmetic Circuitry	155
7.2	Parity Detector/Generator	170
7.3	Comparator Circuit (Made with XOR, X-NOR, and AND Gates)	174
8	Binary-Based Number Codes	178
8.1	Binary, Octal, BCD, and Hex Numbering	178
8.2	BCD Adder/BCD Arithmetic	183
8.3	BCDIC and Hollerith Codes	185
8.4	ASCII Code	191
8.5	Other Alphameric Codes	192
8.6	Numeric Codes: Two's Complement (Signed Numbers)	193
8.7	Mantissa and Exponent (Floating-Point Numbers)	195
8.8	Octal Byte Code	197
8.9	Hexidecimal Byte Code	198
9	D/A and A/D as Parallel (Asynchronous) Circuits	200
9.1	Mechanical A/D (Shaft and Disk Encoder)	201
9.2	Gray Code to Eliminate Glitches in Binary Encoders	203
9.3	Gray-to-Binary Decoder Circuit (Using XOR)	204
9.4	Ladder-Type D/A Converters	206
9.5	Flash Conversion A/D Ladder	211
part II		
	SYNCHRONOUS CIRCUITS	217
10	Introduction to Flip-Flops	219
10.1	NAND Latch (R-S Flip-Flop)	220
10.2	Clocked NAND Latch (Clocked R-S Flip-Flop)	227
10.3	D-type Flip-Flop	230
10.4	J-K Flip-Flop without Master/Slave	232
10.5	J-K Master/Slave Flip-Flop—R-S Master/Slave	237
10.6	T-Type Flip-Flop	241

11	Applications of Flip-Flops (Shift Registers/Parallel Registers)	246
11.1	Shift Registers	246
11.2	Serial Full-Adders With Timing Diagrams	251
11.3	Parallel Registers	253
11.4	Shift Counters/Ring Counters	253
11.5	Parallel-to-Serial Conversion	256
11.6	Serial-to-Parallel Conversion	259
12	More Applications of Flip-Flops (Counter Circuits)	264
12.1	Odd-Even (Divide-by-2, Mod-2) Counter (Recap T-Type Flip-Flop)	264
12.2	Binary Ripple Up-Counters (Mod-4, Mod-8, Mod-16)	266
12.3	Binary Ripple Down-Counters	269
12.4	Counters With Moduli Other Than 2^n (Feedback and Forced-Reset)	270
12.5	Glitch States in Ripple Counters	274
12.6	Synchronous Counters to Eliminate Glitches	275
12.7	Excitation Tables and Their Use with State-Transition Diagrams and Timing Diagrams	277
12.8	The Mod-5, Mod-10, and Mod-6 Synchronous Counter as Applications of the Excitation Table	283
12.9	The Digital Clock Circuit as an Application of Synchronous Counters	287
12.10	Programmable Up/Down-Counters	291
12.11	Time-Base Generators for Clock Pulses	292
13	Counter-Type A/D Conversion	295
13.1	Review of Flash Conversion	295
13.2	Single-Slope Counter-Type A/D Conversion	296
13.3	Tracking A/D (Counter-Type A/D Conversion)	298
13.4	Resolution, Conversion Time, etc.	298
13.5	Application to Input of Data into Digital Systems	300
13.6	VCO (Analog-to-Frequency) Method of A/D Conversion	300
13.7	Pulse-Width Modulation	301
14	Memory Devices	305
14.1	Core Memory (Toroids)	305
14.2	Semiconductor Memory (RAM)	316
14.3	Read-Only Memory (ROM)	322
14.4	"Exotic" Types	326

part III

MICROPROCESSORS AND MICROCOMPUTERS

333

15 Bus-Oriented Systems

335

- 15.1 Multiplexed-Readout Display 335
- 15.2 Block Diagram of a Bus-Oriented Computer 339
- 15.3 Address Decoding for Memory and I/O Blocks 341
- 15.4 Open-Collector and Wire-OR Busing for Common Buses 351
- 15.5 Tri-State Logic in Bus-Oriented Systems 352
- 15.6 Practical Considerations in Bus-Oriented Design 354
- 15.7 Bus-Oriented Input/Output 357

16 Peripherals: I/O Devices (Input)

368

- 16.1 Binary Input (Bit Switches) 369
- 16.2 Numerical Input (Keyboards) 370
- 16.3 Alphameric Input 378
- 16.4 Magnetic (I/O) Storage Devices 382
- 16.5 Hollerith/Punched Paper Tape, etc. 391
- 16.6 OCR/Barcode Reader 396
- 16.7 A/D Converters Used with Sensors 397
- 16.8 DMA Controller 400
- 16.9 UARTs, Modems, Serial I/O for Telecommunications 401
- 16.10 Audio Interface (Microphone) 402

17 Peripherals: I/O Devices (Output)

404

- 17.1 Binary Readouts (LED Panel) 404
- 17.2 Segmented Displays 406
- 17.3 LCD Segmented Displays 410
- 17.4 Gas Discharge/Incandescent/Mechanical Displays 412
- 17.5 CRT Displays/Video Displays 414
- 17.6 Hardcopy (People-Readable Documents) 417
- 17.7 Hardcopy (Machine-Readable Documents) 429
- 17.8 Magnetic (I/O) Storage Devices 431
- 17.9 Electromechanical Output (Controllers) 434
- 17.10 Summary: Input and Output Devices 441

18 Introduction to Microprocessors

444

- 18.1 How Computers Decode Instructions 445
- 18.2 8-Bit Microprocessors 449
- 18.3 16-Bit Microprocessors 459

19	Machine Language Programming for the 8080 and Z-80	464
19.1	Why a Technician Should Know about Programming	465
19.2	The Data Transfer Group of Instructions	466
19.3	Arithmetic-Logic Instructions	478
19.4	Control Instructions	483
19.5	Miscellaneous Operations	489
19.6	Assemblers and Assembly Language	492
20	Software Structures/Flowcharting	495
20.1	Branching and Looping	495
20.2	Subroutines	502
20.3	Stack Operations with Subroutines	511
20.4	Interrupts	516
21	High-Level Language Programming	525
21.1	The BASIC Language	528
21.2	Example BASIC Program	537
21.3	Accessing Machine Language through BASIC	541
22	Telecommunications	544
22.1	Long-Distance Communication between Computers	545
22.2	DC and AC Signaling and Bandwidth	555
22.3	Transmission Media and Techniques	559
22.4	Pulse-Code Modulation	566
22.5	Imperfections	567
	Appendix	573
	One-Byte Opcodes Used for Both 8080 and Z-80	573
	Opcodes for Z-80 Only	575
	Index	577

Part I

ASYNCHRONOUS CIRCUITS

1

Switch Circuits and Logic

All digital logic circuits are switching circuits. Whatever else they may contain, each circuit begins with a circuit component that does the same job as a switch or pushbutton. At the beginning, we are going to look at circuits made out of pushbuttons, and later, we'll find out what types of switches are really used in logic circuits.

Let's begin with some basic ideas.

1.1 LOGIC STATES 1 AND 0

There are several schematic symbols for switches. The one you're probably most familiar with looks like that shown in Figure 1-1.

Since there's not much point to having a switch unless there's something to switch on and off, we included a light at one end of the circuit and a power supply ($V+$) at the other. The first piece of information that's new in this picture is the way we indicate whether the light is off or on. A light that's ON is marked with a 1 or with the word TRUE. When the light is OFF, we've marked it with a 0 or the word FALSE. The use of the words TRUE and FALSE makes sense if you remember that the thing at the end of the wire is called a light bulb. When the bulb is operating normally, it's lighted. That's the TRUE condition for an operating light bulb. (Otherwise, we'd have to call it a *dark* bulb!) When the bulb is not operating normally, we indicate that with the word FALSE. Remember that anything, when it's operating normally, is TRUE, and when it's not operating normally, it's FALSE.

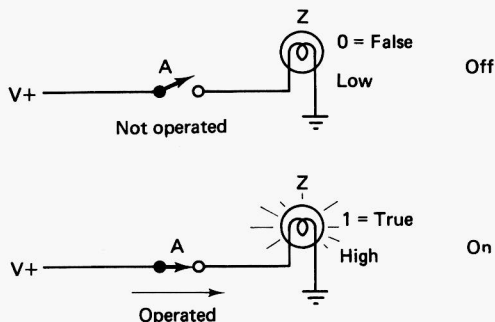


Figure 1-1 Output 1 and 0 conditions.

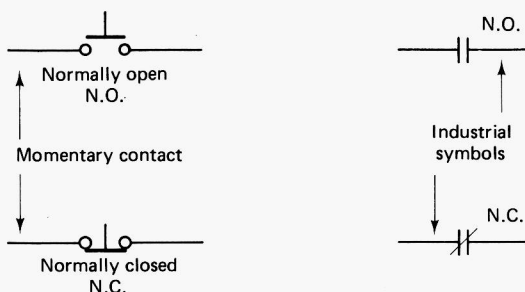


Figure 1-2 Momentary-contact switch symbols.

The numbers 1 and 0 are another way to indicate TRUE and FALSE. For the light that is lighted, a 1 indicates that there's something there. For the light that's not lighted, the 0 indicates that there's nothing there. On the diagram, there's a third way of indicating what's happening to the light bulb. The word "LOW" by the bulb that's OFF and "HIGH" by the bulb that's ON indicates what *voltage* is applied. A LOW voltage (in this case, nothing) is applied to the bulb when the switch is open. The HIGH voltage is the voltage of the power supply, connected to the light bulb when the switch is closed.

Now that we've seen the meaning of FALSE, LOW, and 0 and TRUE, HIGH, and 1, let's look at another type of switch (Figure 1-2).

1.2 SWITCH CIRCUIT REPRESENTATION FOR 1 AND 0 AS INPUTS AND OUTPUTS

The two switches in the picture are called **normally open (N.O.)** and **normally closed (N.C.)** types. Another name for switches of this type is *momentary contact*, or *pushbuttons*. The N.O. (normally open) pushbutton is open until you push the button, then it closes and completes the connection. The N.C. (normally closed) pushbutton is closed until you push the button and break the connection. We've also included the industrial standard sym-

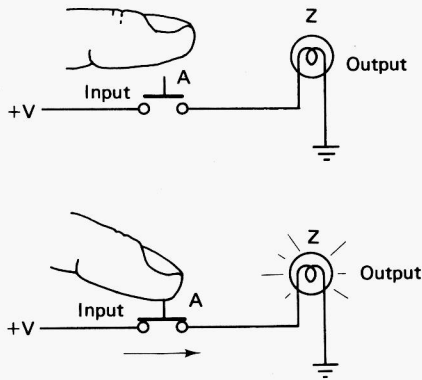


Figure 1-3 Input 1 and 0 conditions (normally open).

bol for these types of switch contacts, which isn't as easy to recognize, but is easier to draw. The circuit with a momentary-contact pushbutton instead of the switch in Figure 1-1 looks like the one shown in Figure 1-3.

Now, we have a circuit with an input and an output. The **input** operation, *A*, is whatever you're doing to the pushbutton, and the **output** operation, *Z*, is whatever the light bulb is doing. How is the pushbutton being operated? Remember the definition we had for 1 and 0; if the pushbutton is pushed, it's being operated, so it's a 1 (TRUE); if the pushbutton is not being pushed, it's a 0 (FALSE). For the input of the pushbutton circuit, a 1 is a pushed pushbutton, and a 0 is one that's been released.

Now, let's look at the output. We already know that a lighted light bulb is a 1, and a dark one is a 0. How does the input affect the output? If what you're doing at *A* is a 1 (pushing the button), the output at *Z* is also a 1 (the light is lit). If what you're doing at *A* is a 0 (not pushing the button), the output at *Z* (the state of the bulb) is also a 0 (dark). Notice that whatever you do at *A*, the **logic** state of *Z* (the 1 or 0) is the same. We could say that "*Z* is a 1 if *A* is a 1" and "*Z* is a 0 if *A* is a 0," or we could say the same thing with the expression

$$Z = A \quad (\text{Boolean expression})$$

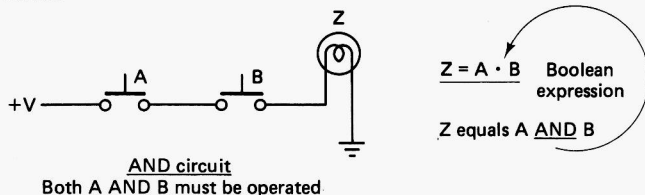
The **Boolean expression** is named after George Cayley Boole, who first used this kind of representation to show the relation between a cause and an effect. Actually, Boole did a great deal more, but we'll see that later.

The circuit in Figure 1-3 shows everything you can do with one (N.O.) switch. Let's see how many different ways we can use two switches:

1. We can attach two switches together in a **series** connection.
2. We can use a **parallel** connection.

Figure 1-4 shows both of these.

Series:



Parallel:

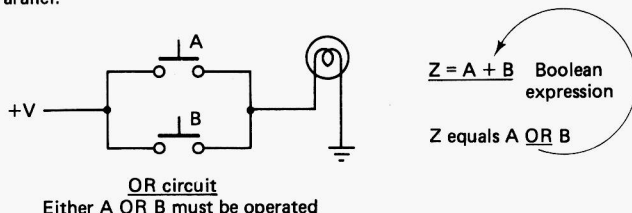


Figure 1-4 Switch logic (normally open contacts in series and parallel).

1.3 SWITCH CIRCUIT REPRESENTATION OF AND LOGIC AS A SERIES CIRCUIT OF NORMALLY OPEN SWITCHES

The *series* circuit in Figure 1-4 is identified as an **AND circuit**. Both switch A AND switch B must be operated before the light will go on. We say that a 1 must be input to A AND B to get a 1 at the output, Z. There is a Boolean expression for this relationship written to the right of the series diagram. We'll find out later just why the "dot" is used to represent the word AND in this expression.

1.4 SWITCH CIRCUIT REPRESENTATION OF OR LOGIC AS A PARALLEL CIRCUIT OF NORMALLY OPEN SWITCHES

The *parallel* circuit in Figure 1-4 is identified as an **OR circuit**. Either switch A OR switch B (or both) must be operated before the light will go on. We say that a 1 must be input to A OR B to get a 1 at the output, Z. There is a Boolean expression for this relationship written to the right of the parallel diagram. We'll find out later just why the "plus" is used to represent the word OR in the Boolean expression.

1.5 SWITCH CIRCUIT REPRESENTATION OF NOT LOGIC AS A SWITCH WITH NORMALLY CLOSED CONTACTS

There's one more thing that can be done using the pushbuttons. Up to now, we've only seen what normally open contacts do. Let's take a look at the