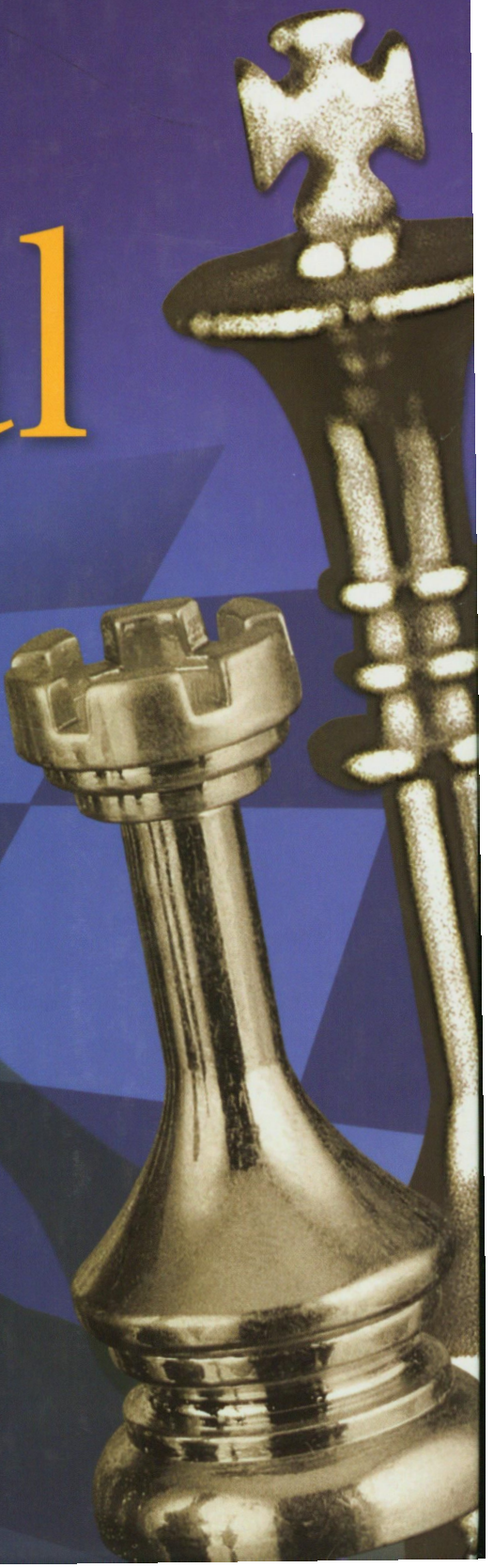
Fundamentals of

Digital Logic

with Verilog Design

Second Edition



Stephen Brown
Zvonko Vranesic

Petitioners
Ex. 1030, p. 1

FUNDAMENTALS OF DIGITAL LOGIC WITH VERILOG DESIGN

SECOND EDITION

Stephen Brown and Zvonko Vranesic
Department of Electrical and Computer Engineering
University of Toronto

McGraw-Hill Higher Education

Boston Burr Ridge, IL Dubuque, IA New York San Francisco St. Louis
Bangkok Bogotá Caracas Kuala Lumpur Lisbon London Madrid Mexico City
Milan Montreal New Delhi Santiago Seoul Singapore Sydney Taipei Toronto

Petitioners
Ex. 1030, p. 2



FUNDAMENTALS OF DIGITAL LOGIC WITH VERILOG DESIGN, SECOND EDITION

Published by McGraw-Hill, a business unit of The McGraw-Hill Companies, Inc., 1221 Avenue of the Americas, New York, NY 10020. Copyright © 2008 by The McGraw-Hill Companies, Inc. All rights reserved. No part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written consent of The McGraw-Hill Companies, Inc., including, but not limited to, in any network or other electronic storage or transmission, or broadcast for distance learning.

Some ancillaries, including electronic and print components, may not be available to customers outside the United States.

This book is printed on acid-free paper.

1 2 3 4 5 6 7 8 9 0 DOC/DOC 0 9 8 7

ISBN 978–0–07–338033–9
MHID 0–07–338033–4

Global Publisher: *Raghothaman Srinivasan*
Executive Editor: *Michael Hackett*
Director of Development: *Kristine M. Tibbetts*
Developmental Editor: *Darlene M. Schueller*
Executive Marketing Manager: *Michael Weitz*
Project Manager: *April R. Southwood*
Senior Production Supervisor: *Kara Kudronowicz*
Associate Media Producer: *Christina Nelson*
Lead Media Project Manager: *Judi David*
Associate Design Coordinator: *Brenda A. Rolwes*
Cover Designer: *Studio Montage, St. Louis, Missouri*
(USE) Cover Image: *Royalty-Free/CORBIS*
Senior Photo Research Coordinator: *Lori Hancock*
Compositor: *Techsetters, Inc.*
Typeface: *10/12 Times Roman*
Printer: *R. R. Donnelley Crawfordsville, IN*

Library of Congress Cataloging-in-Publication Data

Brown, Stephen D.
Fundamentals of digital logic with Verilog design / Stephen Brown, Zvonko Vranesic. — 2nd ed.
p. cm.
ISBN 978–0–07–338033–9 — ISBN 0–07–338033–4 (hard copy : alk. paper)
1. Logic circuits—Design and construction—Data processing. 2. Verilog (Computer hardware description language). 3. Computer-aided design. I. Vranesic, Zvonko G. II. Title.

TK7868.L6B76 2008
621.39′2–dc22 2007008622

www.mhhe.com

the inverter is not turned off, because its gate-to-source voltage is −1.5 V, which is larger than V_T. Static current flows from V_{DD} to *Gnd* through the inverter. A similar situation occurs when the AND gate produces the low output $V_f = 1.5$ V. Here the PMOS transistor in the inverter is turned on, but the NMOS transistor is not turned off. The AND gate implementation in Figure 3.52 is not used in practice.

3.8.8 FAN-IN AND FAN-OUT IN LOGIC GATES

The *fan-in* of a logic gate is defined as the number of inputs to the gate. Depending on how a logic gate is constructed, it may be impractical to increase the number of inputs beyond a small number. For example, consider the NMOS NAND gate in Figure 3.53, which has k inputs. We wish to consider the effect of k on the propagation delay t_p through the gate. Assume that all k NMOS transistors have the same width W and length L. Because the transistors are connected in series, we can consider them to be equivalent to one long transistor with length $k \times L$ and width W. Using equation 3.4 (which can be applied to both

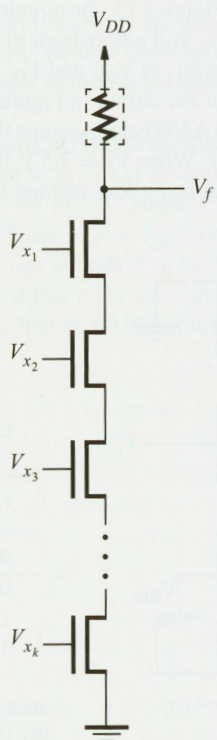


Figure 3.53 High fan-in NMOS NAND gate.

Petitioners
Ex. 1030, p. 132

CMOS and NMOS gates), the propagation delay is given by

$$t_p \cong \frac{1.7\,C}{k_n' \frac{W}{L} V_{DD}} \times k$$

Here C is the equivalent capacitance at the output of the gate, including the parasitic capacitance contributed by each of the k transistors. The performance of the gate can be improved somewhat by increasing W for each NMOS transistor. But this change further increases C and comes at the expense of chip area. Another drawback of the circuit is that each NMOS transistor has the effect of increasing V_{OL}, hence reducing the noise margin. It is practical to build NAND gates in this manner only if the fan-in is small.

As another example of fan-in, Figure 3.54 shows an NMOS k-input NOR gate. In this case the k NMOS transistors connected in parallel can be viewed as one large transistor with width $k \times W$ and length L. According to equation 3.4, the propagation delay should be decreased by the factor k. However, the parallel-connected transistors increase the load capacitance C at the gate's output and, more importantly, it is extremely unlikely that all of the transistors would be turned on when V_f is changing from a high to low level. It is thus practical to build high fan-in NOR gates in NMOS technology. We should note, however, that in an NMOS gate the low-to-high propagation delay may be slower than the high-to-low delay as a result of the current-limiting effect of the pull-up device (see Examples 3.13 and 3.14).

High fan-in CMOS logic gates always require either k NMOS or k PMOS transistors in series and are therefore never practical. In CMOS the only reasonable way to construct a high fan-in gate is to use two or more lower fan-in gates. For example, one way to realize a six-input AND gate is as 2 three-input AND gates that connect to a two-input AND gate. It is possible to build a six-input CMOS AND gate using fewer transistors than needed with this approach, but we leave this as an exercise for the reader (see problem 3.4).

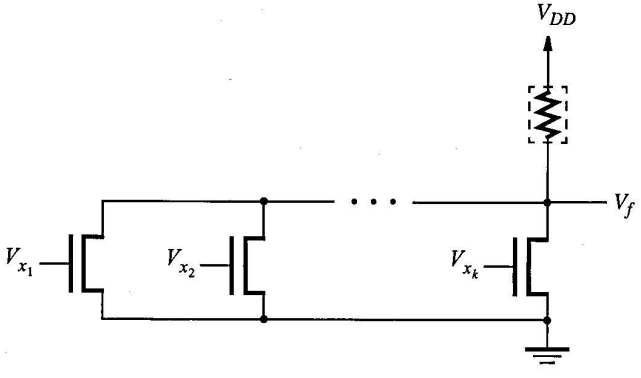
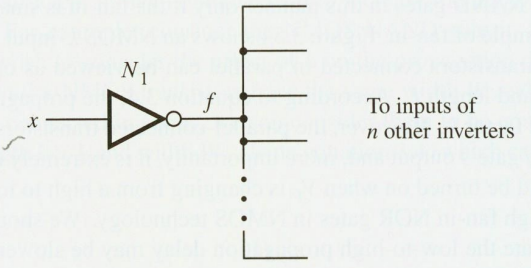


Figure 3.54 High fan-in NMOS NOR gate.

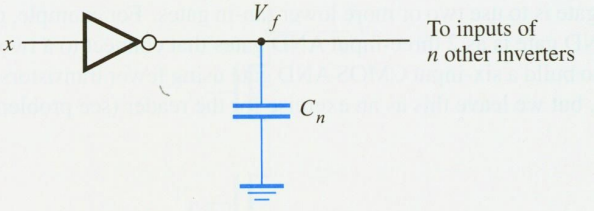
Petitioners
Ex. 1030, p. 133

Fan-out

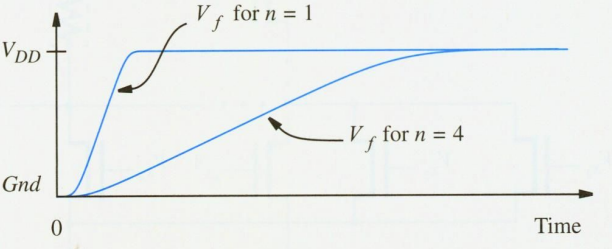
Figure 3.48 illustrated timing delays for one NOT gate driving another. In real circuits each logic gate may be required to drive several others. The number of other gates that a specific gate drives is called its *fan-out*. An example of fan-out is depicted in Figure 3.55*a*, which shows an inverter N_1 that drives the inputs of n other inverters. Each of the other inverters contributes to the total capacitive loading on node f. In part (*b*) of the figure, the n inverters are represented by one large capacitor C_n. For simplicity, assume that each inverter contributes a capacitance C and that $C_n = n \times C$. Equation 3.4 shows that the propagation delay increases in direct proportion to n.



(a) Inverter that drives n other inverters



(b) Equivalent circuit for timing purposes



(c) Propagation times for different values of n

Figure 3.55 The effect of fan-out on propagation delay.

Petitioners
Ex. 1030, p. 134

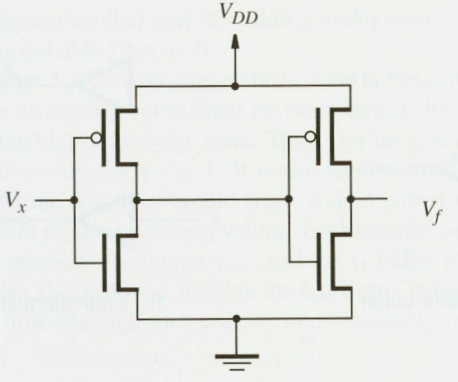
Figure 3.55*c* illustrates how n affects the propagation delay. It assumes that a change from logic value 1 to 0 on signal x occurs at time 0. One curve represents the case where $n = 1$, and the other curve corresponds to $n = 4$. Using the parameters from Example 3.7, when $n = 1$, we have $t_p = 0.1$ ns. Then for $n = 4$, $t_p \approx 0.4$ ns. It is possible to reduce t_p by increasing the W/L ratios of the transistors in N_1.

Buffers

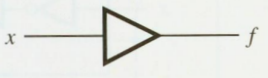
In circuits in which a logic gate has to drive a large capacitive load, buffers are often used to improve performance. A *buffer* is a logic gate with one input, x, and one output, f, which produces $f = x$. The simplest implementation of a buffer uses two inverters, as shown in Figure 3.56*a*. Buffers can be created with different amounts of *drive capability*, depending on the sizes of the transistors (see Figure 3.49). In general, because they are used for driving higher-than-normal capacitive loads, buffers have transistors that are larger than those in typical logic gates. The graphical symbol for a noninverting buffer is given in Figure 3.56*b*.

Another type of buffer is the *inverting buffer*. It produces the same output as an inverter, $f = \bar{x}$, but is built with relatively large transistors. The graphical symbol for the inverting buffer is the same as for the NOT gate; an inverting buffer is just a NOT gate that is capable of driving large capacitive loads. In Figure 3.55 for large values of n an inverting buffer could be used for the inverter labeled N_1.

In addition to their use for improving the speed performance of circuits, buffers are also used when high current flow is needed to drive external devices. Buffers can handle



(a) Implementation of a buffer



(b) Graphical symbol

Figure 3.56 A noninverting buffer.

Petitioners
Ex. 1030, p. 135

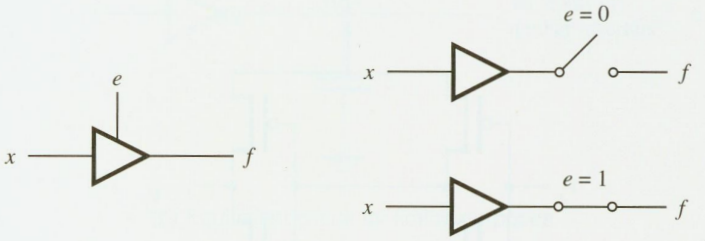
relatively large amounts of current flow because they are built with large transistors. A common example of this use of buffers is to control a light-emitting diode (LED). We describe an example of this application of buffers in section 7.14.3.

In general, fan-out, capacitive loading, and current flow are important issues that the designer of a digital circuit must consider carefully. In practice, the decision as to whether or not buffers are needed in a circuit is made with the aid of CAD tools.

Tri-state Buffers

In section 3.6.2 we mentioned that a type of buffer called a tri-state buffer is included in some standard chips and in PLDs. A tri-state buffer has one input, x, one output, f, and a control input, called *enable*, e. The graphical symbol for a tri-state buffer is given in Figure 3.57*a*. The enable input is used to determine whether or not the tri-state buffer produces an output signal, as illustrated in Figure 3.57*b*. When $e = 0$, the buffer is completely disconnected from the output f. When $e = 1$, the buffer drives the value of x onto f, causing $f = x$. This behavior is described in truth-table form in part (*c*) of the figure. For the two rows of the table where $e = 0$, the output is denoted by the logic value Z, which is called the *high-impedance state*. The name *tri-state* derives from the fact that there are two normal states for a logic signal, 0 and 1, and Z represents a third state that produces no output signal. Figure 3.57*d* shows a possible implementation of the tri-state buffer.

Figure 3.58 shows several types of tri-state buffers. The buffer in part (*b*) has the same behavior as the buffer in part (*a*), except that when $e = 1$, it produces $f = \bar{x}$. Part (*c*) of the figure gives a tri-state buffer for which the enable signal has the opposite behavior; that is, when $e = 0, f = x$, and when $e = 1, f = Z$. The term often used to describe this type

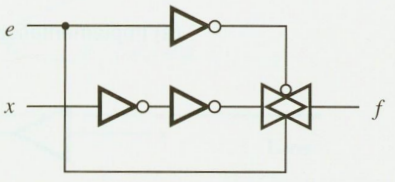


(a) A tri-state buffer

(b) Equivalent circuit

e	x	f
0	0	Z
0	1	Z
1	0	0
1	1	1

(c) Truth table



(d) Implementation

Figure 3.57 Tri-state buffer.

Petitioners
Ex. 1030, p. 136

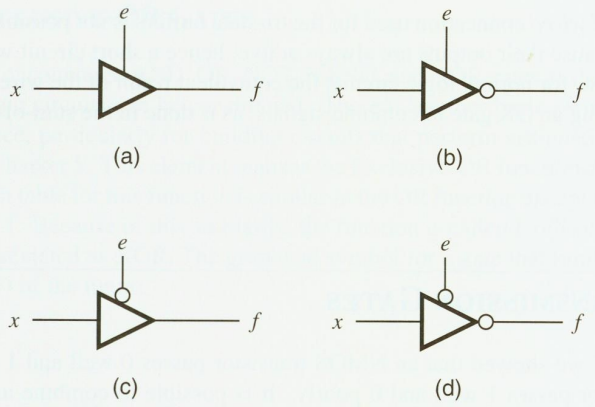


Figure 3.58 Four types of tri-state buffers.

of behavior is to say that the enable is *active low*. The buffer in Figure 3.58*d* also features an active-low enable, and it produces $f = \bar{x}$ when $e = 0$.

As a small example of how tri-state buffers can be used, consider the circuit in Figure 3.59. In this circuit the output f is equal to either x_1 or x_2, depending on the value of s. When $s = 0, f = x_1$, and when $s = 1, f = x_2$. Circuits of this kind, which choose one of the inputs and reproduce the signal on this input at the output terminal, are called *multiplexer* circuits. A circuit that implements the multiplexer using AND and OR gates is shown in Figure 2.26. We will present another way of building multiplexer circuits in section 3.9.2 and will discuss them in detail in Chapter 6.

In the circuit of Figure 3.59, the outputs of the tri-state buffers are wired together. This connection is possible because the control input s is connected so that one of the two buffers is guaranteed to be in the high-impedance state. The x_1 buffer is active only when $s = 0$, and the x_2 buffer is active only when $s = 1$. It would be disastrous to allow both buffers to be active at the same time. Doing so would create a short circuit between V_{DD} and *Gnd* as soon as the two buffers produce different values. For example, assume that $x_1 = 1$ and $x_2 = 0$. The x_1 buffer produces the output V_{DD}, and the x_2 buffer produces *Gnd*. A short circuit is formed between V_{DD} and *Gnd*, through the transistors in the tri-state buffers. The amount of current that flows through such a short circuit is usually sufficient to destroy the circuit.

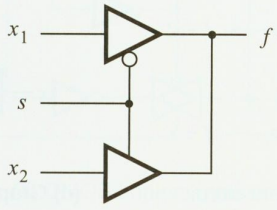


Figure 3.59 An application of tri-state buffers.

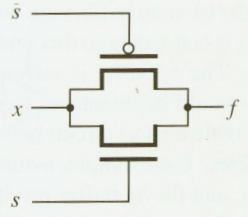
Petitioners
Ex. 1030, p. 137

The kind of *wired* connection used for the tri-state buffers is not possible with ordinary logic gates, because their outputs are always active; hence a short circuit would occur. As we already know, for normal logic circuits the equivalent result of the wired connection is achieved by using an OR gate to combine signals, as is done in the sum-of-products form.

3.9 TRANSMISSION GATES

In section 3.8.7 we showed that an NMOS transistor passes 0 well and 1 poorly, while a PMOS transistor passes 1 well and 0 poorly. It is possible to combine an NMOS and a PMOS transistor into a single switch that is capable of driving its output terminal either to a low or high voltage equally well. Figure 3.60*a* gives the circuit for a *transmission gate*. As indicated in parts (*b*) and (*c*) of the figure, it acts as a switch that connects x to f. Switch control is provided by the *select* input s and its complement $\bar{s}$. The switch is turned on by setting $V_s = 5$ V and $V_{\bar{s}} = 0$. When V_x is 0, the NMOS transistor will be turned on (because $V_{GS} = V_s - V_x = 5$ V) and V_f will be 0. On the other hand, when V_x is 5 V, then the PMOS transistor will be on ($V_{GS} = V_{\bar{s}} - V_x = -5$ V) and V_f will be 5 V. A graphical symbol for the transmission gate is given in Figure 3.60*d*.

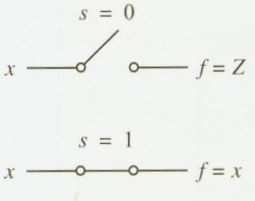
Transmission gates can be used in a variety of applications. We will show next how they lead to efficient implementations of *Exclusive OR (XOR)* logic gates and multiplexer circuits.



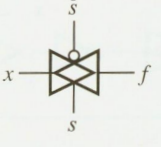
(a) Circuit

s	f
0	Z
1	x

(b) Truth table



(c) Equivalent circuit



(d) Graphical symbol

Figure 3.60 A transmission gate.

Petitioners
Ex. 1030, p. 138