THE ART OF ELECTRONICS

SECOND EDITION

THE ART OF ELECTRONICS

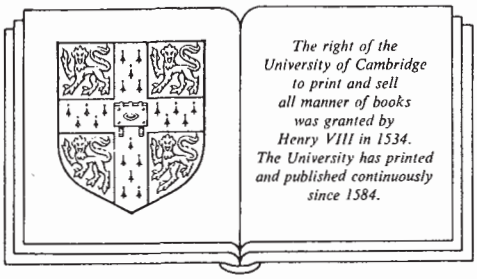
Second Edition

THE ART OF ELECTRONICS

Second Edition

Paul Horowitz HARVARD UNIVERSITY

Winfield Hill ROWLAND INSTITUTE FOR SCIENCE, CAMBRIDGE, MASSACHUSETTS



CAMBRIDGE UNIVERSITY PRESS

Cambridge

New York Port Chester Melbourne Sydney

Published by the Press Syndicate of the University of Cambridge
The Pitt Building, Trumpington Street, Cambridge CB2 1RP
32 East 57th Street, New York, NY 10022, USA
10 Stamford Road, Oakleigh, Melbourne 3166, Australia

First published 1980
Second edition 1989

Printed in the United States of America

Library of Congress Cataloging-in-Publication Data

Horowitz, Paul, 1942–

The art of electronics / Paul Horowitz, Winfield Hill. – 2nd ed.

p. cm.

Bibliography: p.

Includes index.

ISBN 0-521-37095-7

1. Electronics. 2. Electronic circuit design. I. Hill, Winfield. II. Title.
TK7815.H67 1989
621.381 – dc19 89-468
CIP

British Library Cataloguing in Publication Data

Horowitz, Paul, 1942–

The art of electronics. – 2nd ed.

1. Electronic equipment
I. Title II. Hill, Winfield
621.381

ISBN 0-521-37095-7 hard covers

TO CAROL, JACOB, MISHA, AND GINGER

CONTENTS

TABLES

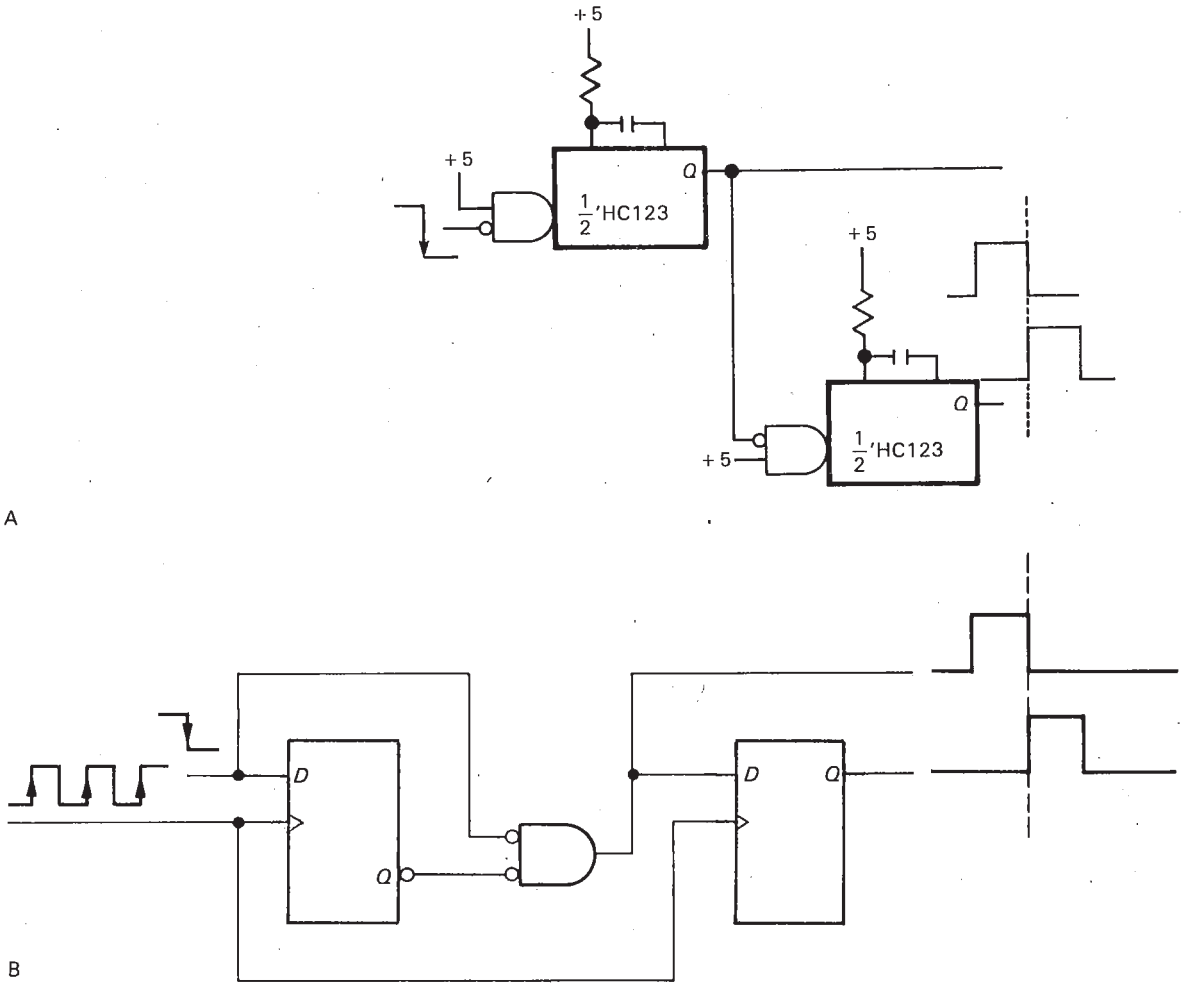


Figure 8.67. A digital delay can replace one-shot delays.

system clocks is common in synchronous circuits.

8.23 Timing with counters

As we have just emphasized, there are many good reasons for avoiding the use of monostables in logic design. Figure 8.68 shows another case where flip-flops and counters (cascaded toggling flip-flops) can be used in place of a monostable to generate a long output pulse. The '4060 is a 14-stage CMOS binary counter (14 cascaded flip-flops). A rising edge at the input brings Q HIGH, enabling the counter. After 2^{n-1} clock pulses, Q_n goes HIGH, clearing the flip-flop and the counter. This circuit generates an accurate long pulse whose length may be varied by factors of 2. The '4060 also includes internal oscillator circuitry that can substitute for the external clock reference. Our experience is that the internal oscillator has poor frequency tolerance and (in some HC versions) may malfunction.

You can get complete integrated circuits to implement timing with counters. The ICM7240/50/60 (Intersil, Maxim) have 8-bit or 2-digit internal counters and the necessary logic to make delays equal to an integral number of counts (1–255 or 1–99 counts); you can set the number either with "hardwired" connections or with external thumbwheel switches. The ICM7242 is similar, but with prewired divide-by-128 counter. Exar makes a close cousin, called the XR2243, which has a fixed divide-by-1024 counter.

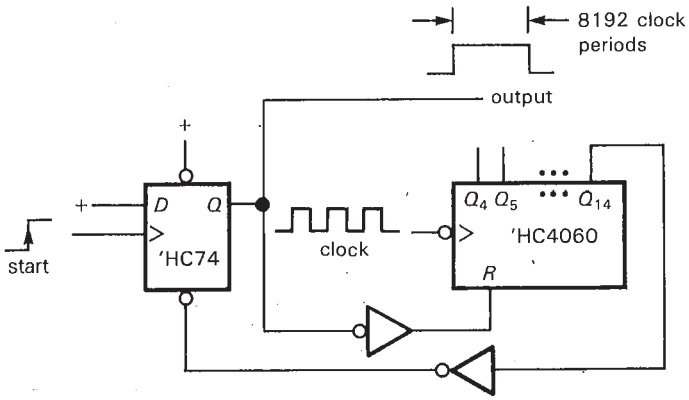


Figure 8.68. Digital generation of long pulses.

SEQUENTIAL FUNCTIONS AVAILABLE AS ICs

As with the combinational functions we described earlier, it is possible to integrate various combinations of flip-flops and gates onto a single chip. In the following sections we will present a survey of the most useful types, listed according to function.

As with pure combinational logic, *programmable* logic (PALs and GALs in particular) provides an attractive alternative to the use of prewired sequential functions. We'll talk about them, also, after looking at the standard functions.

8.24 Latches and registers

Latches and registers are used to "hold" a set of bits, even if the inputs change. A set of D flip-flops constitutes a register, but it has more inputs and outputs than necessary. Since you don't need separate clocks, or SET and CLEAR inputs, those lines can be tied together, requiring fewer pins and therefore allowing 8 flip-flops to fit in a 20-pin package. The popular '574 is an octal D register with positive clock edge and three-state outputs; the '273 is similar, but has a reset instead of three-state outputs. Figure 8.69 shows a quad D register with both true and complemented outputs.

The term "latch" is usually reserved for a special kind of register: one in which the outputs follow the inputs when enabled, and hold the last value when disabled. Since the term "latch" has become ambiguous with use, the terms "transparent latch" and "type D register" are often used to distinguish these closely related devices. As an example, the '573 is the octal transparent-latch equivalent of the '574 D register.

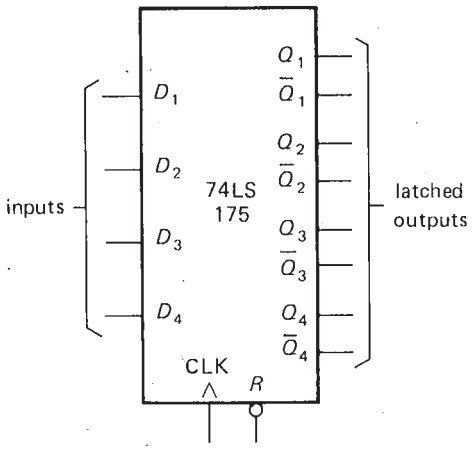


Figure 8.69. '175 4-bit D register.

Some variations on the latch/register are as follows: (a) random-access memories (RAMs), which let you write to, and read from, a (usually large) set of registers, but only one (or at most a few) at a time; RAMs come in sizes from a handful of bytes up to 1M bytes or more and are used primarily for memory in microprocessor systems (see Chapters 10 and 11); (b) addressable latches, a multibit latch that lets you update individual bits while keeping the others unchanged; (c) a latch or register built into a larger chip, for example a

digital-to-analog converter; such a device only needs the input applied momentarily (with appropriate clocking edge), since an internal register can hold the data.

Table 8.9 at the end of the chapter lists most of the useful registers and latches. Note features such as input enable, reset, three-state outputs, and "broadside" pinout (inputs on one side of the chip, outputs on the other); the latter is very convenient when you are laying out a printed-circuit board.

8.25 Counters

As we mentioned earlier, it is possible to make a "counter" by connecting flip-flops together. There is available an amazing variety of such devices as single chips. Here are some of the features to look for:

Size

You can get BCD (divide-by-10) and binary (or *hexadecimal*, divide-by-16) counters in the popular 4-bit category. There are larger counters, up to 24 bits (not all available as outputs), and there are *modulo-n* counters that divide by an integer n, specified as an input. You can always cascade counters (including synchronous types) to get more stages.

Clocking

An important distinction is whether the counter is a "ripple" counter or a "synchronous" counter. The latter clocks all flip-flops simultaneously, whereas in a ripple counter each stage is clocked by the output of the previous stage. Ripple counters generate transient states, since the earlier stages toggle slightly before the later ones. For instance, a ripple counter going from a count of 7 (0111) to 8 (1000) goes through the states 6, 4, and 0 along the way. This doesn't cause trouble in well-designed circuits, but it would in a circuit that used gates to look for a particular state (this is a good place to use something like a D flip-flop, so that the state is examined only at the clock edge). Ripple counters are slower than synchronous counters, because of the accumulated propagation delays. Ripple counters clock on negative-going edges for easy expandability (by connecting the Q output of one counter directly to the clock input of the next); synchronous counters clock on the positive edge.

We favor the '160–'163 family of 4-bit synchronous counters for most applications that don't require some special feature. The '590 and '592 are good 8-bit synchronous counters. Figure 8.70 shows the '390 dual BCD ripple counter.

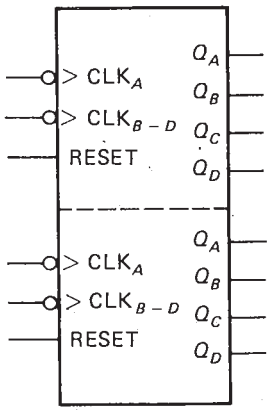


Figure 8.70. '390 dual BCD ripple counter.

Up/down

Some counters can count in either direction, under control of some inputs. The two possibilities are (a) an U/D′ input that sets the direction of count and (b) a pair of clocking inputs, one for UP, one for DOWN. Examples are the '191 and '193, respectively. The '569 and '579 are useful 8-bit up/down counters.

Load and clear

Most counters have data inputs so that they can be preset to a given count. This

is handy if you want to make a modulo-n counter, for example. The load function can be either synchronous or asynchronous: the '160–'163 have synchronous load, which means that data on the input lines are transferred to the counter coincident with the next clock edge, if the LOAD′ line is also asserted LOW; the '190–'193 are asynchronous, or *jam*-load, which means that input data are transferred to the counter when LOAD′ is asserted, independent of the clock. The term "parallel load" is sometimes used, since all bits are loaded at the same time.

The CLEAR (or RESET) function is a form of presetting. The majority of counters have a jam-type CLEAR function, though some have synchronous CLEAR; for example, the '160/161 are jam CLEAR, while the '162/163 are synchronous CLEAR.

Other counter features

Some counters feature latches on the output lines; these are always of the transparent type, so the counter can be used as if no latch were present. (Keep in mind that any counter with parallel-load inputs can function as latch, but you can't count at the same time as data are held, as you can with a counter/latch chip.) The combination of counter plus latch is sometimes very convenient, e.g., if you want to display or output the previous count while beginning a new counting cycle. In a frequency counter this would allow a stable display, with updating after each counting cycle, rather than a display that repeatedly gets reset to zero and then counts up.

There are counters with three-state outputs. These are great for applications where the digits (or 4-bit groups) are multiplexed onto a bus for display or transfer to some other device. An example is the '779, an 8-bit synchronous binary counter whose three-state outputs also serve as parallel inputs; by sharing input/output lines, the counter fits in a 16-pin package. The '593 is similar, but in a 20-pin package.

If you want a counter to use with a display, there are several that combine counter, latch, 7-segment decoder, and driver on one chip. An example is the 74C925–74C928 series of 4-digit counters. Another amusing chip is the TIL306/7, a counter *with display* on one chip: You just look at the IC, which lights up with a digit telling the count! Figure 8.71 shows a nice LSI (large-scale-integration) counter circuit that doesn't require a lot of support circuits.

Table 8.10 at the end of the chapter lists most of the counter chips that you might want to use. Many of them are only available in one family (e.g., LS or F), so be sure to check the data books before you design with them.

8.26 Shift registers

If you connect a series of flip-flops so that each Q output drives the next D input, and all clock inputs are driven simultaneously, you get what's called a "shift register." At each clock pulse the pattern of 0's and 1's in the register shifts to the right, with the data at the first D input entering from the left. As with flip-flops, the data present at the serial input just prior to the clock pulse are entered, and there is the usual propagation delay to the outputs. Thus they may be cascaded without fear of a logic race. Shift registers are very useful for conversion of parallel data (n bits present simultaneously, on n separate lines) to serial data (one bit after another, on a single data line), and vice versa. They're also handy as memories, particularly if the data are always read and written in order. As with counters and latches, shift registers come in a pleasant variety of prefab styles. The important things to look for are the following:

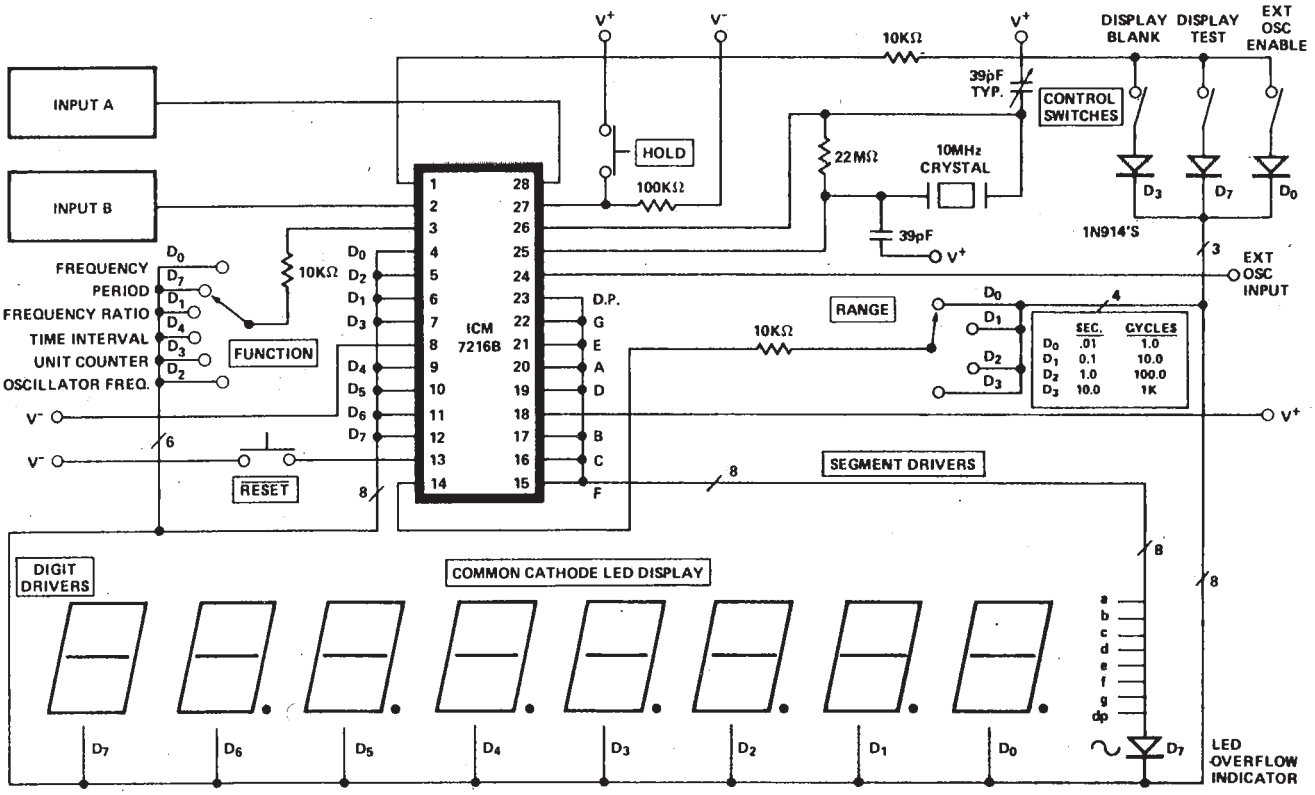


Figure 8.71. Intersil 7216 8-digit 10MHz universal counter on a chip. (Courtesy of Intersil, Inc.)

Size

The 4-bit and 8-bit registers are standard, with some larger sizes available (up to 64 bits or more). There are even variable-length registers (e.g., the 4557: 1 to 64 stages, set by a 6-bit input).

Organization

Shift registers are usually 1 bit wide, but there are also dual-, quad-, and hex-width registers. Most shift registers only shift right, but there are bidirectional registers like the '194 and '323 that have a "direction" input (Fig. 8.72). Watch out for trickery like the "bidirectional" '95, which can shift left only by tying each output bit to the previous input, then doing a parallel load.

Inputs and outputs

Small shift registers can provide parallel inputs or outputs, and usually do; an example is the '395, a 4-bit parallel-in, parallel-out (PI/PO) shift register with three-state outputs. Larger registers may only provide *serial* input or output, i.e., only the input to the first flip-flop or the output from the last is accessible. In some cases a few selected intermediate taps are provided. One way to provide both parallel input and output in a small package is to share input and output (three-state) on the same pins, e.g., the '299, an 8-bit bidirectional PI/PO register in a 20-pin package. Some shift registers include a latch at the input or output, so shifting can go on while data are being loaded or unloaded.

As with counters, parallel LOAD and

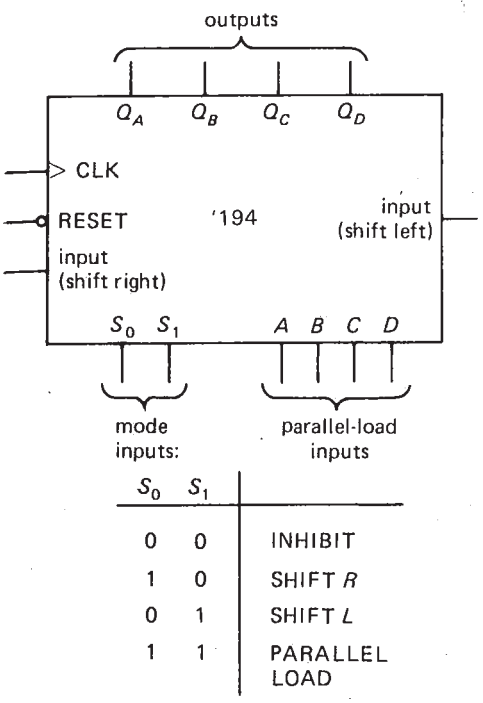


Figure 8.72. '194 4-bit bidirectional shift register.

CLEAR can be either synchronous or jam-load; for example, the '323 is the same as the '299, but with synchronous clear.

Table 8.11 at the end of the chapter lists the shift registers you're likely to use. As always, not all types are available in all logic families; be sure to check the data books.

RAMs as shift registers

A random-access memory can always be used as a shift register (but not vice versa) by using an external counter to generate successive addresses. Figure 8.73 shows the idea. An 8-bit synchronous up/down counter generates successive addresses for a 256-word×4-bit CMOS RAM. The combination behaves like a quad 256-bit shift register, with left/right direction of shift selected by the counter's UP/DOWN′ control line. The other inputs of the counter are shown enabled for counting. By choosing a fast counter and memory, we were able to achieve a maximum clocking rate of 30MHz (see timing diagram), which is the same as that of an integrated (but much smaller) HC-type shift register. This technique can be used to produce very large shift registers, if desired.

EXERCISE 8.28

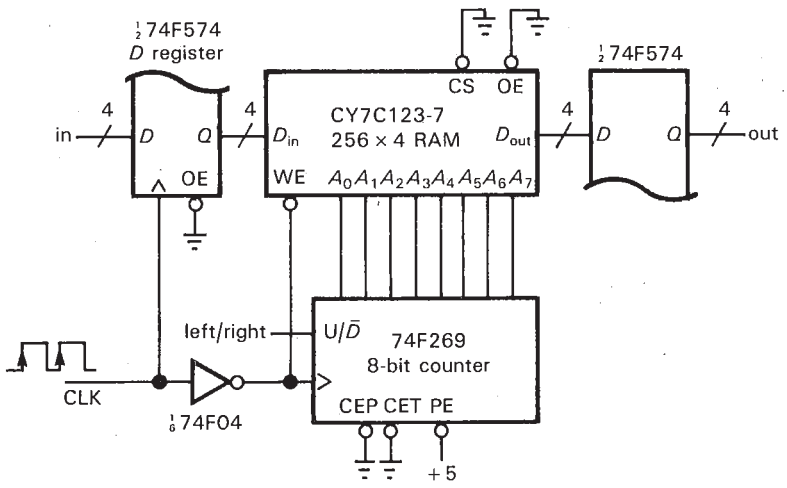
In the circuit of Figure 8.73, input data seem to go into the same location that output data are read from. Nevertheless, the circuit behaves identically to a classic 256-word shift register. Explain why.

8.27 Sequential PALs

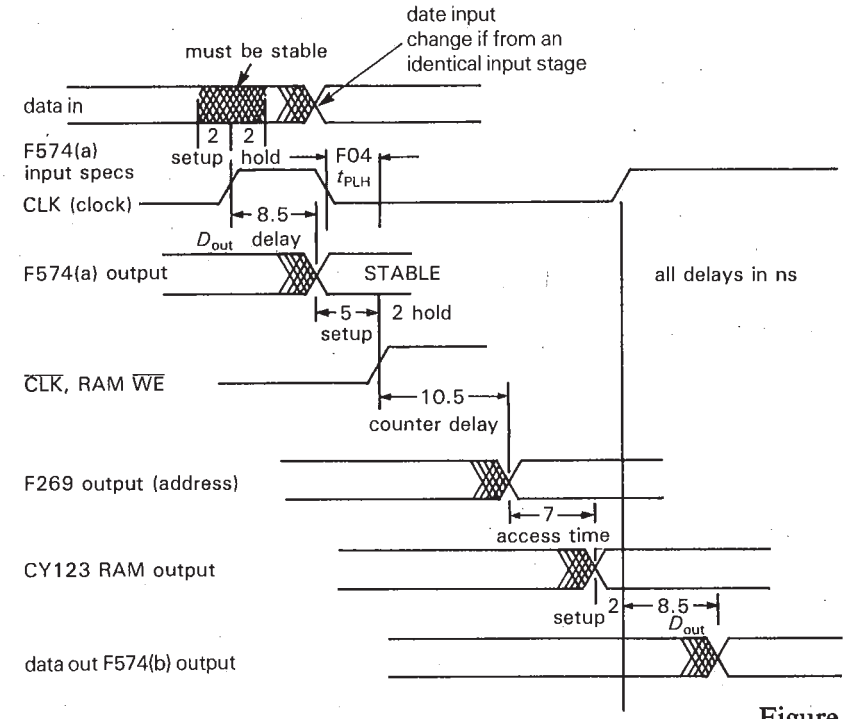
The combinational (gates-only) PALs we talked about in Section 8.15 belong to a larger family that includes devices with various numbers of on-chip D-type registers (called "registered PALs"). Typical of these PALs is the 16R8, shown in Figure 8.74. The programmable-AND/fixed-OR array typical of combinational PALs generates the input levels for 8 synchronously clocked D-type registers with three-state outputs; the register outputs (and their inverts) are available, along with the standard input pins, as inputs to the logic array. If you look back at Figure 8.57, you'll see that a registered PAL is a general-purpose sequential circuit element; within limits set by the number of registers and gates available, you can construct just about anything you want. For instance, you could make a shift register or counter, or some of both! In practice, you're more likely to make some custom piece of logic that is part of a larger circuit, for which the alternative is "discrete" logic built with gates and flip-flops. Let's look at some examples.

Hand-generated fuse maps

Simple designs can be implemented in PALs by figuring out the logic, then burning the appropriate pattern into the fuse array with a "PAL programmer." As an



A



B

F04 t_{PLH} delay	3.7ns	
F374 D_{out} delay	8.5ns	} 13.5 − 3.7 = 9.8ns HI
CYC123 setup time	5	
F269 counter out delay	10.5	
CYC123 access time	7	} 19.5 + 3.7 = 23.2ns LO
F374 D_{IN} setup time	2	
shift clock period(min)	33 ns	

C

Figure 8.73. A. Large shift register made from RAM plus counter; the slash indicates multiple lines, in this case a 4-bit-wide data path.
B. Timing diagram to calculate maximum clocking rate, assuming worst-case timing specifications.
C. Calculation showing worst-case sequence of delays in a single clock cycle.

example, let's imagine we want a 4-input multiplexer with latched outputs. We can write the logic equation for the multiplexer portion (i.e., the D-input to the flip-flop) as

$$Q.d = I_0 * S_0' * S_1' + I_1 * S_0 * S_1' + I_2 * S_0' * S_1 + I_3 * S_0 * S_1$$

where the inputs S_0 and S_1 address the selected input I_0–I_3 and "*" and "+" represent AND and OR. A registered PAL makes it easy to latch the result. Note that we have used the OR of 3-input ANDs, rather than first decoding the select address in 2-input ANDs, because we