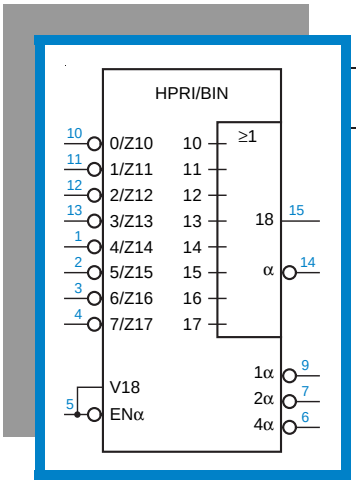


APPENDIX **A**

IEEE STANDARD SYMBOLS

Together with the American National Standards Institute (ANSI), the Institute of Electrical and Electronic Engineers (IEEE) has developed a standard set of logic symbols. The most recent revision of the standard is ANSI/IEEE Std 91-1984, *IEEE Standard Graphic Symbols for Logic Functions*. It is compatible with standard 617 of the International Electrotechnical Commission (IEC), and must be used in all logic diagrams drawn for the U.S. Department of Defense.

ANSI
IEEE

A.1 GENERAL DEFINITIONS

The IEEE standard supports the notion of bubble-to-bubble logic design with the following definitions:

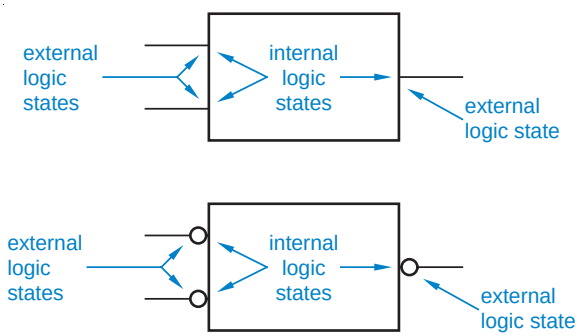
- An *internal logic state* is a logic state assumed to exist inside a symbol outline at an input or an output. *internal logic state*
- An *external logic state* is a logic state assumed to exist outside a symbol outline either (1) on an input line prior to any external qualifying symbol at that input, or (2) on an output line beyond any external qualifying symbol at that output. *external logic state*

qualifying symbol

A *qualifying symbol* is graphics or text added to the basic outline of a device's logic symbol to describe the physical or logical characteristics of the device. The "external qualifying symbol" mentioned above is typically an inversion bubble, which denotes a "negated" input or output, for which the external 0-state corresponds to the internal 1-state. This concept is illustrated in Figure A–1. When the standard says that a signal is in its *internal 1-state*, we would say that the signal is asserted. Likewise, when the standard says that a signal is in its *internal 0-state*, we would say that the signal is negated.

internal 1-state
internal 0-state

Figure A–1
Internal and external logic states.



distinctive-shape symbols
rectangular-shape symbols

The IEEE standard provides two different types of symbols for logic gates. One type, called *distinctive-shape symbols*, is what we've been using all along. The other type, called *rectangular-shape symbols*, uses the same shape for all the gates, along with an internal label to identify the type of gate. Figure A–2 compares the two types. According to the IEEE standard, "the distinctive-shape symbol is not preferred." Some people think this statement means that rectangular-shape symbols *are* preferred. However, all the standard really says is that it gives no preference to distinctive-shape symbols compared to rectangular-shape symbols. On the other hand, since most digital designers, authors, and computer-aided design systems prefer the distinctive-shape symbols, that's what we use in this book.

Before the promulgation of the IEEE standard, logic symbols for larger-scale logic elements were drawn in an ad hoc manner; the only standard rule was to use rectangles with inputs on the left and outputs on the right. Although the logic symbol might contain a short description of the element (e.g., "3–8

ANOTHER KIND OF BUBBLE

In addition to the familiar bubble, the IEEE standard also allows an external, triangular "polarity symbol" to be used to specify active-low inputs and outputs, for which the external LOW level corresponds to the internal 1-state. However, under a positive-logic convention, the bubble and the triangular polarity symbol are equivalent, so we use the more traditional bubble in this appendix.

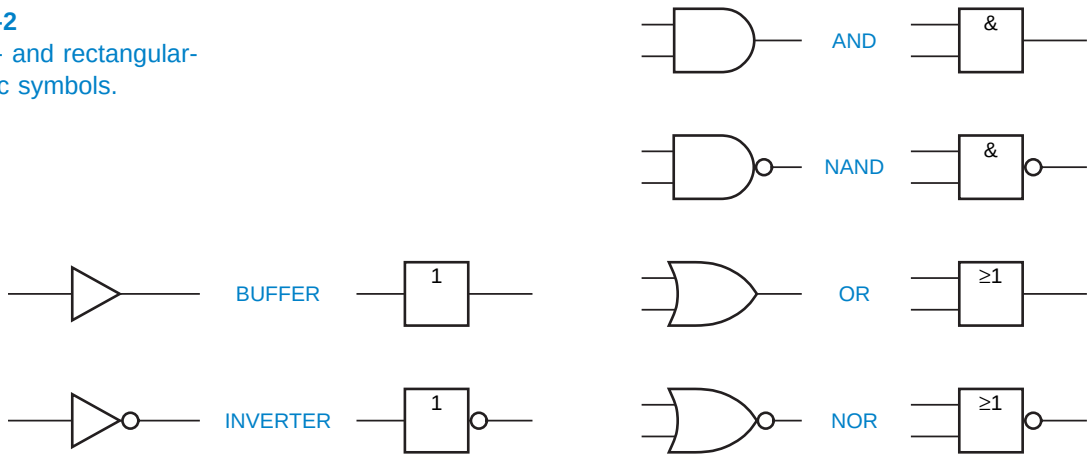


Figure A–2
Distinctive- and rectangular-shape logic symbols.

decoder," "2–1 multiplexer"), it was usually necessary to refer to a separate table to determine the element's logic function. However, the IEEE standard contains a rich set of concepts, such as bit grouping, common control blocks, and dependency notation, that allow some or all of a larger-scale logic element's function to be displayed in the symbol itself. We'll introduce these concepts as appropriate as we cover the symbols for various categories of devices in the sections that follow.

A.2 DECODERS

Chapter 5 used "traditional" logic symbols for decoders and other MSI logic elements. Although traditional symbols show the active levels of the inputs and outputs, they do not indicate the logical function of the device—you already have to know what a 74x138 or 74x139 does when you read its symbol.

The IEEE standard for logic symbols, on the other hand, allows a decoder's logic function to be displayed as part of the symbol, shown in Figure A–3. These symbols use several concepts of the standard:

- *Internal qualifying symbols.* Individual input and output signals may be labeled with qualifying symbols inside the logic-symbol outline to describe the signals' characteristics. In this book, we call such symbols *qualifying labels* for short.

internal qualifying symbol

qualifying label

- *General qualifying symbols.* The top of a logic symbol may contain an alphanumeric label to denote the general function performed by the device. Decoders and encoders (called *coders*) use the general qualifying symbol `X/Y` to indicate the type of coding performed, where `X` is the input code

general qualifying symbol

coder

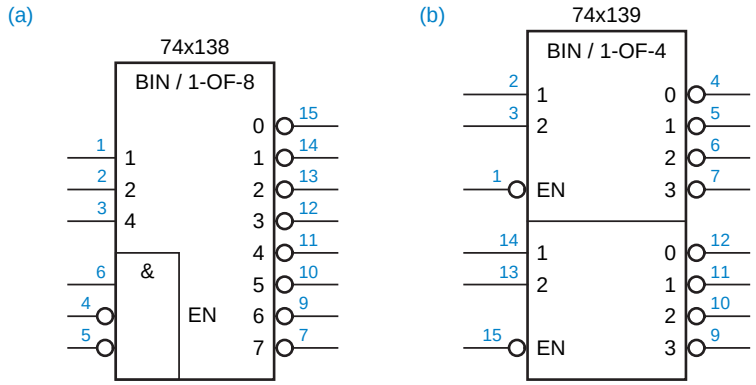


Figure A–3
IEEE standard symbols for decoders: (a) 74x138; (b) 74x139.

and Y is the output code. For example, a 3-to-8 decoder may be labeled BIN/1-OF-8.

internal value

- *Internal values.* Each input combination of a coder produces an "internal value" that is displayed by the coder's outputs. The internal values for a 3-to-8 decoder are 0–7.

input weight

- *Input weights.* The inputs of a coder may have qualifying labels indicating the numerical weights associated with those inputs. In this case, the internal value at any time is the sum of the weights of the asserted inputs. The input weights for a 3-to-8 decoder are 1, 2, and 4.

output value

- *Output values.* Each output may have a qualifying label listing the internal values that cause that output to be asserted. In a binary decoder, each output is asserted for just one internal value.

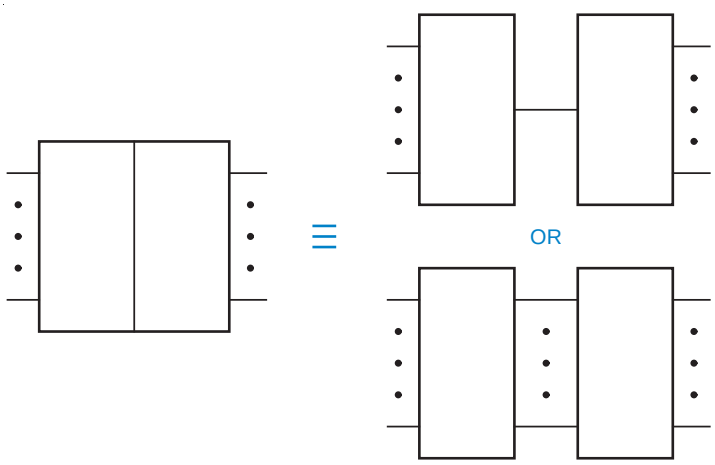
enable input

- *Enable input*. An enable input has the qualifying label EN and permits action when asserted. When negated, an enable input imposes the external high-impedance state on three-state outputs, and the negated state on other outputs. The 74x138 and 74x139 have active-low outputs, which are therefore 1 when the enable input is negated.

embedded element
abutted element

- *Embedded and abutted elements.* The outlines of individual logic elements may be embedded or abutted to form a larger composite symbol. There is at least one common logic connection when the dividing line between two outlines is perpendicular to the direction of signal flow, as shown in Figure A–4. For example, the 74x138 symbol has an embedded 3-input AND gate that drives the internal EN input. There is no connection between the elements when the dividing line is in the direction of signal flow, as shown in Figure A–5. For example, the 74x139 contains two separate 2-to-4 decoders.

Figure A–4
A composite symbol with one or more logic connections between its elements.



The ability to embed individual logic elements in a larger symbol is probably the most useful feature of the IEEE standard. For example, Figure A–6 shows the symbol for a 3-to-8 decoder with a different set of enable inputs than the 74x138. The fictitious *74x328* decoder is enabled when pin 6 is 0 or both pins 4 and 5 are 1. *74x328*

Figure A–5
A composite symbol with no connection between its elements.

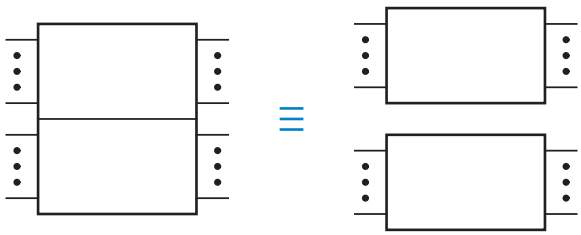
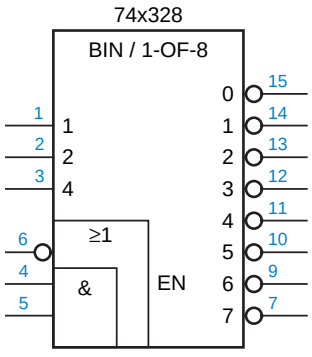


Figure A–6
IEEE standard symbol for a fictitious decoder, the 74x328.



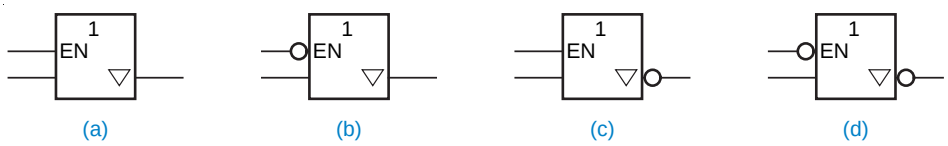


Figure A–7 IEEE standard rectangular-shape symbols for three-state buffers: (a) noninverting, active-high enable; (b) noninverting, active-low enable; (c) inverting, active-high enable; (d) inverting, active-low enable.

A.3 THREE-STATE BUFFERS

Three-state-buffer symbols use one more feature of the IEEE standard:

downward-pointing triangle

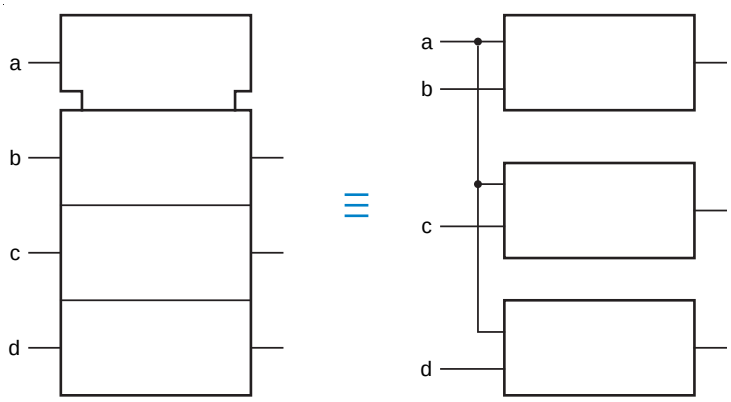
- *Downward-pointing triangle.* This denotes a three-state output.

Also recall that an enable input, labeled EN, is specifically understood to force all affected outputs to a disabled state. For three-state outputs, the disabled state is Hi-Z. Thus, the three-state buffers of Figure 5–35 are drawn as shown in Figure A–7. Another important feature of the IEEE standard is introduced in the symbols for MSI three-state buffers:

common control block

- *Common control block.* This concept, illustrated in Figure A–8, may be used with an array of related elements. Inputs to the common control block are understood to affect all the elements of the array.

Figure A–8
Common control block in an IEEE standard symbol.



Thus, symbols may be drawn for the 74x541 and 74x245 as shown in Figure A–9. The enable and direction inputs (pins 1 and 19) apply to all elements of the device. The ’541 and ’245 symbols introduce several other features of the standard:

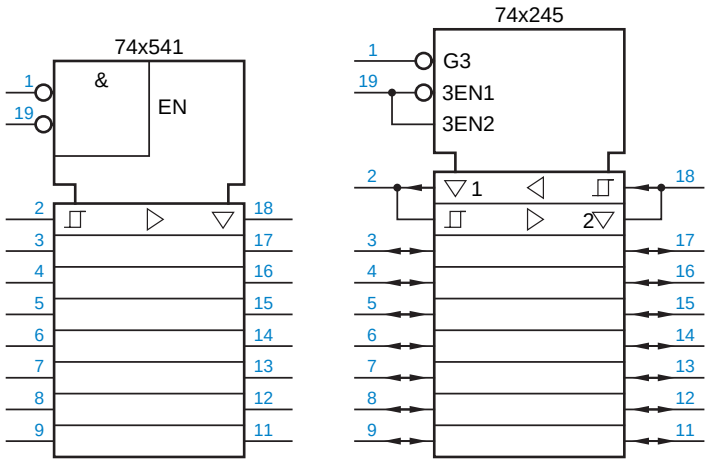


Figure A–9
IEEE standard logic symbols for the 74x541 and 74x245.

- *Hysteresis symbol.* Inputs bearing this symbol have hysteresis. *hysteresis symbol*
- *Right-pointing or left-pointing triangle.* These symbols are used to denote "amplification"; in the case of three-state buffers, they indicate an output circuit that has more fanout and can drive a heavier load than an ordinary output circuit. *right-pointing triangle* *left-pointing triangle*
- *Arrows.* These denote the direction of signal flow when it is not strictly left to right, as in the '245 symbol. *arrows*
- *Identical elements.* Only the first of two or more identical elements in an array must be drawn in detail. Thus, the bottom seven elements of the '245 symbol are understood to be identical to the top element (which in this case happens to be divided into two subelements, the three-state buffers for the two directions). *identical elements*

The '245 symbol also introduces *dependency notation*, a means of displaying some of the more common logical relationships among input and output signals. A few more concepts are needed to make this notation fly: *dependency notation*

- *Affecting signals.* An input signal (or, occasionally, an output signal) may affect other inputs or outputs in a way that can be displayed on the symbol. Such a signal has a qualifying label *L*i, where *L* is a letter that indicates the type of relationship or effect, and i is an integer that identifies the affected signals. The '245 internal signal labeled G3 is such a signal. *affecting signal*
- *Affected signals.* Inputs or outputs that are affected by a signal *L*i bear the *affected signal*

WHOOPS!

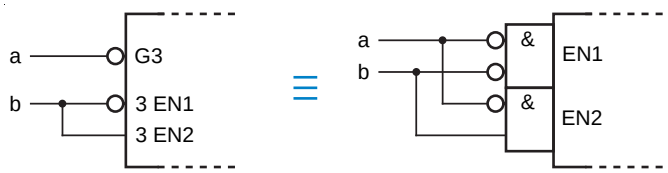
> The discussion of "affected signals" said "G3" when it should have said "the signal labeled G3." In the IEEE standard, G3 is a qualifying label, and other signals may have the same label. The standard does not specify unique names for internal and external signals. However, in the rest of this appendix, we'll take the liberty of using an internal signal's qualifying label as its name if no ambiguity results.

qualifying label i. If an affected signal requires a qualifying label of its own, then i is used as a prefix to that label. Thus, the signal labeled 3EN1 in the '245 symbol affects signals labeled 1, and is itself affected by G3.

AND dependency
Gi

- *AND dependency.* This relationship is denoted by Gi, and is a sort of enable function. Affected signals perform their "normal" functions only if Gi is asserted; otherwise, they are negated. In the '245 symbol, inputs 3EN1 and 3EN2 can "do their thing" only if G3 is asserted. Figure A–10 shows an equivalent notation for the dependent signals. Notice that dependency is defined in terms of internal, not external, signal values.

Figure A–10
Illustration of AND dependency.



enable dependency
ENi

- *Enable dependency.* This dependency is denoted by ENi, and has the same effect as EN inputs defined earlier. Thus, in the '245 symbol, the internal signal 3EN1, which is asserted only if pins 1 and 19 are LOW, enables the three-state drivers for pins 2 through 9.

Whewww! You might be thinking that this is a lot of trouble for a few lousy three-state buffers, but just wait until you see some of the IEEE-standard counter and shift-register symbols later in this appendix!

A.4 PRIORITY ENCODERS

Figure A–11 shows the IEEE standard symbol for a 74x148. Still more features of the standard are used in this symbol:

solidus

- *Solidus.* The slash in a label such as 0/Z10, called a *solidus* in the standard, separates multiple functions of a single internal signal. An equivalent notation is shown in Figure A–12.

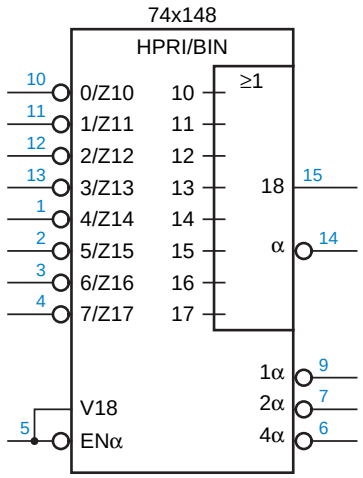


Figure A–11
IEEE standard logic symbol for the 74x148 8-input priority encoder.

- *OR dependency.* This relationship is denoted by Vi; the Vi signal is OR'ed with affected signals. Thus, affected signals perform their "normal" functions only if the Vi signal is negated; otherwise, they are asserted. *OR dependency* *Vi*

- *Virtual inputs and outputs.* These are internal signals, denoted by a horizontal line going nowhere, such as the ones labeled 10–17 in the '148 symbol. These signals have no external connection but affect or are affected by other signals via dependency notation. *virtual input* *virtual output*

- *Interconnection dependency.* This relationship is denoted by Zi, and indicates an internal connection. Affected signals equal the Zi signal, unless modified by additional dependency notation. Think of the Z as a zig-zag internal wire. *interconnection dependency* *Zi*

- *Greek letters.* A Greek letter may be used instead of an integer in qualifying labels to avoid ambiguity when the affected signals have a numeric function label, as in the inputs or outputs of a coder. *Greek letters*

If you understand these and the previously introduced features of the standard, you can "read" the functional behavior of a '148 right from its symbol. However, most people do the opposite—already knowing how a '148 works, they try to deduce how the standard works from the '148 symbol!

Figure A–12
Equivalent notation for solidi.

A.5 MULTIPLEXERS AND DEMULTIPLEXERS

bit-grouping symbol
internal value

The IEEE standard provides a special notation for multiplexers and demultiplexers. For example, Figure A–13 shows the IEEE symbols for multiplexer ICs that we discussed in Chapter 5. The general qualifying symbol MUX identifies a multiplexer. The bracket is called a *bit-grouping symbol* and indicates that the grouped inputs produce an *internal value* that is a weighted sum. The weights are given by the qualifying labels on the individual inputs; if the weights are all powers of 2, they may be replaced by the corresponding exponents, and all but the first and last exponents may be omitted "if no confusion is likely." Thus, the weights of pins 9, 10, and 11 of the 74x151 are 2^2, 2^1, and 2^0, respectively; if the input signal on these pins is 110, the internal value is 6.

range notation

The internal value produced by bit grouping affects other internal values or outputs according to the qualifying label written to the right of the grouping symbol. In the 74x151 multiplexer symbol, the notation $G\frac{0}{7}$ indicates AND dependency with signals whose labels are in the range 0–7. In other words, input *i* is selected (and transferred to the output) if and only if the internal value is *i*. There are two outputs, normally equal to the selected input and its complement. However, these outputs are asserted only if the enable input EN is asserted.

The 74x153 symbol also uses bit grouping for the select inputs, and it uses a common control block to indicate that the select inputs affect both sections of

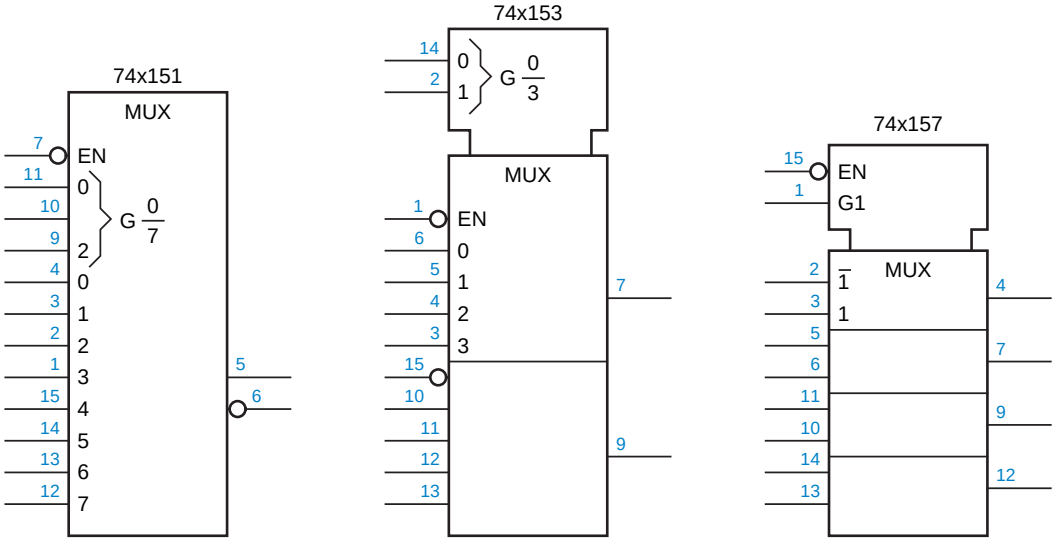


Figure A–13 IEEE standard symbols for multiplexers.

the multiplexer. Since the bottom half of the multiplexer is identical in function to the top, its qualifying labels are not repeated. Notice that each half has an independent EN input.

The symbol for the 74x157 does not use bit grouping, because it has only one select input. Instead, the select input is labeled G1, indicating that it has an AND dependency with signals bearing the identifier "1." Thus, pin 3 is selected only if G1 is asserted. Pin 2, on the other hand, bears the identifier "$\overline{1}$"; the overbar indicates that pin 2 is selected only if G1 is negated. All four sections are controlled by G1 in this way.

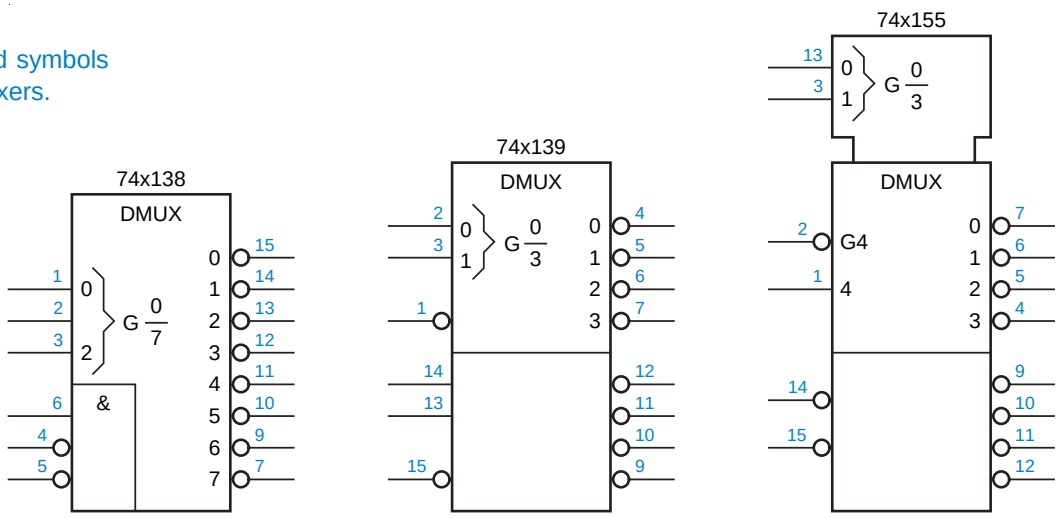


Figure A–14
IEEE standard symbols for demultiplexers.

Figure A–14 shows the IEEE standard demultiplexer symbols for the MSI decoder/demultiplexer ICs that we discussed in Chapter 5. The general qualifying symbol DMUX identifies a demultiplexer. The notation $G\frac{0}{7}$ indicates AND dependency with outputs whose labels are in the range 0–7. In other words, output *i* may be asserted only if the internal value is *i*. In addition, all of the other inputs (pins 4–6) must be asserted.

A similar notation is used in the 74x139, which contains two independent demultiplexers. Since the second demultiplexer is identical in function to the first, the qualifying labels are not repeated.

The symbol for the 74x155 uses a common control block to show that the same select inputs (and internal value produced by bit grouping) are used for both sections of the demultiplexer. Also, the input labeled "4" has an AND dependency on the input labeled "G4," so the selected output is asserted only if both of these inputs are asserted. Now, is that all clear?

A.6 EXCLUSIVE-OR AND PARITY FUNCTIONS

The IEEE standard symbols for EXCLUSIVE OR and EXCLUSIVE NOR gates are shown in Figure A–15. Either of two different notations may be used, depending on how you're thinking about the gate's function. The top symbols, with "=1" inside the symbol outline, assert their outputs when exactly one of the inputs is asserted. The bottom symbols, with "=" inside the symbol outline, assert their outputs when their inputs are equal.

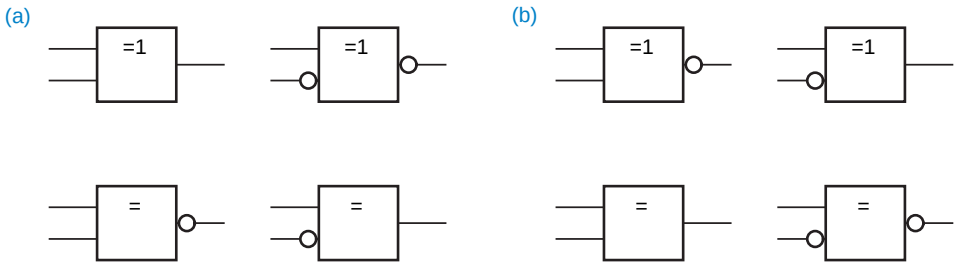
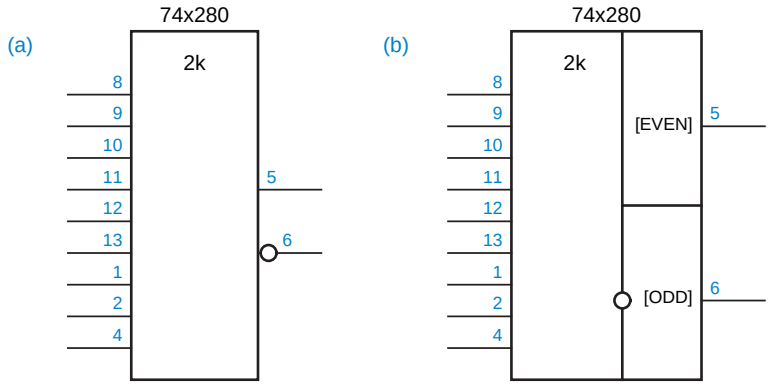


Figure A–15 IEEE standard symbols: (a) EXCLUSIVE OR gates; (b) EXCLUSIVE NOR gates.

The IEEE standard symbol for 74x280 9-bit parity generator is shown in Figure A–16(a). The general qualifying symbol "2k" at the top of the symbol indicates that the outputs are asserted if 2*k* of the inputs are asserted for some integer *k*. Thus, pin 5, an active-high output, and pin 6, an active-low output, both indicate even parity. The nature of the function, of course, is such that pin 6 could instead be viewed as an active-high output that denotes odd parity. The standard symbol for this interpretation of the device function is shown in (b).

Figure A–16
IEEE standard symbols for the 74x280 9-bit parity generator: (a) normal symbol; (b) both outputs active high.



A.7 COMPARATORS

IEEE standard symbols for MSI comparators are shown in Figure A–17. Like the select inputs of multiplexers, the data inputs have qualifying labels indicating their arithmetic weights in powers of 2 (0–3 corresponding to 2^0–2^3 in the ’85). The cascading inputs and outputs are labeled with the appropriate arithmetic function. In the ’682 symbol, the right-pointing triangle indicates that the outputs have high fanout capability, and hysteresis is indicated on the data inputs.

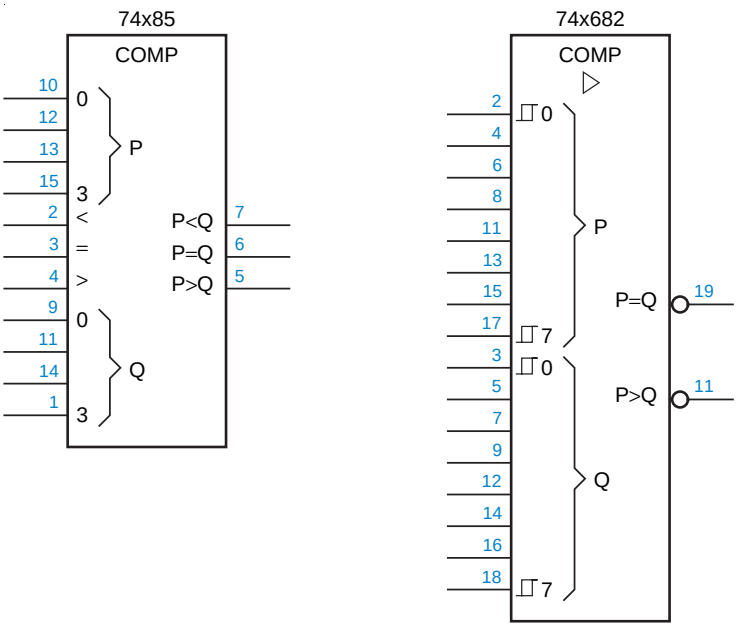


Figure A–17
IEEE standard symbols for MSI comparators.

A.8 ADDERS

The IEEE standard uses the general qualifying symbol “Σ” to identify an adder or addition function. Figure A–18(a) shows the symbol for a 74x283 4-bit adder. The numbers on the addend inputs and sum output indicate the weight of each pin, as a power of 2.

The IEEE symbol for a 74x181 4-bit ALU is shown in Figure A–18(b). The first five inputs of the common control block form a “mode control” word, which the standard designates by the letter M. The weights of the mode control bits are shown as powers of 2, and they designate a mode number in the range 0–31. According to the standard, a separate table accompanies the logic symbol to define the functions performed in each mode. The CP, CG, and CO outputs are enabled in modes 0–15. The four individual ALU blocks are labeled with

the weights of the bits they process. The IEEE symbol for the 74x182 carry lookahead circuit is much simpler, and is shown in (c).

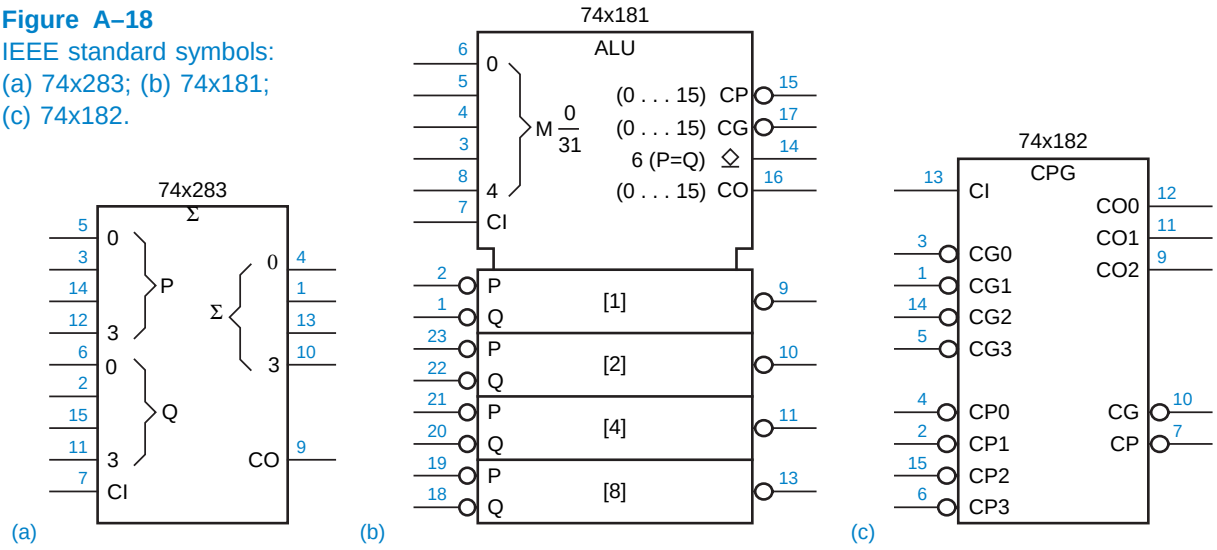


Figure A–18
IEEE standard symbols: (a) 74x283; (b) 74x181; (c) 74x182.

A.9 LATCHES, FLIP-FLOPS, AND REGISTERS

There are a few differences between traditional and IEEE symbols for latches and flip-flops. Figure A–19 shows IEEE standard symbols for SSI latches and flip-flops. A major difference is that the asynchronous preset and clear inputs are drawn on the left, not on the top and bottom. The names for these inputs are different, too: *S (set)* and *R (reset)*. Also, a clock input is simply named Ci, where i is an integer and all other inputs labeled with i (e.g., iD) are controlled by Ci; this is just another instance of dependency notation:

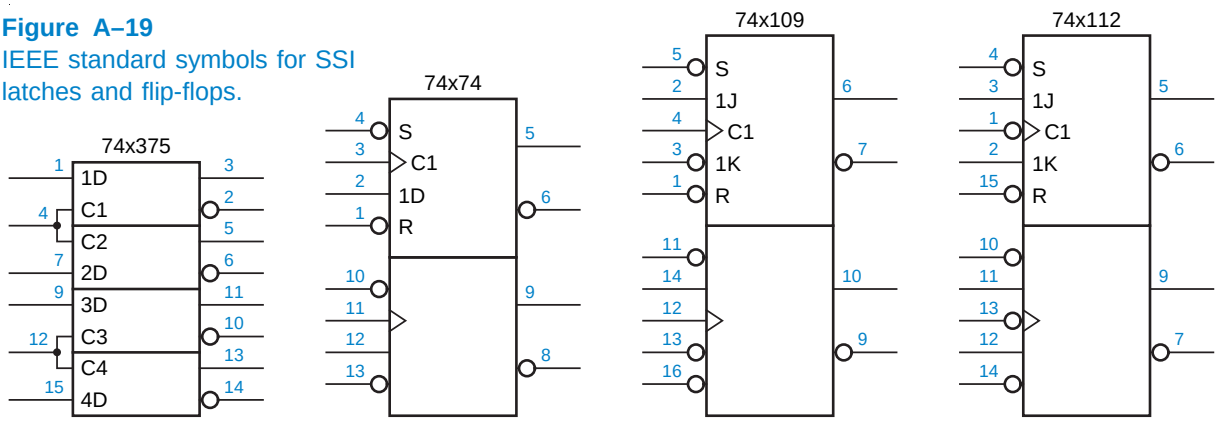
S (set)
R (reset)

control dependency
Ci

- *Control dependency.* This relationship is denoted by Ci and, like enable dependency, is a sort of enable function. It is intended to be used only for clock or timing inputs. Affected signals are enabled when the Ci input is in the internal 1-state or, if the Ci input has a dynamic indicator, on the 0-to-1 change in internal state.

The symbols for the ’74, ’109, and ’112 follow the usual IEEE convention that only the first of two or more identical elements must be drawn in detail when they are abutted in an array. Alternatively, the two independent sections of each SSI package in Figure A–19 may be drawn separately, as are other traditional and IEEE symbols for devices with independent sections, such as the 74x139.

Figure A–19
IEEE standard symbols for SSI latches and flip-flops.



This is especially useful if the logic functions performed by the two sections in a particular application are unrelated.

Figure A–20 shows the IEEE standard symbols for MSI latches and registers. All of these symbols make good use of the IEEE notation for common control blocks. In the ’373 and ’374 symbols, the label C1 on the clock input indicates that it controls all inputs that bear the label 1, that is, all of the 1D inputs. The downward-pointing triangles indicate three-state outputs; by convention, an input labeled EN enables such outputs.

In the ’377 symbol, the input labeled G1 is an enable for inputs bearing the label 1, that is, for the clock input 1C2. The clock input in turn controls all of the inputs bearing the label 2, that is, data inputs 2D. As usual, only the first of the eight identical flip-flop elements is drawn in detail.

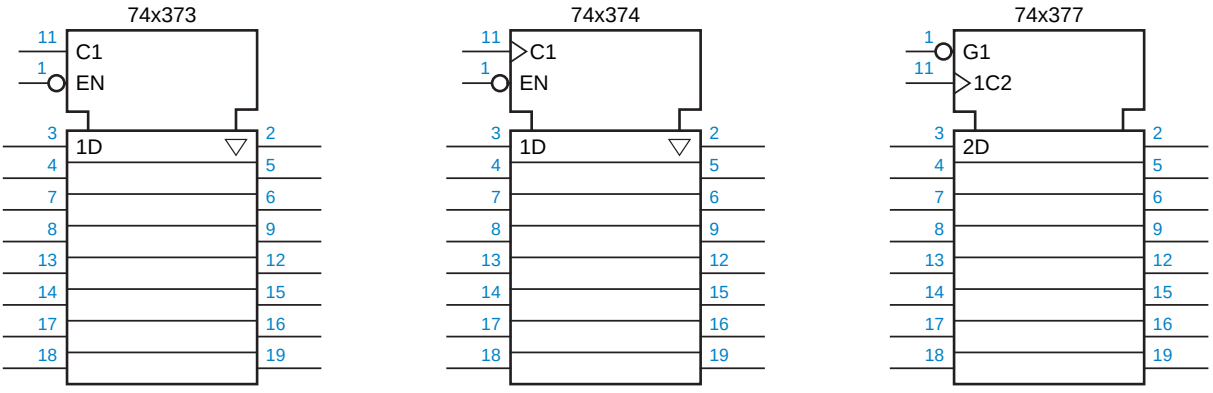


Figure A–20 IEEE standard symbols for MSI latches and registers.

A.10 COUNTERS

IEEE standard symbols for popular counters are shown in Figure A–21. Like the IEEE symbols for other MSI devices, the counter symbols make good use of the common-control-block and array features of the standard. The general qualifying symbol CTRDIV16 indicates that the device is a divide-by-16 counter, and labels [1]–[8] indicate the arithmetic weight of each counter bit. However, the additional notation used within the common control blocks to describe the devices' functions, though precise, is hardly intuitive. We'll have to describe a few more features of the standard to understand these symbols:

content input
CT=m input

- *Content input.* When an input signal bearing the label CT=m is asserted, the value m is loaded into the device. In the counter symbols, you might read "CT" as "count," but in general it means "content."

The only difference between the '161 and '163 symbols in Figure A–21 is that the 163's 5CT=0 has a control dependency on the clock input (C5), and is therefore a synchronous clear; the '161 has an asynchronous clear.

content output
CT=m output

- *Content output.* An output bearing the label CT=m is asserted when the content of the device is m.

In the '161 and '163 symbols, the output 3CT=15 is asserted when the counter is in state 15 and G3 is asserted.

mode dependency
Mi

- *Mode dependency.* This dependency is denoted by Mi and, like enable dependency, is a sort of enable function. Affected signals perform their

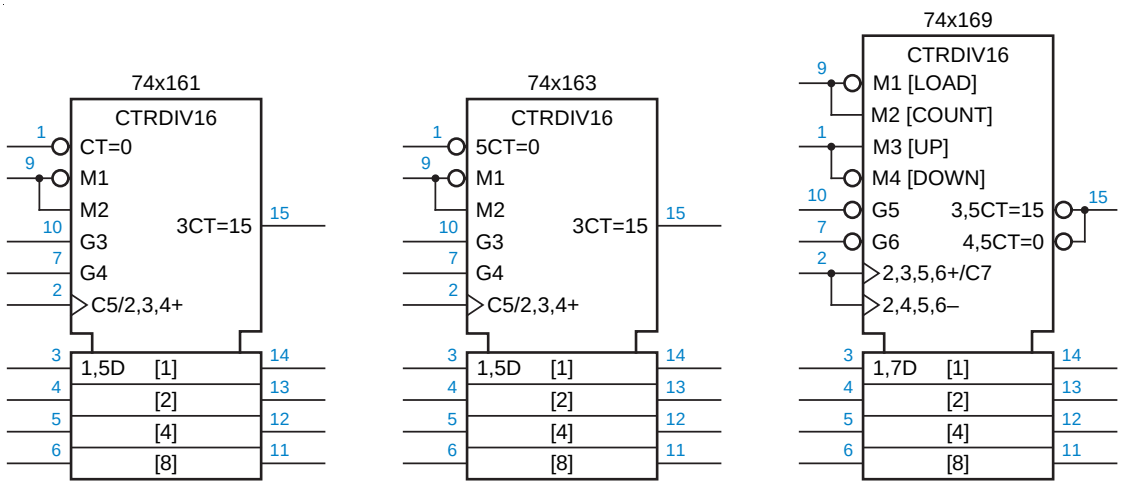


Figure A–21 IEEE standard symbols for counters.

"normal" functions only if Mi is asserted; otherwise, the affected signals have no effect on the device's function and are ignored.

- *Multiple dependencies.* A signal may be affected by several other signals. The identifiers of the affecting signals are listed, separated by commas, in the qualifying label of the affected signal. The dependencies are applied in the order that they are written, from left to right. If all of the dependencies are of the same type (e.g., AND dependency), then the order is irrelevant. *multiple dependencies*

- *Multifunction outputs.* A single external output signal, such as pin 15 in the '169 symbol, may have several sets of qualifying labels corresponding to multiple modes of operation. Such a signal may be represented by multiple outputs that are connected together externally. Such outputs normally have a functional OR relationship—the external signal is asserted if any internal signal is asserted. *multifunction output*

In the '161 and '163 symbols, the label 1,5D indicates that the data will be stored when M1 and C5 are asserted (but C5 is "asserted" only on an edge, because of its dynamic indicator). In the '169 symbol, the output 3,5CT=15 is asserted if M3 and G5 are asserted and the counter is in state 15; and 4,5CT=0 is asserted if M4 and G5 are asserted and the counter is in state 0; the external signal (pin 15) is asserted (LOW) if either of these internal signals is asserted.

- *Counting inputs.* When asserted, an input labeled with a + causes the device to count up once. An input labeled with a – causes the device to count down once. *counting input* *+* *–*

According to the '161 and '163 symbols, the device counts up on the rising edge of the signal on pin 2 if M2, G3, and G4 are asserted. The '169 counts up if M2, M3, G5, and G6 are asserted, and down if M2, M4, G5, and G6 are asserted. In each device, the qualifying labels for two separate functions of pin 2—load and count up—are drawn on the same input line, separated by a solidus; they could also have been drawn on two separate input lines, as we showed in Figure A–12 on page 799.

- *Nonstandardized information.* Descriptive function names and other nonstandardized (i.e., helpful) information may be written in brackets next to the qualifying labels in a symbol. *nonstandardized information*

The '169 symbol has four "nonstandard" labels to describe the traditional /LD and /ENP inputs. Theoretically, you don't need such labels if you understand the standard. Conversely, if signals are given meaningful names (as in traditional logic symbols), then you don't need the standard!

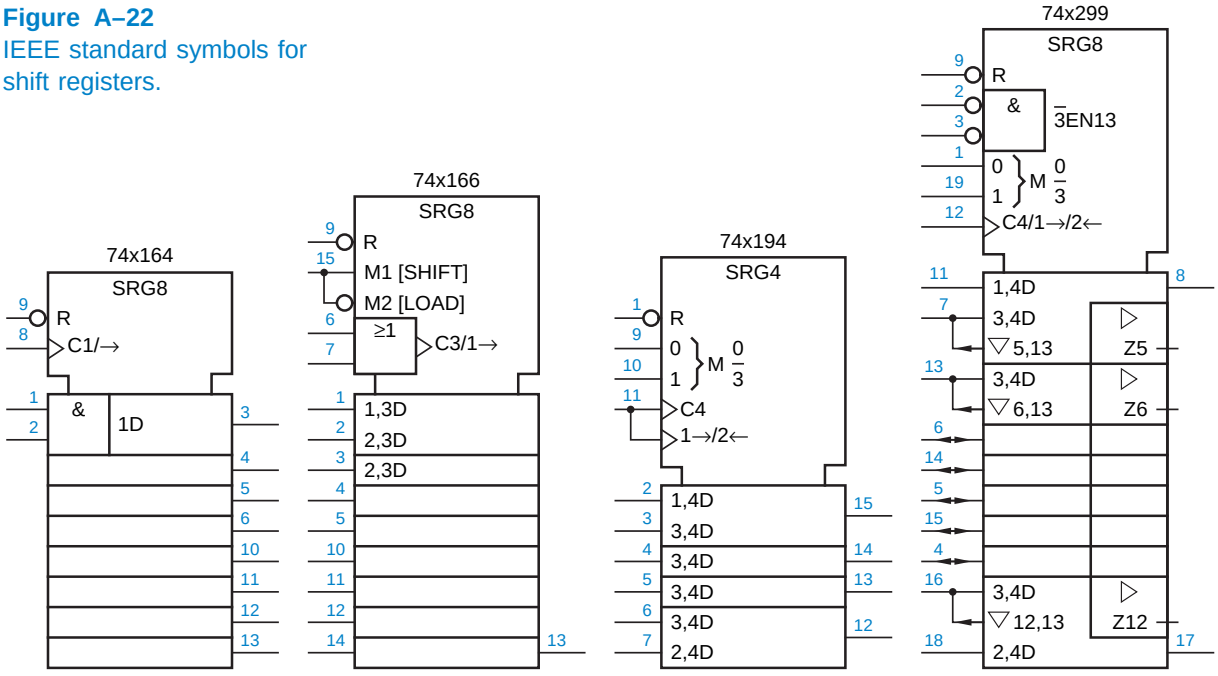
A.11 SHIFT REGISTERS

Figure A–22 shows the IEEE standard symbols for popular shift registers. The general qualifying symbol SRGn denotes an *n*-bit shift register. We've used most of the other notation in these symbols previously; there's just one new thing:

shifting input
→
←

- *Shifting inputs.* When asserted, an input labeled with a → causes the device to shift its information one position in the direction from left to right or from top to bottom. An input labeled with a ← causes a shift in the opposite direction.

Figure A–22
IEEE standard symbols for shift registers.



A.12 PROS AND CONS OF IEEE STANDARD SYMBOLS

The absence of a standardized method of giving unique *names* to pins and displaying the names on the logic symbol is probably the biggest defect of the IEEE standard. For example, how do we name the internal signals on pins 4, 5, and 6 of the 74x328 in Figure A–6? Worse, how can we distinguish between the internal signals on pins 3 and 11, which are both labeled "4" on the logic symbol? Apparently, the standard makers expected us to refer to these signals by pin number only. But this is awkward and inconvenient, not only in textbooks,

but also in design and debugging. It is far more natural to refer to a signal by a functional name than by a pin number. (Indeed, in ASIC design there are no pin numbers for internal signals!)

Still, the standard has several strengths:

- It's a standard.
- It's consistent.
- It supports, indeed, defines, the symbology used in bubble-to-bubble design.
- It is very complete. In addition to the basic devices covered in this book, the full IEEE standard covers many other less commonly used devices and structures.
- In most cases, the standard allows the function of a logic device to be defined unambiguously by the device symbol. For example, the '299 symbol in Figure A–22 conveys as much information as the function table in Table 9–3, and more.

That's the good news; now here's more bad news, if you hadn't already noticed:

- Although the standard allows you to figure out, with moderate effort, the precise function of an unfamiliar symbol, it does not provide any standard way to remind you quickly of the function of a familiar symbol. That is, it does not provide any standard, descriptive names for internal signals. Such items are relegated to the status of "nonstandard information," and their use is not encouraged.
- Many signals in IEEE symbols have no qualifying labels, while others have duplicates. This makes it awkward to talk about signals during design and debugging (e.g., "Connect X to the third bit up from the bottom. . . ").
- The standard encourages the omission of qualifying labels "if no confusion will result." That's like saying "Don't use your turn signal if nobody's nearby to see it." It's precisely when someone is driving in your blind spot that the habit of *always* using your turn signal can avert an accident. IEEE symbols have lots of "blind" inputs and outputs that may be hooked up incorrectly while drawing a schematic, or misapplied during debugging.
- Many possible symbols exist for moderately complex devices. For example, in the '194 symbol in Figure A–22, pin 11 could be handled as shown, or could be drawn as a single input line with the label C4/1→/2← or as three lines with the labels C4, 1→, and 2←. If used, these variations make it even more difficult to "eyeball" the symbol for a familiar device and recognize its functions.

What logic designers really need is a standard set of descriptive, functional, naming conventions for the inputs and outputs of MSI and LSI devices, one that is consistently followed by all data books and CAD systems. For example, it's ridiculous that in the present environment, a single manufacturer can produce 2-, 4-, and 8-input multiplexers with descriptive input names ranging from A–B to C0–C3 to D0–D7! Unfortunately, the industry has too much invested in documentation and CAD software for both inconsistent traditional symbols and unhelpful IEEE standard symbols, for a consistent, helpful naming standard to be deployed anytime soon. At least, today's ASIC designers can work to ensure a modicum of naming consistency and helpfulness in the logic "macrocells" that they define and use in their own designs.

EXERCISES

A.1 Draw and explain the IEEE standard symbol for a 74x49 seven-segment decoder.

A.2 Draw the IEEE standard symbol for the device in Exercise 5.43.

A.3 Explain all of the notation used in the IEEE standard symbol for a 74x148. (Some research is required to answer this one.)

A.4 Draw the IEEE standard symbol for the 74x155 used as a dual 2-to-4 decoder.

A.5 Draw the IEEE standard symbol for a 74x251 multiplexer.

A.6 Draw the IEEE standard symbol for the circuit in Exercise 5.56.

A.7 How does the meaning of the label 1D differ between the traditional and the IEEE symbols for registers like the 74x374? (*Hint:* How might pin 4 of a '374 be labeled in the IEEE standard?)

A.8 Draw an IEEE-standard symbol for the modified multiplexer of Figure 10–15.

