Recognized as an
American National Standard (ANSI)

ANSI/IEEE Std 91-1984

ANSI/IEEE Std 91a-1991

IEEE Standard Graphic Symbols for Logic Functions

(Including and incorporating IEEE Std 91a-1991, Supplement to IEEE Standard Graphic Symbols for Logic Functions)

Sponsor
IEEE Standards Coordinating Committee 11,
Graphic Symbols and Designations

ANSI/IEEE Std 91-1984
Approved 17 September 1981, IEEE Standards Board
Approved 26 October 1982, American National Standards Institute
Adopted for Mandatory Use 10 May 1984 Adopted U.S. Department of Defense (DoD)

ANSI/IEEE Std 91a-1991
Approved 27 June 1991 IEEE Standards Board
Approved 3 June 1994 American National Standards Institute
Adopted for Mandatory Use 1 July 1991 U.S. Department of Defense (DoD)

Abstract: Graphic symbols for representing logic functions or physical devices capable of carrying out logic functions are presented. Descriptions of logic functions, the graphic representation of these functions, and examples of their applications are provided. The symbols are presented in the context of electrical applications, but most may also be applied to nonelectrical systems (for example, pneumatic, hydraulic, or mechanical). The supplement provided additional internationally approved graphic symbols and made corrections as needed to IEEE Std 91-1984.
Keywords: dependency notation, industry standards, logic diagrams, logic function, logic symbols, military standards

---

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

ANSI/IEEE Std 91-1984
10 May 1984
(Superseding
ANSI Y32.14-1973
IEEE Std 91-1973)

Acceptance Notice

This non-government document was adopted on 10 May 1984, and is approved for use by the Department of Defense (DoD). The IEEE has furnished the clearances required by existing regulations. Copies of the document are stocked at the Standardization Documents Order Desk, Bldg 4D, 700 Robbins Avenue, Philadelphia, PA 19111-5094 for issue to DoD activities only. Other Government activities, contractors, private concerns, or other requestors must obtain the document from:

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017-2394 USA

Title of Document: **IEEE Standard Graphic Symbols for Logic Functions**

Document No: **ANSI/IEEE Std 91-1984**

Date of Specific Issue Adopted: **9 June 1984**

Releasing Industry Group: **The Institute of Electrical and Electronics Engineers, Inc.**

Custodians:
Army—AR
Navy—SH
Air Force—13

Military Coordinating Activity:
Army—AR

(Project DRPR—0231)

Review Activities:
Army—AV, MI, AM CR, ER
Navy—OS, AS
Air Force—15, 17, 11

User Activities:
Army—AT
Navy—MC

NOTICE: When reaffirmation, amendment, revision, or cancellation of this standard is initially proposed, the industry group responsible for this standard shall inform the military coordinating activity of the proposed change and request participation.

NOTICE OF ADOPTION
ADOPTION NOTICE 1
1 JULY 1991
IEEE Std 91a-1991
27 JUNE 1991

The Institute of Electrical and Electronics Engineers, Inc., IEEE Std 91a-1991, was adopted on 1 July, 1991 and is approved for use by the Department of Defense (DoD). Copies of this document are stocked at the Standardization Documents Order Desk, Bldg 4D, 700 Robbins Avenue, Philadelphia, PA 19111-5094 for issue to DoD activities only. Other Government activities, contractors, private concerns, or other requestors must obtain the document from:

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017

Title of Document: **Supplement to IEEE Standard Graphic Symbols for Logic Functions**

Date of Specific Issue Adopted: **27 June 1991**

Releasing Industry Group: **The Institute of Electrical and Electronics Engineers, Inc.**

Custodians:
Army—AR
Navy—OS
Air Force—13

Military Coordinating Activity:
Army—AR

(Project DRPR—0314)

Review Activities:
Army—AT, AV, CR, ER, MI
Navy—AS, EC, MC, SH
Air Force—11, 13, 15, 16, 17, 99
DLA—DH, ES
NSA—NS

User Activities:
Army—AT

AMSC N/A AREA DRPR

DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE that have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying all patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Introduction

(This introduction is not part of IEEE Std 91-1984, IEEE Standard Graphic Symbols for Logic Functions, or of IEEE Std 91a-1991, Supplement to IEEE Std 91-1984.)

Two standards are included in this document: IEEE Std 91-1984, IEEE Standard Graphic Symbols for Logic Functions, and IEEE Std 91a-1991, Supplement to IEEE Std 91-1984. In this edition (published in 1996), the two standards have been merged to make it more convenient for the user.

The following sections of the original IEEE Std 91-1984 were extensively revised and replaced by IEEE Std 91a-1991:

Pictorial Table of Contents
Table of Contents
Section 6: Symbols for Highly Complex Functions
Appendix A: Recommended Symbol Proportions
Appendix D: Integrated Circuits Used as Examples, Commercial Part Numbers vs. Symbol Numbers
Index

Numerous other revisions to IEEE Std 91-1984 in Sections 1 through 5 were made. The new Table of Contents and the new Index indicate specifically where revisions were made. Further, change bars marked in the outside margins indicate the locations of significant changes. Each standard has a unique foreword, included below.

IEEE Std 91-1984 foreword

This standard defines an international *language* by which it is possible to determine the functional behavior of a logic circuit as described on a logic or circuit diagram with minimal reference to supporting documentation. Like natural languages, the *language* set forth in this standard has been designed to allow a single concept to be expressed in one of several different ways, according to the demands of a particular situation. Consequently, this standard does not attempt, nor intend to establish single *correct* symbols for particular devices. A symbol appropriate for one application of a device may not be appropriate for another.

The contributors to this standard represent a broad range of institutions, technologies, and documentation needs. They include industrial, governmental, and educational organizations, producers and consumers of devices and equipment, users and non-users of computer-aided design and drafting, and a considerable range of aesthetic preferences. That a consensus of such diverse interests could be achieved in producing this standard is indicative of not only the utility of the approach, but more importantly, of the increasing need among designers and maintainers of digital systems for a common and more nearly complete symbolic language.

This revision is the result of a continuing activity to arrive at a useful notation to permit free interchange of information on the design of binary-operated controls and systems. It is the latest step in a program that began in 1956 within the IEEE to develop a comprehensive single standard, consistent with ongoing developments in technology and logic symbology, from several ad hoc, industry, military, and international standards. In 1960, an ad hoc group on logic diagram graphic symbols was formed within the American National Standards Institute in order to develop a draft American Standard. In 1961, this committee became a permanent subcommittee, Y32.14, of the Graphic Symbols Committee, Y32, under the cosecretariat of ASME and IEEE. Its work resulted in the publication of IEEE Std 91-1962 (ANSI Y32.14-1962), adopted in 1965 by the US Navy. The subcommittee was reorganized in 1969 to prepare a new draft standard that would have broader acceptance and be in accord with the developments within the International Electrotechnical Commission (IEC). ANSI/IEEE Std 91-1973 (Y32.14-1973) subsequently received approval from ANSI, and the US Department of Defense, and was substantially compatible with IEC Pub 117-15, Recommended Graphical Symbols: Binary Logic Elements. Since 1977 the preparing committee, IEEE SCC 11.9, has worked closely with IEC Technical Committee 3 to prepare major new revisions of this standard and IEC Pub 617, Part 12 (the successor to Pub 117, Part 15). The aim was for a US standard that would be mutually compatible with the IEC standard, broadly acceptable, and that would provide notation or guidelines by which any SSI through VLSI

device might be usefully and accurately represented. Fourteen drafts of this document were prepared in parallel with drafts of the new IEC document by an ad hoc working group of SCC 11.9 before the preparers and reviewers believed that these goals had been met.

Symbology, such as language and technology, will continue to evolve, and IEEE SCC 11.9 will continue to work with IEC TC3 to update logic symbol standards as future needs dictate. Suggestions for improvement of this standard are welcomed. They should be addressed to:

Secretary
IEEE Standards Board
The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017-2394 USA

When this standard was approved, the IEEE Standards Coordinating Committee on Graphic Symbols and Designations, Subcommittee SC 11.9 on Logic Symbols, had the following membership:

Thomas R. Smith, *Chair**

Charles J. Andrasco	Verle W. Harris	V. Tom Rhyne*
Richard R. Barta	Gordon A. Knapp*	John P. Russell* **
M. Paul Chinitz*	Joseph Kochanski*	Henry H. Seaman
Edwin D. Edwards†	George E. Lemieux†	Thomas R. Smith*
Philip H. Enslow, Jr.	Fred A. Mann*‡	Robert Tobias
Charles A. Fricke†	Rollin P. Mayer	James L. Vargo*
Arthur C. Gannett	Mark Mrozowski*	Frank B. Wahl, Jr.*
James J. George	Conrad R. Muller*	R. J. Yuhas
Tommy J. Gibbons	John B. Peatman*	
Edward J. Harrington*	Richard W. Pinger*	

US Air Force Logistics Command LOLME (Robert C. Gardner)
US Army Armament Research and Development Command DRDAR-TST-S (M. E. Taylor)

*Member of ad hoc working group
†Resigned
**Secretary, September 1979 to April 1980; prepared all drafts through 1981
‡Head of delegation to IEC TC3/SC3A/WG2 on Logic Symbols

The IEEE Standards Coordinating Committee on Graphic Symbols and Designations SCC 11 had the following membership:

C. A. Fricke†, *Chair*
C.R. Muller, *Secretary*

Robert B. Angus, Jr.	John B. Peatman	Roger M. Stern
John M. Carroll	J. William Siefert	Leter H. Warren
Gordon A. Knapp	Thomas R. Smith	Steven A. Wasserman

†Resigned

At the time the standard was approved on September 17, 1981, the IEEE Standards Board had the following membership:

Irvin N. Howell, Jr., *Chair*
Irving Kolodny, *Vice Chair*
Sava I. Sherr, *Secretary*

G. Y. R. Allen	Jay Forster	Frank L. Rosa
J. J. Archambault	Kurt Greene	Robert W. Seelbach
James H. Beall	Loering M. Johnson	Jay A. Stewart
John T. Boettger	Joseph L. Koepfinger	Virginius N. Vaughan, Jr.
Edward Chelotti	John E. May	Art Wall
Edward J. Cohen	Donald T. Michael*	Robert E. Weiler
Len S. Corey	J. P. Riganati	

*Member emeritus

IEEE Std 91a-1991 foreword

IEEE Std 91-1991 provides a way to describe a single logic function with a symbol. The symbols that result from using IEEE Std 91-1984 have sometimes been called “the new logic symbols,” despite the fact that one can still find distinctive-shape AND and OR gates, J and K inputs to flip-flops, and most other time-honored features of logic symbols. However, prior to IEEE Std 91-1984, there were no standardized symbols or symbology methods, for example, for memories, demultiplexers, arithmetic elements, multifunction registers, open-collector outputs, digital switches, or devices with multiple clocks or address ports. What was truly new was that these features were integrated into a comprehensive system for constructing logic symbols to meet not only current, but future needs. Where the “old” system relied on an inventory of symbols, the “new” system relies on an inventory of symbol elements and techniques. The inherent ability of the “language” to describe new devices can be seen in the relatively few additions that have been required in this supplement.

Because IEEE Std 91-1984 is not based on English-language mnemonics, symbols based on it may communicate less to the uninitiated reader who has already memorized a component catalog. However, if the reader knows the language, and especially if he or she does not intimately know the device, the system can communicate far more than any nonstandard mnemonic-based system, with a smaller set of things to be remembered. The notation does more than suggest to the knowledgeable user what various inputs and outputs do. Instead, it provides a rather detailed description.

The use of this system requires training, as does any other system of notation. The difference between this system and those that preceded it is that, once trained, the user has available more information in less space, from more sources, and with less need for supporting documentation. Once the user has learned the rules, he or she has learned them not only for existing devices, but for new ones as well. That IEEE Std 91-1984 is virtually identical to the national standards of dozens of other countries is more than a small bonus.

Future editions of IEEE Std 91 and IEC 617-12 are expected to continue to evolve to address developments in the technology of logic devices. Suggestions for the improvement of this standard are welcomed.

The supplement to IEEE Std 91-1984, IEEE Std 91a-1991 was the result of nearly ten years of ongoing work by an ad hoc Working Group of IEEE SCC 11.9 and IEC TC3 (International Electrotechnical Commission Technical Committee 3) to harmonize IEEE Std 91 and its international counterpart, IEC 617-12. The supplement brings IEEE Std 91-1984 into conformance with IEC 617-12 (1991) and, once more, includes some additional symbols and techniques published in advance of their publication by IEC.

In addition to corrections and clarifications to IEEE Std 91-1984, the supplement included new material to cover simplification of arrays (2.3.1.2), inputs with special amplification (3.3-9.5), in-line negation indication (4.3.1), definitions of codes used in coders (5.4.1, 5.4.2), display elements (5.15), bus indicators (6.1.9), and representation of data paths on internal diagrams (6.2.2).

This document was prepared by an ad hoc Working Group of Subcommittee 11.9 on Logic Symbols of the IEEE Standards Coordinating Committee 11 on Graphic Symbols and Designations. The members of the working group were:

Thomas R. Smith, *Chair*†

C. Andrews
R. B. Angus
R. R. Barta
L. Burns
N. Caparossi
T. Connors
L. Ciskowski
E. Draper
R. Farni
B. Fiekel
J. George
L. Hanson*
G. F. Iacovoni
S. Ingram
G. A. Knapp
D. Kollmorgan
R. Landry
D. Lauer
F. Mann†
J. Massaro
J. F. Morongiello
C. R. Muller
J. A. Pendell
J. Rouse
J. P. Russell
L. Sandoval
J. Sents
P. Simison
R. Sohan
R. M. Stern
J. Vargo
L. Wong*

*Assisted in the preparation of drafts and publication copy.
†U.S. member of IEC TC3/SC3A/WG2 on Logic Symbols

The IEEE Standards Coordinating Committee 11 on Graphic Symbols and Designations delegated the balloting of this supplement to the following persons who constituted the members of SCC11.9 that approved this supplement for submission to the IEEE Standards Board:

R. B. Angus
R. R. Barta
J. Bloodgood
N. Caporossi
G. Carnovale
L. Ciskowski
D. M. Clark
P. H. Enslow
A. Gannett
J. George
L. Hanson
A. Hartikainen
G. F. Iacovoni
S. Ingram
S. Johnson
R. Kaskowitz
G. A. Knapp
R. Landry
F. Mann
J. Massaro
R. P. Mayer
C. A. Mitchell
C. R. Muller
N. Raskhodoff
J. Rouse
R. Sandige
L. Sandoval
T. R. Smith
T. Suzuki
P. Tremblay
J. Vargo
L. H. Warren
A. Wiseman

When the IEEE Standards Board approved this standard on June 27, 1991, it had the following membership:

Marco W. Migliaro, *Chair*
Donald C. Loughry, *Vice Chair*
Andrew G. Salem, *Secretary*

Dennis Bodson
Paul L. Borrill
Clyde Camp
James M. Daly
Donald C. Fleckenstein
Jay Forster*
David F. Franklin
Ingrid Fromm
Thomas L. Hannan
Donald N. Heirman
Kenneth D. Hendrix
John W. Horch
Ben C. Johnson
Ivor N. Knight
Joseph L. Koepfinger*
Irving Kolodny
Michael A. Lawler
John E. May, Jr.
Lawrence V. McCall
Donald T. Michael*
Stig L. Nilsson
John L. Rankine
Ronald H. Reimer
Gary S. Robinson
Terrance R. Whittemore

Deborah A. Czyz *IEEE Standards Project Editor*

* Member Emeritus

Acknowledgements

This standard owes its existence to the cooperative efforts of many people and organizations, most especially to the excellent technical contributions of the members of the ad hoc working group and the support given each of them by their staffs and employers. Particular thanks goes to the following organizations for their invaluable assistance:

For preparation and distribution of meeting, draft, and final documents: General Electric Co, Texas Instruments, Inc, Control Data Corp, Digital Equipment Corp, Allen-Bradley, and D.A.T.A. Inc.

For financial support of US delegations at IEC working-group meetings, without which this work would not have progressed successfully: Texas Instruments, Inc, McGraw-Hill Book Co, Hewlett-Packard, Digital Equipment Corp, and Control Data Corp.

An American National Standard
IEEE Standard Graphic Symbols for Logic Functions

1. Introduction

1.1 Purpose

This standard (the combined documents of ANSI/IEEE 91-1984 and its supplement, ANSI/IEEE Std 91a-1991) establishes symbols for representing logic functions or devices. These symbols enable users to understand the logic characteristics of these functions or devices without requiring specific knowledge of their internal characteristics. ANSI/IEEE Std 91a-1991 supplements ANSI/IEEE 91-1984 by providing additional internationally approved graphic symbols for logic functions or devices as well as corrections to the 1984 document.

1.2 Scope

This standard contains graphic symbols for representing logic functions or physical devices capable of carrying out logic functions. Descriptions of logic functions, the graphic representation of these functions, and examples of their applications are given. The symbols are represented in the context of electrical applications, but most may also be applied to nonelectrical systems (for example, pneumatic, hydraulic, or mechanical).

1.3 Organization

This document includes material from both ANSI/IEEE Std 91-1984 and its supplement document, ANSI/IEEE Std 91a-1991.

1.3.1 Supplementary material from ANSI/IEEE Std 91a-1991

All material from the supplement standard, ANSI/IEEE Std 91a-1991, has been included in this document. This supplement either replaces or adds to the original text in ANSI/IEEE 91-1984. Change bars in the margin mark the location where the supplementary material has been implemented.

1.3.2 Revised or deleted symbols

Symbols from superseded editions of this standard that have been revised or deleted are shown in Appendix B.

1.4 Applicable documents

1.4.1 Industry standards

The latest editions of the following industry documents may also be of interest:

American National Standards[1]

ANSI Y1.1-1972, Abbreviations for Use on Drawings and in Text.

ANSI Y10.20-1975 with Supplement Y10.20a-1975, Mathematical Signs and Symbols for Use in Physical Science and Technology.

ANSI/IEEE Std 315-1975 (Y32.2-1975), Graphic Symbols for Electrical and Electronics Diagrams. (Including Reference Designation Class Designation Letters).

ANSI/IEEE Std 200-1975 (Y32.16-1975), Reference Designations for Electrical and Electronics Parts and Equipments.

1.4.2 Military standards[2]

The latest edition of the following military document may also be of interest:

MIL-STD-12, Military Standard Abbreviations for Use on Drawings, Specifications, Standards, and in Technical Documents.

1.4.3 International standards[3]

The following International Electrotechnical Commission (IEC) Publications may be of interest:

617 Part 12, Recommended Graphical Symbols, Binary Logic Elements. (All of the symbols in 617 Part 12 are included in this national standard, ANSI/IEEE Std 91-1984.)

113 Part 7, Diagrams, Charts and Tables; Application of Logic Symbols and the Preparation of Logic Diagrams.

All symbols and rules not shown, either explicitly or implicitly, in IEC Publications 113 or 617 at the time of publication of this standard are identified by the mark "•" (which is not a part of the symbol).[4]

1.5 Definitions

1.5.1 logic state: One of two possible abstract states that may be taken on by a logic (binary) variable.

1.5.1.1 0-state: The logic state represented by the binary number 0 and usually standing for an inactive or false logic condition.

[1]ANSI publications are available from the Sales Department of American National Standards Institute, 11 West 42nd Street, New York, NY 10036.
[2]MIL publications are available from Customer Service, Defense Printing Service, 700 Robbins Avenue, Philadelphia, PA 19111-5094.
[3]IEC Standards are available in the US from American National Standards Institute, 11 West 42nd Street, New York, NY 10036.
[4]Except as individually noted, these rules and symbols are, in the opinion of the preparing committee, likely to be published in future editions of IEC Publications 113 or 617.

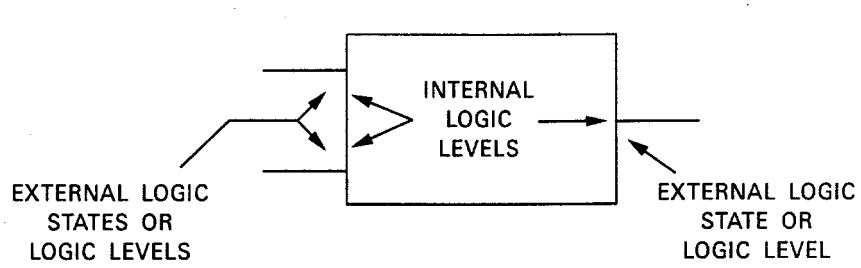
1.5.1.2 1-state: The logic state represented by the binary number 1 and usually standing for an active or true logic condition.

1.5.1.3 external logic state: A logic state assumed to exist outside a symbol outline

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.

1.5.1.4 internal logic state: A logic state assumed to exist inside a symbol outline at an input or an output.

Illustration of the concept



1.5.2 logic level: Any level within one of two nonoverlapping ranges of values of a physical quantity used to represent the logic states.

NOTE — A logic variable may be equated to any physical quantity for which two distinct ranges of values can be defined. In this standard, these distinct ranges of values are referred to as logic *levels* and are denoted H and L.

H is used to denote the logic level with the more positive algebraic value, and L is used to denote the logic level with the less positive algebraic value.

In the case of systems in which logic state are equated with other physical properties (for example, positive or negative pulses, presence or absence of a pulse), H and L may be used to represent these properties or may be replaced by more suitable designations.

1.5.2.1 high (H) level: A level within the more positive (less negative) of the two ranges of the logic levels chosen to represent the logic states.

1.5.2.2 low (L) level: A level within the more negative (less positive) of the two ranges of logic levels chosen to represent the logic states.

1.5.3 Logic conventions and polarity indication:

1.5.3.1 positive logic convention: The representation of the external 1-state and the external 0-state by the high (H) and low (L) levels, respectively.

1.5.3.2 negative logic convention: The representation of the external 1-state and the external 0-state by the low (L) and high (H) levels, respectively.

1.5.3.3 direct polarity indication: The indication of the relationship between the internal logic state and the external logic level at each input and output of every logic element directly by means of the presence or absence of the polarity symbol (◺). (See Symbols 3.1-4 through 3.1-8.)

1.5.4 logic function: A definition of the relationships that hold among a set of input and output logic variables.

1.5.4.1 combinational logic function:: A logic function in which there exists one and only one resulting combination of states of the outputs for each possible combination of input states.

NOTE — The terms combinative and combinatorial have also been used to mean combinational.

1.5.4.2 sequential logic function: A logic function in which there exists at least one combination of input states for which there is more than one possible resulting combination of states at the outputs.

NOTE — The outputs are functions of variables in addition to the present states of the inputs, such as time, previous internal states of the element, etc.

1.5.4.3 bistable logic function (flip-flop): A sequential logic function that has two and only two stable internal output states.

1.5.5 element: As used within this standard, a representation of all or part of a logic function within a single outline, which may, in turn, be subdivided into smaller elements representing subfunctions of the overall function. Alternatively, the function so represented.

1.5.6 qualifying symbol: A symbol added to the basic outline of an element to designate the physical or logic characteristics of an input or output of the element or the overall logic characteristics of the element.

1.5.7 dependency notation: A means of obtaining simplified symbols for complex elements by denoting the relationships between inputs, outputs, or inputs and outputs, without actually showing all the elements and interconnections involved. (See Section 4.)

1.5.8 distributed function (dot logic, wired logic): A logic function (either AND or OR) implemented by connecting together outputs of the appropriate type (see Symbols 3.3-3 through 3.3-7); these outputs are the inputs of the logic function thus formed; the joined connection is the output.

1.6 Orientation of qualifying symbols

All lettering inside a symbol, including alphanumeric qualifying symbols, should be oriented parallel to the input and output lines.

The following symbols shall be oriented as described within this standard with respect to the inputs, outputs, and edge of the elements in which they appear.

2.3.2-1	Common control block
2.3.3-1	Common output element
3.3-1 through 3.1-11	Negation, polarity, and dynamic input symbols
3.2-1 through 3.2-6	Internal connection symbols
3.3-25 3.3-26	Bit-grouping symbols
3.3-37 3.3-38	Line-grouping symbols
▷	Amplifier symbol (Symbol 3.3-9 and 5.2)

All other qualifying symbols shall be oriented as shown with respect to the text inside the element.

2. Symbol construction

2.1 Composition of a symbol

2.1.1 General composition

A symbol comprises an outline or combination of outlines together with one or more qualifying symbols. Application of the symbols requires, in addition, the representation of input and output lines.

Symbols often make use of internal labels. It is essential to maintain clarity between internal labels that apply to signal lines and internal labels that serve as general qualifying symbols.

Labels shall be clearly separated, both horizontally and vertically, from each other and from other symbols or parts of the outline with which they are not meant to be associated. Signal-line labels shall also be placed such that they are not likely to be interpreted as general qualifying symbols.

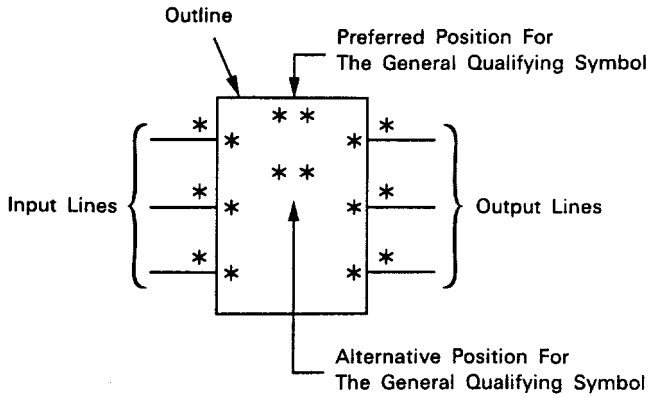
The preferred location for the general qualifying symbol is in the top of the element to which it applies, centered horizontally. An alternative position is centered vertically and horizontally. Application of these positioning rules for the general qualifying symbols should take into account the positions of embedded elements or labels associated with signal lines that alter the horizontal area available for placement of the general qualifying symbol.

Labels associated with signal lines shall be centered vertically next to those lines. Horizontally, there shall be a space between the signal-line label and the outline of the symbol or symbols attached to the outline. This horizontal space shall be large enough to allow a distinct separation, yet small enough to unambiguously associate the label with the given input or output.

See Appendix A for recommended proportions of symbols and IEEE Std 991 for more detailed recommendations on sizes and spacing.

Because of limitations in the publishing systems used, symbols in this document do not always follow the preceding recommendations.

Illustration of the concept



NOTE 1—The single * denotes possible positions for qualifying symbols relating to inputs and outputs.
NOTE 2—If, and only if, the function of an element is completely determined by the qualifying symbols associated with its inputs or outputs or both, no general qualifying symbol is needed.

General additional information may be included in a symbol outline as permitted by an applicable standard for preparation of diagrams (for example, IEC Publication 111-3 clause 3.4).

For supplementary information relating to the function of a complex logic element, see Section 6.

In some figures, lowercase letters that are not part of the symbols have been shown outside the outline just to identify the inputs and outputs as referenced in the description.

2.1.2 Nonstandardized information

Information not standardized in this standard relating to a specific input (output) may be shown in square brackets inside the outline adjacent to the relevant input (output) and should follow (precede) any qualifying symbols applying to the input (output), as shown in Symbol 5.13-17.

Other nonstandardized information may also be shown in square brackets inside the symbol outline, for example, Symbols 5.2-3 and 5.4-2.

2.1.3 Logic states of outputs

All outputs of a single, unsubdivided element always have identical internal logic states determined by the function of the element except when indicated otherwise by an associated qualifying symbol or label inside the outline.

The subdivision of an element and the qualifying symbols referred to here include those explicitly shown and also those only implied according to the simplification rules of 2.3.1.1.

2.2 Outlines

The length-width ratio of rectangular outlines is arbitrary.

For permitted distinctive-shape symbol outlines see Section 5. The proportions of the symbol should remain constant in accordance with Appendix A.

For combination of outlines see 2.3.

No	Symbol	Description
2.2-1		Element outline (square shown, see 2.2)

2.3 Use and combination of outlines

2.3.1 Embedded and abutted elements

To reduce the space required for the representation of a group of associated elements, the outlines of the elements may be embedded or abutted provided the following rules are observed.

1) Embedded or abutted symbols may use nonrectangular outlines composed of vertical and horizontal lines to make efficient use of space (for example, Symbol 5.12-6).
   Usage of distinctive-shape symbols in combination to form complex symbols (for example, used as embedded symbols) is discouraged.
2) There is no logic connection between elements when the line common to their outlines is in the direction of signal flow.
   NOTE — This rule does not necessarily apply in those arrays in which there exist two or more directions of signal flow, for example, in symbols with a common control block or a common output element.

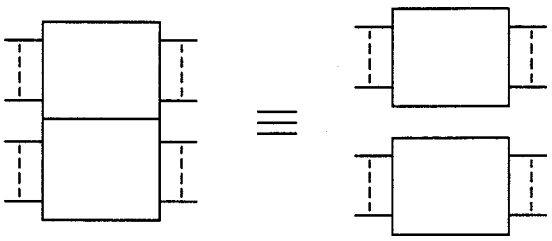


Illustration of the concept

3) There is at least one logic connection between elements if the line common to the two outlines is perpendicular to the direction of signal flow. If no indications are shown on either side of the common line, it is assumed that there exists only one logic connection. Because common control blocks are not elements, no logic connections to or from a common control block exist except those to the attached array and those that are explicitly shown.

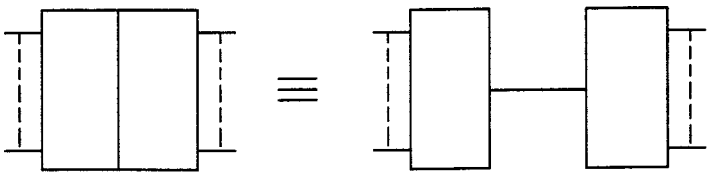


Illustration of the concept

4) Each connection can be shown by the presence of qualifying symbols at one or both sides of the common line. If confusion is likely about the number of logic connections, use can be made of the internal connection symbol (Symbol 3.2-1).

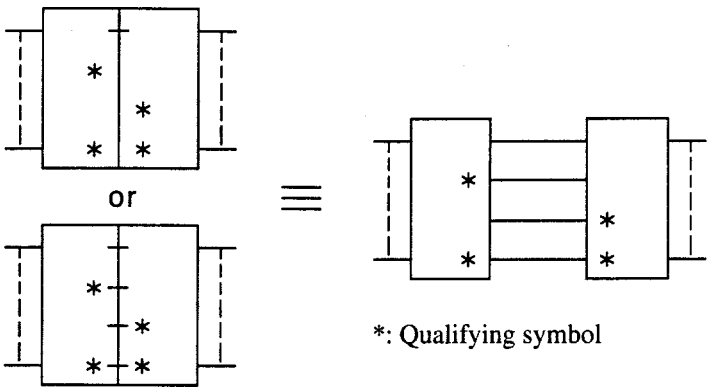


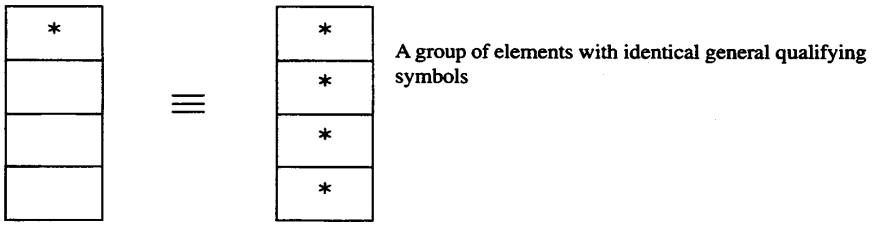
Illustration of the concept

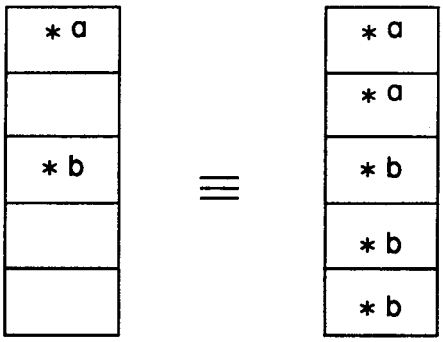
For an example of combining the preceding concepts, see Symbol 5.1-19.

2.3.1.1

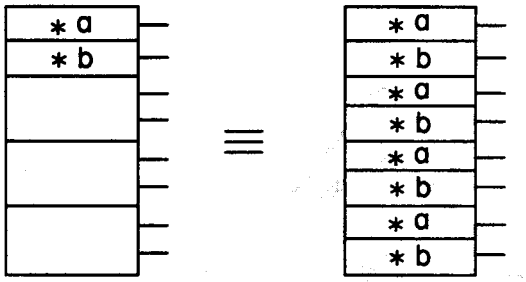
To represent an array of elements having the same qualifying symbols, it may be sufficient to show the symbols that are inside the outlines in only the first of the outlines, provided no confusion is likely. Similarly, identical elements that are subdivided may each be represented by an unsubdivided outline.

Illustration of the concept

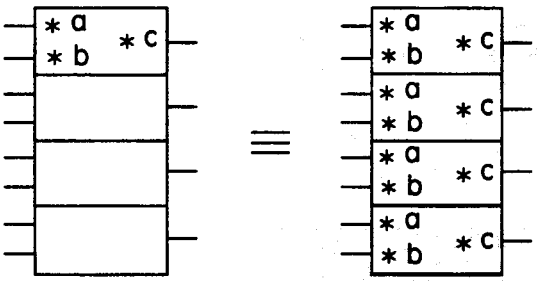




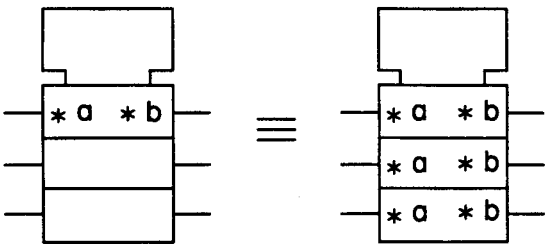
Two successive groups of elements



Two interlaced groups of four elements each



A group of elements with identical input and output qualifying symbols



A group of elements with identical input and output qualifying symbols and common control block (Symbol 2.3.2–1)

If there are affecting inputs or outputs in the sense of dependency notation (Section 4) within the elements of the array, it is assumed that the identifying numbers of those inputs or outputs and of inputs or outputs affected thereby differ in each element of the array.

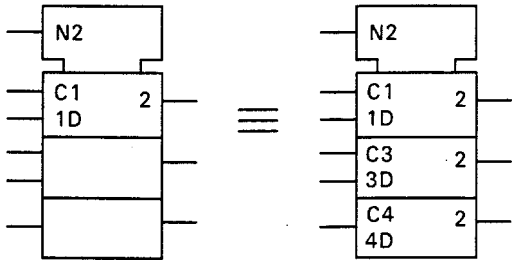
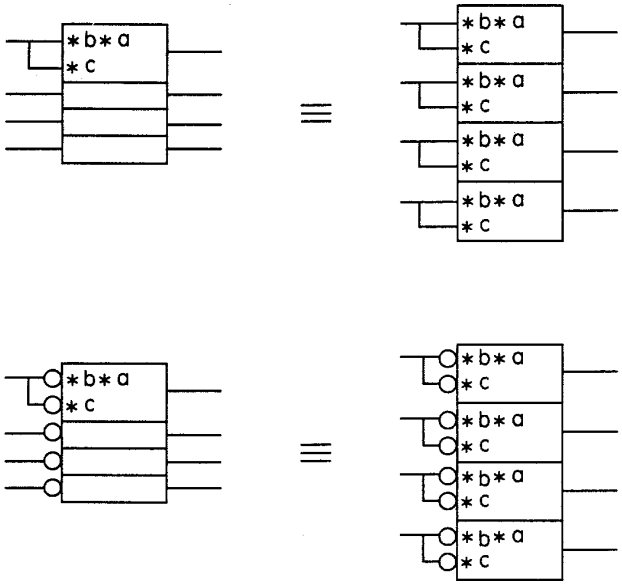


Illustration of the concept

2.3.1.2

If, in a simplified array of identical elements, the representation of the functions of a terminal requires two or more lines connected together outside the outline, it is sufficient to show these lines with only the first element and represent them by a single line with each simplified element. Symbols outside the outline common to all the lines connected together shall be shown with this single line. Symbols outside the outline not common to all the lines connected together may be omitted, or the more suitable set may be shown.



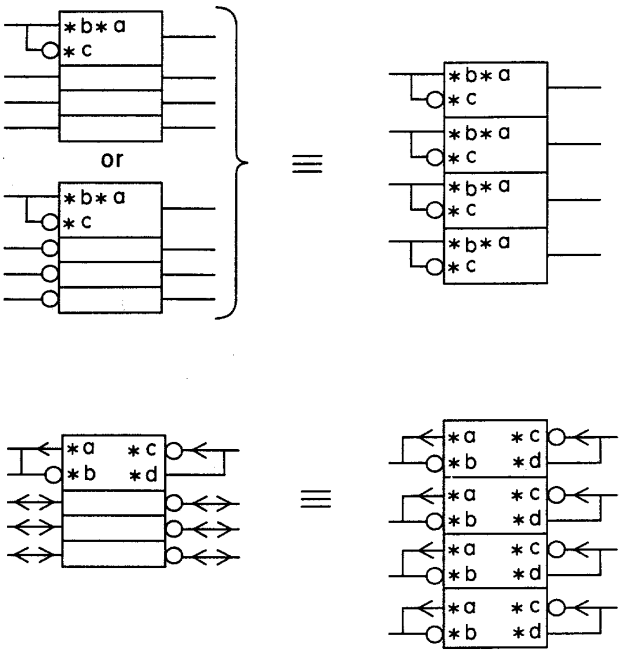


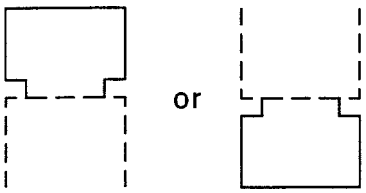
Illustration of the concept

2.3.2 Common control block

The common control block may be used in conjunction with an array of related elements as a point of placement for inputs or outputs associated with more than one element of the array, or with no element of the array. Such inputs and outputs shall be labeled if appropriate.

When an input shown at a common control block is an affecting input in the sense of dependency notation, it is connected as an input only to those elements of the array in which its identifying number appears. When an input shown at a common control block is not an affecting input in the sense of dependency notation, it is an input common to or affecting all elements of the array.

The common control block may be placed at either end of an array of related elements.

No	Symbol	Description
2.3.2-1		Common control block outline

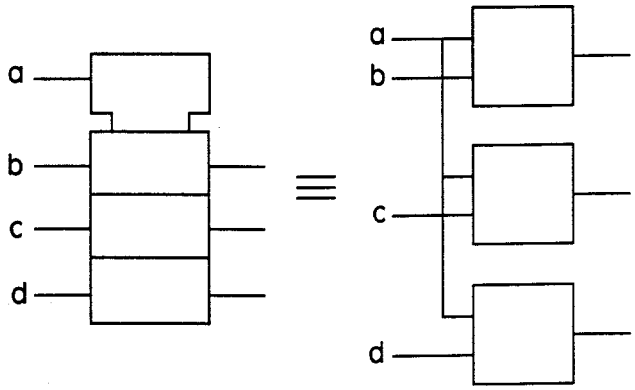


Illustration of the concept

2.3.2.1 Weighted arrays

If the elements in an array have an intrinsic weighted relationship (for example, as in an up/down counter) and it is not otherwise indicated, the element next to the common control block is assumed to be the element with the smallest weight.

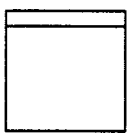
2.3.3 Common output element

A common output depending on all elements of the array can be shown as the output of a common output element. In the case where any array element has more than one output, the common output element may be used only if those outputs always have identical internal logic states. There is one internal connection from each of the elements to the common output element and these shall not be shown. In addition, the common output element may have other inputs and they shall be explicitly shown. The function of the common output element shall be indicated.

Each input of a common output element that is an output of the array has the same internal logic state as that output.

A common output element shall be shown either

1) Inside a common control block, or
2) At either end of the array, but opposite the common control block if there is one.

No	Symbol	Description
2.3.3-1		Common output element outline NOTE — When embedded in a common control block, the double line shall be placed at the edge of the element away from the array to which the control block is connected. Otherwise, the double line shall be placed at the edge nearest the array.

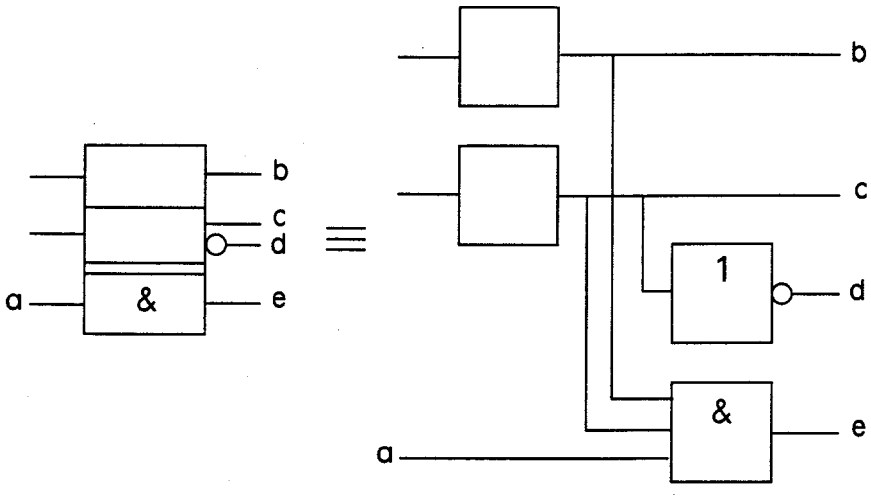
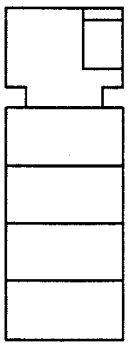


Illustration of the concept

NOTE — The common output element, like any element, must have at least one qualifying symbol to indicate its function. The "&" (Symbol 5.1-3) is the qualifying symbol for AND.

Array with common control block and common output element:



2.3.3.1 Array of common output elements

If it is appropriate to show an array of common output elements, the double line need be shown only once if no confusion is likely. In this case, there is one internal connection from each of the elements of the array to each of the common output elements. There is no connection between common output elements unless explicitly shown.

Array with two common output elements:

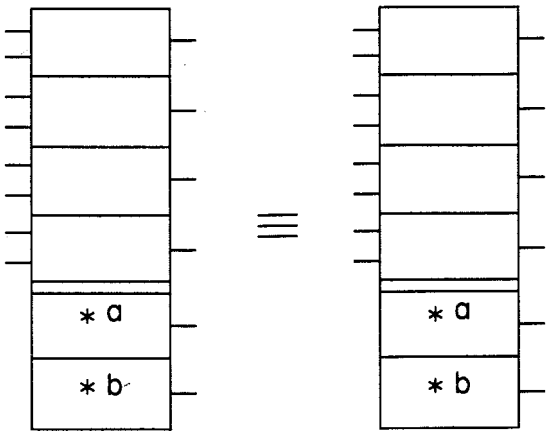


Illustration of the concept

3. Qualifying symbols associated with inputs, outputs, and other connections

3.1 Negation, polarity, and dynamic input symbols

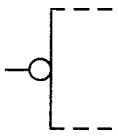
In this section qualifying symbols are shown that indicate the relationship between internal logic state and external logic state or level.

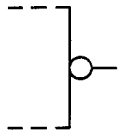
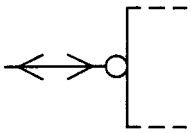
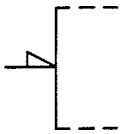
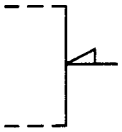
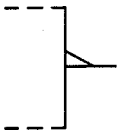
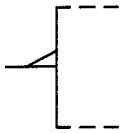
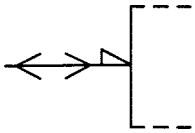
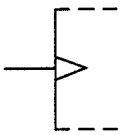
If none of the qualifying symbols of this section is shown at an input or output, it is assumed that the internal logic 1-state corresponds with

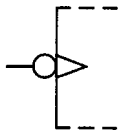
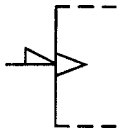
1) The external logic 1-state in a diagram using the positive logic convention or the negative logic convention, or
2) The logic H-level in a diagram using direct polarity indication

3.1.1

The qualifying symbols for negation and polarity shall not be used on the same diagram to relate internal logic states to external logic states or levels. However, the polarity symbol may be used on the same diagram with an internal negation symbol as defined in Symbols 3.2-2 or 3.2-4. For example, see Symbol 5.4-7.

No	Symbol	Description
3.1-1		Negated input The external 0-state produces the internal 1-state. NOTE — The connecting line may be drawn through the negation symbol.

No	Symbol	Description
3.1-2		Negated output The internal 1-state produces the external 0-state. See NOTE to Symbol 3.1-1.
3.1-3		Negated input/output port, shown on the left-hand side. See 3.4-5.
3.1-4		Active-low input The L-level on the input produces the internal 1-state NOTE — This qualifying symbol is designated the polarity symbol and shall point in the direction of signal flow.
3.1-5		Active-low input in the case of signal flow from right to left See NOTE to Symbol 3.1-4.
3.1-6		Active-low output The internal 1-state produces the L-level on the output. See NOTE to Symbol 3.1-4.
3.1-7		Active-low output in the case of signal flow from right to left See NOTE to Symbol 3.1-4.
3.1-8		Active-low input/output port, shown on the left-hand side. See 3.4-5.
3.1-9		Dynamic input The transition from the external 0-state to the external 1-state produces a transitory internal 1-state. At all other times the internal logic state is 0. On diagrams using direct polarity indication, the transition from the L-level to the H-level on the input produces a transitory internal 1-state. At all other times the internal logic state is 0.

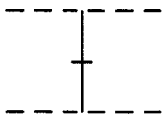
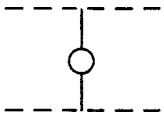
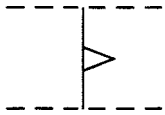
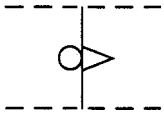
No	Symbol	Description
3.1-10		Dynamic input with negation The transition from the external 1-state to the external 0-state on the input produces a transitory internal 1-state. At all other times the internal logic state is 0.
3.1-11		Dynamic input with polarity symbol The transition from the H-level to the L-level on the input produces a transitory internal 1-state. At all other times the internal logic state is 0.

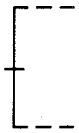
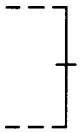
3.2 Internal connections

An internal connection is a connection within an element.

In this section, qualifying symbols are shown that indicate the relationship between internal logic states at internal connections.

The symbols in this section may be used to show the logic relationships between elements whose outlines are combined.

No	Symbol	Description
3.2-1		Internal connection The internal 1-state (0-state) of the output of the element on the left produces the 1-state (0-state) at the input of the element on the right. NOTE — The internal connection symbol may be omitted if no confusion is likely. (See also 2.3.)
3.2-2		Internal connection with negation The internal 1-state (0-state) of the output of the element on the left produces the 0-state (1-state) at the input of the element on the right.
3.2-3		Internal connection with dynamic character The transition from the internal 0-state to the internal 1-state of the output of the element on the left produces a transitory 1-state at the input of the element on the right. At all other times the logic state at the input of the element on the right is 0.
3.2-4		Internal connection with negation and dynamic character Transition from the internal 1-state to the internal 0-state of the output of the element on the left produces a transitory 1-state at the input of the element on the right. At all other times the logic state at the input of the element on the right is 0.

No	Symbol	Description
3.2-5		Internal input (virtual input) This input always stands at its internal 1-state unless it is affected by a dependency relationship that has an overriding effect. (See Symbols 5.9-13 and 5.13-3.) NOTES: 1 — Internal inputs and outputs have internal logic states only. 2 — The qualifying symbols of 3.1 shall not be applied to internal inputs and outputs, except for the dynamic input symbol. 3 — This symbol should not be confused with Symbol 3.2-1, which is used for abutted elements.
3.2-6		Internal output (virtual output) The effect of this output must be indicated by dependency notation. See NOTES to Symbol 3.2-5.

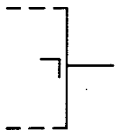
3.3 Symbols inside the outline

3.3.1

If two or more inputs have the same qualifying symbol for their functions, they are assumed to stand in an OR relationship, except in the cases of bithreshold and extension inputs (Symbols 3.2-2 and 3.3-10) where the relationship should be appropriately indicated.

3.3.2

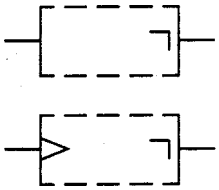
The descriptions of Symbols 3.3-14 through 3.3-22 may give the reader the impression that these are always dynamic inputs. This is not the case, as it must be remembered that the internal logic state, as determined by the external logic state or level, may be modified by the state of other inputs (for example, Cm inputs, 4.3.7). If such inputs have a dynamic character, Symbol 3.1-9 should be added (for example, Symbol 5.13-13).

No	Symbol	Description
3.3-1		Postponed output The change of the internal state of this output is postponed until the input signal that initiates the change returns to its initial external logic state or logic level. The internal logic state of any input(s) affecting or affected by the *initiating* input must not change while this *initiating* input stands at its internal 1-state or the resulting output state will not be specified by the symbol. If the input signal that initiates the change appears at an internal connection, the change of state is postponed until the output at the internal connection returns to its initial logic state. NOTES: 1 — If the postponed output symbol is shown without a prefix, it should be assumed that the output is postponed with respect to each Cm, +, → , ← , -, or T input (for the Cm input see 4.3.7); in all other cases the identifying numbers (or if necessary the full labels) of all inputs with respect to which the output is postponed must be shown as a prefix to this symbol (for example, Symbol 5.13-l7). 2 — The postponed output symbol should be a right angle with legs of equal length to avoid confusion with other symbols, for example, the number 7. 3 — For the application of this symbol and additional explanation, see 5.9.

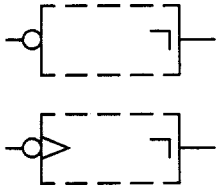
The following illustrations assume there are no other inputs with overriding effects.

When using the positive or the negative logic convention, the transition at the output takes place when the input changes:

1) From its external 1-state to its external 0-state, or

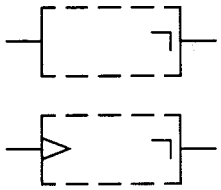


2) From its external 0-state to its external 1-state

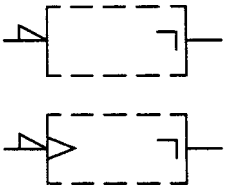


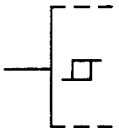
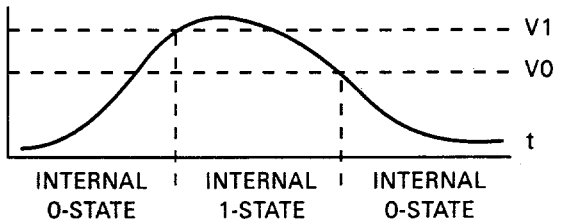
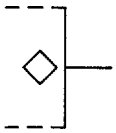
On a diagram using direct polarity indication, the transition at the output takes place when the input changes:

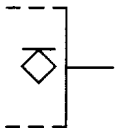
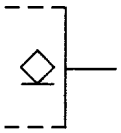
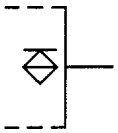
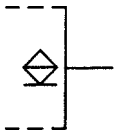
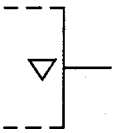
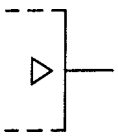
1) From its H-level to its L-level, or

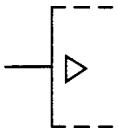
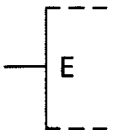
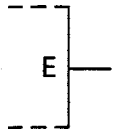
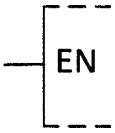


2) From its L-level to its H-level

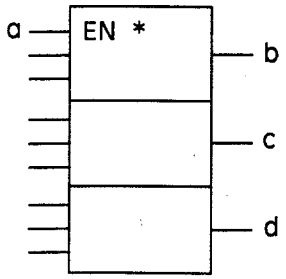


No	Symbol	Description
3.3-2		Input with hysteresis Bithreshold input The input takes on its internal 1-state when the external signal level reaches a threshold value, V1. It maintains this state until the external signal level has returned through V1 and reaches another threshold value, V0. If this symbol (without the negation symbol or polarity symbol) appears on a diagram that uses either direct polarity indication or the positive-logic convention, V1 is more positive than V0. If it appears on a diagram that uses the negative-logic convention, V1 is more negative than V0. If the negation symbol or polarity symbol is present, then the relationship between V1 and V0 is reversed. EXTERNAL SIGNAL (POSITIVE-LOGIC CONVENTION, NO NEGATION SYMBOL)  NOTES: 1 — When used with a dynamic input symbol (Symbols 3.1-9 through 3.1-11), the hysteresis symbol shall be shown following the dynamic input symbol even though its effect occurs first. For use with the special amplification symbol (Symbol 3.3-9), the line-grouping symbol (Symbol 3.3-37), or dependency notation, see 4.4.3. 2 — Most practical devices exhibit this characteristic to some extent. This symbol is only used when an identification of the characteristic is important to the application of the device. For use as a general qualifying symbol, see 5.3.
3.3-3		Open-circuit output (for example, open-emitter, open-collector, open-source, or open-drain output) One of the two possible internal logic states of this type of output corresponds to an external high-impedance condition. In order to produce the proper logic level in this condition, an externally connected component, often a resistor, is required. This type of output is usually capable of forming part of a distributed function. NOTES: 1 — This symbol shall be drawn adjacent to but not touching the output line, except when using the bit-grouping symbol (Symbol 3.3-26) in the manner defined in 4.4.4, which permits an alternative position. 2 — Although this symbol is shown inside the outline, it refers to external states and levels only. The meaning of this symbol is not altered by the presence of a negation or polarity symbol (Symbol 3.1-2, 3.l-6, or 3.1-7). 3 — To indicate which logic level is the one with the low impedance, use of Symbol 3.3-4 or 3.3-5 is recommended. 4 — If all outputs in a device family (e.g., 10 K ECL) are of the same open-circuit type, and if no confusion is likely, this symbol may be omitted from all outputs in the family.

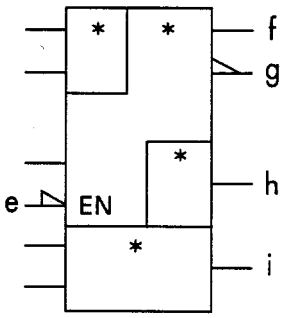
No	Symbol	Description
3.3-4		Open-circuit output (H-type) (for example, PNP open-collector, NPN open-emitter, P-channel open-drain, or N-channel open-source) When not in its external high-impedance state, this type of output produces a relatively low-impedance H-level. When used a part of a distributed function, a positive-logic OR function or a negative-logic AND function is performed. See also Symbols 5.1-15 and 5.1-16. See NOTES 1, 2, and 4 to Symbol 3.3-3.
3.3-5		Open-circuit output (L-type) (for example, NPN open-collector, PNP open-emitter, N-channel open-drain, or P-channel open-source) When not in its external high-impedance state, this type of output produces a relatively low-impedance L-level. When used as part of a distributed function, a positive-logic AND function or a negative-logic OR function is performed. See also Symbols 5.1-15 and 5.1-16. See NOTES 1, 2, and 4 to Symbol 3.3-3.
3.3-6		Passive-pulldown output This type of output (like the H-type open-circuit output, Symbol 3.3-4) can be used as part of a distributed function to perform a positive-logic OR function or a negative-logic AND function but it produces both the H-level and the L-level without the need for an additional external component. See also Symbols 5.1-15 and 5.1-16. See NOTES 1 and 2 to Symbol 3.3-3.
3.3-7		Passive-pullup output This type of output (like the L-type open-circuit output, Symbol 3.3-5) can be used as part of a distributed function to perform a positive-logic AND function or a negative-logic OR function but it produces both the H-level and the L-level without the need for an additional external component. See also Symbols 5.1-15 and 5.1-16. See NOTES 1 and 2 to Symbol 3.3-3.
3.3-8		3-state output This output can take on a third external state, which is a high-impedance state, having no logic significance. For example, see Symbol 5.2-4. See NOTES 1 and 2 to Symbol 3.3-3.
3.3-9		Output with special amplification (drive capability) The symbol ▷ emphasizes the function of amplification. It shall point in the direction of signal flow. The absence of this symbol does not necessarily indicate the absence of special amplification. 5.2 describes its use as a general qualifying symbol for an element. See NOTE 1 to Symbol 3.3-3. NOTE — If this symbol is used with Symbols 3.3-3 through 3.3-8, those symbols are placed between the amplification symbol and the edge of the element.

No	Symbol	Description
3.3-9.5		Input with special amplification (sensitivity) The symbol ▷ emphasizes the function of amplification. It shall point in the direction of signal flow. The absence of this symbol does not necessarily indicate the absence of special amplification. 5.2 describes its use as a general qualifying symbol for an element. Its use at an input, rather than as a general qualifying symbol, shows that the input is unusually sensitive rather than that the output has increased drive capability. NOTE — For use with the hysteresis symbol (Symbol 3.3-2), the line-grouping symbol (Symbol 3.3-37), or dependency notation, see 4.4.3.
3.3-10		Extension input An input of an element to which an extender output may be connected (see Symbol 3.3-11). NOTE — The description that characterizes the relationship between the external logic states of binary variables and their corresponding physical quantities is normally not valid for extension inputs and extender outputs.
3.3-11		Extender output An output of an element that may be connected to the extension input of another combinational element in order to extend the number of inputs to that element. See NOTE to Symbol 3.3-10.
3.3-12		Enable input When this input stands at its internal 1-state, all outputs stand at their normally defined internal logic states and have their normally defined effect on elements or distributed functions that may be connected to the outputs, provided no other inputs or outputs have an overriding and contradicting effect. When this input stands at its internal 0-state, all open-circuit outputs stand at their external high-impedance states, all passive-pulldown outputs stand at their high-impedance L-levels, all passive-pullup outputs stand at their high-impedance H-levels, all 3-state outputs stand at their normally defined internal logic states and at their external high-impedance states, and all other outputs stand at their internal 0-states. In a composite symbol this input affects all outputs that are not shown as internal connections or internal outputs (making use of one of the Symbols 3.2-1 through 3.2-6). See also 4.3.9. The “composite symbol” excludes outputs that are not otherwise shown to be influenced by the unsubdivided portion of the symbol containing the EN Input.

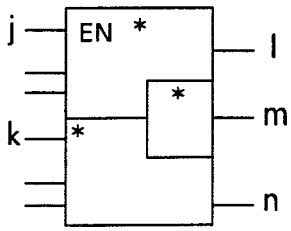
No	Symbol	Description



a affects b
but not c and d.

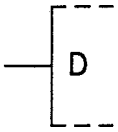


e affects f,
g, and h
but not i.



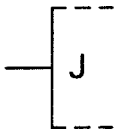
j affects l and m
but not n. If k is
also an EN input,
it will also affect
m as well as n
but not l.

3.3-13



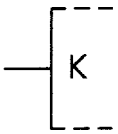
D input
The internal logic state of the D input is stored by the element.
NOTE — This symbol is used only in conjunction with dependency notation.
For example, see Symbol 5.9-2.

3.3-14



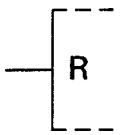
J input of a JK bistable element
When this input takes on its internal 1-state, a 1 is stored by the element. When the input stands at its internal 0-state, it has no effect on the element. Each occurrence of the combination J=K=1 causes a single change of the internal state of the output to its complement.
See 3.3.2.

3.3-15



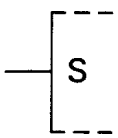
K input of a JK bistable element
When this input takes on its internal 1-state, a 0 is stored by the element. When the input stands at its internal 0-state, it has no effect on the element. Each occurrence of the combination J=K=1 causes a single change of the internal state of the output to its complement.
See 3.3.2.

3.3-16

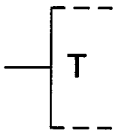
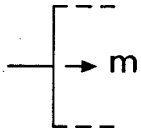
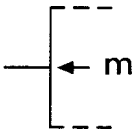
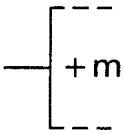
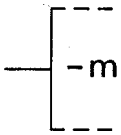
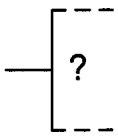
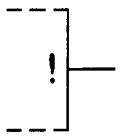


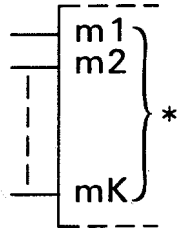
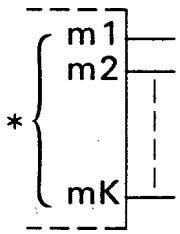
R input
When this input takes on its internal 1-state, a 0 is stored by the element. When the input stands at its internal 0-state, it has no effect on the element. The effect of the combination R=S=1 is not specified by the symbol; this effect may be indicated by means of set/reset dependency (see 4.3.8), if necessary.
See 3.3.2.

3.3-17

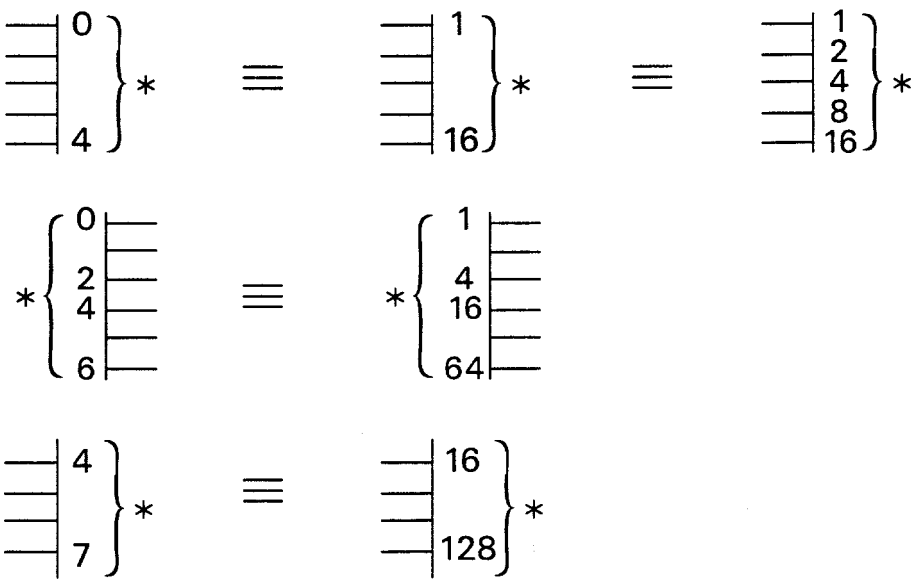


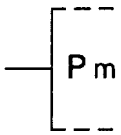
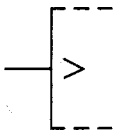
S input
When this input takes on its internal 1-state, a 1 is stored by the element. When the input stands at its internal 0-state, it has no effect on the element. The effect of the combination R=S=1 is not specified by the symbol; this effect may be indicated by means of set/reset dependency (see 4.3.8), if necessary.
See 3.3.2.

No	Symbol	Description
3.3-18		T input of a bistable element Each time this input takes on its internal 1-state, a single change of the internal state of the output to its complement takes place. When the input stands at its internal 0-state, it has no effect on the element. See 3.3.2.
3.3-19		Shifting input of a register Each time an input with a right-pointing arrow takes on its internal 1-state, the information contained in the element will be shifted once m positions from left to right or from top to bottom. When the input stands at its internal 0-state, it has no effect on the element. See 3.3.2. NOTES: 1 — m must be replaced by the relevant value. If m = 1, the 1 may be omitted if no confusion is likely. 2 — All directions above are relative to the orientation of the text within the symbol.
3.3-20		Shifting input of a register Each time an input with a left-pointing arrow takes on its internal 1-state, the information contained in the element will be shifted once m positions from right to left or from bottom to top. When the input stands at its internal 0-state, it has no effect on the element. See 3.3.2, also NOTE 1 to Symbol 3.3-19.
3.3-21		Count-up input Each time this input takes on its internal 1-state, the content of the element is increased once by m. When the input stands at its internal 0-state, it has no effect on the element. See 3.3.2, also NOTE 1 to Symbol 3.3-19.
3.3-22		Count-down input Each time this input takes on its internal 1-state, the content of the element is decreased once by m. When the input stands at its internal 0-state, it has no effect on the element. See 3.3.2, also NOTE 1 to Symbol 3.3-19.
3.3-23		Query input (interrogate) of an associative memory When this input takes on its internal 1-state, an interrogation of the content of the element takes place. When the input stands at its internal 0-state, it has no effect on the element.
3.3-24		Compare output (match) of an associative memory. The internal 1-state at this output indicates a match.

No	Symbol	Description
3.3-25		Bit-grouping symbol for inputs (qualifying symbol for multibit input), general symbol Inputs grouped by this symbol produce a value that is the sum of the individual weights of the inputs standing at their internal 1-states. The individual inputs shall be shown in ascending or descending order by weight. This value can be regarded as: 1)A value on which a mathematical function is performed, or 2)Defining an identifying number in the sense of dependency notation according to 4.4.2, or 3)A value to become the content of the element. The labels m1...mk shall be replaced by the decimal equivalents of the actual weights. If all the weights are powers of 2, m1...mk may be replaced by the exponents of the powers of 2 if no confusion is likely. Labels between m1 and mk may be omitted to the extent that no confusion is likely. The "*" must be replaced by an indication of the operand on which the mathematical function is performed (for example, P or Q), by an indication in the sense of dependency notation (see 4.4.2), by CT, or by an appropriate label in the case of a "gray box" (Section 6). In the case of CT the value is produced by the inputs is the value that is loaded into the element. For example, see Symbol 5.7-5.
3.3-26		Bit-grouping symbol for outputs (qualifying symbol for multibit output), general symbol Outputs grouped by this symbol represent a value that is the sum of the individual weights of the outputs standing at their internal 1-states. The individual outputs shall be shown in ascending or descending order by weight. This value can be regarded as: 1)The result of the performance of a mathematical function, or 2)The value of the content of the element. The labels m1...mk shall be replaced by the decimal equivalents of the actual weights. If all the weights are powers of 2, m1...mk may be replaced by the exponents of the powers of 2 if no confusion is likely. Labels between m1 and mk may be omitted to the extent that no confusion is likely. The "*" must be replaced by an indication of the result of the performance of the mathematical function, by CT, or by an appropriate label in the case of a "gray box" (Section 6). In the case of CT the value represented by the outputs is the actual value of the content of the element. For example, see Symbol 5.13-18.

Illustrations of the concept for Symbols 3.3–25 and 3.3–26



No	Symbol	Description
3.3-27		Operand input of an arithmetic element (P input shown) This input represents one bit of an operand on which one or more mathematical functions are performed. NOTES: 1 — m shall be replaced by the decimal equivalent of the weight of the bit. If the weights of all P inputs of the element are powers of 2 and if no confusion is likely, at each P input m may be replaced by the exponent of the power of 2. 2 — If an operand consists of two or more bits represented by adjacent input lines, the bit-grouping symbol (Symbol 3.3-25) may be used. 3 — Preferred letters for operands are P and Q. When these letters are not suitable or if more than two operands are involved, other characters may be used providing no confusion is likely.
3.3-28		Greater-than input of a magnitude comparator NOTES: 1 — This symbol is intended for use when representing cascaded comparators. For example, see Symbol 5.7-11. 2 — Symbols 3.3-28, 3.3-29, and 3.3-30 may be combined to qualify other inputs of magnitude comparators such as ≥. The symbol ≠ shall be used instead of the combination $\lessgtr$. 3 — Care should be taken that this symbol is not drawn touching the outline to avoid confusion with the dynamic input indicator (Symbol 3.1-9).

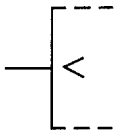
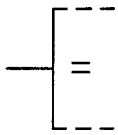
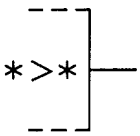
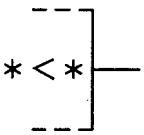
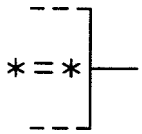
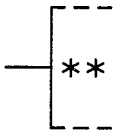
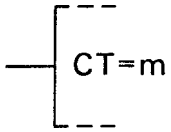
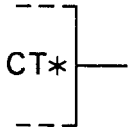
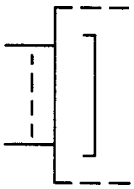
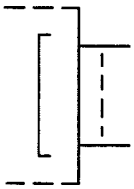
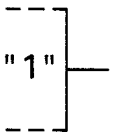
No	Symbol	Description
3.3-29		Less-than input of a magnitude comparator See NOTES 1 and 2 to Symbol 3.3-28.
3.3-30		Equal input of a magnitude comparator See NOTES 1 and 2 to Symbol 3.3-28.
3.3-31		Greater-than output of a magnitude comparator NOTES: 1 — Each "*" must be replaced by the designation of an operand (for example, P or Q). 2 — Symbols 3.3-31, 3.3-32, and 3.3-33 may be combined to qualify outputs such as *≥*. The symbol *≠* shall be used instead of the combination * $\lessgtr$ *. 3 — If this symbol appears in one element of a series of cascaded comparators, the output marked with this symbol is affected not only by the operands, but also by inputs marked with the Symbol 3.3-28, 3.3-29, or 3.3-30.
3.3-32		Less-than output of a magnitude comparator See NOTES to Symbol 3.3-31.
3.3-33		Equal-output of a magnitude comparator See NOTES to Symbol 3.3-31. NOTE — If this symbol is not combined with Symbol 3.3-31 or Symbol 3.3-32, the designations of the operands may be omitted if no confusion is likely.
3.3-34		Symbols for commonly used logic signals (shown at an input) NOTES: 1 — Table Clause lists and describes symbols available for designating commonly used logic signals. 2 — Each "**" symbol must be replaced by an appropriate alphabetic signal designator. 3 — A numerical suffix may be added to this alphabetic designator, if appropriate, to indicate the weight or position of the signal within a group of related signals.

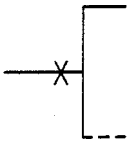
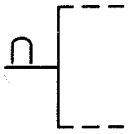
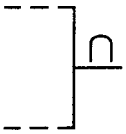
Table 3.1—Commonly used logical signal designations

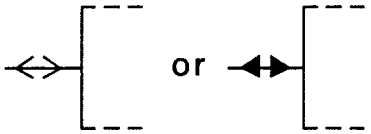
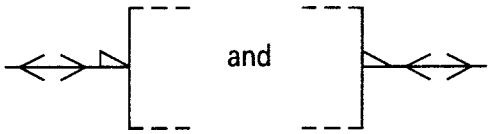
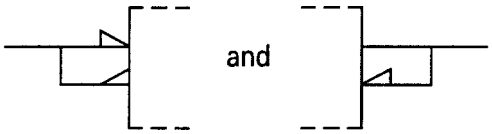
Designation	Input/Output	Description
BI (Borrow input)	Input only	If at its internal 1-state, indicates that a subtraction operation performed by a lower-order arithmetic element produces an arithmetic borrow.
BO (Borrow output)	Output only	If at its internal 1-state, indicates that a subtraction operation performed by an arithmetic element produces an arithmetic borrow (see BI signal above).
BG (Borrow generate)	Output	If at its internal 1-state, indicates that an arithmetic element performing subtraction is in the "Borrow-Generate" state, that is, that the subtrahend applied to the element is larger than the minuend, causing a borrow from that element independent of the state of the BI input to that element.
	Input	If at its internal 1-state, indicates to a borrow-acceleration element that the arithmetic element that produces the BG signal is in the "Borrow-Generate" state (see BG output above). The borrow-acceleration element uses its BG, BP, and BI input signals to determine, with reduced propagation delays, the states of the arithmetic borrow signals for a group of arithmetic elements performing binary subtraction.
BP (Borrow propagate)	Output	If at its internal 1-state, indicates that an arithmetic element performing subtraction is in the "Borrow-Propagate" state, that is, that the subtrahend and the minuend applied to the element are equal in value, so that the BO output will stand at its internal 1-state if and only if the BI input is at its internal 1-state.
	Input	If at its internal 1-state, indicates to a borrow-acceleration element that the arithmetic element that produces the BP signal is in the "Borrow-Propagate" state.
CI (Carry input)	Input only	If at its internal 1-state, indicates that an addition operation performed by a lower-order arithmetic element produces an arithmetic carry.
CO (Carry output)	Output only	If at its internal 1-state, indicates that an addition operation performed by an arithmetic element produces an arithmetic carry (see CI signal above).
CG (Carry generate)	Output	If at its internal 1-state, indicates that an arithmetic element performing addition is in the "Carry-Generate" state, that is, that the sum of its addends is sufficiently large to cause a carry from the element independent of the state of the CI input to that element.
	Input	If at its internal 1-state, indicates to a carry-acceleration element whether or not the arithmetic element that produces the CG signal is in the "Carry-Generate" state (see CG output above). The carry-acceleration element uses its CG, CP, and CI input signals to determine with reduced propagation delays, the states of the arithmetic carry signals for a group of arithmetic elements performing addition.
CP (Carry propagate)	Output	If at its internal 1-state, indicates that an arithmetic element performing addition is in the "Carry-Propagate" state, that is, that the sum of its addends is one less than the value at which the element produces an output carry. As a result, the CO output will stand at its internal 1-state if and only if its CI input is at its internal 1-state.
	Input	If at its internal 1-state, indicates to a carry-acceleration element that the arithmetic element that produces the CP signal is in the "Carry-Propagate" state.

No	Symbol	Description
3.3-35		Content input m shall be replaced by an appropriate indication of the value of the content of the element (for example, a counter) that results whenever this input takes on its internal 1-state. For example, see Symbol 5.13-18.
3.3-36		Content output "*" shall be replaced by an appropriate indication of those values of the content of the element (for example, a counter) for which the output stands at its internal 1-state. For example, see Symbol 5.13-14.
3.3-37		Line-grouping symbol for inputs This symbol indicates that two or more terminals are needed to implement a single logic input. For example, see Symbol 5.1-22. NOTES: 1 — The description that characterizes the relationship between the external logic states of binary variables and their corresponding physical quantities may not be valid for lines grouped by symbol. 2 — For use with the hysteresis symbol (Symbol 3.3-2), the special amplification symbol (Symbol 3.3-9.5), or dependency notation, see 4.4.3.
3.3-38		Line-grouping symbol for outputs This symbol indicates that two or more terminals are needed to implement a single logic output. See NOTE to Symbol 3.3-37. For example, see Symbol 5.1-23.
3.3-39		Fixed-mode input This representation may be used to identify an input that always must be in the internal 1-state for the element to perform the function indicated by the complete symbol. A fixed-mode input must neither be affected by dependency notation nor have other functions. See Symbols 5.13-9 and 5.13-10.
3.3-40		Fixed-state output This representation may be used to identify an output that always stands at its internal 1-state. A fixed-state output must neither be affected by dependency notation nor have other functions. For example, see Symbol 5.4-10.

3.4 Subsidiary (formerly nonlogic) connections and signal-flow indicators

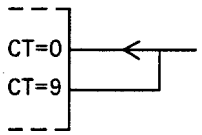
In principle, the direction of signal flow within a symbol is from left to right. If the direction of signal flow on any input or output line is not obvious, that line should be marked with an arrowhead (Symbol 3.4-4) pointing in the direction of signal flow. Such an arrowhead shall not touch the outline or any qualifying symbol.

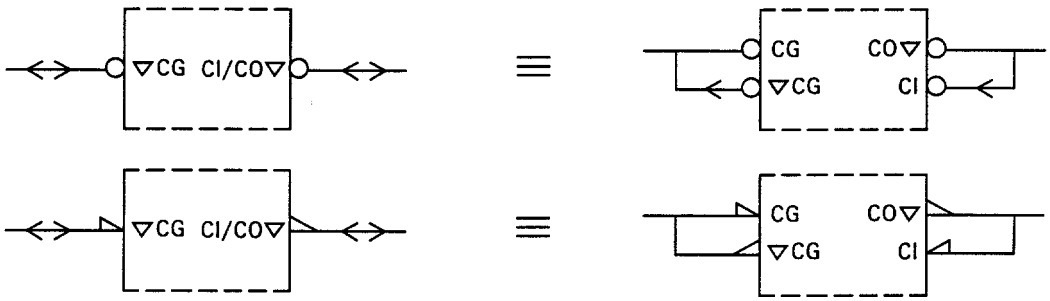
No	Symbol	Description
3.4-1		Subsidiary connection This symbol may be used to designate an input supplying power to the device or a connection, the knowledge of whose level is not important to understand the function of the element and the circuit (such as a connection to an external supplementary resistor or capacitor). NOTES: 1 — This symbol may be shown on any side of the outline (including the top and bottom). 2 — Additional information associated with subsidiary connections may be shown without brackets inside the outline. See Symbol 5.12-7. 3 — In a symbol representing an array of elements, any subsidiary connection shown at either end of the array is considered to be common to all elements of the array. See Symbol 5.12-8.
3.4-2		Analog input NOTE — This symbol shall be used only when it is necessary to distinguish analog signals. See NOTE 2 to Symbol 3.4-1.
3.4-3		Analog output See NOTE 2 to Symbol 3.4-1 and NOTE to Symbol 3.4-2.
3.4-4		Right-to-left signal flow, shown on the left-hand side

No	Symbol	Description
3.4-5		Bidirectional signal flow, shown on the left-hand side For an example of use, see Symbol 5.2-8. NOTES: 1 — On a connecting line with bidirectional signal flow, any symbol for logic polarity should point to the right or down, e.g.,  2 — In some cases, use of the symbol for logic polarity makes the use of the symbol for bidirectional signal flow unnecessary, e.g.,  3 — Bidirectional signal flow shall not be shown on a single input-output line: —if the negation or polarity indications are different for the input and output functions (except as permitted by 2.3.1.2), or —if the labels associated with the input and output functions are different and confusion is likely regarding which portion of the labels applies to the input and which to the output.

Illustrations of the concept

In this case, bidirectional signal flow may not be shown on a single line because confusion would result.





4. Dependency notation

4.1 General explanation

Dependency notation is a means of denoting the relationships between inputs, outputs, or inputs and outputs, without actually showing all the elements and interconnections involved.

Apart from its use in complex elements, dependency notation should not be used to replace the symbols for combinational elements.

The information provided by dependency notation supplements that provided by the qualifying symbols for an element's function.

In the convention for dependency notation, use will be made of the terms "affecting" and "affected." In the case where it is not evident which inputs must be considered as being the affecting or the affected ones (for example, if they stand in an AND relationship), the choice may be made in any convenient way.

Because of the existence of feedback in some complex elements, outputs sometimes have an effect on inputs and other outputs. For the sake of simplicity, the text of the following sections often refers to "affecting inputs" only, but it should be understood that the recommended notation applies to affecting outputs also.

4.2 Summary of types of dependencies

Eleven types of dependencies are defined.

AND, OR, and Negate dependencies are used to denote Boolean relationships between inputs and outputs, in any combination.

Interconnection dependency is used to indicate connections inside the symbol.

Transmission dependency is used to indicate controlled transmission paths between affected ports.

Control dependency is used to identify a timing input or a clock input of a sequential element and to indicate which inputs are controlled by it.

Set and reset dependencies are used to specify the internal logic states of an RS bistable element when the R and S inputs both stand at their internal 1-states.

Enable dependency is used to identify an enable input and to indicate which inputs and outputs are controlled by it (for example, which outputs take on their high-impedance state).

Mode dependency is used to identify an input that selects the mode of operation of an element and to indicate the inputs and outputs depending on that mode.

Address dependency is used to identify the address inputs of a memory.

Table Clause lists the various dependencies and summarizes their effects. More detailed definitions appear in the following sections, together with illustrations of the concept. In this Table the word "action" implies:

1) That affected inputs will have their normally defined effect on the function of the element, and
2) That affected outputs will take on the internal logic states determined by the function of the element.

Table 4.1—Summary of dependency notation

		Effect on internal logic state of, or action of, the affected input or output:	
Type of dependency	**Letter***	**Affecting input at its 1-state**	**Affecting input at its 0-state**
ADDRESS	A	Permits action (address selected)	Prevents action (address not selected)
CONTROL	C	Permits action	Prevents action
ENABLE	EN	Permits action	1)Prevents action of affected inputs 2)Imposes external high-impedance state on open-circuit and 3-state outputs (internal state of 3-state output is unaffected) 3)Imposes high-impedance L-level on passive-pulldown outputs and high-impedance H-level on passive-pullup outputs 4)Imposes 0-state on other outputs
AND	G	Does not alter state (permits action)	Imposes 0-state
MODE	M	Permits action (mode selected)	Prevents action (mode not selected)
NEGATE	N	Complements state	Does not alter state (no effect)
RESET	R	Affected output reacts as it would to S=0, R=1	No effect
SET	S	Affected output reacts as it would to S=1, R=0	No effect
OR	V	Imposes 1-state	Does not alter state (permits action)
TRANSMISSION	X	Transmission path established	No transmission path established
INTER-CONNECTION	Z	Imposes 1-state	Imposes 0-state

*These letters appear at the affecting input (or output) and are followed by a number represented in the general cases in Section 4 by the letter "m." Each input or output affected by that input (or output) is labeled with that same number.

4.2.1 Comparison of C, EN, and M effects on inputs

Cm, ENm, and Mm inputs all have the same effect on affected inputs. However, their intended applications are different:

1) Cm should be used to identify an input that produces action, for example, the edge-triggered clock of a bistable circuit or the level-operated data enable of a transparent latch
2) ENm should be used to identify an input that produces a single preparatory effect
3) Mm should be used to identify one or more inputs that singly or together produce alternative preparatory effects

4.3 Application of dependency notation

4.3.1 General techniques and conventions

Dependency notation usually defines relationships between internal logic states. However, in the case of open-circuit outputs, passive-pullup and passive-pulldown outputs, and 3-state outputs (3.3-3 through 3.3-8), enable dependency (see 4.3.9) defines relationships between the internal logic states of affecting inputs and the external states of affected outputs.

Application of dependency notation is accomplished by:

1) Labelling the input affecting other inputs or outputs with the letter symbol denoting the relationship involved followed by an appropriately chosen identifying number, and
2) Labelling each input or output affected by the affecting input with that same number.

If it is the complement of the internal logic state of the affecting input or output that does the affecting, a bar is placed over the identifying number at the affected input or output (see Symbol 5.9-12). For a technique avoiding the use of the bar, see the note with symbol 5.1-5.

• If only an in-line notation can be used, the symbol ¬ shall be used instead of the negation bar, and it shall precede the identifying number. The tilde (~) may be substituted on computer systems that do not have the logic negation symbol as part of their character set.

If the affected input or output requires a label to denote its function, this label shall be prefixed by the identifying number of the affecting input.

If an input or output is affected by more than one affecting input, the identifying numbers of each of the affecting inputs shall appear in the label of the affected one separated by commas. The left-to-right reading order of these identifying numbers is the same as the sequence of the affecting relationships (see also 4.4.3 and 4.4.4).

Two affecting inputs labelled with different letters shall not have the same identifying number, unless one of the letters is A (see 4.3.11).

If two affecting inputs have the same letter and the same identifying number, they stand in an OR relationship to each other.

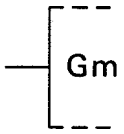
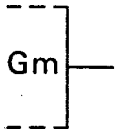
If the labels denoting the functions of affected inputs or outputs must be numbers (for example, outputs of a coder), the identifying numbers to be associated with both affecting inputs and affected inputs or outputs shall be replaced by another character selected to avoid ambiguity (for example, Greek letters as in Symbol 5.4-6).

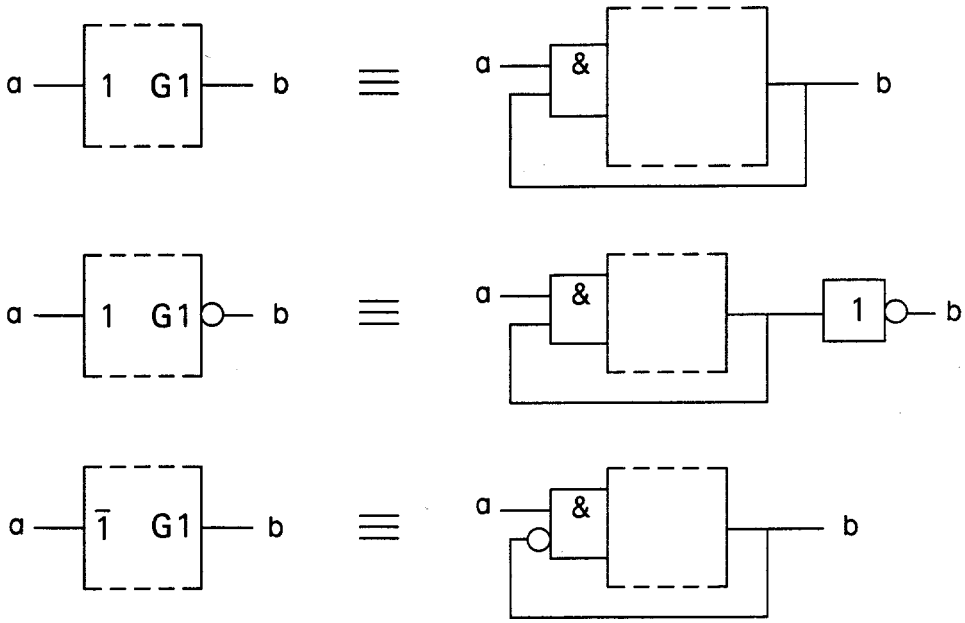
An affecting input affects only the corresponding affected inputs and outputs of the symbol.

4.3.2 AND dependency (G dependency)

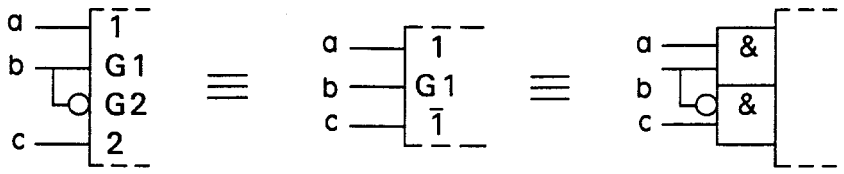
The symbol denoting AND dependency is the letter G.

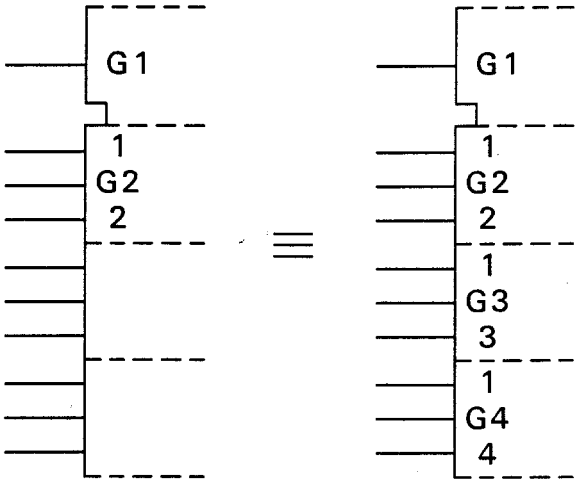
Each input or output affected by a Gm input or Gm output stands in an AND relationship with this Gm input or Gm output.

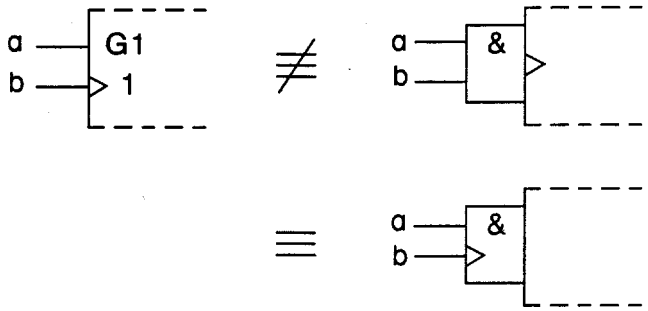
No	Symbol	Description
4.3.2-1		Gm input
4.3.2-2		Gm output When a Gm input or Gm output stands at its internal 1-state, all inputs and outputs affected by this Gm input or Gm output stand at their normally defined internal logic states. When a Gm input or Gm output stands at its internal 0-state, all inputs and outputs affected by this Gm input or Gm output stand at their internal 0-states. NOTE — m must be replaced by the identifying number.



Illustrations of the concept



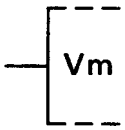
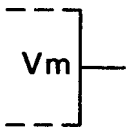


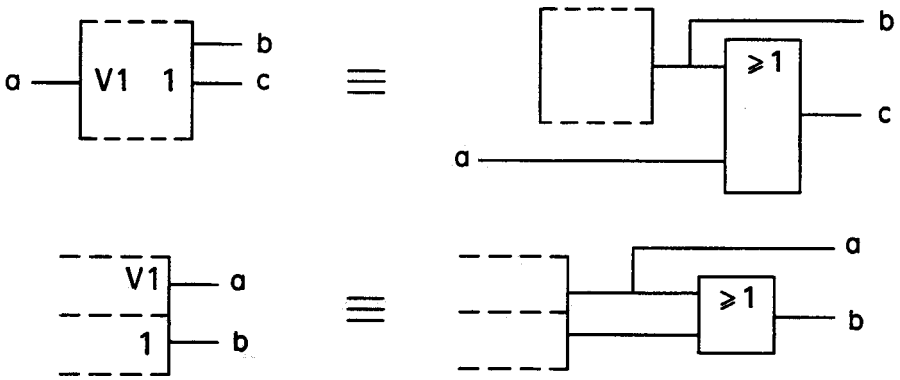


4.3.3 OR dependency (V dependency)

The symbol denoting OR dependency is the letter V.

Each input or output affected by a Vm input or Vm output stands in an OR relationship with this Vm input or Vm output.

No	Symbol	Description
4.3.3-1		Vm input
4.3.3-2		Vm output When a Vm input or Vm output stands at its internal 1-state, all inputs and outputs affected by this Vm input or Vm output stand at their internal 1-states. When a Vm input or Vm output stands at its internal 0-state, all inputs and outputs affected by this Vm input or Vm output stand at their normally defined internal logic states. See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.

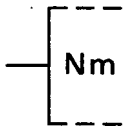
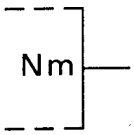


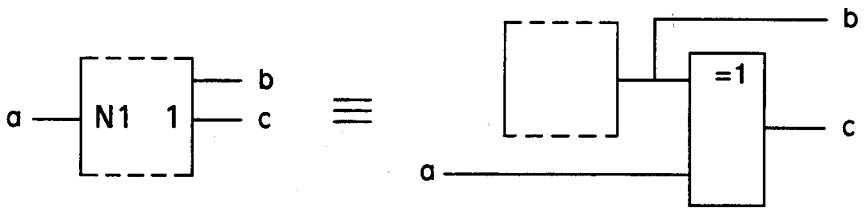
Illustrations of the concept

4.3.4 Negate dependency (N dependency)

The symbol denoting negate dependency is the letter N.

Each input or output affected by an Nm input or Nm output stands in an Exclusive-OR relationship with this Nm input or Nm output.

No	Symbol	Description
4.3.4-1		Nm input
4.3.4-2		Nm output When an Nm input or Nm output stands at its internal 1-state, the internal logic state of each input and each output affected by this Nm input or Nm output is the complement of the normally defined internal logic state of that input or output. When an Nm input or Nm output stands at its internal 0-state, all inputs and outputs affected by this Nm input or Nm output stand at their normally defined internal logic states. See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.



If $a = 0$, then $c = b$
If $a = 1$, then $c = \bar{b}$

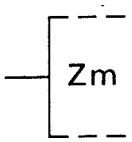
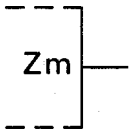
Illustrations of the concept

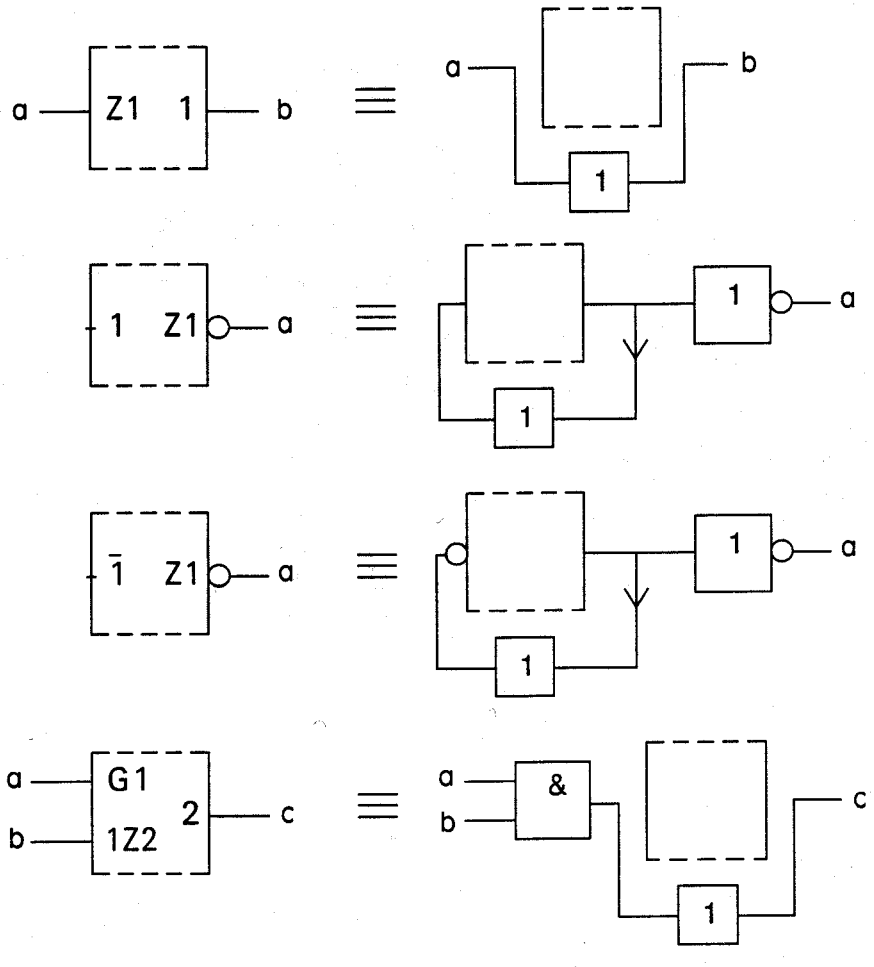
4.3.5 Interconnection dependency (z dependency)

The symbol denoting interconnection dependency is the letter Z.

Interconnection dependency is used to indicate the existence of internal logic connections between inputs, outputs, internal inputs, and internal outputs, in any combination (see Symbol 5.1-29).

The internal logic state of an input or an output affected by a Zm input or Zm output is identical to the internal logic state of its affecting Zm input or Zm output, unless modified by additional dependency notation.

No	Symbol	Description
4.3.5-1		Zm input
4.3.5-2		Zm output When a Zm input or Zm output stands at its internal 1-state, all inputs and outputs affected by this Zm input or Zm output stand at their internal 1-states, unless modified by additional dependency notation. When a Zm input or Zm output stands at its internal 0-state, all inputs and outputs affected by this Zm input or Zm output stand at their internal 0-states, unless modified by additional dependency notation. See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.

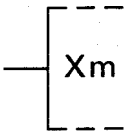
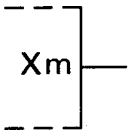


Illustrations of the concept

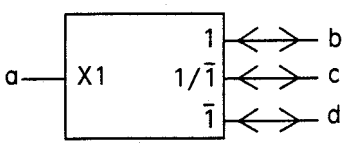
4.3.6 Transmission dependency (X dependency)

The symbol denoting transmission dependency is the letter X.

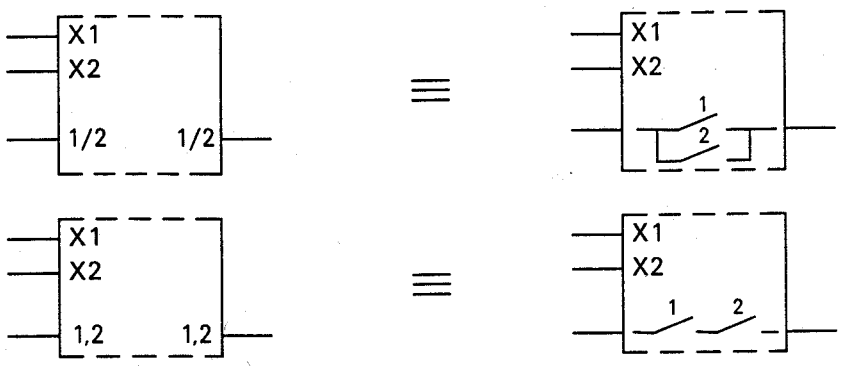
Transmission dependency is used to indicate controlled transmission paths between affected ports (inputs, outputs, input/outputs, or a combination of these). Unless otherwise indicated, the transmission paths are bidirectional. Transmission dependency provides an alternative way of symbolizing simple analog switches such as those shown in IEEE Std 315, and it enables more complicated devices to be shown in a concise manner.

No	Symbol	Description
4.3.6-1		Xm input
4.3.6-2		Xm output If an Xm-input [Xm-output] stands at its internal 1-state, a transmission path is established to which all ports affected by this input [output] are connected. However, if a port is affected by two or more Xm- inputs and/or Xm-outputs whose identifying numbers are separated by commas, then the port is connected to the transmission paths established by these Xm-inputs and/or Xm-outputs if and only if all these affecting inputs and/or outputs stand at their internal 1-states. All ports connected to a transmission path stand at the same analog signal level or internal logic state unless modified by additional notation, e.g., dependency notation. If an Xm-input [Xm-output] stands at its internal 0-state, no transmission paths are established by this input or output. If an Xm-input [Xm-output] is modified to have no effect on the function of the element, there is no transmission path established by that input or output. See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.

Illustrations of the concept

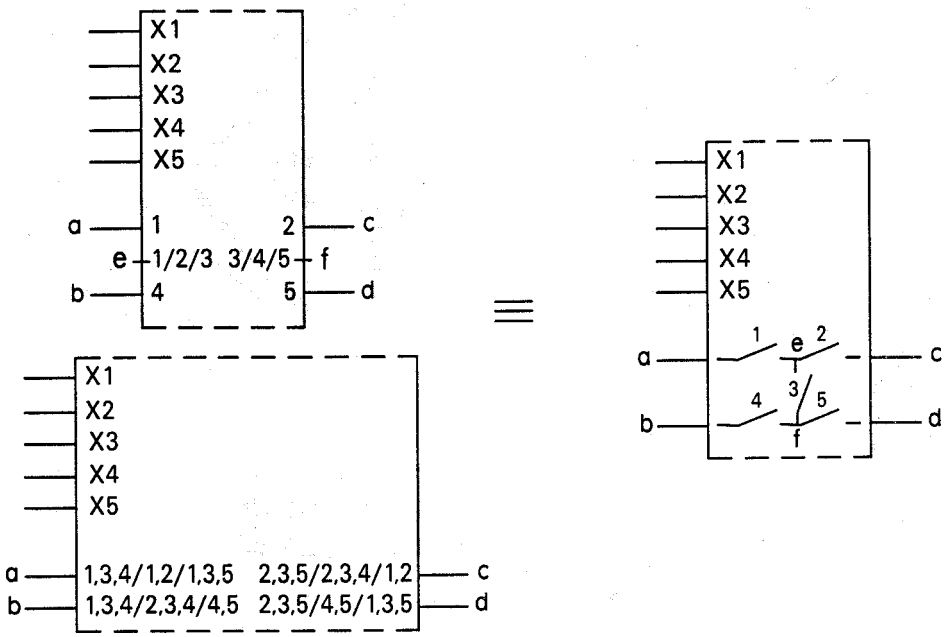


If input *a* stands at its internal 1-state, there is a bidirectional transmission path between *b* and *c*.
If input *a* stands at its internal 0-state, there is a bidirectional transmission path between *c* and *d*.



Illustrations of the concept

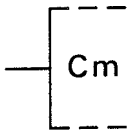
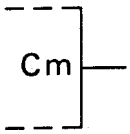
The use of virtual inputs and outputs can simplify an otherwise complicated string of dependency notation.

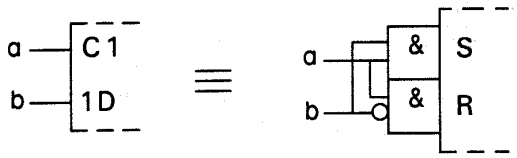


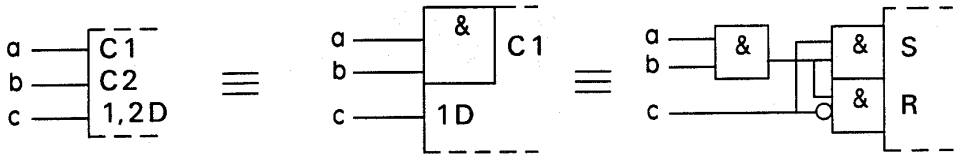
4.3.7 Control dependency (C dependency)

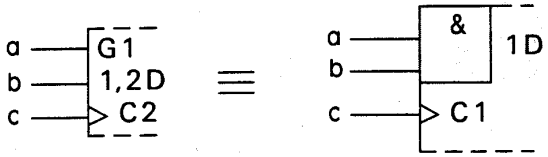
The symbol denoting control dependency is the letter C.

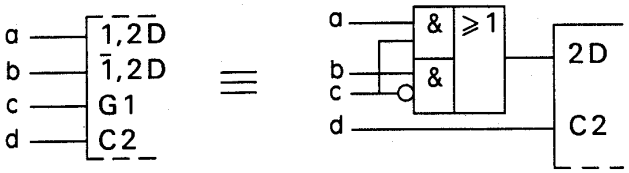
Control dependency shall be used for sequential elements only and may imply more than a simple AND relationship. It identifies an input that produces action, for example, the edge-triggered clock of a bistable circuit or the level-operated data enable of a transparent latch. For a comparison of C, EN, and M dependency effects on inputs, see 4.2.1.

No	Symbol	Description
4.3.7-1		Cm input
4.3.7-2		Cm output When a Cm input or Cm output stands at its internal 1-state, the inputs affected by this Cm input or Cm output have their normally defined effect on the function of the element. When a Cm input or Cm output stands at its internal 0-state, the inputs affected by this Cm input or Cm output have no effect on the function of the element. See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.

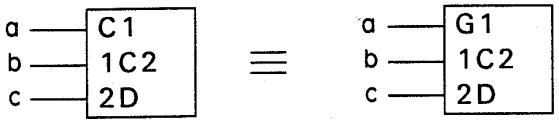








NOTE—Use has been made of rule 3.3.1.



This illustration has been included only for explanatory purposes; the use of a Cm input to affect a second Cm input is not recommended.

Illustrations of the concept

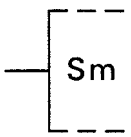
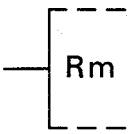
4.3.8 Set and reset dependencies (S and R dependencies)

The symbol denoting set dependency is the letter S.

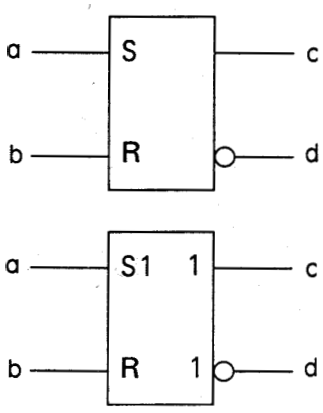
The symbol denoting reset dependency is the letter R.

Set and reset dependencies are used if it is necessary to specify the effect of the combination R=S=1 on a bistable element. These dependencies should not be used if such specification is not necessary.

Affecting Sm and Rm inputs can affect outputs only.

No	Symbol	Description
4.3.8-1		Sm input When an Sm input stands at its internal 1-state, the outputs affected by this Sm input will take on the internal logic states they normally would take on for the combination S=1, R=0, regardless of the state of any R input. When an Sm input stands at its internal 0-state, it has no effect. See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.
4.3.8-2		Rm input When an Rm input stands at its internal 1-state, the outputs affected by this Rm input will take on the internal logic states they normally would take on for the combination S=0, R=1, regardless of the state of any S input. When an Rm input stands at its internal 0-state, it has no effect. See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.

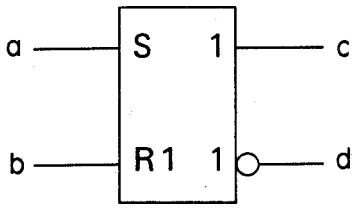
In the illustrations below the truth tables refer to *external* logic states.



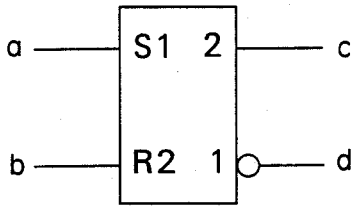
a	b	c	d
0	0	No change	
0	1	0	1
1	0	1	0
1	1	Not specified	

a	b	c	d
0	0	No change	
0	1	0	1
1	0	1	0
1	1	1	0

Illustrations of the concept

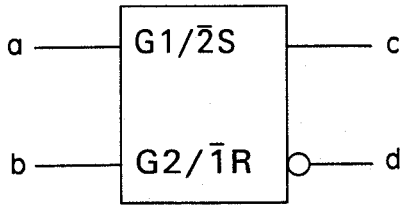


a	b	c	d
0	0	No change	
0	1	0	1
1	0	1	0
1	1	0	1



a	b	c	d
0	0	No change	
0	1	0	1
1	0	1	0
1	1	0	0

NOTE—This noncomplementary output pattern is only pseudo stable. The simultaneous return of *a* and *b* to 0 produces an unforeseeable stable and complementary output pattern.



a	b	c	d
0	0	No change	
0	1	0	1
1	0	1	0
1	1	No change	

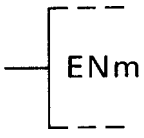
For the use of the solidus, see 4.4.3.

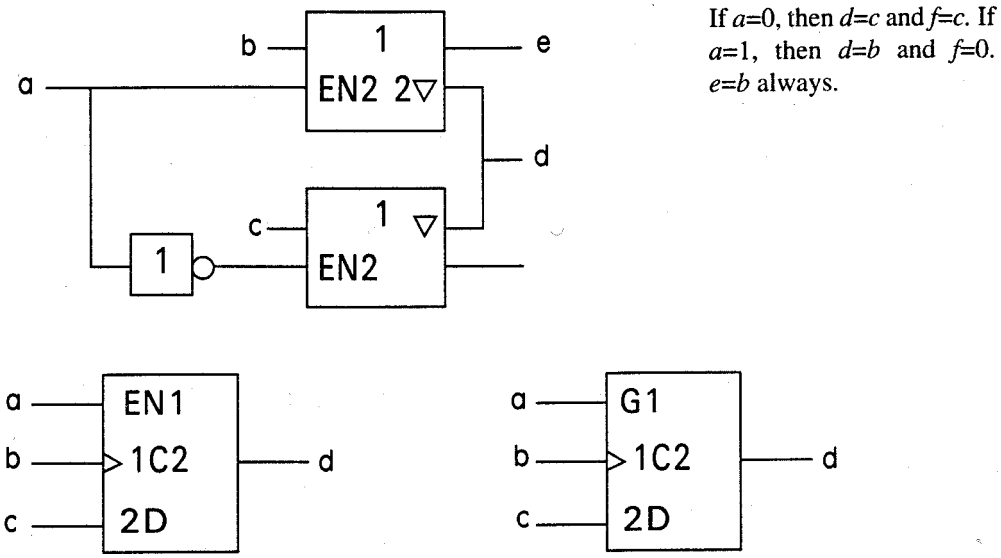
NOTE—This example does not make use of S and R dependencies but completes the set of alternatives to the unspecified case and shows that S and R dependencies cannot affect inputs.

4.3.9 Enable dependency (EN dependency)

The symbol denoting enable dependency is the combination of letters EN.

Enable dependency is used to indicate an Enable input that does not necessarily affect all outputs of an element. It can also be used when one or more inputs of an element are affected. For a comparison of C, EN, and M dependency effects on inputs, see 4.2.1.

No	Symbol	Description
4.3.9-1		ENm input The effect of this input on its affected outputs is the same as that of an EN input (Symbol 3.3-12). The effect of this input on its affected inputs is the same as that of a Cm input (Symbol 4.3.7-1) or an Mm input (Symbol 4.3.10-1). See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.



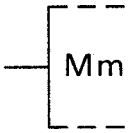
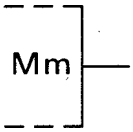
If a=0, then d=c and f=c. If a=1, then d=b and f=0. e=b always.

Illustration of the concept

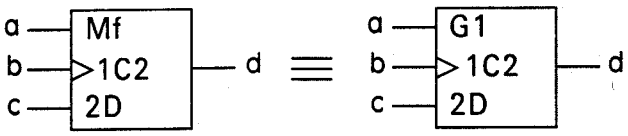
4.3.10 Mode dependency (M dependency)

The symbol denoting mode dependency is the letter M.

Mode dependency is used to indicate that the effects of particular inputs and outputs of an element depend on the mode in which the element is operating. For a comparison of C, EN, and M dependency effects on inputs, see 4.2.1.

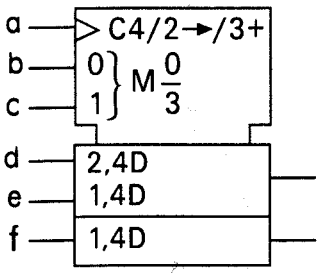
No	Symbol	Description
4.3.10-1		Mm input
4.3.10-2		Mm output If an Mm-input (Mm-output) stands at its internal 1-state, any input affected by this Mm-input (Mm-output) has its normally defined effect on the function of the element, and the outputs affected by this Mm-input (Mm-output) stand at their normally defined internal logic states, i.e., the inputs and outputs are enabled. If an Mm-input (Mm-output) stands at its internal 0-state, its effect on inputs and outputs is as follows: —Any input affected by this Mm-input (Mm-output) has no effect on the function of the element. —If an affected input has several sets of labels separated by solidi, any set containing the identifying number of the Mm-input (Mm-output) has no effect and is to be ignored. This represents disabling some of the functions of a multi-function input. —At each output affected by this Mm-input (Mm-output), any set of labels containing the identifying number of that Mm-input (Mm-output) has no effect and is to be ignored. —If an output has several sets of labels separated by solidi (see 4.4.4), any set containing the identifying number of this Mm-input (Mm-output) is to be ignored. This represents disabling or selection of some of the functions of a multi-function output, or modifying some of the characteristics or dependent relationships of the output. See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.

In complex elements with a large number of different modes, application of the convention for Mode dependency may lead to a very extended labeling. In such cases the inputs and outputs affected by an affecting Mm input are simply labeled with the letter M, but then the diagram containing the symbol shall also contain either a table in which the effects of these inputs in the different modes are clearly explained or a statement as to where such a table is to be found. If no confusion is likely, these letters M may be omitted.



M dependency affecting inputs:

For the use of the solidus and the bit-grouping symbol see 4.4.



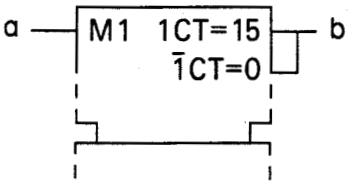
Note that all operations are synchronous.

In mode 0 (b=0, c=0), the outputs remain at their existing states because none of the inputs has an effect.

In mode 1 (b=1, c=0), parallel loading takes place through inputs e and f.

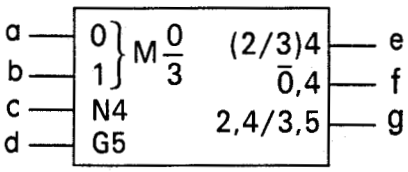
In mode 2 (b=0, c=1), shifting down and serial loading through input d take place.

In mode 3 (b=c=1), counting up by increment of 1 per clock pulse takes place.



Determining the function of an output:

If input a stands at its internal 1-state establishing mode 1, output b will stand at its internal 1-state when the content of the register equals 15. If input a stands at its internal 0-state, output b will stand at its internal 1-state when the content of the register equals 0.



Modifying dependent relationships of outputs:

At output e the label set causing negation (if c = 1) is effective only in modes 2 and 3. In modes 0 and 1 this output stands at its normally defined state as if it had no labels.

At output f the label set has effect when the mode is not 0 so output f is negated (if c = 1) in modes 1, 2 and 3. In mode 0 the label set has no effect so the output stands at its normally defined state. In this example 0̄,4 is equivalent to (1/2/3)4.

At output g there are two label sets. The first set, causing negation (if c = 1), is effective only in mode 2. The second set, subjecting g to AND dependency on d, has effect only in mode 3.

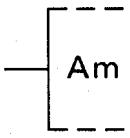
Note that in mode 0 none of the dependency relationships has any effect on the outputs, so e, f, and g will all stand at the same state.

Illustration of the concept

4.3.11 Address dependency (A dependency)

The symbol denoting address dependency is the letter A.

Address dependency provides a clear representation of those elements, particularly memories, that use address control inputs to select specified sections of a multi-dimensional array. Address dependency allows a symbolic representation of only a single general case of the sections of the array, rather than requiring a symbolic representation of the entire array.

No	Symbol	Description
4.3.11-1		Am input When this input stands at its internal 1-state, the inputs affected by this input (that is, the inputs of the section of the array selected by this input) have their normally defined effect on the elements of the selected section. Also, the internal logic states of the outputs affected by this input (that is, the outputs of the selected section) have their normal effect on the OR functions (or the indicated functions) determining the internal logic states of the outputs of the array. When the input stands at its internal 0-state, the inputs affected by this input (that is, the inputs of the section selected by this input) have no effect on the elements of this section. Also, the outputs affected by this input (that is, the outputs of the section selected by this input) have no effect on the outputs of the array. See the NOTE to Symbols 4.3.2-1 and 4.3.2-2.

An affecting address input is labelled with the letter A followed by an identifying number that corresponds to the address of the particular section of the array selected by this input.

Within the general section presented by the symbol, inputs and outputs affected by an Am input are labelled with the letter A, which stands for the identifying numbers, that is, the addresses, of the particular sections. This letter A is subject to the rules of dependency notation concerning identifying numbers associated with affected inputs and outputs.

The identifying numbers of affecting Address inputs need not necessarily differ from those of other affecting dependency inputs (for cxample, G, V, N, ...), because in the general section represented by the symbol they are replaced by the letter A.

Inputs that are not affected by any affecting Address input have their normally defined effect on all sections of the array, whereas inputs affected by an Address input have their normally defined effect only on the section selected by that Address input.

If the address structure allows only one section of an element to be selected at a time, then an affected input of that element is an input to only the selected section, and an affected output of that element is an output of only the selected section.

For the use of the bit-grouping symbol, see 4.4.2.

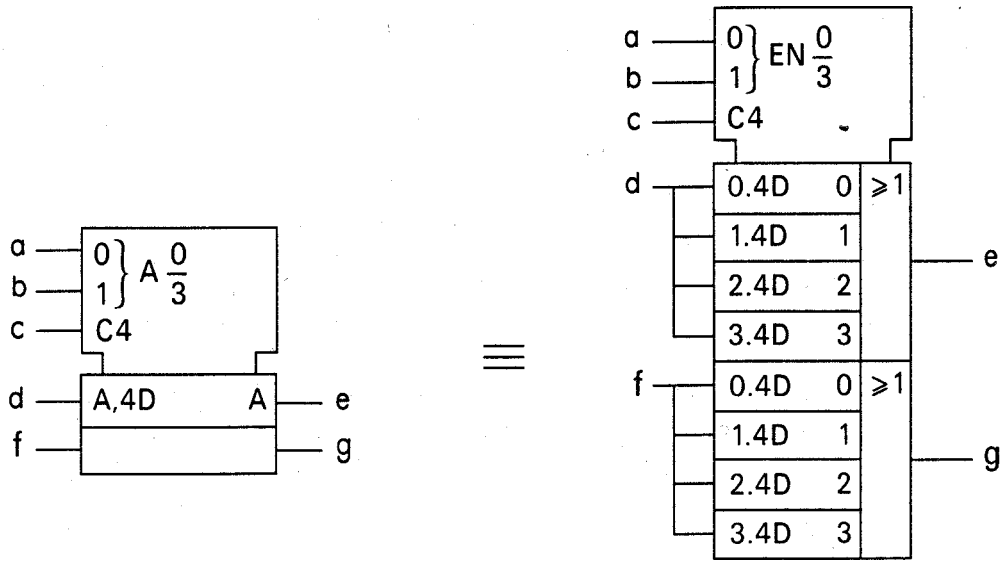
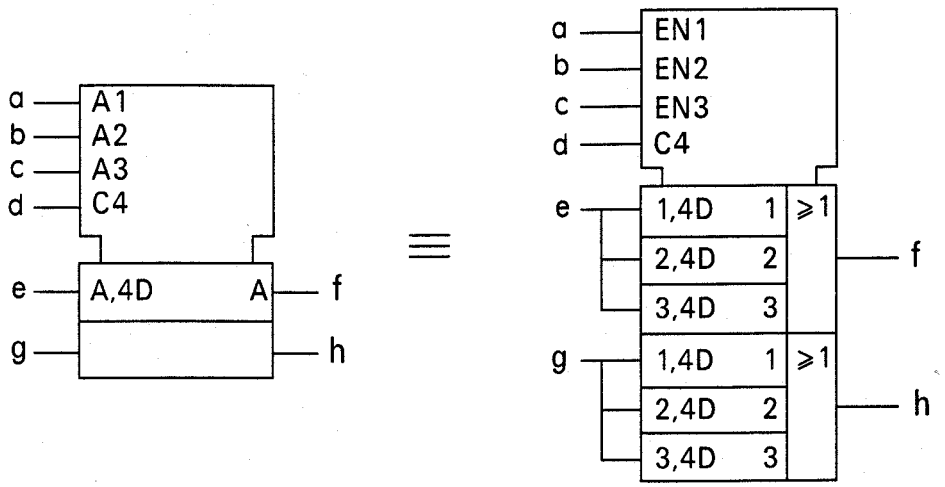


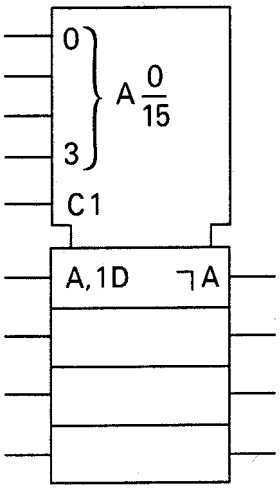
Illustration of the concept

If the address structure allows more than one section of an element to be selected at a time, then an affected input of that element is an input to all of the selected sections, and an affected output of that element is the OR function (unless otherwise noted) of the outputs of all the selected sections.



Illustrations of the concept

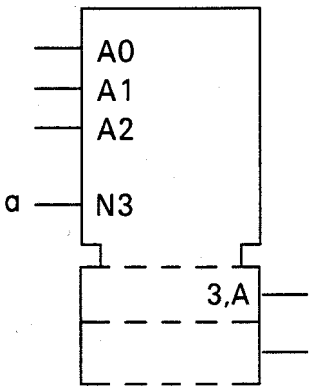
The following example shows an array of 16 sections, each composed of four pulse-triggered D flip-flops (see 5.9).



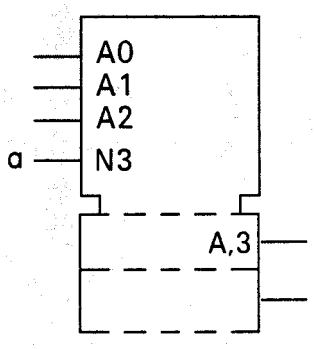
If an output label indicates that the associated output is an open-circuit, passive-pulldown, passive-pullup, 3-state, or specially amplified output, then this refers to the output of the array and not to the individual sections of the array.

If an output affected by an Am input also has other labels, then the labels preceding the letter A affect the output of the selected section and the labels placed behind the letter A affect the output of the array, that is, after the application of the OR function (or the indicated function) to the corresponding outputs of the selected sections of the array.

In the following example if *a* = 1 then the internal logic state of each output is the result of the OR function of the complemented states of the outputs of the selected sections.



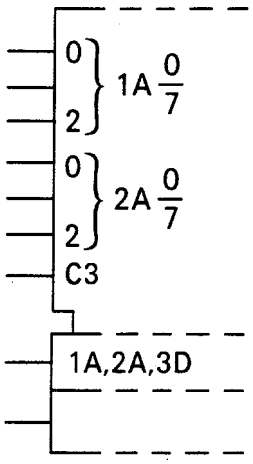
In the following example, if *a* = 1 then the internal logic state of each output is the complement of the OR function of the states of the outputs of the selected sections.



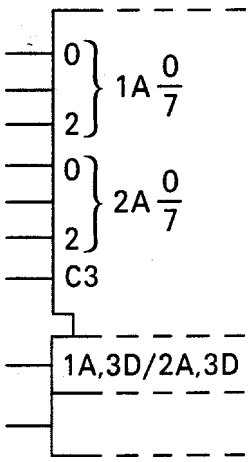
If there are several sets of affecting Am inputs for the purpose of independent and possible simultaneous access to sections of the array, then the letter A is modified to 1A, 2A, ... Because they have access to the same sections of the array, these sets of Am inputs may have the same identifying numbers.

Two affecting address inputs having the same identifying number stand in no fixed relationship with each other nor to any affecting dependency input (for example, G, V, N ...) having the same identifying number.

In the following example a particular section is selected if it is selected by *both* sets of Am inputs.



In the following example a particular section is selected if it is selected either by *one* or by *both* sets of Am inputs.



4.4 Special techniques used in dependency notation

4.4.1 Use of a coder to produce affecting inputs

It often occurs that a set of affecting inputs is produced by decoding the signals on certain inputs to an element. In such a case, the symbol for a coder (see 5.4) may be used as an embedded symbol.

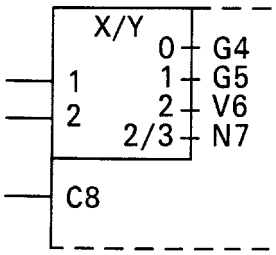


Illustration of the concept

• NOTE—Alternatively, the general qualifying symhol BIN/4 may be used instead of X/Y in this figure. (See 5.4.1.1 and 5.4.1.2.)

If all affecting inputs produced by a coder are of the same type and their identifying numbers correspond with the numbers shown at the outputs of the coder, the Y in the qualifying symbol X/Y may be replaced by the letter denoting the type of dependency. The indications of the affecting inputs should then be omitted.

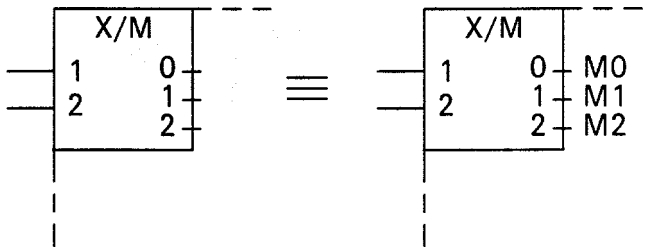
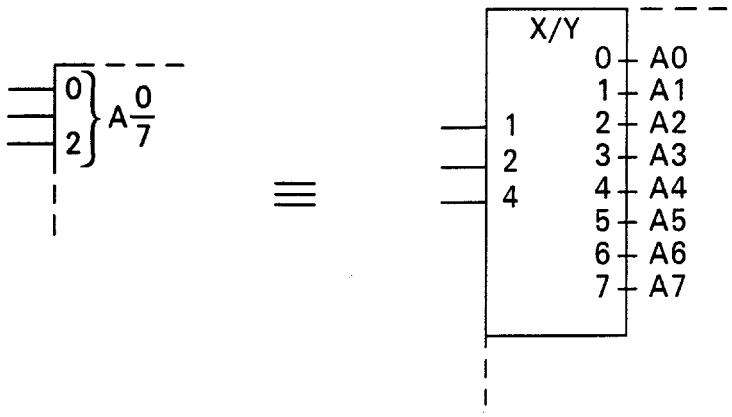


Illustration of the concept

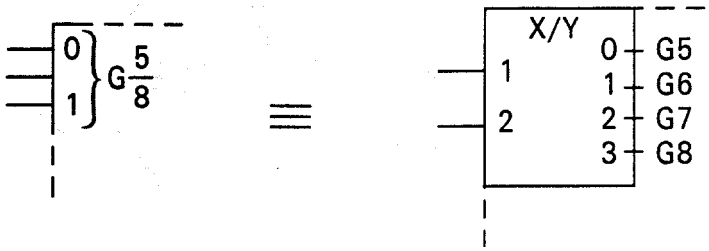
• NOTE—Alternatively, the general qualifying symbol BIN/4 may be used instead of X/Y in this figure. (See 5.4.1.1 and 5.4.1.2.)

4.4.2 Use of bit-grouping to produce affecting inputs

If all affecting inputs produced by a coder are of the same type and have consecutive identifying numbers (not necessarily corresponding with the numbers that would have been shown at the outputs of the coder), the bit-grouping symbol (Symbol 3.3-25) can be used. In this case, the * shall be replaced by the letter denoting the type of dependency followed by $\frac{m1}{m2}$. The m1 shall be replaced by the smallest identifying number, and the m2 shall be replaced by the largest. The range of the identifying numbers (m2 - m1 + 1) must equal the number of outputs of the coder.



• NOTE—Alternatively, the general qualifying symbol BIN/OCT or BIN/8 may be used instead of X/Y in this figure. (See 5.4.1.1 and 5.4.1.2.)



• NOTE—Alternatively, the general qualifying symbols BIN/4 may be used instead of X/Y in this figure. (See 5.4.1.1 and 5.4.1.2.)

In general, dependency notation shown at the inputs to the left of the bit-grouping symbol applies to the inputs of the coder, and dependency notation shown after the bit-grouping symbol applies to the outputs of the coder. However, for inputs with inherent storage, see 4.4.5.

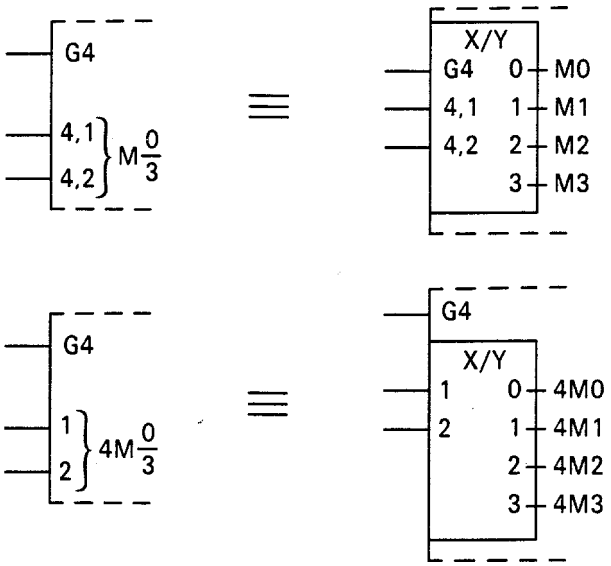


Illustration of the concept

• NOTE—Alternatively, the general qualifying symbol BIN/4 may be used instead of X/Y in this figure. (See 5.4.1.1 and 5.4.1.2.)

4.4.3 Order of input labels

If one or more of the following symbols:

line grouping	3.3-37	]
amplification	3.3-9.5	▷
hysteresis	3.3-2	⊓

are required at an input, they shall be shown, as needed, in that order (3.3-37, 3.3-9.5, 3.3-2) reading from the input towards the interior of the element.

These symbols shall be shown between the input line(s) and any dependency notation.

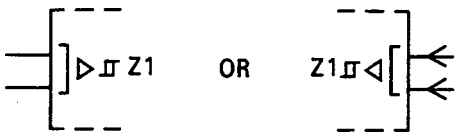
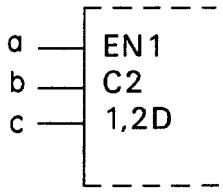


Illustration of the concept

If any input having a single function is affected by other inputs, the qualifying symbol (if there is any) for that function shall be preceded by the identifying numbers of the affecting inputs. The left-to-right order of these identifying numbers shall be the order in which the effects or modifications apply. The affected input has no effect on the element if the logic state of any one of the affecting inputs, considered separately, would cause the affected input to have no effect, regardless of the logic states of other affecting inputs.

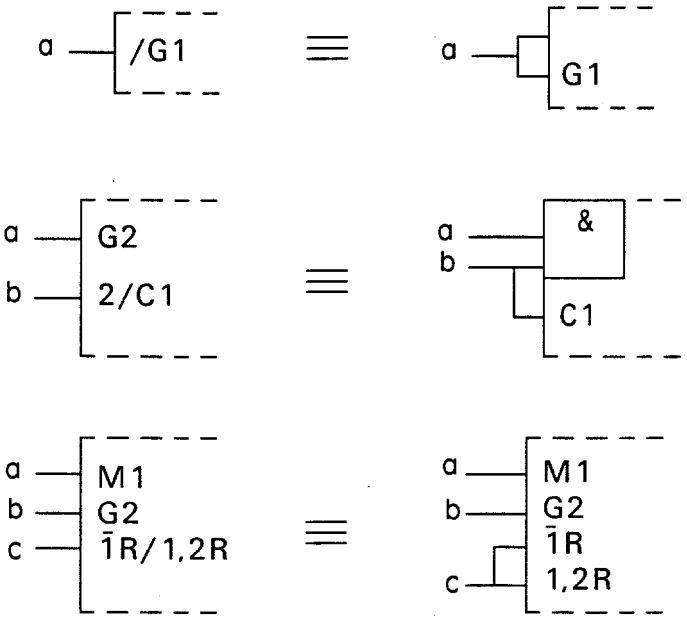


If either $a = 0$ or $b = 0$, then c has no effect.

Illustration of the concept

If any input has several different functions requiring different labeling, the different functions and any associated labels may be shown on different input lines connected together outside the outline (for example, Symbols 5.4-8, 5.13-4, and 5.13-13). However, there are cases in which this method of presentation is not advantageous. In those cases, the input may be shown once with the different labels separated by solidi. No meaning is attached to the order of these labels. If one of the functions is that of an unlabelled input, a solidus shall precede the first label shown (for example, Symbol 5.13-15).

Illustrations of the concept

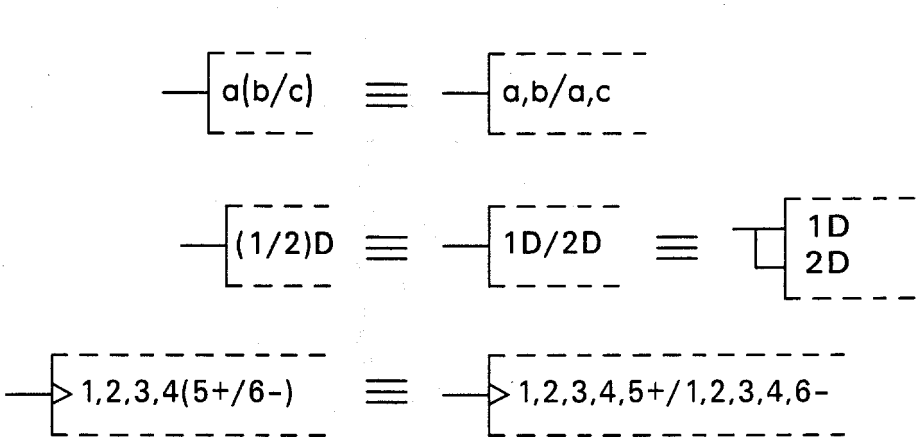


If all inputs of a combinational element are disabled (caused to have no effect on the function of the element), the internal logic states of the outputs of the element are not specified by the symbol.

If all inputs of a sequential element are disabled (caused to have no effect on the function of the element), the content of this element is not changed and the outputs remain at their existing internal logic states.

It may be necessary to set off a specific label or to factor labels. Parentheses may be used to accomplish the separation of labels. In setting off a specific label, the parentheses have no algebraic connotation but are used to separate the label and to minimize confusion, for example, (CT=9)1,2Z3. In addition, the use of parentheses allows factoring of labels parallel in concept to the distributive law in algebra (for example, A(B+C) = AB+AC). In this notation, factoring would be as follows:

Illustrations of the concept



NOTE — Since brackets ([,]) or braces ({,}) have other defined meanings, they are not the equivalent of parentheses.

4.4.4 Order of output labels

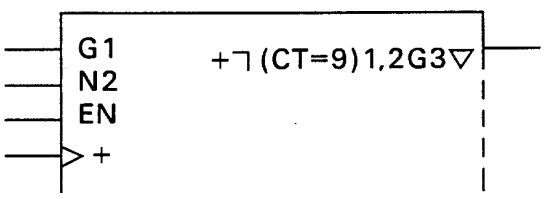
If an output has a number of different labels, regardless of whether they are identifying numbers of affecting inputs or outputs or not, these labels shall be shown in the following order:

1) If the postponed output symbol (Symbol 3.3-1) has to be shown, this comes first, if necessary preceded by the indications of the inputs to which it must be applied
2) Followed by the labels determining or modifying the internal logic state of the output, such that the left-to-right reading order of these labels corresponds with the order in which their effects must be applied, (see Symbol 5.13-17)
3) Followed by the he label indicating the effect of the output on inputs and other outputs of the element.

Symbols for open-circuit, passive-pulldown, passive-pullup, 3-state, or specially amplified outputs (Symbols 3.3-3 through 3.3-9), shall be drawn adjacent to the output line, except as otherwise permitted within this section. For example, see Symbol 5.2-6.

Two adjacent identifying numbers of affecting inputs in a set of labels not already separated by a nonnumeric character should be separated by a comma.

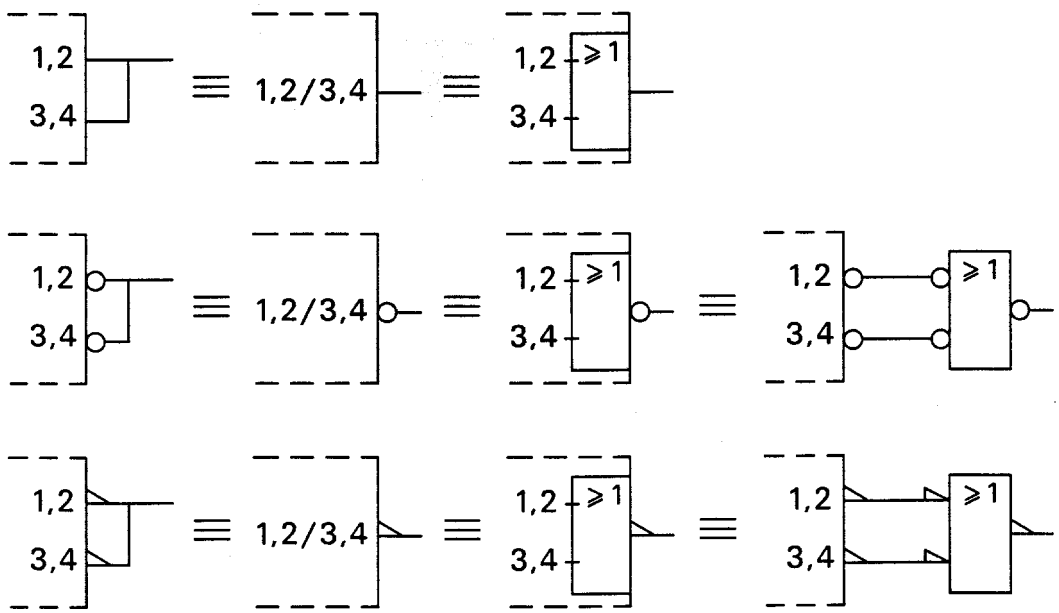
Illustration of the concept



If an output has several sets of labels that stand in an OR relationship to each other or that represent alternative modes of operation, these sets may be shown either

1) On different output lines connected together outside the outline, or
2) On the same output line with the different sets of labels separated by solidi

The two methods are equivalent. The output has but one internal state at a time, regardless of which method of representation is used. In the case of OR relationships, the following equivalencies hold:



Confusion may result from the use of the multi-line form in conjunction with the negated output symbol (Symbol 3.1-1) or the polarity symbol (Symbol 3.1-6 or 3.1-7), and these combinations are not recommended.

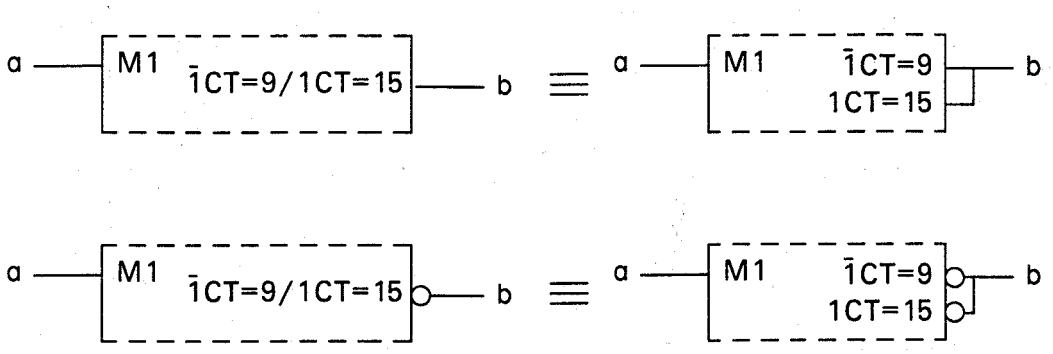
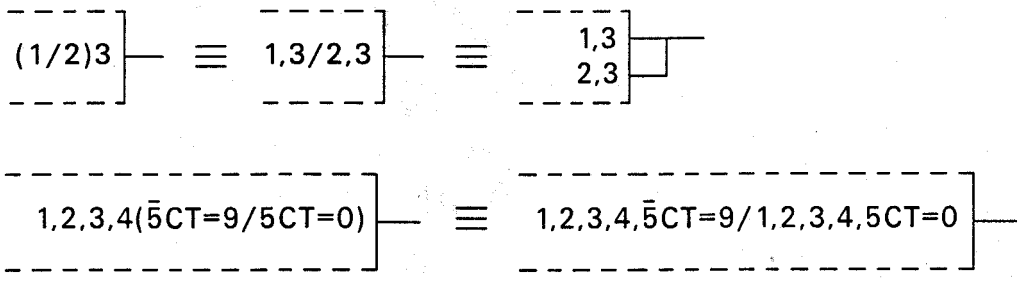


Illustration of the concept

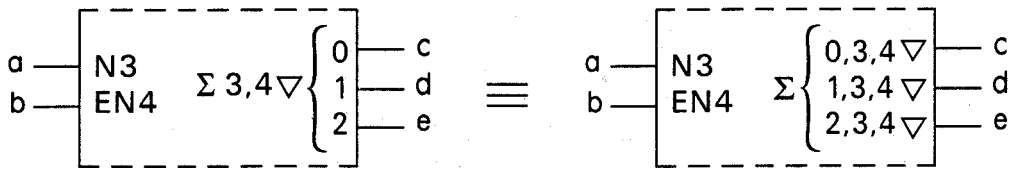
Labels may also be separated or factored using parentheses, as explained in 4.4.3.

Illustrations of the concept



When the bit-grouping symbol for outputs (Symbol 3.3-26) is used and the sets of labels of all outputs grouped together differ only in the indications of the weights, the sets of labels, including the symbols for open-circuit, passive-pulldown, passive-pullup, 3-state, or specially amplified outputs (Symbols 3.3-3 through 3.3-9), but excluding the indications of the weights, may be shown only once between the symbol replacing the * and the grouping symbol, provided that, except for the grouping symbol and weights, the proper order of the labels is maintained.

Illustration of the concept



4.4.5 Labelled inputs other than D inputs having inherent storage

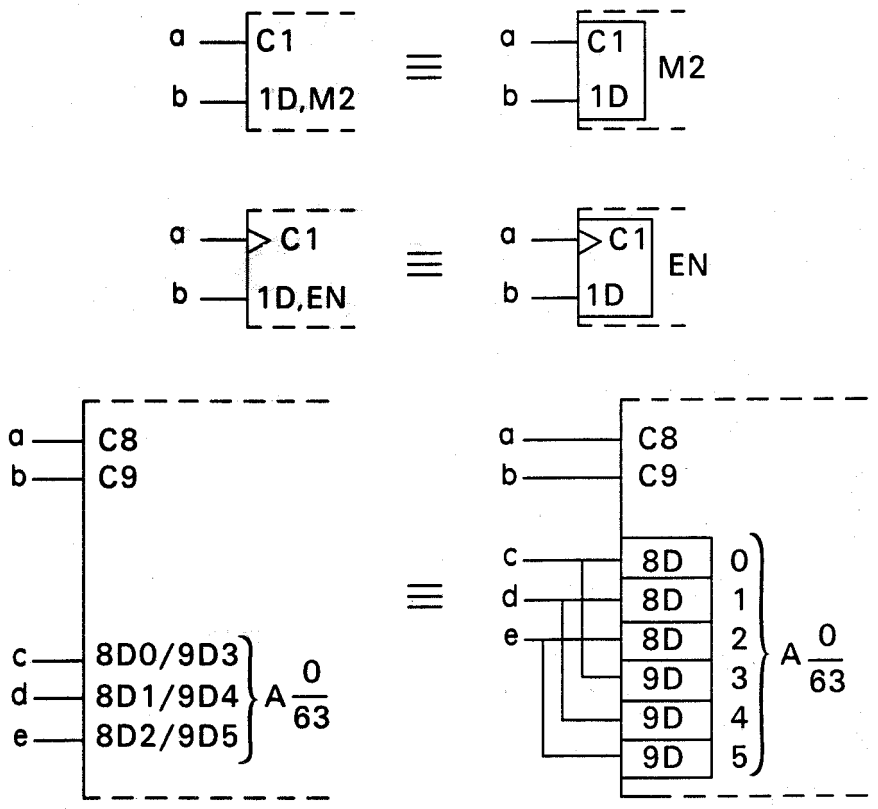
It often occurs that a labelled input other than a D input has inherent storage. Such an input may be labelled as:

mD,*

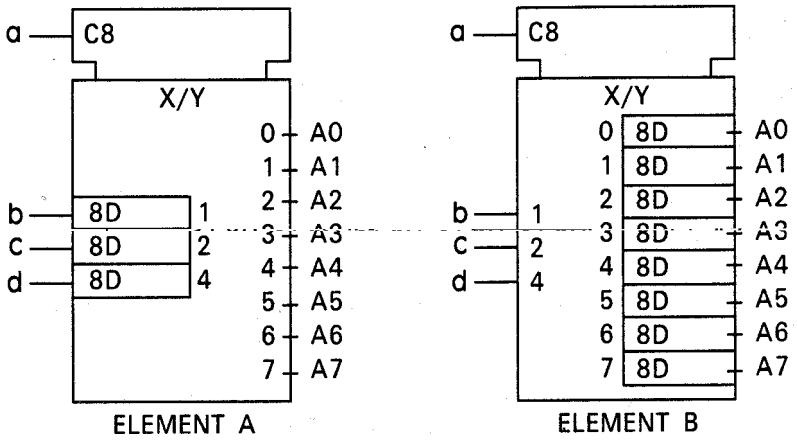
in which

1) "m" shall be replaced by the identifying numbers of the inputs that affect the storage operation, and
2) "*" shall be replaced by the symbol denoting the function of the stored input. If that symbol is a number, the comma following the "D" may be omitted.

Illustration of the concept

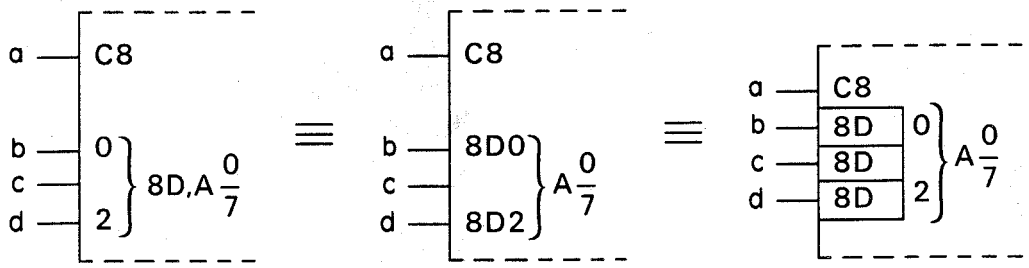


Any combinational logic element with a storage register on all its inputs is functionally equivalent to that same combinational logic element with a storage register on its outputs. Thus, element A below is functionally equivalent to element B.



Because of this, inherent storage may also be indicated by placing the “mD”, as defined above, between the bit-grouping symbol and the symbol denoting the function of the stored input.

Illustration of the concept



5. Combinational and sequential elements

General notes

All qualifying symbols inside the outline are defined in terms of the internal logic states of the relevant inputs and outputs (see Section 1).

Some of the examples in this section illustrate the use of direct polarity indication; others illustrate the use of a single logic convention. Users should select the representational system best suited to their application. Where use has been made of a single logic convention, the positive-logic convention has been used.

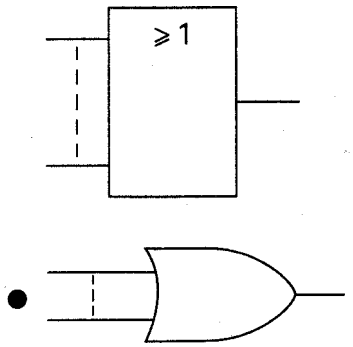
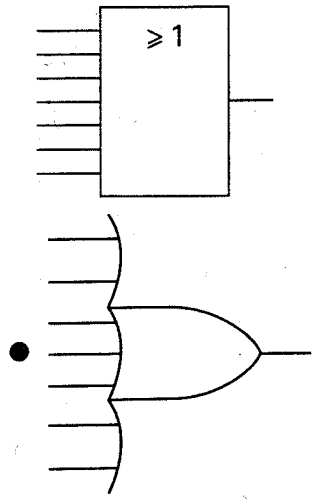
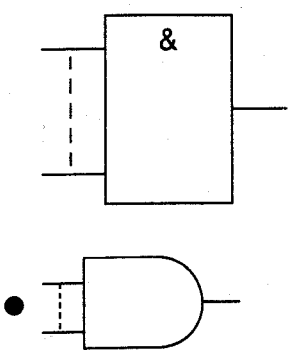
In may cases, examples are based on commercial devices, and pin numbers (for one unspecified package type) have been shown for the assistance of the reader.

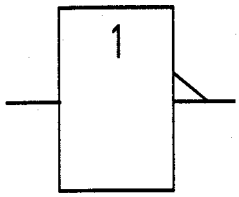
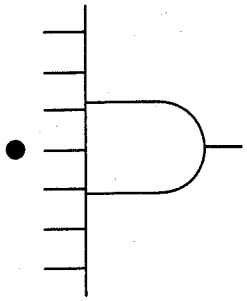
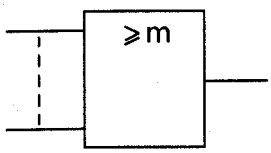
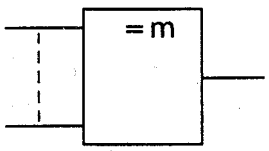
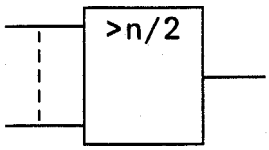
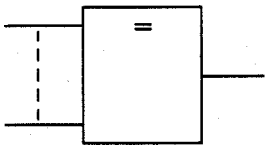
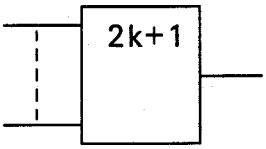
A given function may be symbolized in more than one way depending on the purpose it serves in the system (for example, Symbols 5.1-26 and 5.1-27). Also, use is often made of the complementary representation (for example, Symbols 5.9-7 and 5.9-8), especially of combinational functions, to enhance the understanding of the diagram. For example, an OR function may be shown as an AND element but with negated inputs and outputs. The choice of the symbol should be governed by the relevant application of the function.

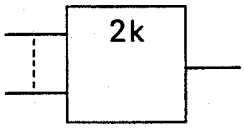
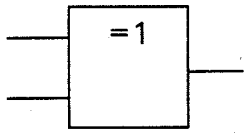
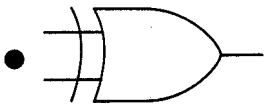
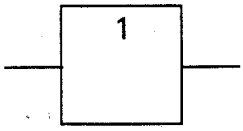
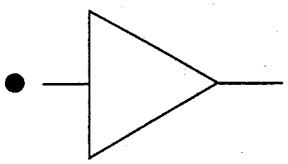
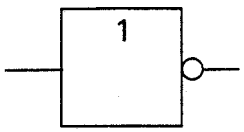
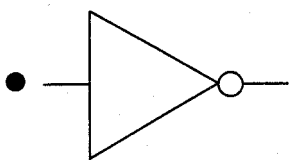
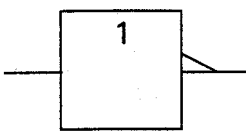
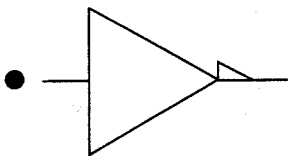
5.1 Basic combinational elements

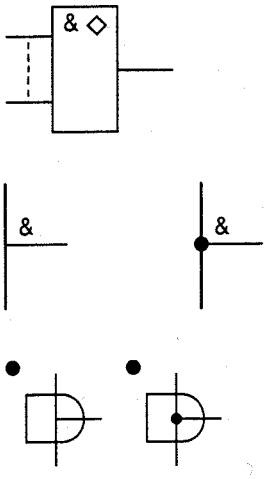
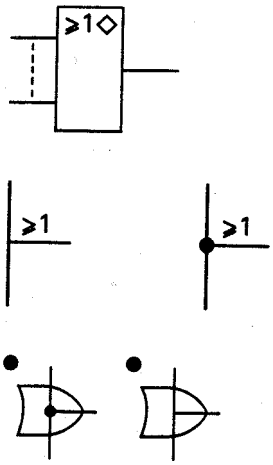
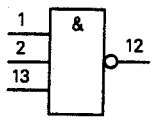
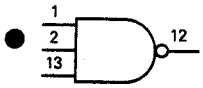
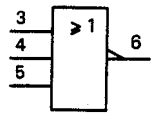
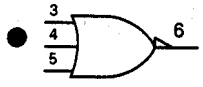
Basic rule

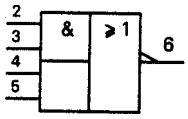
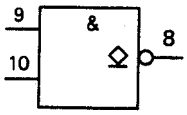
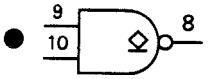
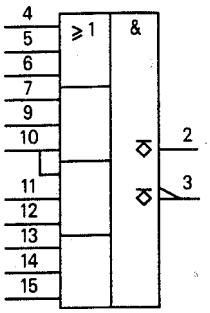
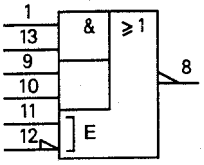
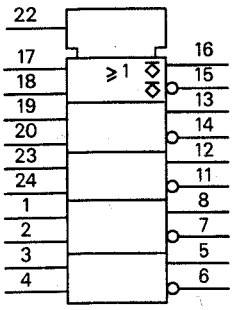
The qualifying symbol for the function of the element indicates the number of inputs that must take on the internal 1-state to cause the outputs to take on their internal 1-states. Subject to this rule other qualifying symbols than those shown in this section may be developed.

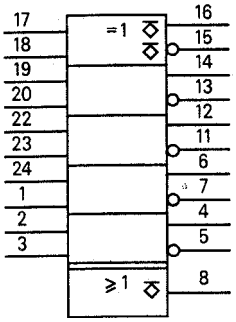
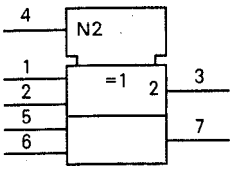
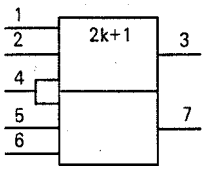
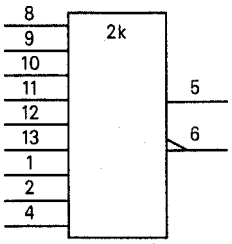
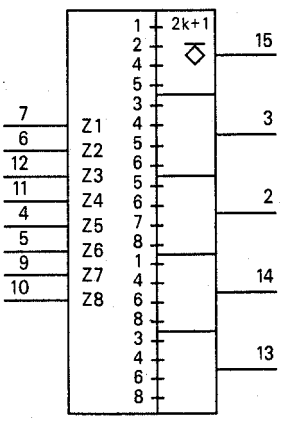
No	Symbol	Description
5.1-1		OR element, general symbol The output stands at its 1-state if and only if one or more of the inputs stand at their 1-states. NOTES: 1 — "≥ 1" may be replaced by "1" if no confusion is likely. 2 — The distinctive-shape symbol is, according to IEC Publication 617, Part 12, not preferred, but is *not considered to be in contradiction* to that standard. Its use in combination to form complex symbols (for example, use as an embedded symbol) is discouraged.
5.1-2		OR element extension to accommodate many inputs (seven shown). NOTE — Only the distinctive-shape symbol may use a line extended from the input side to accommodate a large number of inputs. For rectangular-shape symbols, the aspect ratio may be altered as appropriate. See NOTE 2 to Symbol 5.1-1.
5.1-3		AND element, general symbol The output stands at its 1-state if and only if all inputs stand at their 1-states. See NOTE 2 to Symbol 5.1-1.

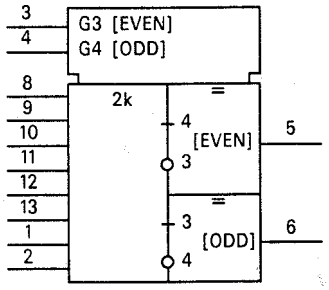
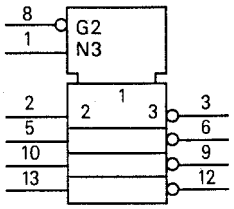
No	Symbol	Description
5.1-4	 	AND element extension to accommodate many inputs (seven shown) See NOTE 2 to Symbol 5.1-1 and NOTE to Symbol 5.1-2.
5.1-5		Logic threshold element, general symbol The output stands at its 1-state if and only if the number of inputs that stand at their 1-states is equal to or greater than the number in the qualifying symbol, represented here by m. NOTE — m shall always be smaller than the number of inputs.
5.1-6		m and only m element, general symbol The output stands at its 1-state if and only if the number of inputs that stand at their 1-state is equal to the number in the qualifying symbol, represented here by m. NOTE — m shall always be smaller than tne number of inputs.
5.1-7		Majority element, general symbol The output stands at its 1-state if and only if the majority of the inputs stand at their 1-states.
5.1-8		Logic identity element, general symbol The output stands at its 1-state if and only if all inputs stand at the same state.
5.1-9		Odd element, general symbol Odd-parity element, general symbol Addition modulo 2 element, general symbol The output stands at its 1-state if and only if the number of inputs that stand at their 1-state is odd (1, 3, 5, etc.).

No	Symbol	Description
5.1-10		Even element, general symbol Even-parity element, general symbol The output stands at its 1-state if and only if the number of inputs that stand at their 1-states is even (0, 2, 4, etc.).
5.1-11	 	Exclusive-OR element, general symbol The output stands at its 1-state if one and only one of the two inputs stands at its 1-state. NOTE — Here this two-input device is represented as a special case of the use of Symbol 5.1-6. However, for representing an exclusive-OR of three or more inputs, Symbol 5.1-9 should be used. See NOTE 2 to Symbol 5.1-1.
5.1-12	 	Buffer without specially amplified output, general symbol The output stands at its 1-state if and only if the input stands at its 1-state. NOTE — See also 5.2. See NOTE 2 to Symbol 5.1-1.
5.1-13	 	Inverter (in the case of device representation using a single logic convention), general symbol Negator, general symbol The output stands at its external 0-state if and only if the input stands at its external 1-state. See NOTE 2 to Symbol 5.1-1.
5.1-14	 	Inverter (in the case of device representation using direct polarity indication), general symbol The output stands at its L-level if and only if the input stands at its H-level. See NOTE 2 to Symbol 5.1-1.

No	Symbol	Description
5.1-15		Distributed-AND function Dot-AND function Wired-AND function A distributed-AND function is a connection of specific outputs of a number of elements that are joined together to achieve the AND function. As an alternative, the distributed-AND function may be shown by one of the symbols shown at the left. NOTE — The distinctive-shape versions are not explicitly permitted by IEC Publication 617.12 (1983).
5.1-16		Distributed-OR function Dot-OR function Wired-OR function A distributed-OR function is a connection of specific outputs of a number of elements that are joined together to achieve the OR function. As an alternative, the distributed-OR function may be shown by one of the symbols shown at the left. NOTE — The distinctive-shape versions are not explicitly permitted by IEC Publication 617-12 (1983).
5.1-17	 	AND element with negated output (NAND) See NOTE 2 to Symbol 5.1-1. *Part of SN7410*
5.1-18	 	OR element with active-low output (NOR) See NOTE 2 to Symbol 5.1-1. *Part of SN427*

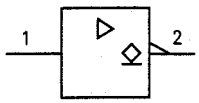
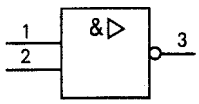
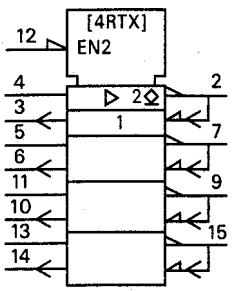
No	Symbol	Description
5.1-19		AND-OR-Invert element *Part of SN74L51*
5.1-20	 	NAND element with NPN open-collector output See NOTE 2 to Symbol 5.1-1. *Part of SN7403*
5.1-21		OR-AND with complementary NPN open-emitter outputs *MC10121*
5.1-22		AND-OR-Invert, expandable NOTE — The line grouping symbol (Symbol 3.3-37) indicates that the two input lines are needed to implement a single extension connection. *Part of SN7450*
5.1-23		Expander See NOTE to Symbol 5.1-22. *Part of SN7460*
5.1-24		OR, quint, with one common input and with complementary outputs *F100102*

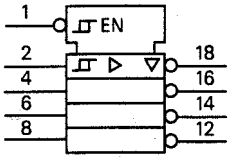
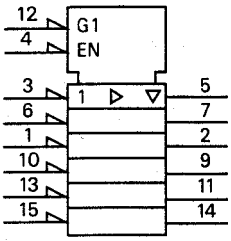
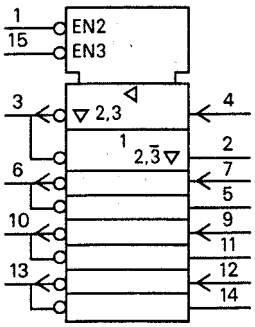
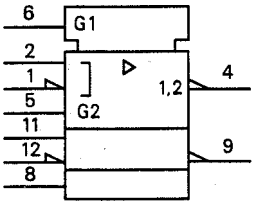
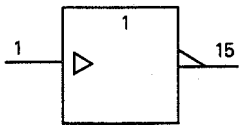
No	Symbol	Description
5.1-25		Exclusive-OR, quint, with complementary outputs and one common output NOTES: 1 — Each of the five elements of the array has two outputs that always have identical internal logic states; the internal state of each of the elements is an input of the common output element (see 2.3.3). 2 — All ECL outputs in this device family are of the same open-circuit type. Therefore, the open-circuit output symbols may be omitted if no confusion is likely. *F100107*
5.1-26		Exclusive-OR/NOR, dual NOTE — Symbol 5.1-27 depicts the same device in another way. *Part of SN74S135*
5.1-27		ODD element, dual with one common input NOTE — Symbol 5.1-26 depicts the same device in another way. *Part of SN74S135*
5.1-28		Parity generator/checker with complementary outputs *SN74280*
5.1-29		Error detection/correction element *MC10163*

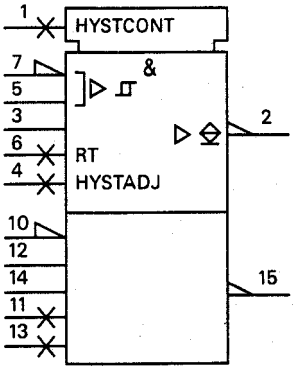
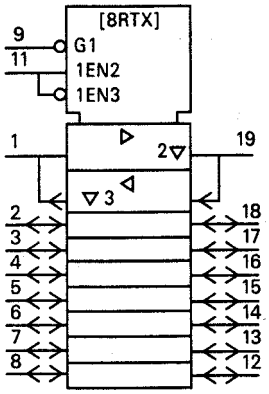
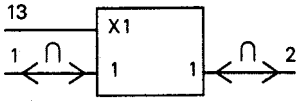
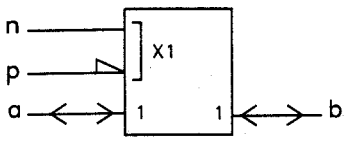
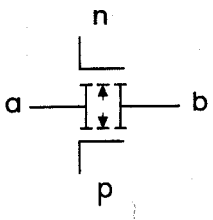
No	Symbol	Description
5.1-30		Parity generator/checker, odd/even *SN74180*
5.1-31		True/complement, zero/one element, quad *SN74H87*

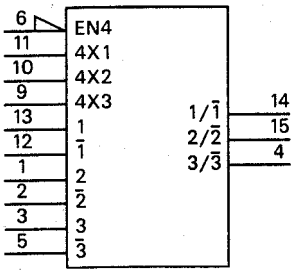
5.2 Buffers with special amplification, drivers, receivers, and bidirectional switches

The symbol ▷ denotes the function of amplification. It may be combined with other qualifying symbols to indicate elements having special amplification. This symbol shall point in the direction of signal flow.

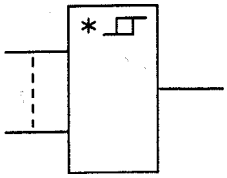
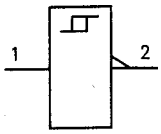
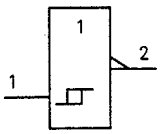
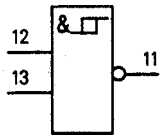
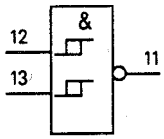
No	Symbol	Description
5.2-1		Buffer/driver with negated NPN open-collector output *Part of SN7406*
5.2-2		NAND buffer *Part of SN7437*
5.2-3		Bus transceiver, quad NOTES: 1 — The arrows at the left-hand side are essential to clarify the function of the device; those on the right-hand side are optional. 2 — The general qualifying symbols and those associated with the outputs of the two outlines forming the first element of the array have been omitted from the remaining elements of the array in accordance with 2.3.1.1. *Am26S10*

No	Symbol	Description
5.2-4		Bus driver with bithreshold inputs and 3-state outputs, quad *Part of SN74S240*
5.2-5		Buffer, inverting, with 3-state outputs, hex *CD4502B*
5.2-6		Bus driver, parallel bidirectional, quad NOTE — Pin 1 could be labeled as an EN input (Symbol 3.3-12) without dependency notation, that is, the identifying number 2 may be omitted in three places inside the outline. *8226*
5.2-7		Line receiver, dual *SN75107*
5.2-7A		Line receiver *Part of SN75127*

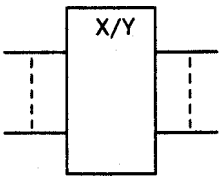
No	Symbol	Description
5.2-7B		Line receiver, dual NOTE — The input sensitivity can be set to ±0.5 V; the output can sink 6.4 mA. *SN75152*
5.2-8		Bus driver, bidirectional, 8-bit parallel *8286*
5.2-9		Bidirectional switch NOTE — The arrowheads or the "∩" symbols, or both, are optional.
5.2-10		CMOS transmission gate equivalent to  Internal transmission gate used in many CMOS circuits such as CD4013B, CD4035B, and CD4042B. NOTE — The arrowheads at the input/output ports are optional.

No	Symbol	Description
5.2-11		Bidirectional change-over switch with common enable, triple *CD4053B*

5.3 Elements exhibiting hysteresis Schmitt triggers Bithreshold detectors

No	Symbol	Description
5.3-1		Element exhibiting hysteresis, general symbol When used within an element as a general qualifying symbol, the hysteresis symbol⊓, designates an element whose overall input/output characteristics include hysteresis. The asterisk must be replaced by a general qualifying symbol designating the logic function of the element unless that qualifying symbol is the numeral 1 (Symbol 5.1-12), in which case it shall be omitted.
5.3-2		Schmitt-trigger inverter Bithreshold detector with active-low output Inverter with hysteresis NOTE — Symbol 5.3-2 is equivalent to (see Symbol 3.3-2):  *Part of SN74LS14*
5.3-3		NAND Schmitt trigger NAND with hysteresis The output takes on its internal 1-state only when the external level applied to each input reaches its V1 threshold (see definition in 3.3-2). The output will maintain the internal 1-state until the external level applied to one of its inputs reaches its V0 threshold. *Part of SN74132* NOTE — Symbol 5.3-3 is *not* equivalent to: 

5.4 Coders

No	Symbol	Description
5.4-1		Coder, general symbol Code converter, general symbol The relationship between inputs and output shall be shown by the use of indications in the general qualifying symbol together with labels at the inputs and outputs or by the use of a referenced table. NOTE — X and Y may be replaced by appropriate indications of the code used to represent the information at the inputs and at the outputs, respectively. See NOTE 1 to Symbol 5.5-1.

5.4.1 Indicating input and output codes in the general qualifying symbol

This method of indicating code conversion is based on the following rule:

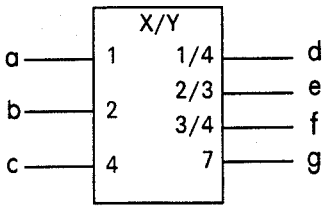
Depending on the input code, the internal logic states of the inputs determine an internal number, value, or its equivalent. This internal number or value is reproduced by the internal logic states of the output depending on the output code.

The relationships between the internal logic states of the inputs and the internal number or value is indicated either by:

1) Labeling the inputs with numbers, in which case the internal number equals the sum of the numbers associated with those inputs that stand at their internal 1-states; or by:
2) Replacing X by an appropriate designation of the input code and labeling the inputs with characters that refer to this code.

The relationships between the internal number or value and the internal logic states of the outputs are indicated either by:

1) Labelling each output with a list of those internal numbers that lead to the internal 1-state of that output. These numbers shall be separated by solidi. This labeling may also be applied when Y is replaced by a letter denoting a type of dependency (see also 4.4.1). If a continuous range of internal values produces the internal 1-state of an output, this can be indicated by two numbers that are inclusively the beginning and the end of the range, with these two numbers separated by three dots, for example, 4...9 = 4/5/6/7/8/9; or by
2) Replacing Y by an appropriate indication of the output code and labelling the outputs with characters that refer to this code.



Output *d* stands at its internal 1-state for the following combinations of internal logic states at inputs *a*, *b*, and *c*:

$a = 1 \quad b = 0 \quad c = 0$
$a = 1 \quad b = 0 \quad c = 1$

● Alternatively, the general qualifying symbol BIN/6 may be used instead of X/Y. (See 5.4.1.1 and 5.4.1.2.)

Illustration of the concept

• If X or Y is replaced by an indication of a specific code, further rules apply. There are three general categories of codes: summing codes, direct-indication codes, and codes representing symbols.

5.4.1.1 • Summing codes

With these codes, like “X”, there is an internal numeric value that corresponds to the sum of the weights of the inputs (or outputs) that stand at their internal 1-states.

The indication of the relationship between the internal logic states of the inputs (or outputs) and the internal value is accomplished by replacing X (or Y) of the qualifying symbol with an appropriate indication of the input (or output) code and by labeling the inputs (or outputs) with numbers indicating their individual weights.

The following summing codes are defined:

BIN Binary code
The number code in which the individual weights are all powers of 2. Inputs (or outputs) are labelled either with decimal weights or with decimal exponents of the powers of 2.

BCD 8-4-2-1 Binary coded decimal
The number code in which each digit in the decimal representation of a number is encoded as a binary number in 4 bits with the relative weights of 8, 4, 2, and 1.

Example:

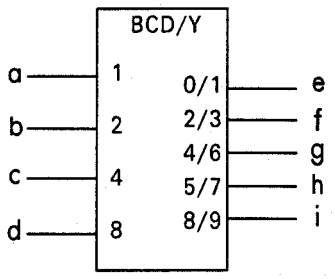
decimal number	BCD code
0	0000
1	0001
8	1000
9	1001
10	0001 0000
11	0001 0001
175	0001 0111 0101

Inputs or outputs are labelled with decimal weights, e.g., 1, 2, 4, 8, 10, 20, ...

NOTE — For inputs, the behavior of the element is unspecified by the symbol if the internal value produced by any set of four inputs exceeds 9 (x 10^n). For outputs, the behavior of the element is unspecified by the symbol if the internal value requires more digits than are provided at the outputs.

X-3 Excess-three code

The BCD code in which the internal value of each 4 inputs (or outputs) is 3 (x 10^n) less than the sum of those inputs (or outputs). See note to BCD.



Output *i* stands at its internal 1-state for the following combinations of internal logic states at inputs *a*, *b*, *c*, and *d*:

$a = 0 \quad b = 0 \quad c = 0 \quad d = 1$
$a = 1 \quad b = 0 \quad c = 0 \quad d = 1$

- Alternatively, the general qualifying symbol BCD/5 may be used instead of BCD/Y. (See 5.4.1.2.)
- For invalid BCD codes, that is, those that would produce an internal value that is greater than 9, the resulting outputs are not specified by this symbol.
- If the general qualifying symbol were BIN/Y, then the symbol would show that all outputs stand at the internal 0-state for internal values greater than 9.

Illustration of the concept

5.4.1.2 • Direct-indication codes

With these codes, like "Y", the relationship between the internal value and the internal logic state of each input (or output) is indicated by replacing X (or Y) of the qualifying symbol with an appropriate indication of the input (or output) code and by labeling each input with a number indicating the internal value produced, or by labeling each output with a list of those internal values that lead to the internal 1-state of the output. These values shall be separated by solidi.

If a continuous range of internal values produces the internal 1-state of an output, this can be indicated by two numbers that are inclusively the beginning and the end of the range, with these two numbers separated by three dots, for example:

```
4...9 = 4/5/6/7/8/9
```

The following codes are defined:

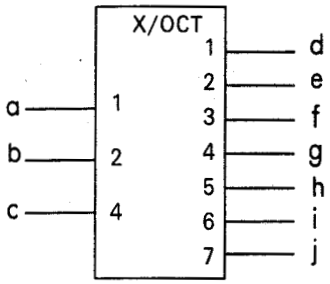
m — General code with m states (m is replaced by a number)
A code in which m combinations of internal logic states are allowable for inputs or possible for outputs.

HPRI — Highest-priority input code
An input code in which the input with the highest weight takes priority if more than one input stands at its internal 1-state. If no input stands at its internal 1-state, the internal value zero is produced.

DEC — Decimal code
The code in which 10 inputs (or outputs) exist and have the weights 0 through 9.
NOTE — If the input (or output) with the weight of zero is omitted, the internal value of zero corresponds to all inputs (or outputs) standing at their internal 0-state.

OCT — Octal code
The code in which 8 inputs (or outputs) exist and have the weights 0 through 7. See note to DEC.

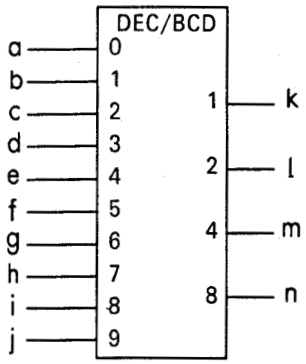
Except for HPRI, if these codes are used for inputs and more than one input stands at its internal 1-state, the behavior of the element is not specified by the symbol.



Output *h* stands at its internal 1-state for the following combinations of internal logic states at inputs *a*, *b*, and *c*:

$a = 1 \quad b = 0 \quad c = 1$

Alternatively, the general qualifying symbol BCD/OCT may be used instead of X/OCT.



If input *j* stands at its internal 1-state, outputs *k* and *n* stand at their internal 1-states.

Illustration of the concept

5.4.1.3 • Codes representing symbols

With these codes, there is no internal numeric value. Instead, each input (or output) pattern represents a symbol (for example, the letter "E") according to a known coding scheme. The equivalent of the internal numeric value is the symbol represented by the input (or output) pattern. Examples of these codes are ISO 646, ANSI X3.4, ASCII, EBCDIC, and 7-segment. The relationship between the internal symbol and the internal logic state of each input (or output) is indicated by replacing X (or Y) of the qualifying symbol with an appropriate indication of the input (or output) code and by labeling each input (or output) with an appropriate indication of its bit position within the code.

If these codes are used in a coder together with a code that is associated with internal numeric values, the conversion to or from these codes is based on the symbolic decimal representation of those internal numeric values. If there is no symbolic representation for a value in the code, the behavior of the element for that value is unspecified by the symbol.

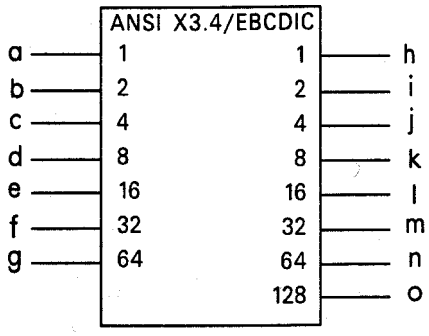


Illustration of the concept

5.4.2 Replacement of X and Y by indications other than designation of the input code or the output code

The internal number or value of a coder may also be produced by other means, e.g., by a counter whose content is the internal number or value, by a multiposition switch whose position produces the internal number of value, etc. In such cases, the X shall be replaced by an appropriate indication of the means involved.

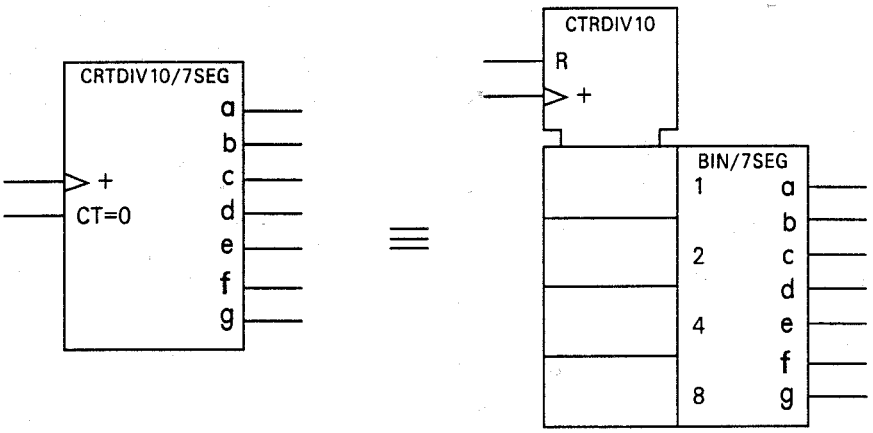
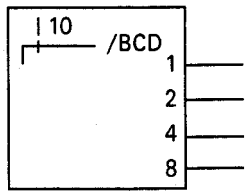


Illustration of the concept

NOTE — For the meaning of CTRDIV10, see symbol 5.13-1.

10-position switch producing a BCD-coded output



The internal number of value of a coder may also be represented by a visual display or be regarded as a value to become the content of an element or as a number on which a mathematical operation is performed. In such cases, the Y shall be replaced by the general qualifying symbol of the function involved.

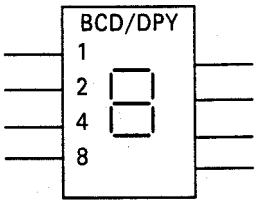


Illustration of the concept

It may be necessary, especially if an internal register is involved, to specify both an input code and an output code in addition to the type of register found in between the inputs and outputs, e.g., “BCD/CTRDIV100/BIN”.

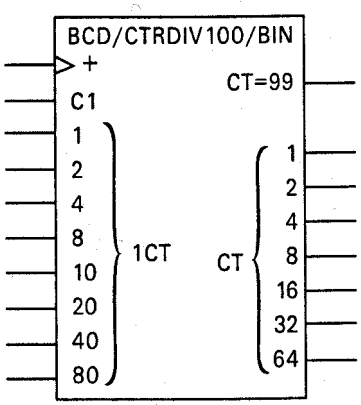


Illustration of the concept

5.4.3 Use of coding tables

As an alternative to one of the codes and labeling defined above, the general qualifying symbol X/Y (or another, more appropriate, qualifying symbol) may be used together with an appropriate reference to a table (for example, Symbol 5.4-9) in which the relationship between the inputs and outputs is indicated. The correspondence between inputs (outputs) and the columns in the table may be given in any convenient way, for example, by using pin numbers. In this case, any internal labeling that might be confused with that arising in one of the other methods must be avoided.

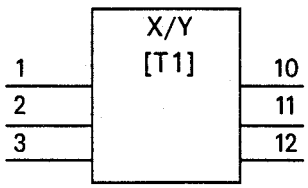
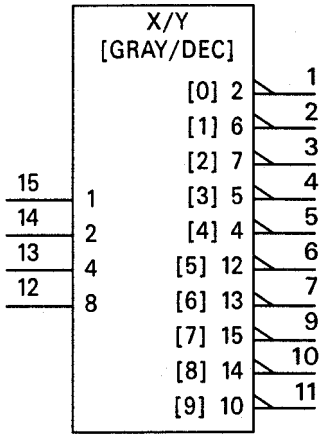
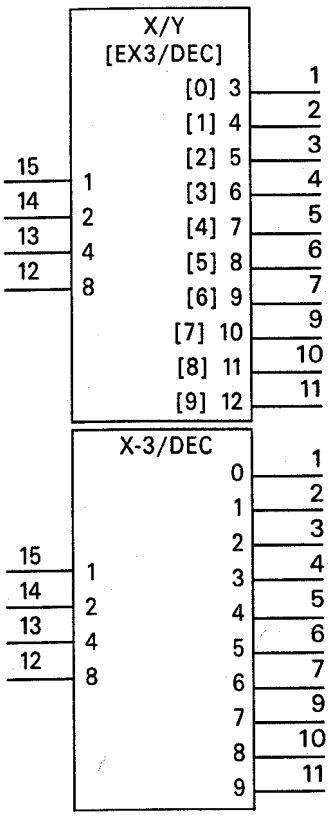


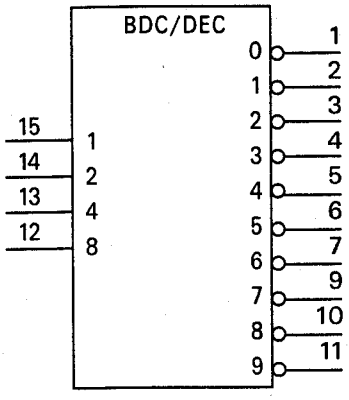
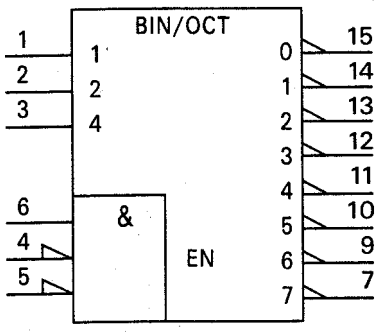
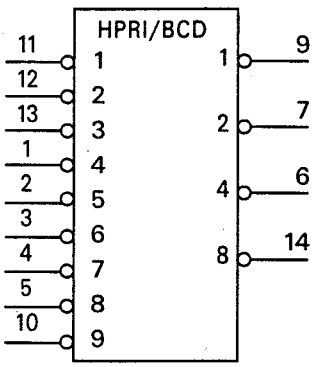
TABLE T1

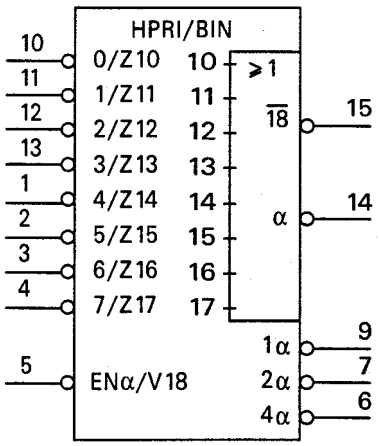
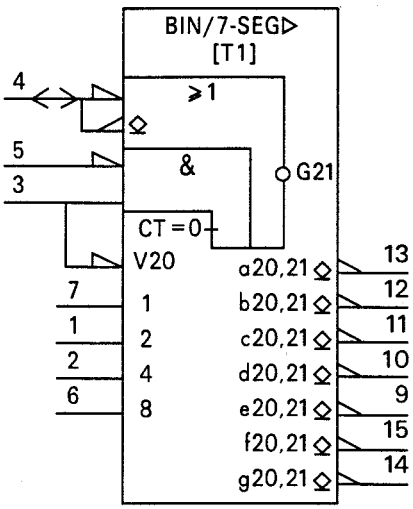
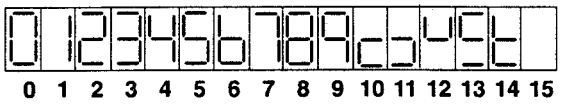
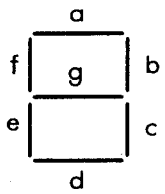
INPUTS 1	2	3	OUTPUTS 10	11	12
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	1
1	1	1	0	0	0

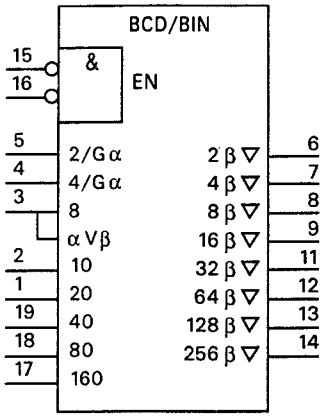
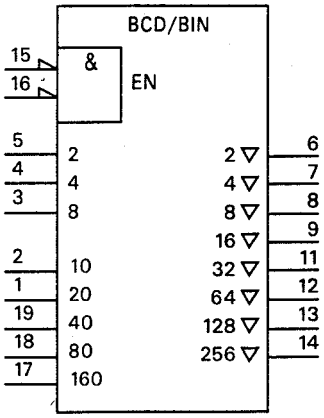
Illustration of the concept

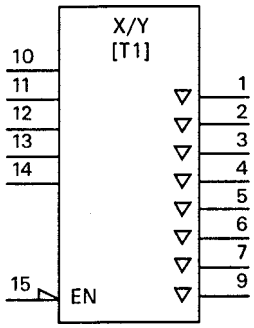
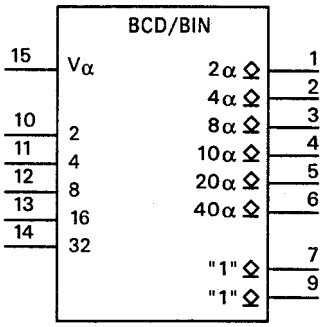
5.4.4 Examples of coders

No	Symbol	Description
5.4-2		Code converter, Gray-to-decimal *SN7444*
5.4-2A		Code converter, excess-3-to-decimal *SN7443*

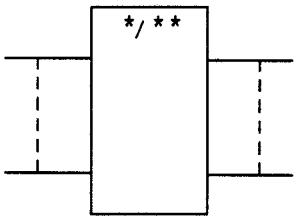
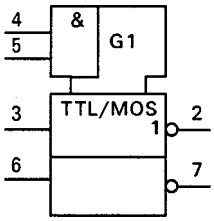
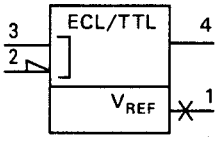
No	Symbol	Description
5.4-3		Code converter, BCD-to-decimal *SN7442*
5.4-4		Code converter, three-to-eight-line NOTE — Symbol 5.6-7 depicts the same device in another way. *SN74LS138*
5.4-5		Highest-priority encoder, encoding 9 data lines to 4-line BCD *SN74147*

No	Symbol	Description
5.4-6		Highest-priority encoder, encoding 8 data lines to 3-line binary (octal) *SN74148*
5.4-7		Decoder/driver, binary-to-seven-segment **Font Table T1:**  **NUMERICAL DESIGNATIONS AND RESULTANT DISPLAYS**  **SEGMENT IDENTIFICATION** NOTE 1—Because the direction of signal flow is implied by the polarity indicators, the arrow heads at pin 4 are optional. NOTE 2—This example shows the use of the polarity symbol at external connections together with the use of the negation symbol at internal connections (see 3.1.1). *SN74LS47*

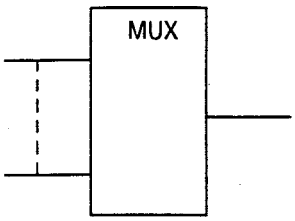
No	Symbol	Description
5.4-8		Code converter, BCD-to-binary NOTES: 1 — Both a detailed version, which accounts for invalid BCD input codes, and a simplified version are shown. 2 — The input code is not pure BCD. Above 80, the coding is actually binary. However, the qualifying symbol BCD/BIN is appropriate in most practical applications of the device.
		*SN745484*

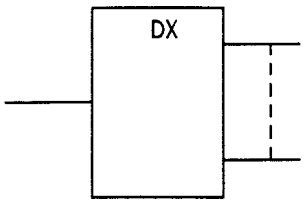
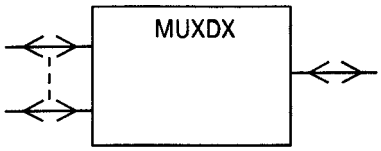
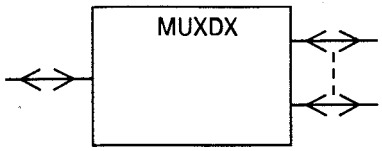
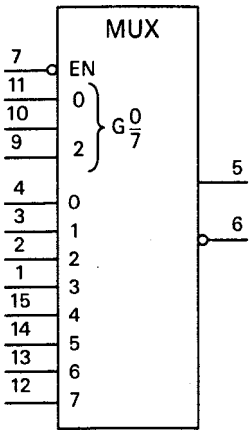
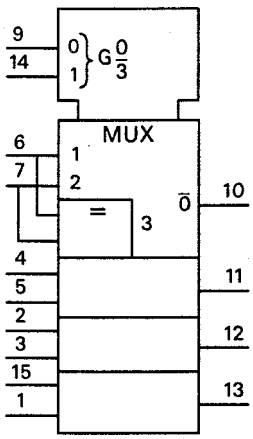
No	Symbol	Description
5.4-9		Coder, in which arbitrary combinational relationships between inputs and outputs are implemented in a PROM (or a ROM) "T1" refers to a table showing the logic function of the device. For instance: INPUTS (14 13 12 11 10) / OUTPUTS (WITH DEVICE ENABLED) (9 7 6 5 4 3 2 1) L L L X X / H L H L L L L L L L H L L / H L H L L L L L L L H L H / H L H L L L L H L L H H X / H H L L L L L H L H L L L / H L H L L L L L L H L L H / H L H L L L H L L H L H X / H H L L L L H L L H H L L / H L H L L L L L L H H L H / H L H L L H L L L H H H X / H H L L L H L L H L L X X / L L H L L L L L H L H X X / L L H L H L L H H H L X X / L L H H L L H L H H H X X / L L H H H H L L X indicates irrelevant (don't care) • NOTE—In this particular example, the alternative general qualifying symbol 32/11 is preferred. This indicates 32 possible input combinations and 11 possible output combinations. There are 14 rows in the table; however, in the output columns, rows 2, 5, and 8 duplicate row 1. *TBP18S030 (formerly SN74S288)*
5.4-10		Code converter, binary-to-BCD *SN74185*

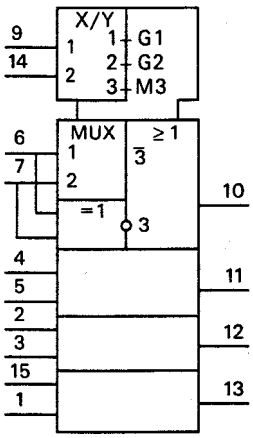
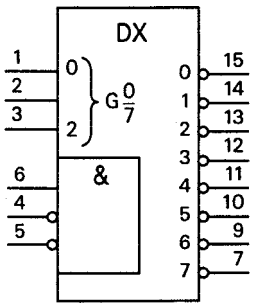
5.5 Signal-level converters

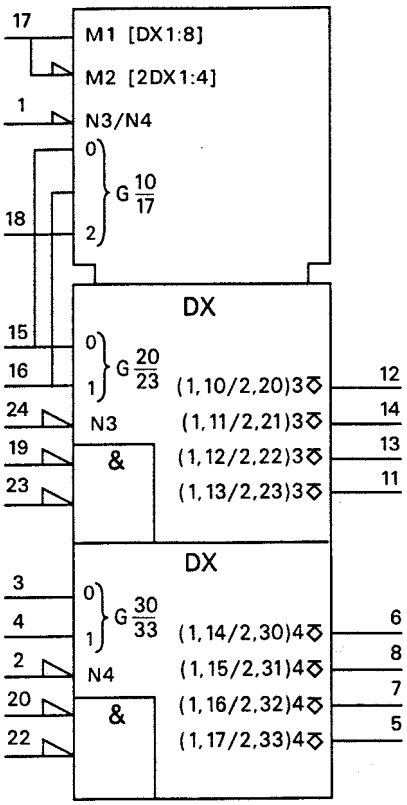
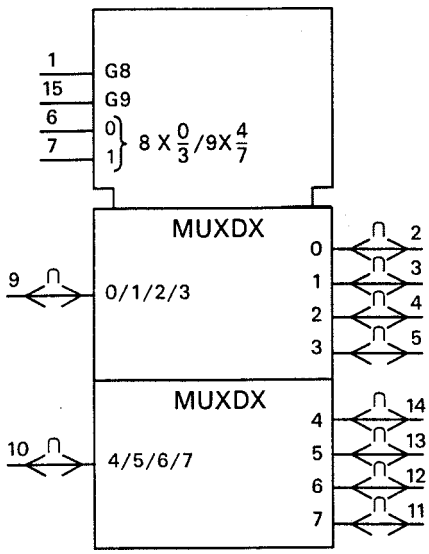
No	Symbol	Description
5.5-1		Signal-level converter, general symbol The * and ** shall be replaced by appropriate indications of the input and output levels, respectively. NOTES: 1 — IEC 617-12 permits replacement of */** by X/Y for a converter between unspecified levels if no confusion with the coder (symbol 5.4-1) is likely. 2 — The "/" in the general qualifying symbol may be replaced by "//" if it is necessary to indicate electrical isolation between the input side and the output side.
5.5-2		Level converter, TTL-to-MOS, dual *Part of SN75365*
5.5-3		Level converter, ECL-to-TTL NOTE — If it is not necessary to show the voltage-reference output, the lower part of the symbol may be omitted. *Part of MC10125*

5.6 Multiplexers and demultiplexers

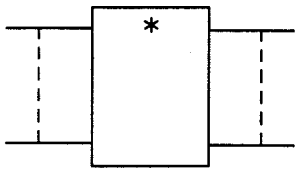
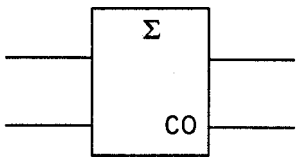
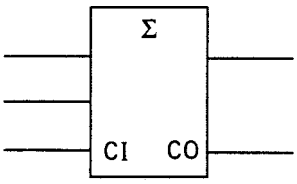
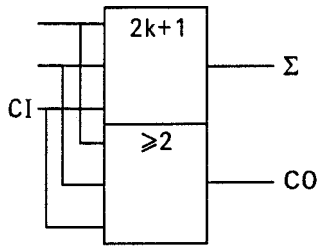
No	Symbol	Description
5.6-1		Multiplexer, general symbol If one input of a multiplexer is selected, the internal logic state of the output takes on the internal state of the selected input. If no input is selected, the output stands at its internal 0-state. NOTE — The inputs and logical relationships that control the selection action must also be shown, for example, by showing those inputs and the associated dependency notation either within the element or within a common control block.

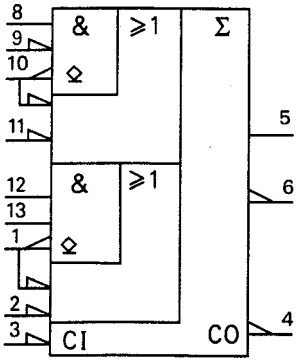
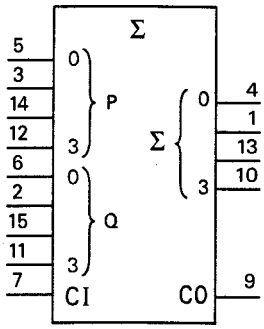
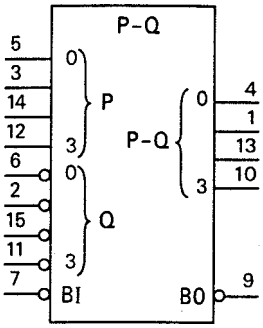
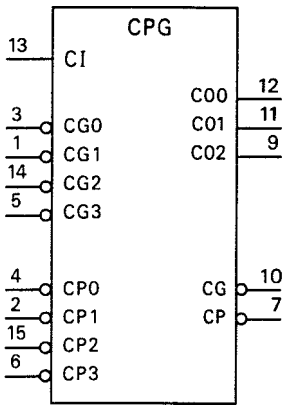
No	Symbol	Description
5.6-2		Demultiplexer, general symbol If an output of a demultiplexer is selected, the internal logic state of that output takes on the internal logic state of the input. Otherwise, the output takes on its internal 0-state. NOTE — If confusion is likely, DX (preferred by IEC) should be replaced by DMUX. See NOTE to Symbol 5.6-1.
5.6-3	 	Bidirectional multiplexer/demultiplexer (selector), general symbol This element establishes a bidirectional connection between a single input/output port and another input/output port selected from a group of input/output ports. See NOTES to Symbols 5.6-1 and 5.6-2. NOTES: 1 — The arrowheads are optional. 2 — IEC abbreviates MUXDX as MDX.
5.6-4		Multiplexer (one-of-eight) *SN74151*
5.6-5		Multiplexer, quad NOTES: 1 — The $\overline{0}$ at the output is optional. See description of Symbol 5.6-1. 2 — Symbol 5.6-6 depicts the same device in another way. *MC14519*

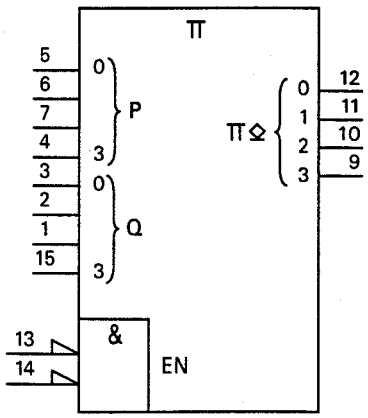
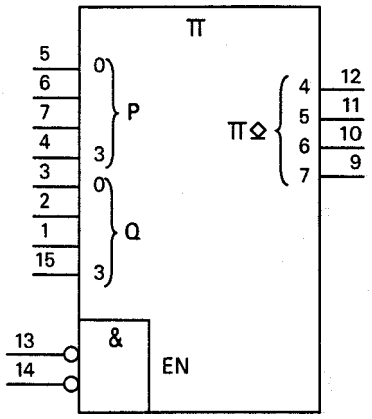
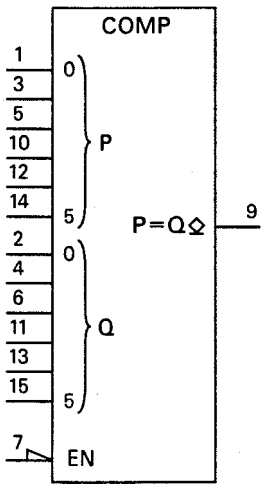
No	Symbol	Description
5.6-6		Multiplexer, quad Exclusive-NOR, quad NOTE — Symbol 5.6-5 depicts the same device in another way. • NOTE 2—BIN/4 may replace X/Y.
5.6-7		Demultiplexer (one-to-eight) NOTE — Symbol 5.4-4 depicts the same device in another way. See NOTE to Symbol 5.6-2. *SN74LS138*

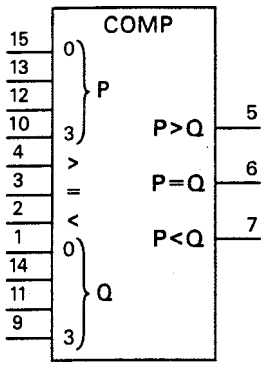
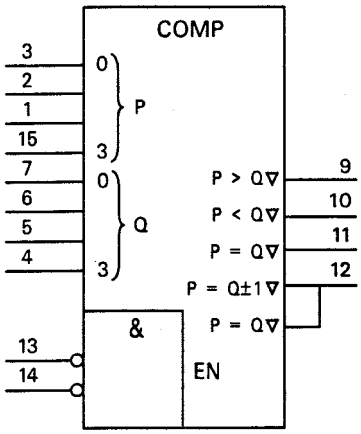
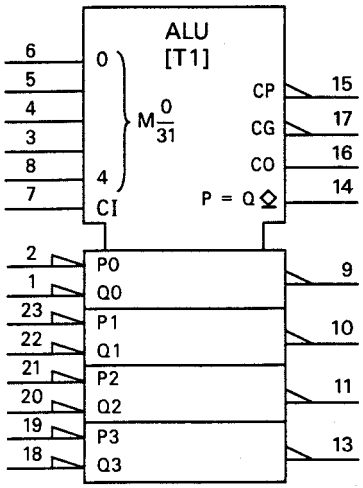
No	Symbol	Description
5.6-8		Demultiplexer/decoder, universal, dual NOTE — In order to perform the alternative 1:8 demultiplexing function correctly, it is necessary to make an external connection between pins 19 and 20, and also between pins 23 and 22. See NOTE 2 to Symbol 5.1-25 and NOTE to Symbol 5.6-2. *F100170*
5.6-9		Analog data selector (multiplexer/demultiplexer), 4-channel, dual See NOTE to Symbol 5.2-9. NOTE — When using the general qualifying symbol MUXDX, the identifying numbers of the X dependencies may be omitted at the multiplexed input/output port if no confusion is likely. See NOTE to Symbol 5.6-2. *MC14529B*

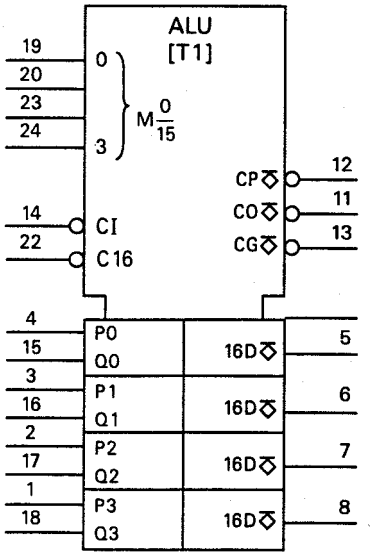
5.7 Arithmetic elements

No	Symbol	Description
5.7-1		Arithmetic element, general symbol The * should be replaced by one of the qualifying symbols from the following list: ΣAdder P-QSubtractor CPGLook-ahead carry generator (carry, propagate, and generate) ΠMultiplier COMPMagnitude comparator A cascadable comparator is assumed to implement a portion of a comparison that proceeds from lower to higher order, unless otherwise indicated, for example, by "[H-L]" placed below the "COMP" qualifying symbol. ALUArithmetic logic unit The general qualifying symbol ALU needs supplementary information to specify the function of the element.
5.7-2		Half adder
5.7-3		Single-bit full adder NOTE — A simple single-bit full adder may alternatively be depicted by the combination of the symbols for the ODD element (modulo 2) and the logic threshold element as shown below: 

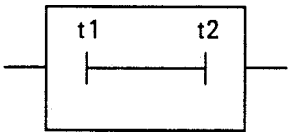
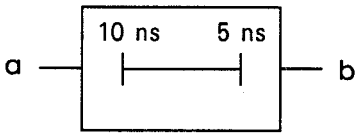
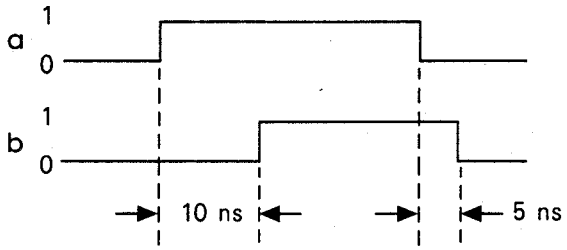
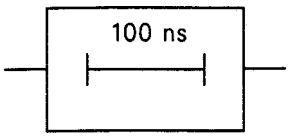
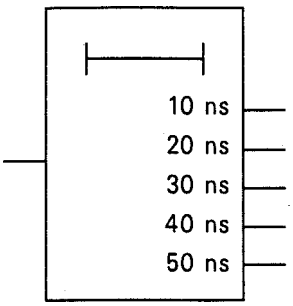
No	Symbol	Description
5.7-4		Single-bit full adder with complementary sum outputs and inverted carry output *SN7480*
5.7-5		Full adder, 4-bit NOTE — Symbol 5.7-6 depicts the same device in another way. *SN74283*
5.7-6		Full Subtractor, 4-bit NOTE — Symbol 5.7-5 depicts the same device in another way. *SN74283*
5.7-7		Look-ahead carry generator, 4-bit *SN74182*

No	Symbol	Description
5.7-8		Multiplier, 4-bit parallel, generating the four least significant bits of the 8-bit product *SN74285*
5.7-9		Multiplier, 4-bit parallel, generating the four most significant bits of the 8-bit product *SN74284*
5.7-10		Comparator with NPN open-collector output, 6-bit *DM7160*

No	Symbol	Description
5.7-11		Magnitude comparator, low-order to high-order, with cascading inputs, 4-bit *SN7485*
5.7-12		Magnitude comparator with 3-state outputs, 4-bit *DM76L24*
5.7-13		Arithmetic logic unit, 4-bit NOTES: 1 — T1 refers to supplementary documentation detailing the element's function in various modes. 2 — The Ms at the outputs of the ALU have been omitted in accordance with 4.3.10. *SN74181*

No	Symbol	Description
5.7-14		Arithmetic logic unit with output latches, 4-bit See NOTE to Symbol 5.1-25 and Symbol 5.7-13. *F100181*

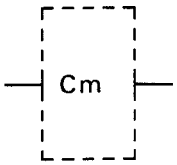
5.8 Delay elements

No	Symbol	Description
5.8-1		Delay element with specified delay times, general symbol A transition from the internal 0-state to the internal 1-state at the output occurs after a delay of t1 with reference to the same transition at the input. The transition from the internal 1-state to the internal 0-state at the output occurs after a delay of t2 with reference to the same transition at the input. NOTE — t1 and t2 may be replaced by the actual delays, expressed in seconds, word units, or digit units, and may be placed inside or outside the outline. If the two delays are equal, it is sufficient to insert one value only.
	 	
	Illustration of the concept	
5.8-2		Delay element (100 ns)
5.8-3		Tapped delay element (in steps of 10 ns)

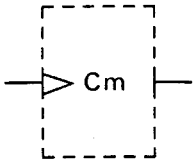
5.9 Basic bistable elements

The symbol for a bistable element (for example, a flip-flop) does not contain a general qualifying symbol. The function of the element is indicated by the qualifying symbols associated with its inputs and outputs. The internal logic states of the outputs of a bistable element represent the content of the element.

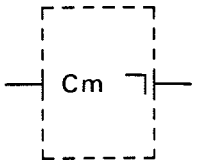
When a bistable element is controlled by a C input (Symbol 4.3.7-1) it is necessary to indicate whether this element is a latch, or an edge-triggered, pulse-triggered, or data-lock-out bistable. For this purpose, use can be made of the symbols for a dynamic input (Symbol 3.1-9) and for a postponed output (Symbol 3.3-1). According to the descriptions of these symbols:



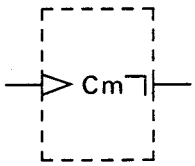
indicates a latch



indicates an edge-triggered bistable



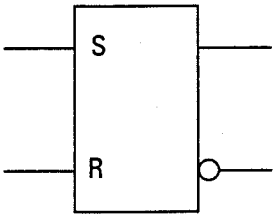
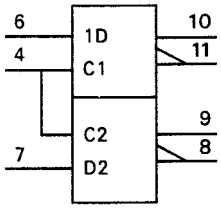
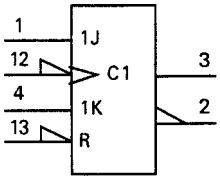
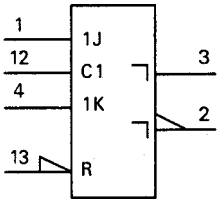
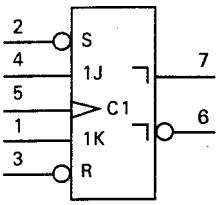
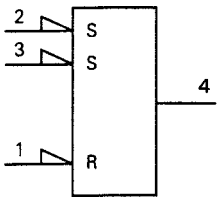
indicates a pulse-triggered bistable

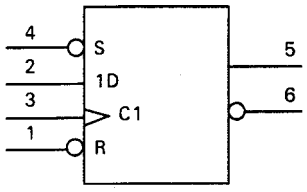
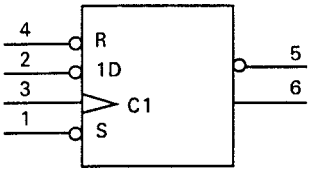
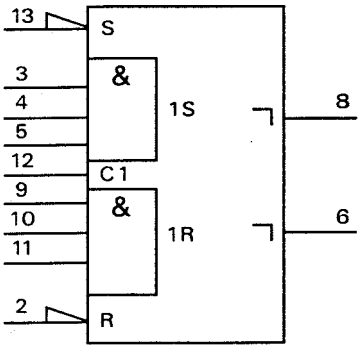
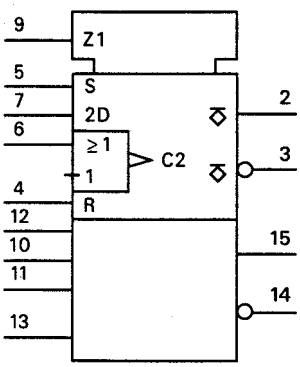
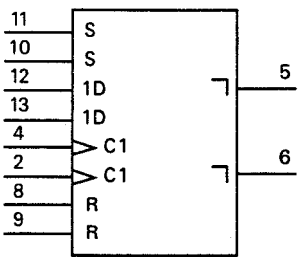


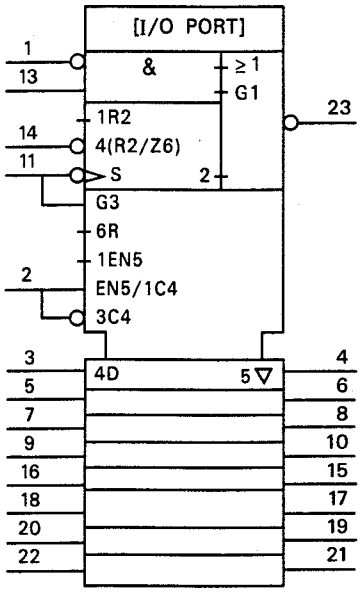
indicates a data-lock-out bistable

In the edge-triggered, pulse-triggered, and data-lock-out bistables, the inputs affected by the C input are assumed to be stable during the period that the C input stands at its internal 1-state. If they do change their states during this period, the function of the element is not specified by the symbol.

The same symbology is used for more complex elements such as shift registers and counters to indicate whether they are of the edge-triggered, the pulse-triggered, or the data-lock-out type. If, in elements of the pulse-triggered or the data-lock-out type, reference is made to the content of the element, this content shall be considered as being the content after the application of the postponed output symbol.

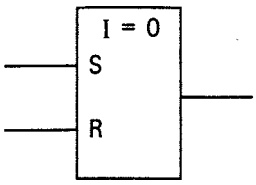
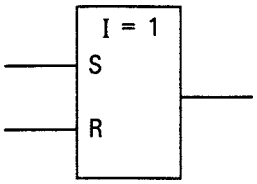
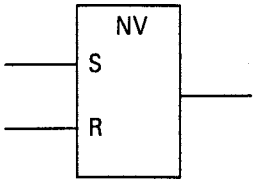
No	Symbol	Description
5.9-1		S-R bistable element
5.9-2		D-type latch, dual *Part of SN7475*
5.9-3		Edge-triggered J-K bistable *Part of SN74LS107*
5.9-4		Pulse-triggered J-K bistable *Part of SN74107*
5.9-5		Data-lock-out J-K bistable *Part of SN74111*
5.9-6		S-R latch with active-low inputs (S-R latch) *Part of SN74279*

No	Symbol	Description
5.9-7		Edge-triggered D-type bistable *Part of SN7474*
5.9-8		Edge-triggered D-type bistable, complementary form *Part of SN7474*
5.9-9		Pulse-triggered S-R bistable *SN74L71*
5.9-10		Edge-triggered D-type bistable, dual *MC10131*
5.9-11		Edge-triggered D-type bistable *MC1222* NOTE — Four implicit OR relationships exist by the rules of 3.3.1 and 4.3.1.

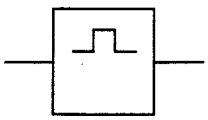
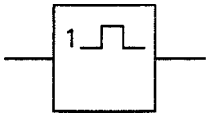
No	Symbol	Description
5.9-12		Multiplexer with storage, quad 2-input NOTE — "M1" at pin 10 may be replaced by "G1" *SN74298*
5.9-13		Input/output port, 8-bit *8212*

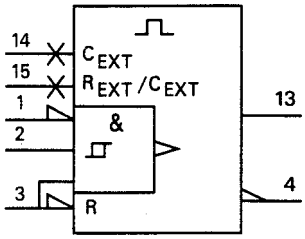
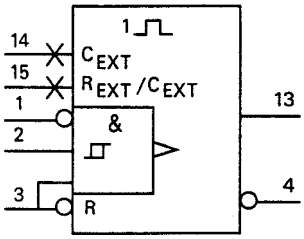
5.10 Bistable elements with special switching properties

In some special applications (for example, fail-safe systems) it is necessary to indicate which state the outputs of a bistable element will take on when the supply is switched on. The symbols below show how this may be done. The qualifying symbol may be applied to other types of bistable elements.

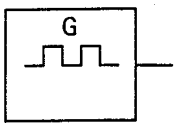
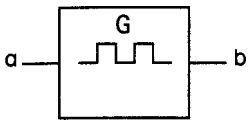
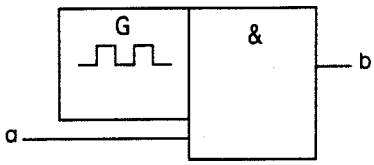
No	Symbol	Description
5.10-1		S-R bistable with initial 0-state At the moment the supply is switched on, the element will be in the 0-state.
5.10-2		S-R bistable with initial 1-state At the moment the supply is switched on, the element will be in the 1-state.
5.10-3		S-R bistable, nonvolatile At the moment the supply is switched on, the element will be in the same state as when the supply was switched off.

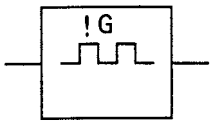
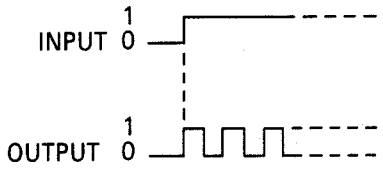
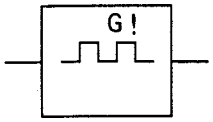
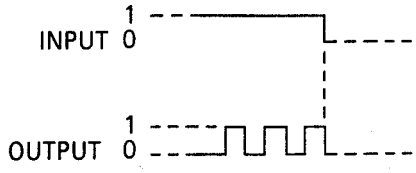
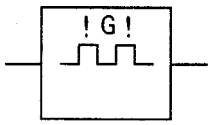
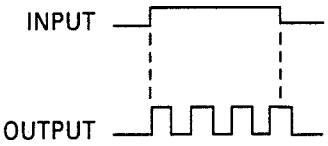
5.11 Monostable elements

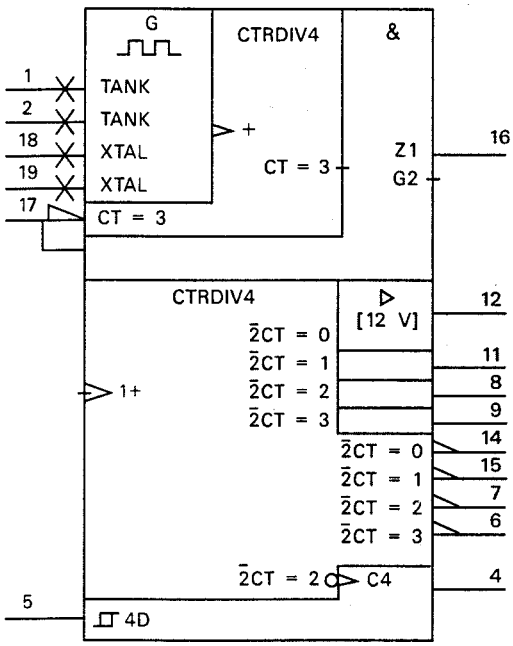
No	Symbol	Description
5.11-1		Retriggerable monostable (retriggerable during the output pulse), general symbol One shot, general symbol Single shot, general symbol The output changes to or remains at its 1-state each time the input changes to its 1-state. The output returns to its 0-state after a period of time that is characteristic of the particular device, beginning at the last change of the input to its 1-state. NOTE — The use of the dynamic input symbol (Symbol 3.1-9) at the input is optional. For example, see Symbol 5.11-3.
5.11-2		Nonretriggerable monostable (nonretriggerable during the output pulse), general symbol The output changes to its 1-state only when the input changes to its 1-state. The output returns to its 0-state after a period of time that is characteristic of the particular device, regardless of any changes of the input variable during this period. NOTE — The use of the dynamic input symbol (Symbol 3.1-9) at the input is optional. For example, see Symbol 5.11-4.

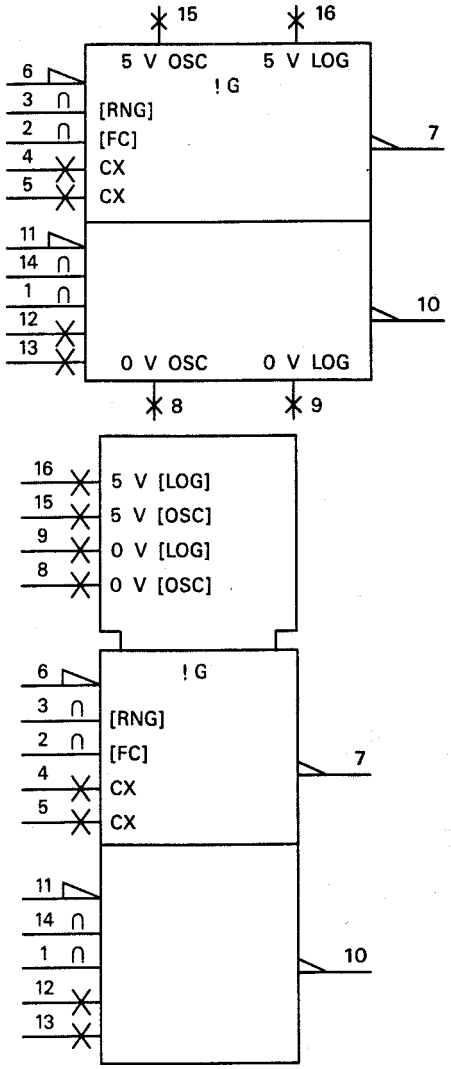
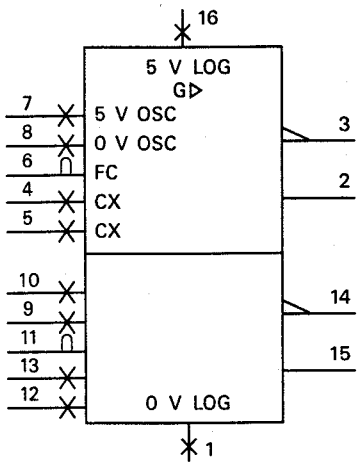
No	Symbol	Description
5.11-3		Monostable, retriggerable *Part of SN74LS123*
5.11-4		Monostable, nonretriggerable *Part of SN74LS221*

5.12 Astable elements

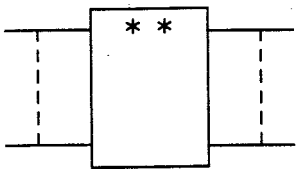
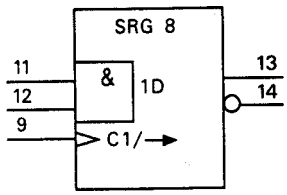
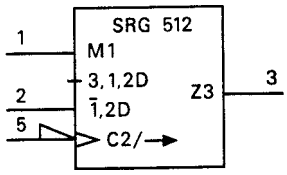
No	Symbol	Description
5.12-1		Astable element, general symbol Signal generator producing an alternating sequence of zeroes and ones. NOTE — In this symbol, th G is the qualifying symbol for generator. If the waveform is evident, it need not be shown.
5.12-2		Controlled astable element, general symbol See NOTE to Symbol 5.12-1.  **Explanatory diagram**

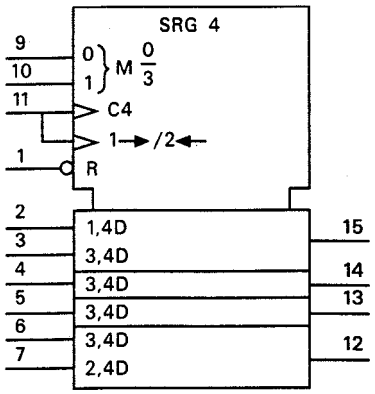
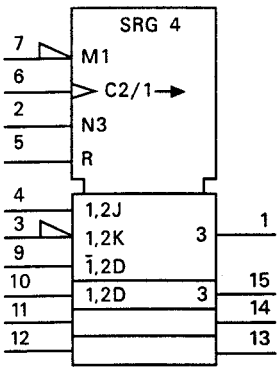
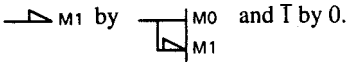
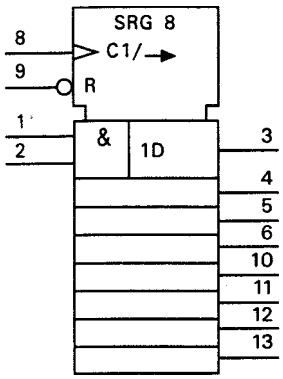
No	Symbol	Description
5.12-3		Astable element, synchronously starting, general symbol The output starts with a complete pulse at the instant at which the input takes on its internal 1-state. See NOTE to Symbol 5.12-1. 
5.12-4		Astable element stopping after completing the last pulse, general symbol When the input returns to its internal 0-state, the output remains at its internal 0-state or completes its final pulse. See NOTE to Symbol 5.12-1. 
5.12-5		Astable element, synchronously starting, stopping after completing the last pulse, general symbol See NOTE to Symbol 5.12-1. 

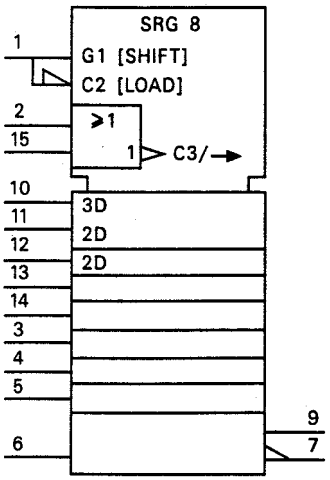
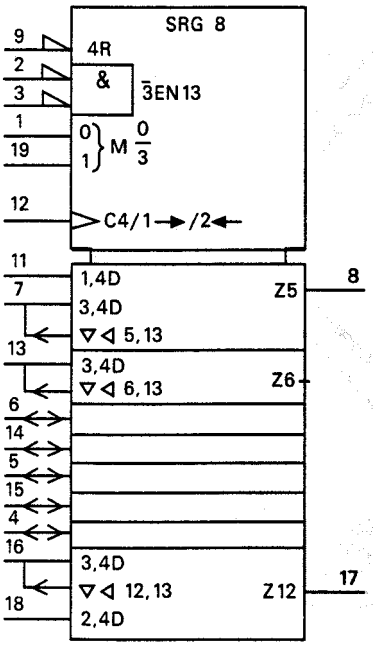
No	Symbol	Description
5.12-6		Clock generator/driver, four-phase NOTES: 1 — For the use of CTRDIV4, see Symbol 5.13-1. 2 — Symbol 6.3-4 depicts the same device as a “gray box.” *TIM9904 (formerly SN74LS362)*

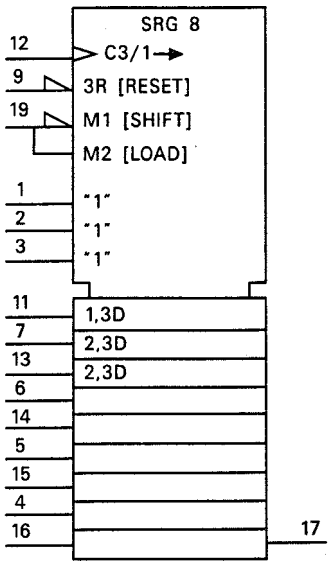
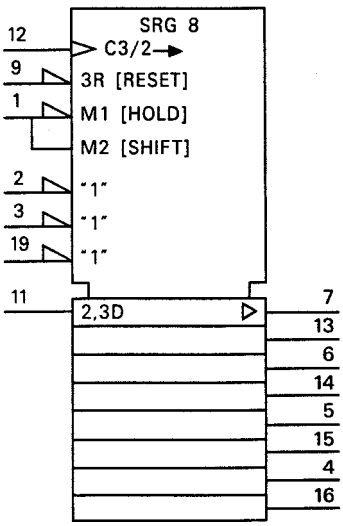
No	Symbol	Description
5.12-7		Voltage-controlled oscillator, dual NOTE — These symbols illustrate two methods of showing supply terminals common to an array of elements. *SN74S124*
5.12-8 ●		Voltage-controlled oscillator, dual (with independent oscillator supply voltage terminals for each section) *SN74LS625*

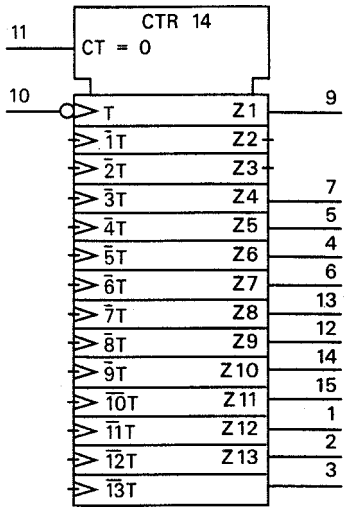
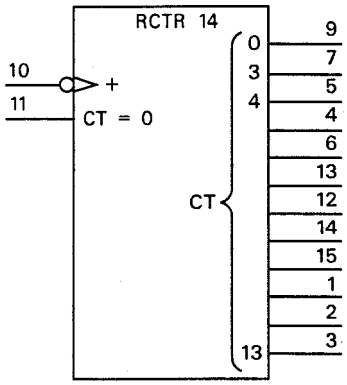
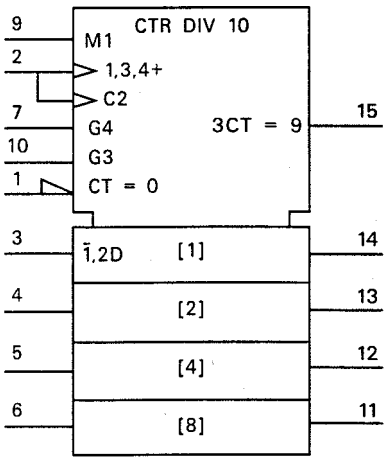
5.13 Shift registers and counters

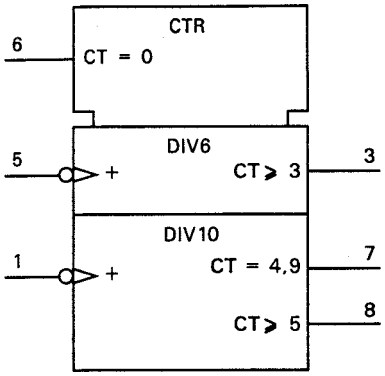
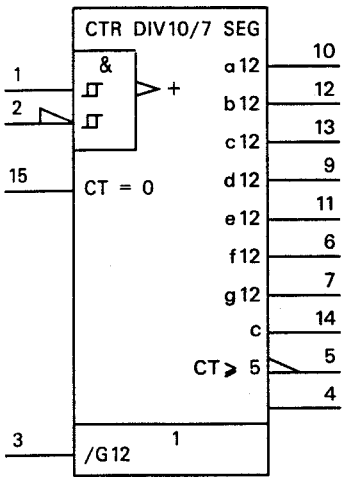
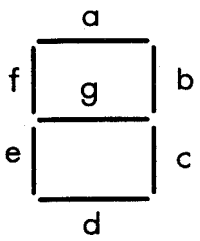
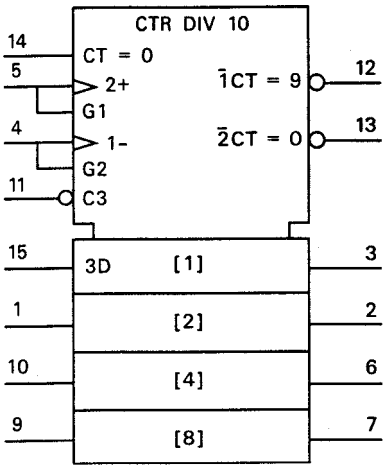
No	Symbol	Description
5.13-1		Shift register or counter, general symbol The "**" shall be replaced by the general qualifying symbol for the function, that is: SRG* Shift register; the "*" shall be replaced by the number of stages CTR* Counter with cycle length equal to 2 to the power represented here by * (Counter modulo 2 to the power *) RCTR* Ripple counter with cycle length equal to 2 to the power represented here by * (Ripple counter modulo to the power *) CTRDIVm Counter with cycle length equal to m (Counter modulo m) RCTRDIVm Ripple counter with cycle length equal to m (Ripple counter modulo m) NOTES: 1 — m in CTRDIVm and RCTRDIVm shall be replaced by the actual value. 2 — In an array of elements having different cycle lengths, the cycle length applying to each should be indicated by DIVm in each element. In such a case, the letters CTR or RCTR need only be shown in the common control block. For example, see Symbol 5.13-14.
5.13-2		Shift register, 8-bit, with serial input and complementary serial outputs *Part of SN7491*
5.13-3		Shift register, 512-bit, static *MM4057*

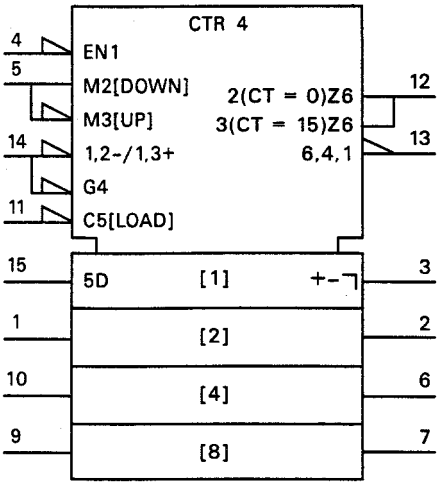
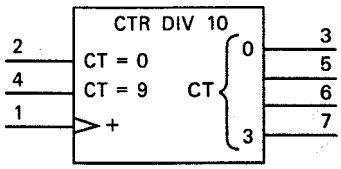
No	Symbol	Description
5.13-4		Shift register, 4-bit, bidirectional universal *SN74194*
5.13-5		Shift register, 4-bit, parallel in/parallel out NOTE — The use of the bar can be avoided by replacing  *CD4035A*
5.13-6		Shift register, 8-bit, with parallel outputs *SN74164*

No	Symbol	Description
5.13-7		Shift register with parallel load, 8-bit *SN74165*
5.13-8		Universal shift register, 8-bit NOTE — Symbols 5.13-9 and 5.13-10 show this same device in simplified versions that do not show all functions. *SN74LS323*

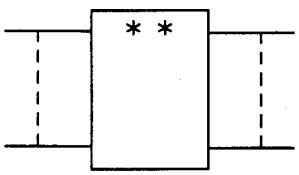
No	Symbol	Description
5.13-9		Universal shift register, 8-bit, for which only the reset, shift, and parallel-load modes are used NOTES: 1 — Use of a complex device in a circuit may leave some of the capability of the device unused. This symbol illustrates how an incompletely utilized device may be represented so that its symbol need only be as complicated as its application requires. Use is made of the fixed-mode input, Symbol 3.3-39. 2 — Symbol 5.13-8 shows this same device in full detail. Symbol 5.13-10 depicts the same device performing another function. *Incompletely utilized SN74LS323*
5.13-10		Universal shift register, 8-bit, for which only the reset, hold, and shift modes are used NOTE — Symbol 5.13-8 shows this same device in full detail. Symbol 5.13-9 depicts the same device performing another function. See NOTE 1 to Symbol 5.13-9. *Incompletely utilized SN74LS323*

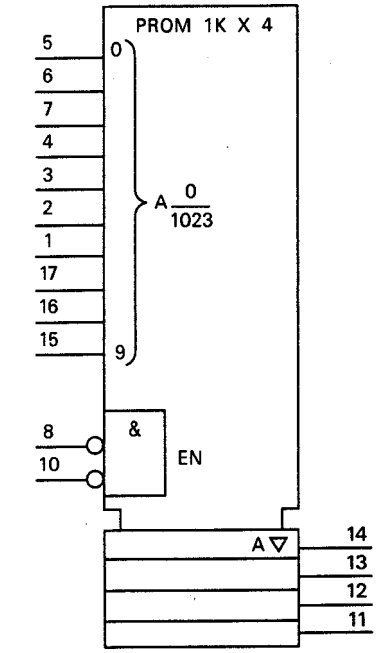
No	Symbol	Description
5.13-11		Binary ripple counter, 14-stage NOTE — Symbol 5.13-12 depicts the same device in another way. *CD4020*
5.13-12		Binary ripple counter, 14-stage NOTE — Symbol 5.13-11 depicts the same device in another way. *CD4020*
5.13-13		Counter, synchronous, decade, 4-bit, with parallel load *SN74LS160*

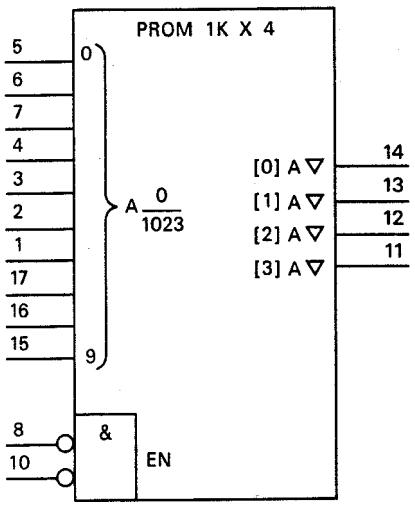
No	Symbol	Description
5.13-14		Counters, one dividing by 5 and 10, and the other by 6 *SN74LS57*
5.13-15		Decade counter/divider with decoded 7-segment display outputs  **SEGMENT IDENTIFICATION** *CD4026A*
5.13-16		Counter, decade, 4-bit, synchronous up/down *SN74192*

No	Symbol	Description
5.13-17		Binary counter, 4-bit, synchronous up/down *SN74191*
5.13-18		Decimal counter, 4-bit, synchronous *Part of SN74490*

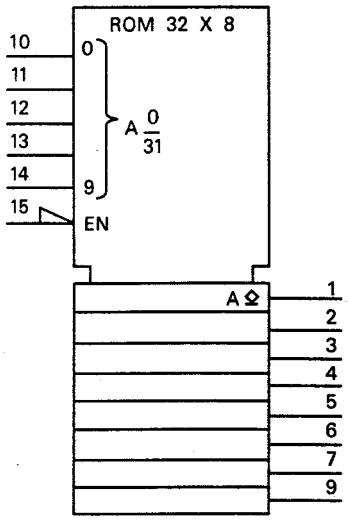
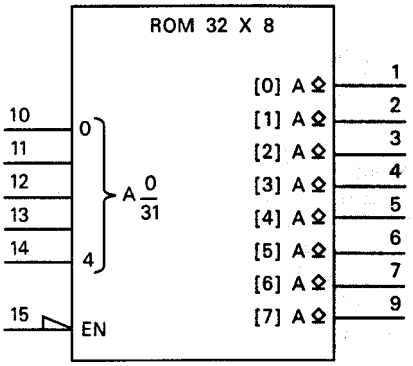
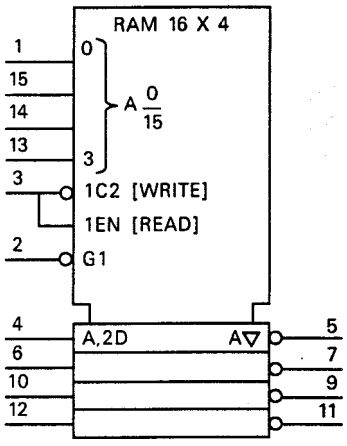
5.14 Memories

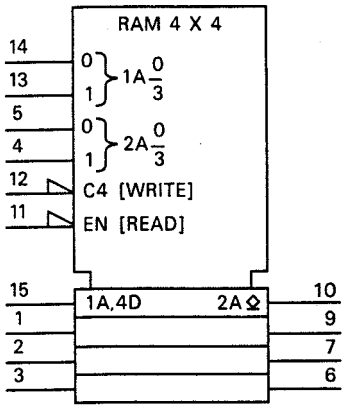
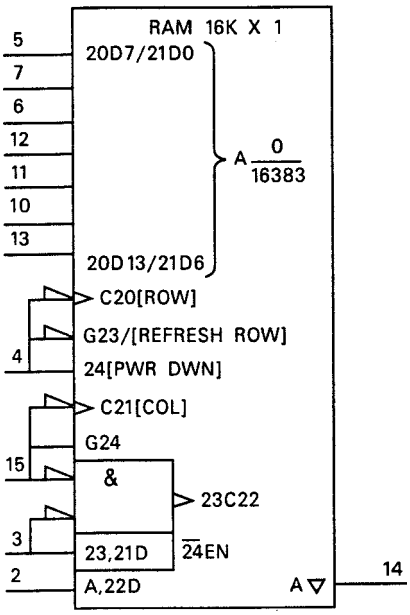
No	Symbol	Description
5.14-1		Memory, general symbol The "**" shall be replaced by the general qualifying symbol for the function, for example, ROM* Read-only memory PROM* Programmable read-only memory RAM* Random-access memory (read/write memory) CAM* Content-addressable memory (associative memory) FIFO* First-in first-out memory (see Symbol 5.14-7) The "*" must be replaced by an appropriate indication of the number of addresses and bits.

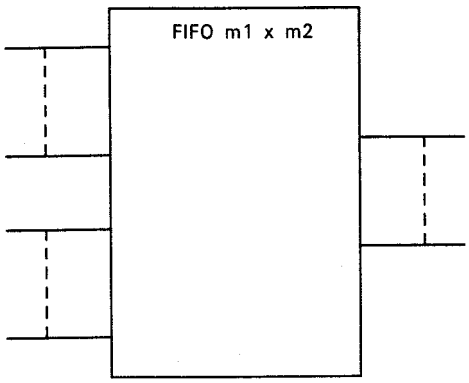
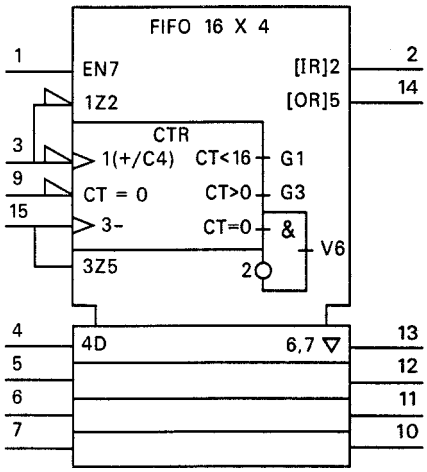
No	Symbol	Description
5.14-2		Programmable read-only memory 1024x4 bits. *3625*

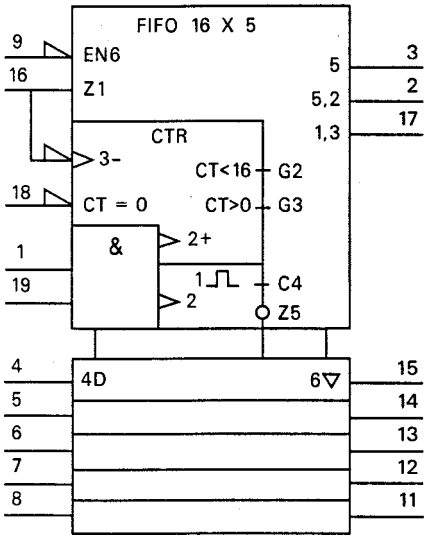
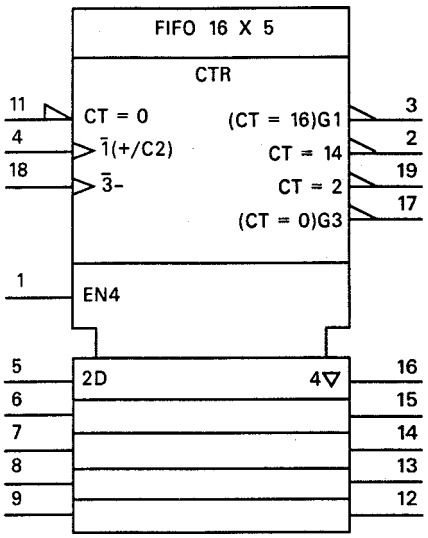


NOTES:
1 — Because no confusion can arise concerning the correspondence between data inputs and data outputs, and because each section has only one output, it is not necessary to show the array.
2 — If the simplified version is used, some method is usually necessary to identify the relative ordering of the individual outputs (for example, the bracketed numbers shown here).

No	Symbol	Description
5.14-3		Read-only memory, 32x8 bits
		See NOTES to Symbol 5.14-2
		*SN7488*
5.14-4		Random-access memory, 16x4 bits
		*SN74S189*

No	Symbol	Description
5.14-5		Random-access memory, 4x4 bits, with separate write and read addresses *SN74170*
5.14-6		Random access memory, 16 384 X 1 bit with refresh *TMS4116*

No	Symbol	Description
5.14-7		First-in, first-out memory, general symbol Each of the associated elements represents a single general case of the section of a multidimensional array in the same way as is described with address dependency. The internal logic states of the m2 data outputs correspond to the values of the bits of the word first entered of those words presently stored. As each word is read out, subsequently stored words become available at the outputs in the order in which they were entered. If no words are presently stored, the internal logic states of the data outputs are not specified by the general qualifying symbol. NOTES: 1 — m1 shall be replaced by the maximum number of words that can be stored. m2 shall be replaced by the number of data outputs. 2 — This symbol will normally require a counter to be shown as an embedded symbol, although this may not necessarily represent the actual implementation. The content of the counter represents the number of words presently stored, that is, the number of words that have been entered less the number that have been removed. The indication of the cycle length of the counter is omitted from the CTR qualifying symbol because, by definition, the counter cannot be incremented beyond m1 to start a new cycle.
5.14-8		First-in, first-out memory, counter-controlled type *SN74LS224*

No	Symbol	Description
5.14-9		First-in, first-out memory, fall-through type *SN74S225*
5.14-10		First-in, first-out memory, counter-controlled type *SN74ALS229*

5.15 Display elements

It should be recognized that the visual (optical) signals produced by display elements, e.g., LED, LCD, bar, or dot-matrix elements, are external outputs of those elements.

For the representation of complex-function display elements, see Chapter VI.

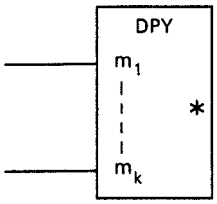
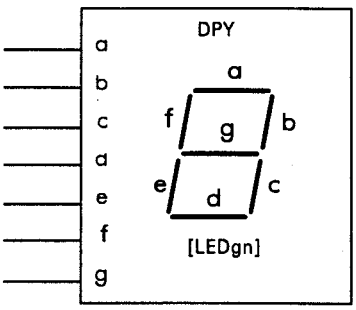
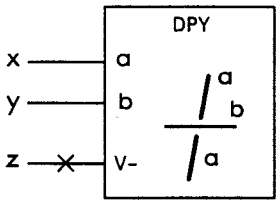
No	Symbol	Description
5.15-1		Display element, general symbol NOTES: 1 — The asterisk shall be replaced either by: 1)an appropriate indication of the display and/ or by 2)a reference to a table. 2 — The elements that make up the display shall always be shown in their correct physical positions relative to each other. Rotation of symbols to preserve the relative orientation of the display with respect to the reader is sometimes advisable. 3 — m1...mk shall each be replaced either by: 1)an appropriate indication of the visual signals controlled by those inputs or by 2)appropriate designations referring to entries in a table. If reference is made to a table in which the terminal designations are used to identify the inputs, these designations may be omitted.

Illustration of the concept

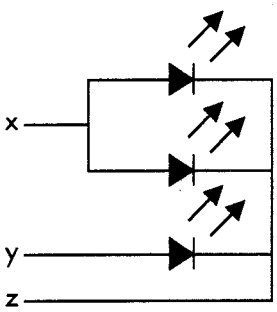


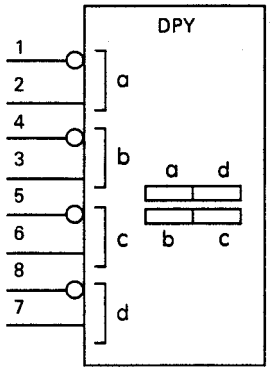
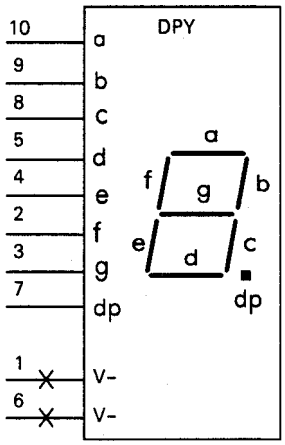
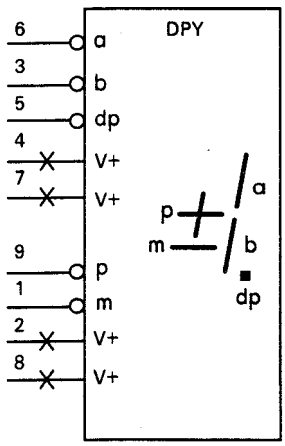
The designations a…g at the inputs refer to the visual signals (segments) on the display.

The indications a…g at the segments may be omitted.



could be



No	Symbol	Description
5.15-2		LED light bars *HLMP 2600*
5.15-3		Seven-segment display *HDSP 3603*
5.15-4		Overflow display *HDSP 5607*

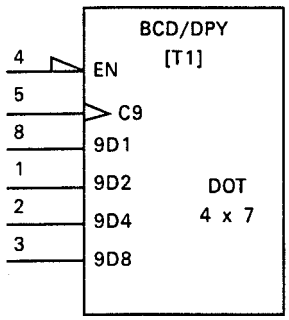
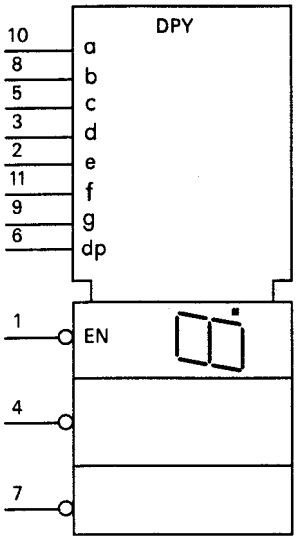
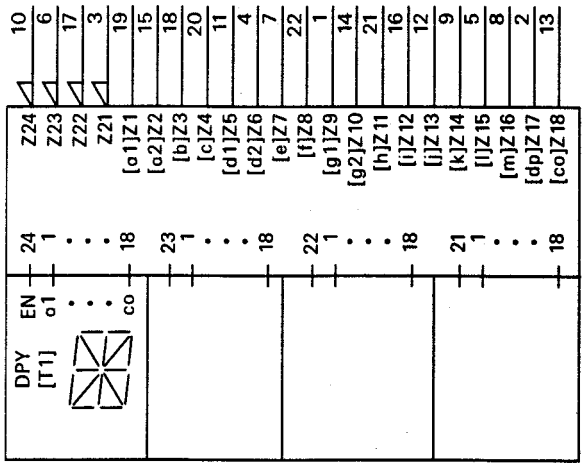
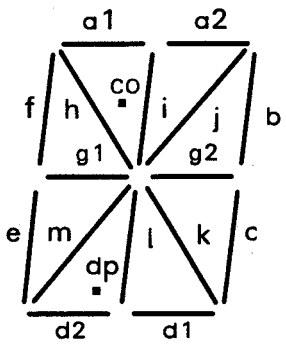
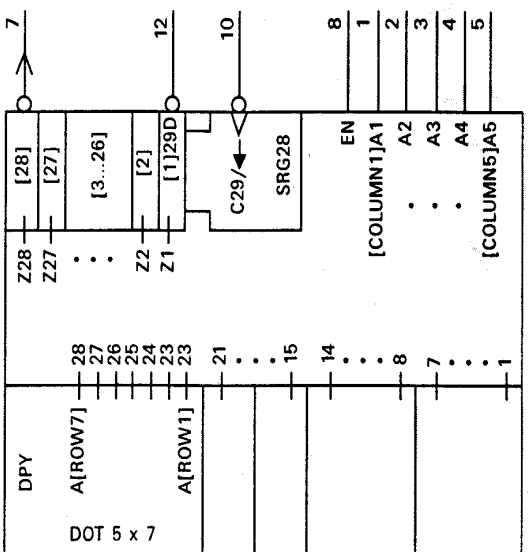
No	Symbol	Description
5.15-5		Hexadecimal display NOTES: 1 — [T1] refers to a font table describing the decoding from BCD to a dot matrix 4x7. 2 — The visual signals are visible only if the EN-input stands at its internal 1-state. The latching of the data is not affected by the EN-input. *5082-7340*
5.15-6		Numeric display, three 7-segment characters with decimal points *5082-7433*
5.15-7	 *HDSP 6504*	Alphanumeric display, four 16-segment characters

Table T1
Segment Identification



No	Symbol	Description
5.15-8		Alphanumeric display, four 5 x 7-dot characters *HDSP 0*

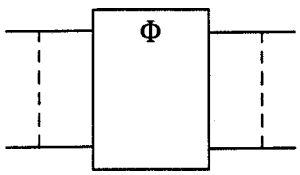
6. Symbols for highly complex functions

General note

Highly complex functions are often required to be represented by a single symbol. One means of representation is by the combination of outlines, input/output labeling, and the use of dependency notation as described in Sections 2 through 4 and applied in Section 5. However, whereas this technique is suitable for small-scale and most medium-scale integrated circuits, it often becomes impractical for more complex circuit assemblies, such as large-scale and very-large-scale integrated circuits, due to the number of functions involved.

If the application of the rules of Sections 2 through 5 results in symbols too cumbersome to use effectively, the following technique may be used.

6.1 General symbol and basic rules

No	Symbol	Description
6.1-1		Highly complex function, general symbol (“gray box”) The Φ shall be supported by an indication of the function, preferably short. In addition, a reference to supporting documentation (for example, a type number) shall be included within or adjacent to the symbol outline.

6.1.1 General

All of the rules and concepts of the previous sections may be employed. However, dependency notation may be used only if no confusion with other labeling is likely.

6.1.2 Input and output designation

Inside the symbol outline, all inputs and outputs should be designated with the terminal names appearing on the selected data sheet or other documentation referenced in the description of symbol 6.1-1 of a selected manufacturer or similar supporting document. This data sheet or other documentation should preferably be one that uses terminal names from a terminal-naming or signal-naming standard. Further abbreviation of these names should be considered only if these names are inconveniently long. For clarity, abbreviated terminal names may be expanded or supplemented. If labels defined in Sections 3 through 5 are used on the data sheet with a meaning other than that defined in those sections, such as labels shall be expanded to prevent confusion (for example, DBUS instead of D).

In cases where the manufacturer's labeling prevents the use of the bit-grouping symbol for a clear representation of a bus, these labels may be modified, provided correspondence with the data sheet is still possible.

6.1.3 Negated terminal names

Negated terminal names shall be shown as described in IEEE Std 991-1986.

Negated terminal names may be converted to the unnegated form inside the symbol by using the negation or polarity symbols, depending on the convention in force. If an input or an output serves two functions that are activated at opposite polarities, a branch on the connecting line may be used to permit two separate labels to be shown, thus avoiding a negation bar.

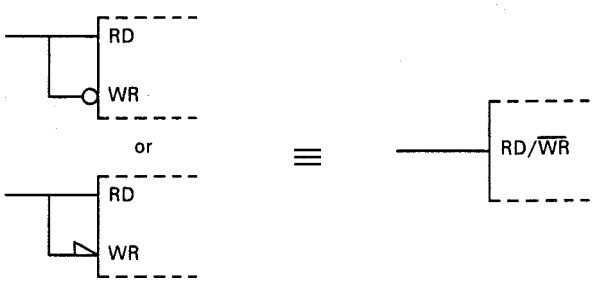


Illustration of the concept

6.1.4 Functional grouping

The connecting lines should be functionally grouped and, where appropriate, be partitioned into control and data lines. The control lines may appear on the "control block outline" for which the common control block outline, as described in 2.3.2, is used.

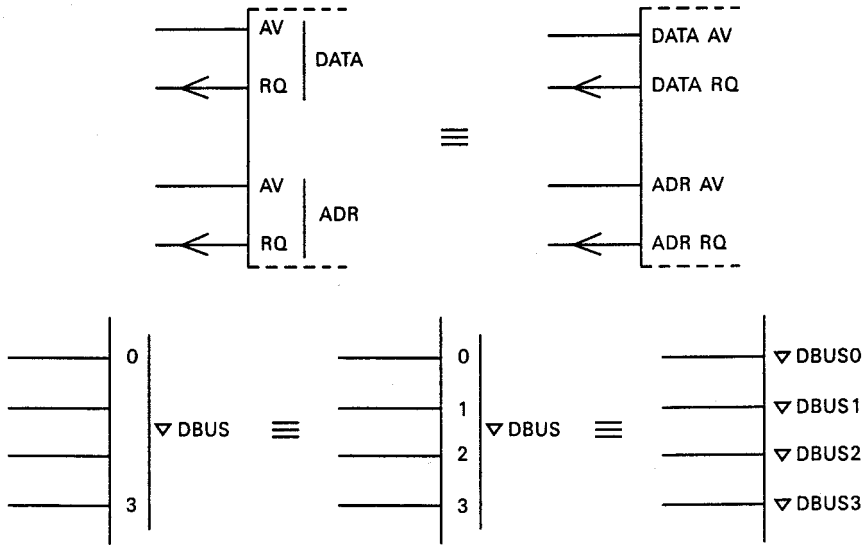
6.1.5 Label grouping

A vertical line may be employed inside the symbol outline to group adjacent and associated connecting lines whose labels are partially alike. The common portion of these labels is then placed only once at one side of this vertical line whereas the individual portions are placed against the connecting lines at the other side of the vertical line.

If the individual portions are numbers, intermediate numbers within consecutive groups may be omitted to the extent that no confusion is likely.

This rule may be applied in cases where the bit-grouping symbol is not applicable because the inputs our outputs grouped together do not produce or represent a number. For example, see Symbol 6.3-3.

Illustration of the concept



6.1.6 Long character strings

Long character strings associated with input or output lines may be narrowed (at the expense of height) by inclusion in an open box as shown below. The box shall open away from the input or output line. The broken character string shall be justified flush against the closed side, taking into account embedded spaces. To avoid ambiguity as to the continuity of a negation bar, multiple characters under a single bar shall not be split. Strings should not be broken in such a way that readability is impaired or an intended space is lost.

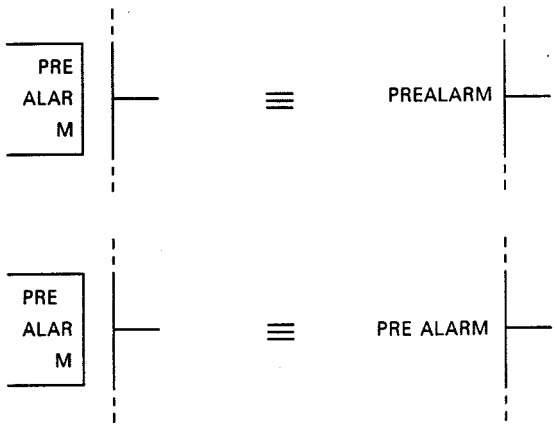


Illustration of the concept

NOTE — PREALARM is derived from “before alarm.”
PRE ALARM is derived from “preset alarm.”

6.1.7 Consecutive labels and terminal numbers

If both the internal labels and the terminal numbers are consecutive, then grouping of inputs [outputs] may be simplified by showing only the first and last connecting lines and their respective labels, separated by dots or short vertical lines outside and inside the symbol outline. The dots or short vertical lines inside the symbol may be omitted if no confusion is likely.

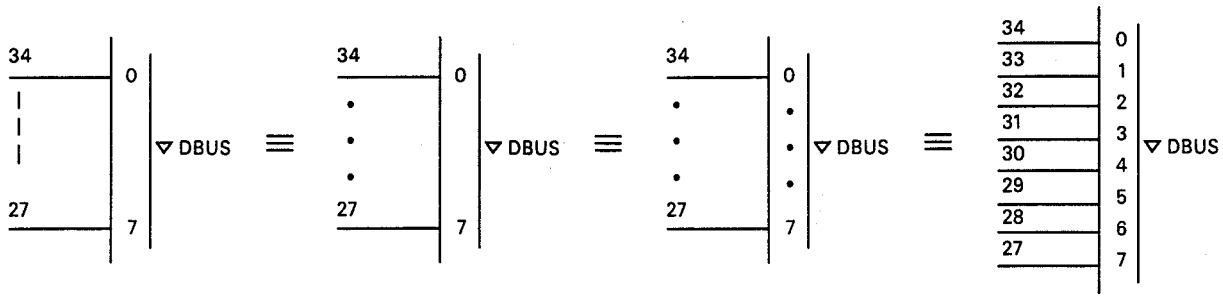


Illustration of the concept

6.1.8 Function tables and truth tables

When tables are used to provide additional information about the behavior of the circuit, the table entries should refer to logic levels or to external logic states.

1) If the table entries refer to logic levels, the entries may be identified in any unambiguous way.
2) If the table entries refer to external logic states on a theoretical logic diagram or when using a single logic convention, then, in the table, any label derived from one appearing inside the symbol at an input or output bearing a negation symbol shall be modified by adding (or removing) a negation bar. All other labels should appear on the table without modification. For example, see Symbol 6.1.9-2.

6.1.9 Bus indicators

In many cases, the use of a symbol for a bus can clarify the function of a complex element.

No	Symbol	Description
6.1.9-1		Bus indicator, unidirectional, shown for signal flow from left to right NOTES: 1 — If a bus name or a common portion of the labels for the terminals is shown, that common name or portion should be placed inside the bus indicator. 2 — If associated with terminals, a bit-grouping symbol (Symbol 3.3-25 or Symbol 3.3-26) or a label grouping symbol (see 6.1.5) as appropriate shall be shown between the bus indicator and the symbol outline. Then the connecting lines grouped together need no arrowheads to indicate the direction of signal flow.
6.1.9-2		Bus indicator, bidirectional See Notes with Symbol 6.1.9-1.

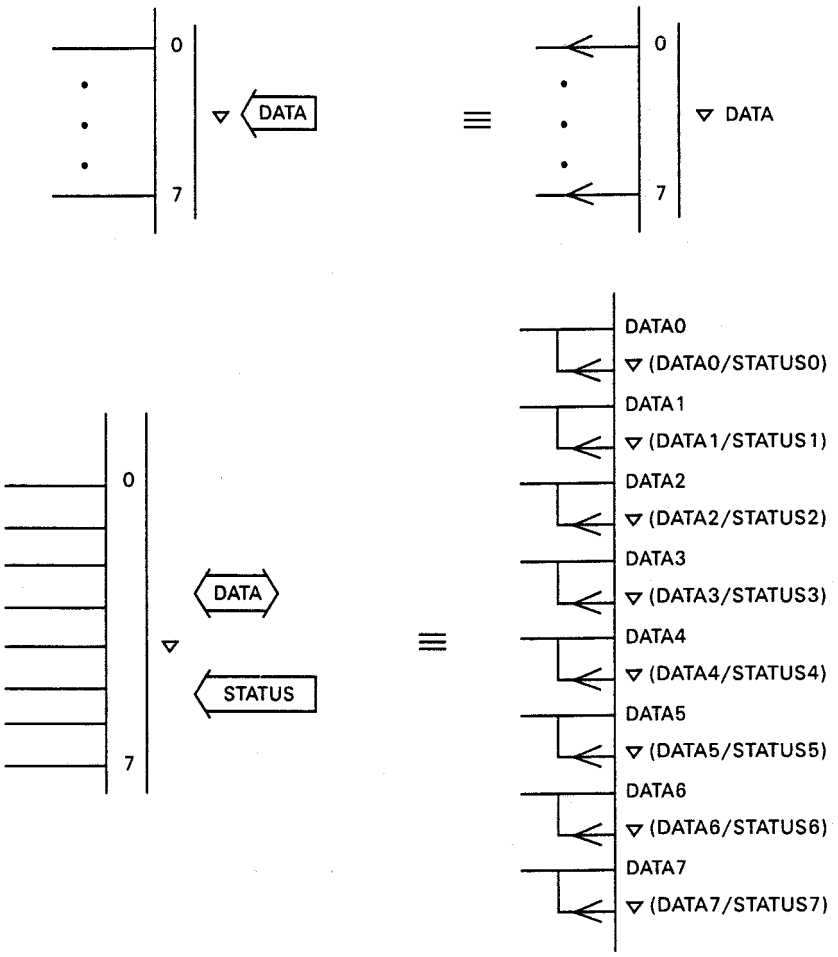


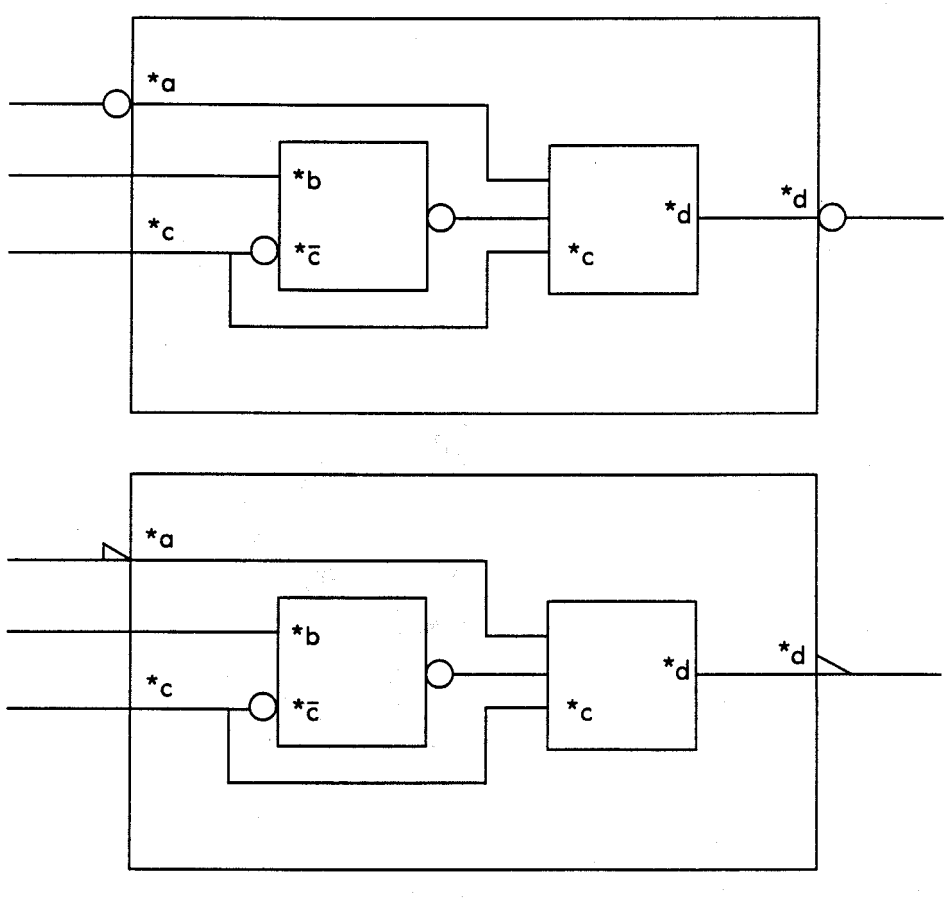
Illustration of the concept

6.2 Internal diagrams

6.2.1 General

A diagram may be used inside the outline of a symbol (an internal diagram) to depict the behavior of a complex function. In this case, the following rules shall be observed.

1) The negation or polarity symbol shall be shown at the symbol outline at those inputs [outputs] to which it applies to indicate the relationship between the internal logic state of the input [output] and its external logic state or logic level.
2) By definition, logic states, and not logic levels, exist within the outline of a symbol. Therefore, the symbol for logic polarity cannot be used on an internal diagram, and the symbol for logic negation shall be applied where appropriate.
3) Input and output labels shall be shown inside and adjacent to the symbol outline, inside and adjacent to the outlines of the symbols appearing on the internal diagram, or in both places. Labels containing identifying numbers in the sense of dependency notation that apply to elements of the internal outline shall be shown only inside the outlines of the internal symbols to which they apply. When such labels are repeated after the application of logic negation, they shall be modified by adding (or removing) a negation indication.
4) Connections solely between elements of the internal diagram need not be labelled.

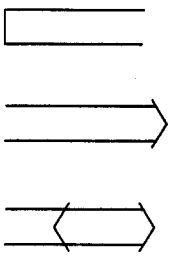


*a … *d represent four different labels.

Illustration of the concept

6.2.2 Data-path representation

The technique used for bus indicators (see 6.1.9) may be extended to represent data paths (buses) on an internal diagram such as those shown in Symbols 6.3-6 and 6.3-7. Portions of a data path may be shown as follows:

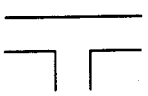


Source of a unidirectional data path

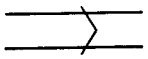
Destination of a unidirectional data path

Source and destination of a bidirectional data path

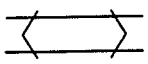
NOTE—The separation of the arrowheads is arbitrary, but they shall both appear on the same branch of a junction.



Junction of data paths

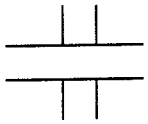


Indication of data flow direction



Indication of bidirectional data flow

NOTE—The separation of the arrowheads is arbitrary but they shall both appear on the same branch of a junction.



Crossing without connection

If no source is shown, all branches are assumed to be bidirectional.

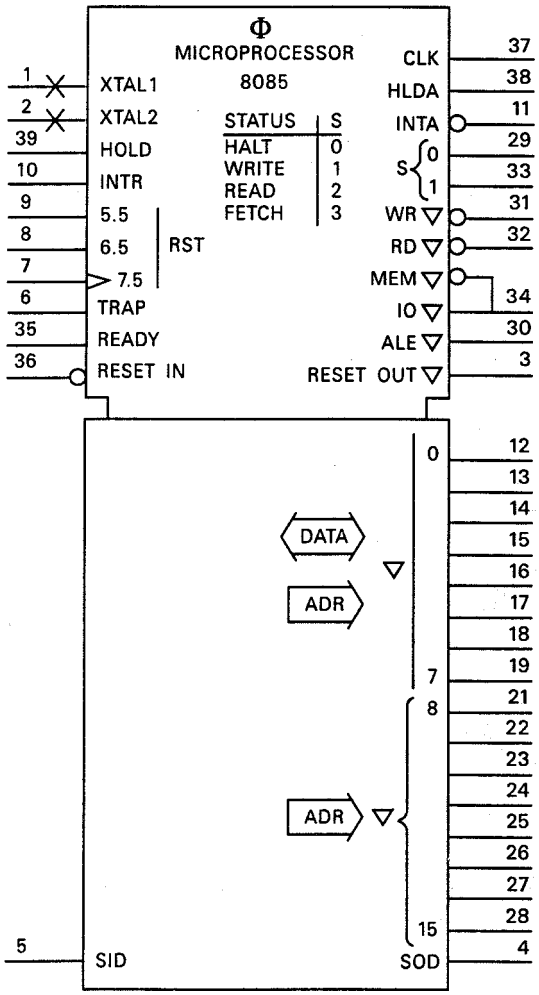
The names and/or the width of the data path may be indicated inside or adjacent to the data path.

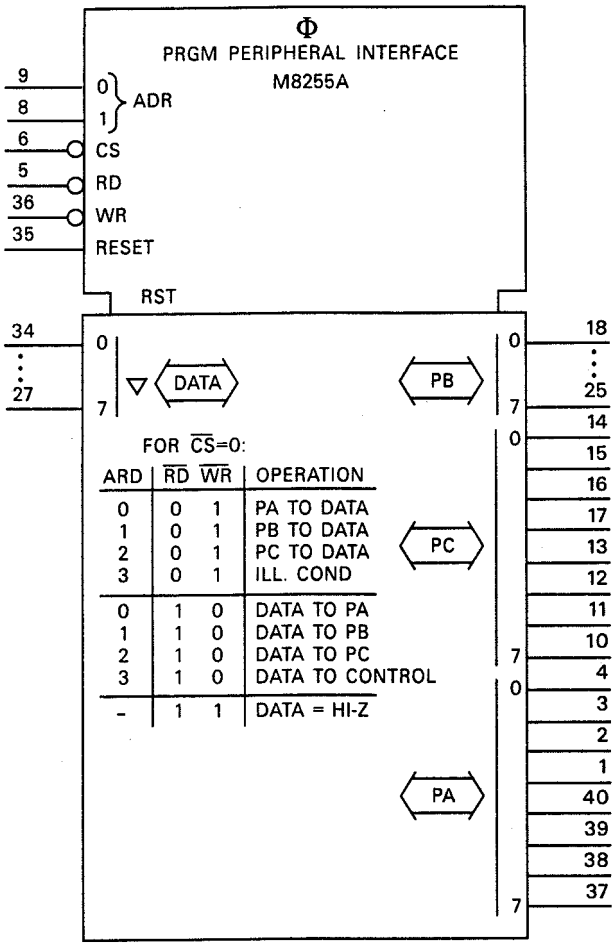
6.3 Examples of complex elements

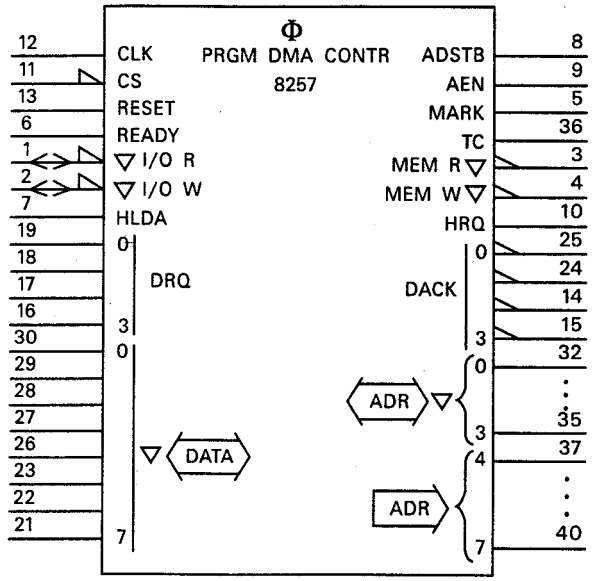
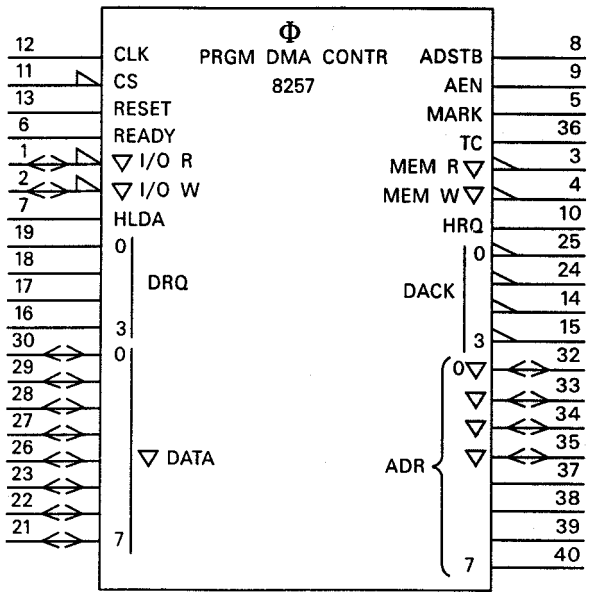
In the examples in this section, the following techniques have been used:

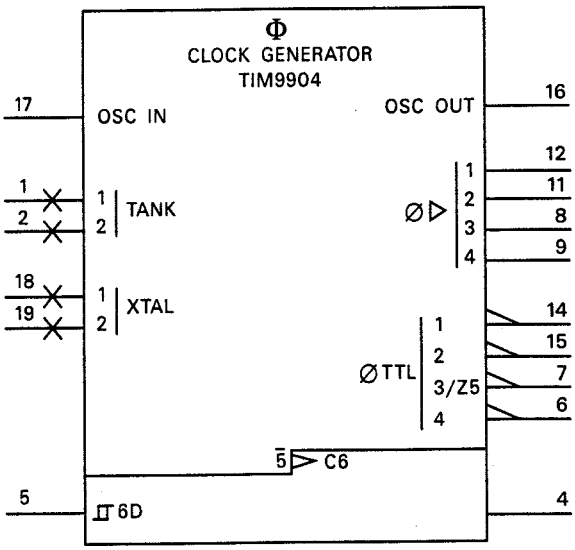
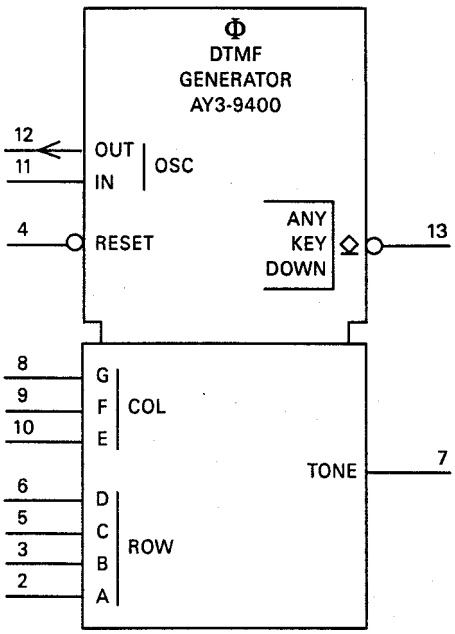
- The type number is shown within the symbol outline to satisfy the requirements of a reference to supporting documentation;
- The indication of the function and the type number are shown on separate lines.

If, on a diagram, other information leads to the specific data sheet or documentation from which the symbol was derived, the type number inside the outline may be omitted.

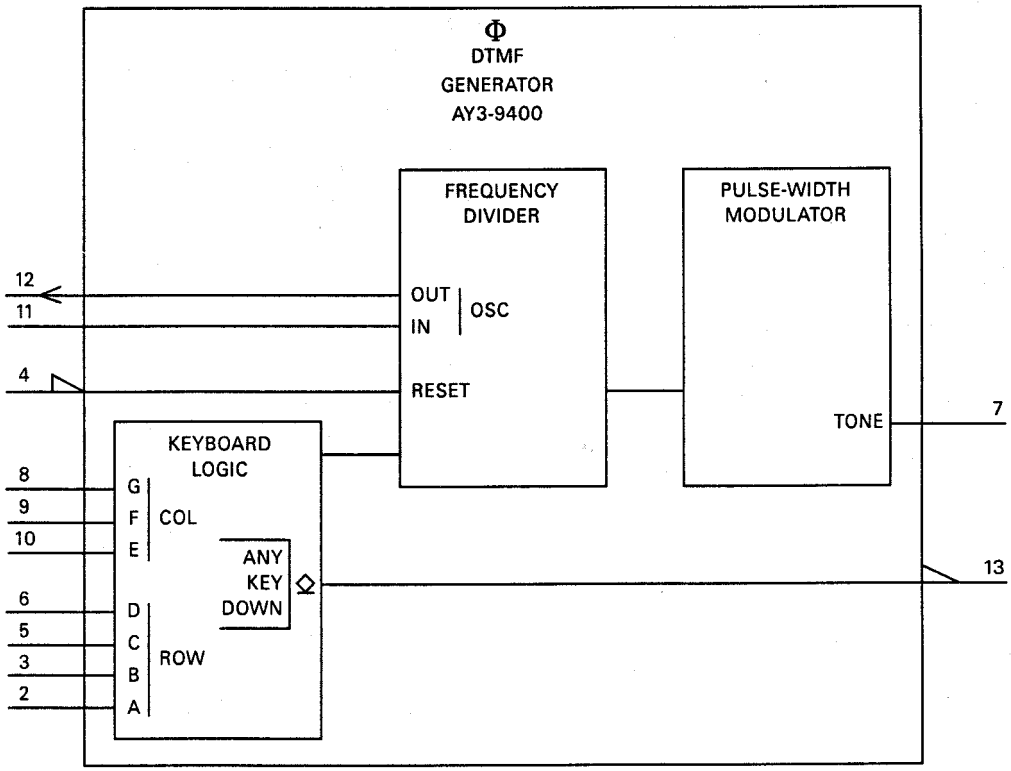
No	Symbol	Description
6.3-1		Microprocessor, 8-bit NOTES: 1 — In accordance with 6.1.2, ADR replaces A, ADR and DATA replaces AD, and MEM replaces M. 2 — The table is shown for the assistance of the reader but may be omitted. *Intel 8085*

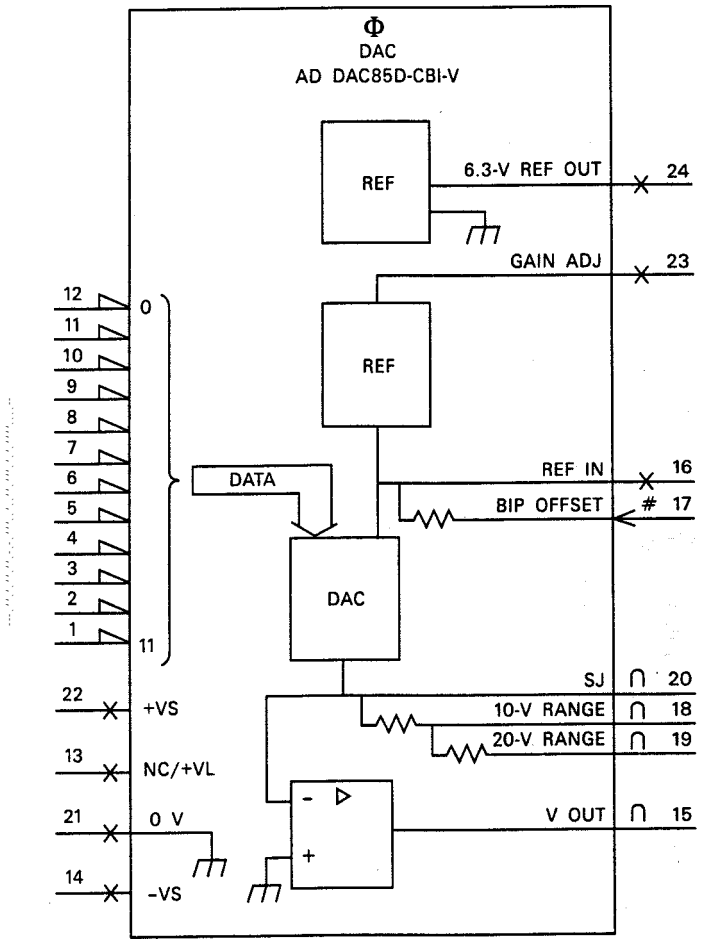
No	Symbol	Description
6.3-2		Programmable peripheral interface NOTE — In accordance with 6.1.2, ADR replaces A and DATA replaces D. See NOTE 2 with Symbol 6.3-1. *Intel M8255A*

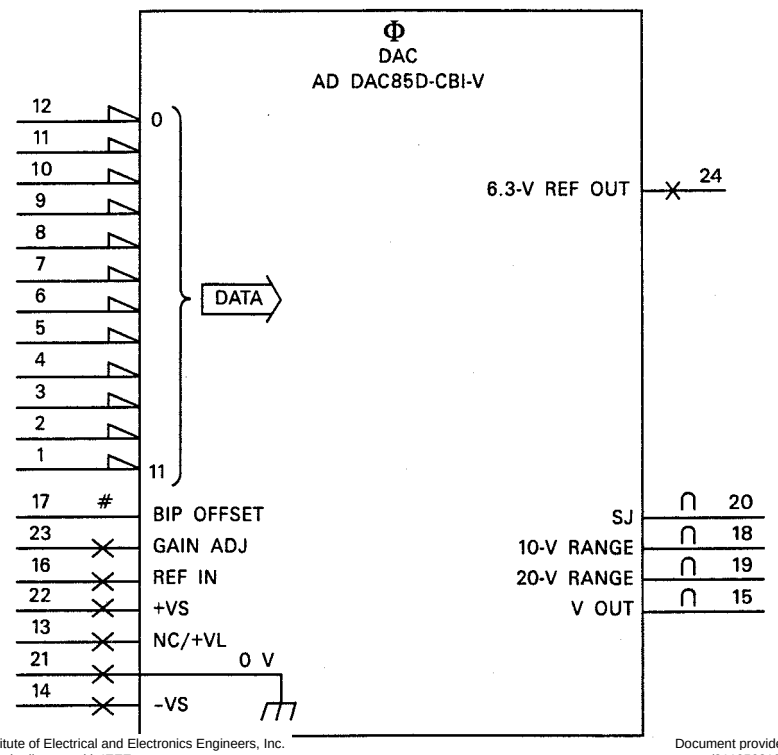
No	Symbol	Description
6.3-3		Programmable DMA controller NOTES: 1 — In accordance with 6.1.2, ADR replaces A and DATA replaces D. 2 — Two versions are shown; the first uses bus indicators, and the second shows signal direction on the lines outside the outline. *Intel 8257*
		

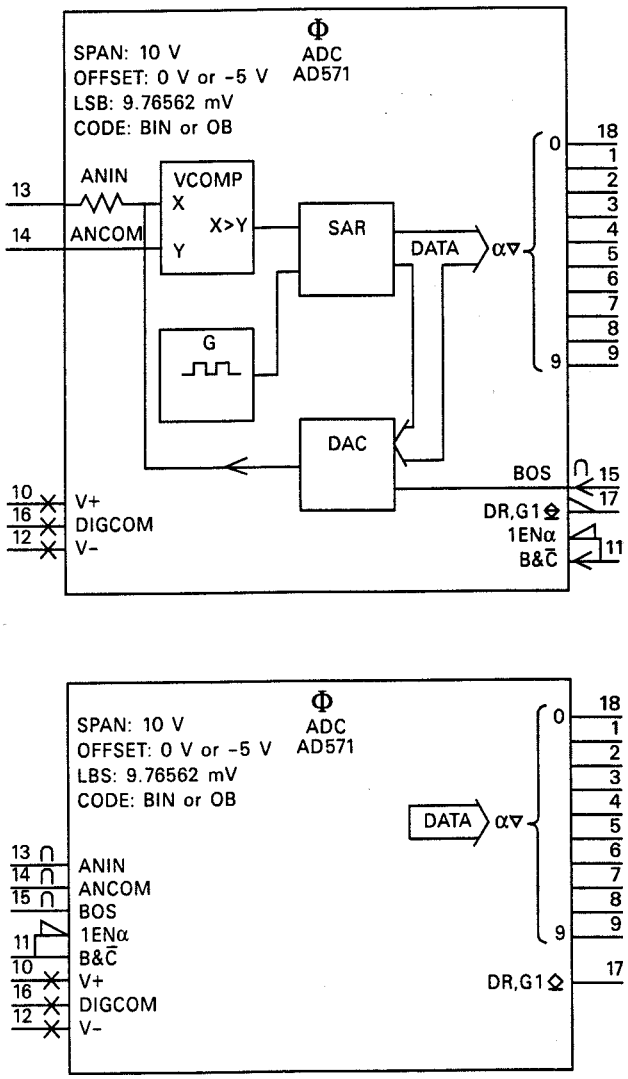
No	Symbol	Description
6.3-4		Clock generator/driver, four-phase NOTE — Symbol 5. 12-6 depicts the same device using only the rules of Sections 1 through 5. *Texas Instruments TIM9904 (formerly SN74LS362)*
6.3-5		Dual-tone multifrequency generator (generates 12 tone-pairs) NOTE — Two versions are shown; the second uses an internal diagram. *General Instruments AY3-9400*

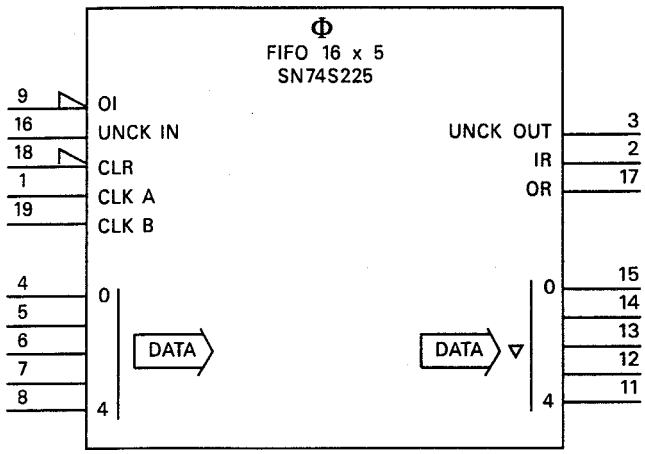
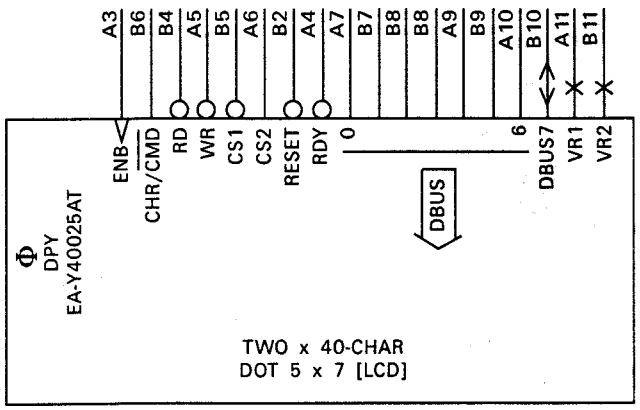
No	Symbol	Description



No	Symbol	Description
6.3-6		Digital-to-analog converter, 12-bit NOTES: 1 — Two versions are shown; the first uses an internal diagram. 2 — Because the logic inputs produce a number, use has been made of the bit-grouping symbol. Consequently the labeling of these inputs differs from that of the manufacturer. 3 — If no confusion is likely, ∩ at the inputs and outputs may be omitted. 4 — Arrowheads may be used on pins 18, 19, and 20, depending on the application. *Analog Devices AD DAC85D-CBI-V*



No	Symbol	Description
6.3-7		Analog-to-digital converter, 10-bit See NOTE 3 with Symbol 6.3-6. NOTES: 1 — Two versions are shown; the first uses an internal diagram. 2 — The layout of the internal diagram has been chosen such that the feedback function of the internal digital-to-analog converter is emphasized. 3 — Because the logic outputs represent a number, use has been made of the bit-grouping symbol. Consequently, the labeling of these outputs differs from that of the manufacturer. *Analog Devices AD571*

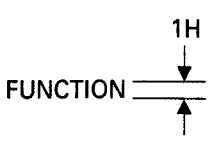
No	Symbol	Description
6.3-8		First-in, first-out memory, fall-through 16x5-bit NOTE — Symbol 5.14-9 depicts the same device using only the rules of Sections 1 through 5. *Texas Instruments SN74S225*
6.3-9		Display element, dot matrix, alphanumeric, with two 40-character lines *Epson EA-Y40025AT*

Annex A Recommended symbol proportions

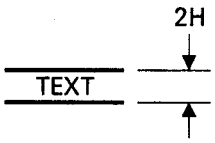
(Informative)

(The following Appendixes are not a part of IEEE Std 91a-1991, Supplement to IEEE Standard Graphic Symbols for Logic Functions, but are included for information only.)

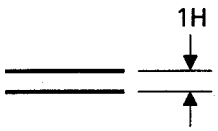
The symbol H in the dimensions that follow corresponds to Hc, the (center-line-to-center-line) lettering height defined in IEEE Std 991-1986. In IEC 617, this dimension is called the Modulus(M). All symbols are dimensioned accordingly. Greek, mathematical, and punctuation symbols not shown are considered as text. See IEEE Std 991-1986 for more detailed recommendations on sizes and spacing. All dimensions shown are center-line-to-center-line lettering height.



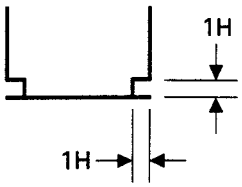
TEXT HEIGHT



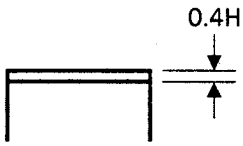
LINE TO LINE WITH TEXT



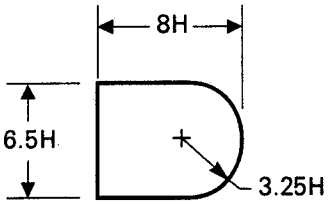
LINE TO LINE WITHOUT TEXT



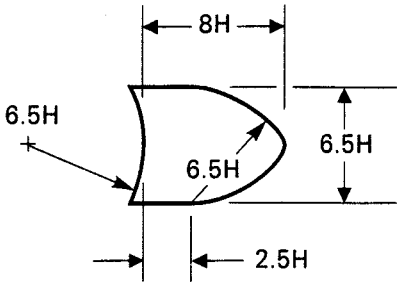
COMMON CONTROL



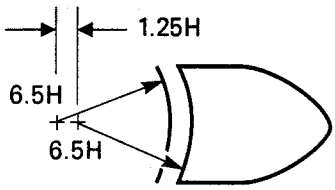
COMMON OUTPUT



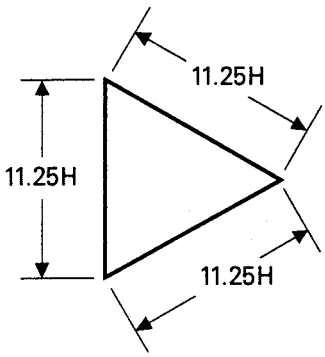
AND*



OR*

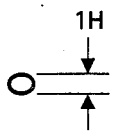


EXCLUSIVE OR* (SEE "OR" FOR ADDITIONAL DIMENSIONS)

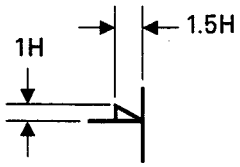


AMPLIFIER*

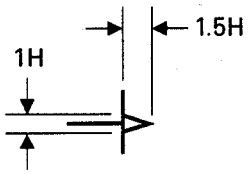
*Half-size versions of these may be used in conjunction with full-size versions of these and other symbols or text shown in this Appendix.



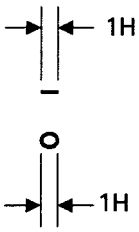
NEGATION



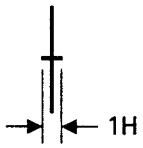
POLARITY
(SHOWN AT INPUT)



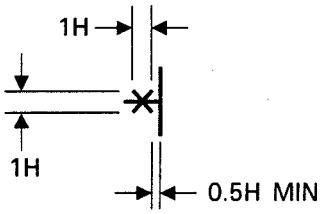
DYNAMIC INPUT
(SHOWN ON
SYMBOL EDGE)



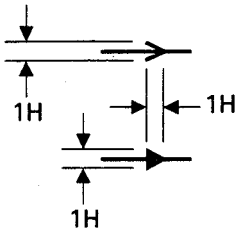
INTERNAL
CONNECTIONS



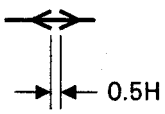
VIRTUAL
INPUT OR OUTPUT
(SHOWN ON
SYMBOL EDGE)



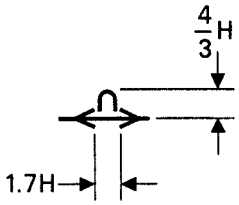
SUBSIDIARY
CONNECTION
(SYMBOL EDGE SHOWN)



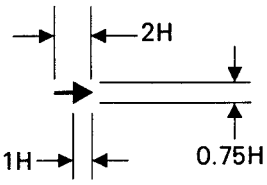
SIGNAL FLOW
(SHOWN ON LINE)



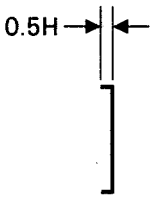
BIDIRECTIONAL
(SEE SIGNAL FLOW FOR
ADDITIONAL DIMENSIONS)



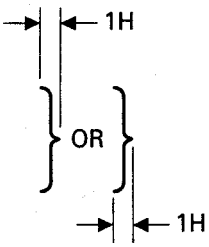
BIDIRECTIONAL ANALOG
(SEE SIGNAL FLOW FOR
ADDITIONAL DIMENSIONS)



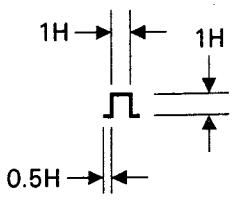
SHIFT INPUT



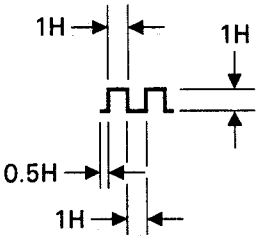
LINE GROUPING



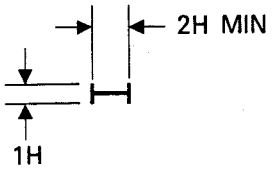
BIT GROUPING
(THE PRECISE SHAPE
IS UNIMPORTANT)



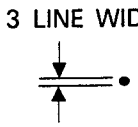
MONOSTABLE



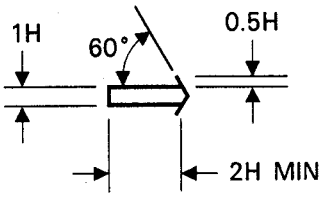
ASTABLE



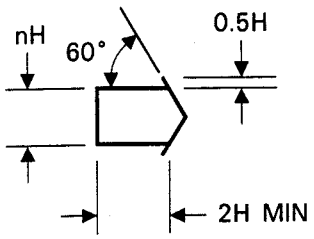
DELAY



CONNECTION DOT

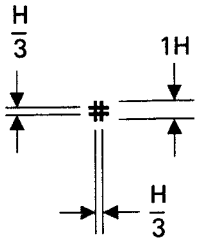


BUS
(WITHOUT TEXT)

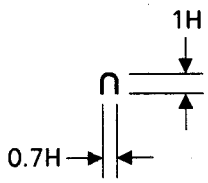


BUS
(WITH TEXT)

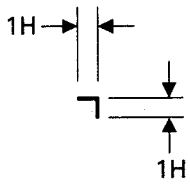
If the following characters are available in the standard character set used for other lettering, they may be used provided they reproduce legibly.



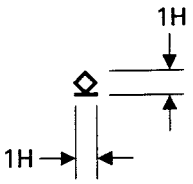
DIGITAL



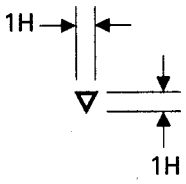
ANALOG



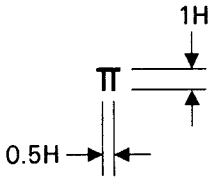
POSTPONED OUTPUT



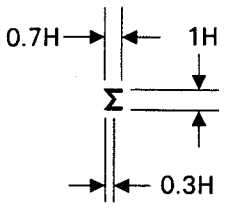
OPEN CIRCUIT
(L-TYPE SHOWN)



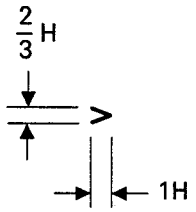
3-STATE



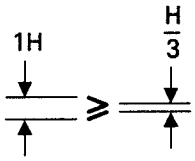
MULTIPLICATION
(UPPERCASE GREEK PI)



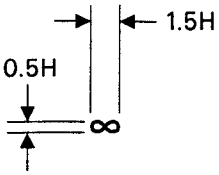
ADDITION
(UPPERCASE GREEK SIGMA)



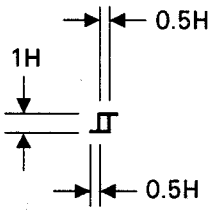
GREATER-THAN



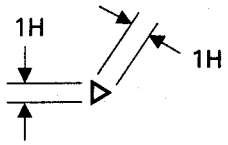
GREATER-THAN-OR-
EQUAL-TO



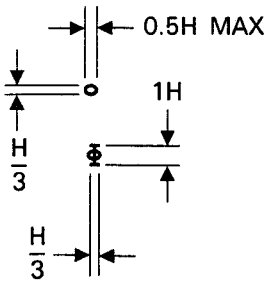
INFINITY



HYSTERESIS



AMPLIFIER
(EQUILATERAL TRIANGLE)

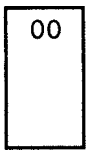
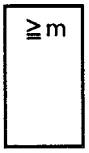
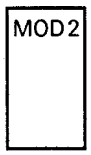
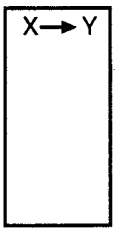


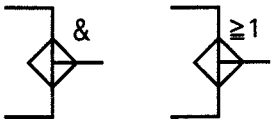
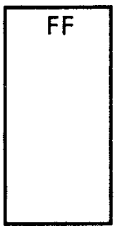
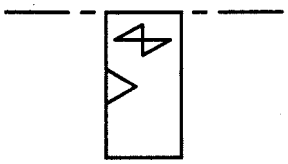
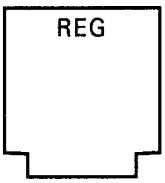
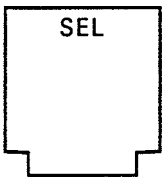
HIGHLY COMPLEX
FUNCTION
(UPPERCASE GREEK PHI)

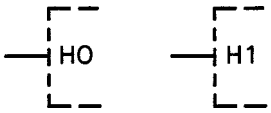
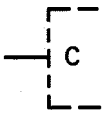
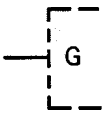
Annex B Revised or deleted symbols

(Informative)

This appendix lists symbols and rules from previous editions for this standard that have been deleted or revised in such a way as to make symbols developed according to those previous editions invalid.

Symbols formerly in ANSI Y32.14-1973/ IEEE Std 91-1973		Recommended symbols in IEEE Std 91-1984 and IEEE Std 91a-1991, if not otherwise specified
Revised:Symbol for OR		See Symbol 5.1-1
Revised:Symbol for Time Delay		See Section 5.8
Deleted:Symbol for Inhibiting Input		See Symbol 3.3-12 and 4.3.9-1
Revised:Symbol for Even Function		See Symbol 5.1-10
Revised:Symbol for Logic Threshold Element		See Symbol 5.1-5
Revised:Symbol for Odd Function		See Symbol 5.1-9
Revised:Symbol for Coder		See Symbol 5.4-1

Symbols formerly in ANSI Y32.14-1973/ IEEE Std 91-1973	Recommended symbols in IEEE Std 91-1984 and IEEE Std 91a-1991, if not otherwise specified
Revised:Symbols for Distributed Connections 	See Symbols 5.1-15 and 5.1-16
Revised:Symbol for Flip-Flop 	See Section 5.9
Deleted:Symbol for Nonreciprocal Directional Coupler 	
Revised:Symbol for Register 	See Sections 5.9 and 5.13
Revised:Symbol for Selector 	See Symbol 5.6-1
Revised:Symbol for Extender-Connection with Input and Output Lines: 	See Symbols 3.3-10 and 3.3-11
Revised:Symbol for Oscillator 	See Section 5.12

Symbols formerly in ANSI Y32.14-1973/ IEEE Std 91-1973		Recommended symbols in IEEE Std 91-1984 and IEEE Std 91a-1991, if not otherwise specified
Deleted:Symbols for Holding Input		See Symbols 3.3-16, 3.3-17, 4.3.2-1, 4.3.2-2, 4.3.7-1, and 4.3.7-2
Revised:Symbol for C Input		See Symbols 4.3.7-1 and 4.3.7-2
Revised:Symbol for G Input		See Symbols 4.3.2-1 and 4.3.2-2
Revised:Symbol for Amplifier		See Section 5.2
Revised: Section 1.5 Diagram Preparation		See IEEE Std 991
Revised:Section 1.3 Contiguous Block		See Section 2.3
Deleted: Section 1.4.2 Separated Block Symbols Array		
Deleted:Section 1.5 Bundling, Grouping		
Revised:Section 3.1.7 Dependency Notation		See Section 4
Revised: Note 4.3.1B Dependency Notation		See Section 4

Symbols formerly in IEEE Std 91-1984		Recommended symbols in IEEE Std 91a-1991 , if not otherwise specified
Revised: 5.7-1 Symbol for multiplier	π	See Symbol 5.7-1(Π)

Annex C Composite chart showing relationship of graphic symbols for logic diagrams from IEEE Std 91-1984, IEEE Std 91a-1991, and the superseded standards (Informative)

Column 1: Description of Logic Functions

Column 2: IEEE Std 91-1984
IEEE Std 91a-1991
2A: Rectangular-shape symbols
2B: Distinctive-shape symbols

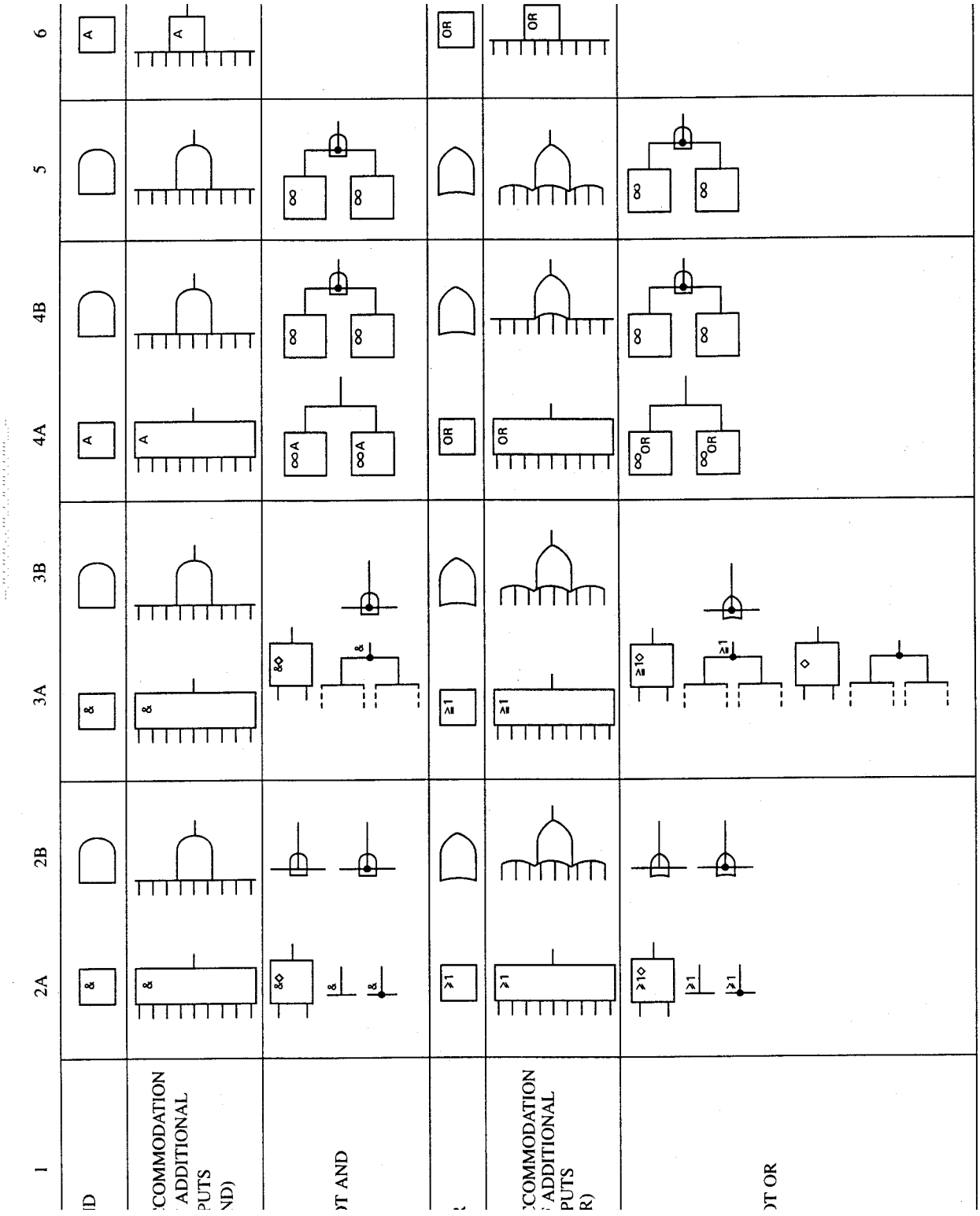
Column 3: ANSI Y32.14-1973
IEEE Std 91-1973
3A: Rectangular-shape symbols
3B: Distinctive-shape symbols

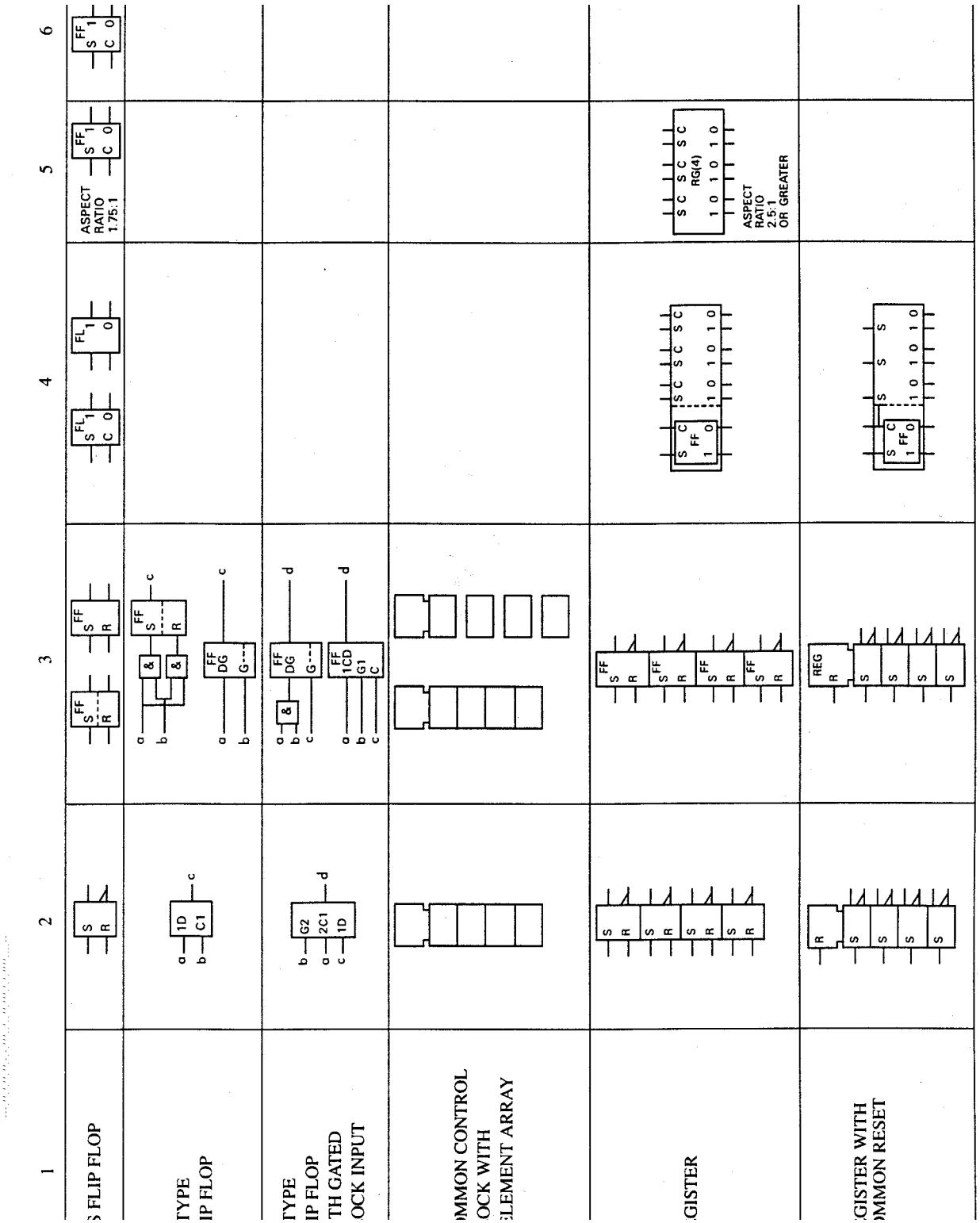
Column 4: ANSI Y32.14-1962
IEEE Std 91-1962
MIL-STD-806C (Navy)
4A: Uniform-shape symbols
4B: Distinctive-shape symbols

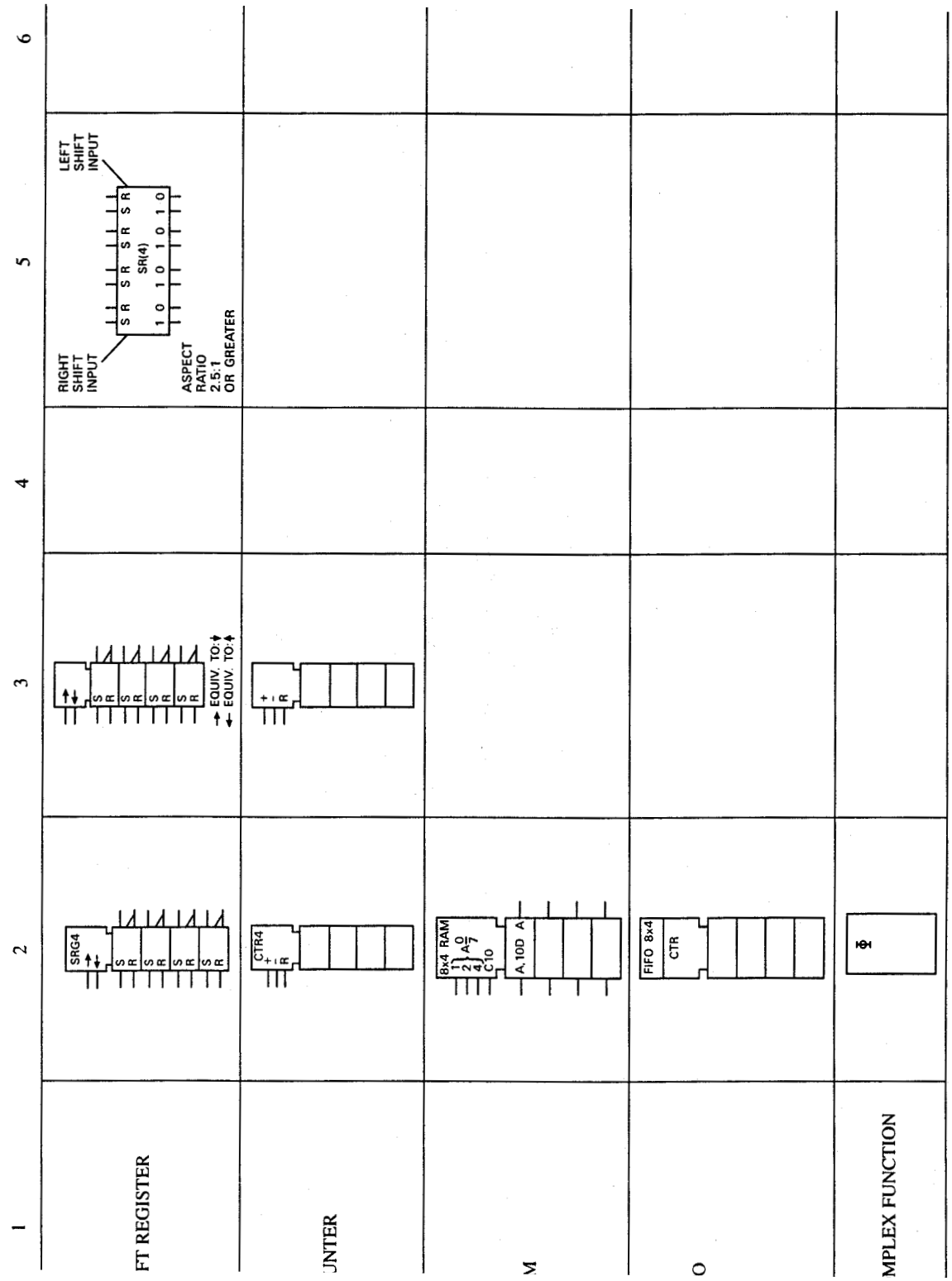
Column 5: MIL-STD 806B

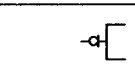
Column 6: Other industry standards

1	2	3	4	5	6
OGICAL INVERSION					
LECTRICAL INVERSION					
YNAMIC INPUT					
ON LOGIC CONNECTION					
NALOG CONNECTION					
EVERSE SIGNAL FLOW					
DIRECTIONAL GNAL FLOW					
PUT WITH HYSTERESIS					
EN CIRCUIT OUTPUTS					
SSIVE-PULLUP OUTPUT SSIVE-PULLDOWN OUTPUT					
STATE OUTPUT					
ECIALLY AMPLIFIED UTPUT					
RTUAL INPUT					
PUT CONTROLLED ELAY AT AN OUTPUT					
ULTIPLE LINES PROVIDING SINGLE LOGIC INPUT					
ROUPING OF LINES REPRE-ENTING A NUMERIC VALUE					
ROUPING OF LINES ITH SIMILAR NAMES	NAME				

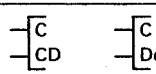
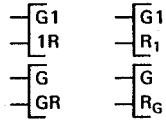






1	2	3	4	5	6
ANDARD LINE LABELS:					
ABLE	EN				
T WHEN 1: SET WHEN 0	D	D			
T (TOGGLE SET WHEN BOTH)	J K	J K	S C		
T SET	S R	S R		S C	S C
GGLE (COMPLEMENT)	T	T	T	T	T
IFT m POSITIONS	→m ←m	→m ←m			
UNT UP BY m	+m	+m			
UNT DOWN BY m	-m	-m			
ERY INPUT OF CONTENT- DRESSABLE MEMORY	?				
ATCH OUTPUT OF CONTENT- DRESSABLE MEMORY	!				
T CONTENT EQUAL m	CT=m				
NTENT OUTPUT	CT=m CT<m CT>m etc.				
PUT MUST BE A "1"	"1"				
UTPUT ALWAYS A "1"	"1"				

1	2	3	4	5	6
ARITHMETIC OPERAND	P Q				
GREATER THAN INPUT	>				
LESS THAN INPUT	<				
EQUAL INPUT	=				
GREATER THAN OUTPUT	P>Q				
LESS THAN OUTPUT	P<Q				
EQUAL OUTPUT	P=Q				
BORROW INPUT	BI				
BORROW OUTPUT	BO				
BORROW GENERATE	BG				
BORROW PROPAGATE	BP				
CARRY INPUT	CI				
CARY OUTPUT	CO				
CARRY GENERATE	CG				
CARRY PROPAGATE	CP				

1	2	3	4	5	6
PENDENCY NOTATION LINE LABELS: (note "m" is replaced by a number)					
DRESS	Am				
NTROL	Cm	C			
LD		H_0 (HOLD 0-STATE) H_1 (HOLD 1-STATE)			
ABLE	ENm				
D (GATE)	Gm	Gm G			
DE	Mm				
GATE (XOR)	Nm				
SET	Rm				
	Sm				
	Vm				
ANSMISSION	Xm				
ERCONNECTION	Zm				
NPUT FECTED BY A NTROL INPUT					
NPUT FECTED BY AN D INPUT					

Annex D Integrated circuits used as examples commercial part numbers versus symbol numbers

(Informative)

(This Appendix is not part of IEEE Std 91a-1991, Supplement to IEEE Standard Graphic Symbols for Logic Functions, but is included for information only.)

The committee members who prepared this standard had a frequent need to refer to an example for which they knew the commercial part number, but not the symbol number. This Appendix has been provided to assist users of the standard with this same need.

The following points should be understood. The symbols given may represent only a part of the integrated circuit. If so, that is noted next to the symbol. The symbols were drawn to illustrate certain points. They were not necessarily intended to be the *best* symbol for the device in question. In several cases, more than one symbol is given for a particular device.

Commercial part number	Symbol	Commercial part number	Symbol
3625	5.14-2	F100170	5.6-8
3625	5.14-2	F100181	5.7-14
(simplified)		HDSP2000	5.15-8
5082-7340	5.15-5	HDSP3603	5.15-3
5082-7433	5.15-6	HDSP5607	5.15-4
8085	6.3-1		
8212	5.9-13	HDSP6504	5.15-7
8226	5.2-6	HLMP2600	5.15-2
8257	6.3-3	M8255A	6.3-2
8286	5.2-8	MC1222	5.9-11
AD571	6.3-7	MC10121	5.1-21
AD DAC85D-CBI-V	6.3-6	MC10125	5.5-3
Am26S10	5.3-3	MC10131	5.9-10
AY3-9400	6.3-5	MC10163	5.1-29
CD4016B	5.2-9	MC14519	5.6-5
CD4020	5.13-11	MC14519	5.6-6
CD4020	5.13-12	MC14529B	5.6-9
(simplified)		MM4057	5.13-3
CD4026A	5.13-15	SN7403	5.1-20
CD4035A	05.13-5	SN7406	5.2-1
CD4053B	5.2-11	SN7410	5.1-17

Commercial part number	Symbol	Commercial part number	Symbol
CD4502B	5.2-5		
DM7160	5.7-10	SN74LS14	5.3-2
DM76L24	5.7-12	SN7427	5.1-18
EA-Y40025AT	6.3-8	SN7437	5.2-2
F100102	5.1-24	SN7442	5.4-3
F100107	5.1-25	SN7443	5.4-2A
SN7444	5.4-2	SN74LS221	5.11-4
SN74LS47	5.4-7	SN74LS224	5.14-8
SN7450	5.1-22	SN74LS225	5.14-9
SN74L51	5.1-19	SN74ALS229	5.14-10
SN74LS57	5.13-14	SN74S240	5.2-4
SN7460	5.1-23	SN74279	5.9-6
SN74L71	5.9-9	SN74280	5.1-28
SN7474	5.9-7	SN74283	5.7-5
SN7474	5.9-8	SN74283	5.7-6
(complementary form)		SN74284	5.7-9
SN7475	5.9-2		
		SN74285	5.7-8
SN7480	5.7-4	SN74S288	5.4-9
SN7485	5.7-11	(TBP18S030)	
SN74H87	5.1-31	SN74298	5.9-12
SN7488	5.14-3	SN74LS323	5.13-8
SN7488	5.14-3	SN74LS323	5.13-9
(simplified)		(incomplete)	
SN7491	5.13-2	SN74LS323	5.13-10
SN74107	5.9-4	(incomplete)	
SN74LS107	5.9-3	SN74LS362	5.12-6
SN74111	5.9-5	(TIM9904)	
SN74LS123	5.11-3	SN74LS362	6.3-4
		(TIM9904)	
SN74LS124	5.12-7	SN74S484	5.4-8
SN74132	5.3-3	SN74490	5.13-10
SN74S135	5.1-26	SN74491	5.13-2
SN74S135	5.1-27	SN74LS0625	5.12-8
SN74LS138	5.4-4	SN75107A	5.2-7

Commercial part number	Symbol	Commercial part number	Symbol
SN74LS138	5.67	SN75127	5.2-7A
SN74147	5.4-5	SN75152	5.2-7B
SN74148	5.4-6		
SN74151	5.6-4	SN75365	5.5-2
SN74LS160	5.13-13	TBP18S030	5.4-9
		(SN74S288)	
SN74164	5.13-6	TIM9904	5.12-6
SN74165	5.13-7	(SN74LS362)	
SN74170	5.14-5	TIM9904	6.3-4
SN74180	5.1-30	(SN74LS362)	
SN74181	5.7-13	TMS4116	5.14-6
SN74182	5.7-7		
SN74185	5.4-10		
SN74S189	5.14-4		
SN74191	5.13-17		
SN74LS194	5.13-4		